Engineering Self-Organising Systems.
Third International Workshop, ESOA 2005
Sven A. Brueckner, Giovanna Di Marzo Serugendo, David Hales, Franco Zambonelli (Eds)

Part I: Self-organising Mechanisms.

T-Man: Gossip-Based Overlay Topology Management. (Mark Jelasity and Ozalp Babaoglu)

Abstract. Overlay topology plays an important role in P2P systems. Topology serves as a basis for achieving functions such as routing, searching and information dissemination; and it has a major impact on their efficiency, cost and robustness. Furthermore, the solution to problems such as sorting and clustering of nodes can also be interpreted as a topology. In this paper we propose a generic protocol, T-MAN, for constructing and maintaining a large class of topologies. In the proposed framework, a topology is defined with the help of a *ranking function.* The nodes participating in the protocol can use this ranking function to order any set of other nodes according to preference for choosing them as a neighbour. This simple abstraction makes it possible to control the self-organization process of topologies in a straightforward, intuitive and flexible manner. At the same time, the T-MAN protocol involves only local communication to increase the quality of the current set of neighbors of each node. We show that this bottom-up approach results in fast convergence and high robustness in dynamic environments. The protocol can be applied as a standalone solution as well as a component for recovery or bootstrapping of other protocols.

Introduction. Large, dynamic, fully distributed systems must be organised so that they can communicate with each other. It is difficult to ensure that every node is aware of every other node because of the size and rate of change. Functions such as routing, searching, dissemination and aggregation have efficiency which is significantly impacted by overlay topology. T-MAN is a generic protocol that is robust, scalable, flexible and simple. It evolves towards the target topology with convergence times dependent only on the log network size.

The Problem. To construct some desirable topology by connecting all nodes in the network to the right neighbours. The proposal is a ranking function which defines the target topology by allowing all nodes to sort any set of potential neighbours in preference order. Now we have a topology construction problem. One way of obtaining ranking functions is through a distance function – a few common forms are Line and Ring, Mesh, tube and Torus, and Binary Trees.

The Proposed Solution. The Protocol. Each node executes the same protocol which consists of two threads: an active thread for initiating communication with other nodes, and a passive threat waiting for incoming messages. Each node maintains a view of a set of node descriptors. Nodes improve their view by using the views of their current neighbours and applying the selected ranking function. Sparce random node selection is also used to break out of local minima. The protocol is optimised by applying a connection limit of 1 and applying hunting (to check different nodes in the view).

Simulation Experiments. Conducted with 3 distance based ranking functions defining the ring, torus and binary tree. Performance measure used was the convergence factor – the number of target links found. The results clearly indicated a logarithmic relationship between network size and convergence speed. This is relatively independent of view size and the characteristics of the target topology.

Self Healing. The protocol was extended to cope with churn by deleting a few old node descriptors on each cycle. Experimental results indicated good performance even under very high rates of churn (10%) and that overly aggressive healing was detrimental. The network is also expected to bootstrap well even with high churn.

Application examples. This technique can be applied to linear directional sorting with significant clusters (i.e. so that the clusters are inter and intra sorted and not just linked within itself. T-MAN can be used to evolve a distributed hash-table (DHT)

Conclusions and future work. The T-MAN protocol is fast and can recover from massive failure, bootstrapping or even changing the ranking function on the fly. Further work into DHT and understanding the noted emperical behaviour of T-MAN.

Basic Approach to Emergent Programming: Feasibility Study for Engineering Adaptive Systems Using Self-Organising Instruction-Agents. (George, Gleizes, Glize)

Abstract. We propose to investigate the concept of an Emergent Programming Environment enabling the development of complex adaptive systems. This is done as a means to tackle the problems of the growth in complexity of programming, increasing dynamisms in artificial systems and environments, and the lack of knowledge about difficult problems and their solutions. For this we use as a foundation the concept of *emergence* and a multi-agent system technology based on cooperative self-organizing mechanisms. The general objective is then to develop a complete programming language in which each instruction is an autonomous agent trying to be in a cooperative state with the other agents of the system, as well as with the environment of the system. By endowing these *instruction-agents* with self-organizing mechanisms, we obtain a system able to continuously adapt to the task required by the programmer (i.e. to program and reprogram itself depending on the needs). The work presented here aims at showing the feasibility of such a concept by specifying, and experimenting with, a core of *instruction-agents* needed for a sub-set of mathematical calculus.

Introduction. **Neo-computing problems**. Huge software construction and management is becoming increasingly difficult –.. IBM quote. **Problem Solving by Emergence**. We should attempt to harness emergent behaviour as nature does for engineering purposes. Start from systems theory, focus on the part of the system and their function, don't impose system behaviour and be a generic approach independent of implementation "hardware". **Going to the lowest level: The Instructions**. We propose focussing at the instruction level and rely on emergence and self-organisation on the one hand and on a multi-agent approach on the other.

