Autonomic and Cognitive Possibilities for Information or Neural-Like Systems using Dynamic Links

KIERAN GREER[1], MATTHIAS BAUMGARTEN[1], MAURICE MULVENNA[1], KEVIN CURRAN[2] AND CHRIS NUGENT[1]

School of Computing and Mathematics and Computer Science Research Institute[1],
School of Computing and Intelligent Systems and Computer Science Research Institute[2],

University of Ulster, Shore Road, Newtownabbey BT37 0QB,

NORTHERN IRELAND, UK.

krc.greer;m.baumgarten;md.mulvenna;kj.curran;cd.nugent@ulster.ac.uk

Abstract : This paper will describe a vision for a new kind of information system that can dynamically self-organise and reason over its contents. The architecture is lightweight and would be suitable for an Internet-based environment, or alternatively, a mobile or pervasive sensorised environment. The lightweight architecture will rely on dynamic links, created through the query process, to self-organise. Results will show that the linking mechanism is reliable and can be combined with knowledge-based approaches, such as semantics, to provide extra functionality over what similar systems can currently provide. An overall architecture that could be called autonomic and cognitive will be suggested. This architecture could be used locally as part of a neural-like brain, or globally in a distributed information system. Autonomic features covered will be self-organisation and self-supervision, while different levels of reasoning will allow for an overall cognitive model that may even be able to think for itself.

Key-Words: Autonomic, Reasoning, Cognitive, Dynamic, Link, Query.

1 Introduction

This paper will describe recent work based on attempts to introduce lightweight and flexible reasoning mechanisms into a distributed information system. Most of the work has already been published, but will be brought together in this paper. The aim of the current work is to build a lightweight, distributed, service-oriented and autonomic system that has cognitive capabilities. The cognitive capabilities will be realised through some level of reasoning, while the lightweight architecture requires that the mechanism for this reasoning is also lightweight.

The context of this work is to have a distributed information system, such as the Internet, but which can also accommodate a mobile or sensorised environment. As written in [13], the information system could be understood to be a network that organises information sources through the use of ontologies and intelligent links, to provide some sort of meaning to the associations. By meaningful it is meant that the associations will be understood by the user of the system. The network can also be loaded with any number of services that can intelligently process or reason over the stored information. This sounds a lot like the Web 3.0, although this work would extend the definition to also accommodate the pervasive sensorised environment. The sensors would act as data sources, providing information to a more intelligent system that could combine such information with existing heterogeneous knowledge sources. Thus dynamic and real-time aspects of an environment can also be used to answer queries.

Autonomic capabilities are realised through self-organisation and self-supervision. The self-organisation is performed through the intelligent links that automatically link sources associated with each other. This helps to optimise the network for any future querying or search activities. The

performance of the linking mechanism can be monitored and it will be shown that the performance can drop as well as improve. Thus supervision is required to monitor this and change the linking mechanism when performance starts to drop again.

The linking structure is generated through the query process. Queries will ask for information that associates certain knowledge sources. If similar types of query are consistently executed, then certain knowledge sources will be consistently associated together. If this can be recognised then these sources can be linked and then when one of the sources is used, the others can automatically be retrieved as well. If there are many potential sources that can answer the query, then this will help to optimise the query process. As these links are built from the results of executed queries, they also reflect the use of the system, or what the users want to retrieve. Thus they also reflect the knowledge of the users of the system and so could be used for more knowledge-intensive queries. In fact, they can be used to perform some level of reasoning over the stored information.

The rest of the paper is structured as follows: Section 2 will describe the network structure and lightweight architecture. The self-organisation is closely associated with the query process and so these will both be described in section 3. Section 4 will describe some related work in the areas of AI and network systems. Section 5 will present some test results and section 6 will describe the self-supervision possibilities. The reasoning possibilities will be described in section 7, while section 8 will describe the overall vision for a new cognitive model. Finally, section 9 will discuss some conclusions on the work.

sub-class relation to associate semantically related concepts, where metadata at each node can describe its sub-tree. Thus existing knowledge can provide very efficient navigation by itself. Also required however are temporary views of the network that reflect the current use of the system. If the use of the system changes, then so do these views. These views can be built from dynamically created links between the sources related through the query process. These links can complement the permanent semantic organisation and can be used to further optimise the search process. They will also help to distinguish between several sources of the same type that might be used to answer a query. Particularly in a pervasive sensorised environment, you could imagine several sensors all returning similar information to a system. If just the relevant sensor, based on its value or even location, could be identified, then this could help to optimise the query process. The organisation is illustrated in Fig. 1.

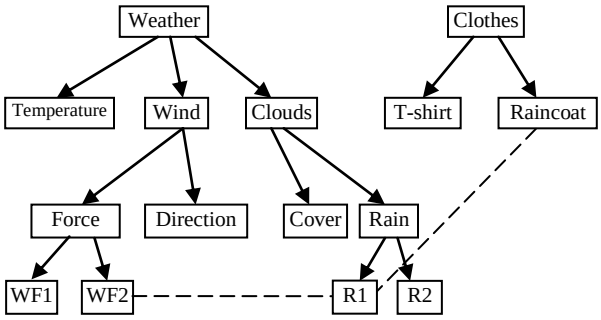


Fig. 1. Example of a ‘knowledge network’.

2 Network Structure

The intention of this work is to provide a lightweight architecture on which to build service-based networks (SOAs). As such, heavy knowledge-based approaches are not preferred, but knowledge provided from structures such as ontologies are essential. The difference is to try and integrate or distribute this knowledge throughout the whole network in a lightweight way. The network structure is to be hierarchical, constructed from semantics. This hierarchical structure is permanent and can be built from existing ontology information. It uses the

Fig. 1 gives an example of one of these information networks, or ‘knowledge networks’ as they have been called in recent work. This network describes the topic of ‘weather’ and has been semantically constructed from an ontology on that subject. Thus ‘wind’ is a sub-concept of weather, and ‘wind force’ and ‘wind direction’ are sub-concepts of wind, etc. A user can query this network and use a ‘Select-From-Where’ statement to ask for specific values for specific concepts. For example, the user might ask what clothes he should wear when it is windy and rainy. This could link wind force nodes, rain nodes

and clothes nodes (raincoat) together. Through the permanent hierarchy, the main navigation is accounted for, but the network contains two wind force nodes and two rain nodes. Depending on the values that these return, only one wind force node (WF2) and one rain node (R1) is typically associated with the raincoat node to answer the query. Thus through consistently executing this query and associating the same nodes, links are built up between the related sources, as shown by the dashed lines. Once these links have been built up, if the same query is executed again, these sources will know about each other and can be returned directly to answer the query. Thus nodes WF1 and R2 do not need to be looked at and so some degree of optimisation has been achieved.

