Local Halting Criteria for Stochastic Diffusion Search Using Nature-inspired Quorum Sensing

Andrew Owen Martin

andrew@aomartin.co.uk

Thesis submitted for the degree of Doctor of Philosophy

of the

University of London

Department of Computing

Goldsmiths College

April 2020

Declaration of Authorship

I, Andrew Owen Martin, hereby declare that this thesis and the work presented in it is entirely my own. Where I have consulted the work of others, this is always clearly stated.

Signed: Date: ____________________

Abstract

Stochastic Diffusion Search (SDS) is a Swarm Intelligence algorithm in which a population of homogeneous agents locate a globally optimal solution in a search space through repeated iteration of partial evaluation and communication of hypotheses. In this work a variant of SDS, Quorum Sensing SDS (QSSDS), is developed in which agents employ only local knowledge to determine whether the swarm has successfully converged on a solution of sufficient quality, and should therefore halt. It is demonstrated that this criterion performs at least as well as SDS in locating the optimal solution in the search space, and that the parameters of Quorum Sensing SDS may be tuned to optimise behaviour towards a fast decision or a high quality solution. Additionally it is shown that Quorum Sensing SDS can be used as a model of distributed decision making and hence makes testable predictions about the capacities and abilities of swarms in nature.

Acknowledgements

First and foremost, thanks must go to Prof. John Mark Bishop, my supervisor and friend who advocated for me to be accepted on their Cognitive Computing MSc in 2009 and gave me the courage to subsequently start a PhD. His guidance was clear and invaluable, but always left me the freedom which makes this work entirely my own. Thanks also to Drs. John Howroyd and Mohammad Majid al-Rifaie who provided both friendship and critical advice.

Special thanks go to my parents Kay and Ron, and my brother Richard who supported me in every conceivable way, consistently and substantially, and occasionally politely enquired as to when a finished thesis may finally emerge.

Not unspecial thanks goes to the boys, Bruno, Chad, and Graham, for their regular company, which kept me sane.

Finally, I would like to thank my wife Emilia, whose love gave me the courage to finish my thesis, who gave me joy outside of the academic world, and who was there for me every day, especially as I became increasingly absent as deadlines loomed.

Contents

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Aim

This work aims to identify swarm intelligence as an approach to Artificial Intelligence (AI) which provides a complimentary direction of research to the current established methods. The historical development of artificial intelligence is described and the strengths and limitations of the two main approaches that dominate its modern form are examined. This is in order to show that swarm intelligence can be considered to be a third way in which to explore artificial intelligence which shares some significant features with the two other approaches and yet is distinct. By examining the many published variants of an existing swarm intelligence algorithm, Stochastic Diffusion Search (SDS), we will aim to show that a diverse range of behaviours can emerge from a swarm as a result of changes in the behaviour of the individuals and that each variant introduces disadvantages as well as advantages. We develop a new variant, Reducing SDS, to demonstrate its ability to propagate halting behaviour within a swarm rather than relying on an external process and to potentially provide insight into distributed decision making observed in the natural swarms. Two further variants are introduced, Running-mean SDS, and Quorum sensing SDS. Running-mean SDS demonstrates capacity for robust and flexible distributed decision making using an exact method. Quorum sensing SDS demonstrates similar capacities but employs a simple stochastic individual behaviour. Both variants have parameters, the effect of which is investigated, showing that they can be selected to optimise performance for a desired outcome, and that the algorithm is stable over a wide range of parameters.

1.2 Structure of the thesis

In identifying the value of swarm intelligence as a method of investigation some key events in the history of automatic performance of intelligent behaviour are examined. Starting with the fully defined clepsydra of the ancient world we see artefacts that can produce a single complex behaviour by virtue of their construction. Eventually machines were developed which had the ability to be reconfigured to perform variations on a single task. Given as examples are L'Ecrivain of 18th century Switzerland, which could ink a message desired by the user, and the Jacquard Loom of 19th century France, which could weave patterns of great complexity as determined by a sequence of punched cards.

The next development that is significant for this investigation is when machines could be configured to perform distinctly different tasks as well as variations of those tasks. The Analytical Engine of Charles Babbage is an early example of a machine which used one medium, punched cards in this case, to procedurally define its behaviour, as well as specifics of that behaviour. Where L'Ecrivain could produce many messages and the Jacquard Loom could produce many patterns, the Analytical Engine in principle could both produce patterns and produce messages. The notion of procedurally defined behaviours was most significantly developed when Turing in 1936 defined a formalism for mechanical procedures, from this formalism Turing proposed what became known as the Universal Turing Machine in which definitions of mechanical procedures were represented in exactly the same medium as the input and output, symbols printed on a tape. The significance of the Universal Turing Machine was that a single machine could be built which would be able to perform any and all mechanical procedures that were possible to define. In practice many mechanical procedures exceed practical limits such as requiring too much time or too much 'tape', but as Universal Turing Machines developed in the form of electronic computers the limits were pushed ever back.

Turing's subsequent work included working towards the invention of the electronic computer which would be a practical example of the Universal Turing Machine, and investigations into whether any of the mechanical procedures that could be implemented on such a machine would produce human-level intelligence. Significantly Turing included emphasis on how achieving this aim would be detected, and introduced the notion of designing machines to perform quintessentially human traits such as game playing. By 1956 a project was proposed based on the notion that all aspects of intelligence could be described as a mechanical procedure that could be implemented in a machine. The subject of the project was eventually named "artificial intelligence" and the subsequent investigations thereof can broadly be separated into two categories, that of making a mind vs modelling a brain, both of which have their own advantages and successes and both of which have been philosophically critiqued as being at best practically challenging and at worst logically impossible.

While the artificial intelligence project progresses in the presence of this dichotomy a related project, swarm intelligence, aims to discover and take advantage of the mechanisms by which a group of individuals may act such that from their combined behaviour a global activity emerges which appears to be more complex or intelligent than could be attributed to any individual. Swarm intelligence can also be seen as a third way between the approaches of artificial intelligence, in one sense each agent is defined explicitly, analogous to the symbolic logic procedures of 'making a mind' and the result of the system is produced by interactions between the elements, analogous to the connectionist architecture of 'modelling the brain'. Furthermore swarm intelligence has a feature that is not strongly present in either of the artificial intelligence approaches, that the evolution of the global state of the system is determined by the global state of the system itself, rather than any predetermined logic, or architecture.

We examine some algorithms which produce behaviour best described as a global state behaving intelligently emerging from interactions of relatively simple individuals. These algorithms are interesting for two reasons, they may be feasible ways of solving existing practical problems, and as they are characterised neither by making the mind or modelling the brain they may offer insight as to a way to progress in the artificial intelligence project against which the existing philosophical criticism does not hold. We also look at some observations of similar behaviour in natural swarms, this also has a twofold purpose, observations of natural behaviour may offer insights into the design of existing swarm intelligence algorithms and unanswered questions about natural behaviour may be explored by comparison with analogous behaviour in swarm intelligence algorithms, in this way we may even be able to explain natural behavioural phenomena by explaining the analogous phenomena in swarm intelligence algorithms.

We pay particular attention to SDS as a good example of a swarm intelligence algorithm which: employs a swarm of simple individuals; can be analysed mathematically to predict its performance over a wide range of situations; produces useful and distinct behaviours as a result of small changes in the manner in which members of the swarm interact. We list the mathematical analysis of SDS which are used to evaluate and compare its many variants. We develop a syntactic model to describe variants of SDS and hence highlight their differences. We look at a number of variants that may be described by this model and hence demonstrate the range of emergent behaviours that are achievable through changes in individual behaviour, even though the changes themselves may initially appear to be minor.

We introduce the variant Reducing SDS inspired by a question of how certain species of ants detect quorum in their collective decision making processes and perform an experiment to show that behaviour analogous to that observed in ants can be performed by SDS.

Building on the successful proof of concept, a further variant is developed Quorum sensing SDS which combines the novel mechanism of Reducing SDS with features of the previously examined variants of SDS, we see that the new variant is a robust and effective search algorithm.

Finally this work will be critically examined, the contributions and shortcomings are listed and areas of potential further work are identified.

1.3 Contributions

The expected contributions of this work are a summary of the key moments in the development of AI, a study of the strengths and weaknesses of the variants of SDS, a demonstration of a mechanism whereby individuals in a swarm may infer information about the global state of the swarm, and a practical algorithm for effective searching of a wide range of search spaces.

1.4 Research questions

Given the success of algorithms which perform well defined actions over data structures and of artificial neural networks which produce useful behaviour as a result of a learning process, identifying the value in studying the abilities of swarms which are neither a completely controlled process or a learning structure.

The useful behaviours which can emerge from a swarm as a result of the actions of the individuals, and how do these behaviours compare with each other.

Whether individuals in a swarm intelligence algorithm can access information about the global state of the swarm through local stochastic processes, and if these behaviours are analogous to those observed of natural swarms.

That these observations may be combined to produce a practical and robust swarm intelligence algorithm.

Chapter 2

History of AI

It is only a question of cards and time.

— Major-General H. P. Babbage (1888) [3]

2.1 Historical background

In this section we chart development of diversity in a machine's behaviour, from Clepsydra which can perform a single behaviour, through machines which include configurable mechanisms to determine a specific routine, to computers which are hugely diverse due to defining algorithms in the same manner as their input data. Though people have been questioning the potential of machine behaviour at least as early as 1842 when Ada Countess of Lovelace wrote "[the Analytical Engine] might compose elaborate and scientific pieces of music of any degree of complexity or extent", the modern study of achieving intelligent behaviour in machines was greatly influenced by Alan Turing's pioneering work following his definition of a model of computing, and subsequently named Artificial Intelligence in 1955 [50].

2.1.1 Machine behaviour

2.1.1.1 Ancient automata — Clepsydra

In Ancient Greece engineering was sufficiently advanced to make artefacts, such as water clocks and automata, that would exhibit a single pre-determined behaviour. The water clock or clepsydra was originally a simple container with an aperture through which water flowed, emptying the container over a known time span. The Greek engineer Ctesibius is known for adding water wheels and floats to drive a further mechanism that would indicate the passage of time [43] (Figure 2.1). Though simple, this behaviour was of great practical use, enabling clepsydra to be used as alarm clocks or stop watches and was only made obsolete by the 17th Century invention of the pendulum clock.

Instilling artefacts with a certain specific behaviour was therefore within the reach of engineers of antiquity. A greater challenge was to instil a single artefact with a range of complex behaviours.

2.1.1.2 Hero's mechanical theatre

A more complex example, given by Hero of Alexandria in a work dated around the second half of the first century AD, posited a mechanism of an automatic theatre (Figure 2.2), itself first described by Philon of Byzantium in the late third century BC. The resulting display showed five scenes with animated characters, scrolling backdrops, timed sound effects and main theatre doors which opened and closed to hide the scene transitions [48, 4]. The energy for the actions was provided by a series of cords attached to a weight which sat in a cylinder of sand that was slowly emptying. Each cord was attached to a single mechanism with an appropriate length such that the action occurred at the intended time. The principle could be used to display any number of scenes of arbitrary complexity, as long as a cord-driven mechanism could be designed for each action.

Hero's theatre could be considered to be a single artefact that exhibited a rich range of behaviours but each individual action required its own separate mechanism. Even two instances of similar action, such as in first scene where two shipbuilders perform a hammering and sawing action, required two separate mechanisms. The range of actions of a single artefact was therefore tightly limited by considerations of space and complexity.

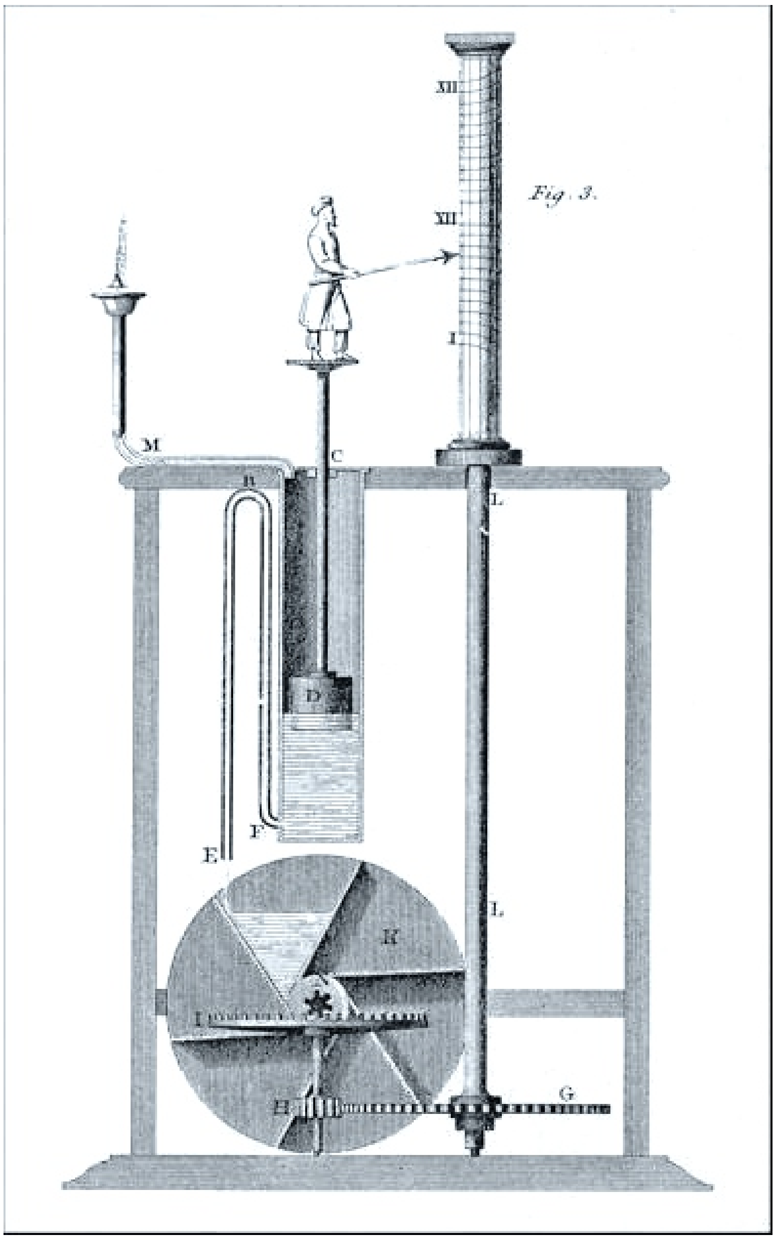


Figure 2.1: An early 19th-century illustration of Ctesibius' (285–222 BC) clepsydra from the 3rd century BCE. The hour indicator ascends as water flows in. Also, a series of gears rotate a cylinder to correspond to the temporal hours. Image in the public domain, via Wikimedia Commons `https://commons.wikimedia.org/wiki/File:Clepsydra-Diagram-Fancy.jpeg`

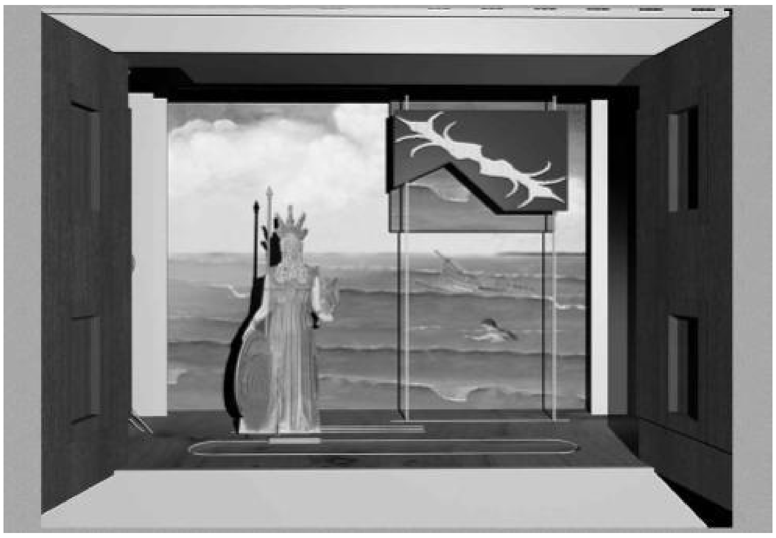


Figure 2.2: The fifth and final scene. The figure of Athena has been raised up from where it lay on the floor, to then move on an elliptical track around the stage. The backcloth shows Ajax in the sea. At the same moment when the flat panel, depicting the thunderbolt, falls from above, and disappears into a slot cut in the stage floor, a small second backcloth (painted so that it exactly matches a portion of the seascape) suddenly descends to cover the figure of Ajax, as if he has vanished into the sea. [4]

To stage a different production, the theatre would have to be significantly modified in a way that required the skill of the original builder, some mechanisms may be reusable but a change in behaviour would effectively necessitate the construction of a new theatre. The theatre therefore, while able to exhibit a range of actions, could only ever perform them in a single sequence.

2.1.1.3 L'Ecrivain

By the 18th century, clockwork engineering had advanced such that automata were built whose behaviour was configurable by a user. Pierre Jaquet-Droz, a Swiss watchmaker is known for a number of exquisite works, of which L'Ecrivain (The Writer) is a sophisticated example [90, p.23].

Inside the automaton is a disc containing forty notches, these notches can each contain one of a set of tabs. Each tab is engraved with a letter of the alphabet so a human operator can tell which letter of the alphabet this tab will induce the automaton to write. All tabs engraved with the same letter are of a specific height, such that when they are rotated to the top of the disc, they offset an arm in the mechanism by a specific amount. The offset of the arm causes an internal cylinder to raise or lower such that the section encoding the chosen letter is aligned with the mechanism for driving the automaton's arm. The automaton's arm is driven by three internal levers, each one causing movement in one of the three spatial dimensions, these levers are bought into contact with the appropriate section of the internal cylinder, which is composed of a series of cams, such that each lever will be in contact with a cam which produces the appropriate movement as the cylinder rotates about its axis. As with Hero's theatre, the number of available actions was strictly limited, one letter could be written for each set of three cams in the cylinder, but in L'Ecrivain they could be arranged into as many unique sequences as there were unique combinations of tabs on the disc, without requiring any fundamental modification of the system.

2.1.1.4 The Jacquard Loom

A critical development in the determination of machine behaviour was first demonstrated in 1801. Designed by Joseph Marie Jacquard, the Jacquard loom responded to patterns of holes punched into cards that passed through a device which would convert the pattern into an appropriate internal configuration. In weaving, a cloth can be considered as composed by a series of parallel "warp" threads interlaced with a series of perpendicular "weft" threads. In plain cloth the weft lies on alternate sides of each warp thread; patterned cloth can be produced by modifying which sides of the warp threads each subsequent weft thread lies. In the Jacquard loom, each card had a space corresponding to each warp thread, the presence or absence of a hole corresponded to the position of a hook which each held a single weft thread. The position of the hook would determine which side of the warp thread the weft thread would be placed [71]. Each card therefore fully described a line in a woven pattern.

2.1.1.5 Comparison of mechanical automata

A tab in L'Ecrivain would correspond to one of the character patterns encoded on its central cylinder but a "punched card" in the Jacquard loom would itself define a single line of a woven pattern. The critical difference is that each line of the pattern was symbolically represented on the chain of cards and that the Jacquard loom was therefore capable of weaving as many patterns as there were unique representations on a series of cards of

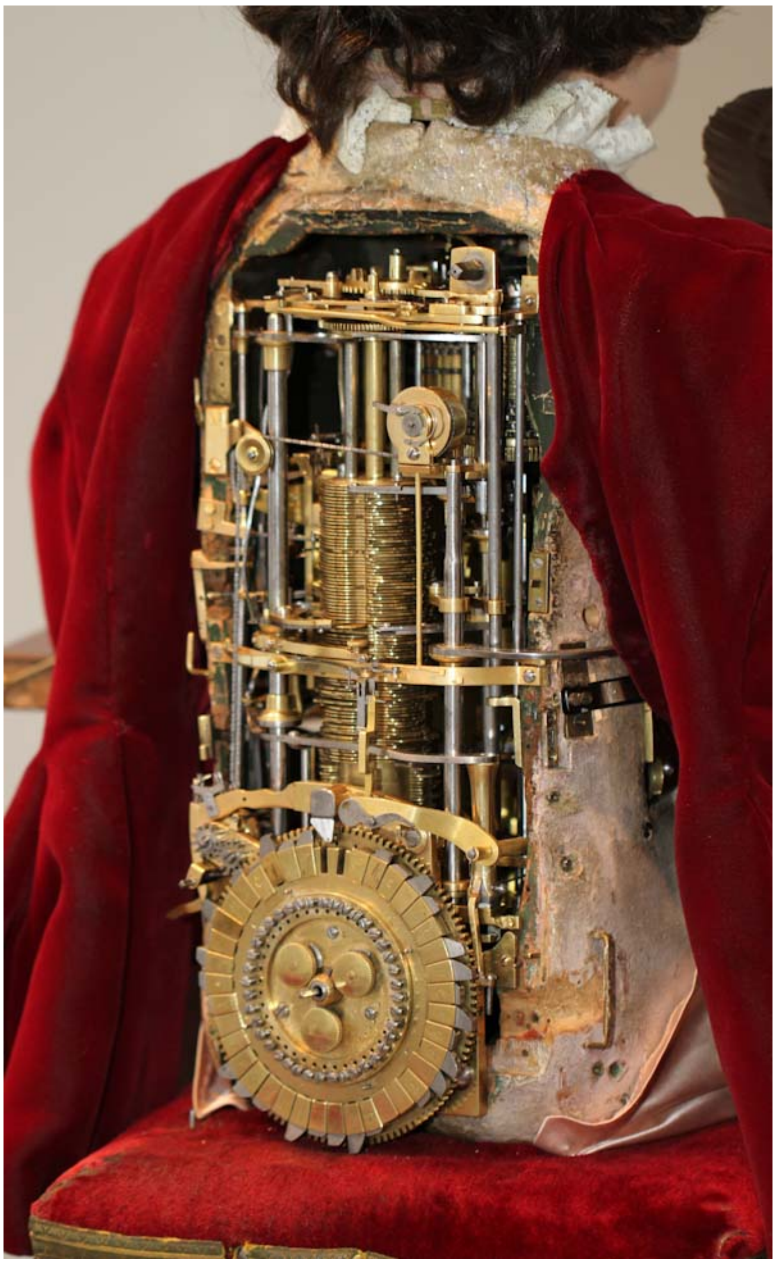


Figure 2.3: A view of the rear of L'Ecrivain, showing the mechanism and the configuration wheel holding 38 of 40 tabs. Used with permission of Ariel Adams, from http://www.ablogtowatch.com/jaquet-droz-the-writer-automata-awesome-antique-android/

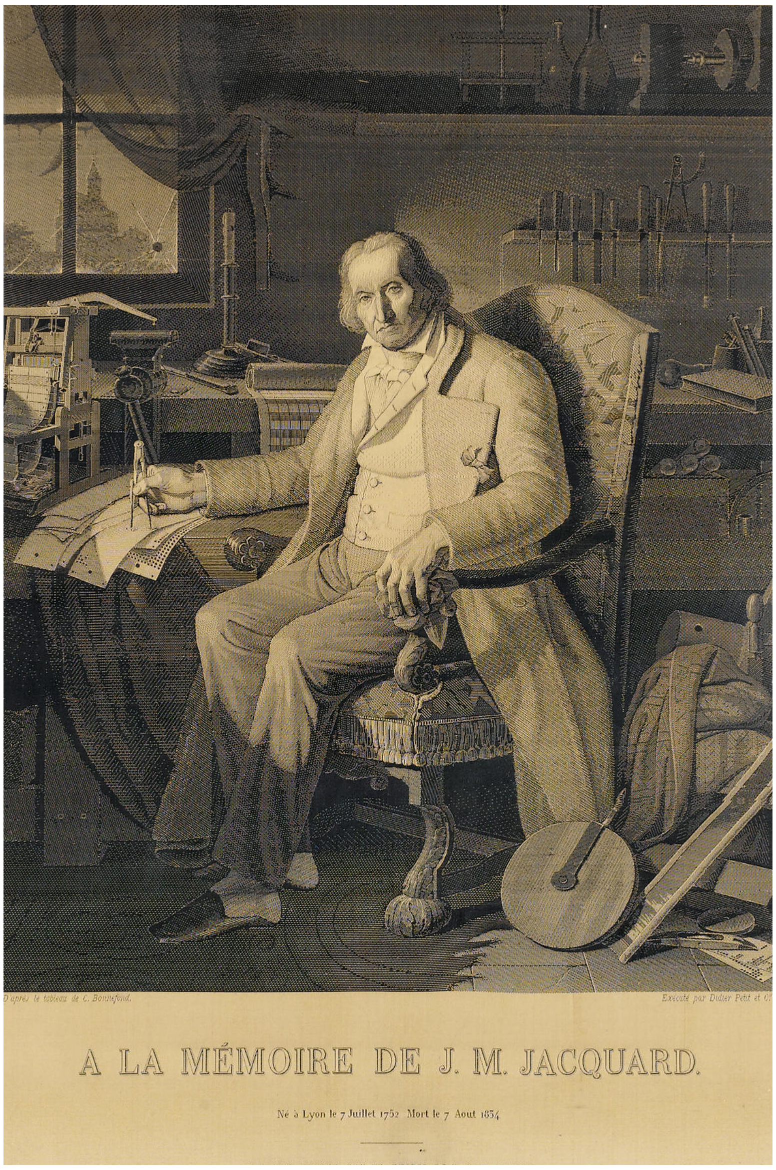

D'après le tableau de C. Bonnefond.
Exécuté par Didier Petit et Cie
A LA MÉMOIRE DE J. M. JACQUARD.
Né à Lyon le 7 Juillet 1752 Mort le 7 Aout 1834

Figure 2.4: This portrait of Jacquard was woven in silk on a Jacquard loom and required 24,000 punched cards to create (1839). Charles Babbage owned one of these portraits; it inspired him in using perforated cards in his analytical engine [2]. It is in the collection of the Science Museum in London, England. Image in the public domain, via Wikimedia Commons `https://commons.wikimedia.org/wiki/File:A_la_m%C3%A9moire_de_J.M._Jacquard.jpg`

arbitrary length.

The range and richness of the behaviours of artefacts clearly grew in superiority since the invention of Greek clepsydra but they were equivalent in their inability to react to events in the world. L'Ecrivain would continue to write its predetermined message just as Hero's Theatre's show must go on regardless of external influence, except in the case where the conditions for the pre-determined behaviour were fatally disrupted.

2.1.2 The question of intelligent behaviour selection

Investigations in to the determination of appropriate behaviours is also traceable as least as far into the past as Ancient Greece. Both Plato and Aristotle wrote on the subject.

2.1.2.1 Plato

Plato, in the Republic describes the soul as being comprised of three parts, the smallest and most critical part for a "just soul" was the thinking part — the *logisticon*. In the allegory of the cave, prisoners see shadows of objects, which are recognised by comparing their shape to ideal "forms" [45]. The description of a thinking part of the soul that makes just decisions guided by recollection of representations of worldly objects can be seen as an early description of intelligent action selection.

2.1.2.2 Aristotle

Aristotle, questioning "how the soul moves the body, and what is the origin of movement in a living creature" posited that the heart was the seat of the rational soul. At the heart, the sensations from the various sense organs arrived in a "central sensorium" allowing perception to occur. Aristotle posits a single location for collection of, and action upon, the senses arguing that the subject that perceives, thinks, and moves is one and the same.

2.1.2.3 Aquinas

Aquinas in translating and interpreting the work of Aristotle discussed intelligent behaviour as directed action defined and chosen by the actor. Physical stimulations from the world are carried into the body as “phantasms” which are then acted upon as a result of comparison with generalisations of internal phantasms formed through passed experience. Aquinas used a metaphor of the intellect lighting the phantasms within the self; in contrast to Plato, the stimulations from the world were compared with a set of forms, internal, imperfect, and unique to the individual [30].

2.1.2.4 Descartes

Landmark work by René Descartes bought the question of intelligent behaviour into the Age of Reason, adopting a position of scepticism in order to avoid any unfounded assumptions [18]. The Cartesian Mind was in some sense a departure from the ideas of antiquity; where Plato distinguished between material “known to us through sensation” and forms, Descartes distinguished between “res extensa” that which is extended (in size and shape) in the world and “res cogitans” that which thinks. This position is known as Substance Dualism.

Aristotle required that there was a similar dichotomy, positing “prime matter” could not exist on its own, and must be “informed by a substantial form” which directs its characteristic behaviour. Therefore Aristotle placed the determination of behaviour of both humans and elements in their substance. Significantly, Descartes placed the behaviour of elements purely in their physical extension, describing it as “matter in motion” and the behaviour of humans was determined by the mind existing in non-corporeal “res extensa”. Some aspects of antiquity remained in Cartesian thought, however, agreeing with the Platonic tradition in holding the intellect in higher standing than sensory knowledge [34].

As described by Wheeler [103], “[Cartesian] intelligent action the form of a series of sense-represent-plan-move cycles”, in this framework nerves in the sense organs receive stimuli from the external world which sets in motion small fibres that connect the nerves with ventricles of the brain, in turn stimulating the flow of “animal spirits”. The pineal gland, considered by Descartes to be the “seat of the soul”, in interacting with the animal spirits was the conduit by which the mind was informed of the world by the body. On reception of the senses, the mind could represent the state of the world and reason upon it, finally commanding the body to move.

Wheeler also identifies that sensing and action were “theoretically disassociated” by being separated by reason. As such the *inner state* was the determining concept for intelligent behaviour in the Cartesian Mind.

2.1.3 Artificial inner states

The number of actions definable in a Jacquard loom was vastly superior to that of any autonomous artefact that had been invented before, but the sequence of steps, though represented symbolically, still had to be represented explicitly. The requirement for one card per line of woven pattern meant that the loom was limited by the number of warp threads the loom could manipulate at once and the practical problem of managing card chains as patterns grew in length.

2.1.3.1 The Analytical Engine

The first machine which could surpass this limitation was the Analytical Engine [54] designed, though never built, by Charles Babbage a contemporary of Jacquard. Babbage's innovation was to have the machine capable of performing a number of operations and to have the performed operation determined by punched cards. Whereas a single pattern of holes will always result in a certain woven pattern in the Jacquard loom, a punched card may be used as an operator or operand for one of a number of operations in the Analytical Engine. The range of available operations included — but was not limited to — elementary arithmetic, conditionally feeding forward or backwards by a certain number of cards in the chain, and halting the machine.

2.1.3.2 Procedurally defined behaviour

It was the ability to determine the subsequent card to be actioned as a condition of the state of the machine which allowed procedures to be described implicitly, rather than explicitly. Consider the procedure for calculating the factorial of 6, explicitly it can be written as the results of $6 \times 5 \times 4 \times 3 \times 2 \times 1$ but as a procedure it could be written in a general form "accumulate the product of all the positive integers less than or equal to n" followed by an input "$n = 6$" which would apply to all positive integers. The Analytical Engine could therefore compose a large number of complex procedures from the small set of operations which it was able to perform.

The defining feature of all the machines described so far is that their operation may continue entirely without human intervention once the action has started, and hence the state of a machine at each instant defines the state in the next instant. There is no capacity for intuition, rumination or inspiration in the tasks given. This type of behaviour is not exclusively limited to machines, there are certain similarities with slavery but the closest analogy amongst humans is with the human computer, who were required to process numbers in accordance with a given procedure that limited any requirement for insight. Alan Turing used the term "(mechanical) process" [100, p.262] to describe this kind of behaviour that could theoretically be carried out by a machine without human intervention.

2.1.3.3 Turing computability

Arguably the most important behaviour for which a corresponding mechanical procedure has been sought is that of determining whether "a given formula $\mathfrak{U}$ of the functional calculus K is provable", in a paper by Alan Turing in 1936 [100]. In investigating the possibility of such a mechanical procedure Turing invented a notation for defining procedures that could be simulated in a physical machine or by hand on paper. This was necessary as to give formal definition of a mechanical procedure, and so allow Turing to propose that an informal term as "a behaviour achievable purely through mechanical methods" may by replaced by the term "a behaviour which may be implemented on a Turing machine".

Related work was published by Alonzo Church, a contemporary of Turing, hence the proposal that all such mechanical computations can be carried out by a Turing machine became known as the Church-Turing thesis [12].

An important proof of Turing's was that certain numbers will not be representable in his notation, though large sets of numbers were [100]. The proof depended on the facts that there could not be infinitely many types of symbols printed on a tape as the difference between types was required to be greater than a lower bound, and the complexity of a symbol needed to be less than an upper bound before it could be identified at a glance. Turing gave the example of the decimal representations of the two numbers 999999999999999 and 9999999999999999 which have to be compared in a series of steps. Turing could identify some large sets of numbers which could be represented in his notation and significantly showed how each Turing machine could correspond to exactly one computable number.

The significance of this observation was that a symbolic reference to a Turing machine could be made upon the tape of another Turing machine. Turing then showed that there existed a Turing machine which could encounter this reference and perform the action of the referred Turing machine. The importance of this discovery was that a single machine implemented thus, known as a Universal Turing Machine, could perform any definable mechanical procedure without any reconfiguration other than modifying symbols on the tape.

Babbage had designed an architecture whereby actions could be symbolically defined as procedures but once running the procedure would output to another medium, the Analytical Engine was equipped with a number of output methods including decimal dials, a bell and a curve drawer [54]. The Universal Turing Machine, in contrast, read input and wrote output to the same tape, therefore if the output was the number of a further Turing Machine and control moved to the location of that number the referenced Turing machine would be executed. In having no distinction between input, output, and instruction, the Universal Turing machine had access to the means of modifying its own behaviour.

Building upon the description of Turing Machines, computable numbers and the Universal Turing machine, Turing gave a further proof; there is no solution to the Entscheidungsproblem. Formally described thus, "there can be no machine which, supplied with any one $\mathfrak{U}$ of these formulae, will eventually say whether $\mathfrak{U}$ is provable". At once Turing's work both introduces a formalisation of computation, demonstrates a remarkable property, and describes an important limitation in its power.

Now machines could be defined in a way that the behaviour of one machine was multiply realisable on any implementation of a universal machine, as the behaviour was now inseparable from the data, what remained was to find the procedure by which a machine could write further procedures of the machine's own volition, the ability to *learn*.

Here we chart the development of a loosely related set of ideas from engineering, philosophy and mathematics into an explicit field of scientific investigation. As a subject, *Artificial Intelligence* more clearly describes the area of study, being largely defined by (i) the principles of intelligence, (ii) the design of artefacts to exhibit those principles, and (iii) methods to indicate whether those artefacts are intelligent.

2.1.4 Turing's introduction of machine intelligence

2.1.4.1 1948 — Intelligent Machinery

Turing first treated the creation of a learning system as a serious scientific investigation in a report for the National Physics Laboratory in 1948 entitled *Intelligent Machinery* [99], however it was described as "not suitable for publication" although demonstrating "rather fundamental studies" [38, p.489]. In the report Turing describes two programmes towards the creation of an intelligent machine. The first approach was to "take a man as a whole and try to replace all the parts of him by machinery", while this was considered a "sure" method it was also admitted to be "too slow and impracticable". The more feasible approach was to "see what can be done ... more or less without a body", which included intellectual tasks such as game playing, language acquisition and mathematics.

This is an early example of a recurring dichotomy in the approach to building intelligent machines. Firstly, physically duplicating an intelligent being to an arbitrary degree was seen as a guaranteed method for success, but one that was clearly infeasible and threatened to grant no insight into intelligence itself. Secondly, investigating the abstract processes which characterise intelligence was immediately actionable, but required that intelligence was characterisable in such a way that the discovered processes were testable in an existent architecture even if it was, as in Turing's case, a paper machine implemented by hand.

In discussing intelligent behaviour in computable terms Turing must have hypothesised that an appropriate Turing machine existed, and hence could be discovered by a human searching for it intelligently. His emphasis in the 1948 report however was on the process by which a machine could reach the state of a universal machine implementing an intelligent machine with as little human interference as possible.

Turing broadly set out an architecture that could be "rewarded" or "punished" for its actions from outside the machine, causing its future behaviour to be respectively repeated or modified. The behaviour of rewarding and punishing was also within the grasp of another machine, extending Turing's investigation to include not only machines that could learn but also those that could *teach*.

Critical to the scientific field being conceptualised by Turing was the notion of determining whether or not any behaviour was intelligent. Even at this early stage Turing was sceptical about an objective measure, claiming that two people with a differing "state of mind and training" and knowledge of the system may commonly disagree on whether the expressed behaviour was intelligent. A scientific approach to determining intelligence could still be conceived of however, in the form of "a little experiment on these lines".

> It is not difficult to devise a paper machine which will play a not very bad game of chess. Now get three men as subjects for the experiment *A*,*B*,*C*. *A* and *C* are to be rather poor chess players, *B* is the operator who works the paper machine. (In order that he should be able to work it fairly fast it is advisable that he be both a mathematician and chess player). Two rooms are used with some arrangement for communicating moves and a game is played between *C* and either *A* or the paper machine. *C* may find it quite difficult to tell which he is playing.
>
> (This is a rather idealised form of an experiment I have actually done).

One role of this experiment is to show that mechanical processes exist which produce behaviour that can be hard to distinguish from characteristically intelligent behaviour. The role of *B*, the intermediary, is to ensure that *C*, the player, only receives information about the behaviour of *A*, which may be a mechanical process. Turing does not specify exactly whether a mechanical process exists that will *always* be *indistinguishable* from intelligent behaviour, though his admission of having conducted a similar experiment is suggestive that the scenario described is not intended to exist solely as a thought experiment.

A number of arguments against the use of words such as "teach" and "learn" with regards to machine behaviour were anticipated and addressed in the same paper. Most pertinently, the criticism that any intelligent behaviour was merely an expression of the intelligence of the human programmer was described as being "rather similar to the view that the credit for the discoveries of a pupil should be given to his teacher".

2.1.4.2 1950 — Computing Machinery and Intelligence

Where *Intelligent Machinery* was seminal work, the philosophical investigation of the creation and recognition of intelligence in a machine was most fully developed, and with most impact, in *Computing Machinery and Intelligence*. Turing outright rejects the question "Can machines think?" as being formed of terms whose definition are too poorly defined. An alternative question is proposed, dependent on the results of a new experiment, apparently inspired by the previously described Chess experiment, called "the 'imitation game' ".

> It is played with three people, a man (A), a woman (B) and an interrogator$\tilde{(}C$) who may be of either sex. The object of the game for the interrogator is to determine which of the other two is the man and which is the woman. He knows them by labels X and Y, and at the end of the game he says either 'X is A and Y is B' or 'X is B and Y is A.' The interrogator is allowed to put questions to A and B thus:
>
> > C: Will X please tell me the length of his or her hair?
>
> Now suppose X is actually A, then A must answer. It is A's object in the game to try to cause C to make the wrong identification. His answer might therefore be:
>
> > 'My hair is shingled, and the longest strands are about nine inches long.'
>
> In order that tones of voice may not help the interrogator the answers should be written, or better still, typewritten. The ideal arrangement is to have a teleprinter communicating between the two rooms. Alternatively the question and answers can be repeated by an intermediary. The object of the game for the third player (B) is to help the interrogator. The best strategy for her is probably to give truthful answers. She can add such things as 'I am a woman, don't listen to him!' to her answers, but it will avail nothing as the man can make similar remarks. We now ask the question, 'What will happen when a machine takes the part of A in this game?' Will the interrogator decide wrongly as often when the game is played like this as he does when the game is played between a man and a woman? These questions replace our original, 'Can machines think?'

This question, equivalently phrased as; "Are there imaginable digital computers which would do well in the imitation game?"; "Are there discrete-state machines which would do well?"; and "Is it true that by modifying [one particular digital computer C] this computer to have an adequate storage, suitably increasing its speed of action, and providing it with an appropriate programme, C can be made to play satisfactorily the part of A in the imitation game, the part of B being taken by a man?"; marks the creation of a process for indicating intelligence. It is not claimed to be a truth statement about the objective presence of intelligence, just a suitable working definition of a goal for the research field.

Turing goes on to say later that he firmly believes that it will be possible to build a machine "in about fifty years' time" that would "play the imitation game so well that the average interrogator will not have more than 70 per cent chance of making the right identification after five minutes of questioning". With this predicted progress in technology, and similar development in the "the use of words and general educated opinion", Turing also predicted that "one will be able to speak of machines thinking without expecting to be contradicted". Turing accepted that thinking as implemented on a digital computer may be "very different from what a man does" but suggested that whilst this objection is "very strong" it is one that "we need not be troubled by" should such a machine be constructed.

There were no explicit limits to the question of which principles founded intelligence, but only "digital computers" were permitted to "take part in our game". In allowing only digital machines to participate in the imitation game, explicitly to avoid any candidates being simply "men born in the usual manner" Turing also prohibited, implicitly, any method of intelligence that does not fall within Turing computability. Here again we see an example of the dichotomy between either building an intelligent being through duplication or simulating intelligent behaviour on a determined architecture.

2.1.5 Definition of the Artificial Intelligence project

Taken together, the delineation of the suitable approaches and goals as described by Turing formed the foundation upon which a full body of scientific study was to be built. The focus was on being able to describe mechanically the things intelligent humans do, the process was designing digital computers to implement these descriptions, and their validity was determined chiefly by evaluation of the resulting behaviour.

2.1.5.1 1956 — The Dartmouth conference

At a conference in held in Dartmouth; New Hampshire, specifically to formalise the field, and coin the name, John McCarthy set out a plan, thus:

> We propose that a 2 month, 10 man study of artificial intelligence be carried out during the summer of 1956 at Dartmouth College in Hanover, New Hampshire. The study is to proceed on the basis of the conjecture that every aspect of learning or any other feature of intelligence can in principle be so precisely described that a machine can be made to simulate it. An attempt will be made to find how to make machines use language, form abstractions and concepts, solve kinds of problems now reserved for humans, and improve themselves. We think that a significant advance can be made in one or more of these problems if a carefully selected group of scientists work on it together for a summer. [50]

This proposal therefore was an explicit declaration of the fundamental assumption. Not only that every feature of intelligence can be simulated but also that a machine can and will be made to do so. At the conference itself some seminal work was presented, including Simon and Newell's Logic Theory Machine (LT).

Even in this proposal, the dichotomy of approaches was apparent, in discussing the requirement for some sort of originality, it was decided that something more sophisticated than simply adding randomness was required and that two procedures were available, "One of these is to find how the brain manages to do this sort of thing and copy it. The other is to take some class of real problems which require originality in their solution and attempt to find a way to write a program to solve them".

An echo of the Turing's imitation principle of intelligence is also present, as McCarthy also avoids defining a formal, objective test for intelligence, stating "For the present purpose the artificial intelligence problem is taken to be that of making a machine behave in ways that would be called intelligent if a human were so behaving." [50]

2.2 Making a Mind versus Modelling the Brain

One approach, fundamentally grounded in the mind, was searching for the mechanisms through which inner representations could be manipulated to result in representations of actions to be actuated in the body. It was mind prior, as it assumed a mind could be built, then a body could be bolted on like a marionette.

The other branch, most distinct by its foundation in the body, attempted to model the action of the brain. Hypothesising the phenomena of the mind would emerge, or arise from the replication of the physical systems. It was brain prior as it assumed the structures of the brain, entirely defined and properly constructed was a sufficient condition for the emergence of a mind.

Notably, at this stage, the approach of duplicating a physically intelligent system was conceptualised in a way that was also Turing computable. The intelligent being was largely equated with the activity of the brain, the brain was largely equated with actions of neurons, and neurons in turn were being equated with the action of the abstract, computable, model developed by Pitts and McCulloch. Either approach, therefore, though distinct in their origin could be carried out through design, implementation and observing the behaviour of programs.

The dichotomy in these approaches was described by Dreyfus as Making a Mind vs. Modelling the Brain [23].

2.2.1 Making a Mind

The founding idea behind Making a Mind was that the mind was an emergent phenomenon that resulted from the formal manipulation of symbols, patterns of which were representative of features of the outside world. There was no upper limit to what could be represented, complex or subtle, object or concept, but there was assumed to be a lower limit, which Dreyfus likened to the ultimate 'simples' described by Leibniz, in terms of which all complex concepts can be understood. A large aspect of the approach was the identification of an appropriate set of such simple terms with which all aspects of human intelligent behaviour could be understood.

Given a fully defined context such as in the game of chess, the simple terms can be clearly seen. All that would be needed to fully describe all aspects of a game of chess can be captured in terms of spaces, pieces, moves and so on. Further problems such as the clothing of one of the players, or the minuscule positioning of a piece within its square e.g. whether it exactly centred, or slightly overlapping an edge, can be ignored.

The solubility of identifying the simple terms of a well defined context allowed the projects founded on the representation of the situation and manipulations thereof to have some early success that was, in appearance, startling and impressive.

2.2.1.1 1956 — The logic theory machine

A project at the RAND Corporation produced the Logic Theory Machine (LT) [70] which would attempt to prove that a given theorem could be proved from a given set of axioms. The significance of LT was that the procedure used was not simply an exhaustive search of the possible solution space, but extra techniques were implemented to apparently guide the search. These techniques, named heuristics, were intended to demonstrate that a machine could reach a solution using methods similar to those used by an insightful expert, as opposed to one merely following an effective method.

In the case of an expert proving a logical theorem, one process by which investigation was deemed to be guided was the recognition when theorems were similar and hence LT was designed with a method of evaluating the similarity of two theorems. Another heuristic employed by LT was to store any proven theorems and use them alongside the initially given axioms to guide further investigation, described by Simon and Newell as "learning".

This approach achieved some success, with LT being able to prove a number of elementary theorems, with some elegance. A novel proof produced by LT of Russell and Whitehead's theorem 2.01 from *Principia Mathematica* was submitted to the *Journal of Symbolic Logic*. Publication was refused on the grounds of the proof being co-authored by a program [24, p. 184].

2.2.1.2 1958 — The Physical Symbol System Hypothesis

The initial success led to two significant beliefs, firstly that progress would continue largely unabated, and secondly that the success of the approach was grounded in its capturing of the nature of intelligence. The first belief is aptly demonstrated in a quote from Newell and Simon relating to the predicted progress in heuristic-based systems "[I]n a visible future ... the range of problems [computers] can handle will be coextensive with the range to which the human mind has been applied." [97]. The second belief is more formally defined in The Physical-Symbol System Hypothesis [69].

> A physical-symbol system has the necessary and sufficient means for general intelligent action.
>
> By "necessary" we mean that any system that exhibits general intelligence will prove upon analysis to be a physical-symbol system. By "sufficient" we mean that any physical-symbol system of sufficient size can be organized further to exhibit general intelligence.

The significance of the thesis is hard to overstate, the claim that a physical-symbol system may exhibit general intelligence (not just intelligent behaviour) and that any system exhibiting general intelligence must be a physical-symbol system implies that a formalism for the intelligence exists, and any other conception of intelligence will be reducible to an equivalent formalism. This is described as an "empirical hypothesis" which therefore largely described how research may progress to work towards finally discovering and implementing a generally intelligent system.

Thus the research community was ready to start embarking on the projects that would exhibit general intelligence.

2.2.1.3 1959 — The General Problem Solver (GPS)

Buoyed by the success of LT, Simon and Newell set out to design a similar system that would be able to solve any well defined problem presented to it [68], the project was something of a success, being practically implemented and leading to a successor, the Soar architecture. Unfortunately the GPS could only solve problems that defined a small context, as more complex problems quickly ran into a combinatorial explosion of intermediate states.