Emergent Programming. **The Concept** "Emergent Programming is the automatic

assembling of instructions of a programming language using mechanisms which are not explicitly informed of the program to be created." Pg 18. It is an exploration of search space. Adaptive Multi-Agent System AMAS Theory can be considered as a guide to endow the agents with the capacity to continuously self organise so as to always tend toward cooperative interactions between them and the environment. It then claims that a cooperative state for the whole system implies the functional adequacy of the system. **The Instruction Agents.** Each manage a particular instruction or function. **The Reorganisation Process**. Everything depends on this step which enable the Agents to progress towards the adequate function, depending on the constraints of the environment but without knowing the organisation to reach or how to do it. **The neo-programming environment**. The neo-programmer will at least initially need to provide the tools which shape the environment and the qty and type of instruction agents.

Using cooperative agents as the engine for self-organisation. **Adapt the system by its parts**. The **theorem of Functional Adequacy**. For any functionally adequate system, there exists at least one cooperative internal medium system that fulfils the equivalent function in the same environment. A cooperative internal medium system is a system where no non-cooperative situations exist. An agent is in a non-cooperative situation (NCS) when (1) a perceived signal is not understood or is ambiguous (2) perceived information does not produce any activity of the agent (3) the conclusions are not useful to others. **Consequence**. We concentrate only on (1) Systems which adapt to a cooperative functional system without knowing the global function (2) Where feedback is the driving function both locally and globally (3) The parts only react to their local knowledge. The Engine for Self-organisation. The designer provides the agents with a mechanism to detect and respond to NCS.

The Elementary Example. Six simple agents each with input and output (3 of constant, +, * and output). The reorganisation mechanism involved simple messages for an agent to communicate that there is a NCS (i.e. missing input) and to respond cooperatively on receipt of such a message. If the recipient cannot respond (ie has an output of the wrong type) it must relay the message to its acquaintances. The mechanism requires an overall directional huristic to judge the output. Agent conflict is managed by requiring the agent to adapt in a manner which causes the least NCS for its neighbours. This system reliably attains functional adequacy and does so by monotonic improvement. Emergent programming should be a Universal Tool and should perform in environments with multiple ill defined constraints.

Emergence and Self-Organisation. Self-organisation is the set of processes within a system, stemming from mechanisms based on local rules which lead the system to produce structures or specific behaviours which are not dictated by the outside of the system. Self organisation is clearly dictated by the environment but we think of something as more self-organising when the environmental constraints are less precise. **Using Emergence in Artificial systems**. It is the function that has to emerge. There must be a mechanism for the adaptation of the system and coupling to the environment. It is the organisation of the parts which must change without knowledge of the collective function or how to reach it.

Conclusions. This approach describe is useful for neo-computing for ambient intelligence (multiply independent systems which are brought together to deliver value)

ETTO: Emergent Timetabling by Cooperative Self-Organisation. (Picard, Bernon, Gleizes)

Abstract. Cooperation is a means for multi-agent systems to function more efficiently and more adaptively. Cooperation can be viewed as a local criterion for agents to self-organize and then to perform a more adequate collective function. This paper mainly aims at showing that with only local rules based on cooperative attitude and without any global knowledge, a solution is provided by the system and local changes lead to global reorganization. This paper shows an application of cooperative behaviours to a dynamic distributed timetabling problem, ETTO, in which the constraint satisfaction is distributed among cooperative agents. This application has been prototyped and shows positive results on adaptation, robustness and efficiency of this approach.

Introduction. Artificial systems are becoming harder to design. This paper follows the Adaptive Multi-Agent Systems (AMAS) approach where non-cooperative situations (NCS) are avoided when an agent c_{per} perceived signals are understood without ambiguity c_{dec} receives information is useful for the agents reasoning c_{act} reasoning leads to useful actions toward other agents. This paper shows that global function can adapt as a result of only locally applied rules.

ETTO Agents. The example application is University timetabling. Each Student and each Lecturer has a **Representative Agent** (RA) which specifies various intrinsic constraints such as equipment requirements or personal availability. Each RA has one **Booking Agent** (BA) for each course being taken/given. The BA's occupy one position in a grid of rooms vs time slots. BA's can partner/unpartner another BA and book/unbook a grid position by sending messages to other agents it knows. Each agent *perceives* its received messages, *decides* what to do and then *acts*. **Basic Behaviour**. During perception a BA checks its mailbox about partnership and reservation requests, and if its goal is not met within its constraints then it evaluates previously encountered BA's and grid positions before randomly moving to a new grid position**. Constraint Management** is done with the information held by the agents themselves about intrinsic constraints (from the RA) and induced constraints (from brother BA's via the RA). The intent is to provide robust and local mechanisms rather than sound, complete or terminating solutions.