These links can be created by the network feeding the results of previous queries back through itself. The nodes used to answer the query can represent the links through weighted references to each other. Through consistent associations the weight will pass a threshold value, when it will be considered to be a reliable link. However, if the association is then lost, the weight value can drop below the threshold again, when the link will eventually be removed. Thus the linking process is dynamic and will reflect the current use of the system. Note that the relation between raincoat and the weather concepts might not be known beforehand, or even be semantically sensible, and so this linking mechanism can add something extra to any semantically organised information.

3 The Query Process and Self-Organisation

A query process is required to allow a user to retrieve information from the network. Tests have tried a standard 'Select-From-Where' statement, although, there are now other types of query construct. This allows a user to select certain values from certain source types where certain conditions are true. The querying process being adopted here is a two stage process. The first stage searches for potential sources that can answer the query and the second stage then queries the sources to retrieve their values. The first stage can also be used as a search engine that is guided by the hierarchy and links. A two stage process is really required because the 'Where' clause comparisons need to compare different source types. This means that the locations of all relevant source types need to be retrieved first, because the results of each comparison cannot be known beforehand. Although, if comparing with a specific value, then this can be done at the source. In this case, only the sources that satisfy the value-based comparison are returned.

The network needs to be fed information that it can use to create dynamic links for optimisation and it is proposed that this can be done by using the results of the querying process. This is like a third stage to the query process. The sources that were used to answer each query can be informed and they can update their related link values. Each source can store a structure (link table) that records the sources related to it through the querying process. If a source is found to be used in the same query that the current source was used in, then its reference can be added to a link table in the current source. This structure can monitor related sources at different levels by assigning weights to them and allows it to recognise when a new source reference should be included or an old source reference removed. In the context of source linking, a weight is simply a numerical amount associated with a source. It can be incremented or decremented and compared to threshold values. If the weight value becomes greater than a threshold value, then the source reference can move up a level in a linking structure. The linking structure is described in more detail in section 5.

The self-organisation is based on the bio-inspired process of Stigmergy, for example, the Ant Colony Optimisation algorithm [6]. It could, however, also be called simply Hebbian reinforcement [15]. The aim of the self-organisation is to link related information sources in a meaningful way, so as to introduce more knowledge into the system. These links can then be used to guide and thus optimise any search process. Stigmergy has been widely used for optimising routes through Mobile and Ad-Hoc networks (MANETs), which is the main use for the links in the system proposed here as well. However, the mechanism for creating the links in the system proposed here is slightly different. The process is purely experience-based and there are no knowledge-based algorithms controlling the linking mechanism. Also, rather than clustering semantically similar nodes, the query process will cluster nodes based on

their use only, which could be semantically very different nodes.

The query process will need to be able to ask different kinds of question. A cognitive model that will be suggested in section 8 will describe three levels of intelligence, related to different types of query. The bottom level allows for direct information retrieval. That means retrieving information that is directly available from the information sources. The second level allows for aggregation through links. This allows for some reasoning that could return preferred or 'best' answers to questions as determined by the users of the network. The third level then provides possibilities for higher level concepts and reasoning. For example, if we consider the network of Fig. 1, this could allow for different types of query as shown below:

- **Direct Information Retrieval:** Is there a local shop that sells jumpers?
- **Low Level Reasoning:** What is the best type of coat to wear in the Summer?
- **Higher Level Reasoning:** What should I wear with a red shirt?

4 Related Work

4.1 Complex Adaptive Systems

One term that covers all systems of the type being described in this paper is 'Complex Adaptive Systems' [16] [29]. The term encompasses more than one theoretical framework and is highly interdisciplinary, seeking the answers to some fundamental questions about living, adaptable and changeable systems. A Complex Adaptive System is a collection of self-similar agents interacting with each other. They are complex in that they are diverse and made up of multiple interconnected elements and adaptive in that they have the capacity to change and learn from experience. Al-Obasiat and Braun [1] describe that Complex Adaptive Systems share common properties and methodologies which assist them to survive, evolve and adapt to changes in the dynamic environment. Based on the work of Holland, some of these can be summarised as:

1. An agent is the main work component in the system.
2. The agents have simple, primitive constructs, communication language or protocols.
3. The agents operate individually according to stimuli-action rule instead of event-condition-action rule.
4. The environment consists of a diverse set of randomly distributed devices and services interacting together through communication protocols.
5. An existing adaptation mechanism whereby agents can vary their responses to changes.

The behaviour of these systems may not be known beforehand. With complex systems, the interactions between individual components in the system give rise to the emergent behaviour. Emergence is the process of complex pattern formation from simpler rules. An emergent behaviour arises at the global or system level and cannot be predicted or deduced from observing the behaviour of the individual components in the lower level entities. These sorts of emergent behaviours can be realised through bio-inspired techniques such as stigmergy, or swarm optimisation processes.

4.2 Stigmergic Self-Oganisation

This work proposes to base the organisation on the bio-inspired mechanism of 'Stigmergy' [11]. Stigmergy is inspired by the actions of colonies of simple biological creatures such as ants [6]. With this mechanism, the system learns relations through input from its environment (stimulus), without any internal knowledge of what the stimulus represents. Such bio-inspired systems can use relatively simple swarm algorithms to organise themselves, where collectively the components of the system can exhibit some form of intelligence. Related work that uses stigmergy for self-organisation has already been written about in [12] and this section repeats that description. [21] discusses the main mechanisms used in this paper for linking (stigmergy, self-organisation, reinforcement). While this paper has described the process as being stigmergic, others might argue that it is more like the Hebbian learning rule, which states that concepts that are activated simultaneously become more strongly associated. This method has been used in [15],

although their method of clustering seems to require manual user interaction rather than an automatic query process.

Stigmergy has been used widely to self-organise in MANETs (Mobile Ad-Hoc Networks). These networks are highly dynamic and need a flexible and robust mechanism that can adapt and allow them to self-organise. As these systems can be on a massive scale, some centralised controlling mechanism may not be practical. The papers [3] and [5] discuss this problem and while the self-organisation may not lead directly from the querying process, it would involve information sharing. The architecture described in [27] is close to the network architecture suggested in this paper. They also have a hierarchical structure based on super-peers aggregating other peers. However, the organisation is slightly different as the super-peers do all of the autonomic clustering. The source clustering tested in this paper is not hierarchical, but there is no reason that aggregating nodes could not also be clustered using the links. The papers [7] and [25] are also relevant, where they use dynamic linking to re-route queries as part of search optimisation.

