Small-World Effects in Lattice Stochastic Diffusion Search

K. De Meyer, J.M. Bishop, and S.J. Nasuto

Department of Cybernetics, University of Reading, Whiteknights,
PO Box 225, Reading, RG6 6AY, United Kingdom
`k.demeyer@rdg.ac.uk`

Abstract. Stochastic Diffusion Search is an efficient probabilistic best-fit search technique, capable of transformation invariant pattern matching. Although inherently parallel in operation it is difficult to implement efficiently in hardware as it requires full inter-agent connectivity. This paper describes a lattice implementation, which, while qualitatively retaining the properties of the original algorithm, restricts connectivity, enabling simpler implementation on parallel hardware. Diffusion times are examined for different network topologies, ranging from ordered lattices, over small-world networks to random graphs.

1 Introduction

Stochastic Diffusion Search (SDS), first introduced in [1], is a population-based best-fit pattern matching algorithm. It shares many similarities with e.g. Evolutionary Algorithms, Memetic Algorithms and Ant Algorithms [2]. During operation, simple computational units or *agents* collectively construct a solution by performing independent searches followed by diffusion through the population of potentially relevant information. Positive feedback promotes better solutions by allocating more agents for their investigation. Limited resources induce strong competition from which a large population of agents corresponding to the best-fit solution rapidly emerges.

SDS has been successfully applied to a variety of real-world problems: locating eyes in images of human faces [3]; lip tracking in video films [4]; self-localisation of an autonomous wheelchair [5]. Furthermore, a neural network model of SDS using Spiking Neurons has been proposed [6]. Emergent synchronisation across a large population of neurons in this network can be interpreted as a mechanism of *attentional amplification* [7]; the formation of dynamic clusters can be interpreted as a mode of *dynamic knowledge representation* [8].

The analysis of SDS includes the proven convergence to the globally optimal solution [9] and linear time complexity [10]. Recently it has been extended to the characterisation of its steady state resource allocation [11].

As search is applied to ever more complex problems with larger search spaces, even the most efficient algorithms begin to require some form of dedicated hardware to meet real-world performance demands. The standard formulation of

SDS is parallel in nature and thus implementing it on parallel hardware seems straightforward. However, the requirement for efficient communication links between all agents means that it is difficult to implement efficiently, both on dedicated hardware (e.g. FPGA's) or general purpose parallel computers.

This paper describes the effects of restricting communication between agents. In particular, the effects of the number of connections and the topology of the underlying connection graph on search performance are investigated empirically. It will be shown that, even for a modest number of connections, the performance of randomly connected networks of agents is close to the performance of standard SDS, and much better than performance of ordered lattices with the same average number of connections. However, small-world networks [12], based on regular lattices with a few long-range connections, perform almost as good as random networks. Two important conclusions can be drawn from the results:

1. Inter-agent communication in SDS can be significantly restricted without decreasing the performance of the algorithm too much, given that either a random or small-world network topology is used. However, the limited number of long-range connections in small-world networks facilitates the layout of the connections, making them the preferred network topology for hardware implementation.
2. Independent from the actual search process of SDS, the paper seems to confirm results in several epidemiological models using the small-world network topology, e.g. [13, 14]: namely that information or disease spreads much easier on small-world networks and random graphs than on ordered lattices.

2 Stochastic Diffusion Search

SDS utilises a population of *agents* to process information from the *search space* in order to find the best fit to a specified target pattern, the *model*. Both the search space and model are composed of *micro-features* from a pre-defined set. For instance, in a string matching problem, both the search space and model are composed of a one-dimensional list of characters.

In operation each agent maintains a hypothesis about the location and possible transformations (the *mapping*) of the model in the search space. It evaluates this hypothesis by testing how a randomly selected micro-feature of the model, when mapped into the search space, compares to the corresponding micro-feature of the search space. This part of the algorithm is called the *testing phase*. Based on the outcome of this test, agents are divided into two modes of operation: *active* and *inactive*. An active agent has successfully located a micro-feature from the model in the search space; an inactive agent has not.

During the *diffusion phase* the information about potential solutions may spread through the entire population. This is because each inactive agent chooses at random another agent for communication. If the selected agent is active, the selecting agent copies its hypothesis: *diffusion* of information. Conversely, if the selected agent is also inactive, then there is no information flow between agents; instead, the selecting agent adopts a new random hypothesis.

By iterating through test and diffusion phases agents will stochastically explore the whole search space. However, since tests will succeed more often in regions having a large overlap with the model than in regions with irrelevant information, an individual agent will spend more time examining 'good' regions, at the same time attracting other agents, which in turn attract even more agents. Potential matches to the model are thus identified by concentrations of a substantial population of agents.

Two important performance criteria for SDS are *convergence time* and *steady-state resource allocation*. Convergence time can in general be defined as the number of iterations until a stable population of active agents is formed and is very clearly defined when a single, perfect match of the model is present in the search space: it is then simply the number of iterations until all agents become active. Resource allocation is a measure for robustness in the case of imperfect matches and presence of noise: it is defined as the average number of active agents during steady-state behaviour, and is dependent on the quality of the match.