Cooperative Self-organisation rules. Five different situations for reorganisation are identified, two relating to c_{dec}, three relating to c_{act}, but none relating to c_{per} as all the agents are the same. The idea is to design these rules as exceptions at the agent level rather than the instruction level. **Partnership incompetence c_{dec}** is when the BA's are incompatible (ie two teachers courses). **Reservation incompetence c_{dec}** is when the reservation does not suit the BA's constraints (ie room too small). **Partnership conflict c_{dec}** is when one BA wishes to partner with an already partnered BA (ie a student wants a fully booked lecture). The BA which most easily finds partnerships must relinquish a reservation. **Reservation conflict c_{act}** is the same as Partner conflict but for a room/timeslot. **Reservation Uselessness c_{act}** is when a BA is in the same cell as one of its brothers (ie a student with two booking in the one room/timeslot)

Prototyping and Experiments. Influence of Cardinality. The more BA's a system has the more efficient the solving. **Constraint Relaxation**. Assists solution finding. **Dynamic Resolution**. Can occur at run time and adaptation is continuous not batch.

Discussion. ETTO has strengths in that the solve process is continuous and can embrace additional agents or constraints at run time. A related advantage is that the solve does not start from scratch each time but builds on the existing solution. A weakness is that processing over constrained problems is not fully efficient because agents continue to explore the grid for more relevant solutions as they are unaware of the global quality of the solution. Also grid exploration is random.

Conclusion. The AMAS approach can solve poorly specified problems in a dynamic adaptive manner.

Self-Adaption and Dynamic Environment Experiments with Evolvable Virtual Machines. (Nowostawski, Epiney, Purvis)

Abstract. Increasing complexity of software applications forces researchers to look for automated ways of programming and adapting these systems. Self-adapting, self-organising software system is one of the possible ways to tackle and manage higher complexity. A set of small independent problem solvers, working together in a dynamic environment, solving multiple tasks, and dynamically adapting to changing requirements is one way of achieving true self-adaptation in software systems. Our work presents a dynamic multi-task environment and experiments with a self-adapting software system. The Evolvable Virtual Machine (EVM) architecture is a model for building complex hierarchically organised software systems. The intrinsic properties of EVM allow the independent programs to evolve into higher levels of complexity, in a way analogous to multi-level, or hierarchical evolutionary processes. The EVM is designed to evolve structures of self-maintaining, self-adapting ensembles, that are open-ended and hierarchically organised. This article discusses the EVM architecture together with different statistical exploration methods that can be used with it. Based on experimental results, certain behaviours that exhibit self-adaptation in the EVM system are discussed..

Introduction. Genetic programming and linear genetic algorithms operate on a predefined fitness landscape and are therefore difficult or impossible to use in multi-task dynamic environments. The proposed framework consists of a set of independent computing cells that compete for limited resources and dynamically change their functionality and functional dependency.

Computation and biological Inspiration. Current Evolutionary Computing (EC) research is information centric inspired by random mutation and natural selection but these are insufficient in principal. The Evolvable Virtual Machine (EVM) architecture is based on an evolutionary model inspired by hypercycles, autopoiesis and sybiogenesis.

Symbiogenesis and specialisation. Mitochondria may have become to exist in other cells as a result of a bacterial infection which turned out to be symbiotic. Specialisation occurs extensively in populations. Working hypothesis that both specialisation and symbiosis are necessary to reach higher levels of complexity. **Abstract self-organisation application architecture**. The EVM system interacts with the environment with tasks and solutions. Multiple virtual machines can share one or more available resources.

EVM Assembly. A new assembly was written to be able to manipulate machine levels by defining a new base level on which to build. The language itself is capable of redefining itself and is highly expressive so that tasks can be expressed in compact form. Recursion and looping are available. The principle objective is to facilitate searches for languages specialised for a given set of problems.

EMV Implementation. The implementation is written in Java and almost entirely comparable to an integer based subset of the JVM. (www.sf.net/projects/cirrus) The basic data unit is a 32bit signed integer. There is a data stack which passes all operands except *push* and there is a list stack. Each machine is a list of lists of instructions. The base machine implements all primitive instructions. Each thread has a maximum number of instructions before it is unconditionally terminated. programs can add, modify and remove instructions from the lowest level virtual machine. Programs can construct higher level machines. A running program can alter the context between lower an upper level machines.

Specialisation of an Individual Machine. The specialisation mechanism attempts to find a set of instructions to solve a task using a search technique such as random, GA or stochastic. **Random** cannot take advantage of the fitness landscape but is simple, fast and requires little memory. **GA** does not dynamically adapt to different exploration strategies for different environments. GA prefer more stable shorter solutions and settles on sub-optimal solutions. **Stochastic** search limits the number of instructions and machines so that a probability distribution can be applied to each possible instruction. This approach is good at remembering solutions but does not take into account correlation between instructions so a program is often disrupted by changing a single instruction (a mechanism which progressively froze a successful set of instructions successfully alleviated this down falling).