2.2.1.4 1964 — Dendral-64

Considered the first *expert system* as it attempted to reproduce the behaviour of experts in the same field, Dendral-64 would receive spectrometry data as input and respond with a number of candidates for the chemical that produced the data [44]. Notably, Dendral-64 used heuristic processing in which rules-of-thumb used by human experts were encoded into the behaviour of the machine, rather than simple, comprehensive searches of the entire search space. This approach traded a guarantee that an answer would be found for processing times that were not so long as to be impractical. This meant the output was likely to be fallible, so Dendral-64 was designed to return a number of likely candidates that a relatively junior expert would be able to review to determine the most likely candidate. Even with the greatly reduced processing times and internal data store requirements of a heuristic process Dendral-64 required to be only applied to a very tightly defined context, such as spectroscopy, and would not ever branch out to other problems.

2.2.1.5 1972 — SHRDLU

Winograd's SHRDLU, could react to and respond with a subset of English which allowed surprisingly coherent conversations and complex behaviour [106]. Its behaviour was determined by the translation of sentences of plain text into a formal declaration of action, or propositions about the environment. This could then be compared against SHRDLU's representation of the world and either the truth of the proposition could be determined logically, or the difference between the required state and the current state could be determined and hence an effective sequence of actions could be determined to perform the requested task.

2.2.2 Critiques of Making a Mind

2.2.2.1 1973 — The Lighthill Report

By the mid-seventies, requests for funding in Artificial Intelligence projects had reached a level of frequency that it came to the attention of the government which had no official stance on the value of related research. A report was commissioned to evaluate the current and potential progress. Professor Sir James Lighthill FRS was chosen as someone who was not actively involved in the field but with the required experience in mathematics, biology and engineering. The report drew a distinction analogous to the dichotomy previously described, splitting work into the categories of; "Advanced Automation" (category A), for any work whose goal was to produce useful behaviour which would be considered intelligent if seen performed by a human; "Computer-based [Central Nervous System] research" (category C), for any work that wanted to further the understanding of the natural central nervous system through study upon models of artificial neurons. A third category, "Building Robots" (category B) was placed theoretically between the other two categories. Category B contained all work which aimed to make progress towards the goal of the creation of an artefact that exhibited behaviour from category A through methods developed by work in category C, it was therefore also referred to as the "Bridge category" [46].

The tone of the Lighthill report was more cautious than much related literature, often referring directly to admissions of naïvety and subsequent disappointment from distinguished researchers. Categories A and C were therefore hailed as promising endeavours within the contexts that individual projects in category A remained within a tightly defined context, and work in category C remained tightly aligned to merely computationally supporting work that would have otherwise been performed in psychology and neurobiology. The tone was largely critical, though not unfair, of category B which was unable to unite categories A and C into a single research field by citing successful projects that drew heavily on both categories.

The central critique of category B was that no projects existed that were both (i) sufficiently related to the other two categories and (ii) successful. This critique can be interpreted as an example of the "No True Scotsman" informal fallacy in that any successful project suggested as a counterexample could be immediately claimed as being not sufficiently in either category A or C, and any project that drew significantly from both categories A and C when suggested as a counterexample could be claimed to be insufficiently successful. This can be seen in the report as any projects that are inarguably successful, are inevitably described as using "general computational theory" which, given that the roots of computational theory lay in Turing's work in decidability and intelligence is not obviously distinct from category A research. Similarly, Winograd's SHRDLU project which is admitted to draw deeply on work from both other categories was claimed to be essentially not successful enough as "suggestions for possible developments in other areas that can properly be inferred from the studies are rather discouraging". Lighthill also noted that "one swallow does not make a summer", though this was "banal" by his own admission.

The Lighthill report had a significantly negative effect on AI research funding, though it was not entirely unfair. It was broadly compatible with the view of Dreyfus and potentially saved the reputation of AI from a more disastrous period which may have occurred in the future had overconfidence and overpromising been allowed to continue unchecked.

2.2.2.2 1980 — The Chinese Room Argument

Two stances on the ultimate potential of the Artificial Intelligence project were identified by John Searle [95]. "Strong" AI held that "the appropriately programmed computer really is a mind" by virtue of instantiating its program alone. "Weak" or "cautious" AI held that artificial intelligence was a useful tool with certain practical applications and academic value, but allowed that there may not exist any program which would instantiate a mind.

Searle described a thought experiment, to argue in favour of weak AI. In its original form the target of the argument was a claim that Schank and Ableson's *Script Applier Mechanism* (SAM) could understand the stories that were presented as input, it should be noted that the claim was not attributed to Schank or Ableson. Searle is clear to state that the argument applies to any formal system, such as any Turing computable process.

Described in work published in 1975 [93], SAM could receive input in the form of a script, a formal syntax for describing a "story" (Figure 2.5) and subsequently respond to further input in the form of "questions" in natural English with "answers to the questions" in natural English (Figure 2.6).

```
script: restaurant
roles:  customer, waitress, chef, cashier
reason: to get food so as to go up in pleasure
           and down in hunger

scene 1: entering
         PTRANS self into restaurant
         ATTEND eyes to where empty tables are
         MBUILD where to sit
         PTRANS self to table
         MOVE sit down

scene 2: ordering
         ATRANS receive menu
         MTRANS read menu
         MBUILD decide what self wants
         MTRANS order to waitress

scene 3: eating
         ATRANS receive food
         INGEST food

scene 4: exiting
         MTRANS ask for check
         ATRANS receive check
         ATRANS tip to waitress
         PTRANS self to cashier
         ATRANS money to cashier
         PTRANS self out of restaurant
```

Figure 2.5: "A sketch of a script for a restaurant from the point of view of the customer." Actions are specified in terms such as MTRANS for mental information transfer and PTRANS to move a physical object. [93]

```
Input:
     John went to a restaurant. The hostess seat-
     ed John.  The hostess gave John a menu. The
     waiter came to the table.  John ordered lob-
     ster.  John was served quickly.  John left a
     large tip.  John left the restaurant.

Questions and Answers:

Q: What did John eat?
A: LOBSTER.
Q: Who gave John the menu?
A: THE HOSTESS.
Q: Who gave John the lobster?
A: PROBABLY THE WAITER.
Q: Who paid the check?
A: PROBABLY JOHN.
Q: What happened when John went to the table?
A: THE HOSTESS GAVE HIM A MENU AND JOHN SAT DOWN.
Q: Why did John get a menu?
A: SO HE COULD ORDER.
Q: Why did John give the waiter a large tip?
A: BECAUSE HE WAS SERVED QUICKLY.
```

Figure 2.6: Example session with the Script Applier Mechanism [93]

The experiment asks one to imagine whether manually processing the input that would be received by SAM and hence producing the same output will give the human processor an understanding of the story described in the script. To avoid the objection that the human processor would be able to read the script as they would a normal text, Searle defines that all text is written in a language he cannot read, Chinese in this case, so all symbols are identified naïvely.

> Suppose that I'm locked in a room and given a large batch of Chinese writing. Suppose furthermore (as is indeed the case) that I know no Chinese, either written or spoken, and that I'm not even confident that I could recognize Chinese writing as Chinese writing distinct from, say, Japanese writing or meaningless squiggles. To me, Chinese writing is just so many meaningless squiggles.
>
> Now suppose further that after this first batch of Chinese writing I am given a second batch of Chinese script together with a set of rules for correlating the second batch with the first batch. The rules are in English, and I understand

> these rules as well as any other native speaker of English. They enable me to correlate one set of formal symbols with another set of formal symbols, and all that 'formal' means here is that I can identify the symbols entirely by their shapes. Now suppose also that I am given a third batch of Chinese symbols together with some instructions, again in English, that enable me to correlate elements of this third batch with the first two batches, and these rules instruct me how to give back certain Chinese symbols with certain sorts of shapes in response to certain sorts of shapes given me in the third batch. Unknown to me, the people who are giving me all of these symbols call the first batch "a script", they call the second batch a "story", and they call the third batch "questions". Furthermore, they call the symbols I give them back in response to the third batch "answers to the questions", and the set of rules in English that they gave me, they call "the program". [95]

The argument demonstrates that an instantiation of SAM is not sufficient for understanding as Searle could *be the instantiation* himself, and would not understand the story. This is taken by adherents to the Chinese Room Argument to show formal symbol manipulation alone does not give rise to understanding, in other words, syntax is not semantics.

In discussing what may give rise to understanding, if not symbol manipulation, Searle distinguishes between computers and machines, stating that the brain is a type of machine, but that understanding must arise from its causal properties, rather than the instantiation of a formal syntax.

2.2.2.3 Domain specificity

Two problems were encountered relating to domain specificity of an AI system. The first was that there was no general procedure for taking two symbolic solutions, such as SHRDLU and GPS and combining them to form a system that could behave intelligently in the union of the two contexts. The second was that as a situation became gradually more complex, the set of combinations and required manipulations became exponentially larger, eventually becoming unmanageable. These problems could be managed initially by manually combining existing systems on a case by case basis, and by increasing the complexity of a system to handle a more detailed domain. For example, SHRDLU's blocks world relied on the combination of "the vision project of David Huffman (1971), the vision and constraint-propagation work of David Waltzi (1975), the learning theory of Patrick Winston (1970), the natural language understanding program of Terry Winograd (1972), and the planner of Scott Fahlmañ(1974)" [92] and could act in a non-trivially complex context of a physical simulation of blocks and a subset of natural English. The process of integrating further systems into SHRDLU or expanding them to handle a more detailed physical simulation would have eventually become so complex as to be infeasible to implement, either by limiting the processing power of the utilised hardware, or by pushing the human designers past their capabilities.

2.2.2.4 The Common Sense Problem

A single game of chess can be broadly captured by recording which space each remaining piece was occupying after every move. To capture a more complex, or less well defined context, such as the entirety of a chess championship, it is not always immediately clear which facts not immediately pertinent to a game may become features that are critically important to explaining further occurrences, such as a judgement about an illegal move. Imagine, for example, that a player's dangling sleeve knocked a piece from one square to another.

An approach that was a potential solution to the domain specificity problems of system integration and combinatorial explosion was to design systems that had a common sense, and therefore had access to all knowledge that may become important. Marvin Minsky has been associated with a claim that "representing a few million facts about objects including their functions" would solve this problem [22], but this simply exacerbated the problem of determining which fact was relevant at any one time. Consider again the example of the chess championship: a system may have sufficient data about invalid moves in chess to know to take account of a players loose clothing, but never acts appropriately as it is taking too long in considering whether to act on a vast array of other facts such as the impossibility of whistling whilst chewing gum, and common misspellings of the word scissors.

2.2.2.5 The first step fallacy

Artificial Intelligence projects that attempted to produce intelligent behaviour through formal manipulation of symbols quickly demonstrated encouraging results, notable examples are Winograd's SHRDLU which could be interacted with through a subset of the English language and appeared to exhibit comprehensive knowledge of its environment "blocks world". There were also impressive early results from projects based around language acquisition, and theorem solving.

Many projects proudly publicised their achievements, and this would often be accompanied by predictions of steady progress until a system had a capability at the level of an expert human. However, the progress inevitably slowed. This became known as a "first step" fallacy, where impressive early results were interpreted as a promise of similarly impressive progress when trying to expand a system into broader or more advanced functionality. The cause of the fallacy was identified by Dreyfus [21] as the combinatorial explosion of complexity encountered as a system was expanded into more varied domains. The problem being that a formal system has to objectively process all symbols and input patterns, where as a human experiences an environment in a way better described as having significant things present themselves to you.

2.2.3 Modelling The Brain

As the brain was taken to be the seat of intelligence, an approach to creating intelligent behaviour was to simply reproduce, at a given level of abstraction, the action of the brain, under the assumption that the same manipulations of stimuli seen in the brain would produce the same responses in artificial systems.

2.2.3.1 Associationism

Philosophically, the character of Connectionism can be traced back at least as far as work by John Locke, named "associationism" and founded on the idea of co-occurrences and correlations of physical events and mental states. Commonly summarised by the epithet *items that "go together" in experience will subsequently "go together" in thought*, it was a typically empiricist approach, explaining where possible learning and behaviour as a result of past experience [33].

Ideas are associated in the mind through experience and ideas are derived from sensations. The sensations themselves are caused by something outside the head, so there is still a concept of "inner" as distinct from the outside world.

This allowed the study of intelligent behaviour to be grounded in visible processes in the world. Experiences will effect someone in such a way that they will be more likely to act in the same way again. The advantage was that behaviour selection could have a causal explanation. This had implications for the notion of free will in intelligent behaviour, as identified by Nietzsche in 1887, "seeking the truly effective and directing agent ... where the intellectual pride of man would least desire to find it (in the vis inertiae of habit ... or in a blind and chance mechanistic hooking together of ideas ... passive, automatic, reflexive, molecular and thoroughly stupid)" [73].

2.2.3.2 1943 — McCulloch and Pitts model of the neuron

Connectionist AI is a fundamentally distinct approach to producing intelligent behaviour. In a seminal paper by W. S. McCulloch and W. H. Pitts [53] and drawing strongly on an "all-or-none" interpretation of the activity of a neuron, a formal model of nervous activity was defined. Various features of physical neurons were declared irrelevant, such as the speed of propagation of signals. The brain was therefore defined by the activity of a network of interacting formal neurons. All logical calculations were theorised to have at least one net with equivalent behaviour and all nets were describable by a formal statement. The advantage of encoding a behaviour in a net, rather than a series of statements was that the process of discovering the appropriate configuration could be mechanised, in a way analogous to Turing's suggestion in *Intelligent Machinery*.

2.2.3.3 1949 — The Organization of Behaviour

The process for discovering the neural network of an appropriate configuration was grounded in the work of Hebb who postulated that repeated co-occurrences of activity of two cells, most significantly in the case of a neuron whose firing is determined by the presence of a certain stimulus, firing at the same time as a neuron which is instrumental in

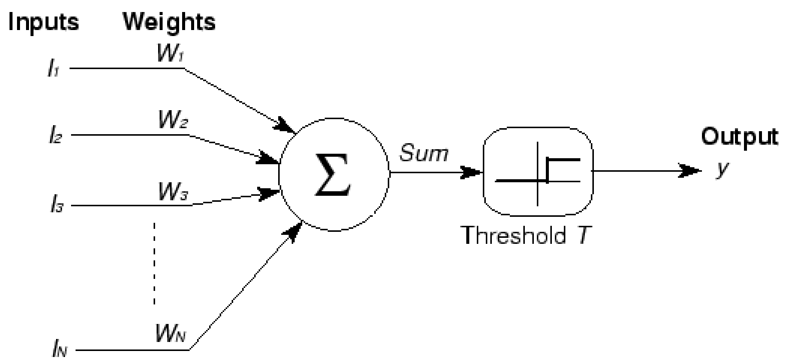


Figure 2.7: The McCulloch and Pitts model of the neuron, showing the weighted sum of inputs 1 to N being passed through a threshold function.

the production of a suitable behaviour in response to the stimulus, increases the probability that activation of the cells will co-occur in the future [35]. Quoting Hebb "the persistence or repetition of a reverberatory activity (or "trace") tends to induce lasting cellular changes that add to its stability". A colloquial description of the same phenomenon is to say "neurons that fire together, wire together", capturing the spirit of Hebb's observation, if not the precise letter.

2.2.3.4 1958 — Mechanisation of thought and the Perceptron

In 1958 many of the delegates present at the Dartmouth conference met again in Teddington, London at an annual symposium hosted by the National Physical Laboratory; the theme for that year was "The Mechanisation of Thought Processes" [52]. Many of the subjects that would come to characterise much of AI were present such as medical expert systems, image recognition, and neural nets for pattern discrimination.

At this event [51] Rosenblatt made a statement that although the brain makes decisions and hence controls the body based on logic rules, it also interprets the external environment. In this recognition of the behaviour of the brain being coupled with the environment Rosenblatt made clear his concern not only with what the brain did, but how it did it. Developing an internal model of logical laws from experience was a distinct approach from the symbolic approach favoured by Minsky and McCarthy as they were looking for a set of logical functions that, statically implemented, would be able to produce the full range of required behaviour. Rosenblatt was clearly aware that an artificial neural network could be simulated on a Universal Machine, and indeed Rosenblatt's belief was that a computer used any other way would be inadequate to express intelligence.

Rosenblatt united the theoretical model of the with biological observations in a biologically plausible model of computation that could be used for both prediction of neuronal activity as well as synthesis of behaving systems in a model he named the Perceptron [89]. Various training methods were applicable to the network which would be initialised with random connections, of random strength, between the neurons. Training would take the form of providing stimuli with known correct responses and in the event of an incorrect output, inhibiting the weights of any units that fired incorrectly, and enhancing the weights of any units that incorrectly failed to fire. If the problem was linearly separable then the network was guaranteed to converge on the set of weights that would perform the specified mapping, should it exist [74].

2.2.3.5 1969 — Perceptrons

A critical event in the Connectionist approach to Artificial Intelligence was the publication in 1969 of *Perceptrons* by Minsky and Papert [57]. The aim of the book was to discover the capabilities of the type of parallel computing seen in artificial neural networks, by formulating various mathematical theorems on the capabilities of the Perceptron. The Perceptron was considered a suitable candidate to represent the entirety of the approach as its characteristic of "adding up evidence obtained from many small experiments" was shared with most, and perhaps all, Artificial Neural Networks, and in recognition of the pioneering work of Frank Rosenblatt. The analysis was not intended to be critical of the Perceptron, merely mathematically objective, reiterating the claim in the 1988 reprint that "romantic claims [of Perceptrons' capabilities] have been less wrong than the pompous criticisms", though important limitations were described [56]. It was shown that some problems were unsolvable within the given architecture, whereas other problems were "surprisingly tractable". Minsky and Papert make clear that their intuition was that research into multi-layer Perceptrons would be 'sterile', but they encouraged research into elucidating, or rejecting, their judgement. Nevertheless, *Perceptrons* has earned the reputation of dealing a near-fatal blow to the attitude of academics and funding bodies towards Perceptrons, and Connectionism in general.

The canonical example of a non-linearly separable problem, hence one unsolvable by the

In_1	In_2	Out
0	0	0
0	1	1
1	0	1
1	1	0

Table 2.1: Minimal training set for defining the "exclusive or" function. A single layer perceptron *will not* converge to a solution when trained against this set.

In_1	In_2	In_3	Out
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

Table 2.2: Training set for defining the "exclusive or" function, augmented with a third input. A single layer Perceptron *will* converge to a solution when trained against this set.

Perceptron, is the XOR function. The mapping from two inputs to one output which defines the XOR function are shown in Table 2.1. Minsky and Papert observe that a Perceptron could be trained reproduce the XOR function by augmenting the training set with a third input column which, so while training a Perceptron on the input-output pattern mappings in Table 2.1 would fail, training against the set in Table 2.2 would succeed. In this case the value of the unit in column In_3 represents whether the value of both In_1 and In_2 is 1. The same effect can be achieved by training a Perceptron against the set in Table 2.1 if it has a hidden unit that accepted input from In_1 and In_2, though a procedure for finding the requisite connection weights, of similar power and utility to the Perceptron convergence procedure was not widely known until the publication of *Parallel Distributed Processing*.

2.2.3.6 1981 — Hinton mapping

In 1981 [37] a technique was introduced to address the problem of stimulus equivalence which by 1992 had been named Hinton mapping [8]. This problem, which is the invariance of symbolic information independent of transformation within a search space, is a problem present in neural networks when applied to visual identification tasks. The occurs when many different patterns of stimulation can identify the same object. For example, a printed letter 'A' should be identified as such regardless of its orientation, size or position but would present a different stimulus to a neural network under such transformations.

Hinton mapping is based in the theoretical assumption that visual processing depends on a representation of a scene in terms of three-dimensional features relative to a point of view. The representation of these features is enabled be groups of retina-based units, object-based units and mapping cells. Each retina-based unit represents the presence of a feature, which in the case of detecting possibly rotated and resized letters in a 2D plane could be a horizontal line of a certain length, or a slightly angled line of another length. Each object-based unit represents a canonical instance of an object to be detected. Mapping cells read from collections of these retina-based units, and hence represent the detection of the features that define a certain presence, for example, the presence of a slightly rotated capital A may be represented for a mapping cell by reading from the retina-based units which are active in the presence of a two-tilted lines and one shorter tilted line. This process requires one mapping cell per valid hypothesis. The mapping cell which represents the best mapping of an object-based unit to multiple retina-based units will become the most active [37]. The detected object is therefore the object cell associated with the most active mapping cell. Hinton mapping can be computationally expensive, as the number of mappings increases combinatorially when each feature is presented in all possible combinations of rotation, position, size, and any other transformations that may be applicable [6].

2.2.3.7 1986 — Parallel Distributed Processing

Parallel Distributed Processing [91] noted that *Perceptrons* justified a critique that was limited to single layer Perceptrons by explaining that the set of multi-layer Perceptrons was so large as to be vacuous; one could implement a Universal Turing Machine with sufficient linear threshold units; and that there was no strong training procedure for multi-layer Perceptrons analogous to the *Perceptron convergence procedure* for single-layer Perceptrons. These arguments were countered by firstly explaining that the focus was on the investigation of multi-layer Perceptrons that were "as parallel as possible" and hence far from implementing a Universal Turing Machine, and that a number of strong training proce-

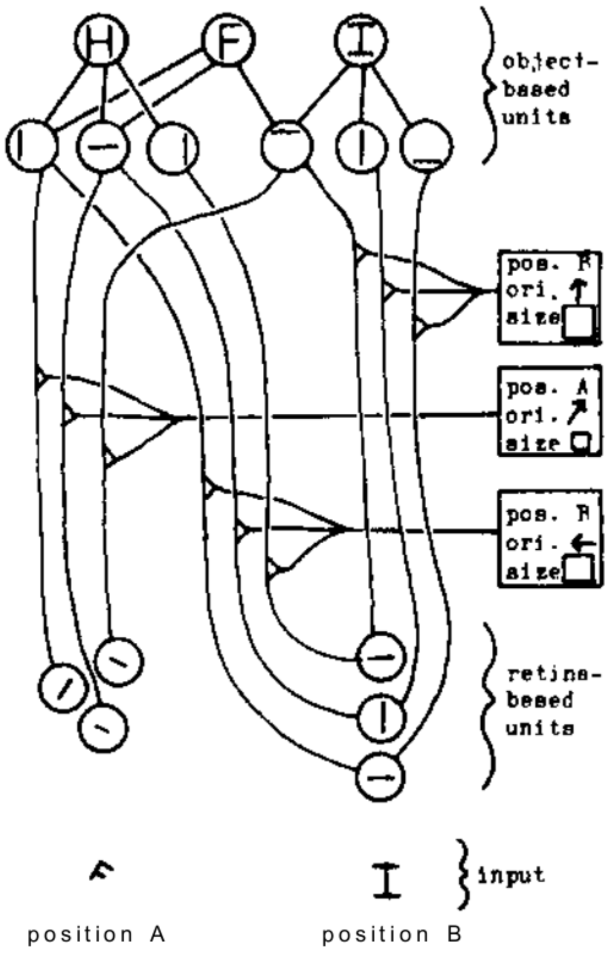


Figure 2.8: A diagram of Hinton Mapping from [37]. The retina based units are sensitive to basic features in a visual scene. Each retina-based unit is joined to an object-based unit by a mapping unit. The activity of an object-based unit is the multiple of the activity of a retina-based unit and the mapping unit which joins them. The activity of a mapping unit is the multiple of the activity of an object-based unit and the mapping unit which joins them. Many more retina, object, and mapping units are required than are depicted.

dures had since been discovered for systems with hidden layers, going so far as to describe a generalised procedure that could be applied to "arbitrary networks with multiple layers and feedback among layers".

2.2.3.8 Comparison of symbolic AI and neural networks

Fundamentally distinct from Symbolic AI, an approach emerged where the process by which intelligent behaviour could be investigated was grounded in the study of mechanisms and biological structures that may may give rise to learning and perception in the human, most importantly, in the human brain. As models of the brain conventionally took the individual neuron to be its simplest constituent and hence the defining feature of any specific model being the characteristic connections between the neurons, the field became known as Connectionist AI.

Neural Networks were in some sense a success, exhibiting traits that bear the mark of the intelligent: they are often described as learning: in degrading gracefully; in being biologically inspired; and able to generalise. However, algorithmic artificial neural networks are still fundamentally computational, existing outside of time, being Turing equivalent. Hence while avoiding some of the criticisms and pitfalls of symbolic formal systems they share some of the problems, not to mention introducing some problems of their own.

2.2.4 Critiques of Modelling the Brain

There are weaknesses in the approach of Modelling the Brain, although the focus is on the structure of the system which performs the learning. Across all neural architectures the quality of the results is dependent on the composition of the chosen training data. This leads to a large amount of time spent collecting and pre-processing a data set which can be seen as 'sneaking intelligence in through the back door'.

The requirements for the size or composition of the training data may become prohibitive. A study of the comparative performance of speech recognition systems trained on varying amounts of data found that performance improved linearly, in terms of percentage word identification error, as the amount of training data increased exponentially, in terms of hours of natural speech [58]. Significantly, a successfully trained network may not offer any insight to the user about how the network has solved the problem, and hence offer no insight into how the human brain solves analogous problems.

Furthermore, the neural architectures which are defining the field are arguably a poor representation of the structure and activity of the mind. Neurons as modelled in neural networks are often modelled as static integrators of input. Even in recurrent neural networks, where the structure of the network may include loops, the full real-time dynamics of neurons are rarely modelled. Neuroanatomy work from Freeman [30] described neurons as releasing their neurotransmitters as an eletric potential over time. Not only is this simple behaviour often ignored in artificial neural networks, it is a process which also induces a negative current in the volume surrounding the neuron, not just at the axon, affecting much more than just the 'connected' neuron. Freeman also shows how collections of neurons interact in such as way as to be able to maintain a number of different steady states of firing patterns each of which effectively implement context-specific behavioural units, a major purpose of which is to modify the steady state of other collections of neurons. This real-time stateful complexity is not modelled in even the most highly publicised neuromorphic systems such as FACETS and SpiNNaker systems of the Human Brain Project.

2.2.5 The state of the art

Despite arguments about the ultimate limits of AI projects, a number of successful projects have been released to public use, using novel techniques and powerful hardware to reduce, if not remove, limitations.

Machine Translation, where text in a source language is automatically translated into text in a target language is a classic application of AI. A simple machine translation can be achieved by mapping each word in the source language to word in the target language. The action of such a simple system will be analogous to using a bilingual dictionary, looking up every word in the source text and replacing it with the associated word in the dictionary. Special behaviour would have to be implemented for each case where the word in the source language did not exist in the mapping, or when one word in the source language mapped to more than one word in the target language. These systems are simple to design and implement and will perform adequately with certain combinations of source text, target text, language pair and dictionary. The meaning of individual words, however, is affected by the context in which they are used. Many examples exist of this contextual dependence, of which the phrases "Time flies like and arrow, fruit flies like a banana" [13] and "Police help dog bite victim" are memorable examples.

Statistical Machine Translation (SMT) is an approach that attempts to address this problem, with considerable success. Firstly, rather than referring to a formal mapping between words, SMT depends on a collection of documents translated by expert humans as training data. Secondly, rather than attempting to create a mapping between two languages that will work in certain contexts, SMT calculates a probability distribution representing the probability of a phrase in the target language being the correct translation of a given source phrase. The calculation is broadly informed by looking up the phrase in the training corpus and seeing how it has been translated previously, taking into account the similarity of the words surrounding the source phrase and similar phrases in the training data.

Google Translate is a product using SMT and a number of publications by Google show various SMT techniques outperforming competing techniques. Google reportedly use United Nations and European Union documents due to the fact that they are routinely translated into six languages, resulting in a huge corpus [98].

A collection of the best techniques, and raw computing power led to the figurehead of modern artificial intelligence called Watson [26]. Watson can do many things often thought beyond the capabilities of AI by its critics, natural language processing as well as the capability to handle ambiguity in its input. Watson competed in and won the American Quiz Show *Jeopardy!*, but made one telling error, answering "Toronto" to a question in the category of "US Cities". The notable error in this response, that Toronto is not located in the US, shows that Watson can be surprisingly incorrect, as well as surprisingly accurate.

Once artificial neural networks had shown themselves to be not only possibly applicable to a large set of problems, but also practically applicable to existent problems, and a training procedure had been developed, research continued and progressed until a number of powerful techniques had been developed. The current state of the art is represented in the techniques: Reservoir Computing [94], and Deep Learning [42].

Reservoir Computing is a technique whereby a network of artificial neurons is used in a randomly initiated and fixed state. In contrast to the Perceptron, where a network is modified until the intended mappings of input patterns to output patterns is achieved, a reservoir will approximate the intended non-linear function to a varying degree depending on which inputs and outputs are considered to be the ones corresponding to a given problem, the task of training is therefore to find which set of input and output nodes appear to map input patterns to the intended output pattern.

In this way, a single reservoir can be used to solve many problems. Each problem would be solved by reading from different sets of nodes from the same network. Training on the weights of the output layer only is equivalent to simply listening to whichever set of output neurons appears to encode the correct behaviour. This is a much simpler task than modifying an entire, often fully connected and many layered, network. The interest of this approach is that a randomly initiated network, with appropriate dynamics implemented almost by chance, and hugely reusable, is broadly biologically plausible.

Deep Learning uses a sequence of non-linear transformations broadly characterised as expecting each layer to output some higher-level features extracted from the lower-level features output by the previous layer. Individual layers are commonly stand-alone neural networks resembling Perceptrons, but may include any architecture that will map input patterns of activation to output patterns of activation. Deep Learning is often used with a minimum of task specific customisation of the architecture, which suggests a general approach which is not restricted to a single domain, as were Expert Systems. This is only achieved, however, by supplying the network with a set of training data which represents the raw data of the problem to be solved. For a Deep Learning system to be applicable to more than one problem it would have to receive training data containing both types of problems, each of which would greatly increase the effective noise in the other.

2.3 Swarm Intelligence — Mind is social

An investigation separate from the search for intelligent machinery was the endeavour to understand the intelligent behaviour of natural systems such as ant colonies, bee colonies and populations of bacteria, all of which are constituted of a number of entities too simple to understand the whole situation individually. The aim could be described as an attempt to understand how intelligent behaviour of a colony can arise from the actions of a number of relatively simple agents.

In the preface to *Swarm Intelligence* [41] Kennedy and Eberhart make two assertions which indicate the motivation for studying Swarm Intelligence in the context of Intelligent Behaviour. Firstly, *Mind is social*, which demonstrates a belief that an investigation of Swarm Intelligence will provide insight into the behaviour of known intelligent entities such as humans and animals. Secondly, *Particle swarms are a useful computational intelligence (soft computing) methodology*, which demonstrates the belief that practical systems can be developed by using the principles of Swarm Intelligence. These two beliefs, taken together, form a new approach to Artificial Intelligence which is less vulnerable to either of the arguments directed towards the approaches of *Making a Mind*, which was criticised for searching for an architecture for intelligent behaviour that may not exist, and *Modelling the Brain*, which was criticised for potentially offering no insight into the production of intelligent behaviour itself.

Some examples detailed in this section, Ant Colony Optimisation (ACO), Particle Swarm Optimisation (PSO), and Dispersive Flies Optimisation (DFO) use a swarm of interacting agents to implement a search algorithm. Search algorithms are among the most fundamental areas of computer science, the task being to retrieve a designated element from a data structure, referred to in this context as a search space. In many cases this can be achieved by simply iterating over every element in the search space and evaluating whether the element is equal to the target element. This naïve approach is often infeasible, for example when the search space is very large, or when it is computationally expensive to perform the evaluation. In cases where certain features of the task definition are known, significant optimisations can be made. For example, in the case of determining whether a specific number is in a list, if the list is known to be sorted then a binary search can be employed bisecting the area of the search space which remains to be searched at each step. In cases where little or no information is available such as when searching for the maximal output of a chaotic black-box function, there may be no applicable optimisations. In such cases, a random search may be the most suitable approach, as this will evaluate all inputs equally, with no assumptions made as to the best location.

Swarm Intelligence may also provide insights into the behaviour of natural swarms. 'Boids', detailed here is an example of an algorithm which produces natural looking behaviour from simple rules and hence may suggest the method by which the behaviour is performed in nature.

2.3.1 1987 — Boids

Boids is an early computational simulation of the phenomena underlying swarm intelligence, that of complex group behaviour arising from simple agent behaviour. Published by Reynolds in 1987, an apparently natural flocking behaviour can be seen in a group of mobile agents who operate by a few simple rules. The resulting flocking and collision avoidance behaviours can be seen in Figure 2.9, p.59.

Quoting from the original description by Reynolds [80]:

> Stated briefly as rules, and in order of decreasing precedence, the behaviours that lead to simulated flocking are:
>
> - Collision Avoidance: avoid collisions with nearby flockmates
> - Velocity Matching: attempt to match velocity with nearby flockmates
> - Flock Centering: attempt to stay close to nearby flockmates

These rules are simple for an individual with local knowledge to adhere to, but can be seen to produce advanced collective behaviour such as coalescing into a flock, and temporarily separating into two flocks to avoid obstacles or predators. Boids was initially presented at the computer graphics conference SIGGRAPH as a technique for producing realistic animations without explicitly defining the paths of each individual, and is an early example of Swarm Intelligence providing an insight into observed intelligent behaviour.

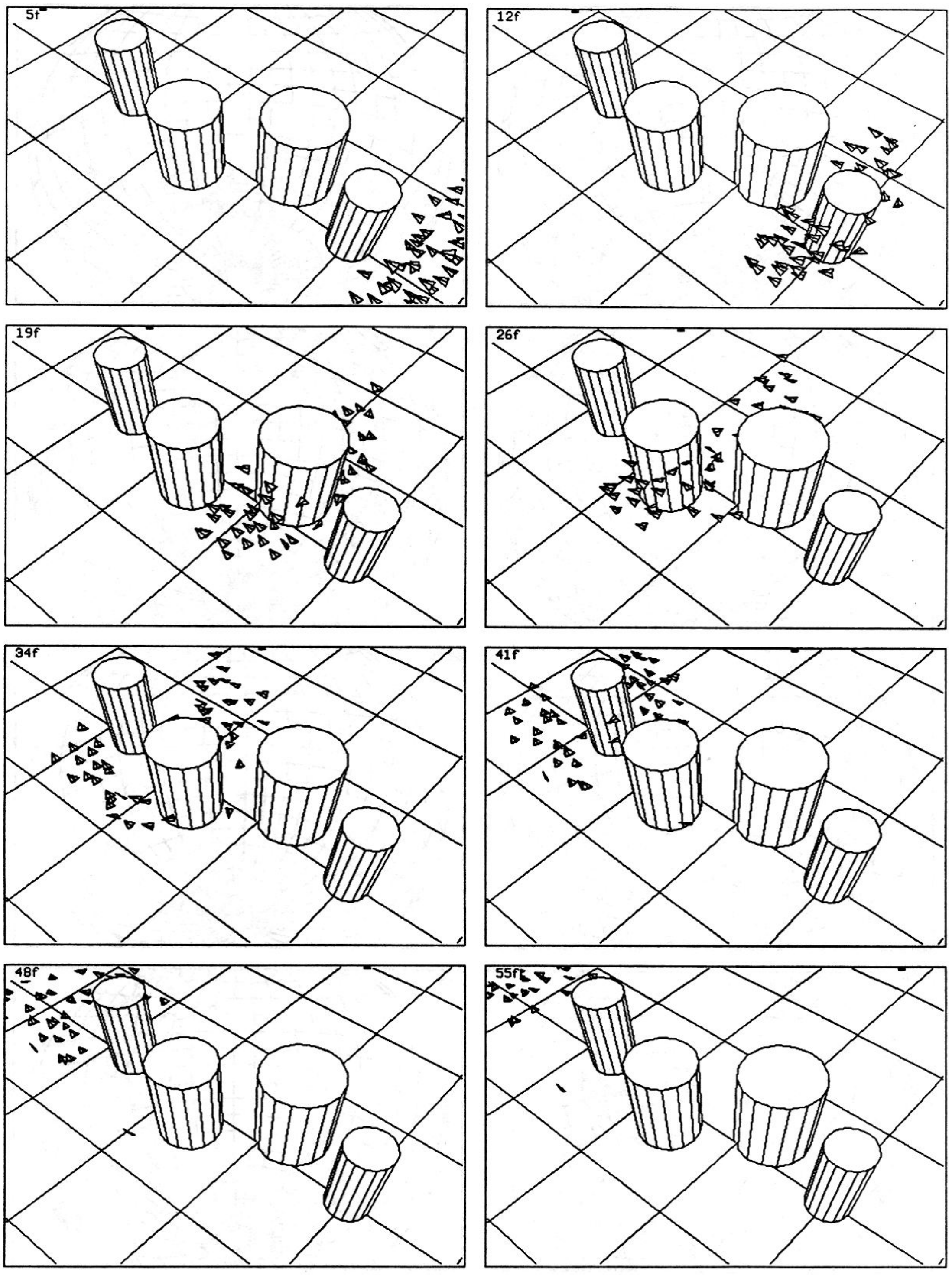


Figure 2.9: A series of frames from an animation depicting A flock of 'Boids' negotiating a field of columns. The image was produced on a Symbolics Lisp Machine in 1986 or 1987. Used with permission of Craig Reynolds, from `https://www.red3d.com/cwr/temp/boids/flock_around_cylinders.jpg`

2.3.2 1992 — Ant Colony Optimisation (ACO)

Ant Colony Optimisation (ACO) (originally called 'the Ant system' [11]) is a swarm intelligence algorithm that is inspired by the foraging behaviour of ants. In ACO, a set of software agents, described as *artificial ants* search for solutions to a given problem, often a combinatorial problem. To apply ACO, the optimization problem is transformed into the problem of finding the best path on a weighted graph. The first application of ACO was to the Travelling Salesman Problem (TSP), where the task is to find the shortest tour, a path which connects all of a set of points distributed in a space. The agents stochastically construct solutions where each element of the solution is influenced by a combination of *a priori* knowledge about the 'desirability' each element and the level of pheromone at that element. In the TSP example each the elements of the solution are the order in which points are visited in the tour, and the *a priori* knowledge is the 'visibility' of each city from each other city calculated as the reciprocal of the distance. A data structure records for each agent the elements which make up its solution, this ensures no agent selects the same element twice in a tour. The exact probability of an agent selecting an edge is the edge's score divided by the score of all edges not yet added to the tour, where the score is calculated by multiplying the desirability associated with that edge raised to the power of a constant parameter β with the amount of pheromone associated with that edge raised to the power of a constant parameter α. Once all agents have generated a solution, the level of pheromone at each edge is calculated by multiplying its current level of pheromone by a constant p in the range of $[0, 1]$ and adding a value which is a function of the score of all solutions which include the edge. In the TSP example, the score of a solution is the reciprocal of the total tour length, weighted by a constant parameter Q. The increase in the amount of pheromone at an edge is analogous to ants leaving a pheromone trail as they perform a tour and leaving a higher concentration of pheromone along shorter tours. The reduction in the amount of pheromone at an edge is analogous to the evaporation of pheromones over time. The result is that the edges which are most likely to be selected are those which combine a low individual cost with a high solution score. This process, shown in algorithm 1 has been shown to produce near optimal solutions to the TSP [19].

A potential disadvantage of ACO is that it requires the selection of six parameters: α, a parameter which weights the importance of the pheromone in the tour construction process; β, a parameter which weights the importance of the 'desirability' in the tour construction process; p, the rate of evaporation of the pheromone; τ_0, the initial amount of pheromone at each edge; $t_{\max}$, a parameter which determines the number of iterations before halting. Q, a parameter which weights the importance of the score of the solution in the calculation of the amount of pheromone at each edge. While some of these parameters are not critical in determining the performance of ACO, the combination of α and β may determine whether the algorithm exhibits one of three behaviours: i) the algorithm finds good solutions, ii) the algorithm does not find good solutions and all agents maintain the same solution, iii) the algorithm does not find good solutions and all agents do not maintain the same solution [19]. Care must therefore be taken to select appropriate parameters for each task.

Algorithm 1 Ant Colony Optimisation

```
1:  function ACO(graph, $\tau_0$, $p$, $\alpha$, $\beta$, $Q$, $t_{\max}$)
2:      $t \leftarrow 0$
3:      for each edge in graph do                                  ▷ initialise pheromone levels
4:          amount of pheromone at edge $\leftarrow \tau_0$
5:      while $t < t_{\max}$ do
6:          for each node in graph do
7:              initialise one ant with a path starting at this node
8:          for each ant do                                        ▷ construct a tour
9:              repeat
10:                 $tail \leftarrow$ last node in path of ant
11:                 unvisited $\leftarrow$ list of nodes not in path of ant
12:                 for each $head$ in unvisited do                ▷ calculate scores of unvisited nodes
13:                     $\tau \leftarrow$ amount of pheromone at edge $(tail \rightarrow head)$
14:                     $d \leftarrow$ distance between $tail$ and $head$
15:                     $\text{scores}_{head} \leftarrow \tau^{\alpha} \times \left(\frac{1}{d}\right)^{\beta}$
16:                 S $\leftarrow \sum_{i=1}^{n} \text{scores}_i$
17:                 probabilities $\leftarrow \frac{\text{scores}_i}{S}$ for $i$ in scores
18:                 probability of selecting node $i$ is $\text{probabilities}_i$
19:                 $n \leftarrow$ node randomly selected from unvisited
20:                 add $n$ to path of ant
21:             until path of ant is a complete tour
22:         for each edge in graph do                              ▷ update pheromone levels
23:             $P_{\text{edge}} \leftarrow$ amount of pheromone at edge
24:             evaporation $\leftarrow P_{\text{edge}} * p$
25:             popularity $\leftarrow 0$
26:             for each ant do
27:                 if edge exists in path of ant then
28:                     $L^k \leftarrow$ length of tour of ant
29:                     popularity $\leftarrow$ popularity + $\frac{Q}{L^k}$
30:             amount of pheromone at edge $\leftarrow$ evaporation $+$ popularity
31:         $t \leftarrow t + 1$
32:         if all ants have the same tour then                    ▷ early halting condition
33:             break
34:     return shortest tour
```

ACO is a stigmergic procedure, where the decisions of agents are guided by modifications made to the search space by other agents, in this case, with pheromone trails. While the exact actions of an agents in ACO are not claimed to be within the capacities of real ants the entire behaviour can be taken as analogous of behaviour which. For example, when constructing a tour an agent in ACO is required to select a edge from a probability distribution constructed by evaluating polynomial expressions, whereas an ant in nature could conceivable achieve a similar behaviour by combining any visible information with pheromonal information of the available options and apply some internal, possible instinctive, bias. ACO therefore demonstrates the possibility of simple agents being able to solve a complex combinatorial problem without ever considering the problem as a whole.

2.3.3 1995 — Particle Swarm Optimisation (PSO)

A subsequently published swarm intelligence algorithm was Particle Swarm Optimisation (PSO) [25]. A swarm of agents, named *particles*, are distributed through a search space and evaluate the solution represented by their current location. The initial locations of the agents are chosen uniformly randomly or with influence from external domain knowledge, but subsequently chosen by moving the agents in a direction which is a combination of its previous movement and information derived from the swarm as a whole. The broad principles of operation can be illustrated by the following metaphor [9].

> Imagine that one dips a finger in a jar of honey and uses that honeyed-finger to trace a line in the air in the presence of a swarm of bees. Fairly soon the bees will begin to buzz around the honeyed finger tip. By using this swarm we can generate a new series of points (defining a new line) simply by noting the centroid of this swarm at successive instants in time.

In this metaphor line defined by the centroid of the swarm at successive instants in time provides an estimate of the position of the honeyed finger over time. For this behaviour to emerge all that needs to happen is for each bee to move as a function of two distances. These distances are: the distance from the bee to the position representing the closest the bee has ever been to the honey, and the distance from the bee to the position representing the closest any bee has ever been to the honey. This is an efficient way of guiding a search through a space without depending on expert domain knowledge being encoded into the implementation. In contrast with Boids, PSO is a useful search algorithm but is unlikely to provide an insight into the behaviour of natural swarms as all agents need to have access to the swarm's 'global best' fitness.

Algorithm 2 PSO

```
 1: Initialise agents
 2: while not halting condition do
 3:     for each particle in swarm do
 4:         fitness ← evaluate fitness of particle                ▷ Larger is better
 5:         if fitness > particle's personal best then
 6:             particle's personal best ← fitness
 7:             if particle's personal best > swarm's best then
 8:                 swarm's best ← particle's personal best
 9:         Update particle velocity                            ▷ Equation 2.2, p.63
10:         Update particle position                            ▷ Equation 2.1, p.63
```

The PSO evolution functions define each agents subsequent position x_{t+1} (Equation 2.1) and subsequent velocity v_{t+1} (Equation 2.2). w is the constant inertia weight, c_1 and c_2 are the cognitive and social learning parameters respectively, r_1 and r_2 are randomly generated numbers between 0 and 1, p_t^i is the particle's personal best position and p_t^g is the swarm's best position.

$$x_{t+1} = x_t + v_{t+1} \tag{2.1}$$

$$v_{t+1} = wv_t + c_1 r_1 (p_t^i - x_t) + c_2 r_2 (p_t^g - x_t) \tag{2.2}$$

2.3.4 2014 — Dispersive Flies Optimisation (DFO)

Inspired by the behaviours of flies hovering over food sources, Dispersive Flies Optimisation (DFO) [81] exhibits a swarming behaviour similar to PSO, but with the extra feature of swarm dispersal, analogous to when natural swarms are dispersed by the presence of a threat, as in nature the swarm will reappear at the original position as the threat moves away, unless a preferred location is found in the interval. To simulate this, in each iteration any component of a fly's position may be reset with a fixed probability, known as the *disturbance threshold*. This guarantees a proportionate disturbance from a solution and hence avoids the swarm becoming trapped in a local minima.

In DFO each individual fly updates its position by updating each dimensional component in turn, the update is calculated as the distance to the highest scoring of a fly's two neighbouring flies plus a portion of the distance to the highest scoring fly in the whole swarm, the portion being a randomly selected value in the interval $[0, 1]$. As the magnitude of any updates are relative to the distances between flies, the algorithm naturally begins with an emphasis on exploration of the search space. As the swarm begins to converge around a single location, and the distances between the flies decrease, the algorithm begins a more exploitative phase as a promising solution is more precisely optimised.

A feature of DFO is that it is a minimalistic algorithm, in this sense it has been designed to have the smallest number of tunable parameters and the smallest number of components [83]. Other than the population size, DFO has one tunable parameter, the disturbance threshold. Additionally, other than the index of the best fly, each fly consists of merely its position vector along with the index of the best neighbouring fly.

2.4 Summary

The project of building machines with diverse behaviour has made significant progress since ancient times, from fully defined Clepysdra, though configurable mechanisms, and reaching maturity with Turing's model of computation which enabled the study of Artificial Intelligence. AI itself has made significant progress but has been characterised as mostly involving projects which can be described as Making a Mind or Modelling the brain. The approach of making a mind is characterised as the use of symbolic logic to perform explicitely defined tasks. It has been criticised as the symboloc logic required appears to become vastly more complicated as the requirements of the task become larger, and less well defined. Modelling the brain is characterised by Connectionist architectures and neural networks, which have also demonstrated great abilities and a certain level of genralisation or adapability. It was criticised on the grounds that even a successful project may lack any insight into the capacities which made a system adaptive and intelligent.

Swarm intelligence is identified as being able to be described as combining both approaches. Father than fully defining behaviour of a system behaviour with an algorithm of symbolic logic, or fully relying on emergent behaviour from a network of reactive elements, swarm intelligence defines a population of interacting agents, each controlled by an internal algorithm, but an algorithm which determines behaviour as the result of interactions within the swarm.