Examples of linking based purely on the query experiences include [18] and [20]. In [18] they try to cluster nodes in a peer-to-peer network based on query workloads. They measure how similar a node's content is to a type of query, which will mean that it is more likely to return an answer to that type of query. They then try to cluster nodes with similar workloads together in workload-aware overlay networks. They describe that the mechanism for calculating the workload value is still an open issue and could be based on a node storing statistics on the queries that pass through it. Another system that tries to associate nodes based on the query experience is called NeuroGrid [17], while [26] describes a service discovery protocol called 'Superstring'. Superstring is used to discover services in an extremely dynamic environment but with a stable central core of nodes. This is exactly the environment envisioned for the system described in this paper. The lightweight framework allows for a dynamic MANET environment, for example, but the stable central core allows for stigmergic principles to work. They also build up a partial hierarchy based on the semantics of the concepts being searched. These examples however may be more along the lines of a search engine than the query evaluations tested in this paper.

4.3 Cognitive AI

The more intelligent aspects of this system have also been looked at previously. The system is to be autonomous [22] and should be able to reason over its contents. Autonomic components are essentially self-managing. They can self-configure, self-heal, self-optimise and self-protect. This work is interested primarily in the autonomous self-organisation of knowledge, but also in self-supervision. A cognitive model that will be ultimately suggested is neural in nature, but will use a symbolic representation for each node. This has important advantages that was recognised early on in AI research. The 'physical symbol system' hypothesis was first attributed to Newell and Simon [24]. Symbols are useful because they can be understood and so it is possible to tell what is happening in the system. An alternative approach to using symbols is PDP (Parallel Distributed Programming). This is more like the human brain and the classic example is a neural network. There are many references on neural networks, for example [2]. A neural network is typically made up of layers of nodes. The neural network however is more of a black box that cannot be as easily analysed. A theory of how symbols reduce to patterns in a network would be an extraordinary contribution to AI.

There is also considerable interest in multi-agent or co-operative process problem solving, including the integration of neural network and symbolic based approaches. This will be a major focus of the new cognitive model. One other area that this research would fall into would be neurocognitive systems, as described in [8]. Many computer science fields cover the problem of learning, but [8] adopts a neurocognitive perspective. This approach tries to draw inspiration from how the current brain copes with difficult learning problems. The paper describes that 'creating artificial systems that could learn in the same way as people may be considered as the most important challenge facing science.' Thus the neural and cognitive problems are addressed. Another cognitive model that has been used to train robots is described in [23]. They use an approach called 'evolutionary robotics', where the robot's behaviour can emerge through evolutionary self-organisation. As with CAS, the behaviour emerges through the interactions among the constituent parts.

Fu [10] is an important paper with respect to this work and notes many of the problems that this system is trying to solve, but does not give any concrete solution. One key method in this architecture is to aggregate values from several nodes to allow for reasoning from experience. This sort of aggregation has already been considered by Stevenson et al. [28] although the approach suggested in this paper is different with respect to how the aggregation is realised and possibly how it is used. If the new architecture being suggested is successful, it should be more flexible and provide more functionality with respect to reasoning, than the other models that have been outlined.

5 Testing the Querying Performance

This section will summarise some results that tested the performance of the linking mechanism. A test environment has been written to test the effectiveness of the linking mechanism. This environment can create random networks and execute random queries on those networks. A detailed description of the test environment, process and results can be found in [12] and so only a summary of the results will be described in this section. More details on the linking mechanism will also be given.

5.1 Linking Methodology

The tests have used a linking structure with three levels. These levels are separate structures that represent possible sources, monitored sources and linked sources. The possible sources structure is at the bottom and records new sources that may possibly become links. They need to be associated several more times before they can be called links. The monitor structure is an intermediary structure that stores more advanced possibilities. Having two levels here (possible and monitor) may not be necessary, but it can be helpful for resource management or may allow for initial communication when a source is shown to be related. For example, specific sources can reinforce links when they reach the middle level, even if the whole query is not answered. The linked level is the top level that then stores the references to sources that are actual links. It is these sources that are returned as possible answers when the appropriate query is executed. Each structure can be assigned a limited amount of memory, or number of allowed references, ensuring that it stays lightweight. When this allocation is reached, to add a new reference it must first remove an existing one.

Source references are stored with weight values that can be either incremented or decremented. The weight values determine in which level the source references are stored. The weight values must reach a certain threshold value before they can be moved up a level. For example, for the simplest case, say we have a weight increment value of 0.1 and a threshold for the next level of 0.5. If a particular source is associated with the current source 6 times in a row, then its weight reaches the value 0.6, which is greater than the threshold and so its reference can be moved up to the next level. If the same query type is run again however and the source is not used, then its value can be decremented, when it will subsequently be moved down a level if it then falls below the threshold.

Additional features are also possible. For example, sources can borrow memory from each other. This means that the total amount of memory stays the same but it is better distributed. More heavily used sources can be allocated a larger amount of memory and thus can more effectively optimise. Learning algorithms might also be included to try and autonomously learn certain parameters, such as the best weight increment or decrement values.

5.2 Linking Structure

The linking structure itself is meant to record related sources for similar queries. However, it is not a caching mechanism where it stores the whole query with the sources used. It is slightly more lightweight and flexible, where it matches parts of new queries to the structure. Thus individual links can be used as part of different query answers.