Experiments with a web of interacting agents. Suppose that several machines lie on an n-dimensional grid. Each machine can execute primitives (add, swap etc) but can also execute its neighbours programs (ie add dup program2ofLeftNeighbour mul program1 ofRightNeighbour …). If a program gets a reward it will share it with any neighbour program used to compute the solution to allow symbiotic relationships to develop. **Environment**. The model should: solve all tasks, reward more difficult tasks, concentrate resources on unsolved tasks, remember solutions while useful, diffuse solution knowledge, allow tasks to change dynamically. Rewards can be adjusted automatically by say making them proportional to the number of units required to solve the task. **Parameters and their impact**. Cells store and consume rewards. The amount of reward storage critically affects the dynamism of the web with little change with high storage and frequent extinction with low storage (no smooth transition). Knowledge diffusion leads to neighbour cells parasiting solutions and then competing with the original. This effect reduces when an independent solution is found.

Film Analysis vs. **Quantitative Studies.** Presenting the web dynamics as a film enables researchers to progress much more rapidly and intuitively.

Summary. The EVM architecture is particularly effective when applied to problems with an inherently organised structure.

Choose Your Tribe! – Evolution at the Next Level in a Peer-to-peer Network. (David Hales)

Abstract. Many peer-to-peer (P2P) applications benefit from node specialization. For example, the use of super-nodes, the semantic clustering of media files or the distribution of different computing tasks among nodes. We describe simulation experiments with a simple selfish re-wiring protocol (SLAC) that can spontaneously self-organise networks into internally specialized groups (or "tribes"). Peers within the tribes altruistically pool their specialisms, sharing tasks and working altruistically as a team - or "tribe". This approach is scalable, robust and self-organising. These results have implications and applications in many disciplines and areas beyond P2P systems..

Introduction. P2P is widely used for file sharing but it could be used for say a backup service if nodes with free storage, high bandwidth and firewalls could spontaneously self-organise.

Behavioural Assumptions in Open Networks. Nodes behave however they like so how do we design a suitable protocol? Assuming rational nodes implies game theory which is restricting and in P2P networks evolutionary game theory is ill fitted because the nodes do not reproduce and the fitness landscape is undefined. So, assume (1) nodes are in the network for what they can get out of it (2) nodes modify their behaviours to improve their benefit and (3) nodes operate with limited knowledge about other peers and the network in general.

The SLAC Algorithm. SLAC is a simple “copy and rewire” rule which stands for *Selfish Link and behaviour Adaptation to produce Cooperation*.

```
DO periodically forever
    select a random node j from the network
compare utility of this node (i) with node j
    IF utility of j is higher (Uj >= Ui)
    drop all current links (clear view of i)
    copy links of node j (copy view from j to i)
    add link to j (add to view i a link to j)
    copy behavioural strategy of j
        with a low probability (mutation rate 1)
            drop all current links (clear view of i)
            add a link to a randomly chosen node
        with a low probability (mutation rate 2)
            change the behavioural strategy randomly
END IF
END DO
```

The Skill-world Scenario. Skill-world is a graph of N nodes with 0-20 undirected connections. Each node has a skill type (1-5), a altruism flag (0-1) and a utility (R+). Periodically, a node in the network is randomly selected and assigned a job requiring a particular skill to complete. A node with a matching skill will complete the job (and gain a credit=1) or request each neighbour to complete the job. The neighbour will complete the job if its skill matches and its altruism = 1 (this costs the neighbour 0.25 credits.

SLAC in the Skill-World. Success is measured as the proportion of submitted jobs completed successfully. **Some Experiments and Results**. High performance (PCJ % completed jobs > 90) was obtained within a 10-50 cycles for networks $N=10^5$ indicating a very low scaling cost. The network recovers from 100% non-altruism. Populations with N<1000 do not achieve high performance. **History in SkillWorld – Tribal Dynamics**. Initially tribes develop quickly but a tribe will wither away when it becomes dominated by non-altruists. **Tribal Structures**. Tribes tend towards an evenly connected network with a range of skills.

Conclusion. The tribe is an emergent phenomena and while the tribes die the nodes do not. SLAC is a simple algorithm that results in complex group level evolutionary dynamics. SLAC is simple to implement and study empirically but it I not know how to study it theoretically with proofs. The concept of utility may not hold in many task domains.

Exchange Values and Self-regulation of Exchanges in Multi-agent Systems: The Provisory, Centralized Model. (G.P.Dimuro & A.C R Costa)

Abstract. This paper introduces systems of exchange values as tools for the self-regulation of multi-agent systems. Systems of exchange values are defined on the basis of the model of social exchanges proposed by J. Piaget. A model of social control is proposed, where exchange values are used for supporting the regulation of the performance of social exchanges. Social control is structured around two coordinated functions: the evaluation of the current balance of exchange values and the determination of the target equilibrium point for such balance, and the maintenance of the balance of exchange values around the current target equilibrium point. The paper focuses on the second function of social control, introducing a (for the moment, centralized) equilibrium supervisor that solves the problem of keeping the system in a state of equilibrium by making use of a Qualitative Markov Decision Process that uses intervals for the representation of exchange values.

The result of interbreeding between accountants and mathematicians.