This implies that swarm intelligence will be susceptible to the criticisms of both making a mind and modelling the brain, but by being fully defined by neither approach, they may be mitigated somewhat.

We will investigate an example of swarm intelligence which has simple agents performing arbitrary symbolic logic, but which uses the interaction of the simple agents to gain determine the behaviour, we will also see the disadvantages that are present with this approach. We will also see the analogies between its operation and natural behaviour which, if strong, will lend some evidence towards there being value in the method.

Chapter 3

Stochastic Diffusion Search (SDS)

Algorithm 3 SDS as introduced by Bishop (1989)

```
INITIALISE (mappings);
REPEAT
    TEST (mappings);
    DIFFUSE (mappings);
UNTIL TERMINATION;
```

3.1 Introduction to SDS

First described in 1989 by J.M. Bishop, Stochastic Diffusion Search (SDS) was introduced as an evolution of the study of Hinton mapping [7] (Section 2.2.3.6, p.50). Where Hinton mapping identifies objects by testing for all possible combinations of stimuli, SDS stochastically tests certain hypotheses against partial combinations of stimuli, and subsequently performs a diffusion process where more promising hypotheses are more likely to be tested against further features. At the time, and at least as late as 1998 [67], the term "Stochastic Search Networks" was used, but SDS will be used throughout.

SDS has a number of important features, including; robustness in the presence of noise; a simplicity that lends itself well to mathematical modelling; and stable global state emerging from the stochastic behaviour of individuals. Its robustness make SDS particularly suited to real world tasks which may be noisy, and may change over time. The relative ease of modelling SDS mathematically has lead to the ability to predict a number of aspects of its behaviour. The stability of SDS means it may also be an important model for methods of guiding intelligent behaviour without a central executive control.

The operation of SDS is most intuitively explained by analogy, a number of which have been used, most commonly *The Restaurant Game* [15], and more recently *The Mining Game* [86]. These are described below, followed by the pseudocode of a single iteration of SDS.

3.1.1 The Restaurant Game

The Restaurant Game, reproduced here from 2003 [15], describes how a group of people may employ SDS to perform a search for the best restaurant in an unfamiliar town.

> A group of delegates attends a long conference in an unfamiliar town. Each night they have to find somewhere to dine. There is a large choice of restaurants, each of which offers a large variety of meals. The problem the group faces is to find the best restaurant, that is the restaurant where the maximum number of delegates would enjoy dining. Even a parallel exhaustive search through the restaurant and meal combinations would take too long to accomplish. To solve the problem delegates decide to employ a Stochastic Diffusion Search.
>
> Each delegate acts as an agent maintaining a hypothesis identifying the best restaurant in town. Each night each delegate tests his hypothesis by dining there and randomly selecting one of the meals on offer. The next morning at breakfast every delegate who did not enjoy his meal the previous night, asks one randomly selected colleague to share his dinner impressions. If the experience was good, he also adopts this restaurant as his choice. Otherwise he simply selects another restaurant at random from those listed in 'Yellow Pages'.
>
> Using this strategy it is found that very rapidly [a] significant number of delegates congregate around the best restaurant in town.

The process of the restaurant game has a number of notable features. Within minimal centralised control the group of delegates acts together to solve a problem that could not be quickly solved by an individual. The delegates will efficiently move to the next best restaurant if the current one has a significant drop in standards or closes down entirely. The restaurants, the menus, or the individual meals need to be directly comparable, all that is required is for each agent to decide for themself whether their experience was good. Delegates will find themselves enjoying many evenings in a relatively high quality restaurant long before all of the meals in all of the restaurants in town could have been evaluated.

This analogy has been criticised on the grounds that delegates are likely to have differing dining preferences and hence it is possible for a delegate to locate a restaurant in which they enjoy all of the meals on offer, but which is unsatisfying to all other delegates. In the case where only one delegate, or a small proportion of the group, remains permanently at such a restaurant the rest of the group will proceed largely as usual and so the majority will still converge on the best restaurant. Taken to the extreme, however, all agents may find themselves dining alone, even when there exists a single superior restaurant which would satisfy the largest portion of the delegates. This superior restaurant would never be located as all delegates are satisfied with the meals at their current restaurant, and hence never select a new restaurant. This critique led to the development of *The Mining Game*, which depends on the less subjective notion of locating gold rather than dining preferences.

3.1.2 The Mining Game

Originally defined in 2010 [85] the Mining Game uses an analogy of searching a landscape for the best mining prospect. A slightly improved version from 2013 [84] is quoted below.

> A group of friends (miners) learn that there is gold to be found on the hills of a mountain range but have no information regarding its distribution. On their maps the mountain range is divided into a set of discrete hills and each hill contains a discrete set of seams to mine. Over time, on any day the probability of finding gold at a seam is proportional to its net wealth.
>
> To maximise their collective wealth, the miners need to identify the hill with the richest seams of gold so that the maximum number of miners can dig there (this information is not available a-priori). In order to solve this problem, the miners decide to employ a simple Stochastic Diffusion Search.
>
> - At the start of the mining process each miner is randomly allocated a hill

to mine (his hill hypothesis, h).

- Every day each miner is allocated a randomly selected seam on his hill to mine.
- At the end of each day, the probability that a miner is happy[1] is proportional to the amount of gold he has found.
- At the end of the day the miners congregate and over the evening each miner who is unhappy selects another miner at random to talk to. If the chosen miner is happy, he happily tells his colleague the identity of the hill he is mining (that is, he communicates his hill hypothesis, h, which thus both now maintain). Conversely, if the chosen miner is unhappy he says nothing and the original miner is once more reduced to selecting a new hypothesis — identifying the hill he is to mine the next day — at random.

In the context of SDS, agents take the role of miners; active agents being 'happy miners', inactive agents being 'unhappy miners' and the agent's hypothesis being the miner's 'hill-hypothesis'. It can be shown that this process is isomorphic to SDS, and thus that the miners will naturally self-organise and rapidly congregate over hill(s) on the mountain range with a high concentration of gold.

The happiness of the miners can be measured probabilistically, or represented with an absolute boolean value, so long as each miner is either happy or unhappy by the end of each day [82]. Furthermore, if the gold is modelled as a finite resource, reducing over time, then the search is sufficiently adaptive that miners change where they congregate as the location with the most gold changes.

3.1.3 Pseudocode

In this example, agents have access to a function for randomly selecting a hypothesis for the set of all possible hypotheses, and for randomly selecting a microtest from a given set. Each microtest evaluates a specific part of a hypothesis and returns a boolean value indicating whether or not the test passed.

[1]Though 'happy' is a term that is similarly subjective as one's dining preferences, in this case it is used in an objective sense. All miners share an identical process whereby the amount of gold they locate on a single

Algorithm 4 A single iteration of standard SDS

```
 1: for each agent in swarm do                                  ▷ Diffusion phase
 2:     if agent is inactive then
 3:         polled ← randomly selected agent in swarm
 4:         if polled is active then
 5:             agent assumes the hypothesis of polled
 6:         else
 7:             agent randomly selects a new hypothesis

 8: for each agent in swarm do                                  ▷ Test phase
 9:     microtest ← randomly selected microtest
10:     agent performs the microtest against their hypothesis   ▷ Partial evaluation
11:     if the microtest passes then
12:         agent becomes active
13:     else
14:         agent becomes inactive
```

To recap definitions. An *agent* is an individual member of a swarm. A *swarm* is a collection of agents. A *search space* is the set of all possible hypotheses. A *hypothesis* is an element of the search space, or an index referring to an element of the search space. A *microtest* is a function which partially evaluates a hypothesis. The *score* of a hypothesis is equal to the probability of an agent with that hypothesis being active after the test phase. A full glossary is available in Chapter 6, p.174.

SDS lends itself well to mathematical analysis. In the next section a simple but powerful model of the convergence behaviour of SDS is described. The features of SDS which have previously been modelled and the insights into its behaviour using this model are then described. Following this a formalism for describing variants of SDS is developed and some of the advantages and disadvantages of all the main variants which have been published are identified.

day defines a probability that the miner will declare themselves 'happy' at the end of the day when miners congregate to potentially share the identity of the hills they are mining.

3.2 Mathematical analysis of SDS

To appreciate the mathematical analysis of SDS which has been published it is required that first the model is described.

This analysis will always assume the search space contains exactly one hypothesis with a score larger than any other hypothesis, called the optimal hypothesis, and many hypotheses with a significantly lower score, the collective effect of which can be described by a single value [63, p.118] known collectively as homogeneous background noise. Furthermore the search space size, the number of hypotheses in a search space, is represented by the symbol S. The swarm size, the number of agents in a swarm, is represented by the symbol A. The proportion of the swarm which are active at a given time, is known as the global activity.

3.2.1 Discrete Dynamical Systems model

Introduced in [60] and explored in [62], the Discrete Dynamical System Model (DDSM) of SDS has been used to clearly express results previously reached via other methods as well as new models of the behaviour of SDS. Similar to previous models, an assumption is made of homogeneous background noise. The DDSM introduces the following notation.

α the score of the optimal hypothesis

β the mean probability of an agent at any non-optimal hypothesis becoming active

$\bar{c}_i$ the mean number of active agents maintaining the optimal hypothesis as a proportion of the total population

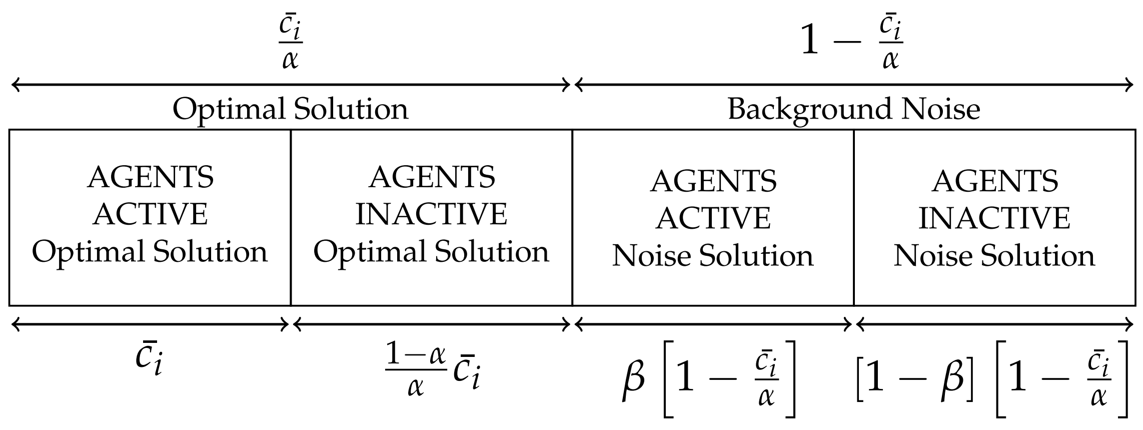
From these definitions, α and β is holds that $0 \leq \beta < \alpha \leq 1$, and the following expressions can be immediately derived; as represented in Figure 3.1.

$\frac{\bar{c}_i}{\alpha}$ Proportion of active and inactive agents at the optimal hypothesis

$\frac{1-\alpha}{\alpha}\bar{c}_i$ Proportion of inactive agents at the optimal hypothesis

$1 - \frac{\bar{c}_i}{\alpha}$ Proportion of active and inactive agents at any noise hypothesis

$\beta\left(1 - \frac{\bar{c}_i}{\alpha}\right)$ Proportion of active agents at any noise hypothesis



$\frac{\bar{c}_i}{\alpha}$
$1 - \frac{\bar{c}_i}{\alpha}$
Optimal Solution
Background Noise
AGENTS ACTIVE Optimal Solution
AGENTS INACTIVE Optimal Solution
AGENTS ACTIVE Noise Solution
AGENTS INACTIVE Noise Solution
$\bar{c}_i$
$\frac{1-\alpha}{\alpha}\bar{c}_i$
$\beta\left[1 - \frac{\bar{c}_i}{\alpha}\right]$
$[1-\beta]\left[1 - \frac{\bar{c}_i}{\alpha}\right]$

Figure 3.1: Myatt's discrete dynamical systems model of SDS.

$(1-\beta)\left(1-\frac{\bar{c}_i}{\alpha}\right)$ Proportion of inactive agents at any noise hypothesis

The properties of SDS which have been modelled can be split into three categories: the convergence criteria, which describe the set of parameters over which SDS can be expected to converge; the convergence time, which measures the mean number of iterations before a cluster forms at the optimal hypothesis; and the steady state, which describes the number of agents at the optimal hypothesis after convergence.

3.2.2 Minimum convergence criteria

Using a Markov Chain model it was shown that for a stable cluster to form it must hold that

$$2\alpha(1-p_\alpha) < 1 \tag{3.1}$$

where p_α is the probability of selecting the optimal hypothesis at random [63, p.72]. As the search space is often large, p_α can be assumed to be small, in which case one obtains

$$\alpha \geq \frac{1}{2} \tag{3.2}$$

A similar result was shown using a probabilistic model [31], the minimum value of optimal hypothesis score at which a stable cluster will form in the presence of homogeneous background noise was shown to be

$$\alpha_{\min} = \frac{1}{2-\beta} \tag{3.3}$$

This result was confirmed using the DDSM. The mean number of inactive agents adopting the optimal hypothesis during the diffusion phase is described by the function $g(\alpha, \beta, \bar{c}_i)\bar{c}_i$ which yields the number of inactive agents for a given iteration. From Figure 3.1 g can be written as

$$g(\alpha, \beta, \bar{c}_i) = \frac{1-\alpha}{\alpha}\bar{c}_i + (1-\beta)(1-\frac{\bar{c}_i}{\alpha}) \tag{3.4}$$

Therefore, the function f which defines the one-step evolution function is

$$\begin{aligned} \bar{c}_{i+1} &= f(\alpha, \beta, \bar{c}_i) \\ &= \alpha(\bar{c}_i + g(\bar{c}_i, \alpha, \beta)\bar{c}_i) \\ &= \bar{c}_i(\beta\bar{c}_i + 2\alpha - \alpha\bar{c}_i - \alpha\beta) \end{aligned} \tag{3.5}$$

For SDS to converge the repeated iteration of f must stabilise at a non-zero value. It therefore must hold that $\frac{df}{d\bar{c}_i} > 1$ when $\bar{c}_i = 0$. Differentiating f by $\bar{c}_i$ and solving for α when $\frac{df}{d\bar{c}_i} > 1$ yields α_{min}.

$$\alpha_{\text{min}} = \frac{1}{2-\beta} \tag{3.6}$$

It has been shown experimentally that an SDS with a value of α such that $\alpha = \alpha_{\text{min}} + 0.01$ would reliably converge, and if $\alpha = \alpha_{\text{min}} - 0.01$ there would be no stable clusters [60]. Hence, the experimental value of α_{min} was shown to be within ± 0.01 of the theoretical, validating the theory.

An advantage of the homogeneous background noise model is that an estimate of β can be achieved by taking the mean value of activity over repetitions of the test phase. This value can be substituted into equation 3.6 to provide a lower bound for convergence.

3.2.2.1 Robustness

By taking all combinations of values of α and β, the minimum convergence criteria can be used to identify each combination as falling into one of three categories; they are (i) invalid search spaces, where $\beta \geq \alpha$, ii) search spaces that successfully converge as $\alpha > \alpha_{\text{min}}$ and iii) search spaces that do not converge [62]. The proportion of valid search spaces for which standard SDS will successfully converge is named the robustness (ζ) and has been calculated to be approximately 0.614 [62, p.54].

$$\zeta = 2(1 - \ln 2) \approx 0.614 \tag{3.7}$$

3.2.3 Convergence time

The convergence of SDS can be described in three phases. In the first phase few agents are active and many new hypotheses are generated at random each iteration, the number of iterations spent in this phase is described by the time to first hit (Section 3.2.3.1). In the second phase a cluster forms once an agent has assumed an appropriate hypothesis, this phase is relatively short as it is dominated by the positive feedback effect of the diffusion (Section 3.2.3.2). In the third phase, after a stable cluster has formed many agents are active and few new hypotheses are selected, this phase can be practically indefinite, owing to the very high probability of a cluster persisting (Section 3.2.3.3). A Markov Chain model was used to show that the convergence time increases dramatically when there was a small relative discrimination between α and β [64].

3.2.3.1 Time to first hit (TH)

The time to first hit (TH) describes the number of iterations after which the probability that the optimal hypothesis has been located reaches a given probability p. In cases where $\beta = 0$ the time to first hit has been shown to be [6, 67]

$$\mathrm{TH} = \frac{\ln(1-p)}{A \ln(\frac{S-1}{S})} \tag{3.8}$$

In cases where $\beta > 0$, the time to first hit will be larger as some agents will be maintaining suboptimal hypotheses and hence will not generate new random hypotheses in the diffusion phase the time to first hit has therefore been shown to be calculated as [61]

$$\mathrm{TH} = S \frac{\ln \frac{1}{p}}{A(1-\beta)^2} \tag{3.9}$$

which is linear with respect to S [61, 6].

3.2.3.2 Positive feedback effect of cluster formation

Once an agent has assumed the optimal hypothesis a process of positive feedback occurs. This occurs because the most important factor in the probability of an agent assuming the optimal hypothesis is the number of agents currently maintaining that hypothesis. As the number of agents maintaining the optimal hypothesis increases, the rate at which new agents assume that hypothesis also increases, so long as there are sufficient inactive agents remaining. This very quickly reaches a steady state in which the number of agents which were maintaining the optimal hypothesis becoming inactive equals the number of agents assuming the optimal hypothesis in the diffusion phase [66]. This phase lasts for a relatively small number of iterations when compared to the time to first hit which is a linear with respect to search space size and the steady state which will be seen to be practically indefinite.

3.2.3.3 Stability of convergence

The stability of convergence may describe the mean number of iterations for which a cluster can be expected to persist once it has formed as well as the property of SDS where clusters are able to form at the optimal hypothesis in the presence of other hypotheses that have similar scores.

Strong convergence criterion It has been proved mathematically that in cases where $\beta = 0$, $\alpha = 1$ and at least one agent is maintaining the optimal hypothesis, that the probability of all agents maintaining that hypothesis approaches 1 [8][63, p.56][62, p.63]. These conditions taken as together define the strong convergence criterion.

In cases where $\alpha < 1$, SDS will not strongly converge and may weakly converge [64]. Weakly converging mean that SDS is only ever statistically stable, there is always a non-zero probability that the swarm will transition into any state from any other state. This means any apparently stable cluster may collapse at any iteration, though it is likely to exist long enough to be detected. An advantage of this behaviour is that an apparently stable cluster will collapse once a superior hypothesis has been located, hence SDS can be applied successfully to problem spaces that change over time.

The stability of the converged state of SDS was derived using a Markov Chain model [65]. Using some selected values $A = 1000, p_\alpha = 0.001, \alpha = 0.8, \beta = 0$ the mean number of iterations before a stable cluster will collapse (m_0) was calculated to be

$$m_0 \approx \frac{1}{(0.25)^{1000}} \propto 10^{602}$$

thus this state is visited very rarely and as such the search is very stable. Even when $A = 10$ $m_0 \propto 10^6$.

As the stability of the search depends exponentially on the number of agents in the swarm, the ratio of logarithms of mean return times to the 'all-inactive' state for a pair of instances of SDS is equal to the ratio of the numbers of agents involved in the search. Hence, the number of agents in a swarm may efficiently control the stability of solutions in practical applications [65].

It has been stated that the rate of convergence is geometric "for some constant" [64]. The DDSM has been used to show the constant to be

$$\alpha(2 - \beta) \tag{3.10}$$

as such, it can be observed that an increase in α will reduce convergence time more than decreasing β by the same amount [62].

3.2.4 Sensitivity of convergence

A distractor hypothesis is a single hypothesis with a score (δ) such that $\beta < \delta < \alpha$. Using an Ehrenfest urn model of SDS it has been shown that even in the presence of a strong distractor hypothesis that there is a significant increase in the order of magnitude of the probability of an agent maintaining the optimal hypothesis against that of an agent maintaining the distractor hypothesis [p.101][63]. In a three-dimensional plot of the optimal hypothesis score (α) and the distractor hypothesis score (δ) against the probability of an agent maintaining the optimal hypothesis three distinct surfaces of equal probability were visible. One surface represented the case when δ was larger than α and so can be ignored by definition ($\alpha > \delta$). The second surface occurred at $\delta < 0.5$ and $\alpha < 0.5$ which represented that an agent was equally likely to maintain either hypothesis when both hypotheses have a score that is too small to form a cluster. The third surface occurred when $\alpha > \max(\delta, 0.5)$ which represented when optimal hypothesis had a score which could form a cluster.

An important feature of these surfaces was the steep transition between them, this represented that SDS was very sensitive to differences between α and δ and so an agent would most likely be found at the optimal hypothesis even when δ was within 1% of α. Another important feature of these surfaces was their flatness which described that the probability of convergence was not affected by the presence of the distractor hypothesis, only the ability of the optimal hypothesis to form a cluster. This means that SDS is particularly insensitive to signal to noise ratio, in cases where the signal (the optimal hypothesis), is strong enough.

3.2.5 Steady state resource allocation

In the case where $\beta = 0$ and $\alpha < 1$, a model of the cluster size mean and standard deviation [66]. The mean number of active agents ($E[n]$) is

$$E[n] = A\pi_1 \tag{3.11}$$

and the standard deviation is

$$\sigma = \sqrt{A\pi_1\pi_2} \tag{3.12}$$

where

$$\pi_1 = \frac{\pi_3 - 1 + \sqrt{[\pi_3 - 1]^2 + 2\pi_3(1-\beta)p_\alpha}}{\pi_3} \tag{3.13}$$

$$\pi_2 = \frac{1 - \sqrt{[\pi_3 - 1]^2 + 2\pi_3(1-\beta)p_\alpha}}{\pi_3} \tag{3.14}$$

$$\pi_3 = 2\alpha(1 - p_\alpha) \tag{3.15}$$

It has also been shown that the attractors are never strange attractors for valid values of α and β. This proves that the convergence behaviour is not chaotic and hence the level of activity in SDS will always tend towards a constant level [62].

For cases where $\alpha > \alpha_{\text{min}}$ the mean stationary state will be [62]

$$\gamma = \frac{\alpha(2-\beta)-1}{\alpha-\beta} \tag{3.16}$$

3.2.5.1 Estimating optimal hypothesis score from steady state

The value for α can be estimated by dividing the number of active agents at the largest cluster by the sum of active and inactive agents at the largest cluster [62].

$$\widehat{\alpha} = \frac{c_{\text{active}}}{c_{\text{active}} + c_{\text{inactive}}} \tag{3.17}$$

This is notable as this expression does not depend on the level of homogeneous background noise.

It is also observed that the same estimate can be achieved using the global activity, and not just the activity at the largest cluster. Given the stationary state for the optimal cluster (Equation 3.16) and Figure 3.1, the stationary state for all active agents must be equal to

$$\begin{aligned}\gamma' &= \gamma + \beta\left[1 - \frac{\gamma}{\alpha}\right] \\ &= 2 - \frac{1}{\alpha}\end{aligned} \tag{3.18}$$

thus α can be estimated by

$$\widehat{\alpha} = \frac{1}{2-\widehat{\gamma}} \tag{3.19}$$

3.3 A formal specification for SDS

Given the initial description of SDS (Algorithm 3, p.66) each line may be considered as indicating potential for variation. Some of these dimensions of variation are relevant only to the specific implementation and do not pertain to the variant themselves. All remaining dimensions are described and their major characteristics identified. As SDS can be modified in a practically unlimited number of ways a strict definition of what is and is not a variant cannot be given, instead a specification which succinctly describes the majority of the published variants is presented. Where any variant has been mathematically analysed, it is presented here.

3.3.1 Areas of potential variation in SDS

In the original description (Algorithm 3, p.66) SDS is defined in five lines. Using these five lines and following the framework defined in [16], the main areas of variation of SDS can be identified. The first of these areas is *initialisation* which is concerned with variations in the initial hypotheses and activities of agents in the swarm. Second, the text `REPEAT` indicates variations in the *iteration* of the test phase and diffusion phase. Third, there are variations in the *test phase*, where hypotheses are partially evaluated. Fourth, there are variations in the *diffusion phase*, where new hypotheses are selected and potentially distributed amongst the swarm. Fifth, there are variations in the *termination*, or *halting* conditions, in which the aim is to halt the procedure once SDS has converged to a solution. Finally, one more area of variation has been identified which is the method of *extraction*, where a solution is indicated by the formation of clusters [16, p. 117]. In order to systematically distinguish and compare the most significant variations of SDS a formalism is developed. A variant of SDS will be defined as a combination of its modes of iteration, diffusion, test and halting.

3.3.2 Formalism

An instance of SDS can be described as a combination of its mode of iteration (I) and halting (H). Before each iteration the halting function is evaluated and the process continues to perform iterations until the halting function evaluates as *true*. I defines the mode of iteration, each evaluation of the I function will count as one iteration.

Algorithm 5 SDS as formalised

```
1: function SDS($I$, $H$)
2:     function SDS′( )
3:         while not $H$() do
4:             $I$()
5:     return SDS′
```

Mode of iteration — I The mode of iteration in the original description of SDS is known as synchronous operation. A single iteration is defined as having passed once all agents perform the diffusion phase followed by all agents performing the test phase. This mode of iteration, named $I^{\text{SYNCHRONOUS}}$ can be seen in 6. The definition of the mode of iteration therefore requires the parameters D to define the mode of diffusion, T to define the mode of testing, and a reference to the swarm. The returned function has no arguments and each call effects a single iteration on the swarm. Algorithm 6 shows synchronous iteration which defines the mode of iteration of standard SDS, for the reasons explained in section 3.3.4.1, the diffusion phase is performed before the test phase.

Algorithm 6 $I^{\text{SYNCHRONOUS}}$ — Synchronous iteration

```
1: function $I^{\text{SYNCHRONOUS}}$($D$, $T$, swarm)
2:     function $I'$( )
3:         for each agent in swarm do
4:             $D$(agent)                                  ▷ Diffusion phase
5:         for each agent in swarm do
6:             $T$(agent)                                  ▷ Test phase
7:     return $I'$
```

Mode of diffusion — D The mode of diffusion is the mechanism by which agents share hypotheses or select new hypotheses, each search space therefore requires a unique method by which hypotheses are selected. The definition of the mode of diffusion therefore requires the parameter DH which defines a function for selecting new hypotheses, and a reference to the swarm. The returned function takes an agent as an argument and each call effects the mode of diffusion on that agent. Algorithm 7 shows passive diffusion which relies on the inactive agents requesting hypotheses from active agents and is the mode of diffusion of standard SDS.

Mode of hypothesis selection — DH The hypothesis selecting function is the method by which agents select new hypotheses. The definition of DH therefore requires the parameter *hypotheses* which represents the set of all the possible hypotheses. When implementing this

Algorithm 7 D^{PASSIVE} — Passive diffusion

```
1: function $D^{\text{PASSIVE}}(DH, \text{swarm})$
2:     function $D'$(agent)
3:         if agent is inactive then
4:             polled ← random agent in swarm
5:             if polled is active then
6:                 hypothesis of agent ← hypothesis of polled
7:             else
8:                 hypothesis of agent ← $DH()$
9:     return $D'$
```

function the set of all possible hypotheses may be fully enumerated, or selected as required by an underlying function. Where full enumeration is likely to require less computational resources it will likely require more memory and conversely selecting hypotheses from a function will likely require more computational resources but less memory. The returned function takes no arguments and each call returns a randomly selected hypothesis.

As originally introduced [6], the standard method for selecting hypotheses is to uniformly randomly distribute new hypotheses over the search space. This does not require any specific a priori knowledge of the task, and will evenly distribute the resources through the search space. This technique does not modify its behaviour as a result of previous activity and so has the advantage of never being trapped indefinitely in locally optimal areas of the search space, but has the disadvantage of not responding to potentially helpful information. Algorithm 8 shows uniformly random hypothesis selection.

Algorithm 8 DH^{UNIFORM} — Uniformly random hypothesis selection

```
1: function $DH^{\text{UNIFORM}}$(hypotheses)
2:     function $DH'$( )
3:         hypothesis ← element selected uniformly at random from hypotheses
4:         return hypothesis
5:     return $DH'$
```

Mode of testing — T The mode of testing for each agent involves selecting a microtest, using that microtest to perform a partial evaluation of their hypothesis and updating their activity correspondingly. The definition of the mode of testing therefore requires the parameter TM which defines a function for selecting a microtest. The returned function takes an agent as an argument and each call effects the mode of testing on that agent. Algorithm 9 shows boolean testing which defines the mode of testing of standard SDS, it is important to note that while the mechanism of boolean testing is simple, it requires that a method of evaluating a hypothesis be defined as a set of boolean functions. This is a significant requirement as the solutions to many search tasks cannot be evaluated with a set of boolean functions, or the functions themselves may include the use of predetermined thresholds.

Algorithm 9 T^{BOOLEAN} — Boolean testing

```
1: function $T^{\text{BOOLEAN}}(TM)$
2:     function $T'$(agent)
3:         microtest ← $TM()$
4:         partial evaluation ← microtest(hypothesis of agent)    ▷ partial evaluation $\in \mathbb{B}$
5:         if partial evaluation = true then
6:             agent becomes active
7:         else
8:             agent becomes inactive
9:     return $T'$
```

Microtest selection — TM The microtest selecting function is the method by which agents select how they will partially evaluate their hypothesis. The definition of TM therefore requires the parameter *microtests* which represents the set of all possible microtests. As with the *hypotheses* parameter to DH the set of microtests may be fully enumerated or selected as required, however the number of microtests is often to be small compared to other features of a search space and so full enumeration is more common. The returned function takes no arguments and each call returns a randomly selected microtest. Algorithm 10 shows uniformly random microtest selection which defines the microtest selection of standard SDS.

Algorithm 10 TM^{UNIFORM} — Uniform microtest selection

```
1: function $TM^{\text{UNIFORM}}$(microtests)
2:     function $TM'$( )
3:         microtest ← element selected uniformly at random from microtests
4:         return microtest
5:     return $TM'$
```

Mode of halting — H The halting function is the method by which an instance of SDS determines whether or not to continue to call I and hence perform further iterations. A halting function can therefore be defined as a function which takes no parameters and returns a boolean value.

$$H = f : () \to \mathbb{B} \tag{3.20}$$

There are many different halting functions, a very simple example is to halt after a predetermined number of iterations. This method, called fixed iterations halting, requires the parameter *maximum iterations* to define the maximum number of iterations that will elapse. The returned function takes no arguments and will return *false* until it has been called a number of times greater than *maximum iterations*. Other halting functions perform more complicated operations over the current and previous states of the swarm and so require a more complicated set of parameters. Algorithm 11 shows fixed iterations halting.

Algorithm 11 H^{FIXED} — Fixed iteration count halting

```
1: function $H^{\text{FIXED}}$(maximum iterations)
2:     iteration count $\leftarrow 0$
3:     function $H'()$
4:         iteration count $\leftarrow$ iteration count + 1
5:         halt $\leftarrow$ iteration count > maximum iterations          ▷ halt $\in \mathbb{B}$
6:         return halt
7:     return $H'$
```

3.3.3 Standard SDS (SSDS)

Standard SDS can be therefore denoted as in equation 3.21.

$$\text{SDS}^{\text{STANDARD}} = \text{SDS}\left(I^{\text{SYNCHRONOUS}}\left(D^{\text{PASSIVE}}(DH^{\text{UNIFORM}}), T^{\text{BOOLEAN}}(TM^{\text{UNIFORM}})\right), H^{\text{FIXED}}\right) \tag{3.21}$$

To simplify the description of further variants, all the features of standard SDS will be denoted as standard forms as this allows further variants to be defined in terms of which features differ from standard SDS and which features remain unchanged. These standard features are I^{STANDARD}, D^{STANDARD}, DH^{STANDARD}, T^{STANDARD}, and TM^{STANDARD} and can be seen in equations 3.22–3.26.

$$I^{\text{STANDARD}} = I^{\text{SYNCHRONOUS}}(D^{\text{STANDARD}}, T^{\text{STANDARD}}) \tag{3.22}$$

$$D^{\text{STANDARD}} = D^{\text{PASSIVE}}(DH^{\text{STANDARD}}) \tag{3.23}$$

$$DH^{\text{STANDARD}} = DH^{\text{UNIFORM}} \tag{3.24}$$

$$T^{\text{STANDARD}} = T^{\text{BOOLEAN}}(TM^{\text{STANDARD}}) \tag{3.25}$$

$$TM^{\text{STANDARD}} = TM^{\text{UNIFORM}} \tag{3.26}$$

3.3.4 Omissions of the formalism

In the formalism defined in section 3.3.2, certain aspects of a variant of SDS are not defined. The initialisation behaviour of a variant is omitted as the effect of initialisation is similar to the effect of the diffusion phase (as described in section 3.3.4.1 below) and hence any duplication in description is avoided. The set of microtests that agents use in the test phase is also ignored as this is specific to each application of SDS, to consider each instance of SDS with a unique set of microtests to be a distinct variant would prohibit any meaningful grouping. When describing a specific instance of SDS it may be necessary to describe the set of microtests, but this is not necessary when describing a variant. The number of agents in a swarm is not considered to define a variant, and is considered to be a parameter which must be manually selected. Outside of extreme values which are so small that stable cluster formation is impossible, or so large that processing the swarm is impractical, swarm size does not effect any qualitative differences in performance.

3.3.4.1 Initialisation

The purpose of initialisation is to set the hypotheses of all agents in preparation for the first test phase which is the same purpose as the diffusion phase. Therefore the initialisation phases can usually by entirely omitted if the diffusion phase will be performed before the first test phase. Reversing the order of the diffusion phase and test phase makes no significant difference in the dynamical behaviour of SDS [14]. As noted by Bishop [6], in cases where knowledge of the search space is available a priori then this can be used to distribute the hypotheses accordingly, such as weighting the initial distribution towards one end of a one dimensional search space if solutions have most commonly been located there. While this is a valid observation, it can also be considered to be an effect of the diffusion phase. Any knowledge about the probable distribution of solutions in a search space may be incorporated into the hypothesis selection function, which will therefore be utilised during any diffusion phase instead of during only the first iteration.

Consider for example an instance of string-search SDS where the target string has a larger probability appearing towards the end of the search space towards the beginning, one could initialise hypotheses of agents with a probability distribution weighted towards the end of the search space, but this would only have any benefit if one of the agents was initialised with the hypothesis that pointed to the solution. A better solution would be to similarly weight the probability distribution for selection of new hypotheses, and potentially benefit from the knowledge at every diffusion phase. Even in cases where the knowledge only applies during the first iteration an effect identical to that of the initialisation phase may be achieved by a method of new hypothesis selection that behaves differently during the first iteration.

Furthermore by removing the initialisation phase the process of SDS can be seen more clearly as a procession of alternating diffusion phases and test phases. This interpretation makes it easier to see how a swarm resulting from any variant of SDS may be subsequently modified by any other variant of SDS. Put another way, each iteration of any variant of SDS can be understood as the initialisation phase of another variant of SDS.

Using the final state of a swarm in one instance of SDS as the initial state of the swarm in another instance of SDS has been investigated by Grech-Cini [32]. In the case studied, facial features were being tracked in the individual frames of a video. The location of the target in one image is likely to be close to its location in the previous image, hence the distribution of the swarm from the previous instance can be used as the initial distribution in a subsequent instance. This technique was found to be successful in the task of locating the ear-nose region in a video of a person reading a passage of a text.

3.3.4.2 Extraction

The method of extraction is also omitted as this mechanism remains largely the same over all variants. Many variants of SDS converge to a single large cluster, some converge to a number of clusters with sizes proportional to the score of the hypothesis, and some distribute agents most densely in the region of the search space near the highest scoring hypotheses, all of these can be interpreted as indicating solution quality with cluster size, the exact way this information will be used is specific to the nature of the task.

3.4 Variants of SDS

The following sections each describe a category of variants of SDS. First, implementation variants are those which modify something about the operation of SDS without having a significant impact on its performance as a search algorithm so as to enable SDS to be implemented in situations otherwise unsuitable. The remaining variants can be seen as biasing the behaviour of SDS towards exploration, evaluating further hypotheses, or exploitation, converging to currently maintained hypotheses, but there are many dimensions in which this bias can be effected.

Convergence variants affect behaviour of agents so as to modify the set of hypotheses from a given search space at which clusters may form. Exploration variants affect the interactions between agents so as to modify the diffusion of hypotheses through the swarm and therefore modify the probability of the optimal hypothesis being located. Heuristic variants use external knowledge of a search task to modify the behaviour of agents to be more suitable for that task, the range of possible optimisations is very large as each search task may have its own peculiarities to be optimised against, and for all heuristic variants the performance is only improved as long as the assumption underlying the heuristic is accurate.

Each variant is described in a number of aspects, there is the variation in individual behaviour, the resulting variation in the population behaviour, and the change in performance of the algorithm. Each change in performance of SDS has both advantages and disadvantages, in accordance with the No Free Lunch (NFL) theorems. Introduced in [107] and applied to search algorithms in [109][2], the NFL theorems state that over the set of all possible search spaces, there can be no algorithm which performs better on average than random search. The implication of the NFL theorems in the context of comparing SDS variants is that over the set of all possible search spaces there are as many search spaces for which standard SDS outperforms a given variant than there are search spaces for which the same variant outperforms standard SDS. This claim can also be applied to any two variants of SDS and also to any variant of SDS and any other search algorithm. Given the NFL theorems, the emphasis of any variant should not be evaluated solely on the advantage it introduces, but rather the trade-off it represents, and how large the effect of the trade-off should be

[2]The apparent anomaly of a 1995 work building on a 1996 work is that the former was based on a pre-print of the latter

to maximise the net-gain in performance after the negative impact of the disadvantage is realised against the positive effect of the advantage. The optimal choice of variant and the associated parameters should therefore be considered in the context of the search space of the given task and the requirements of the solution.

3.4.1 Implementation variants

Some variants of SDS have been shown to exhibit only a small performance decrease when compared to standard SDS, but enable SDS to be implemented in significantly different ways. For example, requirements for SDS to be implemented in hardware have been identified as, i) the internal state of agents is unavailable to other agents, ii) asynchronous mode of iteration, iii) limited connectivity between agents, and iv) a one-way communication protocol [14]. Other variants described here, which affect the implementation of SDS but do not significantly affect performance, are active diffusion, and deterministic SDS.

3.4.1.1 Asynchronous iteration

The behaviour of SDS has been studied in three distinct modes of iteration: synchronous, where all agents perform the mode of testing and then all agents perform the mode of diffusion; asynchronous [14], where each agent individually performs the mode of diffusion followed by the mode of testing; and parallel [14], where all agents perform mode of diffusion and mode of testing in their own time. While these variants are significant, they do not significantly change the performance of most variants except for the time scale [16].

In asynchronous iteration, each agent individually performs the diffusion behaviour followed by the test behaviour, shown in algorithm 12. It is one of the requirements for implementing SDS in hardware, and yet it does not affect the resource allocation behaviour of SDS [14].

Algorithm 12 $I^{\text{ASYNCHRONOUS}}$ — Asynchronous iteration

```
1: function $I^{\text{ASYNCHRONOUS}}(D, T, \text{swarm})$
2:     function $I'()$
3:         for each agent in swarm do
4:             $D(\text{agent})$
5:             $T(\text{agent})$
6:     return $I'$
```

In parallel iteration each agent has a probability distribution over time to update its state, shown in algorithm 13.

Algorithm 13 I^{PARALLEL} — Parallel iteration

```
1: function $I^{\text{PARALLEL}}(D, T, \text{swarm})$
2:     function $I'()$                    ▷ any agent may perform either action at any time
3:         for each agent in swarm do
4:             after some time perform $D(\text{agent})$
5:             after some time perform $T(\text{agent})$
6:     return $I'$
```

One of the aims of formulating asynchronous variants of SDS is to cast the algorithm in such a way that it may be more naturally implemented in electronic hardware, whilst retaining the essential characteristics of the algorithm. Though formulated with hardware implementation in mind, there have been software implementations of each of the individual requirements which retained the characteristic behaviour of SDS. Asynchronous iteration allocates resources in the same way as synchronous iteration, hence so does parallel iteration as it is a special case of asynchronous iteration [14]. Similarly, the strong convergence criterion holds for asynchronous iteration [14] and hence also for parallel iteration. When $\alpha = 1$ all agents will converge on the optimal hypothesis with probability 1 [14, p.24]. In which case the number of iterations before all agents maintain the optimal hypothesis can be computed as

$$TC = \sum_{a=0}^{A-1} \frac{A^2}{(A-a)(a+(A-a)p_\alpha)} \tag{3.27}$$

3.4.1.2 Private internal state

One of the features required for a hardware implementation of SDS was that agents may not directly access the internal state of other agents [14]. A mode of diffusion in which agents do not need to access the activity value of other agents (though still need to access the hypothesis of other agents) is described in algorithm 14. Note that the action of the polling agent is not affected by the activity of the polled agent. The polling agent always assumes the hypothesis of the polled agent, and the polled agent then does or does not generate a new hypothesis depending on its own activity.

It has been shown that the probability of an inactive agent assuming the optimal hypothesis during diffusion is identical for passive diffusion and private internal state diffusion, though the convergence time of an SDS using private internal state diffusion is marginally

Algorithm 14 D^{PASSIVE} — Private internal state diffusion

```
1: function $D^{\text{PASSIVE}}(DH, \text{swarm})$
2:     function $D'$(agent)
3:         if agent is inactive then
4:             hypothesis of agent ← $DH()$
5:             polled ← random agent in swarm
6:             hypothesis of agent ← hypothesis of polled
7:             if polled is inactive then
8:                 hypothesis of polled ← $DH()$
9:     return $D'$
```

longer [14].

3.4.1.3 Limited connectivity

The notion of limited connectivity, in which agents may only communicate with a subset of the swarm, was introduced with the original description of SDS [6]. Where in standard SDS each agent may poll any other agent in the swarm, variants have been developed where an agent may only poll a subset of the swarm.

If an agent may poll a second agent then the first agent can be thought of as being connected to the second agent. It is not required that all connections are reciprocal so one agent being connected to another does not necessitate that both agents are connected to each other. The set of all agents to which one agent is connected is named that agent's neighbourhood. The neighbourhoods of all agents can therefore be represented as the adjacency matrix of an unweighted directed graph. The number of possible connection configurations of a swarm of size A is therefore 2^{A^2}. Of all the possible connection configurations, a few types have been studied for their effects on the performance of SDS.

Reciprocal connectivity in an orthogonal grid, where each agent was bi-directionally connected to their four neighbours, has been shown to be sufficient for the effective performance of SDS [6]. While the reduced number of connections necessarily slows the diffusion of hypotheses through the swarm, the reduced complexity of the swarm significantly increases the feasibility of implementing SDS in hardware.

The performance of swarms with limited connectivity has been investigated experimentally [14]. An variant of SDS using $I^{\text{ASYNCHRONOUS}}$ was implemented with swarms of 16, 32 and 64 agents named 'lattices' and the number of connections between each agent was varied over the experiments. A fully connected lattice is equivalent to standard SDS and hence can be described by existing models. In comparing the convergence time for a fully connected lattice with that of the convergence time of simulations of partially connected lattices, it was shown that convergence is possible and with only a relatively small increase in convergence time even for lattices with few connections. For example, performance was "acceptable" with 4 to 6 connections in a lattice of 16 to 64 agents [14].

Further experimental work showed that performance is only slightly degraded if connectivity is significantly reduced, as long as the lattice exhibited random or small-world connectivity [17]. A lattice exhibits random connectivity when each agent is connected to a small number of other agents selected uniformly randomly. A lattice exhibits small-world connectivity when the majority of each agent's connections are to nearby agents, with a small number of connections to randomly chosen agents. Relatively long-distance connections are more costly in terms of materials and engineering complexity, so small-world networks have the advantage of requiring mostly short connections, but the small number of random and potentially long-distance connections ensure that the most nodes can be reached from all other nodes in a small number of steps. Small-world networks are therefore the preferred network topology for hardware implementation. An implementation of SDS on a chip was designed, which used limited connectivity, resulting in 64 agents arranged in four layers of 4×4 [102]; no performance results have been published. It was noted that the good performance of SDS under limited connectivity has further implications as biological neural structures have been observed as exhibiting analogous small-world connectivity, hence relaxing the requirements of full connectivity in SDS leads to a more biologically plausible architecture.

The most important aspect regarding limited connectivity is that the performance of SDS is negligibly affected unless the neighbourhoods of all agents are extremely limited. This therefore enables SDS to be implemented in various architectures in which the full connectivity of standard SDS would be impractical.

3.4.1.4 Active diffusion

Where passive diffusion defines that inactive agents poll random agents with the possibility of receiving a new hypothesis from an active agents, active diffusion relies on the active agents polling for inactive agents to which they pass their current hypothesis (D^{ACTIVE}, Algorithm 15). The resulting behaviour is very similar to passive diffusion, though slightly less robust due to being more sensitive to the effect of noise [62, p. 164]. The cause of the slightly reduced robustness is that in certain cases clusters increase in size slower than under passive polling. As each active agent may poll at most one other inactive agent, a cluster cannot increase in size faster than doubling each iteration, where in passive diffusion there is a possibility that all inactive agents will assume the same hypothesis in a single iteration.

There is a minor issue in implementing active diffusion; it is not clearly defined what should happen when an active agent attempts to impose their hypothesis on an agent which has already received a hypothesis from another active agent. Either the inactive agent should assume the new hypothesis, and hence slightly reduce the number of diffused hypotheses, or a mechanism needs to be implemented to avoid this feature, which would increase algorithmic complexity. There is also a question of when inactive agents should select a new hypothesis. One option (T^{ACTIVE}, Algorithm 15) is for all agents to select a new hypotheses whenever they become inactive. There is a risk that under this scheme agents may select a new hypothesis which is then discarded should an active agent immediately transmit their hypothesis to the agent, if new hypothesis selection is an expensive action then this extra work may be significant. Alternatively, once all active agents have performed their polling action, any inactive agents which were not polled could select new hypotheses. This method requires that the set of polled inactive agents is recorded somehow, at the cost of some algorithmic complexity. In practice, the effect is often negligible [62, p.148].

Active diffusion is shown to evolve in a "very similar" [62] way to standard SDS, though significantly more difficult to model mathematically, due to complications of the case of an inactive agent being selected by more than one active agent. The one-step evolution function for active diffusion shown in equation 3.28 and minimum convergence criteria is shown in equation 3.29 [62].

$$f_{\text{active}}\left(\alpha,\beta,\bar{c}_i\right) = \alpha\left[\bar{c}_i + \frac{\bar{c}_i\left(1-\beta\right) - \bar{c}_i^{\,2}\left(1-\frac{\beta}{\alpha}\right)}{\bar{c}_i + \beta\left(1-\frac{\bar{c}_i}{\alpha}\right)}\left(1 - e^{-\left(\bar{c}_i + \beta\left(1-\frac{\bar{c}_i}{\alpha}\right)\right)}\right)\right] \tag{3.28}$$

Algorithm 15 D^{ACTIVE} — Active diffusion and T^{ACTIVE} — Testing for active diffusion

```
 1: function $D^{\text{ACTIVE}}$(swarm)    ▷ No $DH$ required as agents select a new hypothesis immediately on becoming inactive
 2:     function $D'$(agent)
 3:         if agent is active then
 4:             polled ← random agent in swarm
 5:             if polled is inactive then
 6:                 hypothesis of agent ← hypothesis of polled
 7:     return $D'$

 8: function $T^{\text{ACTIVE}}(T, DH)$
 9:     function $T'$(agent)
10:         microtest ← $TM()$
11:         partial evaluation ← microtest(hypothesis of agent)    ▷ partial evaluation $\in \mathbb{B}$
12:         if partial evaluation = $true$ then
13:             agent becomes active
14:         else
15:             agent becomes inactive
16:             hypothesis of agent ← $DH()$
17:     return $T'$
```

$$\alpha_{\min} = \begin{cases} \dfrac{\beta}{1-(1-\beta)e^{-\beta}}, & \text{if } \beta > 0 \\ 0.5, & \text{if } \beta = 0 \end{cases} \tag{3.29}$$

The trade-off of active diffusion is that an arguably more natural behaviour is employed, but at the expense of slightly increased complexity in implementing and analysing the algorithm, and slightly reduced robustness. As the resulting practical behaviour is similar to passive diffusion, most variants of SDS employ passive diffusion unless active diffusion is a requirement for a specific implementation.