The structure can allow for any level of nesting that may be suitable to the current problem. The linking structure is a reference-based structure that stores the addresses of nodes rather than the nodes themselves. The reference would typically represent the query part that the reference relates to and also the address

of the linked node. For example, if we ask if a wind force node has the same location as a rain node and consistently retrieve the nodes WF1 and R2, the linking reference in WF1 to R2 might look like:

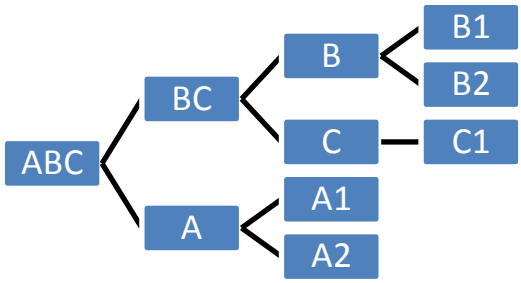
http://123.4.5:8888;location.rain.location.equals.R2

Where the http address is the location of the server that the R2 node runs on. The references can be used in at least two different ways. One way is globally in the network itself. It can be used to link sources that typically answer the same type of query. Then when one of the sources is queried, it can return the sources it is linked to and these can be looked at instead of needing to look at all potential sources. The links stored globally thus optimise the whole network and can be shared between all users. The other possible use is as a local view. A particular application may want to store locally references to the nodes that it typically visits for the queries that it typically answers. These may be a subset of the whole network. A description of the nodes visited by a particular application can be stored locally and monitored in the same way as the source links. Then when a new query is executed, the local view can be searched first and if it contains any relevant information this can be used instead of having to search the whole network.

5.3 Evaluation Metrics

Random networks and queries can be generated to user-specified configurations. The statistical properties of node count and QoS can then be measured for the queries executed on the networks. Queries can be created that use only the equivalence comparison (equal to), or all comparison operators (greater then, greater than or equal to, less than, less than or equal to, equal to, not equal to). The performance of the linking is then measured as the amount of reduction in node count and also as the quality of answer returned. The percentage of reduction in number of nodes visited can be measured by comparing a node count from a full search only with a node count from a linked search. Both searches are guided by the hierarchy. A full search would then look at all relevant nodes, that is, all nodes of the correct type. A linked search however would use only linked nodes if they existed, which would prune certain nodes from the search. A typical random network and query is shown in Fig. 2.

The test queries requested values from sources that were of the integer type. The evaluation function then tried to maximise the sum total for all source values requested to produce the best quality of answer. As the network is service oriented, the term Quality of Service (QoS) is used to mean the same as quality of answer. Using numerical information means that an evaluation function can reliably measure the QoS that is returned. The evaluation function tries to maximise the sum total for all source values in the answer. This was simply used as the metric to distinguish what the best answer would be and would thus specify the nodes that the linking mechanism should try to link. The full search, with access to all nodes, will always return the largest sum. If the correct nodes are linked, then a search that uses links will return a sum total similar to what a full search will return. If the linking mechanism does not link the correct nodes, then the linked search will return a smaller total. So this difference will give an indication of how accurate the linking mechanism is.



Select A.Value1 From A, B Where A.Value2 LT B.Value3

Fig. 2. Random network and query.

So for the query described in Fig. 2, the query engine would return the ‘A’ source that had the largest ‘Value1’ value and also satisfied the ‘Where’ clause comparison. While these tests used numerical values, the equivalence only queries would also be useful for text or concept matching. Comparing two concepts can apply to numbers or text equally. For example,

10 different integer values can be compared in the same way as 10 different textual words.

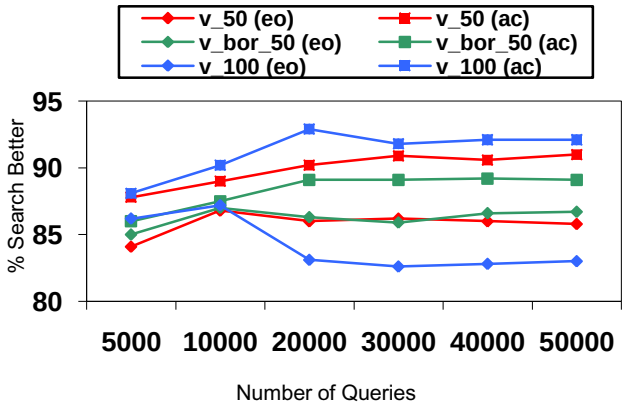
5.4 Test Results

This section will give the results of one set of tests based on one particular linking configuration. The mechanism that was used to dynamically link sources could be constructed from different features. These features included limiting the allowed amount of memory that each source could use in its linking structure. This would mean that the linking structure could remain lightweight. The linking mechanism could also allow sources of the same type to borrow memory from each other and these were the two features included in these tests.

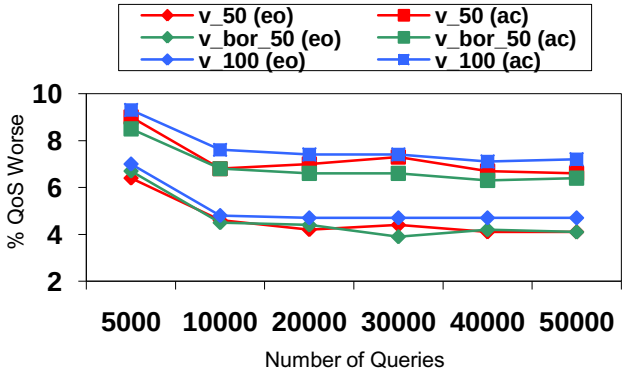
For the linking to be effective it is assumed that consistent types of query needed to be executed. If every query is different and associates different nodes, then it is assumed that the linking mechanism would not work. If the query is always the same however, then it is assumed that the linking mechanism would work. So the tests tried to determine what level of variability in the type of query being asked could the linking mechanism cope with.

The queries were tested on a random network with 10 source types and 30 instances of each type. Thus there were 300 source instances in total in a network with 315 nodes in total. Each instance had 5 value types, each with a random value in the range 1 to 10. For the test results being shown, the queries were skewed with a 90:10 split. This meant that 90% of the time one of 3 source types or 2 value types would be selected in the query construction and 10% of the time one of the remaining 7 source types or 3 value types would be selected. Tests however showed that a split as low as 70:30 could also produce good results. Each test run evaluated 50000 queries and results were averaged over at least 3 test runs.

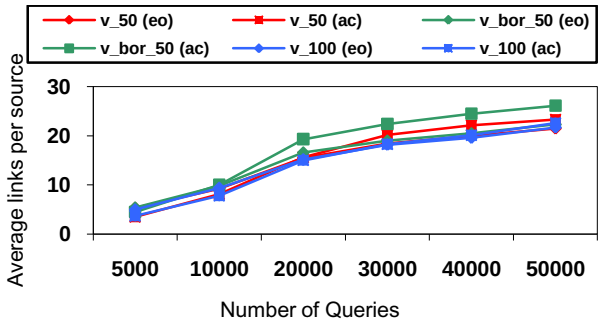
The results are shown in Graphs 1, 2 and 3. These graphs show the amount of search reduction (Graph 1) and related loss in QoS (Graph 2) when using links to answer the queries compared to a full search. The average number of links per source is also shown (Graph 3). In these graphs, 'eo' stands for the queries that contained the equivalence comparison only, while 'ac' stands for queries that allowed all of the comparison operators. '50' or '100' indicates the maximum number of entries that were allowed at each level of the linking structure for each source. 'bor' indicates that the sources were allowed to borrow memory from each other. There was also a local view that allowed 100 entries at each link level. The view would be used to select particular source instances for the first source type selected in the first 'Where' clause comparison.