A New Protocol to Share Critical Resources by Self-organized Coordination. (F. Armetta, S Hassas, S. Pimont)

Abstract. In this paper we propose a new approach to share critical resources through self-organized coordination. Our approach addresses two levels: first, it allows the expression of agents needs in terms of resources occupancy (getting/leaving a resource) with respect to agents objectives; second, it allows to find a global resources exchange scheme between agents, fitting the agent needs thanks to a negotiation process. Negotiation between agents is held in a decentralized way and allows to produce complex contracts. The underlying protocol is applied to schedule manufacturing tasks on a set of critical machines.

Introduction.

Presentation of the General Problem: **The Sharing of Critical Resources in a Constrained Environment**. The problem is considered in a repairing context in that every time a disturbance occurs, one has to adapt the current situation.

Addressing this Problem: A Contextural Approach. We could move throughout the search space in a pre-stated order or non-prestated where the possibilities cannot be estimated or characterised. Complexity of the search space can be reduced by utilising domain specific knowledge (ie branch cutting) or we can take advantage of a process that exhibits a naturally good behaviour (i.e. ant, swarm, neural nets, GA etc). The search process can be centralised or decentralised.

Contribution for the Coordination. The question is how to control the system face to disturbances. “rather than finding efficient ways to reduce the time consuming for the problem solving, we attempt to understand the characteristics of the problem behaviour and then lead the system evolution” pg 92. The proposed approach is characterised by self organisation (defined by entropy), stigmergy (indirect communication mediated by the environment) and eco-resolution (reactive

agents guided by a fitness function). Constraints such as job shop sequencing and due dates are difficult to represent. An agent's decision must have significant positive effect to balance the disturbance that is causes in the network (and the computational and memory resources required to resolve it) To address these difficulties it is proposed to decentralise the contract elaboration process and centralise the contract validation. A stigmergic approach is taken to handle the complexity of the contracts required.

The Coordination Model Proposed. "…. agent activity can prevent the stabilisation of the multi-agent system on a good quality solution. Instead of reducing this activity we propose a model to check the relevance of each agent activity." Pg 95 **The Agent Model** operates at two levels. The application level represents the real time situation of the agents. The collaboration level represents the capacity for agents to exchange parts of resources. The collaboration principle. A collaboration consists in an exchange of critical resources between agents. Each agent has to communicate an intention to get a resource or leave a resource. Local potential collaborations are elaborated by a process which regularly associates complimentary intentions (two or more) but does not evaluate their relevance. Once the collaboration net is elaborated, a process inspired by swarm intelligence. Temporary constrains violations are allowed, which then propagate on the application level. Some collaborations are validated by the system, some are not.

Simulation Results. **Implementation behaviours**. A pheromone (which evaporates) is assigned to each collaboration and this is used by the agent to evaluate and sort the current lists of collaborations (get & leave). A collaboration is *natural* when it is preferred by an agent and *conform* when it is natural for all associated agents. Agents act when they are unsatisfied. An agent reinforces a collaboration when its desirability exceeds a given threshold. **Results.** Promising but require further validation.

Industrial Application. Industrial scheduling is a hard problem.

Conclusion. There are relatively few attempts to solve the Industrial Scheduling problem with a multi-agent approach – probably because of the difficulty in representing constraints.

Part II: Methodologies, Models and Tools.

Information-Driven Phase Changes in Multi-agent Coordination. (Sven A. Brueckner and H.V.D. Parunak)

Abstract. Large systems of agents deployed in a real-world environment face threats to their problem solving performance that are independent of the complexity of the problem or the characteristics of their specific solution mechanism. One such threat is the degrading of the quality of agent coordination mechanisms when faced with delays in the flow of critical information among the agents introduced by communication latencies. In this paper we demonstrate in a simple model of locally interacting agents that the emerging system-level performance may degrade very suddenly as the rate of individual decision making increases against the availability of up-to-date information. We present results from extensive simulation experiment that lead us to select a locally accessible metric to adapt the agent's individual decision rate to values that are below this phase change. Given the generic nature of the coordination mechanism that is analysed and the information-theoretic metric, the adaptation mechanism may increase the deployability of large-scale agent systems in real-world applications.

Introduction. A simple graph colouring model is used to explore the phase change. Knowing the location of the phase change is important to optimise the solution speed vs quality trade-off. Communication latency can be dynamic in time and space. A solution is proposed whereby the agent can determine the optimum decision rate at a local level.

Distributed Graph Colouring. The problem seeks to assign colours to nodes such that no node is connected to another of the same colour. The specific real world application involved the dynamics of a distributed sensing and tracking system comprising a large number of small robotic radar sensors. The colouring problem is an abstract problem but provides common general results when nodes are taken to represent a task, colours a resource and edges an indication that a single resource cannot serve two tasks simultaneously. Each agent cyclically decides whether or not to reconsider its colour choice or not. This *activation level* can be systematically altered as can the *communication latency*. A low rate random node failure was included.

Systematic Parameter Sweeps. Beware artefacts introduced by pseudo random number generators. Set-up, metrics and meta metrics were recorded in XML and analysed with Mathematica, Excel and Java.