The combination of active diffusion and passive diffusion is known as dual diffusion. Each inactive agent attempts to assume a hypothesis from an active agent and each active agent attempts to impose their hypothesis on an inactive agent. As dual diffusion performs the combined behaviour of two different diffusion variants it can be considered to be a form of multi-diffusion, which is described in section 3.4.2.2.

3.4.1.5 Deterministic SDS

SDS relies on a random number generator in a number of steps, notably the selection of random agents in the diffusion phase and the selection of random microtests in the test phase, in a computational context this is assumed to be a pseudo-random number generator as is included in the standard libraries of modern programming languages. This is in some sense already not a random system as most pseudo-random number generators can be seeded with a constant value and will always produce the same sequence of values. The random number generator can be omitted entirely and any step which normally requires a random action can be replaced with a deterministic and regular process. It has been shown, using a simple iterative process, that SDS still performs a characteristic search, with exploration of the search space followed by rapid convergence [16]. This deterministic variant was observed to potentially offer improved performance if the deterministic sampling processes were quicker to compute than the usual pseudo-random process, and that the deterministic sampling process is guaranteed to provide samples evenly. As the performance gains of speeding up the random sampling process are likely to be small, and modern pseudo random number generators already provide fairly uniform sampling, the behaviour of a deterministic SDS is likely to be indistinguishable from standard SDS. The advantage of not requiring a pseudo-random number generator is that it reduces circuit complexity in the case of hardware implementation of SDS.

This group of variants, shows that there are diverse ways in which SDS can be implemented without losing the characteristic behaviour of a non-greedy search utilising partial evaluation and exhibiting a rapid convergence to a solution once identified by means of a positive feedback loop. Each variant has its own trade off, but each are likely to have a negligible impact on performance in practice outside of extreme cases.

3.4.2 Convergence variants

There are a number of variants which significantly effect the convergence of SDS. The aim of these variants is therefore to enable convergence for search spaces in which standard SDS would not converge, or to reduce the amount of the mean probability of an agent at any non-optimal hypothesis becoming active for search spaces in which standard SDS would converge so that convergence comes sooner. As anticipated by the NFL theorems, each variant represents a trade-off. Each variant introduces some algorithmic complexity to implement the additional behaviour and each variant introduces the requirement to select at least one extra parameter. For each application of a convergence variant any extra parameters introduced must be carefully selected else the performance may be worse than standard SDS applied to the same problem.

3.4.2.1 Multi-testing

When investigating the effect of noise on the convergence time , it was observed that the increased in time which resulted from the increase in noise could be reduced exponentially as agents each performed more than one microtest during the test phase [6]. This mechanism was hence named multi-testing. Multi-testing introduces two new parameters. They are 'amount', the number of tests that each agent should perform in each test phase, and 'combinator' the method for combining the multiple results into a single activity value for the agent. There are many possible combinators, the two which have been investigated either return *true* if all partial evaluations are *true*, called AND-multi-testing, or return *true* if any partial evaluations are *true*, called OR-multi-testing. An 'amount' of 1 or a combinator which simply selects one of the partial evaluation results at random is equivalent to standard SDS [62].

It has been shown that the effect of multi-testing is that the effective score of the optimal hypothesis is increased (in the case of OR-multi-testing) or decreased (in the case of AND-multi-testing) and the relative score of all other hypotheses is preserved [59]. The term 'effective score' is used here as the fundamental quality of any hypothesis remains unchanged, but the probability of an agent remaining active at any hypothesis after the test phase changes.

The advantage of this mechanism is that, in being able to modify the effective score of all hypotheses, values may be selected in which otherwise convergence would be impossible due to $\alpha < \alpha_{\min}$ or to reduce the amount of homogeneous background noise (β). For example, OR-multi-testing increases the robustness factor of SDS [62, p.88]. The disadvantage of this mechanism is the increased computational complexity of the test phase, as the number of microtests performed increases as a multiple of the number of agents. Multi-testing is therefore most useful in cases where the optimal hypothesis scores too poorly to form a stable cluster, or where too many suboptimal hypotheses score highly enough to form a stable cluster. Multi-testing is described in algorithm 16.

Algorithm 16 $T^{\text{MULTI-TESTING}}$ — Multi-testing and example combinators

```
1: function $T^{\text{MULTI-TESTING}}(TM, \text{amount}, \text{combinator})$
2:     function $T'$(agent)
3:         evaluations ← empty list
4:         for once per amount do
5:             partial evaluation ← microtest(hypothesis of agent)
6:             append partial evaluation to evaluations
7:         multi-evaluation ← combinator(evaluations)        ▷ multi-evaluation $\in \mathbb{B}$
8:         if multi-evaluation = true then
9:             agent becomes active
10:        else
11:            agent becomes inactive
12:    return $T'$
13: function AND-COMBINATOR(evaluations)
14:    if all partial evaluations in evaluations are true then
15:        return true
16:    else
17:        return false
18: function OR-COMBINATOR(evaluations)
19:    if any partial evaluations in evaluations are true then
20:        return true
21:    else
22:        return false
```

3.4.2.2 Multi-diffusion

Multi-diffusion is similar to multi-testing, except the mode of diffusion includes polling of multiple agents. The effect of multi-diffusion has not been studied directly but the studies of three variants of polling, each of which enabled differing amounts of diffusion in each iteration do provide some information. Active diffusion, which enabled the least diffusion was shown to have the lowest robustness (ζ) [62, p. 161], with passive diffusion having a higher robustness and dual polling, which combined passive and active polling, to enable the most diffusion in a single iteration, and hence the had the highest robustness (See table 3.1). In an experiment utilising 1000 agents, a search space of size 10000, $\beta = 0.1$ and a variant denoted as

$$\text{SDS}(I^{\text{SYNCHRONOUS}}(D^{\text{MULTI-DIFFUSION}}(DH^{\text{STANDARD}}, 2, \text{or-combinator}), T^{\text{STANDARD}}), H^{\text{FIXED}}(1000))) \tag{3.30}$$

the variant converged to the optimal hypothesis 77 times out of 100 when $\alpha = 0.4$ and 100 times out of 100 when $\alpha = 0.5$ whereas the same conditions utilising standard SDS converged to the optimal hypothesis 0 times out of 100. Multi-diffusion is described in algorithm 17.

Polling method	ζ
Active polling	0.543
Passive polling	0.614
Dual polling	0.769

Table 3.1: A table summarising the robustness factors of each recruitment polling method over all homogeneous background noise search spaces [62].

Algorithm 17 $D^{\text{MULTI-DIFFUSION}}$ — Multi-diffusion and example combinators

```
function D^MULTI-DIFFUSION(DH, amount, combinator)
    function D′(agent)
        if agent is inactive then
            polled ← list of random agents in swarm of length ‘amount’
            result ← combinator(polled)
            if combined = false then
                hypothesis of agent ← DH()
            else
                hypothesis of agent ← combined
    return D′
function AND-COMBINATOR(polled)
    if all agents in polled are active then
        return random agent in polled
    else
        return false
function OR-COMBINATOR(polled)
    if any agents in polled are active then
        active polled ← list of active agents in polled
        return random agent in active polled
    else
        return false
```

The advantage of multi-diffusion is that convergence may be possible in search spaces in which standard SDS will not converge due to the optimal hypothesis having an insufficiently high score. The disadvantage is that in cases where convergence was already likely, the steady state cluster sizes will be larger under multi-diffusion than under standard SDS. The effect is that there will be fewer agents generating new hypotheses once a cluster has formed. In cases where the first cluster forms at a sub-optimal hypothesis this reduces the probability of an agent locating the optimal hypothesis. Multi-diffusion is therefore a method which biases the action of SDS towards being greedy, but with the benefit of clusters being more stable.

3.4.2.3 The hermit

Hermit diffusion is the name given to a variant in which agents will occasionally refuse to communicate their hypothesis (D^{HERMIT}, Algorithm 18) [31]. An equivalent mechanism is to include a number of extra agents in the swarm which are permanently inactive and perform no action other than to be available for polling during the diffusion phase. All clusters will therefore have fewer inactive agents assuming their hypothesis each iteration, but the rate at which agents are becoming inactive and leaving the cluster will remain unaffected, hence all hypotheses will appear to have a lower score than in standard SDS.

Algorithm 18 D^{HERMIT} — Hermit diffusion

```
 1: function $D^{\text{HERMIT}}(DH$, hermitage, swarm)
 2:     function $D'$(agent)
 3:         if agent is inactive then
 4:             polled ← random agent in swarm
 5:             p ← true with probability P(hermitage)            ▷ $p \in \mathbb{B}$
 6:             if polled is active and not p then
 7:                 hypothesis of agent ← hypothesis of polled
 8:             else
 9:                 hypothesis of agent ← $DH()$
10:     return $D'$
```

The effect of the hermit is catastrophic in situations where the effective score of the optimal hypothesis is reduced below α_{min}, therefore making convergence impossible. The effect of the hermit is advantageous in situations where the search space contains many suboptimal hypotheses with a high enough score to form a stable cluster, if the effective score of the suboptimal hypotheses is reduced below minimum convergence criteria but the score of the optimal hypothesis remains greater than minimum convergence criteria then convergence time will be improved as there will be more inactive agents generating new hypotheses.

The opposite method, in which some agents are added to the swam which are permanently active is not considered to be worth investigating. Firstly, any active agent needs to be maintaining a particular hypothesis, how this hypothesis is selected is unclear, and secondly the effect is likely to be similar to that of the secret optimist.

3.4.2.4 The secret optimist

One variant introduces a probabilistic element into activity updating. Any agent whose microtest failed will only become inactive a certain proportion of the time and hence will perform further evaluations of their current hypothesis, this behaviour is known as the secret optimist [31].

This has the effect of increasing the effective test scores of all hypotheses. This is useful in the case where the score of the optimal solution is naturally low to form a stable cluster, but has the negative effect of increasing the activity at suboptimal solutions, and hence reducing the number of inactive agents exploring the search space.

This will be most useful in tasks where there is known to be a single optimal solution that will score significantly higher than all other solutions, but does have a sufficiently high score to form a stable cluster on its own.

This has the opposite effect to the hermit (Section 3.4.2.3, p.97) in that it raises the effective scores of all hypotheses.

Algorithm 19 T^{OPTIMIST} — Secret optimist testing

```
1:  function $T^{\text{OPTIMIST}}(TM, \text{optimism})$
2:      function $T'$(agent)
3:          microtest ← $TM()$
4:          optimistic ← true with probability P(optimism)
5:          partial evaluation ← microtest(hypothesis of agent)    ▷ partial evaluation $\in \mathbb{B}$
6:          if partial evaluation = true or optimistic = true then
7:              agent becomes active
8:          else
9:              agent becomes inactive
10:     return $T'$
```

The effect on the minimum convergence criteria (α_{min}) of the T^{OPTIMIST} is given in equation 3.31 [31] where i is the average number of iterations an agent will retain a failed hypothesis.

$$\alpha_{\text{min}} = \frac{1 + i\beta}{2 + i - \beta} \tag{3.31}$$

$$\alpha_{\text{secret optimist}} = \frac{i\alpha}{i + 1} \tag{3.32}$$

While this is a larger value than with standard SDS the effective score of the optimal hypothesis will also increase. The advantage of this variant is that α_{min} will always be lower than the equivalent case using standard SDS as long as $i > 0$. When $i = 0$ the behaviour is identical to standard SDS. The disadvantage of this variant is that the effective score of all hypotheses increases and so there will be an increase in the homogeneous background noise.

3.4.3 Exploration variants

These variants modify the exploration behaviour of SDS which does not change which clusters may form but in modifying the resource allocation during the search it will affect the average number of iterations taken to locate the optimal hypothesis. The trade off is that in order for there to be more agents selecting new hypotheses there must be fewer agents evaluating existing hypotheses, this leads to clusters being less stable and potentially collapsing altogether.

3.4.3.1 Context-free diffusion and context-sensitive diffusion

Standard SDS has been criticised for its behaviour of forming a single large cluster, leaving few agents to test hypotheses in the remainder of the search space, hence possibly missing superior solutions [63]. To decrease the number of active agents and hence to increase the number of agents generating new hypotheses two variants were defined in which active agents may become inactive during the diffusion phase. The variants are named context-free diffusion and context-sensitive diffusion.

Context-free diffusion ($D^{\text{CONTEXT-FREE}}$, Algorithm 20, p.100) introduces a mechanism where active agents poll an agent at random from the swarm and in the case that the polled agent is active, the polling agent will become inactive and will generate a new hypothesis using DH. Context-sensitive diffusion ($D^{\text{CONTEXT-SENSITIVE}}$, Algorithm 21, p.100) introduces

	Context-free SDS	Context-sensitive SDS
$\bar{c}_{i+1}$	$\bar{c}_i\left[2\alpha(1-\beta)\right]-\bar{c}_i{}^2\left[2(\alpha-\beta)\right]$	$\bar{c}_i\left[\alpha(2-\beta)\right]-\bar{c}_i{}^2(2\alpha-\beta)$
$\alpha_{\min}$	$\dfrac{1}{2(1-\beta)}$	$\dfrac{1}{2-\beta}$
ζ	$1-\ln 2\approx 0.307$	$2(1-\ln 2)\approx 0.614$
γ	$\dfrac{2\alpha(1-\beta)-1}{2(\alpha-\beta)}$	$\dfrac{\alpha(2-\beta)-1}{\alpha-\beta}$

Table 3.2: Comparison of the one step evolution function ($\bar{c}_{i+1}$), minimum convergence criteria ($\alpha_{\min}$), robustness (ζ), and steady state (γ) of Context-free SDS and Context-sensitive SDS

a similar mechanism but adds an extra clause. Active agents poll an agent at random from the swarm but the polling agent only becomes inactive and generate a new hypothesis using DH in the case where the polled agent is active and both agents share the same hypothesis.

Algorithm 20 $D^{\text{CONTEXT-FREE}}$ — Context-free diffusion

```
1: function $D^{\text{CONTEXT-FREE}}(DH, \text{swarm})$
2:     function $D'$(agent)
3:         polled ← random agent in swarm
4:         if agent is inactive or polled is active then
5:             if agent is inactive and polled is active then
6:                 hypothesis of agent ← hypothesis of polled
7:             else
8:                 agent becomes inactive
9:                 hypothesis of agent ← $DH()$
10:    return $D'$
```

The one step evolution functions, minimum convergence criteria ($\alpha_{\min}$), robustness (ζ) and steady state (γ) for context-free SDS and context-sensitive SDS have been derived from developments of the DDSM [62] and can be seen in Table 3.2, p.100.

Algorithm 21 $D^{\text{CONTEXT-SENSITIVE}}$ — Context-sensitive diffusion

```
1: function $D^{\text{CONTEXT-SENSITIVE}}(DH, \text{swarm})$
2:     function $D'$(agent)
3:         polled ← random agent in swarm
4:         if agent is inactive and polled is active then
5:             hypothesis of agent ← hypothesis of polled
6:         else
7:             a ← agent is inactive
8:             b ← polled is active and hypothesis of agent = hypothesis of polled
9:             if a or b then
10:                agent becomes inactive
11:                hypothesis of agent ← $DH()$
12:    return $D'$
```

These mechanisms effectively cap the number of agents that can be globally active, in the case of context-free SDS, or active in the same cluster, in the case of context-sensitive SDS. This is useful in cases where the first hypothesis at which a stable cluster forms is unlikely to be the optimal solution.

Some publications [104, 72, 40, 105] refer to a *relate phase* in which the behaviours specific to context-free SDS and context-sensitive SDS take place, but rather than defining a new phase, these will be considered modifications of the diffusion phase to avoid unnecessary complexity.

Three scenarios have been proposed, in which context-free SDS might be a better choice than standard SDS [63].

1. there are multiple hypotheses that share the same score as the optimal hypothesis and the aim is to identify them all
2. the aim is to find the optimal hypothesis and the remaining hypotheses with the highest scores
3. the search space and/or microtests are changing over time, so the best solution at one time may not be so some time later

Where standard SDS will equally allocate agents to two hypotheses of equal score over many iterations, with one cluster being orders of magnitude larger than the other for some time, and then a switch to the other one being larger context-free SDS is able to simultaneously maintain multiple swarms. In the case where there are two hypotheses with equal scores in standard SDS one cluster will be orders of magnitude larger than the other for some time, after which a switch will occur and the other cluster will be the larger. context-free SDS also has the ability to maintain multiple swarms at hypotheses of different score, but the cluster size is dramatically smaller for hypotheses of only slight difference in score, due to the very non-linear gain amplification.

In summary context-free SDS distributes its resources more evenly than standard SDS but maintains a comparable stability of clusters supporting solutions.

This makes the modelling assumption that agents at the noise assumption will never become inactive in the diffusion phase, but this is an imperfect assumption as the diffusion phase will result in noise agents duplicating identical hypotheses.

3.4.3.2 Hypothesis transmission noise

Uniformly random hypothesis selection has been described as a strength and a weakness of SDS It is a strength as SDS does not require that there is a usable gradient in the objective function of the search task, but it is a weakness as SDS is unable to utilise the information should one exist [16].

In this variant, when an agent assumes a hypotheses from another agent, the hypothesis is copied with some random perturbation, known as hypothesis transmission noise. This has the effect that a form of hill-climbing behaviour emerges in the locality of the transmitted hypothesis [16, 76]. This has the advantage of exploiting self-similarity of a search space, which is when similarly located hypotheses have similar scores. This can be understood intuitively by imagining a cluster near an optimal solution in a continuous search space. The time for which agents at the cluster will remain active is a function of the score of that hypothesis. Agents which diffuse to the side further away from the optimal will remain active for a shorter time, and agents on the side towards the optimal will remain active for longer, in this way the cluster will appear to flow towards the optimal until it is centred upon it. This mechanism makes it very rare for agents to have identical hypotheses and so a cluster must be considered to be a collection of agents with similar hypotheses rather than identical hypotheses. The result may be 'extracted' by calculating the average hypothesis of all active agents. A mode of diffusion utilising hypothesis transmission noise can be seen in algorithm 22, it requires an extra parameter 'noise' which is a function which takes a hypothesis as a parameter and returns a perturbed hypothesis.

There are many functions that could be used to perturb a hypothesis, an example using a Gaussian distribution is given in algorithm 22. The amount of noise introduced, by using a larger value for sigma in the example using Gaussian distribution, has an effect on the speed with which a cluster will appear to move through a search space. The trade-off in this context is that a faster moving cluster may miss smaller details of the search space and effectively evaluate multiple hypotheses at once, with potentially misleading results. A small amount of noise will lead to slower convergence and increases the probability that the cluster will not move out of any local optima in the search space, of course a zero or

Algorithm 22 D^{NOISE} — Noisy diffusion and Gaussian noise

```
1:  function $D^{\text{NOISE}}(DH$, noise, swarm)
2:      function $D'$(agent)
3:          if agent is inactive then
4:              polled $\leftarrow$ random agent in swarm
5:              if polled is active then
6:                  $h \leftarrow$ hypothesis of polled
7:                  $n \leftarrow$ noise($h$)
8:                  hypothesis of agent $\leftarrow n$
9:              else
10:                 hypothesis of agent $\leftarrow DH()$
11:     return $D'$
12: function NOISE(hypothesis, sigma)
13:     d $\leftarrow$ gaussian distribution with standard deviation = sigma
14:     x $\leftarrow$ random value from d
15:     return hypothesis + x
```

negligable amount of noise is equivalent to standard SDS where clusters do not appear to move at all. A mechanism has been suggested for estimating a value for the transmission error online from the standard deviation of the test scores of an "elite sample" of the best scoring agents [75].

The feature where, in the case of a large amount of noise, a cluster may effectively evaluate multiple solutions may be an advantage or disadvantage depending on the context. For some tasks it may be useful information that a certain area of the search space contains many high scoring hypotheses, whereas in other tasks it may be essential that a single hypothesis is evaluated in isolation. For example, consider the following variant of SDS. In a search space with a wide 'spike' of high scoring hypotheses around one area, and a much thinner and slightly taller 'spike' in another area this variant of SDS would consistently converge to the hypothesis representing the peak of the wider spike if the value for noise was above a certain value and would consistently converge to the hypothesis representing the peak of the thinner spike if the value for noise was lower than a certain value.

3.4.3.3 Comparative partial evaluations

In cases where the partial evaluations for a search task are not easily described as a set of boolean functions, the functions may instead return scalar values. When using standard SDS each microtest can be treated as if it were a boolean function by providing a threshold for each microtest such that results over a certain value are considered to be *true* and results under the value are considered to be *false*, while this will work for many cases it is required that a threshold is chosen. At whichever level the threshold is selected hypotheses which score just under the threshold for many of the microtests, or signficiantly over the threshold in a small number of microtests will appear to have a poor score, similarly where hypotheses with the opposite properties may appear to be high scoring hypotheses even though the overall quality of the hypothesis may be lower than a hypothesis with a lower effective score.

An alternative approach, which does not require a threshold is for agents to convert the scalar values of the microtests to boolean values by comparing them with the scalar value obtained by an agent at a different hypothesis.

Algorithm 23 $T^{\text{COMPARATIVE}}$ — Comparative testing

```
function $T^{\text{COMPARATIVE}}(TM)$
    function $T'$(agent)
        polled ← randomly selected agent in swarm
        microtest ← $TM()$
        partial evaluation a ← microtest(hypothesis of agent)
        partial evaluation p ← microtest(hypothesis of polled)
        if partial evaluation a > partial evaluation p then
            agent becomes active
        else
            agent becomes inactive
    return $T'$
```

The advantage of this approach is that a threshold does not need to be chosen, the threshold is effectively selected by the action of the evolving swarm, each iteration effectively removes half of the swarm which evaluates lower than average. The disadvantage is that as there is no minimum threshold for activity, the highest scoring hypotheses located so far will always maintain a cluster, convergence therefore cannot be detected from the global activity alone. There is a further disadvantage that it is possible for the superiority of one hypothesis to be intransitive. This means that there may be set of three or more hypotheses such that each hypothesis consistently scores higher than one other hypothesis. This can be seen in the case where three hypotheses A, B, and C each score $[2, 2, 6, 6, 7, 7]$, $[1, 1, 5, 5, 9, 9]$, and $[3, 3, 4, 4, 8, 8]$ respectively. If a partial evaluation is to select at random a single value from the list pertaining to that hypothesis then A will most commonly evaluate higher than B, B will most commonly evaluate higher than C, and C will most commonly evaluate higher than A. In such cases the global activity of an SDS remains high, but the hypothesis of the largest cluster remains unstable.

3.4.4 Heuristic variants

Variants which employ heuristics of a specific task, as is the nature of heuristics the performance of the search will be improved as long as the heuristic is veridical.

3.4.4.1 Fixed heuristics

If some information is known about the distribution of the scores of hypothesis in a search space then new hypothesis selection may be modified so as to select hypotheses more frequently from areas with a higher probability of containing hypotheses with higher scores. If the information is accurate then the higher probability of selecting hypotheses with higher scores will effectively increase the homogeneous background noise as more randomly selected hypotheses will have higher scores but the probability of selecting the optimal hypothesis will also be higher. Similarly, if one microtest is known to be significantly more valuable in determining the quality of a solution represented by a hypothesis the mode of microtest selection can employ a probability distribution to bias selection towards the more informative microtests. A related technique is to select a hypothesis which is furthest away from any previously selected hypothesis [76], this method is almost the opposite of selecting a hypothesis from an area of the search space which has been proven to be relatively dense with high scoring hypotheses, and as such this method is more likely to perform a search which is well distributed through the search space.

In cases where the search space is too large to practically enumerate all hypotheses a mode of hypothesis selection may be employed that uses an existing heuristic. Consider for example the TSP, the number of possible routes is of the order $n!$ where n is the number of cities, selecting one route from the set of all possible routes may be impractical. Therefore a heuristic may be employed such as a stochastic greedy method, or a heuristic which guarantees to only select paths with certain features. Similarly, in the task of estimating a hyperplane that best describes a set of points a hypothesis can be stochastically constructed by fitting a hyperplane to a small randomly selected subset of the points. Heuristics such as these have the potential to dramatically reduce the size of a search space, and hence to speed up the convergence time of SDS, they achieve this however by leaving significant sections of the search space unexplored. It is therefore most suitable to apply a fixed heuristic in cases where a uniformly random search of the search space would be impractical, or when strong heuristics exist for the given task.

SDS is not limited to using existing heuristics however, a method of adapting a heuristic as the search evolves is readily available.

3.4.4.2 Adaptive heuristics

In all the variants examined so far, when an agent selects a new hypothesis they do so entirely independently of the rest of the swarm. As there are likely to be many agents maintaining a range of hypotheses an agent may use the distribution of hypotheses over the set of active agents as information on how to bias the selection of their next hypothesis. A hypothesis may also be constructed from the combination of parts of the hypotheses of various agents. This is analogous to the technique employed by ACO, where a path is constructed probabilistically by combining sections of the route that were previously evaluated. An agent is therefore no longer required to generate the optimal hypothesis entirely at random. A swarm can be imagined to be evaluating hypothesis not just for their standalone value, but for the potential that they consist of strong parts suitable for recombination into other hypotheses. Care should be taken not to lean too heavily on the hypotheses of active agents as it is effectively a form of multi-diffusion, which increases the 'greedyness' of the search. Consider the extreme case where a new hypothesis is 'constructed' by simply taking the hypothesis of an active agent, this is exactly equivalent to passive diffusion.

As with fixed heuristics the performance will be improved as long as the underlying assumption is valid, which in this case is that a strong hypothesis can be combined from the features of other strong hypotheses. This kind of polling based heuristic is therefore most suitable in cases where a uniformly random search of the search space is impractical, and where no strong heuristics are applicable for the task.

3.4.5 Halting variants

There are many techniques to halt SDS, and most can be combined. Some use detection of cluster formation as a means to determine when to finish.

3.4.5.1 Time based halting

Fixed iteration halting and fixed wall-clock time halting A very simple method of halting, fixed iteration count halting (H^{FIXED}, algorithm 11, p.83) has already been described. This method simply keeps track of the number of iterations elapsed, and halts once a given threshold is reached. A suitable value for the maximum number of iterations must be defined beforehand and doing so requires striking a balance between more thorough evaluations of a search space, at the expense of longer execution times, and timely results, at the expense of a higher change of optimal hypothesis. Halting after a certain amount of real time is also a valid method, but this method is rarely explored in academic studies so as to avoid any results being dependant on specifics of the host hardware, the software which is implementing the SDS, and the task-specific microtests. Real-time threshold halting has

Algorithm 24 H^{TIME} — Fixed wall-clock time halting

```
1: function $H^{\text{TIME}}$(maximum time)
2:     elapsed time ← 0
3:     function $H'$( )
4:         update elapsed time
5:         halt ← elapsed time > maximum time                ▷ halt $\in \mathbb{B}$
6:         return halt
7:     return $H'$
```

analogous advantages and disadvantages to fixed iterations halting.

Fixed iteration count halting and Fixed wall-clock time halting are the only halting methods which are guaranteed to halt over all search spaces, whether this is an advantage or disadvantage is dependent on the search task to which it is being applied. This is an advantage in cases where halting before convergence, or before convergence to a hypotheses with a sufficient score is unacceptable, in such cases it may be preferable to loop indefinitely than to proceed with an unsuitable solution. This is a disadvantage in cases where any decision is better than no decision, in such cases even the poorest of hypotheses represent a better case than failing to proceed at all.

Algorithm 25 $H^{\text{INDEFINITE}}$ — Indefinite halting

```
1: function $H^{\text{INDEFINITE}}$( )
2:     function $H'$( )
3:         return *false*
4:     return $H'$
```

Indefinite halting An instance of SDS does not need to halt to be useful, at any time in its execution a separate process may read the state of the swarm and calculate the clusters. This has been described as the preferred method of halting for dynamically changing problems, where SDS is used to maintain a solution to an objective function that is changing over time [16, p.115]. The obvious disadvantage is that this method potentially requires an unlimited amount of resources. If a cluster has formed at the optimal hypothesis, and the search space is not evolving over time, then there is no value to be gained in further searching and there is a small but ever-present risk that the cluster will spontaneously collapse. Furthermore a sampling strategy will need to be defined, which will introduce a number of extra configuration parameters and thresholds. The advantage of this method is that a series of solutions could be reported from an ever changing search space.

3.4.5.2 Cluster size based halting

Unique hypothesis count A method which halts when the number of unique hypotheses maintained by the entire swarm is lower than a certain threshold. This method was used in an application of SDS to realistically place objects in a visual scene [10]. As convergence is characterised by the formation of a single large cluster, this necessitates that many agents will be maintaining the same hypothesis and therefore only the relatively few remaining agents may maintain distinct hypotheses. The advantage of this method is that the chosen value for the threshold parameter controls the minimal quality of a hypothesis that will induce a halt. The disadvantage of this method is that the threshold must be chosen with respect to the background noise, as search spaces with high homogeneous background noise will quickly converge to states with relatively few hypotheses, and search spaces with very low homogeneous background noise will have many distinct hypotheses even after the formation of a large cluster.

Algorithm 26 H^{UNIQUE} — Unique hypothesis count halting

```
1: function $H^{\text{UNIQUE}}$(unique count)                ▷ unique count $\in \mathbb{N}$
2:     function $H'$( )
3:         hypotheses $\leftarrow$ hypotheses of agents in swarm
4:         unique $\leftarrow$ unique values in hypotheses
5:         halt $\leftarrow$ unique < unique count               ▷ halt $\in \mathbb{B}$
6:         return halt
7:     return $H'$
```

Largest cluster size and global activity threshold halting There are two closely related halting methods which halt when a certain value reached a given threshold. The values are i) the largest cluster size, and ii) the global activity. These methods are similar as global activity is a proxy for the largest cluster size. This can be understood intuitively as once a stable cluster has formed, the level of global activity is equal to the size of the largest cluster plus the activity of all background noise hypotheses, and as the background noise is naturally stable, the level of global activity will correlate with the size of the largest cluster.

The advantage of these halting methods is that, like unique hypothesis count halting,

Algorithm 27 H^{LARGEST} — Largest cluster size halting

```
1: function H^LARGEST(swarm, threshold)                    ▷ threshold $\in [0,1]$
2:     function $H'$( )
3:         cluster $\leftarrow$ largest cluster
4:         count $\leftarrow$ number of active agents in cluster       ▷ count $\in \mathbb{N}$
5:         largest $\leftarrow \frac{\text{count}}{\text{swarm size}}$                            ▷ largest $\in \mathbb{R}$
6:         halt $\leftarrow$ largest >= threshold                      ▷ halt $\in \mathbb{B}$
7:         return halt
8:     return $H'$
```

Algorithm 28 H^{ACTIVITY} — Global activity threshold halting

```
1: function H^ACTIVITY(swarm, threshold)                   ▷ threshold $\in [0,1]$
2:     function $H'$( )
3:         active count $\leftarrow$ number of active agents in swarm   ▷ active count $\in \mathbb{N}$
4:         activity $\leftarrow \frac{\text{active count}}{\text{swarm size}}$                       ▷ activity $\in \mathbb{R}$
5:         halt $\leftarrow$ activity >= threshold                     ▷ halt $\in \mathbb{B}$
6:         return halt
7:     return $H'$
```

the chosen value for the threshold parameter controls the minimal quality of a hypothesis which will induce a halt. These halting methods, like unique hypothesis count halting, also require that the threshold is chosen with respect to the amount of noise in a search space as higher values of global activity will be encountered in search spaces with higher homogeneous background noise, even if the score of the optimal hypothesis is the same. The disadvantage of largest cluster size halting in comparison with global activity halting is that the calculation of the largest cluster size is somewhat more computationaly complex than the calculation of the global activity as size of all clusters must be calculated, rather than simply counting the number of active agents.

A disadvantage of all threshold based halting methods is that halting as soon as the threshold is reached introduces the possibility that clusters which has begun forming at superior hypotheses are missed. A way to mitigate this possibility is to detect stable convergence which requires that clusters form and subsequently stabilize, as this will ensure that the clusters detected were stable, and that superior clusrers were no in the process of forming.

3.4.5.3 Convergence based halting

Convergence is detected by a stability in the swarm, theoretically this refers to a stability in the sizes of the largest cluster. In some variants the global activity is a suitable proxy [62] and a more convenient value to calculate. In some variants, such as $D^{\text{CONTEXT-SENSITIVE}}$, multiple significant stable clusters may form, and in others such as $T^{\text{COMPARATIVE}}$ global activity is always stable, in these cases convergence must be detected using other methods.

Strong and weak halting criteria One convergence based halting method is described in the original definition of SDS [6]. This method, known as the strong halting criterion [64, p.8], halts when the size of the largest cluster is determined to be stable over a number of iterations. Stability is defined by two parameters a and b which define the size of largest cluster c, which will induce a halt and a tolerance of error in which the largest cluster size may fluctuate and still be considered stable. The strong halting criteria are defined as

$$\underset{a,b>0}{\exists}(2b < A \wedge b + a \leq A \wedge a - b \geq 0) \underset{n_0}{\exists} \underset{n>n_0}{\forall}(|c - a| < b) \tag{3.33}$$

Given the values; a, the cluster size which is sufficiently large to indicate convergence; and b, an amount the cluster size may vary and still be considered stable, then the algorithm has reached equilibrium under the strong halting criteria if there exists an instant n_0 after which the largest cluster size c is sufficiently stable such that $a - b \leq c_n \leq a + b$, otherwise written as $|c_n - a| < b$. In practice this requires a further variable t, which defines a number of iterations that $|c_n - a| < b$ must hold for the algorithm to halt. This ensures that even when the search converges on a solution that there will be more searching for a hypothesis with a potentially higher score before the search halts. The weak halting criteria, identical in all respects except than c_n refers to the global activity rather than the size of the largest cluster, exhibits identical performance in detecting convergence.

The advantage of these methods is that the algorithm will not halt whilst a cluster is in the process of forming. As with cluster size based halting methods the requirement for

Algorithm 29 H^{STRONG} — Strong halting

```
1: function $H^{\text{STRONG}}$(a, b, T)
2:     t ← 0
3:     function $H'$( )
4:         $c \leftarrow$ largest cluster size
5:         if $|c - a| < b$ then
6:             $t \leftarrow t + 1$
7:         else
8:             $t \leftarrow 0$
9:         halt $\leftarrow t > T$                                  ▷ halt $\in \mathbb{B}$
10:        return halt
11:    return $H'$
```

Algorithm 30 H^{WEAK} — Weak halting

```
1: function $H^{\text{WEAK}}$(a, b, T)
2:     t ← 0
3:     function $H'$( )
4:         $c \leftarrow$ global activity
5:         if $|c - a| < b$ then
6:             $t \leftarrow t + 1$
7:         else
8:             $t \leftarrow 0$
9:         halt $\leftarrow t > T$                                  ▷ halt $\in \mathbb{B}$
10:        return halt
11:    return $H'$
```

the largest cluster size to be consistently within the region defined by $|c_n - a| < b$ ensures that the process will only halt when a hypothesis with a sufficiently high score has been located. A disadvantage of this method is that the region of stability defined by a and b has to be selected carefully, if $b < 1 - a$ holds then there will be a value for c which is both outside of the region, but also representing a relatively high scoring hypothesis. For example, if $a = 0.8$ and $b = 0.1$ then to converge it must hold that $0.7 < c < 0.9$. The process would therefore never halt if c remained stable around 0.999 even though this represents convergence to a hypothesis with a high score.

Running-mean stability An alternative method to detect convergence without selecting a global activity threshold is to record the global activity for a given number of iterations and halt when the standard deviation of the recorded values is within a given threshold. This method requires the selection of a number of parameters: the maximum memory size which determines the number of iterations to record at any one time, after which each new value recorded causes the oldest recorded value to be forgotten; the minimum stability which is the value which standard deviation of the recorded values must be lower than to be considered stable, and; minimum stable iterations, which is the number of consecutive iterations which must be considered stable before halting. The advantage of this method is

Algorithm 31 H^{STABLE} — Stable global activity halting

```
 1: function $H^{\text{STABLE}}$(max memory size, min stability, min stable iterations)
 2:     memory ← empty queue of maximum length ‘max memory size’
 3:     stable iterations ← 0
 4:     halt ← false
 5:     function $H'$( )
 6:         activity ← $\frac{\text{active agents in swarm}}{\text{swarm size}}$
 7:         push activity into memory
 8:         stability ← standard deviation of all values in memory
 9:         if stability <= min stability then
10:             stable iterations ← stable iterations + 1
11:             if stable iterations >= min stable iterations then
12:                 halt ← true
13:         else
14:             stable iterations ← 0
15:         return halt
16:     return $H'$
```

that convergence will be detected for a hypothesis of any score sufficient to form a stable cluster. There is a disadvantage that this method will halt too soon if no solution is located within the first few iterations as the largest cluster size being stable at very low values will induce a halt.

This method therefore benefits from being combined with a cluster size based halting method to avoid the case where it halts before a stable cluster has formed.

3.4.5.4 Combined halting methods

As the output of each halting method is a boolean value they can be combined using boolean logical operators to form new halting methods. Any such combined halting method will exhibit the advantages and disadvantages of the component halting methods but in working together may produce more consistent performance than any individual halting method.

For example, the combined halting method

$$H^{\text{AND}}\left(H^{\text{FIXED}}(1000), H^{\text{OR}}\left(H^{\text{FIXED}}(10000), H^{\text{ACTIVITY}}(0.5)\right)\right)$$

guarantees at least 1000 iterations, after which will only halt once the global activity reaches 0.5 or a maximum of 10,000 iterations elapse. Such a combined halting method will avoid the disadvantage of global threshold halting, where the algorithm may halt after the early formation of a cluster before significant exploration of the search space is performed, as long as a superior solution is found within $10,000$ iterations. Similarly the disadvantage of global activity threshold halting potentially never halting is also mitigated by the fixed iterations halting method. These mitigations come at the cost of some increased algorithmic complexity and the requirement to select values for the three thresholds (minimum iterations, global activity threshold, and maximum iterations) with all the considerations already described.

A practical combination of the halting methods described in this section is defined in algorithm 32, this halting function is guaranteed to halt after a defined upper limit of iterations, and will halt earlier if a stable cluster forms over a given size and will ignore any activity from clusters which form before a defined lower limit. As with the simpler combined halting method described above, there is increased algorithmic complexity, and the requirement of five arguments to be manually determined.

Algorithm 32 Example combined halting

```
1: function $H^{\text{COMBINED}}$(minimum iterations, maximum iterations, memory, stable iterations, minimum activity )

2:     return $H^{\text{OR}}$(
3:         $H^{\text{AND}}$(
4:             $H^{\text{FIXED}}$(minimum iterations),
5:             $H^{\text{STABLE}}$(memory, stable iterations),
6:             $H^{\text{ACTIVITY}}$(minimum activity),
7:         ),
8:         $H^{\text{FIXED}}$(maximum iterations)
9:     )
```

3.4.6 Summary of variants of SDS

The behaviour introduced by each variant brings with it advantages and disadvantages. Each variant therefore has superior performance than standard SDS in certain circumstances and inferior performance than standard SDS in other circumstances. This trade-off is to be expected and is in accordance with the No Free Lunch (NFL) theorems. For example, the effect of multi-testing is to increase the effective score of all hypotheses, increasing the ability of the optimal hypothesis to maintain a larger cluster but also increasing the homogeneous background noise. Multi-testing therefore improves performance in cases where a stable cluster would otherwise not form, and impairs performance in cases where the increase in noise increases the time for the optimal hypothesis to be located. Just as with standard SDS, over all possible search spaces any multi-testing variant will outperform random search in as many search spaces as random search outperforms the multi-testing variant. If any task can be considered to be a search over a subset of all the possible search spaces then to select a variant of SDS without any analysis of such a subset is to make the assumption that by chance alone the subset of search spaces is one for which on average the variant outperforms standard SDS, or indeed, random search [108]. This observation raises the question of how one may identify which subset of search spaces are represented in a given task and hence which variant is most likely to outperform standard SDS. A number of features of search problems have been identified in which standard SDS exhibits greatest performance [16], and this can be interpreted as an identification of the search spaces in which standard SDS outperforms random search. For all other variants some features of search spaces which lead to improved or impaired performance have been identified.

The NFL theorems in context of halting variants are distinct from other variants as they do not change the search or convergence behaviour of the algorithm. Time-based halting functions merely change the number of iterations that will elapse before halting, the trade off is therefore how long the algorithm must run before a user is confident that the optimal hypothesis has been located, while the probability of locating the optimal hypothesis grows over time for any non-greedy variant, optimality can never be guaranteed. A related trade off occurs in the case of halting methods which examine the state of the swarm to detect convergence, every method of detecting convergence introduces the potential for the method to be deceived and hence to erroneously halt. As with time-based halting the user will have to decide whether the risk of an erroneous halt is sufficiently low for each application of SDS.

SDS is most suitable for search tasks where there is little, or no, information available about the distribution of solutions in the search space, and the evaluation of each element of the search space can be decomposed into the evaluation of a number of separate functions. SDS is less suitable for tasks where it is absolutely essential that a single, globally-optimal solution is found, in which case an exhaustive search may be preferable, or when features of the search space can be exploited to calculate the location of strong solutions rather than relying on random hypothesis generation.

Like ACO, SDS is a stochastic procedure, where the locally guided action of individuals gradually leads to the globally optimal behaviour of the swarm. Unlike ACO, the state of an instance of standard SDS can be described with a list of the hypotheses of the active agents, and a count of the inactive agents whereas ACO requires that the modifications to the search space are represented somehow which which adds some memory requirements.

DFO shares an important property with SDS, both algorithms are sufficiently simple as to facilitate mathematical analysis of their performance.

Furthermore some of the characteristic of swarm intelligence is demonstrated here, as variants which modify individual behaviour in simple ways enable useful and varied behaviour to emerge at the population level. Regarding the hermit, we see how a mechanism which is equivalent to a swarm including a number of completely ineffective agents can still result in an effective search, and in specific circumstances may even improve performance. An investigation into the precise conditions in which the hermit improves performance will be of some benefit. Similarly with the secret optimist, the simple case of agents remaining active after a failed microtest could be interpreted as discarding valuable information, but the effect is that search spaces in which standard SDS would not converge due to the optimal hypothesis having an insufficient score can be expected to converge with an appropriate probability of 'optimism'.

Regarding multi-diffusion, we have only observed that a variant in which agents poll two agents, increases the robustness of the variant, at the expense of reduced exploration of the search space by inactive agents. No investigation has been performed into an exact formulation of the affect on robustness. As conceived the multi-diffusion behaviour is related to the logical *OR* function, as diffusion will occur if *any* polled agents are active, an alternative strategy would be related to the logical *AND* function which would require that *all* of the polled agents were active. This alternative would reduce the amount of diffusion, and hence presumably reduce the robustness of the variant, and increase the amount of exploration of the search space.

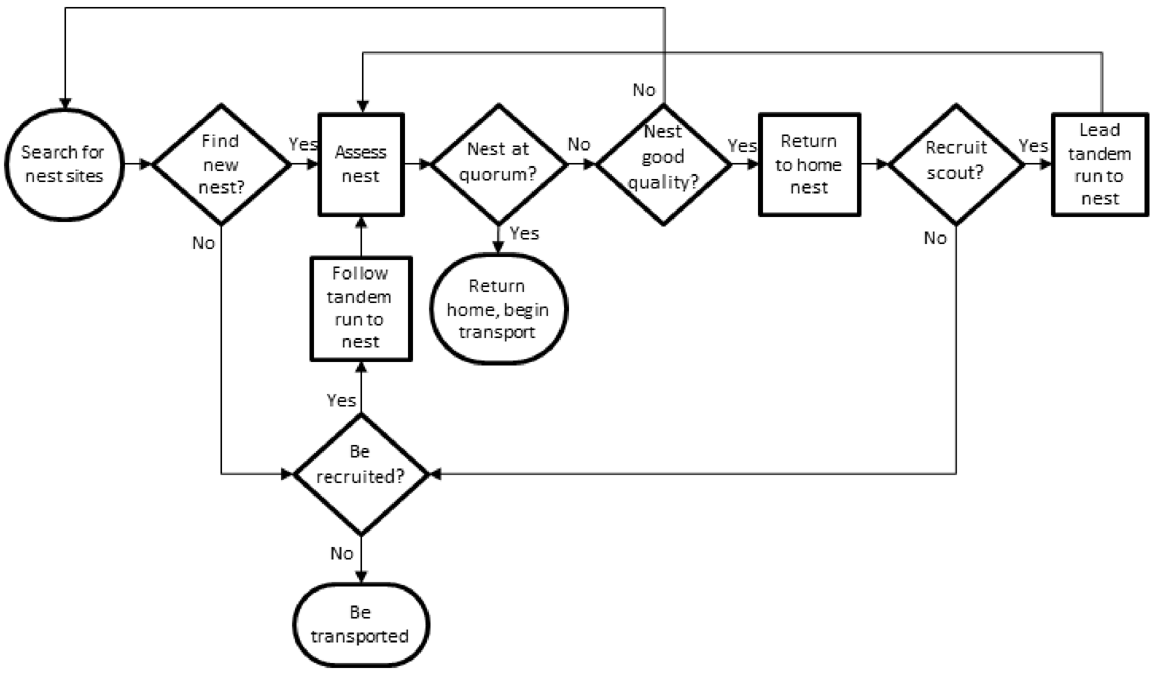


Figure 3.2: A flow chart of the emigration behaviour as observed by Robinson. Begin at "Search for nest sites".

3.5 Natural analogues of SDS

A number of investigations into natural swarm intelligence have described processes that are analogous to various aspects of SDS. These include: Robinson ant nest site selection.

3.5.1 Ant nest site selection

In 2013 J.M. Bishop and the author attended a lecture given by E.J.H. Robinson, *Distributed Decisions: New Insight from Radio Tagged Ants* [87, 88] on the nest-site selection behaviour of ants of the genus *Temnothorax*. In this lecture Robinson showed the observed behaviour of individual ants engaged in the activity of evaluating a candidate site for a new nest.