Graph 1: Percentage of reduction in the number of nodes searche. Link structures with a view that do (v_bor) or do not (v) borrow memory are included.



Graph 2: Percentage of reduction in quality of service. Link structures with a view that do (v_bor) or do not (v) borrow memory are included.



Graph 3: Average number of source links stored for each source. Link structures with a view that do (v_bor) or do not (view) borrow memory are included.

Graph 2 shows the amount of reduction in QoS when using links. That graph shows that a 5 – 10% reduction is QoS could be expected, but this is compared to the optimal answer. Graph 2 also shows that the equivalence only queries performed the best with regard to QoS, as would be expected due to the smaller variation in possible query types. Graph 1 shows that as much as 80 – 90% reduction in the search can be achieved. Tests showed that in general, a larger search would produce a better QoS. This would be expected as the larger search can look at more potential sources. However, if the links being added were key, then both factors could be improved. The results also suggest that allowing the sources to re-distribute their memory allocation (borrow option) could help with the optimisation process. A slightly better QoS is achieved, although the difference is not very large in these tests. Other features were also tried to see if query performance could be improved and the results are described in [12].

6 Self-Supervision Possibilities

This section will take the results just described and show how they indicate a need for the system to be supervised to maintain optimal performance. These results have been presented previously in [12]. Graph 1 also shows that with regard to search reduction, it is possible to have too many links. This can be seen when more links are added but the search node count starts to increase again. Overall performance can be measured as a balance between search reduction and QoS. If the search increases without a corresponding improvement in QoS, then there may be too many links. Graphs 1 and 2 suggest that there may be too many links from 2000 queries (eo) or 30000 queries (ac) onwards. Graph 3 shows that the link numbers are still increasing at this stage. There appear to be optimal configurations that then reduce as more queries are executed. The reason must be that extra links are being added that are not then helping with the search process. However, when there are only 50 allowed entries at the possible links level there is not the same reduction in performance. Thus changing the number of allowed entries suggests that the linking performance could be quite sensitive. One possibility might be that the extra entries have allowed more variety in the references that make the link level, or alternatively, having fewer allowed entries has concentrated the variety at the link level and thus reduced the effect of too many links.

So increasing numbers of queries might produce too many links, or contradicting links and so there is a need for monitoring, to determine when the upper limit on performance has been reached. This monitoring process has been described in some detail in [12] and [12], where [12] argues that the loss in performance could be related to concept drift, a fault in the system or simply a change in its use. A proposed solution is also described in that paper.

7 Reasoning Possibilities

The querying mechanism that has been described in the previous sections is for direct information retrieval. It simply retrieves information that is directly available at the source nodes. The links however also allow the possibility for different levels of reasoning over the stored information. This can be done through just the links themselves, or with the help of rules. This could be divided into experience or knowledge-based reasoning, as described next.

7.1 Experience-Based Reasoning

Experience-based reasoning can use the experience of previous queries to answer questions such as ‘what is the best value for one concept based on other concepts?’, or ‘is a certain concept (or action)

possible based on other concepts?' The query engine answers these types of query through simple mathematical operators such as aggregation or averaging. This sort of aggregation has been used in other dynamic systems as well, for example [28]. In the context of this work, the process is as follows:

A user might ask the system what the best value for one concept is based on other concepts. The query engine searches for answers to the query and retrieves different groups of related sources that can answer the query. For direct information retrieval, an evaluation function would be required to determine what the optimal answer from all of the different possibilities would be. An alternative to this would be to average all of the possible answers to produce some kind of 'best' answer. It is called the best answer because it takes into account all of the previous answers that were linked, or the users' preferred options. Thus the query answer is averaged over all of the previous user's requests that produced links. In particular, if some count is required, for example, the most popular shop visited, this averaging query would be particularly effective. Results have shown that the linking mechanism has a positive effect on the quality of answer retrieved for these 'best' queries. If the evaluation function tries to optimise the value for an answer, then this will naturally skew average values towards a larger total. This will thus produce a better answer quality compared to a full search that averages over all possible answers. So as well as node count being reduced, the answer quality would naturally be improved. The second type of query simply requires links to exist between the concepts that are asked for. If any links exists then the query can return true and if they do not then the query can return false.

7.2 Knowledge-Based Reasoning

It is also possible to use the links with existing rules to answer queries that could not be answered otherwise. The following problem in Fig. 3 has been used previously, but provides a suitable example. The network shows three concepts of Son, Mother and Brother, with related values. Stigmergic links have been created between the Son and Mother, and the Mother and Brother nodes. The user asks if John has an Uncle, or alternatively, if there is a set of linked nodes leading to a Son source with the value of John that has an Uncle.

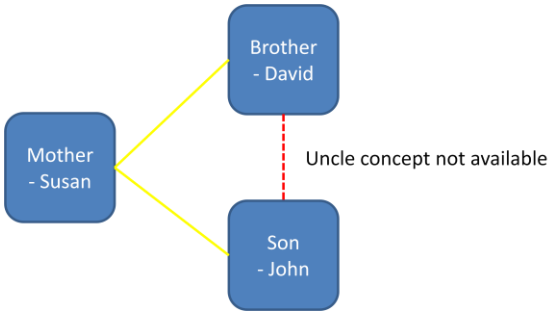


Fig. 3. The Uncle concept requires a rule to be answered.

Because the Uncle concept is missing from the network, a rule is required to answer this query and so existing semantic knowledge is required. The rule would specify that the Brother of a Mother is an Uncle. With this rule we can traverse the links from Son to Mother to Brother and thus conclude that John does in fact have an Uncle with the name of David. A query to ask this could look like the following [13]:

hasUncle Select Son.Details From Son Where Son.Name equals John

Because the example is so simple, this is a slightly contrived query. Essentially, the addition is an extra keyword that acts as a question. The select statement is evaluated and returns a 'Son' source that satisfies the conditions. The extra keyword 'hasUncle' then triggers the execution of a rule. The rule tries to traverse from the Son node to link all other nodes that would mean the hasUncle rule would be satisfied. As links exist, it can be determined that the hasUncle rule is true. However, the rule can only state the relation between the Son, Mother and Brother concepts. It cannot know what actual values of these concepts are linked, for example, that Susan is John's Mother. It is only through the use of the system that associates these values together and thus creates the links that allows this query to be answered. So the linking plays a key role in generating the knowledge required to answer this type of query as well. The

links act as the facts for the knowledge-base. They are distributed throughout the system and can dynamically change with changes in system use.