Information Driven Phase Changes. **Parameter Space**. The parameters (9) can be grouped into problem parameters (e.g. N = number of nodes etc.), solution parameters (e.g. AL = average decision rate etc) and environmental parameters (e.g CL = communication delay). **Metrics**. Global Degree of Conflict (DoC), Option Set Entropy (OSE), & False Information Percentage (FIP) are all presented as a ratio with random performance. **Mapping the Problem Space**. The research focusses on the transition between under-loaded (more colours than is strictly required) and overloaded (less colours than strictly required) states in the system. The results are presented on 3 x 3D charts DoC, OSE and FIP with varying N (number of nodes) and G (number of colours available). A distinct transition is apparent between regions where there is good performance and where there is random equivalent performance or worse. The poorly performing regions are characterised by behaviour such as asymptotic (solution is easy so random does well), thrashing (due to outdated information) and drifting (between equivalent states).

Phase Changes. When individual runs are examined (as opposed to just averages) there is a clear phase transition from stable to thrashing across all key measures. There is a clear region of overlap where both stable and thrashing behaviour are apparent.

Introspection and Learning. When the phase change is examined as a function of varying the rate of decision making (AL) there

is a particularly distinct phase change in FIB, offering a good metric for local agent control. If there is communication latency and AL is to high then the system thrashes. But with a globally low AL the system response is unnecessarily slowed. Each agent can be simply augmented with a basic learning mechanism which monitors FIB and adjusts AL down until thrashing stops. The monitor on FIB also has a "pheromone which evaporates" and slowly forgets history and allows AL to drift back up over time. This technique works well.

Conclusion. The results are generic and broadly applicable. Phase changes are readily apparent as communication latency increases but this can be effectively controlled with a simple local learning mechanism.

Self Organising Applications Using Lightweight Agents (Paul Marrow and Maniolis Koubarakis)

Abstract. Self-organisation in nature is responsible for many complex and persistent phenomena. This suggests that self-organisation may be useful in the creation of complex applications. Multi-agent systems use multiple agents to execute complex activities, and thus may be a basis for self-organising applications. In this paper we describe applications using self-organisation based upon the DIET multi-agent platform that supports lightweight agents. Multi-agent systems can be created that support decentralisation, scalability and adaptability. We show that these application properties are useful for information sharing in mobile communities via self-organising among middle agents, and via peer-to-peer interaction between agents.

Introduction. Self Organising systems need to adjust their properties robustly, to be scalable, and decentralised.

System Outline. **The DIET Agents Platform** is a tree layer architecture. The core layer contains the kernel and enables environments, agent creation, connections and messages between agents, debugging and visualisation. The application reusable component layer includes the likes of remote communication and event scheduling. The application layer contains code specific to particular applications. **Kernel Properties**. A hierarchy of elements is defined: Worlds, environments, Agents, Connections, Messages. The implementation is resource constrained and fail-fast (if an action cannot be executed immediately it fails). **Application Properties**. The agents are decentralised (across multiple environments and worlds if necessary, Scalable (due to light weight) and Adaptable (via collaboration with other agents).

Self Organising Applications. Self Organising Communities are demonstrated with an application with user agent (representing a user) and middle agents (that act as intermediaries). Implementing P2P

Systems with P2P-DIET is demonstrated again with two tiers: peers and super-peers supporting one-time queries and standing queries by the peers.

Discussion. Both the examples relate to information retrieval amongst a diverse community of users.

Solving Dynamic Distributed Constraint Satisfaction Problems with a Modified Weak-Commitment Search Algorythm. (K Saechai, L Benedicenti and R Paranjape).

Abstract. Constraint Programming research is currently aimed at solving problems in a dynamically changing environment. This paper addresses the problem of solving a Dynamic Distributed Constraint Satisfaction Problem (Dynamic DCSP). The solution proposed is an algorithm implemented in a multi-agent system. A Dynamic DCSP is a problem in which variables, values and constraints are distributed among various agents. Agents can be freely added to or removed from the system. Most advanced applications cannot be represented by DCSPs, but they can be modelled by Dynamic DCSPs. The algorithm described in this paper is an extension of the Asynchronous Weak Commitment Search algorithm (AWCS) originally proposed by Yukoo [10]. The extended algorithm is designed to cope with the dynamically changing parameters of a Dynamic DCSP. The proposed algorithm differs from other Dynamic DCSP algorithms because it allows an unlimited number of changes to any of the variables, values, or constraints. This paper describes an agent system implementing the modified AWCS algorithm and verifies its effectiveness by applying it to a dynamic N-Queens problem. The results prove the applicability of the modified algorithm to Dynamic DCSP.

Introduction. This paper details a dynamic extension to the Asynchronous Weak-Commitment Search (AWCS) algorithm. Constraint satisfaction problems typically are defied by variables, values and constraints. Multi-agent solutions typically allocate one agent per variable. Most real-world problems are dynamic (hospital scheduling, job-shop etc.) requiring a dynamic solution. The original AWCS combines iterative improvement (hill-climbing) with min-conflict algorithms.