Individual ants were observed to occasionally perform a search of the area outside of their nest which resembled a random walk. If, during this process, the ant encountered a location that may be a suitable site for a new nest then they would enter the candidate nest site and perform an evaluation. When evaluating a candidate nest site an ant will explore inside the nest and eventually reach an exit, ants did not necessarily visit all of the candidate nest site, and did not follow any fixed path of exploration. If the candidate site was evaluated to be of sufficient quality then the ant would return to their nest, recruit another ant from the swarm, and lead it to the candidate nest site using a technique known as tandem running. While tandem running, the leader ant will move a short distance in the direction of the destination and then wait until the following ant has reached the leading ant, this process is repeated until both ants arrive at the destination. Tandem running not only brings two ants to a destination but also provides an opportunity for the following ant to learn the route from the original nest. When the ants reach the destination both perform the evaluating action and, if candidate nest site is determined to be of sufficient quality, further recruitment. The result is that a strong candidate has an increasing number of ants performing evaluations as the ants which were recruited into the evaluation process begin to perform tandem runs of their own. Robinson observed that if the site was of sufficient quality such that at any time approximately one third of the whole nest had been recruited into the evaluation and recruitment process then those individuals would undergo a behavioural phase change. In the new behaviour, rather than recruiting further ants into the procedure, they would simply lift unrecruited ants (or eggs) in their mandibles and deposit them at the candidate site. Ants transported in this way would not have made the journey to the new nest themselves and no pheromone trail is formed that could be followed, so they would not know the route between the two sites. This means transported ants cannot perform the recruiting behaviour. Transported ants simply adopt the new location as their new home, and play no further part in the process. Once all ants and eggs had either been recruited into the evaluation procedure, or transported to the new site, the nest has been fully migrated to its new site. Similar observations have been made in experiments in which migrations were induced in colonies of the ant *Leptothorax albipennis* [79]. This procedure is described in a flow chart in Figure 3.2.

A similar experiment showed that repeated partial evaluation effectively implemented a measure of the average quality of a resource which fluctuates over time [28]. In the experiment, a colony was presented with two candidate nest sites, one of a quality sufficient to initiate a migration and another which was initially significantly superior, but which was occasionally modified to be of significantly inferior. The observed behaviour was that there was a higher probability that the colony would migrate to whichever nest was superior on average; in cases where the fluctuating nest was superior for half the time and inferior for half the time there was an average probability of approximately 0.5 that the colony would migrate to either nest.

3.5.2 Bee nest site selection

Similar experiments were performed involving the migration behaviour of colonies of bees [96]. Primarily, the behaviour described had many similarities between the behaviour observed in ants. The same process was followed, where scouts would search an area, and on locating a candidate nest, recruit others to also evaluate it, leading to the formation of a large swarm at the best located candidate. The most distinct difference being the method by which one bee would locate a candidate nest site for another bee. The tandem running of ants was replaced by the waggledance, and the transporting behaviour of ants is replaced by a mass migration as a swarm

The waggle dance is a behaviour whereby a bee communicates information on various attributes of a feature outside of the nest to other nearby bees. A feature may be a source of nectar, or other features of interest, but in this context is a candidate nest site. The bee repeatedly walks a path describing a short line in the nest whilst 'waggling' its abdomen; on reaching the end of the line the bee alternately circles back to the beginning turning left or right. The orientation of the line relative to gravity indicates the direction of the feature, the length of the line is indicative of the distance of the feature from the nest, and features such as the speed of waggling, and the number of times the bee walks along the line is indicative of the bee's evaluation of the quality of the feature.

3.5.3 Comparisons between SDS and nest site selection in ants and bees

The similarity between the recruiting behaviours of SDS and recruiting behaviours in ants have been previously recorded [16, 15], though this was in the context of ants foraging for food, rather than selecting a nest site, nevertheless a process which employed tandem running observed. This similarity was used as evidence which suggested that ants do not require the cognitive capacity to directly compare two nest sites as, in SDS, the selection of one hypothesis over another occurs without any direct comparison of hypotheses.

The processes described above have many close analogies with SDS. An ant or bee which is not currently evaluating a nest site and is exploring their search space with a path resembling a random walk is similar to an inactive agent selecting a hypothesis at random. An ant or bee performing a partial evaluation of a candidate nest site by exploring the nest using a path resembling a random walk is similar to an agent partially evaluating a hypothesis by applying a randomly selected microtest. An ant or bee subsequently performing one of two actions after a partial evaluation of a nest site is similar to an agent becoming either active or inactive after performing a partial evaluation. An ant or bee which was satisfied with the partial evaluation of a candidate nest site and recruiting a random ant or bee to the evaluation process is similar to the hypothesis of an active agent being copied by an inactive agent. An ant or bee which was not satisfied with the partial evaluation of a candidate nest site and abandoning the evaluation of the candidate nest site is similar to an inactive agent selecting a new hypothesis. A significant proportion of the ant or bee colony simultaneously evaluating the same candidate nest site is similar to a large number of agents maintaining the same hypothesis.

The experiments that showed the ability of ants to effectively calculate an average score for a candidate nest site which fluctuated in value is analogous to the behaviour of SDS resulting from partial evaluations. Similarly in ant nest site selection the effective score of a candidate nest site is a result of the average result of many partial evaluations. This reflects the behaviour of SDS under certain circumstances, depending on the frequency at which the score of hypothesis fluctuated between two values. If the number of iterations between a fluctuation was large in comparison to the number of iterations required for a cluster to stabilise, for example, once every thousand iterations, then the largest cluster would similarly fluctuate between the superior hypothesis and the stable hypothesis. If the number of iterations between a fluctuation was small in comparison to the number of iterations required for a cluster to stabilise, for example, the hypothesis alternated in quality every time an agent performed a microtest, then the fluctuating hypothesis would be evaluated as if it had a score between the superior and inferior values. This latter case, of frequent fluctuations is what was observed in ants [28], as the time for two fluctuations was always ten minutes, and the time for a population to achieve quorum at a nest site was of the order of one hundred minutes. This is further evidence that SDS models aspects of the nest site selection behaviour of ants, and that this successful evaluation can be conducted by a swarm of agents with no global knowledge.

There are also a few differences between SDS and the process of nest site selection in ants and bees. Candidate nest sites are not discovered with a uniform probability distribution, as ants and bees explore their search space with a path resembling a random walk they is a greater probability of locating candidate nest sites which are closer to their current nest site than candidate nest sites which are further away from the their current nest site. Ants or bees which abandon the evaluation of one candidate nest site are not required to immediately begin evaluation on another candidate nest site, whereas an agent which becomes inactive immediately selects a new hypothesis, either from an active agent, or uniformly at random. Ants or bees which return to the nest recruit other ants or bees they encounter, this is unlike the recruitment action of agents in two ways. First, the selection of other ants or bees is arbitrary rather than uniformly at random. Second, 'active' ants i.e. ants which have recently performed a successful partial evaluation, recruit other ants, whereas in SDS inactive agents assume the hypothesis of other agents. This can be seen as more analogous to active diffusion (D^{ACTIVE}) than passive diffusion (D^{PASSIVE}) as it is individuals which have discovered a potential solution that perform the activity of recruitment, ants and bees which have located a good candidate nest site recruit other ants and bees. The equivalent behaviour for bees is mode analogous to passive diffusion as a small portion of the colony observing a waggledance is similar to a number of inactive agents polling the same active agent.

There is no close analogy in SDS with the behavioural change, when ants or bees moved from a recruiting behaviour to a migration behaviour. The aspect of SDS closest to the behavioural change is the mode of halting, especially when applying a convergence based halting variant, though the significant difference is that in SDS the decision to halt is performed by a single function with access to the state of the entire swarm, whereas the decision of an ant or a bee to change behaviour must be informed by their individual experience.

What remained unclear was the mechanism by which individual ants or bees decided to move from recruiting behaviour to migration behaviour. This ability, called quorum sensing, is essential for any swarm to act as a unit. Individuals may make decisions based on local information, or on contacts with other members of the swarm, but decisions that require mass action also require that a certain proportion of the swarm is in agreement with the decision. Investigations into this area are described in the next section.

3.5.4 Mechanism of quorum sensing in ants

There have been a number of investigations into the mechanism that enables quorum sensing in ants, a simple hypothesis is that ants follow a fixed programme of a certain number of tandem runs, followed by transporting behaviour. However there is evidence against this hypothesis as the tandem running phase has been observed to be completely absent in cases of migration over very short distances [79].

In experiments involving the migration behaviour of colonies of the ant *Leptothorax albipennis* [79] the switch from tandem running to transporting was observed to be determined by the population size at the new nest. Reverse tandem running was also observed, where an ant would lead another from the new nest back to the old nest. It was suggested that this was to induce idle workers at the new site to join them in taking items from the old nest. A potential weakness was identified, that ants from one colony may be transported to two different sites if two separate groups commence transporting at similar times. It was indicated that a requirement for a level of quorum, named a quorum threshold, to be reached before initiating transport would mitigate this possibility. It was concluded that the entire process allowed a colony to indirectly compare the quality of multiple candidate nest sites by initially performing the relatively slow process of tandem running, and later switching to the significantly faster process of transporting at a time determined by the number of ants at the new site. Evidence was provided against the hypothesis that the switch from tandem running to transporting was determined by the distance between the old and new nest; a colony was observed to perform the same number of tandem runs for a site only ten centimetres away from the old nest as for nests sixty centimetres away.

Further experiments investigated the mechanisms underlying ant nest site selection [78]. An apparatus which enabled the experimenter to grant and restrict access to a new nest at will allowed the experimenter to separate ants arriving at the nest from any ants currently evaluating it. The experimenter could therefore determine the importance of direct contact with other ants in contrast with the pheromonal affect of highly populated nests with no direct contact. They found that ants only transitioned to their transporting behaviour after having made direct contact with other ants in the process of evaluating the same nest. They also found that once an ant had begun transporting, it would not continue to evaluate the criteria for quorum. This was achieved by manually removing the ants from nests where transporting had begun and observing that ants would remain transporting when the new nest contained many fewer other ants that would be required to initiate transporting an individual ant. It was observed that the time an ant which has begun transporting spends at a candidate nest drops from two to one minutes, also indicating that it has ceased performing a certain action. To test the hypothesis that quorum sensing was based on the absolute number of ants evaluating a nest, the time before ants begun transporting was compared in two cases with nests of different size. An equal number of ants evaluating two nests of different sizes would lead to a lower rate of direct contact in the larger nest than in the smaller nest due to a lower population density. The ants were seen to reach quorum sooner at the smaller nest hence encounter rate, not the absolute number of ants at a candidate nest, was predictive of quorum sensing [78]. The experimental evidence collected supported two hypotheses for the mechanism of quorum sensing both founded on direct contact with other ants at a candidate nest side. First, average encounter rate, which hypothesised that an ant would sense quorum when the average rate at which they encountered other ants at a candidate nest was required to be over a certain threshold. Second, time for initial contact, which hypothesised that an ant would sense quorum when the average time before their first direct contact with another ant after arriving at a candidate nest was below a certain threshold. Neither hypothesis was determined to be superior to the other, though they are closely linked as any nest with a higher encounter rate would, on average, lead to shorter times before the first encounter.

3.5.4.1 Selection of quorum threshold in ants

Further experiments observing ant migration manipulated the size of the migrating colony. The quorum threshold was observed to be proportional to the number of ants in the colony [20]. When a large colony of ants was artificially split into two, the individual ants were observed to adopt a proportionately reduced quorum threshold, it was hypothesised that this adaptation may be achieved by comparing interaction rates at the new and old nests [20]. Comparison with natural colonies showed that colony size was not the only factor for the selection of a quorum threshold, age could be relevant for example, as well as level of experience of the scouts.

Similar experiments induced migration under various conditions. The emigration behaviour of a nest was compared under benign conditions and harsh conditions. Ants were observed to sense quorum and hence select a new nest site more quickly, and with less discrimination, in harsh conditions than in benign ones. In this context benign conditions had still air and an enclosed nest with adequate humidity and harsh conditions had high wind and an exposed nest with poor humidity. It was shown that a lower quorum threshold determined whether a decision would be taken quickly at the risk of selecting a suboptimal solution, or accurately, at the cost of a more lengthy search process [27, 29]. This showed that the quorum threshold can modulate behaviour depending on the current circumstances.

3.5.5 Mechanism of quorum sensing in bees

In the previously described experiment involving bee migrations (Section 3.5.2) the mechanism of quorum sensing was described in two stages. First individual bees needed to sense that quorum had been reached and second a signal needed to reach the entire colony to induce migration as a swarm.

The experiment compared the time for sensing quorum when evaluating nest sites of equal quality, but at which different numbers of other bees were present. The method by which this was achieved was to compare the speed at which quorum was sensed for a nest of a certain quality and compare it to the speed in a situation where five nests of equal quality were located close together but at such a distance from the swarm that the accuracy of the waggle dance was insufficient to unambiguously indicate a single nest. In this sense a bee indicating a candidate nest site was effectively indicating five sites, from which one would be randomly selected by any bee attempting to locate the indicated nest. As the nests were all of equal quality the recruitment process continued as normal but as the bees were spread randomly across five nests, the population at each nest was on average one fifth of its size. The swarm were observed to take significantly longer before swarming to the location of the new nest in the case with five nests than in the case with just one.

Two important observations were drawn from this experiment. Firstly that the size of the population at the candidate nests is important, and secondly the distinction between quorum and consensus. Where consensus requires that all hypotheses held by a population are in agreement, quorum requires that a single hypothesis is held by a certain threshold number of the population. This experiment showed that consensus was neither necessary or sufficient for a colony to swarm to a candidate location. It was not sufficient as in the case where there were multiple nests of identical quality, all waggle dances would have indicated the same location, yet in this case the colony did not swarm to the new nest site until a sufficient population had gathered at one of the sites. It was not even necessary as the requirement for the population at a candidate nest site to reach a threshold is not contingent on there being no bees indicating other sites.

There are a number of mechanisms by which bees could detect the population level at a nest site. Visual perception is a possibility as bees may be able to spot other bees on the surface of a nest site and near the entrance so long as the light level is sufficient. Alternatively, bees may use direct contact as do ants as bees are seen frequently coming into contact with each other during the process of evaluating a nest site. There may also be olfactory cues, and while this has not been tested for directly it would be analogous to observations of ants utilising pheromones.

Once individual bees had sensed quorum, they would begin to induce the migratory behaviour in the colony with an activity called worker piping. Worker piping is described as 'the prepare-for-takeoff signal' [96], worker bees will perform this vibrational signal while running over and through the bees inhabiting the outer edge of the nest known as the mantle. Piping has a stimulatory effect on a shivering behaviour which leads to an increase in the temperature at the mantle. The mantle is usually cooler than the core so an increase in temperature at the mantle leads to a uniformity of temperature through the nest. Once a sufficient uniform temperature has been achieved the colony will swarm to a location previously encountered by the scout bees.

3.5.6 Mechanism of quorum sensing in bacteria

Some bacteria use quorum sensing as mechanism for regulating gene expression and others use it to coordinate a bioluminescent behaviour, ensuring that the appropriate behaviour is performed by the population in unison [55]. The unison action is important for bacteria as their behaviour, if individually expressed, is often too little to be of any value. In the example of bioluminescence a single bacterium will emit very little light but an entire population fluorescing at once may emit enough light to enable a host organism to hunt more effectively, as is the case with the splitfin flashlightfish *Anomalops katoptron* [36], or to camouflage the host organism, as with the bobtail squid *Euprymna scolopes* [39].

The bacteria, *vibrio fischeri*, which colonise the light organ of the bobtail squid use two quorum sensing systems [47], both are characterised by the secretion of signal molecules by individuals. The frequency with which individuals encounter the signal molecules is an effective proxy for the density of the bacteria population. Under low population density the production of the luminescence signal molecule is suppressed by a negative modulator, minimising any energy being expended on bioluminescence before it is beneficial to the population. Under a medium population density, achieved when the bacteria are attached to the mucus secreted by the light organ of a bobtail squid, a second signal molecule will induce colonisation behaviour, where the bacteria expend energy moving into the light organ itself in an attempt to locate and colonise 'crypts'. Once colonisation behaviour has been successful, the population density inside a crypt is much larger than outside the organ. The higher population density means that a second effect of the colonisation signal molecule becomes significant, it suppresses the production of the negative modulator, effectively stimulating the production of the luminescence signal molecule. A sufficient density of the luminescence signal molecule therefore suggests that colonisation has been successful and that producing light will now be beneficial use of energy.

These two mechanisms show that by the simple method of detecting the presence of a molecule it is possible to effectively detect a related occurrence, which in this case is the presence of a certain population density. Furthermore, detecting subsequent molecules are indicative of very specific occurrences which in this case is the individuals location within a crypt. It is significant that as the second system requires a population density that is not achieved outside of very specific circumstances that even such a single 'bit' of information, that the presence of a molecule is greater than a given threshold, can be taken as a rich message about the state of the world of a bacterium, namely that the successful colonisation of a light organ. This is an indication how, the simple 'boolean' signals between members of any swarm can provide valuable information to an individual as long as the signals are restricted to only being active in specific situations meaningful to the individual. Put another way, the bacteria do not 'need to know' that they have colonised the light organ of a flashlightfish, they simply detect that a situation has occurred in which a specific behaviour may be expressed. It is also worth noting that one of the effects of quorum sensing is to stimulate a behaviour, not by direct induction, but by suppressing the secretion of an inhibitor. While it is true that in the domain of logic a negated false value is equivalent to a true value, in nature there may be reasons why one behaviour is preferred over another, this may also simply be a result of the stochastic nature of evolution.

3.5.7 Comparisons between SDS and quorum sensing in nature

There are some analogues with SDS and the mechanisms of quorum sensing observed in ants, bees and bacteria. Direct contact between ants and bees has been identified as critical in the quorum sensing process, in the terms of SDS two ants encountering each other at their candidate nest site is analogous to one agent polling another agent with the same hypothesis. One mode of diffusion which depends on two agents maintaining the same hypothesis is context-sensitive diffusion ($D^{\text{CONTEXT-SENSITIVE}}$, algorithm 21) the resulting behaviour is that one agent becomes inactive and hence may change their hypothesis, this is superficially analogous to one ant in a candidate nest site encountering another ant and leading it back to the original nest, though it is no evidence that the either ant changes their behaviour as a result.

It has been determined that the distance between candidate nests and the original nest did not directly influence the mechanism of quorum sensing, as the same number of tandem runs were observed between candidate nest sites of difference distances from the original nest. Initially this appears to be analogous to the behaviour of SDS as the evaluation hypotheses is not affected by the position of the hypothesis in the search space. However, it should be noted that tandem running is a relatively slow process and so any number of tandem runs over a longer distance will take longer than the same number of tandem runs over a shorter distance. The closest analogy to this in SDS would be in the situation where parallel iteration is employed and certain hypotheses took much longer to partially evaluate than others, in this case clusters would still form at various hypotheses after the same number of evaluations, but one hypothesis may form a cluster sooner due to the evaluations being quicker.

The closest analogue between SDS and quorum sensing is its mode of halting as both are mechanisms whereby the search procedure yields its result. Quorum sensing can be interpreted as mode of halting enacted by the individuals which make up a swarm. The mechanisms of natural quorum sensing described above are all guided by the level of population at a candidate nest site, or in the case of bacteria, in the local area. These mechanisms are related to some mode of halting methods described in section 3.4.5, p.106.

In the simplest sense, population at a candidate nest site may be modelled by global activity threshold halting (H^{ACTIVITY}, algorithm 28) but this considers all activity to be pertaining to the same hypothesis. A better model is largest cluster size halting (H^{LARGEST}, algorithm 27) as population at a single candidate nest site is therefore modelled as the size of a cluster at a single hypothesis. This variant is not a good model of observed behaviour on its own as the ants or bees were not observed to immediately change their behaviour once a sufficient population had been detected. Another variant which also models some of the behaviour of natural quorum sensing is stable global activity halting (H^{STABLE}, algorithm 31) as this method requires that a hypothesis forms a stable cluster, and that the cluster remains stable for a given number of iterations. This halting method is therefore analogous to the behaviour of an ant or a bee requiring that their candidate nest site has a sufficiently high population for a number of visits before they switch behaviour. This variant is also not a good model of observed behaviour on its own as an instance of zero, or very low, global activity is considered stable, but represents a situation which would not induce migration in ants or bees. The closest model to natural quorum sensing of the variants described in section 3.4.5 would therefore be a combination of both algorithms, so both requirements would be modelled. Those being a sufficiently high population which is stable over time. This variant may be denoted as $H^{\text{AND}}(H^{\text{LARGEST}}, H^{\text{STABLE}})$

The most significant difference between mode of halting and quorum sensing in nature is that the quorum sensing is an action that results from the interaction of individuals, whereas each variant of mode of halting results from an analysis of the state of the entire swarm. Rather than being enabled by the individuals of the swarm, it is a separate process, outside of the swarm. As such, certain aspects of individual behaviour are not represented in SDS. There is no time when an agent in SDS switch into a new behaviour, or modify their internal state so as to be more or less likely to switch into the behaviour in the future. These aspects of quorum sensing will be added into a variant of SDS and explored in the next chapter.

Chapter 4

Local termination criteria for SDS

This section details some investigations into the robustness of the distributed decision making processes observed in ants and bees. SDS is used as a model of the distributed decision making process as there are significant similarities. A variant of SDS is introduced which introduces the possibility for agents to switch behaviours in a manner similar to ants switching their behaviours from tandem running to transporting, and bees switching their behaviours from waggledancing to piping. The new behaviour is most analogous to the transporting behaviour of ants as individual agents impose their decision on other agents, effectively removing them from the search process. As this behaviour results in a decrease in the number of agents in the swarm, the variant is called reducing SDS.

4.1 Introduction of reducing SDS

The key difference between reducing SDS and standard SDS is that a new behaviour is introduced, to reflect the behavioural switch observed in migrating ants. This behaviour removes other agents from the swarm. This effectively terminates the action of removed agents, hence the new behaviour is named 'terminating'. The terminating behaviour is analogous to the transporting behaviour described in section 3.5.1 in that terminating agents no longer evaluate their hypothesis and proceed to impose their decision on other agents in the swarm. The most important factor in the method by which agents become terminating is the cluster size at their hypothesis as population size at a candidate nest site was determined to be the most important factor for ants (Section 3.5.4) and bees (Section 3.5.5).

The detection of population size at a candidate nest site by ants was determined to be enabled though direct contact. Therefore agents commence terminating in reducing SDS when an active agent polls another agent which maintains the same hypothesis, as this is analogous to a chance encounter between two ants evaluating the same candidate nest site. As ants which have commenced transporting perform no further evaluations of their candidate nest site, agents which have commenced terminating perform no further action during the test phase (Algorithm 33). Both boolean testing and ant nest site selection appear to rely on a partial evaluation followed by a decision to continue evaluating the same potential solution determined by whether the partial evaluation was satisfactory. Therefore, no modifications were made to the mode of testing as boolean testing is already an acceptable analogue of observed ant and bee behaviour. Instances of reducing SDS halt once all agents have been removed from the swarm or once all agents are terminating, as this state is analogous to the situation where all ants have been transported to a new nest site (Algorithm 34). When a terminating agent removes another agent the removed agent is considered to be permanently maintaining the hypothesis of the terminating agent. To calculate the size of a cluster in reducing SDS therefore requires the summing of the number of active agent maintaining a particular hypothesis plus the number of agents that were removed by agents maintaining that hypothesis.

Two variants of reducing SDS have been developed and are detailed in the following sections, both variants use the same mode of testing and the same mode of halting, these are detailed in algorithms 33 and 34 respectively. Note that T^{REDUCING} receives a mode of testing (T) as a parameter, this mode of testing is performed as usual, but only if the agent is not terminating.

Algorithm 33 T^{REDUCING} — Reducing testing

```
1: function $T^{\text{REDUCING}}(T)$
2:     function $T'$(agent)
3:         if agent is not terminating then
4:             $T$(agent)
5:     return $T'$
```

Algorithm 34 H^{REDUCING} — Empty or all terminating swarm halting

```
1: function $H^{\text{REDUCING}}$(swarm)
2:     function $H'$( )
3:         empty ← swarm size = 0                                         ▷ empty $\in \mathbb{B}$
4:         all terminating ← all agents in swarm are terminating          ▷ all terminating $\in \mathbb{B}$
5:         halt ← empty *or* all terminating                              ▷ halt $\in \mathbb{B}$
6:         return halt
7:     return $H'$
```

4.2 Independent reducing SDS

The first variant of reducing SDS to be developed is independent reducing SDS, its mode of diffusion is intended to be as close an analogy of observed ant behaviour as possible within the framework of SDS. In independent reducing diffusion an agent polls another agent from the swarm and performs one of five behaviours, each one represents a behaviour observed in ant nest site selection.

1. If both agents are inactive each agent generates a new hypothesis. This is analogous to two ants neither of which have located a candidate nest site encountering each other and continuing to search in a random fashion.
2. If one agent is active and the other is inactive, the inactive agent assumes the hypothesis of the active agent. This is analogous to one ant leading another ant to a candidate nest site by tandem running.
3. If a terminating agent encounters another agent the other agent is removed from the swarm, if both agents are terminating the removed agent is chosen arbitrarily. This is analogous to one ant transporting another ant to a candidate nest site.
4. If both agents are active and maintaining the same hypothesis they both become terminating. This is analogous to two ants encountering each other at a candidate nest site and commencing transporting.

As with dual polling (a combination of passive polling and active polling, described in section 3.4.2.2) both active and inactive agents take action during diffusion but unique to independent reducing SDS both the polled agent and the polling agent may change state. Independent reducing diffusion is shown in algorithm 35.

One further feature of independent reducing SDS is that it implements asynchronous

Algorithm 35 $D^{\text{INDEPENDENT}}$ — Independent reducing diffusion

```
1:  function $D^{\text{INDEPENDENT}}(DH, \text{swarm})$
2:      function $D'(\text{agent})$
3:          polled ← random agent in swarm
4:          if both agents are inactive then
5:              hypothesis of agent ← $DH()$
6:              hypothesis of polled ← $DH()$
7:          else if one agent is inactive and other is active but not terminating then
8:              hypothesis of inactive agent ← hypothesis of active agent
9:          else if exactly one agent is terminating then
10:             hypothesis of non-terminating agent ← hypothesis of terminating agent
11:             non-terminating agent is removed from the swarm
12:         else if both agents are terminating then
13:             arbitrarily chosen agent is removed from the swarm
14:             hypothesis of removed agent ← hypothesis of other agent
15:         else if hypothesis of agent = hypothesis of polled then
16:             both agents become terminating
17:     return $D'$
```

iteration as it is a closer analogy to the action of individual ants, though as noted in section 3.4.1.1 this is not expected to result in behaviour significantly different to that of synchronous iteration. Independent reducing SDS may therefore be denoted as

$$\text{SDS}\left(I^{\text{ASYNCHRONOUS}}\left(D^{\text{INDEPENDENT}}(DH^{\text{STANDARD}}), T^{\text{REDUCING}}(T^{\text{STANDARD}})\right), H^{\text{REDUCING}}\right)$$

No claims are made that this procedure duplicates real world behaviour, other than to posit that the requirements of an agent are broadly within the capacities of natural ants. An agent only requires access to global state when polling another agent, which is taken to be a fair analogy to a chance encounter between any two ants moving around in the world. Similarly, the requirement that two agents are able to compare their hypotheses for equality is taken to be a fair analogy to a pair of ants encountering each other whilst evaluating the same nest. Furthermore, interaction is bi-directional, no distinction is made between the polling and polled agent. When two terminating agents encounter each other, the agent to be removed must be chosen randomly, though in this case both agents will be functionally identical so which agent is chosen for removal is immaterial. This decision to remove one agent, rather than both, is to maintain similarity with the behaviour of ants, where one ant will transport the other to the new nest site.

Two experiments were performed to examine the stability of this algorithm, both employed a model inspired by the DDSM which models search spaces with a two values representing the optimal hypothesis score (α) and homogeneous background noise (β). While this model appears restrictive it captures the characteristic behaviour of reducing SDS in the large majority of cases. In any search space there will be a maximum value which represents the score of the optimal hypothesis, it is possible that there are multiple hypotheses with the same maximum score but these can be considered to be equivalent and convergence to either hypothesis can be considered a success. In most search spaces there will also be multiple hypotheses that are sub-optimal, the DDSM is based upon the observation that the effect of these hypotheses can be modelled by a single value representing homogeneous background noise [62]. In some search spaces there may also be a small number of distractors, sub-optimal hypotheses with a score strong enough to form stable clusters but still distinctly lower than that of the optimal hypothesis. While the DDSM does not appear to model the effect of distractors directly, it can still be represented. In the presence of distractors one of three cases my occur, i) a cluster begins to form at the optimal hypothesis before the forming at a distractor, ii) a cluster begins to form at the distractor before forming at the optimal hypothesis, iii) clusters form at the optimal hypothesis and distractor simultaneously. The third case can be discounted as unlikely as the time taken for an agent to assume specific hypothesis is much longer than the time taken for a cluster to reach stability once it has begun to form. This is because cluster formation is best described as rapid process dominated by a positive feedback loop [66], and hypothesis selection is a process that acts in linear time with respect to the number of inactive agents over the size of the search space. In the first case the effect of the distractor is small, a cluster will form at the optimal hypothesis exactly as if the distractor did not exist as there will be no active agents maintaining the distractor hypothesis. The only effect of the distractor is that the stable cluster size at the optimal hypothesis may be negligibly smaller as the some occasional activity at the distractor will effectively increase the amount of homogeneous background noise otherwise present. In the second case the algorithm can be expected to converge to the distractor at which point there will be a stable value for global activity. This state, where a stable cluster has formed at a distractor and not the optimal hypothesis can then be accurately modelled with an increase in the value of homogeneous background noise such that the global activity caused by active agents at the noise hypothesis matches the global activity of the algorithm after convergence to the distractor.

4.2.1 Robustness of independent reducing SDS

The first experiment aimed to demonstrate the probability of independent reducing SDS converging to the optimal hypothesis over a range of conditions which represented a sample of all valid search spaces. An instance of Independent reducing SDS was performed with a swarm of 10^3 agents against search spaces of various size and with various values for optimal hypothesis score and homogeneous background noise. The search space sizes were $10^2, 10^3, 10^4, 10^5$, and the values for α and β were every value in the interval $[0, 1]$ where $\alpha > \beta$ for a total of 55 unique search spaces. A search was performed on each combination of α, β, and search space size, and each search was repeated 100 times. Each search used a halting method which combined the reducing halting criteria (algorithm 34) with fixed iteration count halting (algorithm 11), $H^{\text{OR}}(H^{\text{REDUCING}}, H^{\text{FIXED}}(10001))$. Fixed iteration count halting was used to avoid cases that would otherwise not halt within a practical time.

After each halt the cluster sizes were analysed as follows:

- If the number of iterations was greater than or equal to 10001 the search was considered a failure.
- If the search was not a failure and the largest cluster was at the optimal hypothesis the search was considered to have converged successfully.
- If the search was not a failure and the largest cluster was **not** at the optimal hypothesis the search was considered to have converged to a noise hypothesis.

4.2.1.1 Results

Some results of this experiment are shown in Table 4.1. The full results for this experiment are shown in Table A.2 (Section A.2, p.194).

The robustness of independent reducing SDS was significantly affected by the search space size. Successful convergences occurred in 49.27% of all cases, occurring less frequently in larger search spaces 76.24%, 60.89%, 40.47%, 19.49% for search spaces of size 10^2, 10^3, 10^4, 10^5 respectively, a correlation of -0.85 $(P = 0.1)$. Failed convergences occurred in 3.20% of all cases, occurring more frequently in larger search spaces 0.00%, 1.82%, 4.29%,

S	success	P(success)	noise	P(noise)	fail	P(fail)	total
10^2	4193	0.762	1307	0.238	0	0.0	5500
10^3	3349	0.609	2051	0.373	100	0.018	5500
10^4	2226	0.405	3038	0.552	236	0.043	5500
10^5	1072	0.195	4061	0.738	367	0.067	5500

Table 4.1: Robustness of independent reducing SDS with varing search space size. Showing search space size (S), the number of successes (success), convergences to noise hypotheses (noise), and failures (fail), the probability of each result for a given search space size and the total number of runs (total).

6.67% for search spaces of size 10^2, 10^3, 10^4, 10^5 respectively, a correlation of 0.85 $(P = 0.2)$. Convergences to noise solutions occurred in 47.53% of all cases, occurring more frequently in larger search spaces 23.76%, 37.29%, 55.24%, 73.84% for search spaces of size 10^2, 10^3, 10^4, 10^5 respectively, a correlation of 0.85 $(P = 0.2)$. Correlations calculated using the Pearson correlation coefficient for testing non-correlation [101, `scipy.stats.pearsonr`].

There was a significant difference in convergence between noiseless search spaces $\beta = 0$, compared with noisy search spaces $\beta > 0$. Out of the 18000 noisy search spaces, 7543 cases (41.91%) converged successfully, 0 cases were recorded as failures. The mean average number of iterations before a successful convergence was 30.63 with a standard deviation of 30.18. Out of the 4000 noiseless search spaces there was a significantly higher chance of successful convergence, with 3297 cases (82.42%) converging successfully. The mean average number of iterations before a successful convergence in the noiseless case was 537.75 with a standard deviation of 1368.84. Welch's t-test for equal means [101, `scipy.stats.ttest_ind_from_stats`] calculates the ratios of the means (t) to be 21.27 $P = 3.5 \times 10^{-94}$, showing that the convergence time between noiseless search spaces and noisy search spaces to be significantly different.

These results show that independent reducing SDS can converge, but it is highly sensitive to noise. The effect of noise is more pronounced in cases where the search space size (S) is greater than the number of agents (A). This is expected as the probability of an agent locating the optimal hypothesis is $\frac{1-\text{activity}}{S}$ and the probability of an agent maintaining a noise hypothesis commencing terminating is $\frac{c_\beta}{A}$ where c_β is the size of the cluster at the noise hypothesis. In cases where $S >> A$ it is expected that some agents will commence terminating before a cluster forms at the optimal hypothesis so long as $\beta > 0$. Furthermore, there is a higher chance of a cluster forming earlier at the optimal hypothesis when $\alpha = 1$ as there is no chance that the cluster will collapse once a single agent has assumed the optimal hypothesis.

4.2.2 Diffusion of termination in independent reducing SDS

While it has been shown that consensus is neither necessary or sufficient for quorum sensing (Section 3.5.5), once some agents have sensed quorum it is essential that this is communicated to the rest of the swarm to achieve consensus as soon as possible. This is to reduce the number of agents which would otherwise perform further redundant evaluation, and to avoid the case where other agents sense quorum at a separate hypothesis. The second experiment therefore aimed to examine the time between a single agent locating the optimal hypothesis and the process halting. This allows the convergence behaviour to be examined separately from the effect of noise enabling simultaneous convergence to other hypotheses.

As with the previous experiment, an instance of independent reducing SDS was performed with a swarm of 10^3 agents against every search space described by all the values for α and β in the interval $[0, 1]$ where $\alpha > \beta$. Each search was repeated 100 times. In a departure from the previous experiment the search space size was not varied, instead the model was modified to reflect a search space of infinite size. This was to ensure that the only mechanism by which an agent could assume the optimal hypothesis was by being the agent selected to initially maintain the optimal hypothesis or by receiving it via diffusion. Using equation 3.8, the time to first hit (TH) in an infinite search space can be seen to be infinite, therefore the mode of hypothesis selection employed (DH^{INFINITE}) returned only the noise hypothesis, this reduces the probability of an agent selecting the optimal hypothesis to zero, and hence TH $= \infty$. To enable the convergence behaviour to be examined a single agent was initialised as active and maintaining the optimal hypothesis, furthermore only agents which were maintaining the optimal hypothesis were allowed to become terminating. Each search used a halting method $H^{\text{OR}}(H^{\text{REDUCING}}, H^{\text{COLLAPSE}})$ which combined the reducing halting criteria (algorithm 34) with a new halting method (algorithm 37) which halts if no agents were maintaining the optimal hypothesis. Standard SDS was also performed with the same DH but halting with $H^{\text{FIXED}}(1001)$ to provide a comparison for the stability of swarms.

Algorithm 36 DH^{INFINITE} — Noise hypothesis selection

```
1: function $DH^{\text{INFINITE}}$
2:     function $DH'$( )
3:         return noise hypothesis
4:     return $DH'$
```

Algorithm 37 H^{COLLAPSE} — Optimal hypothesis cluster collapse halting

```
1: function $H^{\text{COLLAPSE}}$(swarm)
2:     function $H'$( )
3:         opt $\leftarrow$ number of active agents maintaining the optimal hypothesis
4:         halt $\leftarrow$ opt $= 0$                                  $\triangleright$ halt $\in \mathbb{B}$
5:         return halt
6:     return $H'$
```

After each halt the swarm was analysed as follows

- If no agents maintained the optimal hypothesis the search was considered a failure.
- If the search was not a failure the search was considered to have converged successfully.

4.2.2.1 Results

Independent reducing SDS was shown to rapidly diffuse the detection of quorum through the swarm in cases where the cluster did not collapse. The cluster did not collapse in 2482 out of 5500 cases (45%), and halted in an average of 33.36 iterations, with a standard deviation of 52.689. The probability of successfully halting was strongly correlated with $\alpha - \alpha_{\text{min}}$ (the amount the optimal hypothesis score was greater than the minimum convergence criteria for standard SDS), with a correlation of 0.883 ($P = 4.54 \times 10^{-19}$). The average number of iterations before halting was strongly negatively correlated with $\alpha - \alpha_{\text{min}}$, with a correlation of -0.796 ($P = 3.91 \times 10^{-11}$). The standard deviation of the number of iterations before halting was moderately negatively correlated with the probability of successfully halting, with a correlation of -0.581 ($P = 2.27 \times 10^{-5}$)

The results for independent reducing SDS were similar for standard SDS under the same conditions. As with independent reducing SDS, the probability of standard SDS successfully halting was strongly correlated with $\alpha - \alpha_{\text{min}}$, with a correlation of 0.890 ($P = 1.04 \times 10^{-19}$).

		Correlation	P
$\alpha - \alpha_{\text{min}}$	SSDS_c	0.890	1.04e-19
$\alpha - \alpha_{\text{min}}$	IrSDS_c	0.883	4.54e-19
$\alpha - \alpha_{\text{min}}$	IrSDS_μ	-0.796	3.91e-11
IrSDS_c	SSDS_c	0.978	8.60e-38
IrSDS_c	IrSDS_σ	-0.581	2.27e-05

Table 4.2: Correlations between the difference between α and α_{min}, the count (subscript c) of successful convergences, mean average (subscript μ) and standard deviation (subscript σ) of number of iterations before a successful convergence, for independent reducing SDS (IrSDS) and standard SDS (SSDS).

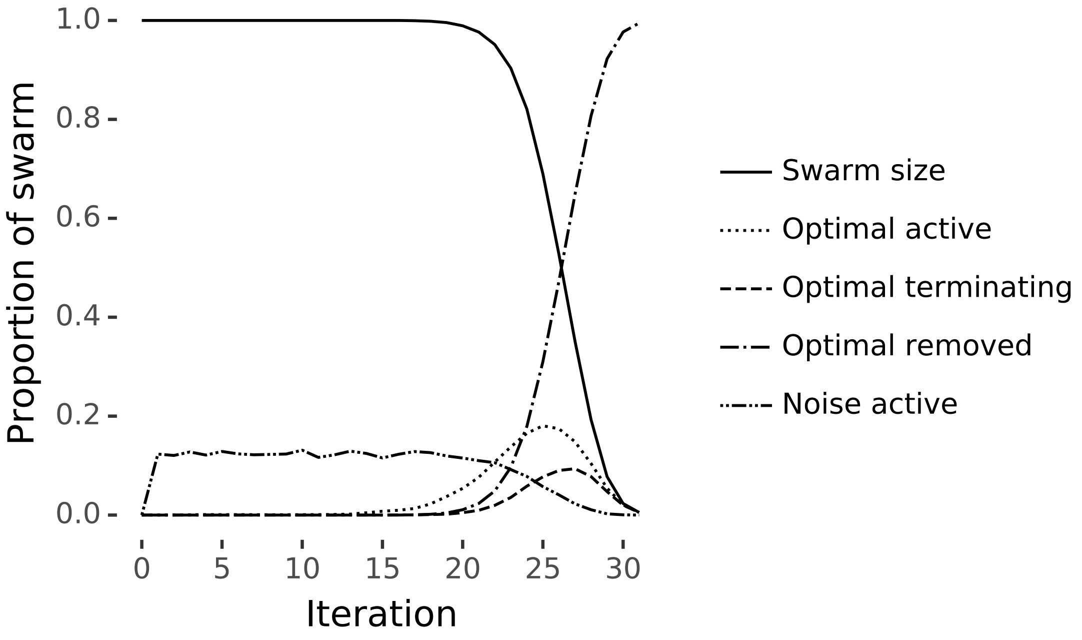
Independent reducing SDS and standard SDS successfully halted under silmilar conditions, the correlation between the probability of independent reducing SDS successfully halting and the probability of standard SDS successfully halting was 0.978 ($P = 8.60 \times 10^{-38}$).

These correlations are shown in table 4.2, the full results for this experiment are shown in Table A.1 (Section A.1, p.191).

4.2.2.2 Plot of swarm evolution

An instance of independent reducing SDS was performed with a swarm of 10^3 agents against a search space of size 10^4 where $\alpha = 0.75$ and $\beta = 0.125$.

This plot helps demonstrate that the number of terminating agents decreases after an initial increase. This occurs as although terminating agents may never stop terminating they may be removed from the swarm altogether. Initially the number of agents becoming terminating and the number of terminating agents being removed are both zero. Once a cluster of active agents has formed, some agents become terminating but there are still few terminating agents such that few terminating agents are being removed, hence the number of terminating agents increases. Eventually the number of remaining agents which are active but not terminating reduces and the number of terminating agents increases so the number of agents commencing terminating equals the number of terminating agents being removed, at this point the number of terminating agents is stable. Once the probability of a terminating agent polling another terminating agent is greater than the probability of an



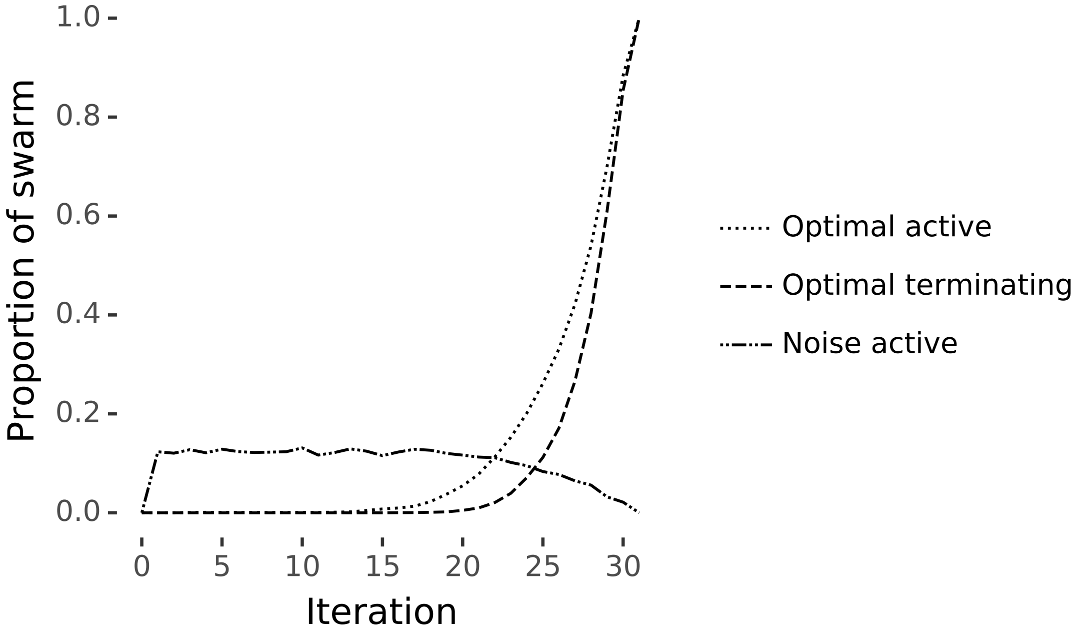


Figure 4.1: Cluster size evolution over time for independent reducing SDS. The x-axis shows iterations. In the upper graph the y-axis shows cluster size as a proportion of the total number of agents, in the lower graph the y-axis shows cluster size as a proportion of the agents which have not yet been removed. The graphs show an accelerating growth in the number of agents at the optimal hypothesis followed by a similar growth of terminating agents at the optimal hypothesis until the entire swarm is terminating or removed.

active but not terminating agent polling an active agent the number of terminating agents begins to decrease. Finally all agents are terminating or removed and the algorithm halts.

4.2.3 Discussion

These experiments show that independent reducing SDS behaves similar to standard SDS in many ways. The conditions under which stable clusters may form is similar for independent reducing SDS and standard SDS as seen in the strong correlations between the conditions in which they do and do not exhibit cluster collapse. Once a cluster has formed the behaviour of independent reducing SDS is similar to standard SDS with a greedy halting method, with independent reducing SDS halting on average after only 33.36 iterations after the first agent became active. This greedyness can be seen as independent reducing SDS halts in some cases where the conditions would not permit a stable cluster to form under standard SDS, this may occurr as an agent remaining active for only a single iteration may poll itself in $D^{\text{INDEPENDENT}}$ and become terminating. This behaviour leads to independent reducing SDS being highly sensitive to the effect of homogeneous background noise in cases where $S >> A$ as even a small amount of activity is likely to cause an agent to begin terminating before an agent assumes the optimal hypothesis. Independent reducing SDS can therefore be described as being highly reliant on the optimal hypothesis being located before any other moderately scoring hypothesis, this demonstrates that it cannot reasonably be described as indirectly comparing hypotheses as has been observed in ant nest site selection [28].

These features mean that independent reducing SDS may be an impractical algorithm for all cases other than those where any hypothesis that has a score greater than zero is considered to be a suitable solution, and where the swarm size is at least as big as the search space size. However, the feature for which independent reducing SDS was developed was termination using only local information, and this has been adequately demonstrated, the switch in behaviour from evaluating to terminating diffuses rapidly through the swarm and within a small number of iterations the entire swarm is either terminating or removed. This is notable as independent reducing SDS uses individual actions which are inspired by natural observations and arguably within the capacities of ants and bees. Furthermore the mode of halting arises from individual actions, not an external algorithm which is also analogous to that in ants. This process of detecting quorum and diffusing it through the swarm is demonstrated to be feasible though highly greedy and sensitive to the probability of a minimally acceptable candidate being located before the optimal. Mitigating the effect of noise by introducing the requirement for multiple reinforcements of a hypothesis are therefore be investigated in the next section.

Hypotheses are not being directly compared, as the process converges to the first hypothesis which maintains a cluster long enough for a single agent to become active. Once an agent is active the number of terminating agents increases monotonically at a rate proportional to the number of terminating agents.

4.2.3.1 Analogies with nature

In the natural case this implies that natural swarms will need to use a method more resistant to the effect of noise if there is a significant probability of certain candidate nest sites not being located.

Independent reducing SDS models recruiting as one agent copying the hypothesis of another and transporting as one agent removing an agent from the swarm. Both of these actions can be performed in a single iteration by two interacting agents, this ignores the observation that transporting is three times faster than tandem running [29, 28, 77]. One way this could be represented in $D^{\text{INDEPENDENT}}$ is by adding a random factor such that agents only copy hypotheses in one in three cases, a feature similar to D^{HERMIT}. Alternatively this could be represented by adding a feature similar to $D^{\text{MULTI-DIFFUSION}}$ for terminating agents such that they poll, and potentially remove, multiple agents from the swarm. These mechanisms are likely to affect the range of search spaces in which reducing SDS will converge, or the speed thereof, but are not investigated here.

In independent reducing SDS terminating agents have the ability to remove other terminating agents, it is not clear from the referenced experiments whether this occurs in nature. Even if it is a rare occurrence it is not clear whether this would be due to a decision of the individual, or emergent form the fact that transporting agents spend less time at the original nest and so are unlikely to be encountered for transporting.