8 Overall Architecture

The previous work has shown how stigmergy can be used as part of an autonomous system, to efficiently self-optimise and also provide some level of reasoning. This section will extend the model further to provide higher levels of reasoning and suggest how a new cognitive model might be developed from it. While this section will focus on a local small-scale network, the principles would apply to a large distributed network as well. [10] is an important paper with respect to this area and notes many of the problems that this model would solve. It also considers knowledge discovery and self-organisation in neural networks and bridging the symbolic-distributed gap. A neural-like structure that can understand what its nodes represent is a key feature of the model.

A smaller local network could receive inputs from its environment and then form associations between them. The environment of the network can be defined simply as its inputs and outputs. This can be controlled by an external system, or alternatively, in an autonomous system, the network could read its environment through electronic signals, for example sensors [9] or image recognition [19]. While humans with real intelligence can be selective with what they associate, a computer would also need an 'understanding' of its environment to do the same thing. The network would need to associate the related nodes and use a reinforcement algorithm such as Hebbian or stigmergy to form links between them.

As the network receives inputs from its environment, it must be able to select the concepts to associate with each other. By associating certain concepts together it is creating higher level and more complex concepts. The network can define the higher level concept by using stigmergy or reinforcement to form links between the related lower level concepts to create a single path through all of them. Each link in this path is also assigned the same unique key. In this way the network can recognise that this set of concepts actually represents something. This design is not only flexible with regard to the concepts that can be learned, but also with regard to the number of nodes, types of nodes and general organisation of the nodes. If a new concept is introduced, then a new node can simply be created to represent it and added to the network. An alternative way to index higher level concepts would be to create a hierarchical network as discussed previously, where a node at a higher level would reference nodes at lower levels that make up the concept. With this architecture, you could even have two higher level concepts being combined to an even higher one, and so on. It might only be a matter of convenience whether you use links with unique keys or hierarchical networks. Labelling higher level nodes is then an issue and they may also require more memory for storing. But in some cases they may improve search time as they clearly define paths through the network to the source concepts.

8.1 Integrating Rules

This network structure however is still restricted to limited reasoning over the concepts and structure that it contains. In a knowledge-based system for example, there are sets of rules that allow the system to derive new knowledge. For example, say we have the set of relation concepts described previously in Fig. 3 and we know that Susan is John's mother and David is Susan's brother. These relations are shown again by the stigmergic links in Fig. 4.

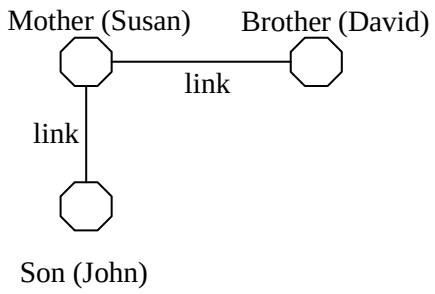


Fig. 4: A network with stigmergic links between the concepts Susan, John and David.

As already described, if a user asks if John has an uncle however, the network cannot answer this, because this information is missing. If we have the rule that 'the brother of a mother is an uncle', then

the system can derive that David is John's uncle. Through direct information retrieval, stigmergic links cannot answer this, but there are two ways in which a system can use the links to answer this type of query. One way to do it is to have a query-rewriting tool that re-writes the query [13] so that it can be executed on existing data, as described previously.

This query re-writing is a centralised approach but there is also a distributed approach. The alternative would be to code the rule into the network structure itself. This cannot be done through a stigmergic reaction and so a central algorithm is still required to code the uncle concept into the network, but after this the use is distributed. The concept of uncle does not exist, so the network might query a knowledge-base, which returns the rule that the brother of a mother is an uncle. A higher level concept can then be created between the nodes mother, brother, son and a new node called uncle. The mother, brother and son nodes also store data of people that have these roles, while the brother node links to the uncle node. This type of rule is shown in Fig. 5. If links between mother and son, and mother and brother can be established based on the input values, then there is a complete chain and the uncle concept can be realised. This is a slightly different query process to the one discussed previously but still relatively simple and could be implemented.

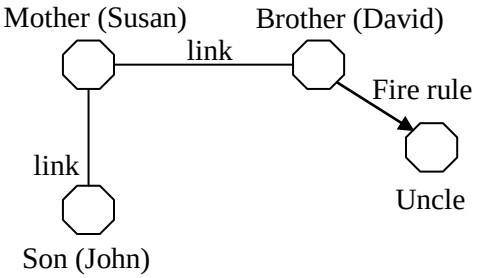


Fig. 5: A rule that can be autonomically triggered through links.

A more interesting possibility would be for the higher level concept chain of mother, brother and son to trigger the uncle concept when they are all active. After the rule is created, the links that determine when it is fired are still generated stigmergically and so this is still the main organising mechanism. This method would also mean that the network could autonomously realise concepts for itself without being queried about them. For example, if one concept is satisfied, then this could trigger another concept. If the other concept was part of another chain that did not have all related values, then the system could ask the environment for values and evaluate them. These values might then complete the new chain, which would trigger other concepts, and so on. This kind of autonomous triggering of one concept after another might be the beginning of thinking.

8.2 Generalisation

Concepts could also be generalised, as shown by the example illustrated in Fig. 6. Consider this figure, where there are two higher level concepts defined by the chains A-B-C-D and C-D-E.

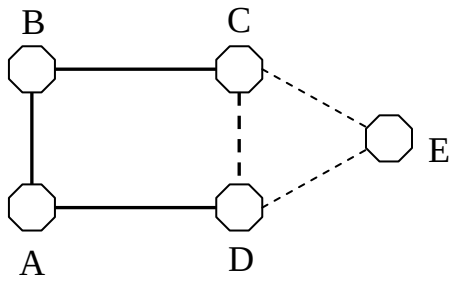


Fig. 6: Example network with two higher level concepts A-B-C-D and C-D-E.