Proof of Concept. A dynamic version of the n-queen problem (locate the n queens on a chess board such that all are safe) where

queens and rows are added when a solution is found to the current problem.

Modified AWCS algorithm. Queen agents may receive three types of message: OK? Nogood and agent-list.

Results. The algorithm could solve the n-queen problem. The nogood part of the algorithm was not needed for problems with less than 8 queens but with greater than 8 queens the nogood algorithm prevented backtracking.

Conclusions and Future Work. Further study and extensions

Development of Self-Organising Emergent Applications with Simulation-Based Numerical Analysis. (Tom De Wolf, Tom Holvoet, Gionanni Samaey)

Abstract. The goal of engineering self-organising emergent systems is to acquire a macroscopic system behaviour solely from autonomous local activity and interaction. Due to the non-deterministic nature of such systems, it is hard to guarantee that the required macroscopic behaviour is achieved and maintained. Before even considering a self-organising emergent system in an industrial context, e.g. for Automated Guided Vehicle (AGV) transportation systems, such guarantees are needed. An empirical analysis approach is proposed that combines realistic agent-based simulations with existing scientific numerical algorithms for analysing the macroscopic behaviour. The numerical algorithm itself obtains the analysis results on the fly by steering and accelerating the simulation process according to the algorithm's goal. The approach is feasible, compared to formal proofs, and leads to more reliable and valuable results, compared to mere observation of simulation results. Also, the approach allows to systematically analyse the macroscopic behaviour to acquire macroscopic guarantees and feedback that can be used by an engineering process to iteratively shape a self-organising emergent solution.

Introduction. Centralised control systems for Automated Guided Vehicles in warehouses do not scale well and are inflexible. “The goal of engineering self-organising emergent systems is to acquire a system with a coherent macroscopic behaviour which meets the requirement and results solely from autonomous local activity and interaction. This paper describes an approach to systematically analyse macroscopic behaviour with an algorithm that guides and accelerates the simulation process. These techniques could be embedded into the engineering process to shape the self organising emergent solution.

The Case-Study: Automated Guided Vehicles.

The Analysis Approach. Constructing a formal model and correctness proof of a complex interacting system is infeasible. Wegner proves that computer systems using interaction are more powerful problem solving engines than mere algorithms. Empirical analysis is an alternative which focusses on relevant properties and ignoring irrelevant ones by defining macroscopic variables. **Analysis of Self-organising Emergent Systems: Trends**. Self-organising systems promise to be scalable, robust, stable, efficient, and exhibit low latency, but also behave non-deterministically. While the exact evolution is not predictable, the macroscopic trends (averaged over a number of system runs) that are predictable. Robustness is preferred over optimal behaviour. We seek macroscopic guarantees. **Modelling: Individual-Based Versus Aggregate Based**. Aggregate-based models define (typically equation based) relationships between macroscopic variables. Individual-based models consist a set of agents with defined behaviours and the macroscopic behaviour is explored thru simulation. Individual models are easier to construct, easier to adjust and simulation probably yields more realistic results but numerous simulations are required to guarantee macroscopic behaviour. Equation-based models benefit from a broad arsenal of numerical tools. **Equation-free Macroscopic Analysis**. This paper combines numerical analysis techniques with realistic simulations. A simulation (rather than an equation) is used to evaluate the value of macroscopic variables required by the numerical analysis technique towards locating steady state behaviour or some other goal. Numerical analysis often assumes smooth behaviour but the main focus is indeed analysing gradually evolving trends. **The approach**. Most numerical techniques have no explicit need for the macroscopic equation – just the evaluated result. The *macroscopic-time-stepper* takes initial values for the macroscopic variables and then initialises a number of simulations which meet the macroscopic requirements but with randomly selected agent values. The simulations are then run for the required duration and the resulting averages (across all simulations) calculated for the macroscopic variables. In this way the simulation(s) replace the equation. Then the analysis algorithm determines the next set of macroscopic variables to be evaluated/simulated. An example of a numerical technique is the projective integration algorithm which has the goal of accelerating the simulations by reducing the number of simulation steps by using extrapolation. A few “close” macroscopic values are evaluated/simulated then a extrapolation is made over a much “larger” distance. Newtons algorithm is another acceleration technique to locate steady state behaviour. **Road Map**. An ant foraging scenario is provided as an example to work thru the step involved in setting up an equation-free analysis. This approach bridges the gap between the macroscopic behaviour and the individual agent behaviour. **Macroscopic Variables in the AGV Case**. Example description. The greatest challenge is to find suitable variables that reflect the evolution of the macroscopic properties for which one needs guarantees.

Engineering Based on Analytics Results. There is a lack of a systematic approach to build a solution that meets the requirements. Traditional engineering involves assembling known components into subsystems and then into a system that is then tested. This paper proposes incorporating the integration of the systematic analysis approach into the iterative engineering design cycle.