4.3 Confirmation reducing SDS

As the aim of independent reducing SDS was to develop a variant of SDS which represent the observed behaviour of ant nest site selection as closely as possible, it employs a mode of diffusion that is unlike any previously modelled variant. A new variant is introduced that aims to be as close as possible to standard SDS such that it may benefit from existing models and intuition while also exhibiting the characteristic behaviours of independent reducing SDS. The variant retains the behaviour of independent reducing SDS where agents become terminating by a method where the quality of their hypothesis is confirmed by contacting other agents in the same cluster, it is therefore named confirmation reducing SDS.

Confirmation reducing SDS implements the terminating and reducing effects of Independent reducing SDS but implements the synchronous iteration of standard SDS and a mode of diffusion more closely related to passive diffusion. The diffusion behaviour includes the behaviour of alias of D^{PASSIVE} with the additional conditions that inactive agents may be removed from the swarm and active agents may begin terminating. An active agent begins terminating when the polled agent is also terminating and both agents share a hypothesis. An inactive agent is removed from the swarm when they poll a terminating agent. The mode of diffusion of confirmation reducing SDS can be seen in Algorithm 38.

Algorithm 38 $D^{\text{CONFIRMATION}}$ — Confirmation reducing diffusion

```
1:  function $D^{\text{CONFIRMATION}}(DH, \text{swarm})$
2:      function $D'$(agent)
3:          polled ← random agent in swarm
4:          if agent is active then
5:              same hypotheses ← hypothesis of agent = hypothesis of polled
6:              if polled is active *and* same hypotheses then
7:                  agent becomes terminating
8:          else
9:              if polled is active then
10:                 if polled is terminating then
11:                     agent is removed from swarm
12:                 else
13:                     hypothesis agent ← hypothesis of polled
14:             else
15:                 hypothesis of agent ← $DH()$
16:     return $D'$
```

Confirmation reducing SDS may be denoted as

$$\text{SDS}\left(I^{\text{SYNCHRONOUS}}\left(D^{\text{CONFIRMATION}}(DH^{\text{STANDARD}}), T^{\text{REDUCING}}(T^{\text{STANDARD}})\right), H^{\text{REDUCING}}\right)$$

As with independent reducing SDS three experiments were performed. The first experiment investigates the robustness of confirmation reducing SDS over various values for α, β and S. The second experiment investigates the diffusion of the halting behaviour through the swarm in the case where the termination behaviour is suppresed at any hypothesis other than the optimal hypothesis. The third experiment plots the evolution of the swarm in an example case of successful convergence.

4.3.1 Robustness of confirmation reducing SDS

The first experiment aimed to demonstrate the probability of that the probability of confirmation reducing SDS converging to the optimal hypothesis was similar to that of independent reducing SDS over a range of conditions. As with the equivalent experiment for independent reducing SDS, an instance of confirmation reducing SDS was performed with a swarm of 10^3 agents against search spaces of various size and with various values for optimal hypothesis score and homogeneous background noise. The search space sizes were $10^2, 10^3, 10^4, 10^5$, and the values for α and β were every value in the interval $[0, 1]$ where $\alpha > \beta$ for a total of 55 unique search spaces. A search was performed on each combination of α, β, and search space size, and each search was repeated 100 times. Each search used employed combined halting of $H^{\text{OR}}(H^{\text{REDUCING}}, H^{\text{FIXED}}(10001))$.

After each halt the cluster sizes were analysed as follows:

- If the number of iterations was greater than or equal to 10001 the search was considered a failure.
- If the search was not a failure and the largest cluster was at the optimal hypothesis the search was considered to have converged successfully.
- If the search was not a failure and the largest cluster was **not** at the optimal hypothesis the search was considered to have converged to a noise hypothesis.

S	success	P(success)	noise	P(noise)	fail	P(fail)	total
10^2	4866	0.885	634	0.115	0	0.0	5500
10^3	3981	0.724	1496	0.272	23	0.004	5500
10^4	2617	0.476	2676	0.487	207	0.038	5500
10^5	1104	0.201	4040	0.735	356	0.065	5500

Table 4.3: Robustness of confirmation reducing SDS with varing search space size. Showing search space size (S), the number of successes (success), convergences to noise hypotheses (noise), and failures (fail), the probability of each result for a given search space size and the total number of runs (total).

4.3.1.1 Results

Some results of this experiment are shown in Table 4.3. The full results for this experiment are shown in Table A.2 (Section A.2, p.194).

As with independent reducing SDS, the performance of confirmation reducing SDS was significantly affected by the search space size.

4.3.2 Diffusion of termination in confirmation reducing SDS

The second experiment aimed to demonstrate that the diffusion of the termination behaviour in confirmation reducing SDS was similar to that of independent reducing SDS.The same experiment for independent reducing SDS was performed with confirmation reducing SDS, employing DH^{INFINITE} and combined halting $H^{\text{OR}}(H^{\text{REDUCING}}, H^{\text{COLLAPSE}})$.

4.3.2.1 Results

There was a strong correlation 0.987 ($P = 9.30 \times 10^{-44}$) between the probability of confirmation reducing SDS successfully converging and standard SDS successfully converging. There was also a strong correlation 0.986 ($P = 2.82 \times 10^{-43}$) between the probability of confirmation reducing SDS successfully converging and independent reducing SDS successfully converging. The correlation between the probability of successful convergence and the standard deviation of the number of iterations before halting was not strong, and with

		Correlation	P
$\alpha - \alpha_{\text{min}}$	SSDS_c	0.890	1.04e-19
$\alpha - \alpha_{\text{min}}$	CrSDS_c	0.906	1.75e-21
$\alpha - \alpha_{\text{min}}$	CrSDS_μ	-0.765	9.69e-11
CrSDS_c	SSDS_c	0.987	9.30e-44
CrSDS_c	CrSDS_σ	-0.159	2.70e-01
CrSDS_c	IrSDS_c	0.986	2.82e-43

Table 4.4: Correlations between the difference between α and α_{min}, the count (subscript c) of successful convergences, mean average (subscript μ) and standard deviation (subscript σ) of number of iterations before a successful convergence, for confirmation reducing SDS (CrSDS), independent reducing SDS (IrSDS) and standard SDS (SSDS).

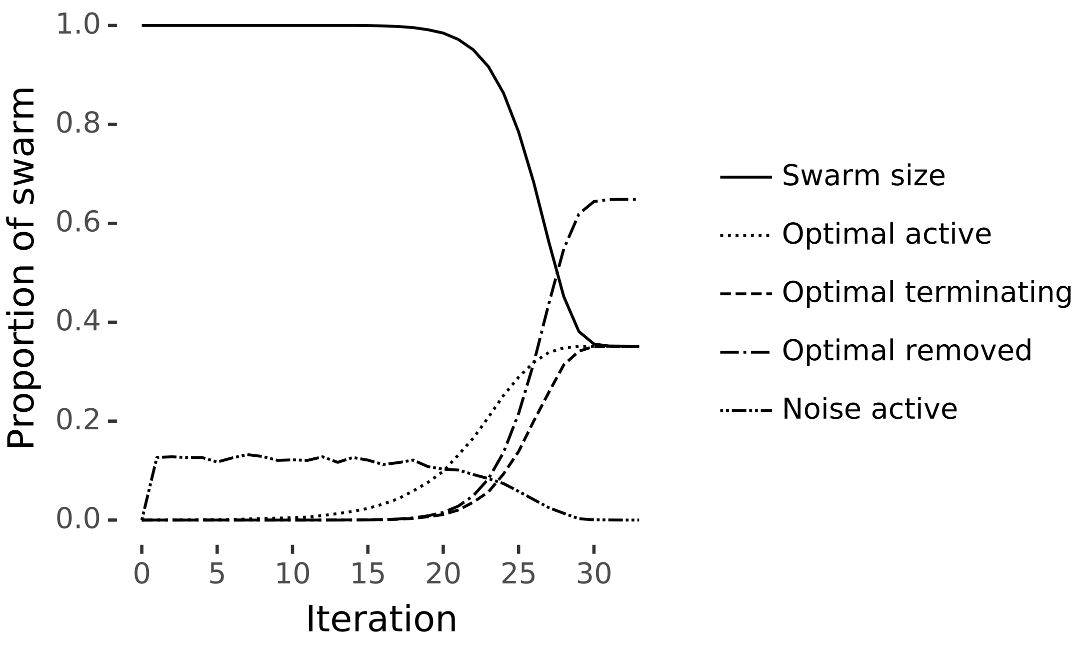
a poor P-value.

This is taken as evidence that confirmation reducing SDS behaves in a was similar to both standard SDS and independent reducing SDS.

4.3.2.2 Plot of swarm evolution

An instance of Confirmation reducing SDS was performed was a swarm of 10^3 agents against a search space of size 10^4 where $\alpha = 0.75$ and $\beta = 0.125$.

The plots of swarm evolution for confirmation reducing SDS (Figure 4.2) has some similarities and some differences with independent reducing SDS (Figure 4.1). The upper plot (which shows the states of agents as a proportion of the original swarm size) for both confirmation reducing SDS and independent reducing SDS shows swarm size decreasing in the shape of an S-curve, the proportion of removed agents increasing in the shape of an S-curve, and the proportion of agents at noise solutions as initialy stable and then reducing to zero. There is less agreement between the evolution of the proportion of agents active at the optimal hypothesis and the proportion of agents which are terminating. In independent reducing SDS these proportions eventually reach zero whereas under confirmation reducing SDS these proportions stabilise between 0.2 and 0.4. This difference occurs because only inactive agents may be removed from the swarm under confirmation reducing SDS whereas any agent may be removed from the swarm in independent reducing SDS.



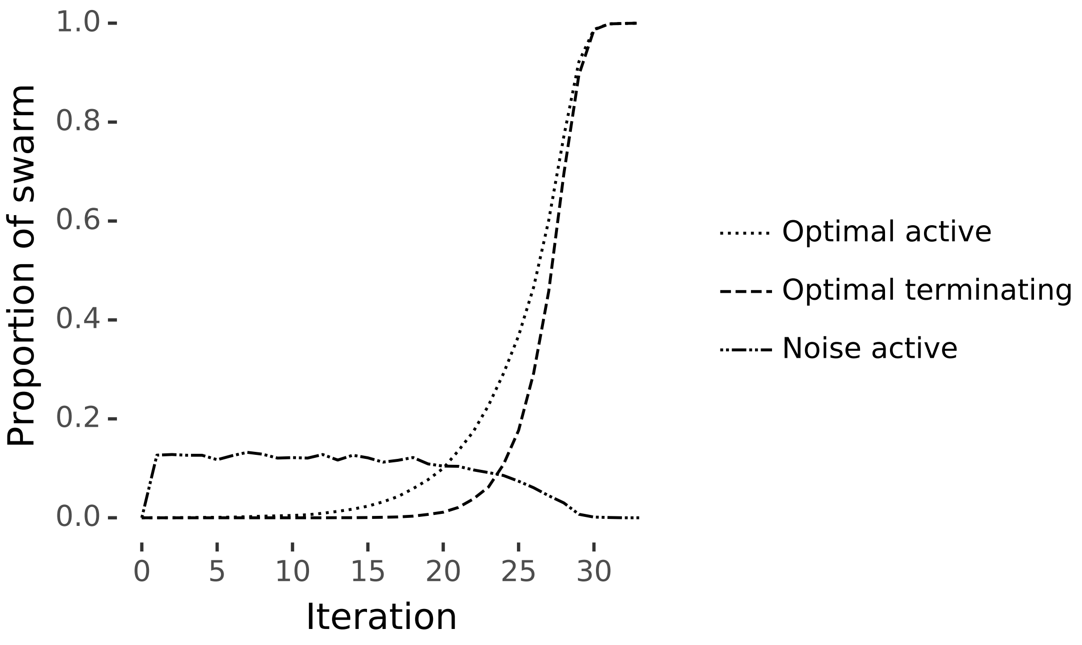


Figure 4.2: Cluster size evolution over time for confirmation reducing SDS. The x-axis shows iterations. In the upper graph the y-axis shows cluster size as a proportion of the total number of agents, in the lower graph the y-axis shows cluster size as a proportion of the agents which have not yet been removed. The graphs show an accelerating growth in the number of agents at the optimal hypothesis followed by a similar growth of terminating agents at the optimal hypothesis until the entire swarm is terminating or removed.

The second plot (which shows the states of agents as a proportion of the agents not yet removed) is much closer to the equivalent plot for independent reducing SDS. The proportion of agents at the optimal hyptothesis, the proportion of terminating agents, and the proportion of agents at noise solutions all evolve in similar ways.

From these two graphs it can be seen that both local halting criteria exhibit the behaviour of a short period of apparent inactivity followed by the total swarm size reducing with a characteristic S-curve. This shape is due to the positive-feedback effect of the reducing behaviour as a larger number of terminating agents increases the chance of further agents becoming terminating, as fewer agents remain this effect slows down. In the case of confirmation reducing SDS the number of terminating agents stabilises as there are no more non-terminating. In the case of independent reducing SDS the number of terminating agents reaches zero as all agents are removed from the swarm.

4.3.3 Discussion

The experiment performed on confirmation reducing SDS and independent reducing SDS show that both variants exhibit the same features and the same problems. Both variants are able to demonstrate the ability to diffuse the termination behaviour rapidly throughout the swarm. Both variants are also highly greedy, with a high probability of converging to the first hypothesis located in cases where $S >> A$.

Reducing SDS therefore has demonstrated the ability for a halting behaviour to be instantiated and spread by the actions of individuals, but has not demonstrated the ability to indirectly compare a number of potential hypothesis outside of the case where only the optimal hypothesis has a non-negligable score. The ability of standard SDS to indirectly compare hypothesis is enabled by the stability of one cluster being requiring that there are no other stable clusters present, in such a case the cluster with the lower score will collapse. Therefore, in the next section a further variant is introduced in which the decision of an agent to become terminating depends on a cluster's stability.

4.4 Running-mean SDS

A further variant of reducing SDS adds an extra variable to the state of each agent, representing the agent's confidence that the swarm has reached quorum at a given hypothesis. This variable is introduced because ants were observed to require a number of repeated visits to a candidate nest site before they would begin transporting, whereas an agent in independent reducing SDS or confirmation reducing SDS could begin terminating the first iteration after becoming active. The new variable is named the agent's confidence. An agent is determined to have sensed quorum and hence will commence terminating once their confidence has reached a predetermined value called the quorum threshold. In the new variant an agent's confidence will be equal to the size of the cluster at the agent's hypothesis averaged over a number of iterations. The average cluster size is calculated at each iteration, so the variant is called running-mean SDS.

The calculation of an agent's confidence is controlled by two parameters, the minimum interaction count m_{min}, and maximum memory length m_{max}. For every iteration an agent remains active, they record the size of the cluster at their hypothesis. Until the number of recorded cluster sizes equals m_{min} the agents confidence is always zero. Once the number of recorded cluster sizes equals m_{min} the agent's confidence is set to the mean average of the recorded cluster sizes. Once the number of recorded cluster sizes equals m_{max} the oldest value is replaced with the newest one.

When an agent is inactive their confidence is reset to zero and their recorded values are discarded. The purpose of requiring that at least a minimum number of values have been recorded is to reflect that ants were observed to begin transporting only after a number of visits to the same candidate nest site, and that the decision to begin transporting was influenced by the size of the population at the candidate nest site [78]. It also has the effect of reducing the amount of variance of the calculated mean which helps to ensure that an agent will only sense quorum at a hypothesis which has formed a stable cluster.

Running-mean SDS may be denoted as

$$\text{SDS}\left(I^{\text{SYNCHRONOUS}}\left(D^{\text{RUNNING MEAN}}(DH^{\text{STANDARD}}), T^{\text{REDUCING}}(T^{\text{STANDARD}})\right), H^{\text{REDUCING}}\right)$$

Algorithm 39 $D^{\text{RUNNING MEAN}}$ — Running mean diffusion

```
1: function $D^{\text{RUNNING MEAN}}(DH, m_{\max}, m_{\min}$, quorum threshold, swarm)
2:     function $D'$(agent)
3:         polled ← random agent in swarm
4:         if agent is inactive then
5:             memory ← empty queue of maximum length $m_{\max}$
6:             if polled is active then
7:                 hypothesis of agent ← hypothesis of polled
8:             else
9:                 hypothesis of agent ← $DH()$
10:        else
11:            if agent is terminating then
12:                if hypothesis of agent $\neq$ hypothesis of polled then
13:                    polled is removed from swarm
14:            else
15:                cluster size ← number of active agents with hypothesis of agent
16:                activity ← $\frac{\text{cluster size}}{\text{swarm size}}$
17:                push activity into memory
18:                if number of activities in memory $\geq m_{\min}$ then
19:                    confidence ← average activity value in memory
20:                    if confidence $\geq$ quorum threshold then
21:                        agent becomes terminating
22:    return $D'$
```

4.4.1 Greediness of running-mean SDS

To examine whether running-mean SDS was less greedy than reducing SDS a number of instances were run in conditions of very high noise. The conditions were not varied, only the level of the quorum threshold to investigate whether this would suppress convergence to a distractor hypothesis. A distractor hypothesis is a single hypothesis with score δ where $\beta < \delta < \alpha$ The conditions used were $\alpha = 0.9$, $\delta = 0.8$, $\beta = 0.1$, $A = 100$ and employing halting $H^{\text{FIXED}}(200)$. The experiment performed an instance of running-mean SDS with values $m_{\max} = 8$, $m_{\min} = 4$ and every value for quorum threshold in the interval [0.0, 1.0] with an step of 0.1. To simulate an search space of infinite size one agent was initialised at each of the optimal hypothesis and the distractor hypothesis, and DH^{INFINITE} was used. For each value of quorum threshold, the running-mean SDS was run 10 000 times and the state of the swarm after halting was recorded. The resulting state of the swarm could be any one of {collapse, convergence to distractor, divided convergence, no convergence, convergence to optimal}. The states were defined as follows.

Collapse No agents were active at the optimal hypothesis or distractor hypothesis.

Convergence to distractor All agents were active at the distractor hypothesis.

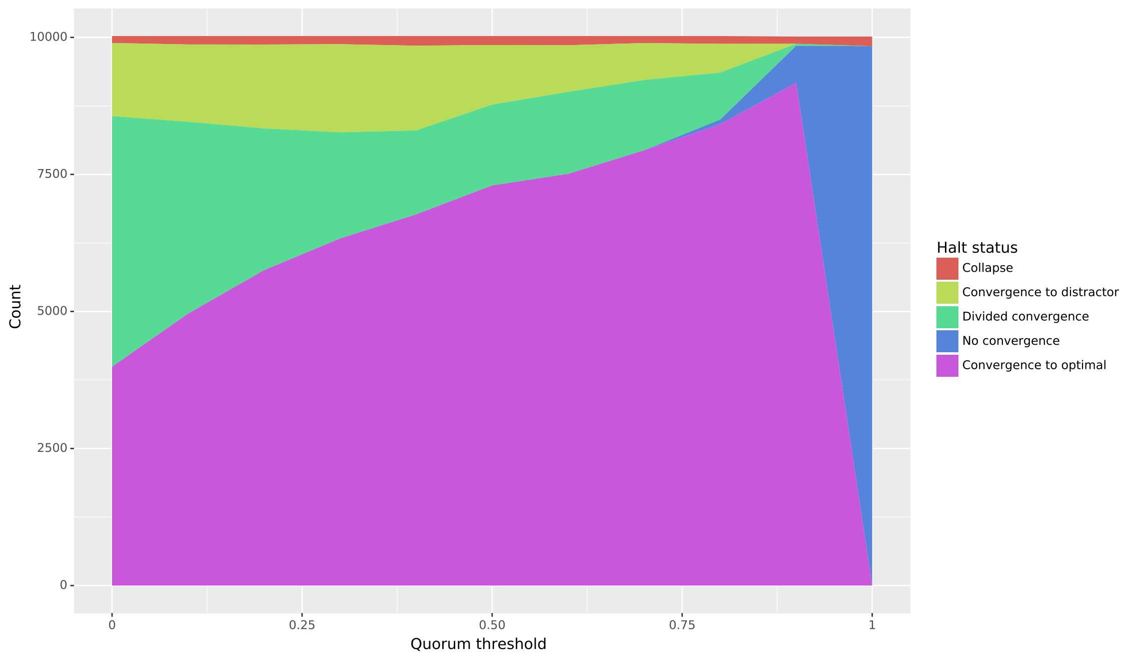


Figure 4.3: Effect of varying quorum threshold on the halt status of running-mean SDS when $\alpha = 0.9$, $\delta = 0.8$, $\beta = 0.1$

Divided convergence All agents were either active at the optimal hypothesis or active at the distractor hypothesis.

No convergence None of the above halting criteria had occurred within two thousand iterations.

Convergence to optimal All agents were active at the optimal hypothesis and each agent had a confidence greater than the quorum threshold.

4.4.1.1 Results

The results are plotted in Figure 4.3, p.149. The plot shows the effect of varying the quorum threshold for set values of optimal hypothesis score and distractor hypothesis score. There are five distinctly coloured regions, red for collapse, yellow for convergence to distractor, green for divided convergence, blue for no convergence, purple for convergence to optimal.

The probability of collapse remains constant over all values of quorum threshold, this is expected as collapse is only likely before a stable cluster has formed and until that point there is no effect from the quorum sensing behaviour or running-mean SDS. When quorum

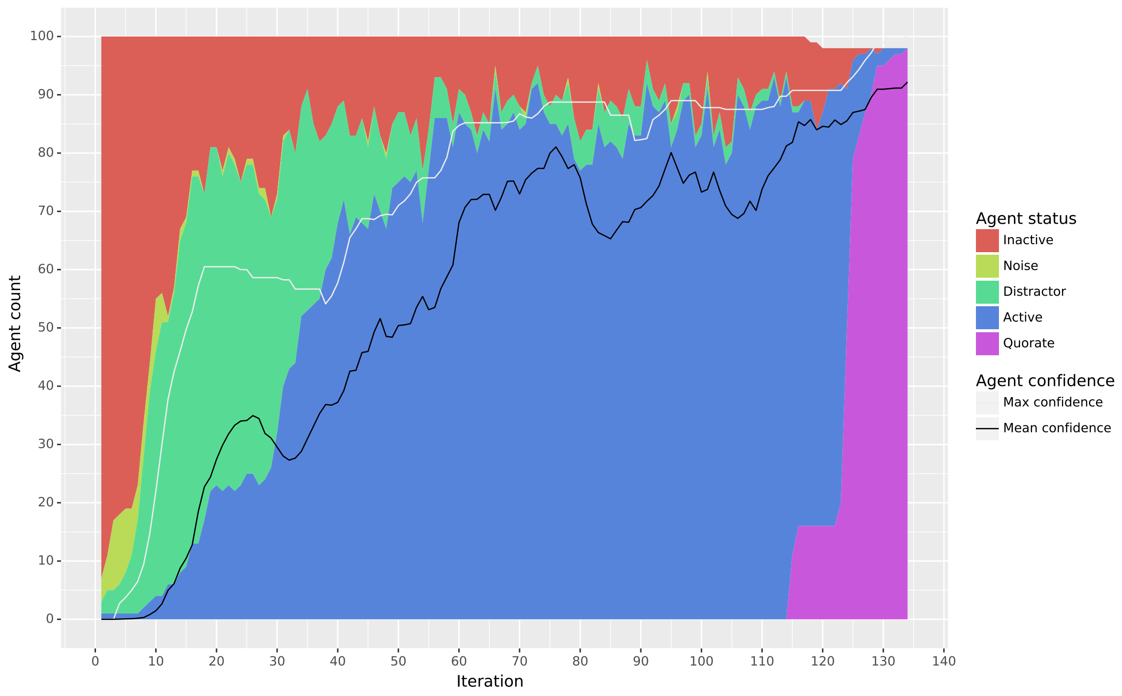


Figure 4.4: State of individual agents, mean average confidence, and maximum confidence values of the entire swarm over time in running-mean SDS with $\alpha = 0.9$, $\delta = 0.8$, $\beta = 0.1$, $qt = 0.9$

threshold is zero there is a significant probability of either convergence to the distractor hypothesis, divided convergence, or convergence to the optimal hypothesis, this is similar to the greedy behaviour of reducing SDS, as any value of confidence over zero will induce convergence. The probability of convergence to the distractor also remains fairly stable until the quorum threshold reaches a point where it is greater than the expected steady state cluster size for the distractor hypothesis, using equation 3.16 for standard SDS this is predicted to be 0.74, when $qt = 0.8$ there is still a small chance, but almost zero at $qt = 0.9$. Once $qt = 1.0$ almost all results are no convergence, or collapse. Results of no convergence represent two possible outcomes; firstly, that the algorithm would converge if a more iterations were performed; secondly, that the algorithm would never converge. Successful convergence to the optimal hypothesis, and not to the distractor hypothesis are observed more frequently in cases where the quorum threshold has a value that is unlikely to be attained by agents at the distractor hypothesis.

The state of the swarm, including the average and maximum values for the confidence of all agents can be seen in Figure 4.4, p.150. Change in agent states over time demonstrates that running-mean SDS behaves in a similar way to standard SDS until at least one agent has detected quorum. This can be understood by considering the proportion of the swarm that is in each state over time. Initially, one agent maintains the optimal hypothesis, one agent maintains the distractor hypothesis, and all other agents are inactive. In the next iteration, the number of agents at noise hypotheses is larger than the number of agents at either the distractor hypothesis or the optimal hypothesis. This is expected as the collection of all noise hypothesis has a much higher probability of selection whilst the other clusters are small. The probability of an active agent at a noise hypothesis is the product of the homogeneous background noise, in this case $\beta = 0.1$, and the number of inactive agents, which is high in the early iterations. In the following iterations, both the distractor hypothesis and the optimal hypothesis experience rapid growth due to the positive-feedback loop of diffusion, in the plotted case the distractor hypothesis initially grows faster than the optimal hypothesis. The slightly larger probability of agents remaining active at the optimal hypothesis means that over time the population migrates from the cluster at the distractor hypothesis into the cluster at the optimal hypothesis. The evolution of the swarm begins to diverge from that of standard SDS once a single agent begins terminating. This can be seen when the plot of maximum agent confidence reaches 0.9. Within a small number of iterations, the number of quorate agents increases rapidly, and the total number of agents in the swarm begins to reduce, firstly the number of inactive agents reaches zero, finally the number of active agents reaches zero and the process halts.

4.4.2 Discussion

Running-mean SDS has the advantage of an intuitive method of quorum sensing, a hypothesis at which there is a consistently high average number of agents at a single hypothesis is an intuitive method of detecting a stable cluster. The disadvantage of this method is that it requires extra memory for every agent, and extra computation to calculate the mean averages.

The effect of the quorum threshold is beneficial in cases where both the optimal hypothesis score and distractor hypothesis score are strong, but $\alpha < 1$. The plot showing when $\alpha = 0.9$ and $\delta = 0.8$ shows a large increase in the probability of a successful convergence to the optimal with increasing quorum threshold. In cases where there is no distractor hypothesis, or no homogeneous background noise the quorum threshold will have no positive effect on the proportion of successful convergences, as in such cases being greedy is the optimal behaviour.

4.5 Quorum sensing SDS

Where running-mean SDS requires calculating an exact value for the cluster size, it is not necessarily required that an exact value is calculated when the important information is simply whether the cluster is of sufficient size. A simpler method is investigated for its ability to detect the result of comparing cluster size against a threshold without ever actually calculating the cluster size.

In quorum sensing SDS, an active agent's confidence is incremented by one for every diffusion phase in which they poll an active agent which shares their hypothesis. Quorum sensing SDS does not rely on the memory length and minimum interactions variables of running-mean SDS but employs one new variable, decay. At the end of each diffusion phase the confidence of all agents is reduced by being multiplied by the decay which is a constant value in the interval $(0, 1)$. While an agent is inactive their confidence is set to zero.

Cases with a higher value for decay (and hence the confidence of each agent decreases by a smaller proportion each iteration) have a higher maximum achievable value for confidence. This is because each iteration confidence can at most increase by 1 before decaying, and the amount the confidence decreases each iteration is proportional to its current value. Therefore there is a point at which the maximum increase per iteration equals the guaranteed decrease per iteration. Consider an active agent at a perfect hypothesis, in a cluster that consisted of the entire swarm, that agent is guaranteed to increase their confidence by one every iteration, their confidence would stop increasing when the confidence decreased every iteration by the same amount. That is, when $c_{\max} - (c_{\max} * d) = 1$, where d is the decay, and $c_{\max}$ is the maximum value confidence would reach. Rearranging gives $c_{\max} = \frac{1}{1-d}$. Therefore taking a small value of decay, $d = 0.5$ yields $c_{\max} = 2$, and taking a large value $d = 0.99$ yields $c_{\max} = 100$, in the cases presented here $d = 0.9$ and so $c_{\max} = 10$. Quorum sensing diffusion ($D^{\text{QUORUM SENSING}}$) is shown in algorithm 40.

Quorum sensing SDS may be denoted as

$$\text{SDS}\left(I^{\text{SYNCHRONOUS}}\left(D^{\text{QUORUM SENSING}}\left(DH^{\text{STANDARD}}\right), T^{\text{REDUCING}}\left(T^{\text{STANDARD}}\right)\right), H^{\text{REDUCING}}\right)$$

Algorithm 40 $D^{\text{QUORUM SENSING}}$ — Quorum sensing diffusion

```
1: function $D^{\text{QUORUM SENSING}}(DH$, decay, quorum threshold, swarm)
2:     max confidence $\leftarrow \frac{1}{1-\text{decay}}$
3:     confidence threshold $\leftarrow$ max confidence $\times$ quorum threshold
4:     function $D'$(agent)
5:         polled $\leftarrow$ random agent in swarm
6:         if agent is inactive then
7:             confidence of agent $\leftarrow 0$
8:             if polled is active then
9:                 hypothesis of agent $\leftarrow$ hypothesis of polled
10:            else
11:                hypothesis of agent $\leftarrow DH()$
12:        else
13:            if agent is terminating then
14:                if hypothesis of agent $\neq$ hypothesis of polled then
15:                    polled is removed from swarm
16:            else
17:                if polled is active *and* hypothesis of agent $=$ hypothesis of polled then
18:                    confidence of agent $\leftarrow$ confidence of agent $+ 1$
19:                confidence of agent $\leftarrow$ confidence of agent $\times$ decay
20:                if confidence of agent $\geq$ confidence threshold then
21:                    agent becomes terminating
22:    return $D'$
```

4.5.1 Experiment on the robustness of quorum sensing SDS

To observe the effect of varying the quorum threshold, an instance of quorum sensing SDS was initialised, with one agent active at the optimal hypothesis, one agent active at the distractor hypothesis, and all other agents inactive. The algorithm was then iterated until halting, for a maximum of 1,000 iterations. This experiment was repeated 100 for each valid combination of the four variables (i) quorum threshold (qt), (ii) optimal hypothesis score, (iii) distractor hypothesis score (δ), and (iv) uniform background noise, where hypothesis scores and quorum threshold took the values in the range [0,1] in steps of 0.1. A valid combination is one where $0 \leq \beta < \delta < \alpha \leq 1$. Two parameters were not varied; the swarm was always initialised with exactly one hundred agents, and the decay was always 0.9, meaning that at the end of every iteration, the confidence c of each active agent would be reduced to $0.9c$. After each instance had halted, the resulting state of the swarm was recorded, using the same categorisation as in the experiment with running-mean SDS, {collapse, convergence to distractor, divided convergence, no convergence, convergence to optimal}.

4.5.1.1 Results

Figure 4.5, p.155 shows the results for all experiments where uniform background noise was 0.1. Each plot shows the effect of varying the quorum threshold for set values of optimal hypothesis score and distractor hypothesis score. There are six distinctly coloured regions, red for collapse, yellow for convergence to distractor, green for divided convergence, blue for no convergence, purple for convergence to optimal, and grey for areas that represent when $\delta < \alpha$ does not hold.

Collapse For values where $\alpha \leq 0.5$ the most common result was collapse. Other results are observed once α or δ become large enough to form stable clusters, and collapse becomes increasingly rare as the hypothesis scores become stronger. The probability of collapse is zero in cases where the optimal hypothesis score equals one. As with independent reducing SDS and confirmation reducing SDS there is a small possibility of convergence in cases where $\alpha < \alpha_{\text{min}}$ and when quorum threshold is low. When $qt = 0$ quorum sensing SDS behaves exactly as confirmation reducing SDS and is therefore very greedy.

Convergence to distractor For a swarm to converge entirely to the distractor, with no agents at the optimal hypothesis, the cluster at the optimal hypothesis must have collapsed entirely, and an agent at the distractor hypothesis must have become terminating. This result is therefore seen most clearly in the cases where the optimal hypothesis score is weak enough to have a significant probability of collapse but the distractor hypothesis is strong enough to maintain a stable cluster.

Divided convergence Divided convergence requires that agents become terminating at both the optimal hypothesis and distractor hypothesis. Two distinct patterns can be seen in the instances of divided convergence. Firstly when $\alpha = 1$, the quorum threshold appears to have little effect, with the ratio of convergence to optimal results to divided convergence results being determined only by the distractor hypothesis score. Secondly, in cases where $\alpha < 1$, the proportion of divided convergence results can be seen to be influenced by the quorum threshold; as the value of quorum threshold increases the proportion of divided convergence results decreases to zero.

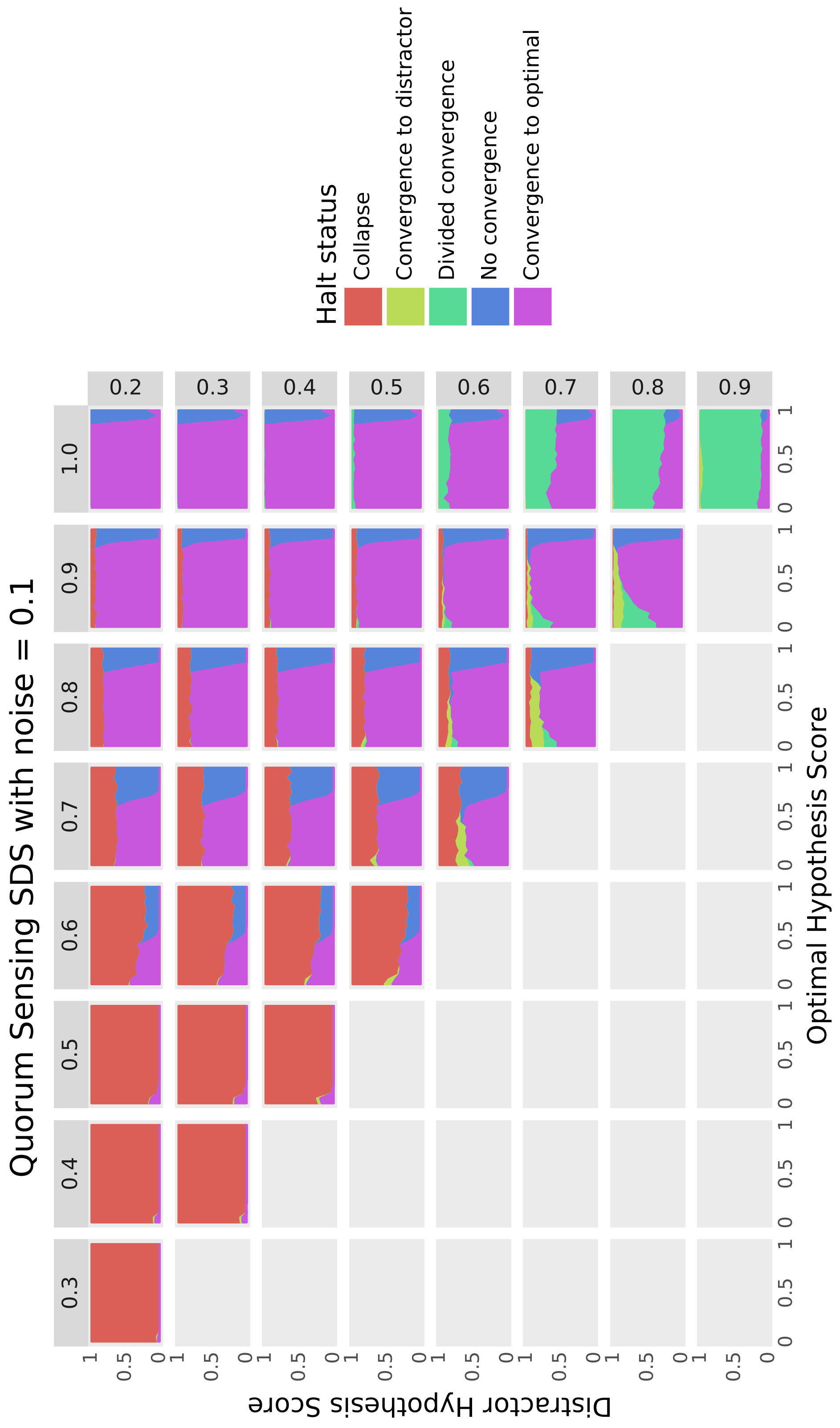


Figure 4.5: Effect of varying quorum threshold on the halt status of quorum sensing SDS when $\beta = 0.1$. For each plot the x-axis measures quorum threshold and the y-axis measures the proportion of each result.

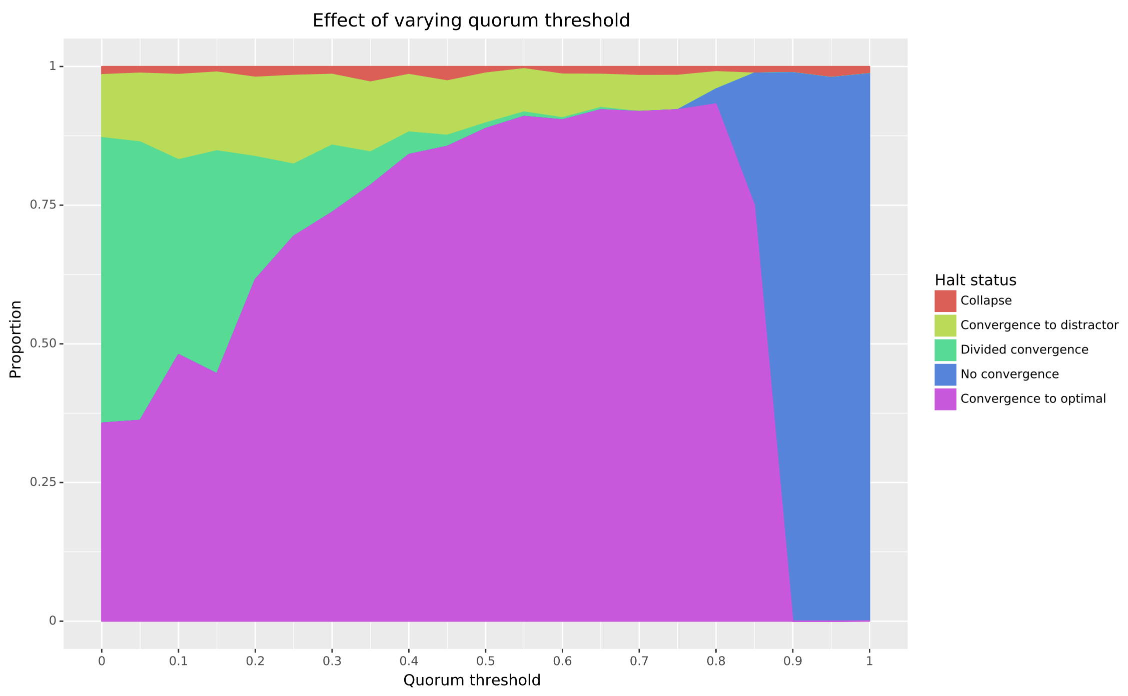


Figure 4.6: Effect of varying quorum threshold on the halt status of quorum sensing SDS when $\alpha = 0.9$, $\delta = 0.8$, $\beta = 0.1$

No convergence Results of no convergence represent two possible outcomes; firstly, that the algorithm would converge if a certain number of iterations were performed beyond the two thousand used in the experiment; secondly, that the algorithm would never converge. This result only appears in cases where the quorum threshold is greater than zero, the absolute value being larger as the optimal hypothesis score increases.

Convergence to optimal Successful convergence to the optimal hypothesis, and not to the distractor hypothesis are observed in cases where the optimal hypothesis is strong enough to maintain a stable cluster, and the quorum threshold has a value that can be attained by agents at the optimal hypothesis and cannot be attained by agents at the distractor hypothesis. The effect of the quorum threshold is most beneficial in cases where both the optimal hypothesis score and distractor hypothesis score are strong, but $\alpha < 1$. Therefore the plot showing when optimal hypothesis score is 0.9 and the distractor hypothesis score is 0.8 shows the greatest increase in convergence to optimal results with increasing quorum threshold. As with running-mean SDS, in cases where there is no distractor hypothesis, or no homogeneous background noise the quorum threshold will have no positive effect on the proportion of successful convergences, as in such cases being greedy is the optimal behaviour.

4.5.2 Experiment on the effect of decay

Introducing the notion of an agent's confidence value, which decays over time, raises the question of the rate at which the value decays. In quorum sensing SDS, the value of decay is a real number in the range $[0, 1]$. As this is a free variable it is essential to investigate how the behaviour of quorum sensing SDS will change as different values are chosen. To this end the following experiment is proposed.

As with other experiments, a search space was be defined with only three hypotheses, one representing the optimal hypothesis, one representing the distractor hypothesis, and the other representing homogeneous background noise. An infinite search space size was simulated with DH^{INFINITE}, so one agent was initialised as active at the optimal hypothesis, and at the distractor hypothesis. The confidence values of all agents at any noise hypothesis was reset to zero at the beginning of each test phase. The search space used always had the scores $\alpha = 0.9, \delta = 0.8, \beta = 0.1$.

This experiment recorded the number of iterations before a successful convergence, and the probability of a successful convergence, across a range of values for quorum threshold when the value for decay is low (0.5), medium (0.8, and 0.9), and high (0.99).

4.5.2.1 Results

For each value of decay, the average number of iterations before convergence can be seen in Table 4.5, p.160, these values are also plotted in Figure 4.8, p.159.

For any value of decay, the number of iterations before a successful convergence increase as the value for quorum threshold increases, this is expected as a higher quorum threshold requires agents to reach a higher confidence value. There are cases where no successful convergences are detected, even though $qt < 1$ this represents cases where no agents reach the theoretical maximum because they do not remain active long enough to reach it. Given infinite time an agent would eventually remain active long enough and increase its confidence sufficiently to become terminating but this becomes increasingly unlikely as c_t increases. Therefore the practical value of quorum threshold must be lower than 1 for all cases where agents are not expected to remain permanently active, i.e. when $\alpha < 1$.

For each value of decay, the probability of a successful convergence for a range of qt can be

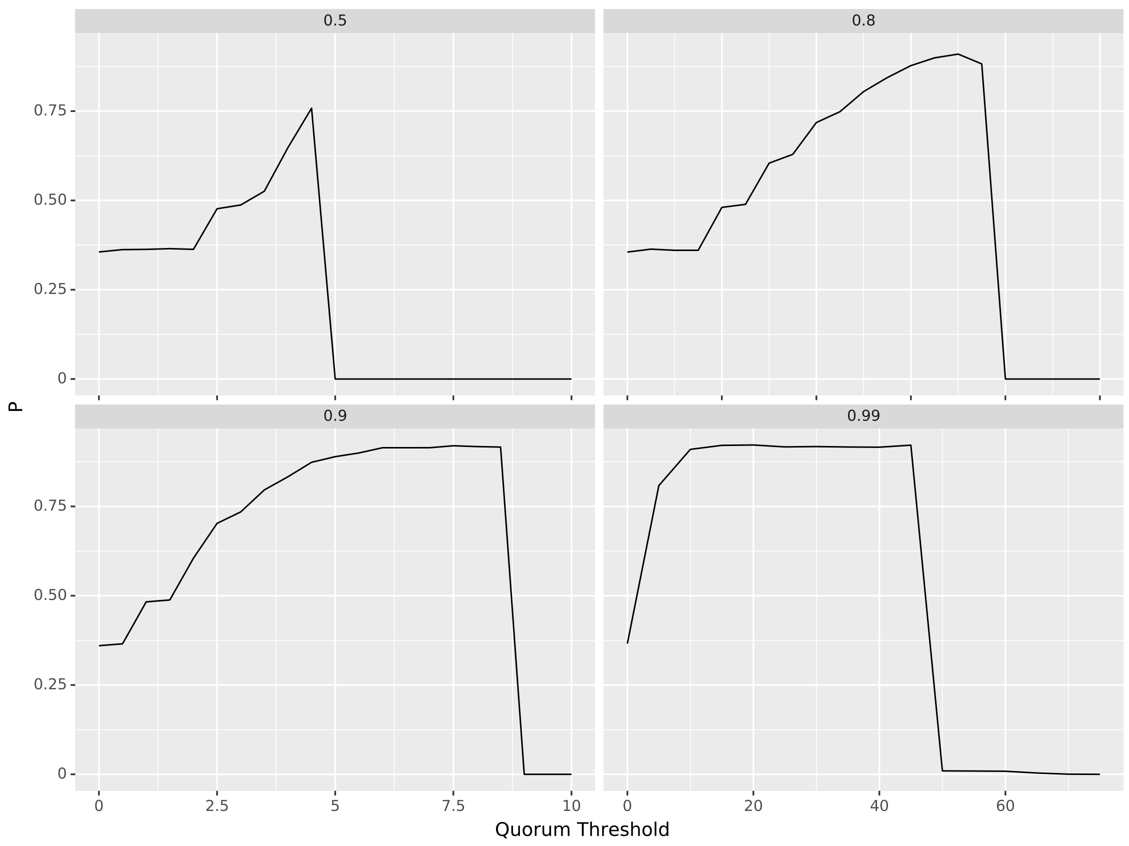


Figure 4.7: Effect of varying quorum threshold on the probability of successful convergence when decay = 0.5, 0.8, 0.9, 0.99.

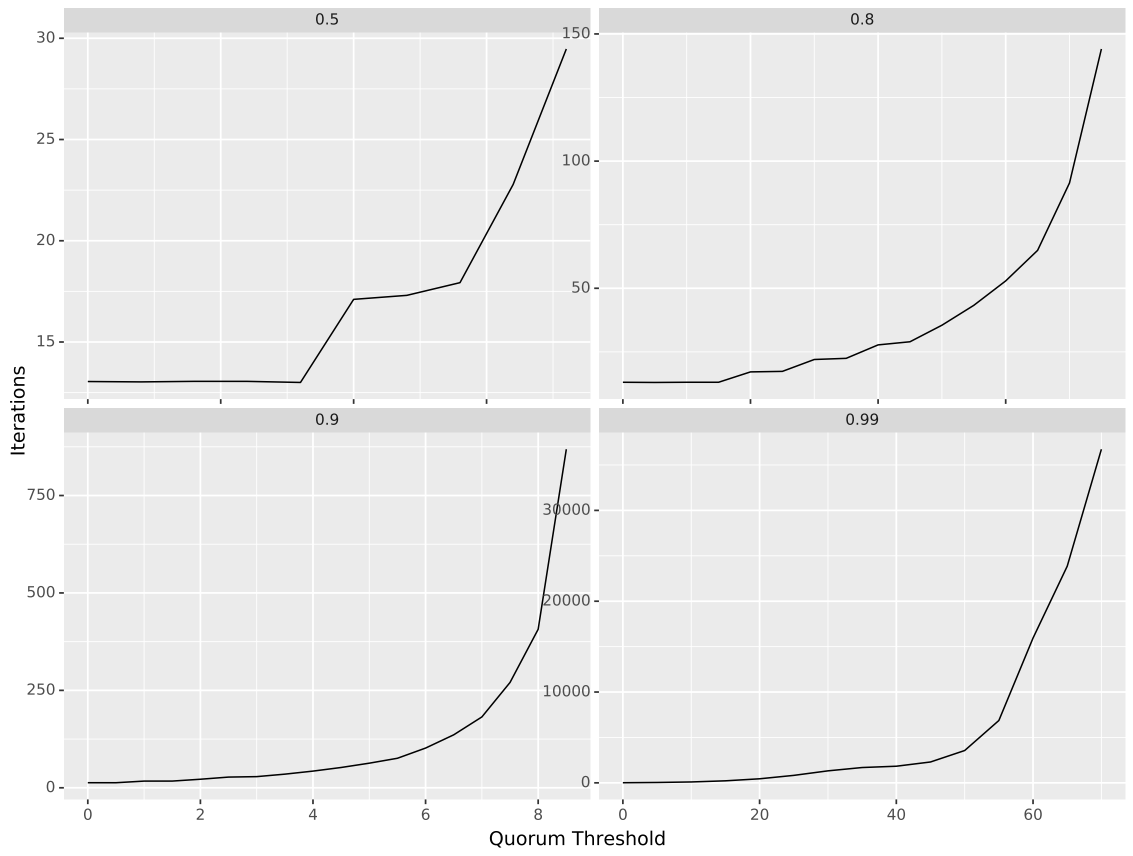


Figure 4.8: Effect of varying quorum threshold on the number of iterations before successful convergence when decay = 0.5, 0.8, 0.9, 0.99.