The event A-B-C-E-D then occurs. This will also form a chain but it is not recognised as a valid concept. To recognise that this is not actually a concept, each link in a concept can check its keys, where all concepts in the path must have at least one key the same. Then there is no chance of an alternative route being considered as valid. If a new route is found to be acceptable however, then either a new key is added to all links in the route, or the two routes can be given the same key to indicate alternative valid possibilities. This might be beginning to generate some understanding, where the same concepts can be applied to different situations or slightly different concepts can be recognised as

being essentially the same. While the mechanisms look plausible, this is still all theory however. The probabilistic algorithms and similarity measures required to make these decision could still be complex.

8.3 Hierarchical Architecture

The research has thus revealed three different levels of knowledge, as described by the cognitive model shown in Fig. 7.

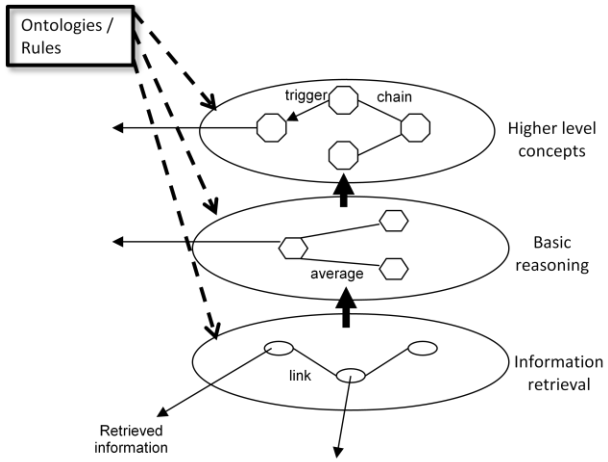


Fig. 7. Overall architecture for a cognitive querying system.

Ontologies or semantic rules can be used at any level to organise the network or provide missing knowledge. The dynamic links then provide for three different levels of intelligence. The first level is represented by the source nodes, associated by stigmergic links. This level stores information that may be directly retrieved. The second level has been discussed in section 7. It allows for basic reasoning over the information sources through simple aggregating mathematical operators. The first two levels of this architecture have been described in the paper already, as direct information retrieval through 'select-from-where' statements and basic reasoning. Tests have shown that these levels are reliable. The linking mechanism can produce an 80 – 90% reduction in search with maybe only a 5 - 10% related reduction in QoS. Tests have also shown that the linking mechanism helps with the low level reasoning. On top of this, a third level could be represented by chains of the source concepts, where each chain represents a higher level concept. This would allow for more sophisticated reasoning and understanding of the information stored in the network.

Future work will extend the architecture by adding this third level to the network. This level will autonomically link concepts associated together through user input, or through information provided by the system's environment. For example, a monitoring system could read sensor or image input to determine the key concepts in its environment. These higher level chains would have different uses. For example, the user could ask what he should cook with a tin of beans and spaghetti, when all recipe chains with these ingredients in them would be returned. More interesting is the following:

If the system monitors concepts for itself, then when one chain is realised, this could trigger a concept in another chain. The system would then try to realise the other chain by retrieving appropriate input, which could trigger another concept, and so on. The linked values would not all have to be retrieved from the environment. The original plan was for a sensorised environment to continually send information to a system that would then reason over it. However, the system could also store its own memory of related concepts or values. This might be as simple as storing the linking structure with related values. Then these values could be reasoned over when triggered.

This model thus represents a flexible way in which a network can begin to understand its contents and even reason autonomously over them. While the third level has yet to be proven, this offers genuine opportunities for introducing intelligence into the system. The model could even provide the beginning of autonomous thinking – realising things for yourself.

9 Conclusions

This work has focused on building an autonomic service-oriented architecture that can accommodate the pervasive sensorised environment. Specifically, reasoning has been a key requirement. An obvious

application of this work is in helping something like the Semantic Web [4], or mobile networks, to organise and reason over their information. The information network described would use both semantics (knowledge-based) and links (experience-based) to organise. Tests have shown that using dynamic links can effectively optimise the network with regard to the query process.

The way that the links are created allows them to be used in more knowledge-intensive queries and higher levels of reasoning have been suggested. For low level reasoning, the knowledge that is missing in the network is compensated by the knowledge of the users, allowing the network to remain lightweight. The architecture presented in section 8 however also suggests some sort of cognitive brain for a robot, or intelligent monitoring system. For this higher level reasoning, the system needs to be able to generate the knowledge associations for itself.

With increasing intelligence and reasoning capabilities, the boundaries between the localised neural-like systems and the massively distributed information systems may become more blurred, as their building blocks are quite similar in some cases. So while the system is neural in nature, the symbolic meaning of each node means that it can understand what each node represents. This provides a major advantage over existing neural network models. Thus the vision is ultimately a system that can think for itself, with all of the implications that this may provide. The distant goal will be for information or other systems to be able to operate independently of human assistance.

Some open source software used on this project, that can be used to build dynamic service-based networks, can be downloaded from the sourceforge.net web site http://sourceforge.net/projects/licas/. The software includes the linking mechanism used in this paper.

Acknowledgements

The authors would like to acknowledge the European Commission for funding the Integrated Project CASCADAS 'Component-ware for Autonomic, Situation-aware Communications, And Dynamically Adaptable Services' (FET Proactive Initiative, IST-2004-2.3.4 Situated and Autonomic Communications) within the 6th IST Framework Program, which has also funded this work.

10 References

[1] Al-Obasiat, Y. and Braun, R., A multi-agent flexible architecture for autonomic services and network management, *Computer Systems and Applications, AICCSA '07, IEEE/ACS International Conference*, 13–16 May, 2007, pp.132–138, ISBN: 1-4244-1031-2.

[2] Arbib, M. A., The Handbook of Brain Theory and Neural Networks: Second Edition, Bradford Books; 2Rev Ed edition (3 Jan 2003)

[3] Babaoglu, O., Canright, G., Deutsch, A., Di Caro, G., Ducatelle, Gambardella, F., Ganguly, N., Jelasity, M., Montemanni, R., Montresor, A. and Urnes, T., Design Patterns from Biology for Distributed Computing, *ACM Transactions on Autonomous and Adaptive Systems*, Vol. 1, No. 1, 2006, pp. 26 – 66.

[4] Berners-Lee, T., Hendler, J. and Lassila, O., The Semantic Web, *Scientific American*, May 2001, pp. 28 - 37.