Examples of search behaviour, resource allocation and a more detailed description of the algorithm can be found in [11, 15].

3 Lattice Stochastic Diffusion Search

SDS gains its power from the emergent behaviour of a population of communicating agents and as such is inherently a parallel algorithm - notionally each agent is independent and its behaviour can be computed by an independent processor. However, a fundamental difficulty in implementing standard SDS efficiently on either a parallel computer or dedicated hardware is its requirement that each agent is able to directly communicate with all others. An obvious alteration to the algorithm is thus to restrict agent communication to a smaller number of agents. In the resulting algorithm, Lattice Stochastic Diffusion Search (LSDS), agents are assigned to spatial locations (e.g. on a 2D square grid) and connections between agents are specified. During the diffusion phase, agents will only communicate with agents they are connected to. Regular, local connections lead to an ordered lattice; or connections can be specified at random, thus effectively constituting a random graph.

An important question is how the performance and robustness of LSDS compares to standard SDS. The performance of standard SDS has previously been extensively analysed using Markov chain theory [9–11]. However, in LSDS the probability distribution determining communication between agents defines a neighbourhood structure over the entire set of agents. Analysis of this kind of process as a Markov chain is extremely complex: the process is not characterised by a simple integer denoting the number of active agents, but by the exact topological location of both active and inactive agents. These types of Markov processes are also known as Markov random fields. Work on a mathematical model incorporating the effects of restricted connectivity is ongoing, but at present performance measures for LSDS are investigated through simulations.

	$k = 4$		$k = 8$		$k = 12$		$k = 24$		$k = N$
	random	lattice	random	lattice	random	lattice	random	lattice	
$N = 64$	15.5	15.7	11.5	13.4	10.9	11.8	10.3	10.4	10.2
$N = 256$	21.3	29.5	15.0	23.8	13.9	20.1	13.1	16.3	12.5
$N = 1024$	25.8	55.5	18.1	44.1	16.7	36.0	15.6	27.3	14.8
$N = 4096$	32.3	106.9	21.1	83.4	19.5	66.9	18.1	49.3	17.1

Table 1. T_d in iterations for 4 different populations sizes N. Results are reported for random graphs with a mean number of connections per agent k; and for regular 2-dimensional square lattices with k-nearest neighbours connections and periodic boundary conditions. The case where $k = N$ corresponds to standard SDS. All results are averaged over 1000 runs, and for random graphs over 10 different graphs each.

4 Experimental Results

4.1 Convergence Time

[16] introduced the terms 'time to hit' (T_h) and 'diffusion time' (T_d) in the analysis of convergence time (T_c) of standard SDS. T_h is the number of iterations before at least one agent of the entire population 'guesses' the correct mapping and becomes active. T_d is the time it takes for this mapping to spread across the population of agents. It is clear that T_h is independent of the connectivity of the population and only depends on search space size M and number of agents N. T_d, on the other hand, is very much dependent on the connectivity within the population and on population size N, but independent of M. To focus attention on the effect of connectivity, experimental results for T_d are reported. It could be argued that for complex, high-dimensional problems $T_h \gg T_d$, and thus that the effect of T_d on T_c can be neglected with respect to T_h. However, T_d should not just be regarded as a measure for rate of convergence, but more as a measure for 'ease of information spread'. As such, it is also indirectly a measure for robustness: experiments indicate that the more freely information spreads through the network, the more robust the algorithm is in the case of imperfect matches or noise [15].

T_d is studied by initialising one randomly chosen agent with the correct mapping and recording the number of iterations until this mapping has spread to all other agents. Results are reported in Table 1. T_d for regular lattices does not scale very well with population size for a fixed number of connections k. For random graphs, T_d scales much better with population size and performance remains close to performance of standard SDS, even for a small number of connections and large population sizes.

4.2 Small-Worlds: Between Order and Randomness

Regular lattices have poorer T_d than random graphs, but are easier implemented in hardware, since connections are local and thus shorter, and regular. However,

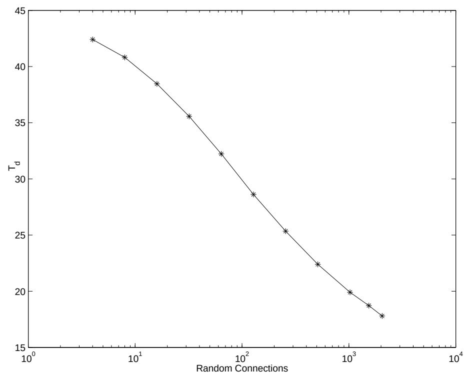

Fig. 1. T_d in iterations for $N = 1024$ and variable number of random extra connections x. Small-world networks are constructed starting from an ordered lattice with $k = 8$ and with x extra connections added at random. Note that for $x = 2048$, the last measurement, the mean connectivity in the network is $k = 12$. All results are averaged over 1000 runs, and over 10 different networks each.