Decay	QT	c_t	s	n	P	μ	σ	C_V
0.50	0.20	0.40	3 667	10 100	0.363	13.002	1.782	0.137
0.50	0.25	0.50	4 815	10 100	0.477	17.106	2.235	0.131
0.50	0.30	0.60	4 921	10 100	0.487	17.303	2.307	0.133
0.50	0.35	0.70	5 310	10 100	0.526	17.931	2.465	0.137
0.50	0.40	0.80	6 544	10 100	0.648	22.780	3.305	0.145
0.50	0.45	0.90	7 658	10 100	0.758	29.471	4.735	0.161
0.50	0.50	1.00	0	10 100	0.000	—	—	—
0.80	0.50	2.50	8 128	10 100	0.805	35.469	5.818	0.164
0.80	0.55	2.75	8 523	10 100	0.844	43.292	7.405	0.171
0.80	0.60	3.00	8 862	10 100	0.877	52.896	9.524	0.180
0.80	0.65	3.25	9 081	10 100	0.899	64.904	11.895	0.183
0.80	0.70	3.50	9 188	10 100	0.910	91.400	17.056	0.187
0.80	0.75	3.75	8 910	10 100	0.882	144.086	22.699	0.158
0.80	0.80	4.00	0	10 100	0.000	—	—	—
0.90	0.60	6.00	9 234	10 100	0.914	101.918	19.425	0.191
0.90	0.65	6.50	9 234	10 100	0.914	135.928	26.123	0.192
0.90	0.70	7.00	9 235	10 100	0.914	181.802	35.838	0.197
0.90	0.75	7.50	9 290	10 100	0.920	270.267	54.128	0.200
0.90	0.80	8.00	9 267	10 100	0.918	406.803	81.237	0.200
0.90	0.85	8.50	9 253	10 100	0.916	868.723	168.400	0.194
0.90	0.90	9.00	0	10 100	0.000	—	—	—
0.99	0.45	45.00	9 307	10 100	0.921	2 296.990	365.583	0.159
0.99	0.50	50.00	97	10 100	0.010	3 558.577	849.051	0.239
0.99	0.55	55.00	93	10 100	0.009	6 865.022	2 584.689	0.377
0.99	0.60	60.00	87	10 100	0.009	15 944.920	9 284.443	0.582
0.99	0.65	65.00	37	10 100	0.004	23 854.649	11 317.487	0.474
0.99	0.70	70.00	4	10 100	0.000	36 732.750	6 124.616	0.167
0.99	0.75	75.00	0	10 100	0.000	—	—	—

Table 4.5: The number of iterations of quorum sensing SDS before a successful halt for various values of decay. Columns are, confidence threshold at which an agent becomes terminating (c_t), number of successful convergences (s), number of repeats (n), probability of successful convergence (P), mean average number of iterations (μ), standard deviation of number of iterations (σ), coefficient of variation of iterations (C_V)

seen in Table 4.5, p.160, these values are also plotted in Figure 4.7, p.158.

4.5.3 Discussion

We have examined the behaviour of quorum sensing SDS over a range of values for parameters which describe all search spaces which will be encountered in practical applications, either as computational search problems or by ants or bees in nature.

The mechanism which defines the activity and hypotheses of agents in quorum sensing SDS is identical to that of standard SDS until a single agent has detected quorum, that is their confidence value has reached a predetermined quorum threshold, as so the behaviour of quorum sensing SDS up until that point can be adequately modelled by the existent models of SDS.

The memory requirements for quorum sensing SDS are higher than standard SDS as each agent needs to record their confidence value, though one value per agent is unlikely to be a limiting factor for memory in an algorithm. Similarly the computational complexity of quorum sensing SDS is greater than standard SDS as each agent may need to increment, decay, and compare their confidence values once per iteration. Both of the increased memory requirements and increased computational complexity are small and will increase linearly with the number of agents.

4.5.3.1 Effect of quorum threshold

When $qt = 0$, both running-mean SDS and quorum sensing SDS behave like confirmation reducing SDS. As can be seen figure 4.5 there is always at least a small chance of convergence when $qt = 0$, even in cases where stable clusters would not form. This is because an active agent requires only a single confirmatory contact to commence terminating. Even a single agent at a hypothesis with score s will become active and subsequently poll itself in the diffusion phase with a probability $\frac{s}{A}$, after which it will remain terminating. This is also possible when $qt > 0$ but with a diminishing probability at quorum threshold increases. Conversely, an instance of quorum sensing SDS or running-mean SDS where $qt = 1$ will behave identically to an instance of standard SDS where $H = H^{\text{INDEFINITE}}$.

As an active agent's confidence value depends on the number of other agents with the same hypothesis, an agent's confidence value will have the greatest probability of reaching its maximum value once the cluster at the hypothesis has reached its largest size, and hence stabilised. This behaviour therefore is likely to converge on a hypothesis which has not only got the largest cluster, but has also stabilised at a sufficiently large size. Once this has occurred the effect of quorum threshold is act as the threshold determining whether the cluster is of sufficient size to represent an acceptable solution.

4.5.3.2 Effect of decay

The immediate effect of decay is to reduce larger confidence values by more than smaller confidence values. The impact on quorum sensing SDS is that with higher values of decay (and hence smaller reductions in confidence each iteration) there is a larger maximum value which an agent's confidence may reach. As an agent's confidence can only increase by a maximum of 1 each iteration, before decaying, larger values take more iterations to be reached. A larger decay therefore enables the selection of values for quorum threshold which can only be reached after many repeated tests. Many repeated tests will reduce the effect of stochastic noise, in accord with the law of large numbers. This reduces the chance of rare results due to the stochastic nature of SDS but increases the time taken for quorum sensing. This stabilising time also helps ensure the hypothesis is of genuinely high quality, reducing the probability of a stochastic aberrations, and also allowing some time for other agents to evaluate more of the search space and potentially locate a superior hypotheses.

One case not explored was the possibility of having no decay at all, equivalent to setting decay $= 1$, in this case an agents confidence value would increase monotonically so long as they remained active. This would remove the theoretical upper bound for confidence values, but practically the maximum confidence value will be capped by the probability of an agent eventually becoming inactive, a probability that increases over time for an active agent in all cases except when an agent has located a perfect hypothesis. This variant would have the advantage that it would always halt regardless of the value chosen for the quorum threshold in cases that a perfect hypothesis is present in the search space, whereas any variant which uses decay implies a maximum theoretical value on an agents confidence value; in cases of no decay even imperfect hypotheses have a probability, however small, of an agent's confidence reaching any value. A disadvantage of this variant is that it requires the storage of arbitrarily large integers, at least as large as the quorum threshold, to record the confidence values of agents, this is slightly less biologically plausible than the variant which uses decay, as discrete digital counters are seen less in nature than continuous values which decrease quicker as they increase in size.

4.5.3.3 Comparison of running-mean SDS with quorum sensing SDS

The ability of a swarm of ants to be able to effectively calculate a mean average evaluation for a location has been previously observed [28] and the evidence that quorum sensing SDS behaves in a similar way to running-mean SDS indicates that a similar mechanism may be in use in nature. If quorum sensing SDS is therefore interpreted as effectively identifying the mean cluster size at a hypothesis, it follows that the effect of quorum threshold is to enforce a threshold for the mean cluster size of the hypothesis below which the algorithm will not converge.

Similarly they can both be modulated for the minimum acceptable hypothesis quality with the quorum threshold, and they can both be modulated for the minimum amount of validation required, using decay in the case of quorum sensing SDS and minimum interactions in the case of running-mean SDS. This indicates that simple biological agents, and simple computational agents are able to effectively decide whether the population at a given candidate nest site is over a certain proportion of the entire colony with a simple mechanism.

Both running-mean SDS and quorum sensing SDS require agents to remain active at a hypothesis for a number of iterations, the precise number of iterations has a lower bound set by a parameter. In running-mean SDS the parameter is minimum interactions, in quorum sensing SDS the parameter is a combination of decay and quorum threshold. Once an agent is active in both quorum sensing SDS and running-mean SDS the agent will terminate if the value of $\frac{\text{cluster size}}{\text{swarm size}}$ is sufficiently large, in the case of running-mean SDS this is the calculation of the average swarm size, in the case of quorum sensing SDS this is the probability that a randomly polled agent will maintain the same hypothesis.

The distinction between running-mean SDS and quorum sensing SDS is that where running-mean SDS calculates an exact value and tests it against a threshold, quorum sensing SDS manages to test an equivalent value against the same threshold but without explicitly calculating it. This observation is reminiscent of an observation made in section 3.5.6 on the ability of bacteria to detect an occurrence, due to their context they need not be able to detect the exact situation they are in, simply that they are in a situation in which to perform a suitable behaviour. In this case the action of quorum sensing SDS is not to calculate a cluster size, merely to detect whether the cluster is of sufficient size.

The similar behaviours can be seen as in the experiment for the same parameters with quorum sensing SDS (Figure 4.6, p.156). The similarity in halting behaviour of running-mean SDS and quorum sensing SDS shows that the simple mechanism employed by quorum sensing SDS provides the swarm with similar powers of discrimination as the more complex mechanism employed by running-mean SDS.

The quorum threshold of ants during nest site evaluation was determined to be constant proportional to colony size and modulated depending on the conditions (Section 3.5.4.1). In running-mean SDS and quorum sensing SDS the quorum threshold is constant for each instance, but can be selected depending on conditions of the task. Quorum threshold can be chosen to bias towards a lower bound on the quality of hypothesis which may induce terminating behaviour, and decay can be chosen to affect the amount of confirmation an ant will require before becoming terminating. A high quorum threshold forces the rejection of lower quality hypothesis but at the risk of never halting, whereas a high decay forces an increase in the amount of validation a hypothesis received, but at the expense of a longer execution time.

4.6 Differences between quorum sensing SDS and ant nest-site selection

Although the design of quorum sensing SDS was inspired by the nest-site selection of ants, there are certain important differences. Importantly, there is no discrete halt state for a swarm of ants, once a new nest site has been located and the migration completed, the swarm continues performing the multitude of other tasks required for a colony of ants to survive. This has two implications for the suitability of quorum sensing SDS as a model for ant behaviour, firstly the behaviour of ants removed from the swarm is ignored, secondly, the probability of future migrations is ignored. An agent removed from the swarm in quorum sensing SDS performs no further actions, whereas an ant transported to a new nest will immediately perform other actions.

Reverse tandem running is not directly analogous to any activity in quorum sensing SDS. One possible reason for this as that agents which have begun terminating are still treated as active agents. If terminating agents were considered inactive in $D^{\text{QUORUM SENSING}}$ then their hypotheses would not be assumed by inactive agents. In such a case a number of terminating agents may begin removing other agents from the swarm without increasing the cluster size at their hypothesis. In this case a mechanism analogous to reverse tandem running may be useful to return some of the terminating agents to their active, recruiting, state. A potential implementation of mechanism is to have all agents who have detected quorum randomly polling another agent, and becoming active, but not terminating, if the polled agent is also terminating and maintaining the same hypothesis. This mechanism may be useful in two situations: firstly, if a number of agents begin terminating at a poor quality hypothesis, this would provide a means by which such agents could become merely active and later, inactive; secondly, if all of the agents in a small cluster begin terminating at a high quality hypothesis, this would provide a means by which they could become active and then attract more agents to the cluster. Both of these behaviours would decrease the number of situations in which this variant of quorum sensing SDS would converge to a suboptimal solution, either by speeding up the speed of convergence to a high quality hypothesis, or halting the process of convergence to a low quality hypothesis. This may be analogous to behaviour in ants as at any one time each ant may only be performing at most one of the two behaviours of recruitment via tandem running, or migration by transporting.

This process may even show how the accuracy of the process does not require full cooperation from all individuals, and is even robust in the face of belligerent agents. As the formation of stable clusters in quorum sensing SDS is identical to that of standard SDS before ant agents begin terminating, the mathematical models of over variants can be employed. The effect of the secret optimist (Section 3.4.2.4) for example will increase the effective score of all hypotheses. This suggests that the effect of an ant performing the same activity, ignoring the results of its evaluation, even a multitude of these ants do not lead to a collapse of the system, simply an increased chance of converging to a low quality solutions, in cases where no solutions of sufficient quality exist this may even be a benefit, forcing the swarm to choose "any port in a storm". Similarly, the hermit (Section 3.4.2.3) causes agents to refuse to share their hypothesis, and hence the effective score of all hypotheses decreases. An agent that does not share its hypothesis is analogous to an ant which is not participating in the recruitment. This has the effect of increasing the chance of failure to converge to a solution. As with the secret optimist this may be a benefit, in relatively benign cases where a bad decision is worse than no decision.

4.7 Conclusion

Four methods for implementing local termination in SDS have been examined. independent reducing SDS closely modelled the behaviour of ants, confirmation reducing SDS remained close to the definition of standard SDS. Both demonstrated the ability to diffuse a termination behaviour through a swarm without using an external halting method, they were also very greedy due to not requiring any validation of the quorum sensing mechanism. running-mean SDS and quorum sensing SDS used alternative methods to validate the quorum sensing mechanism, before diffusing the terminating behaviour through the swarm. running-mean SDS used an exact method, and quorum sensing SDS used a stochastic process to model the observation that ants performing nest site selection determined their behaviour as a result of the population level at a candidate nest site. The requirement for validation of the quorum sensing mechanism means that both running-mean SDS and quorum sensing SDS are not so greedy as independent reducing SDS and confirmation reducing SDS. In both cases an agent begins terminating once their confidence has reached a given quorum threshold. The discrimination abilities of running-mean SDS and quorum sensing SDS have been shown to be very similar.

Furthermore the requirement that clusters are stable for a number of iterations ensures an important feature of standard SDS is expressed. As seen in section 3.2.4 agents will have orders of magnitude greater probability of being in one cluster than another, even with a fairly small difference between the score of the hypotheses, this means that should ants implement such a system then they will effectively compare nest sites indirectly, by maintaining only one stable population at the superior nest site.

Quorum sensing SDS therefore demonstrates the ability of ants to to select only candidate nest sites of sufficient quality, but also to select the best nest site out of all candidates encountered, and furthermore to balance between a quick decision in harsh conditions and an accurate decision in benign conditions. All without executive control over the entire swarm, and without direct comparison of two sites.

Chapter 5

Conclusion and Future Work

This thesis introduced AI as a field with historical roots and a modern mature state in which progress can largely by identified as sharing an underlying approach of either making a mind or modelling the brain. Swarm intelligence was presented as an alternative approach which is related to both previous approaches, but perfectly described by neither. Certain behaviours of some natural swarms were presented which were used as the basis for further investigation. SDS was described in detail as an example of a swarm intelligence algorithm which had both practical application and behaviour analogous to that observed in natural swarms. All major variants of SDS were categorised and individually examined for their population level behaviour and the resulting advantages and disadvantages. A new variant of SDS, reducing SDS, was defined which aimed to demonstrate whether a individual level activity inspired by an activity observed in natural swarm would effect quorum sensing, a population level activity observed in ants. The variant demonstrated that quorum sensing was achievable with individual action, but the practicality of the variant was limited due to greedy convergence. A further variant of SDS, quorum sensing SDS, was developed which utilised the characteristic behaviour of reducing SDS and also introduced a variable measure of confidence for each agent, hence reducing the probability of early convergence. Two methods of measuring confidence were compared, one performed an exact calculation and the other utilised stochastic heuristic, both performed similarly which indicated that an exact method was not required for quorum sensing.

5.1 Conclusion

Section 2.1 presented the history of AI as a series of methods for producing increasingly adaptable behaviours from artefacts. Initially each behaviour had to be defined explicitly, such as in Hero's Mechanical Theatre. Later, individual behaviours could be selected from a fixed range of behaviours by reconfiguring the artefact, as with L'Ecrivain, or determined by means of an input, as with The Jacquard Loom. Eventually, behaviours could be defined as procedures which could logically branch depending on some internally stored data, as would have been the case with the Analytical Engine. This series of methods reaches a logical conclusion with the definition of the Universal Turing Machine in which data and procedure existed within the same medium, both being represented as symbols on a tape. Put another way, the set of behaviours that may be performed by an artefact built upon the principles of the Turing machine is the set of behaviours which can be described by any symbolic process. This thesis therefore concludes that AI comes of age with the definition of the Turing machine as it provides the foundation for building systems that can implement other systems without external reconfiguration.

Section 2.2 presented the two categories into which modern AI can be broadly categorised, those which utilised symbolic logic as Making a Mind and those which utilised Connectionist architectures as Modelling the Brain. Where projects which could be described as attempting to make a mind initially made remarkable progress, and some genuinely useful systems, they struggled to reach the goal of a system which performed adequately outside of contexts which were restricted in scope, or well defined. Objections also emerged from Philosophy and Mathematics which cast doubt on whether this was logically possible, let alone practically achievable. Projects which could be described as attempting to model the brain were more resistant to such arguments, as they were attempting to reproduce the phenomena of a system which was known to produce intelligent behaviour. However other objections emerged in which doubt was cast on whether the system could be said to genuinely have learned a concept, and that even a successfully trained system will provide no insight into the phenomena to researchers. This thesis therefore concludes that it is a valuable endeavour to investigate approaches to AI that are less vulnerable to these critiques.

Section 2.3 presented swarm intelligence algorithms as able to perform practical tasks as well as to provide insight into the mechanisms underlying natural phenomena. Swarms are able perform intelligent behaviours at population level due to a combination of the actions of individuals and their interactions with each other. Swarm intelligence combines the explicitly defined procedures characteristic of symbolic logic, and the interactions within a population of Connectionism. This thesis therefore concludes that swarm intelligence is an approach to AI with the potential to grant insight into natural phenomena as well as develop various practical algorithms. Furthermore, by combining features of both making a mind and modelling the brain, swarm intelligence is less susceptible to the critiques of either.

Section 3.3 introduced a formalism which defined standard SDS as a set of interacting functions. This enabled variants of SDS to be described in terms of the behaviour which is distinct from standard SDS, in doing so description of variants of SDS are more concise and the aspect of variation is more clearly identified than if the entire variant needed to be described. This demonstrated the modularity of SDS and the implementation of the variants for use in experiments. This thesis therefore concludes that the formalism facilitates the expression of variants of SDS in literature and in algorithmic implementation.

Section 3.4 described a number of variants of SDS. For each variant the individual level rules that define the variant are described, as is the resulting population level behaviour. The advantages and disadvantages of the behaviour of the variant are described including the situation in which such behaviour is useful. This thesis therefore concludes that no variant should be considered objectively superior to any other. Each variant represents a trade-off of one beneficial behaviour for another, and so certain variants are simply more suited to certain situations than others.

Section 3.5 presented the observations from a number of investigations into the group decision making behaviours of natural swarms. Many of the individual level behaviours and the population level behaviours had analogues in SDS. One aspect present in natural swarms but not in SDS was the apparent ability to sense when a certain proportion of the swarm had reached consensus. As noted in section 3.5.3 the modes of halting all relied on a function which analysed the entire state of the swarm and did not emerge from the action of the individuals. This thesis therefore concludes that SDS is a suitable model of the decision making behaviour of the natural swarms described, and provided a means to explore potential mechanisms for quorum sensing.

Section 4.2 introduced independent reducing SDS, a variant which modelled the observed behaviour of ant nest site selection as closely as possible within the formalism of SDS. This method has two significant advantages, it demonstrated the ability to diffuse a terminating behaviour through the swarm by individual level behaviour alone, and it exhibited the characteristic dynamics of ant nest site selection. An experiment demonstrated the ability of independent reducing SDS to terminate in a wide range of possible search spaces, and with performance that was strongly correlated with standard SDS. The method had two significant disadvantages, the mode of diffusion was unlike any which had been previously modelled and the sensitivity to noise was too high. Section 4.3 introduced confirmation reducing SDS, a variant of SDS which used individual level behaviours that were similar to those which had been mathematically modelled and still exhibited the terminating behaviour of independent reducing SDS. An experiment demonstrated the ability of confirmation reducing SDS to terminate in a wide range of possible search spaces, and with performance that was strongly correlated with independent reducing SDS. This thesis therefore concludes that independent reducing SDS and confirmation reducing SDS are suitable bases for modelling the distributed decision making behaviour of ants, both being inspired by the observed behaviour, stable enough to be valuable as a practical algorithm, and understood as a variant of SDS. The quorum sensing ability of both variants is greedy, while it is true that a hypothesis with a higher score is more likely to have terminating agents than a hypothesis with a lower score, the termination is strongly biased towards whichever hypothesis is located first.

Section 4.4 introduced running-mean SDS which implemented a new method for quorum sensing. For each iteration they remained active agents would calculate and record the cluster size at their hypothesis. Once an agent had recorded a minimum number of values and the mean average was over a threshold value the agent would begin terminating. The quorum sensing behaviour was controlled by two values. The minimum number of values recorded could be selected to determine the number of iterations at which a cluster size must persistently be sufficiently high. The mean population size threshold could be selected to determine the minimum hypothesis score at which an agent would begin terminating. Running-mean SDS requires that agents access the global state of the swarm and perform a relatively complex calculation. Nevertheless it is able to discriminate between hypotheses of similar scores, but at the expense of being a poor model of natural behaviour and increased computational complexity. This thesis therefore concludes that calculating the average mean population size at a cluster is an robust method of evaluating the score of a hypothesis, but one which is computationally inefficient, and a poor analogue of natural behaviour.

Section 4.5 introduced quorum sensing SDS which implemented a simple individual contact-based method for quorum sensing. For each iteration they remained active agents would poll a random agent in the swarm and if the polled agent shared their hypothesis the polling agent would increment their confidence, in either case the confidence of the agent then decayed by an amount proportional to its value. This was shown to have similar powers of discrimination as running-mean SDS, and to be modulated in the same ways. Increasing the quorum threshold increased the minimum hypothesis score at which an agent would begin terminating, the same effect as increasing the running mean threshold. Increasing the decay increased the number of iterations at which a cluster size must persistently by sufficiently high, the same effect as increasing the minimum number of values recorded. This thesis therefore concludes that the quorum sensing behaviour of quorum sensing SDS has the same capacities as the quorum sensing behaviour of running-mean SDS, only implemented as a stochastic process and depending on local information.

Section 4.6 examines the dynamics of ant nest site selection as an analogy of quorum sensing SDS. As far as the analogies hold, the performance of quorum sensing SDS shows that the method employed by ants for migration allow for individual detection of quorum sensing, a rapid diffusion of such sensing through the swarm, robustness to noise, flexibility in the face of many strong options or many weak options, flexibility in the face of the requirement for a quick or a validated solution, and graceful degradation in the presence of belligerent agents. This thesis therefore concludes that by performing a process analogous to quorum sensing SDS that natural swarms are able to indirectly compare the quality of a nest site with that of another nest site, and with that of a threshold value.

This thesis has produced corroborating evidence for two hypotheses as to the nest site selection behaviour of ants. It was observed that a swarm of ants had the capacity to estimate the average quality of a fluctuating resource [28]. Using quorum sensing SDS as a model for ant nest site selection predicts that ants will exhibit two difference capacities for discrimination. Stable populations would not form at nests with an average quality under a certain threshold as this case can be modelled as a stable cluster failing to form at a hypothesis with a low score. Stable populations would also not form at a nest with an average quality less than that of a nest which had already been located, as this case can be modelled as a cluster forming at the optimal hypothesis, due to sensitivity of convergence. Given these this prediction from the model, further work could be done to whether both of these capacities are present.

It has also been hypothesised that the method of quorum sensing employed by ants was based either on the rate of contact with other ants at a candidate nest site or the time before the first encounter with another ant at a candidate nest site[78]. The ability of quorum sensing SDS to converge to hypothesis with a high score indicates that frequency of contact with agents that share a hypothesis is sufficient to produce this behaviour, though it may not be necessary. Further work may therefore include experiments in which the frequency of contact is maintained as the time before the first contact is manipulated.

5.2 Future work

There are three main areas left unexplored by this thesis. The difference in speed of tandem running and transporting, the role of reverse tandem running and the role of agents which have been removed from the swarm.

Observations of ant nest site behaviour have reported that transporting is three times faster than tandem running [29, 28, 77], the effect of including this in quorum sensing SDS could be investigated. One mechanism for achieving this has been hypothesised in section 4.2.3.1, but not implemented.

Reverse tandem runs, where an ant leads another ant from a candidate nest site back to their original nest is a situation not directly analogous to any in quorum sensing SDS. A purpose of reverse tandem runs in nature has been investigated [77], and using a modified version of quorum sensing SDS which includes reverse tandem runs may help validate some hypotheses.

Another important aspect of ant behaviour which is not captured by quorum sensing SDS, is that ants are often performing other behaviours than searching for a new nest. In quorum sensing SDS, all agents are involved in the search, some will be evaluating hypotheses and others will be terminating, but real ants have many more activities that must be performed for a colony to survive and so must intelligently allocate time to these activities. Presumably each instance of SDS intends to locate a solution to a problem, a further variant could be developed which searches for a solution as normal but also allocates resources to the application of the current best hypothesised solution to the problem. For example, consider a case where SDS is being applied to find the best set of parameters for a video filter. Once a cluster has formed a portion of the swarm could be allocated to the task of applying the video filter to a video feed with the given parameters, the portion of the swarm being somehow determined by the size of the cluster. This would be particularly suitable for cases where the ideal solution changes over time. This form of online application of a solution to a search algorithm is likely to have many applications, and would be an interesting area of future research.

5.3 Impact

This thesis provides a formal specification for SDS which facilitates the development of variants and their combination. A library implementing the variants mentioned in this thesis is published online at the author's home page[1] and on GitHub[2] under the permissive Apache 2.0 free software license [1].

This thesis has provided a model which corroborates some hypotheses of ant behaviour such as indirect comparison of nest sites, comparison of nest site quality with a threshold value, and global quorum sensing from local information. This reinforces existing work and provides direction for investigation in the future.

This thesis collects the main variants of SDS and details their individual use cases.

This thesis presents a nature inspired algorithm that, is stable over a wide range of possible search spaces, and has two parameters by which the minimum solution quality threshold and minimum hypothesis validation threshold can be conveniently tuned.

[1]`https://www.aomartin.co.uk/sds-library/`
[2]`https://github.com/AndrewOwenMartin/sds`

Chapter 6

Glossary

activity the state of an agent which indicates the strength of the its current hypothesis.

agent an element of a swarm which maintains a hypothesis and an activity.

cluster a number of agents maintaining the same hypothesis.

confidence a value internal to an agent in quorum sensing SDS which represents the amount of evidence an agent has of a swarm reaching quorum..

confirmation reducing SDS a variant of SDS in which agents may remove other agents from the swarm after having had their hypothesis 'confirmed' by polling other agents with the same hypothesis.

context-free SDS a variant implementing $D^{\text{CONTEXT-FREE}}$.

context-sensitive SDS a variant implementing $D^{\text{CONTEXT-SENSITIVE}}$.

convergence time the number of iterations before a stable cluster forms.

diffusion phase the phase of SDS in which new hypotheses are generated, and potentially good hypotheses are distributed amongst the swarm.

distractor hypothesis the hypothesis with the second highest score in the search space.

distractor hypothesis score the score of the distractor hypothesis.

global activity the proportion of the swarm which are active at a given time.

homogeneous background noise the mean probability of an agent at any non-optimal hypothesis becoming active.

hypothesis a potential solution to a search problem.

independent reducing SDS a variant of SDS in which agents act independently.

iteration the performance of a diffusion phase and a test phase for all agents in a swarm.

mean cluster size the mean number of active agents maintaining the optimal hypothesis as a proportion of the total population.

microtest a function which partially evaluates a hypothesis. It takes a hypothesis as an argument and returns `true` if the hypotheses passes the partial evaluation.

minimum convergence criteria the minimum score of the optimal hypothesis below which a stable cluster cannot form.

multi-diffusion a variant of the diffusion phase where multiple diffusion behaviours are performed each iteration.

multi-testing a variant of the test phase where each agent performs more than one microtest.

one-step evolution function mean 1-step optimal cluster size evolution function.

optimal hypothesis the hypothesis with the maximum score in the search space.

optimal hypothesis score the score of the optimal hypothesis.

quorum sensing SDS a reducing SDS in which agents confidence value is determined by contacts with agents maintainint the same hypothesis.

quorum threshold a value against which an agent's internal confidence value is compared when deciding whether to begin terminating in quorum sensing SDS.

reducing SDS a variant of SDS in which agents may remove other agents from the swarm.

robustness the proportion of possible search spaces to which SDS can be successfully applied.

running-mean SDS a reducing SDS in which agents commence reducing once the average activity over time at their hypothesis reaches a given threshold.

score the probability of a specific hypothesis passing a randomly selected microtests.

search space the set of all possible hypotheses, representing the set of all potential solutions to a task.

search space size the number of possible hypotheses.

steady state the average number of agents at the optimal hypothesis upon convergence.

string-search SDS an application of SDS to locate the closest instansiation of a model substring of a larger search space string. For example, finding the location of the word 'circumambulate' in the text of Moby Dick, or the encoding of a protein in a string of DNA base pairs. A hypothesis denotes the location of the model in the search space, and there is one microtest for each character in the model. Each microtest passes if the letter at a particular offset in the model is the same letter in the search space at the same offset to the hypothesised location.

strong convergence criterion the requirements for a cluster to be stable over any amount of iterations..

swarm a collection of agents.

swarm size the number of agents in a swarm.

test phase the phase of SDS in which agents update their hypotheses as a result of partial evaluations.

time to first hit the number of iterations at which the probability that an agent is maintaining the optimal hypothesis reaches a given probability p.

Chapter 7

SDS Formalism

DH^{INFINITE} noise hypothesis selection.

DH^{STANDARD} alias of DH^{UNIFORM}.

DH^{UNIFORM} uniformly random hypothesis selection.

DH mode of hypothesis selection.

DP^{ACTIVE} active polling.

DP^{PASSIVE} passive polling.

D^{ACTIVE} active diffusion.

$D^{\text{CONFIRMATION}}$ confirmation reducing diffusion.

$D^{\text{CONTEXT-FREE}}$ context-free diffusion.

$D^{\text{CONTEXT-SENSITIVE}}$ context-sensitive diffusion.

D^{DUAL} dual diffusion.

D^{HERMIT} hermit diffusion.

$D^{\text{INDEPENDENT}}$ independent reducing diffusion.

$D^{\text{MULTI-DIFFUSION}}$ multi-diffusion.

D^{NOISE} noisy diffusion.

D^{PASSIVE} passive diffusion.

D^{PASSIVE} private internal state diffusion.

$D^{\text{QUORUM SENSING}}$ quorum sensing diffusion.

$D^{\text{RUNNING MEAN}}$ running mean diffusion.

D^{STANDARD} alias of D^{PASSIVE}.

D mode of diffusion.

H^{ACTIVITY} global activity threshold halting.

H^{AND} AND halting combinator.

H^{COLLAPSE} optimal hypothesis cluster collapse halting.

H^{FIXED} fixed iteration count halting.

$H^{\text{INDEFINITE}}$ indefinite halting.

H^{LARGEST} largest cluster size halting.

H^{OR} OR halting combinator.

H^{REDUCING} empty or all terminating swarm halting.

H^{STABLE} stable global activity halting.

H^{STRONG} strong halting.

H^{TIME} fixed wall-clock time halting.

H^{UNIQUE} unique hypothesis count halting.

H^{WEAK} weak halting.

H mode of halting.

$I^{\text{ASYNCHRONOUS}}$ asynchronous iteration.

I^{PARALLEL} parallel iteration.

I^{STANDARD} alias of $I^{\text{SYNCHRONOUS}}$.

$I^{\text{SYNCHRONOUS}}$ synchronous iteration.

I mode of iteration.

$TM^{\textbf{STANDARD}}$ alias of TM^{UNIFORM}.

$TM^{\textbf{UNIFORM}}$ uniform microtest selection.

TM mode of microtest selection.

$T^{\textbf{ACTIVE}}$ testing for active diffusion.

$T^{\textbf{BOOLEAN}}$ boolean testing.

$T^{\textbf{COMPARATIVE}}$ comparative testing.

$T^{\textbf{MULTI-TESTING}}$ multi-testing.

$T^{\textbf{OPTIMIST}}$ secret optimist testing.

$T^{\textbf{REDUCING}}$ reducing testing.

$T^{\textbf{STANDARD}}$ alias of T^{BOOLEAN}.

T mode of testing.

Chapter 8

Acronyms

SDS	Stochastic Diffusion Search.
ACO	Ant Colony Optimisation.
AI	Artificial Intelligence.
DDSM	Discrete Dynamical System Model.
DFO	Dispersive Flies Optimisation.
NFL	No Free Lunch theorems.
PSO	Particle Swarm Optimisation.
SSDS	standard SDS as defined in section 3.3.3, p.83.
TSP	Travelling Salesman Problem.

Chapter 9

Mathematical symbols

A Swarm size.

S Search space size.

α Optimal hypothesis score.

$\bar{c}_i$ Mean cluster size.

β Homogeneous background noise.

δ Distractor hypothesis score.

γ Steady state.

$\alpha_{\mathbf{min}}$ Minimum convergence criteria.

ζ Robustness.

$m_{\mathbf{max}}$ maximum memory length.

$m_{\mathbf{min}}$ minimum interactions.

p_α the probability of selecting the optimal hypothesis at random.

qt Quorum threshold.

Bibliography

[1] Apache Software Foundation. *Apache license, version 2.0.* URL: `https://www.apache.org/licenses/LICENSE-2.0.txt` (visited on 04/28/2020).

[2] C. Babbage. *Science and reform: Selected works of Charles Babbage*. Cambridge University Press, 1989.

[3] C. Babbage, P. Morrison, and E. Morrison. *On the principles and development of the calculator and other seminal writings*. Dover Publications, 2013. ISBN: 9780486320526. URL: `https://books.google.co.uk/books?id=FTXyAAAAQBAJ`.

[4] R. Beacham. "Heron of Alexandria's 'toy theatre' automaton: Reality, allusion and illusion". In: *Theatre, Performance and Analogue Technology: Historical Interfaces and Intermedialities*. Ed. by K. Reilly. Palgrave Studies in Performance and Technology. Palgrave Macmillan, 2013. ISBN: 9781137319685. URL: `http://books.google.co.uk/books?id=HeTQAQAAQBAJ`.

[5] J. M. Bishop, A. O. Martin, and E. J. Robinson. "Local termination criteria for Stochastic Diffusion Search: a comparison with the behaviour of ant nest-site selection". In: *International Conference on Computational Collective Intelligence*. Springer. 2016, pp. 474–486.

[6] J. M. Bishop. "Anarchic techniques for pattern classification." PhD thesis. University of Reading, 1989.

[7] J. M. Bishop. "Stochastic searching networks". In: *Proc. 1st IEE Conf. on Artifical neural networks*. 1989, pp. 329–331.

[8] J. M. Bishop and P. Torr. "The stochastic search network". In: *Neural networks for vision, speech and natural language*. Springer, 1992, pp. 370–387.

[9] J. M. Bishop and M. M. al-Rifaie. "Autopoiesis, creativity and dance". In: *Connection science* 29.1 (2017), pp. 21–35.

[10] R. J. Cant and C. S. Langensiepen. “Methods for automated object placement in virtual scenes”. In: *Computer Modelling and Simulation, 2009. UKSIM’09. 11th International Conference on*. IEEE. 2009, pp. 431–436.

[11] A. Colorni, M. Dorigo, and V. Maniezzo. “An investigation of some properties of an "Ant Algorithm"”. In: *PPSN*. Vol. 92. 1992, pp. 509–520.

[12] B. J. Copeland. “The Church-Turing thesis”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Winter 2017. Metaphysics Research Lab, Stanford University, 2017.

[13] M. De Mey. *The cognitive paradigm*. Sociology of the sciences Monographs. University of Chicago Press, 1992. ISBN: 9780226142593. URL: `http://books.google.co.uk/books?id=tMCXlAXnVb8C`.

[14] K. De Meyer. *Explorations in Stochastic Diffusion Search: Soft-and hardware implementations of biologically inspired spiking neuron Stochastic Diffusion Networks*. Tech. rep. Technical Report KDM/JMB/2000, 2000.

[15] K. De Meyer, J. M. Bishop, and S. J. Nasuto. “Stochastic diffusion: Using recruitment for search”. In: *Evolvability and interaction: Evolutionary substrates of communication, signalling, and perception in the dynamics of social complexity (ed. P. McOwan, K. Dautenhahn & CL Nehaniv) Technical Report* 393 (2003), pp. 60–65.

[16] K. De Meyer. “Foundations of Stochastic Diffusion Search”. PhD thesis. University of Reading, 2004.

[17] K. De Meyer, J. M. Bishop, and S. J. Nasuto. “Small-world effects in lattice Stochastic Diffusion Search”. In: *Artificial Neural Networks-ICANN 2002*. Springer, 2002, pp. 147–152.

[18] R. Descartes and J. Veitch. *Meditations on first philosophy*. Charles River Editors, 2018. ISBN: 9781632956460. URL: `https://books.google.co.uk/books?id=sApaDwAAQBAJ`.

[19] M. Dorigo and G. Di Caro. “Ant colony optimization: a new meta-heuristic”. In: *Proceedings of the 1999 congress on evolutionary computation-CEC99 (Cat. No. 99TH8406)*. Vol. 2. IEEE. 1999, pp. 1470–1477.

[20] A. Dornhaus and N. R. Franks. “Colony size affects collective decision-making in the ant Temnothorax albipennis”. In: *Insectes Sociaux* 53.4 (2006), pp. 420–427. ISSN: 1420-

9098. DOI: 10.1007/s00040-006-0887-4. URL: https://doi.org/10.1007/s00040-006-0887-4.

[21] H. L. Dreyfus. "A history of first step fallacies". In: *Minds and Machines* 22.2 (2012), pp. 87–99.

[22] H. L. Dreyfus. "Why Heideggerian AI failed and how fixing it would require making it more Heideggerian". In: *Philosophical psychology* 20.2 (2007), pp. 247–268.

[23] H. L. Dreyfus and S. E. Dreyfus. "Making a mind versus modelling the brain: Artificial intelligence back at a branchpoint". In: *Daedalus* 117.1 (1988), pp. 185–197.

[24] S. Dzeroski and L. Todorovski. *Computational discovery of scientific knowledge: Introduction, techniques, and applications in environmental and life sciences*. LNCS sub-library: Artificial intelligence. Springer, 2007. ISBN: 9783540739197. URL: https://books.google.co.uk/books?id=-0EEQrnY7yQC.

[25] R. Eberhart and J. Kennedy. "Particle swarm optimization". In: *Proceedings of the IEEE international conference on neural networks*. Vol. 4. Citeseer. 1995, pp. 1942–1948.

[26] D. Ferrucci et al. "Watson: Beyond Jeopardy!" In: *Artificial Intelligence* 199 (2013), pp. 93–105.

[27] N. R. Franks et al. "Speed versus accuracy in collective decision making". In: *Proceedings of the Royal Society of London B: Biological Sciences* 270.1532 (2003), pp. 2457–2463. ISSN: 0962-8452. DOI: 10.1098/rspb.2003.2527. eprint: http://rspb.royalsocietypublishing.org/content/270/1532/2457.full.pdf. URL: http://rspb.royalsocietypublishing.org/content/270/1532/2457.

[28] N. R. Franks et al. "How ants use quorum sensing to estimate the average quality of a fluctuating resource". In: *Scientific reports* 5 (2015), p. 11890.

[29] N. R. Franks et al. "Speed versus accuracy in decision-making ants: Expediting politics and policy implementation". In: *Philosophical Transactions of the Royal Society of London B: Biological Sciences* 364.1518 (2009), pp. 845–852.

[30] W. Freeman. *How brains make up their minds*. Maps of the Mind Series. Phoenix, 2000. ISBN: 9780753810682. URL: https://books.google.co.uk/books?id=a%5C_GuQgAACAAJ.

[31] H. Grech-Cini. "Locating facial features". PhD thesis. University of Reading, 1995.

[32] H. Grech-Cini and G. T. McKee. “Locating the mouth region in images of human faces”. In: *Optical Tools for Manufacturing and Advanced Automation*. International Society for Optics and Photonics. 1993, pp. 458–465.

[33] R. M. Harnish. *Minds, brains, computers: An historical introduction to the foundations of cognitive science*. Blackwell Publishers, 2001.

[34] G. Hatfield. “René Descartes”. In: *The Stanford Encyclopedia of Philosophy*. Ed. by E. N. Zalta. Summer 2014. The Metaphysics Research Lab, Stanford University, 2014.

[35] D. O. Hebb. *The organization of behaviour*. New York: Wiley & Sons, 1949.

[36] J. Hellinger et al. “The flashlight fish Anomalops katoptron uses bioluminescent light to detect prey in the dark”. In: *PLoS One* 12.2 (2017).

[37] G. F. Hinton. “A parallel computation that assigns canonical object-based frames of reference”. In: *Proceedings of the 7th international joint conference on Artificial intelligence-Volume 2*. Morgan Kaufmann Publishers Inc. 1981, pp. 683–685.

[38] A. Hodges. *Alan Turing: The enigma*. Random House, 2014. ISBN: 9781784700089.

[39] B. Jones and M. Nishiguchi. “Counterillumination in the hawaiian bobtail squid, Euprymna scolopes Berry (Mollusca: Cephalopoda)”. In: *Marine Biology* 144.6 (2004), pp. 1151–1155.

[40] D. Jones. “Constrained Stochastic Diffusion Search”. In: *SCARP 2002*. 2002.

[41] J. Kennedy, R. Eberhart, and Y. Shi. *Swarm intelligence*. The Morgan Kaufmann Series in Evolutionary Computation. Morgan Kaufmann Publishers, 2001. ISBN: 9781558605954. URL: `http://books.google.co.uk/books?id=vOx-QV3sRQsC`.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Advances in neural information processing systems*. 2012, pp. 1097–1105.

[43] J. G. Landels. “Water-clocks and time measurement in classical antiquity”. In: *Endeavour* 3.1 (1979), pp. 32–37.

[44] J. Lederberg. “DENDRAL-64 a system for computer construction”. In: *Enumeration, and Notation of Organic Molecules as Tree Structures and Cyclic Graphs. NASA Interim Report* (1964).

[45] H. Lee. *The Republic*. Classics Series. Penguin Books, 2003. ISBN: 9780140449143. URL: `https://books.google.co.uk/books?id=v%5C_BpLwLKFPsC`.

[46] J. Lighthill. "Artificial Intelligence: A general survey". In: *Artificial Intelligence: A paper symposium* (1973).

[47] C. Lupp and E. G. Ruby. "Vibrio fischeri uses two quorum-sensing systems for the regulation of early and late colonization factors". In: *Journal of bacteriology* 187.11 (2005), pp. 3620–3629.

[48] C. W. Marshall. "Sophocles' Nauplius and Heron of Alexandria's mechanical theatre". In: *Shards from Kolonos: Studies in Sophoclean fragments*. Ed. by A. H. Sommerstein. Vol. 34. Le Rane : Collana di studi e testi: Studi. Levante, 2003. ISBN: 9788879493079.

[49] A. O. Martin et al. "Local termination criteria for Swarm Intelligence: a comparison between local Stochastic Diffusion Search and ant nest-site selection". In: *Transactions on Computational Collective Intelligence XXXII*. Springer, 2019, pp. 140–166.

[50] J. McCarthy et al. *A proposal for the dartmouth summer research project on artificial intelligence*. 1955. URL: `http://www-formal.stanford.edu/jmc/history/dartmouth/dartmouth.html`.

[51] J. McCarthy. "Mechanisation of thought processes". In: *Proc. Symposium*. Vol. 1. 1958.

[52] J. McCarthy. "Programs with common sense". In: *Proc. Symposium on the Mechanization of Thought Processes*. Vol. 1. Proc Teddington Conference on the Mechanization of Thought Processes,59,p75-91;sip;fcs;luger. 1958, pp. 77–84.

[53] W. S. McCulloch and W. Pitts. "A logical calculus of the ideas immanent in nervous activity". In: *The bulletin of mathematical biophysics* 5.4 (1943), pp. 115–133.

[54] L. F. Menabrea and A. Lovelace. *Sketch of the analytical engine invented by Charles Babbage*. 1842.

[55] M. B. Miller and B. L. Bassler. "Quorum sensing in bacteria". In: *Annual Reviews in Microbiology* 55.1 (2001), pp. 165–199.

[56] M. L. Minsky et al. *Perceptrons: An Introduction to Computational Geometry*. Mit Press. Cambridge, 1988. ISBN: 9780262631112. URL: `https://books.google.co.uk/books?id=U-O9BHlTvJIC`.

[57] M. Minsky and S. Papert. *Perceptrons: An introduction to computational geometry*. The MIT Press, 1969. URL: `https://books.google.co.uk/books?id=%5C_q6EnQEACAAJ`.

[58] R. K. Moore. "A comparison of the data requirements of automatic speech recognition systems and human listeners." In: *INTERSPEECH*. 2003.

[59] D. R. Myatt and J. M. Bishop. *Data driven Stochastic Diffusion Search for robust estimation*. 2003.

[60] D. R. Myatt, J. M. Bishop, and S. J. Nasuto. "Minimum stable convergence criteria for Stochastic Diffusion Search". In: *Electronics Letters* 40.2 (2004), pp. 112–113.

[61] D. R. Myatt, S. J. Nasuto, and J. M. Bishop. "Exploration and exploitation in Stochastic Diffusion Search". In: *Exploration vs Exploitation in Naturally Inspired Search*. Symposium on Nature Inspired Systems. AISB. 2006.

[62] D. R. Myatt. "Analysis of Stochastic Diffusion Search and its application to robust estimation". PhD thesis. University of Reading, 2005.

[63] S. J. Nasuto. "Resource allocation analysis of the Stochastic Diffusion Search". PhD thesis. University of Reading, 1999.

[64] S. J. Nasuto and J. M. Bishop. "Convergence analysis of Stochastic Diffusion Search". In: *PARALLEL ALGORITHMS AND APPLICATION* 14.2 (1999), pp. 89–107.

[65] S. J. Nasuto and J. M. Bishop. "Stabilizing swarm intelligence search via positive feedback resource allocation". In: *Nature Inspired Cooperative Strategies for Optimization (NICSO 2007)*. Springer, 2008, pp. 115–123.