[5] Breukner, S. and Van Dyke Parunak, H., Self-Organising MANET Management, G. Di Marzo Serugendo et al. (Eds.): *AAMAS 2003 Ws ESOA, Lecture Notes in Artificial I ntelligence (LNAI) 2977*, 2004, pp. 20 – 35.

[6] Dorigo, M. and Di Caro, G., Ant colony optimization: a new meta-heuristic, *Evolutionary Computation 2, CEC 99. Proceedings of the 1999 Congress on*, 1999, pp. -1477.

[7] Dragan, F., Gardarin, G. and Yeh, L., MediaPeer: a safe, scalable P2P architecture for XML query processing, *Database and Expert Systems Applications, Proceedings. Sixteenth International Workshop on*, 22-26 Aug. 2005, pp. 368- 373.

[8] Duch, W., Learning data structures with inherent complex logic: neurocognitive perspective, *6th WSEAS Int. Conference on Computational Intelligence, Man-Machine Systems and*

Cybernetics, Tenerife, Spain, December 14-16, 2007, pp. 293 - 302.

[9] Dutta, P., Grimmer, M., Arora, A., Bibyk, S. and Culler, D., Design of a Wireless Sensor Network Platform for Detecting Rare, Random, and Ephemeral Events, *Proceedings of the 4th international symposium on Information processing in sensor networks*, Article No. 70, 2005.

[10] Fu, L., Knowledge Discovery based on Neural Networks, *Communications of the ACM*, Vol. 42, no. 11, 1999, pp. 47 - 50.

[11] Grassé P.P., La reconstruction dun id et les coordinations internidividuelles chez Bellicositermes natalensis et Cubitermes sp., La théorie de la stigmergie: essais d'interprétation du comportment des termites constructeurs, *Insectes Sociaux*, Vol. 6, 1959, pp. 41-84.

[12] Greer, K., Baumgarten, M., Mulvenna, M., Curran, K. and Nugent, C., Stigmergic Linking for Optimising and Reasoning Over Information Networks, *Research Report*, 2008, http://licas. sourceforge.net/index_files/ReasonLink.pdf.

[13] Greer, K., Baumgarten, M., Nugent, C, Mulvenna, M. and Curran, K., Autonomous Querying for Knowledge Networks, *The* 5th *International Conference on Autonomic and Trusted Computing (ATC-08)*, June 23 - 25, Oslo, Norway, 2008, *Lecture Notes in Computer Science (LNCS)*, Eds. Rong et al., Vol. 5060, pp. 249 - 263, Springer-Verlag.

[14] Greer, K., Baumgarten, M., Mulvenna, M., Curran, K. and Nugent, C., Autonomic Supervision of Stigmergic Self-Organisation for Distributed Information Retrieval, *Workshop on Technologies for Situated and Autonomic Communications (SAC), at 2nd International Conference on Bio-Inspired Models of Network, Information, and Computing Systems (BIONETICS 2007)*, December 10-13, Budapest, Hungary, 2007, ISBN 978-963-9799-05-9.

[15] Heylighen, F. and Bollen, J., Hebbian Algorithms for a Digital Library Recommendation System, *Proceedings of the International Conference on Parallel Processing Workshops (ICPPW'02)*, 2002, pp. 439-.

[16] Holland J., Hidden Order: How Adaptation Builds Complexity. *Reading, MA: Perseus*, 1995.

[17] Joseph S., NeuroGrid: Semantically Routing Queries in Peer-to-Peer Networks. *In Proceedings of the International Workshop on Peer-to-Peer Computing (co-located with Networking 2002)*, Pisa, Italy, May 2002.

[18] Koloniari, G., Petrakis, Y., Pitoura, E., and Tsotsos, T., Query workload-aware overlay construction using histograms, *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, 2005, pp. 640 – 647.

[19] Kulkarni, A., Gunturu, H. and Datla, S., Association-Based Image Retrieval, *WSEAS Transactions on Signal Processing*, Vol. 4, Issue 4, April 2008, pp. 183 - 189.

[20] Michlmayr, E., Pany, A., and Graf, S., Applying Ant-based Multi-Agent Systems to Query Routing in Distributed Environments, in *Proceedings of the 3rd IEEE Conference On Intelligent Systems (IEEE IS06)*, September 2006.

[21] Mano, J-P., Bourjot, C., Lopardo, G. and Glize, P., Bio-inspired Mechanisms for Artificial Self-organised Systems, *Informatica*, Vol. 30, 2006, pp. 55 - 62.

[22] Menasce D.A. and Kephart., J.O., Autonomic Computing, *IEEE Internet Computing*, 2007, pp. 18 – 21.

[23] Neruda, R., Slusny, S. and Vidnerova, P., Evolution of Simple Behavior Patterns for Autonomous Robotic Agent, *6th WSEAS International Conference on System Science and Simulation in Engineering*, Venice, Italy, November 21-23, 2007, pp. 411 - 417.

[24] Newell, A. and Simon, H.A., Computer science as empirical inquiry: symbols and search, *Communications of the ACM*, Vol. 19, No. 3, 1976, pp. 113 – 126.

[25] Raschid, L., Wu, Y., Lee, W., Vidal, M., Tsaparas, P., Srinivasan, P., and Kumar Sehgal A., Ranking Target Objects of Navigational

Queries, *8th ACM International Workshop on Web Information and Data Management WIDM '06*, 2006, pp. 27 – 34.

[26] Robinson, R. and Indulska, J., Superstring: a scalable service discovery protocol for the wide-area pervasive environment, *Networks, 2003. ICON2003. The 11th IEEE International Conference on*, 2003, pp. 699- 704, ISSN: 1531-2216, ISBN: 0-7803-7788-5.

[27] Sartiani, C., Manghi, P., Ghelli, G. and Conforti, G**.,** XPeer: A Self-organising XML p2p Database System, http://www.di.unipi.it/~ghelli/papers/SarManGhe04-p2pdb.pdf, 2004.

[28] Stevenson, G., Nixon P. and Dobson, S., Towards a reliable wide-area infrastructure for context-based self-management of communications. In *Autonomic communication: 2nd International IFIP Workshop on Autonomic Communication*, 2006, pp. 115 - 128. Ioannis Stavrakakis and Mikhail Smirnov, Eds, Vol. 3854 of LNCS. Springer-Verlag.

[29] Waldrop, M.M., Complexity: The Emerging Science at the Edge of Order and Chaos, *by Lawrence Sternlieb. Simon & Schuster*, 1993, ISBN-10: 0671872346, ISBN-13: 978-0671872342, 384 pages.