diffusion of information in a population of searching agents shows an obvious correspondence with epidemiological models. Such models of disease or information spread have recently received much attention, due to the interest in the so called 'small-world effect'. It was shown in e.g. [12, 17] that only a limited amount of long-range connections is necessary to turn a lattice with k-nearest neighbour connections into a small-world network, in which spreading of disease or information behaves much more like spreading on random graphs. To test whether the same is true for LSDS, small-world networks were generated as described in [18]: a number of random links is added to an ordered lattice, and no connections are removed. T_d is recorded for various numbers of random connections; the results can be seen in Fig. 1. Randomly adding connections decreases T_d more or less exponential for a wide interval of parameter x, leading to an almost linear curve in the semilog plot. The benefits of adding relatively few long range connections seem obvious: a small-world network with only 256 extra connections (mean connectivity $k = 8.5$) outperforms a regular lattice with $k = 24$; a small-world network with 512 extra connections (thus $k = 9$) diffuses information twice as fast as the underlying regular lattice with $k = 8$, and is only 1.5 times slower in diffusing than fully connected SDS. Note that, even when adding much more connections, T_d will never become less than the value for standard SDS, in this case 14.8 (see Table 1).

5 Conclusions

The effect of mean number of connections and connection topology on diffusion time T_d was investigated empirically. T_d is an important performance parameter, not just because of its effect on T_c, but more importantly because it is also an indicator for resource allocation stability [15].

The good performance of 'small-world' LSDS has wider implications than just implementation in hardware. It has been suggested (e.g. in [12]) that biological neural structures can show small-world connectivity. The neural network architecture implementing standard SDS [6] uses biologically inspired neurons operating as filters on the information encoded in the temporal structure of the spike trains. Relaxing the requirements of full connectivity in these networks leads to a more plausible architecture, while still allowing for self-synchronisation across a large population of neurons [7] to occur.

References

1. Bishop, J.M.: Stochastic Searching Networks. Proc. 1st IEE Conf. ANNs, London (1989) 329–331
2. Corne, D., Dorigo, M., Glover, F.: New Ideas in Optimisation. McGraw-Hill (1999)
3. Bishop, J.M., Torr, P.: The Stochastic Search Network. In Lingard, R., Myers, D.J., Nightingale, C.: Neural Networks for Images, Speech and Natural Language. Chapman & Hall, New York (1992) 370–387
4. Grech-Cini, E.: Locating Facial Features. PhD Thesis, University of Reading (1995)
5. Beattie, P.D., Bishop, J.M.: Self-Localisation in the SENARIO Autonomous Wheelchair. Journal of Intellingent and Robotic Systems **22** (1998) 255–267
6. Nasuto, S.J., Dautenhahn, K., Bishop, J.M.: Communication as an Emergent Methaphor for Neuronal Operation. Lect. Notes Art. Int. **1562** (1999) 365–380
7. De Meyer, K., Bishop, J.M., Nasuto S.J.: Attention through Self-Synchronisation in the Spiking Neuron Stochastic Diffusion Network. Consc. and Cogn. **9(2)** (2000)
8. Bishop, J.M., Nasuto, S.J., De Meyer, K.: Dynamic Knowledge Representation in Connectionist Systems. ICANN2002, Madrid, Spain (2002)
9. Nasuto, S.J., Bishop, J.M.: Convergence Analysis of Stochastic Diffusion Search. Parallel Algorithms and Applications **14:2** (1999) 89–107
10. Nasuto, S.J., Bishop, J.M., Lauria, S.: Time Complexity of Stochastic Diffusion Search. Neural Computation (NC'98), Vienna, Austria (1998)
11. Nasuto, S.J., Bishop, J.M.: Steady State Resource Allocation Analysis of the Stochastic Diffusion Search. Submitted (2002) `cs.AI/0202007`
12. Watts, D.J., Strogatz, S.H.: Collective Dynamics of 'Small-World' Networks. Nature **393** (1998) 440–442
13. Zanette, D. H.: Critical Behavior of Propagation on Small-World Networks. Physical Review E **64:5** (2001) 901–905
14. Kuperman, M., Abramson, G.: Small-World Effect in an Epidemiological Model. Physical Review Letters **86:13** (2001) 2909–2912
15. De Meyer, K.: Explorations in Stochastic Diffusion Search. Technical Report KDM/JMB/2000-1, University of Reading (2000)
16. Bishop, J.M.: Anarchic Techniques for Pattern Classification, Chapter 5. PhD Thesis, University of Reading (1989)
17. Moukarzel, C. F.: Spreading and Shortest Paths in Systems with Sparse Long-Range Connections. Physical Review E **60:6** (1999) R6263–R6266
18. Newman, M.E.J., Watts, D.J.: Scaling and Percolation in the Small-World Network Model. Physical Review E **60:6** (1999) 7332–7342