[66] S. J. Nasuto and J. M. Bishop. "Steady state resource allocation analysis of the Stochastic Diffusion Search". In: *Biologically Inspired Cognitive Architectures* 12 (2015), pp. 65–76.

[67] S. J. Nasuto, J. M. Bishop, and S. Lauria. "Time complexity analysis of the Stochastic Diffusion Search". In: *NC*. 1998, pp. 260–266.

[68] A. Newell, J. C. Shaw, and H. A. Simon. "Report on a general problem-solving program". In: *IFIP Congress*. 1959, pp. 256–264.

[69] A. Newell and H. A. Simon. "Computer science as empirical inquiry: Symbols and search". In: *Communications of the ACM* 19.3 (1976), pp. 113–126.

[70] A. Newell and H. A. Simon. "The logic theory machine–A complex information processing system". In: *Information Theory, IRE Transactions on* 2.3 (1956), pp. 61–79.

[71] W. Newton. "Newton's London journal of arts and sciences". In: Newton and son, 1866, pp. 333–334. URL: `http://books.google.co.uk/books?id=ElUEAAAAQAAJ`.

[72] Q. Nguyen and J. M. Bishop. “Exploration of Data Driven SDS vs. Coupled SDS”. Cognitive Computing MSc. MA thesis. London, United Kingdom: Goldsmiths, University of London, 2008.

[73] F. Nietzsche. “On the genealogy of morals. 1887”. In: *Basic Writings of Nietzsche* (1967), pp. 439–599.

[74] A. B. Novikoff. *On convergence proofs for perceptrons*. Tech. rep. STANFORD RESEARCH INST MENLO PARK CA, 1963.

[75] M. G. H. Omran and A. Salman. “Probabilistic stochastic diffusion search”. In: *Swarm Intelligence*. Springer, 2012, pp. 300–307. URL: `http://link.springer.com/chapter/10.1007/978-3-642-32650-9_31`.

[76] M. G. H. Omran et al. “Stochastic Diffusion Search for continuous global optimization”. In: *ICSI 2011: International conference on swarm intelligence*. 2011.

[77] R. Planqué et al. “Why do house-hunting ants recruit in both directions?” In: *Naturwissenschaften* 94.11 (2007), pp. 911–918. ISSN: 1432-1904. DOI: `10.1007/s00114-007-0273-8`. URL: `https://doi.org/10.1007/s00114-007-0273-8`.

[78] S. C. Pratt. “Quorum sensing by encounter rates in the ant Temnothorax albipennis”. In: *Behavioral Ecology* 16.2 (2005), pp. 488–496.

[79] S. C. Pratt et al. “Quorum sensing, recruitment, and collective decision-making during colony emigration by the ant Leptothorax albipennis”. In: *Behavioral Ecology and Sociobiology* 52.2 (2002), pp. 117–127.

[80] C. W. Reynolds. “Flocks, herds and schools: A distributed behavioral model”. In: *ACM SIGGRAPH computer graphics* 21.4 (1987), pp. 25–34.

[81] M. M. al-Rifaie. “Dispersive flies optimisation”. In: *Computer Science and Information Systems (FedCSIS), 2014 Federated Conference on*. IEEE. 2014, pp. 529–538.

[82] M. M. al-Rifaie. “Information sharing impact of Stochastic Diffusion Search on population-based algorithms”. PhD thesis. Goldsmiths, University of London, 2011.

[83] M. M. al-Rifaie. “Perceived simplicity and complexity in nature”. In: *AISB 2017: Computational Architectures for Animal Cognition* (2017), pp. 299–305.

[84] M. M. al-Rifaie and J. M. Bishop. “Stochastic Diffusion Search review”. In: *Paladyn, Journal of Behavioral Robotics* 4.3 (2013), pp. 155–173.

[85] M. M. al-Rifaie and J. M. Bishop. "The mining game: A brief introduction to the Stochastic Diffusion Search metaheuristic". In: *Q: The magazine of AISB* 130 (2010), pp. 8–9.

[86] M. M. al-Rifaie, J. M. Bishop, and T. Blackwell. "An investigation into the merger of Stochastic Diffusion Search and Particle Swarm Optimisation". In: *Proceedings of the 13th annual conference on genetic and evolutionary computation*. ACM. 2011, pp. 37–44.

[87] E. J. H. Robinson. "Distributed decisions: new insight From radio tagged ants". Whitehead Lecture at Goldsmiths, University of London. 2013. URL: `https://www.gold.ac.uk/calendar/?id=6158`.

[88] E. Robinson and W. Mandecki. "Distributed decisions: new insights from radio-tagged ants". In: *Ant Colonies: Behavior in Insects and Computer Applications* (2011), pp. 109–128.

[89] F. Rosenblatt. "The perceptron: A probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), p. 386.

[90] M. Rosheim. *Robot evolution: The development of anthrobotics*. Wiley interscience publication. Wiley, 1994. ISBN: 9780471026228. URL: `https://books.google.co.uk/books?id=IxtL54iiDPUC`.

[91] D. Rumelhart and J. McClelland. *Parallel distributed processing: Explorations in the microstructures of cognition. Volume 1: Foundations*. A Bradford Book. MIT Press, 1986. ISBN: 9780262680530. URL: `https://books.google.co.uk/books?id=7JZE1YFZ4CUC`.

[92] S. Russell and P. Norvig. *Artificial intelligence: A modern approach*. Prentice Hall series in artificial intelligence. Prentice Hall, 2003. ISBN: 9780130803023. URL: `http://books.google.co.uk/books?id=0y0rQgAACAAJ`.

[93] R. C. Schank and R. P. Abelson. *Scripts, plans, and knowledge*. Yale University, 1975.

[94] B. Schrauwen, D. Verstraeten, and J. Van Campenhout. "An overview of reservoir computing: Theory, applications and implementations". In: *Proceedings of the 15th european symposium on artificial neural networks. p. 471-482 2007*. 2007, pp. 471–482.

[95] J. R. Searle. "Minds, brains, and programs". In: *Behavioral and brain sciences* 3.03 (1980), pp. 417–424.

[96] T. D. Seeley and P. K. Visscher. "Quorum sensing during nest-site selection by honeybee swarms". In: *Behavioral Ecology and Sociobiology* 56.6 (2004), pp. 594–601.

[97] H. A. Simon and A. Newell. "Heuristic problem solving: The next advance in operations research". In: *Operations research* 6.1 (1958), pp. 1–10.

[98] A. Tanner. *Google seeks world of instant translations*. `http://www.reuters.com/article/2007/03/28/us-google-translate-idUSN1921881520070328`. (Accessed on 24th March 2015). Reuters, 2007. (Visited on 03/24/2014).

[99] A. Turing. "Intelligent machinery (1948)". In: *The essential turing*. Ed. by B. J. Copeland. Clarendon Press, 2004, p. 395.

[100] A. M. Turing. "On computable numbers, with an application to the Entscheidungsproblem". In: *Proceedings of the London mathematical society* 2.1 (1937), pp. 230–265.

[101] P. Virtanen et al. "SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python". In: *Nature Methods* 17 (2020), pp. 261–272. DOI: `https://doi.org/10.1038/s41592-019-0686-2`.

[102] M. Warriner. *Hardware Stochastic Diffusion Search*. Tech. rep. Dept. Cybernetics, University of Reading, UK, 2000.

[103] M. Wheeler. *Reconstructing the cognitive world: The next step*. MIT press, 2005.

[104] H. Williams. "Stochastic Diffusion Search processes". Cognitive Computing MSc. MA thesis. London, United Kingdom: Goldsmiths, University of London, 2010.

[105] H. Williams and J. M. Bishop. "Stochastic Diffusion Search: A comparison of swarm intelligence parameter estimation algorithms with RANSAC". In: *Algorithms* 7.2 (2014), pp. 206–228.

[106] T. Winograd. "Understanding natural language". In: *Cognitive psychology* 3.1 (1972), pp. 1–191.

[107] D. H. Wolpert. "The lack of a priori distinctions between learning algorithms". In: *Neural computation* 8.7 (1996), pp. 1341–1390.

[108] D. H. Wolpert. "What the no free lunch theorems really mean; how to improve search algorithms". In: *Santa Fe Institute*. Vol. 7. 2012.

[109] D. H. Wolpert, W. G. Macready, et al. *No free lunch theorems for search*. Tech. rep. Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

Appendix A

Tables of results

A.1 Speed of diffusion of consensus in independent reducing SDS and confirmation reducing SDS

			SSDS			IrSDS			CrSDS		
α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c	μ	σ	c	μ	σ
0.1	0.0	-0.40	0	-	-	0	-	-	1	1043	0.00
0.2	0.0	-0.30	0	-	-	0	-	-	0	-	-
0.2	0.1	-0.33	0	-	-	0	-	-	0	-	-
0.3	0.0	-0.20	0	-	-	0	-	-	0	-	-
0.3	0.1	-0.23	0	-	-	0	-	-	0	-	-
0.3	0.2	-0.26	0	-	-	0	-	-	0	-	-
0.4	0.0	-0.10	0	-	-	0	-	-	1	558	0.00
0.4	0.1	-0.13	0	-	-	1	403	0.00	2	1072	3.00
0.4	0.2	-0.16	0	-	-	0	-	-	1	1229	0.00
0.4	0.3	-0.19	0	-	-	0	-	-	1	1432	0.00
0.5	0.0	$+0.00$	0	-	-	6	259	151.38	11	440	302.08
0.5	0.1	-0.03	0	-	-	3	237	120.51	4	1141	26.98
0.5	0.2	-0.06	0	-	-	3	373	77.19	1	1283	0.00
0.5	0.3	-0.09	0	-	-	2	276	10.50	4	1500	41.19
0.5	0.4	-0.12	0	-	-	2	280	22.50	4	1728	26.42
0.6	0.0	$+0.10$	36	1001	0.00	29	32	5.92	39	89	217.01
0.6	0.1	$+0.07$	28	1001	0.00	21	61	96.36	42	147	264.32
0.6	0.2	$+0.04$	25	1001	0.00	13	50	12.52	23	218	386.54

Continued on next page

Table A.1 – *Continued from previous page*

			SSDS			IrSDS			CrSDS		
α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c	μ	σ	c	μ	σ
0.6	0.3	+0.01	0	-	-	11	194	145.92	16	603	590.09
0.6	0.4	−0.02	0	-	-	5	304	126.92	10	1139	592.73
0.6	0.5	−0.07	0	-	-	1	378	0.00	2	2020	15.50
0.7	0.0	+0.20	66	1001	0.00	48	20	2.98	67	25	3.56
0.7	0.1	+0.17	67	1001	0.00	45	22	2.26	59	29	3.49
0.7	0.2	+0.14	64	1001	0.00	42	42	70.96	61	114	297.46
0.7	0.3	+0.11	54	1001	0.00	35	40	38.42	45	143	365.24
0.7	0.4	+0.08	31	1001	0.00	32	51	35.94	42	103	237.30
0.7	0.5	+0.03	2	1001	0.00	15	160	130.68	34	508	672.75
0.7	0.6	−0.01	0	-	-	8	219	116.57	9	1497	1012.34
0.8	0.0	+0.30	89	1001	0.00	61	15	1.85	75	19	2.17
0.8	0.1	+0.27	80	1001	0.00	62	17	2.27	74	22	2.27
0.8	0.2	+0.24	78	1001	0.00	64	19	3.15	80	41	133.05
0.8	0.3	+0.21	75	1001	0.00	56	29	51.11	72	31	3.50
0.8	0.4	+0.18	67	1001	0.00	54	26	3.58	71	38	4.45
0.8	0.5	+0.13	63	1001	0.00	61	36	31.16	68	79	233.91
0.8	0.6	+0.09	40	1001	0.00	53	53	40.59	52	142	375.73
0.8	0.7	+0.03	19	1001	0.00	28	162	129.18	25	637	794.28
0.9	0.0	+0.40	97	1001	0.00	83	13	2.01	94	17	1.85
0.9	0.1	+0.37	91	1001	0.00	85	14	2.07	91	19	2.40
0.9	0.2	+0.34	92	1001	0.00	86	15	2.02	89	21	2.60
0.9	0.3	+0.31	89	1001	0.00	86	17	2.05	95	24	2.24
0.9	0.4	+0.28	95	1001	0.00	83	20	2.66	87	29	3.72
0.9	0.5	+0.23	83	1001	0.00	83	23	3.72	91	36	4.42
0.9	0.6	+0.19	90	1001	0.00	83	33	37.83	87	75	257.85
0.9	0.7	+0.13	83	1001	0.00	72	39	6.88	77	72	11.67
0.9	0.8	+0.07	60	1001	0.00	60	108	105.34	68	234	591.30
1.0	0.0	+0.50	100	1001	0.00	100	11	1.25	100	14	1.25
1.0	0.1	+0.47	100	1001	0.00	100	12	1.50	100	16	1.47
1.0	0.2	+0.44	100	1001	0.00	100	13	1.81	100	18	1.71
1.0	0.3	+0.41	100	1001	0.00	100	14	2.17	100	21	1.90
1.0	0.4	+0.38	100	1001	0.00	100	16	2.03	100	25	2.42
1.0	0.5	+0.33	100	1001	0.00	100	18	2.62	100	69	277.63
1.0	0.6	+0.29	100	1001	0.00	100	22	3.00	100	37	3.77
1.0	0.7	+0.23	100	1001	0.00	100	27	4.32	100	51	5.05
1.0	0.8	+0.17	100	1001	0.00	100	39	28.07	100	131	498.36

Continued on next page

Table A.1 – *Continued from previous page*

			SSDS			IrSDS			CrSDS		
α	β	$\alpha - \alpha_{\text{min}}$	c	μ	σ	c	μ	σ	c	μ	σ
1.0	0.9	+0.09	100	1001	0.00	100	65	13.89	100	282	966.45

Table A.1: Stability of cluster formation of reducing SDS. For each value of α and β in the interval $[0, 1]$ with a step of 0.1 where $\alpha > \beta$ and showing the amount which the optimal hypothesis score is greater than the minimum convergence criteria ($\alpha - \alpha_{\text{min}}$). Showing for each of standard SDS, independent reducing SDS and confirmation reducing SDS: c, the number of times out of 100 the algorithm successfully converged to the optimal hypothesis; μ and σ, the mean average and standard deviation of the number of iterations before halting in cases of successful convergence

A.2 Robustness of reducing SDS

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
100	0.1	0.0	−0.40	62	1001	0	38	1001	0	0	100	819	492	0	-	-	0	100	889	376	0	-	-	0
100	0.2	0.0	−0.30	97	1001	0	3	1001	0	0	100	326	129	0	-	-	0	100	354	96	0	-	-	0
100	0.2	0.1	−0.33	9	1001	0	91	1001	0	0	5	92	16	95	100	23	0	17	117	17	83	113	12	0
100	0.3	0.0	−0.20	100	1001	0	0	-	-	0	100	151	41	0	-	-	0	100	173	34	0	-	-	0
100	0.3	0.1	−0.23	61	1001	0	39	1001	0	0	44	86	15	56	95	20	0	61	103	13	39	111	15	0
100	0.3	0.2	−0.26	13	1001	0	87	1001	0	0	6	52	6	94	52	8	0	9	76	7	91	71	6	0
100	0.4	0.0	−0.10	100	1001	0	0	-	-	0	100	65	13	0	-	-	0	100	77	12	0	-	-	0
100	0.4	0.1	−0.13	98	1001	0	2	1001	0	0	96	68	12	4	89	21	0	100	84	10	0	-	-	0
100	0.4	0.2	−0.16	59	1001	0	41	1001	0	0	43	50	8	57	51	7	0	66	67	7	34	71	6	0
100	0.4	0.3	−0.19	15	1001	0	85	1001	0	0	7	34	2	93	36	4	0	14	56	4	86	57	4	0
100	0.5	0.0	+0.00	100	1001	0	0	-	-	0	100	31	5	0	-	-	0	100	35	4	0	-	-	0
100	0.5	0.1	−0.03	100	1001	0	0	-	-	0	100	38	6	0	-	-	0	100	46	5	0	-	-	0
100	0.5	0.2	−0.06	100	1001	0	0	-	-	0	98	40	5	2	44	4	0	100	54	5	0	-	-	0
100	0.5	0.3	−0.09	82	1001	0	18	1001	0	0	45	33	4	55	34	4	0	83	54	4	17	55	3	0
100	0.5	0.4	−0.12	25	1001	0	75	1001	0	0	7	29	4	93	27	2	0	17	50	4	83	49	3	0
100	0.6	0.0	+0.10	100	1001	0	0	-	-	0	100	18	2	0	-	-	0	100	21	2	0	-	-	0
100	0.6	0.1	+0.07	100	1001	0	0	-	-	0	100	21	2	0	-	-	0	100	26	2	0	-	-	0
100	0.6	0.2	+0.04	100	1001	0	0	-	-	0	100	25	3	0	-	-	0	100	33	3	0	-	-	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
100	0.6	0.3	$+0.01$	100	1001	0	0	-	-	0	100	27	3	0	-	-	0	100	40	4	0	-	-	0
100	0.6	0.4	-0.02	99	1001	0	1	1001	0	0	49	25	2	51	25	3	0	98	46	3	2	50	0	0
100	0.6	0.5	-0.07	46	1001	0	54	1001	0	0	5	21	1	95	22	2	0	25	47	3	75	47	3	0
100	0.7	0.0	$+0.20$	100	1001	0	0	-	-	0	100	13	1	0	-	-	0	100	16	1	0	-	-	0
100	0.7	0.1	$+0.17$	100	1001	0	0	-	-	0	100	15	1	0	-	-	0	100	19	1	0	-	-	0
100	0.7	0.2	$+0.14$	100	1001	0	0	-	-	0	100	17	1	0	-	-	0	100	23	2	0	-	-	0
100	0.7	0.3	$+0.11$	100	1001	0	0	-	-	0	100	19	2	0	-	-	0	100	27	2	0	-	-	0
100	0.7	0.4	$+0.08$	100	1001	0	0	-	-	0	97	21	2	3	23	2	0	100	34	3	0	-	-	0
100	0.7	0.5	$+0.03$	100	1001	0	0	-	-	0	61	21	2	39	22	2	0	100	42	3	0	-	-	0
100	0.7	0.6	-0.01	91	1001	0	9	1001	0	0	14	19	1	86	19	2	0	38	47	3	62	47	4	0
100	0.8	0.0	$+0.30$	100	1001	0	0	-	-	0	100	11	1	0	-	-	0	100	13	1	0	-	-	0
100	0.8	0.1	$+0.27$	100	1001	0	0	-	-	0	100	12	1	0	-	-	0	100	15	1	0	-	-	0
100	0.8	0.2	$+0.24$	100	1001	0	0	-	-	0	100	13	1	0	-	-	0	100	18	1	0	-	-	0
100	0.8	0.3	$+0.21$	100	1001	0	0	-	-	0	100	14	1	0	-	-	0	100	21	1	0	-	-	0
100	0.8	0.4	$+0.18$	100	1001	0	0	-	-	0	100	16	1	0	-	-	0	100	26	2	0	-	-	0
100	0.8	0.5	$+0.13$	100	1001	0	0	-	-	0	98	18	2	2	18	0	0	100	33	2	0	-	-	0
100	0.8	0.6	$+0.09$	100	1001	0	0	-	-	0	52	18	2	48	18	2	0	100	42	4	0	-	-	0
100	0.8	0.7	$+0.03$	100	1001	0	0	-	-	0	10	16	1	90	17	1	0	60	51	4	40	51	4	0
100	0.9	0.0	$+0.40$	100	1001	0	0	-	-	0	100	9	1	0	-	-	0	100	12	1	0	-	-	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha-\alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
100	0.9	0.1	$+0.37$	100	1001	0	0	-	-	0	100	10	1	0	-	-	0	100	13	1	0	-	-	0
100	0.9	0.2	$+0.34$	100	1001	0	0	-	-	0	100	11	1	0	-	-	0	100	16	1	0	-	-	0
100	0.9	0.3	$+0.31$	100	1001	0	0	-	-	0	100	12	1	0	-	-	0	100	18	1	0	-	-	0
100	0.9	0.4	$+0.28$	100	1001	0	0	-	-	0	100	14	1	0	-	-	0	100	22	2	0	-	-	0
100	0.9	0.5	$+0.23$	100	1001	0	0	-	-	0	100	15	1	0	-	-	0	100	27	2	0	-	-	0
100	0.9	0.6	$+0.19$	100	1001	0	0	-	-	0	89	16	1	11	17	2	0	100	35	3	0	-	-	0
100	0.9	0.7	$+0.13$	100	1001	0	0	-	-	0	39	16	1	61	16	1	0	100	45	4	0	-	-	0
100	0.9	0.8	$+0.07$	100	1001	0	0	-	-	0	8	15	1	92	15	1	0	79	62	7	21	63	6	0
100	1.0	0.0	$+0.50$	100	1001	0	0	-	-	0	100	8	0	0	-	-	0	100	10	1	0	-	-	0
100	1.0	0.1	$+0.47$	100	1001	0	0	-	-	0	100	9	1	0	-	-	0	100	12	1	0	-	-	0
100	1.0	0.2	$+0.44$	100	1001	0	0	-	-	0	100	9	1	0	-	-	0	100	14	1	0	-	-	0
100	1.0	0.3	$+0.41$	100	1001	0	0	-	-	0	100	10	1	0	-	-	0	100	16	1	0	-	-	0
100	1.0	0.4	$+0.38$	100	1001	0	0	-	-	0	100	11	1	0	-	-	0	100	19	1	0	-	-	0
100	1.0	0.5	$+0.33$	100	1001	0	0	-	-	0	100	13	1	0	-	-	0	100	24	2	0	-	-	0
100	1.0	0.6	$+0.29$	100	1001	0	0	-	-	0	100	14	1	0	-	-	0	100	30	2	0	-	-	0
100	1.0	0.7	$+0.23$	100	1001	0	0	-	-	0	86	15	1	14	16	1	0	100	39	4	0	-	-	0
100	1.0	0.8	$+0.17$	100	1001	0	0	-	-	0	30	14	1	70	15	1	0	100	56	5	0	-	-	0
100	1.0	0.9	$+0.09$	100	1001	0	0	-	-	0	4	14	1	96	14	1	0	99	97	12	1	91	0	0
1000	0.1	0.0	-0.40	15	1001	0	85	1001	0	0	15	5787	3037	0	-	-	85	79	4218	2219	0	-	-	21

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
1000	0.2	0.0	-0.30	28	1001	0	72	1001	0	0	85	3080	1949	0	-	-	15	98	3048	1863	0	-	-	2
1000	0.2	0.1	-0.33	2	1001	0	98	1001	0	0	0	-	-	100	210	60	0	0	-	-	100	149	23	0
1000	0.3	0.0	-0.20	51	1001	0	49	1001	0	0	100	1184	949	0	-	-	0	100	1255	622	0	-	-	0
1000	0.3	0.1	-0.23	3	1001	0	97	1001	0	0	1	153	0	99	222	79	0	0	-	-	100	148	21	0
1000	0.3	0.2	-0.26	1	1001	0	99	1001	0	0	0	-	-	100	101	20	0	0	-	-	100	107	12	0
1000	0.4	0.0	-0.10	69	1001	0	31	1001	0	0	100	354	146	0	-	-	0	100	459	134	0	-	-	0
1000	0.4	0.1	-0.13	11	1001	0	89	1001	0	0	14	202	58	86	211	50	0	8	148	16	92	148	22	0
1000	0.4	0.2	-0.16	5	1001	0	95	1001	0	0	2	93	5	98	103	21	0	1	83	0	99	106	12	0
1000	0.4	0.3	-0.19	0	-	-	100	1001	0	0	2	71	11	98	68	11	0	0	-	-	100	87	8	0
1000	0.5	0.0	$+0.00$	98	1001	0	2	1001	0	0	100	98	29	0	-	-	0	100	106	32	0	-	-	0
1000	0.5	0.1	-0.03	63	1001	0	37	1001	0	0	82	118	32	18	160	36	0	88	126	22	12	141	14	0
1000	0.5	0.2	-0.06	31	1001	0	69	1001	0	0	21	94	22	79	100	19	0	22	95	10	78	105	11	0
1000	0.5	0.3	-0.09	8	1001	0	92	1001	0	0	7	63	10	93	66	11	0	5	83	5	95	89	9	0
1000	0.5	0.4	-0.12	0	-	-	100	1001	0	0	0	-	-	100	49	8	0	0	-	-	100	77	6	0
1000	0.6	0.0	$+0.10$	100	1001	0	0	-	-	0	100	30	5	0	-	-	0	100	37	6	0	-	-	0
1000	0.6	0.1	$+0.07$	100	1001	0	0	-	-	0	100	40	9	0	-	-	0	100	46	8	0	-	-	0
1000	0.6	0.2	$+0.04$	100	1001	0	0	-	-	0	96	53	13	4	87	10	0	98	64	13	2	94	0	0
1000	0.6	0.3	$+0.01$	100	1001	0	0	-	-	0	53	55	9	47	64	10	0	78	77	10	22	86	6	0
1000	0.6	0.4	-0.02	24	1001	0	76	1001	0	0	3	48	3	97	48	7	0	21	75	6	79	77	5	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\text{min}}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
1000	0.6	0.5	-0.07	0	-	-	100	1001	0	0	0	-	-	100	38	5	0	2	74	5	98	72	5	0
1000	0.7	0.0	$+0.20$	100	1001	0	0	-	-	0	100	20	3	0	-	-	0	100	24	3	0	-	-	0
1000	0.7	0.1	$+0.17$	100	1001	0	0	-	-	0	100	22	3	0	-	-	0	100	28	4	0	-	-	0
1000	0.7	0.2	$+0.14$	100	1001	0	0	-	-	0	100	28	6	0	-	-	0	100	35	5	0	-	-	0
1000	0.7	0.3	$+0.11$	100	1001	0	0	-	-	0	96	34	7	4	52	7	0	100	44	6	0	-	-	0
1000	0.7	0.4	$+0.08$	100	1001	0	0	-	-	0	70	39	7	30	47	6	0	94	58	9	6	73	5	0
1000	0.7	0.5	$+0.03$	100	1001	0	0	-	-	0	16	35	3	84	38	4	0	63	66	6	37	70	5	0
1000	0.7	0.6	-0.01	14	1001	0	86	1001	0	0	3	32	2	97	30	3	0	6	74	4	94	71	4	0
1000	0.8	0.0	$+0.30$	100	1001	0	0	-	-	0	100	15	2	0	-	-	0	100	19	2	0	-	-	0
1000	0.8	0.1	$+0.27$	100	1001	0	0	-	-	0	100	17	3	0	-	-	0	100	22	3	0	-	-	0
1000	0.8	0.2	$+0.24$	100	1001	0	0	-	-	0	100	19	3	0	-	-	0	100	26	3	0	-	-	0
1000	0.8	0.3	$+0.21$	100	1001	0	0	-	-	0	100	23	4	0	-	-	0	100	31	5	0	-	-	0
1000	0.8	0.4	$+0.18$	100	1001	0	0	-	-	0	97	28	5	3	45	7	0	100	39	7	0	-	-	0
1000	0.8	0.5	$+0.13$	100	1001	0	0	-	-	0	71	30	4	29	35	4	0	99	50	7	1	74	0	0
1000	0.8	0.6	$+0.09$	100	1001	0	0	-	-	0	15	29	3	85	31	3	0	69	63	6	31	71	5	0
1000	0.8	0.7	$+0.03$	100	1001	0	0	-	-	0	1	23	0	99	26	3	0	15	70	4	85	74	5	0
1000	0.9	0.0	$+0.40$	100	1001	0	0	-	-	0	100	12	2	0	-	-	0	100	16	2	0	-	-	0
1000	0.9	0.1	$+0.37$	100	1001	0	0	-	-	0	100	14	2	0	-	-	0	100	18	2	0	-	-	0
1000	0.9	0.2	$+0.34$	100	1001	0	0	-	-	0	100	16	2	0	-	-	0	100	21	2	0	-	-	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
1000	0.9	0.3	$+0.31$	100	1001	0	0	-	-	0	100	18	3	0	-	-	0	100	25	3	0	-	-	0
1000	0.9	0.4	$+0.28$	100	1001	0	0	-	-	0	100	21	5	0	-	-	0	100	30	4	0	-	-	0
1000	0.9	0.5	$+0.23$	100	1001	0	0	-	-	0	90	24	5	10	37	4	0	99	37	5	1	69	0	0
1000	0.9	0.6	$+0.19$	100	1001	0	0	-	-	0	55	25	4	45	31	3	0	96	48	7	4	74	8	0
1000	0.9	0.7	$+0.13$	100	1001	0	0	-	-	0	13	25	1	87	26	2	0	80	62	7	20	73	5	0
1000	0.9	0.8	$+0.07$	100	1001	0	0	-	-	0	1	27	0	99	23	2	0	22	82	6	78	86	7	0
1000	1.0	0.0	$+0.50$	100	1001	0	0	-	-	0	100	11	2	0	-	-	0	100	14	1	0	-	-	0
1000	1.0	0.1	$+0.47$	100	1001	0	0	-	-	0	100	12	2	0	-	-	0	100	16	2	0	-	-	0
1000	1.0	0.2	$+0.44$	100	1001	0	0	-	-	0	100	13	2	0	-	-	0	100	19	3	0	-	-	0
1000	1.0	0.3	$+0.41$	100	1001	0	0	-	-	0	100	15	2	0	-	-	0	100	22	3	0	-	-	0
1000	1.0	0.4	$+0.38$	100	1001	0	0	-	-	0	100	17	3	0	-	-	0	100	26	4	0	-	-	0
1000	1.0	0.5	$+0.33$	100	1001	0	0	-	-	0	96	20	4	4	32	1	0	100	31	4	0	-	-	0
1000	1.0	0.6	$+0.29$	100	1001	0	0	-	-	0	85	22	4	15	29	2	0	100	40	6	0	-	-	0
1000	1.0	0.7	$+0.23$	100	1001	0	0	-	-	0	44	22	2	56	26	2	0	94	52	7	6	100	38	0
1000	1.0	0.8	$+0.17$	100	1001	0	0	-	-	0	14	22	2	86	23	2	0	79	73	7	21	111	44	0
1000	1.0	0.9	$+0.09$	100	1001	0	0	-	-	0	1	20	0	99	22	2	0	65	121	12	35	135	28	0
10000	0.1	0.0	-0.40	0	-	-	100	1001	0	0	3	5679	1060	0	-	-	97	5	4900	1990	0	-	-	95
10000	0.2	0.0	-0.30	7	1001	0	93	1001	0	0	12	5279	2548	0	-	-	88	30	6179	2608	0	-	-	70
10000	0.2	0.1	-0.33	0	-	-	100	1001	0	0	0	-	-	100	294	104	0	0	-	-	100	152	21	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
10000	0.3	0.0	-0.20	10	1001	0	90	1001	0	0	50	5179	2584	0	-	-	50	60	4327	2334	0	-	-	40
10000	0.3	0.1	-0.23	0	-	-	100	1001	0	0	0	-	-	100	278	89	0	0	-	-	100	154	21	0
10000	0.3	0.2	-0.26	0	-	-	100	1001	0	0	0	-	-	100	124	26	0	0	-	-	100	117	13	0
10000	0.4	0.0	-0.10	16	1001	0	84	1001	0	0	99	2686	2164	0	-	-	1	98	2981	1717	0	-	-	2
10000	0.4	0.1	-0.13	2	1001	0	98	1001	0	0	1	283	0	99	286	104	0	1	168	0	99	155	25	0
10000	0.4	0.2	-0.16	0	-	-	100	1001	0	0	1	113	0	99	124	28	0	0	-	-	100	113	12	0
10000	0.4	0.3	-0.19	0	-	-	100	1001	0	0	0	-	-	100	81	15	0	0	-	-	100	96	9	0
10000	0.5	0.0	$+0.00$	45	1001	0	55	1001	0	0	100	349	206	0	-	-	0	100	528	296	0	-	-	0
10000	0.5	0.1	-0.03	7	1001	0	93	1001	0	0	19	172	63	81	257	80	0	3	136	19	97	154	22	0
10000	0.5	0.2	-0.06	2	1001	0	98	1001	0	0	2	112	5	98	124	29	0	4	105	5	96	116	13	0
10000	0.5	0.3	-0.09	1	1001	0	99	1001	0	0	0	-	-	100	80	13	0	0	-	-	100	94	9	0
10000	0.5	0.4	-0.12	1	1001	0	99	1001	0	0	0	-	-	100	56	9	0	0	-	-	100	84	7	0
10000	0.6	0.0	$+0.10$	100	1001	0	0	-	-	0	100	69	39	0	-	-	0	100	89	48	0	-	-	0
10000	0.6	0.1	$+0.07$	100	1001	0	0	-	-	0	84	87	49	16	223	80	0	74	82	30	26	152	21	0
10000	0.6	0.2	$+0.04$	100	1001	0	0	-	-	0	22	79	29	78	120	28	0	44	84	18	56	111	12	0
10000	0.6	0.3	$+0.01$	63	1001	0	37	1001	0	0	6	75	20	94	83	16	0	14	83	8	86	94	8	0
10000	0.6	0.4	-0.02	3	1001	0	97	1001	0	0	0	-	-	100	57	9	0	1	78	0	99	84	8	0
10000	0.6	0.5	-0.07	0	-	-	100	1001	0	0	0	-	-	100	43	6	0	0	-	-	100	79	6	0
10000	0.7	0.0	$+0.20$	100	1001	0	0	-	-	0	100	41	20	0	-	-	0	100	48	23	0	-	-	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
10000	0.7	0.1	$+0.17$	100	1001	0	0	-	-	0	98	49	28	2	182	26	0	96	53	24	4	120	9	0
10000	0.7	0.2	$+0.14$	100	1001	0	0	-	-	0	83	51	21	17	111	15	0	75	60	21	25	111	14	0
10000	0.7	0.3	$+0.11$	100	1001	0	0	-	-	0	45	51	13	55	81	15	0	48	65	15	52	95	9	0
10000	0.7	0.4	$+0.08$	99	1001	0	1	1001	0	0	16	47	9	84	56	8	0	25	67	9	75	84	7	0
10000	0.7	0.5	$+0.03$	94	1001	0	6	1001	0	0	2	42	4	98	43	6	0	10	72	8	90	79	6	0
10000	0.7	0.6	-0.01	4	1001	0	96	1001	0	0	0	-	-	100	35	3	0	1	79	0	99	77	5	0
10000	0.8	0.0	$+0.30$	100	1001	0	0	-	-	0	100	29	15	0	-	-	0	100	36	17	0	-	-	0
10000	0.8	0.1	$+0.27$	100	1001	0	0	-	-	0	98	35	17	2	114	25	0	99	44	21	1	140	0	0
10000	0.8	0.2	$+0.24$	100	1001	0	0	-	-	0	85	38	19	15	106	17	0	94	48	20	6	107	8	0
10000	0.8	0.3	$+0.21$	100	1001	0	0	-	-	0	67	38	14	33	75	14	0	80	50	17	20	95	8	0
10000	0.8	0.4	$+0.18$	100	1001	0	0	-	-	0	30	36	8	70	56	9	0	56	54	14	44	82	6	0
10000	0.8	0.5	$+0.13$	100	1001	0	0	-	-	0	13	35	3	87	43	6	0	36	61	11	64	79	4	0
10000	0.8	0.6	$+0.09$	100	1001	0	0	-	-	0	1	31	0	99	34	4	0	18	70	5	82	77	5	0
10000	0.8	0.7	$+0.03$	72	1001	0	28	1001	0	0	0	-	-	100	29	3	0	3	72	2	97	79	5	0
10000	0.9	0.0	$+0.40$	100	1001	0	0	-	-	0	100	24	11	0	-	-	0	100	31	14	0	-	-	0
10000	0.9	0.1	$+0.37$	100	1001	0	0	-	-	0	100	32	20	0	-	-	0	100	33	17	0	-	-	0
10000	0.9	0.2	$+0.34$	100	1001	0	0	-	-	0	97	32	16	3	112	5	0	98	41	16	2	106	3	0
10000	0.9	0.3	$+0.31$	100	1001	0	0	-	-	0	82	30	12	18	76	13	0	93	42	15	7	100	8	0
10000	0.9	0.4	$+0.28$	100	1001	0	0	-	-	0	51	32	10	49	55	10	0	73	47	12	27	84	7	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha-\alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
10000	0.9	0.5	$+0.23$	100	1001	0	0	-	-	0	26	30	7	74	44	6	0	52	51	14	48	80	6	0
10000	0.9	0.6	$+0.19$	100	1001	0	0	-	-	0	11	29	4	89	35	4	0	41	58	12	59	76	5	0
10000	0.9	0.7	$+0.13$	100	1001	0	0	-	-	0	2	22	0	98	29	3	0	12	72	7	88	78	5	0
10000	0.9	0.8	$+0.07$	81	1001	0	19	1001	0	0	0	-	-	100	25	2	0	6	92	10	94	90	5	0
10000	1.0	0.0	$+0.50$	100	1001	0	0	-	-	0	100	20	8	0	-	-	0	100	23	7	0	-	-	0
10000	1.0	0.1	$+0.47$	100	1001	0	0	-	-	0	100	25	13	0	-	-	0	100	28	11	0	-	-	0
10000	1.0	0.2	$+0.44$	100	1001	0	0	-	-	0	98	30	17	2	111	10	0	99	33	15	1	123	0	0
10000	1.0	0.3	$+0.41$	100	1001	0	0	-	-	0	85	29	15	15	69	16	0	93	38	15	7	94	9	0
10000	1.0	0.4	$+0.38$	100	1001	0	0	-	-	0	67	29	11	33	54	8	0	81	43	16	19	88	7	0
10000	1.0	0.5	$+0.33$	100	1001	0	0	-	-	0	40	26	7	60	42	5	0	68	47	12	32	78	5	0
10000	1.0	0.6	$+0.29$	100	1001	0	0	-	-	0	22	26	5	78	35	4	0	60	57	13	40	77	5	0
10000	1.0	0.7	$+0.23$	100	1001	0	0	-	-	0	7	24	3	93	28	2	0	39	63	10	61	79	6	0
10000	1.0	0.8	$+0.17$	97	1001	0	3	1001	0	0	1	24	0	99	25	2	0	21	82	9	79	91	10	0
10000	1.0	0.9	$+0.09$	69	1001	0	31	1001	0	0	0	-	-	100	24	2	0	6	129	12	94	137	19	0
100000	0.1	0.0	-0.40	0	-	-	100	1001	0	0	0	-	-	0	-	-	100	2	7587	2012	0	-	-	98
100000	0.2	0.0	-0.30	1	1001	0	99	1001	0	0	2	3746	2834	0	-	-	98	3	5561	3218	0	-	-	97
100000	0.2	0.1	-0.33	0	-	-	100	1001	0	0	0	-	-	100	288	113	0	0	-	-	100	153	23	0
100000	0.3	0.0	-0.20	1	1001	0	99	1001	0	0	3	4670	3812	0	-	-	97	5	4067	2707	0	-	-	95
100000	0.3	0.1	-0.23	0	-	-	100	1001	0	0	0	-	-	100	276	101	0	0	-	-	100	154	20	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\text{min}}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
100000	0.3	0.2	-0.26	0	-	-	100	1001	0	0	0	-	-	100	133	31	0	0	-	-	100	112	13	0
100000	0.4	0.0	-0.10	1	1001	0	99	1001	0	0	30	4240	2935	0	-	-	70	35	4830	2708	0	-	-	65
100000	0.4	0.1	-0.13	1	1001	0	99	1001	0	0	0	-	-	100	272	92	0	0	-	-	100	154	23	0
100000	0.4	0.2	-0.16	0	-	-	100	1001	0	0	0	-	-	100	127	30	0	0	-	-	100	111	13	0
100000	0.4	0.3	-0.19	0	-	-	100	1001	0	0	0	-	-	100	82	15	0	0	-	-	100	95	10	0
100000	0.5	0.0	$+0.00$	3	1001	0	97	1001	0	0	98	1909	1623	0	-	-	2	99	2178	1856	0	-	-	1
100000	0.5	0.1	-0.03	2	1001	0	98	1001	0	0	3	335	94	97	266	91	0	0	-	-	100	157	21	0
100000	0.5	0.2	-0.06	0	-	-	100	1001	0	0	0	-	-	100	129	30	0	0	-	-	100	114	14	0
100000	0.5	0.3	-0.09	0	-	-	100	1001	0	0	0	-	-	100	81	15	0	0	-	-	100	95	10	0
100000	0.5	0.4	-0.12	0	-	-	100	1001	0	0	0	-	-	100	59	11	0	0	-	-	100	86	7	0
100000	0.6	0.0	$+0.10$	87	1001	0	13	1001	0	0	100	461	417	0	-	-	0	100	485	440	0	-	-	0
100000	0.6	0.1	$+0.07$	70	1001	0	30	1001	0	0	22	154	108	78	247	72	0	15	90	37	85	154	20	0
100000	0.6	0.2	$+0.04$	54	1001	0	46	1001	0	0	4	66	20	96	129	29	0	5	93	22	95	112	13	0
100000	0.6	0.3	$+0.01$	9	1001	0	91	1001	0	0	2	84	15	98	82	16	0	1	98	0	99	94	10	0
100000	0.6	0.4	-0.02	0	-	-	100	1001	0	0	0	-	-	100	59	8	0	0	-	-	100	85	7	0
100000	0.6	0.5	-0.07	0	-	-	100	1001	0	0	1	44	0	99	45	5	0	0	-	-	100	80	6	0
100000	0.7	0.0	$+0.20$	100	1001	0	0	-	-	0	100	258	214	0	-	-	0	100	226	225	0	-	-	0
100000	0.7	0.1	$+0.17$	96	1001	0	4	1001	0	0	42	111	61	58	254	75	0	32	79	37	68	151	24	0
100000	0.7	0.2	$+0.14$	91	1001	0	9	1001	0	0	15	75	30	85	131	31	0	20	67	22	80	112	13	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\text{min}}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
100000	0.7	0.3	$+0.11$	73	1001	0	27	1001	0	0	10	54	9	90	84	16	0	9	63	15	91	95	8	0
100000	0.7	0.4	$+0.08$	61	1001	0	39	1001	0	0	0	-	-	100	59	10	0	3	66	10	97	85	7	0
100000	0.7	0.5	$+0.03$	26	1001	0	74	1001	0	0	0	-	-	100	45	6	0	3	74	4	97	79	5	0
100000	0.7	0.6	-0.01	1	1001	0	99	1001	0	0	0	-	-	100	35	4	0	0	-	-	100	78	5	0
100000	0.8	0.0	$+0.30$	100	1001	0	0	-	-	0	100	161	123	0	-	-	0	100	198	141	0	-	-	0
100000	0.8	0.1	$+0.27$	100	1001	0	0	-	-	0	49	97	52	51	237	75	0	41	81	40	59	157	21	0
100000	0.8	0.2	$+0.24$	95	1001	0	5	1001	0	0	22	67	36	78	126	30	0	21	63	20	79	113	10	0
100000	0.8	0.3	$+0.21$	94	1001	0	6	1001	0	0	7	44	12	93	86	17	0	16	59	16	84	95	9	0
100000	0.8	0.4	$+0.18$	84	1001	0	16	1001	0	0	5	34	10	95	59	9	0	14	64	14	86	86	7	0
100000	0.8	0.5	$+0.13$	69	1001	0	31	1001	0	0	0	-	-	100	44	6	0	5	56	14	95	80	5	0
100000	0.8	0.6	$+0.09$	43	1001	0	57	1001	0	0	0	-	-	100	35	4	0	1	83	0	99	77	5	0
100000	0.8	0.7	$+0.03$	14	1001	0	86	1001	0	0	0	-	-	100	29	2	0	0	-	-	100	79	5	0
100000	0.9	0.0	$+0.40$	100	1001	0	0	-	-	0	100	137	149	0	-	-	0	100	134	118	0	-	-	0
100000	0.9	0.1	$+0.37$	100	1001	0	0	-	-	0	58	89	52	42	209	65	0	50	67	34	50	153	23	0
100000	0.9	0.2	$+0.34$	100	1001	0	0	-	-	0	39	54	27	61	122	27	0	27	61	20	73	113	14	0
100000	0.9	0.3	$+0.31$	95	1001	0	5	1001	0	0	12	49	25	88	82	17	0	22	59	22	78	96	10	0
100000	0.9	0.4	$+0.28$	95	1001	0	5	1001	0	0	14	37	10	86	59	9	0	11	60	20	89	86	7	0
100000	0.9	0.5	$+0.23$	81	1001	0	19	1001	0	0	1	37	0	99	44	6	0	9	53	10	91	80	6	0
100000	0.9	0.6	$+0.19$	69	1001	0	31	1001	0	0	0	-	-	100	35	4	0	5	59	11	95	77	4	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\min}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail
100000	0.9	0.7	$+0.13$	39	1001	0	61	1001	0	0	0	-	-	100	29	3	0	2	74	4	98	79	5	0
100000	0.9	0.8	$+0.07$	18	1001	0	82	1001	0	0	0	-	-	100	25	2	0	0	-	-	100	91	6	0
100000	1.0	0.0	$+0.50$	100	1001	0	0	-	-	0	100	107	87	0	-	-	0	100	98	74	0	-	-	0
100000	1.0	0.1	$+0.47$	100	1001	0	0	-	-	0	59	75	49	41	229	78	0	64	55	32	36	153	22	0
100000	1.0	0.2	$+0.44$	100	1001	0	0	-	-	0	34	58	37	66	128	30	0	31	56	24	69	114	16	0
100000	1.0	0.3	$+0.41$	100	1001	0	0	-	-	0	19	34	11	81	84	15	0	18	51	18	82	94	8	0
100000	1.0	0.4	$+0.38$	95	1001	0	5	1001	0	0	9	33	10	91	58	9	0	15	44	12	85	85	7	0
100000	1.0	0.5	$+0.33$	93	1001	0	7	1001	0	0	6	30	9	94	45	6	0	11	53	16	89	78	6	0
100000	1.0	0.6	$+0.29$	87	1001	0	13	1001	0	0	4	22	4	96	35	4	0	6	60	12	94	78	5	0
100000	1.0	0.7	$+0.23$	51	1001	0	49	1001	0	0	2	26	4	98	29	3	0	1	42	0	99	79	5	0
100000	1.0	0.8	$+0.17$	34	1001	0	66	1001	0	0	0	-	-	100	25	2	0	2	78	6	98	89	6	0
100000	1.0	0.9	$+0.09$	12	1001	0	88	1001	0	0	0	-	-	100	24	2	0	0	-	-	100	133	11	0

Continued on next page

Table A.2 – *Continued from previous page*

				SSDS							IrSDS							CrSDS						
S	α	β	$\alpha - \alpha_{\text{min}}$	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail	c	μ	σ	c_β	μ_β	σ_β	fail

Table A.2: Robustness of reducing SDS. For each value of α and β in the interval $[0, 1]$ with a step of 0.1 where $\alpha > \beta$ and showing the amount which the optimal hypothesis score is greater than the minimum convergence criteria ($\alpha - \alpha_{\text{min}}$). Showing for each of standard SDS (SSDS), independent reducing SDS (IrSDS) and confirmation reducing SDS (CrSDS): c, the number of times out of 100 the algorithm successfully converged to the optimal hypothesis; μ and σ, the mean average and standard deviation of the number of iterations before halting in cases of successful convergence; c_β, the number of times out of 100 the algorithm converged to a sub-optimal hypothesis; μ_β and σ_β, the mean average and standard deviation of the number of iterations before halting in cases of convergence to a sub-optimal hypothesis; fail, the number of times the algorithm performed 10001 iterations without halting.