Conclusion and Future Work. The analysis approach can add value to engineering self-organising systems. Future work will be in the AGV domain and into how to choose macroscopic variables.

Further reading. Wegner

On the Role of Simulations in Engineering Self-organising MAS: The Case of an Intrusion Detection System in TuCSoN. (Luca Gadelli, Mirko Viroli and Andrea Omicini)

Abstract. The intrinsic complexity of self-organising MASs (multi-agent systems) suggests the use of formal methods at early stages of the design process in order to predict global system evolutions. In particular, we evaluate the use of simulations of high-level system models to analyse properties of a design, which can anticipate the detection of wrong design choices and the tuning of system parameters, so as to rapidly converge to given overall requirements and performance factors. We take intrusion detection (ID) as a case, and devise an architecture inspired by principles from human immune systems. This is based on the TuCSoN infrastructure, which provides agents with an environment of artefacts-most notably coordination artefacts and agent coordination contexts. We then use stochastic π-calculus for specifying and running quantitative, large-scale simulations, which allow us to verify the basic applicability of our ID and obtain a preliminary set of its main working parameters.

Introduction. The cost of software maintenance is approaching that of development – and self-organising techniques are one promising solution. This paper promotes pi-stochastic π-calculus process algebra to simulate MAS dynamics early in the design process. The example uses is an IDS inspired by the principles of the human immune system.

Human Immune System and a MAS Architecture. **Security and IDS's in Information Systems**. Authentication and authorisation approaches are often circumvented by application flaws. An IDS attempts to detect abnormal behaviour and prevent it. **Human Immune System Overview.** An antigen is any molecule that triggers an immune response. General passive resistance includes the skin, temperature, pH etc. Active systems include scavenger cells that detect and kill a fixed set of antigens and the acquired immune system which is adaptive. The adaptive system comprises different types of cells with only lymphocytes considered here. The lymphocytes are produced in bone marrow and mature in the thymus where they are eliminated if they bind to self-cells. Lymphocytes have about a two day lifespan which is extended if they bind to an antigen (a form of memory). **A General Architecture for MAS Applications**. A class of agents are introduced which each detect and learn suspicious behaviour or are replaced if ineffective.

Simulations in π–calculus. The π-calculus model is a formal model developed to reason about concurrency in a system of agents which interact and continuously change neighbours. The basic entity is a name which is used as an unstructured reference to a synchronous communication channel that can send and receive messages. **On Stochastic Models**. Formal models who's semanticsm is given by a transition system can be extended to a stochastic version (i.e. Markov transition system). Each transition is ladled with a rate that scales how the transition probability increases with time. Rates can be used to express aspects such as probability, speed, delays and so on. **Stochastic π-Calculus and the π-Machine**. (I could not make sense of this very brief summary – further research probably worthwhile).

Simulating Self-organising Systems. Three scenarios are given which begin with a basic system with agents entering and leaving with a proportion being malicious. Scenario #1 adds a variable number of agents, rate of inspection and the probability of detection. Scenario #2 adds a limited agent lifetime which varies based on it ability to detect malicious agents. Scenario #3 the hypothesis that malicious behaviour are not homogeneous and that agents can accept behavioural upgrades from its neighbours. Each scenario performs better than the last.

Conclusion. π–calculus is effective as a design tool

Mesoscopic Modelling of Emergent Behaviour – A Self-organising Deliberative Minority Game. (Wolfgang Renz and Jan Sudeikat)

Abstract. Recent research discussed several approaches to understand the relation between microscopic agent behaviour and macroscopic multi-agent system (MAS) behaviour. A structured methodology to derive these models will have impact on MAS design, evaluation and debugging. Current results have established the description of macroscopic behaviour, including cooperation, by Rate Equations derived from markovian agent-states transitions. Emergent phenomena elude these descriptions. In this paper, we argue that mesoscopic modelling is needed to provide appropriate descriptions of emergent system behaviour. The mesoscopic agent states reflect the emergent behaviour and allow for a deliberative implementation of the rules and conditions which cause the MAS to self-organize as wanted. In a case study, we construct such a mesoscopic model for the socio-economic inspired Minority Game. The mesoscopic description leads us to a deliberative implementation, which exhibits equivalent self-organizing behaviour, confirming our results.

Introduction. Includes a broad survey of the MAS analysis literature. This paper introduces mesoscopic modelling as a refinement to previous techniques that derives mathematical descriptions of systems exhibiting emergent behaviour. A mesoscopic modelling level introduces hidden agent states (not directly observable). In addition, a method is proposed for constructing macroscopic equivalent reliable self-organising systems.

Microscopic Models, Macroscopic Descriptions and Mesoscopic Modelling. Macroscopic Description of Reactive Multi-Agent Systems. When it is possible to describe the probabilities of transitions between agent states then the Rate Equations for the observables can be written down directly. The global MAS behaviour is characterised by the proportion of Agents in a particular state. However, Rate Equations are

hampered by three sources of complication: 1. correlations between agent states vanish, 2. fluctuations in the numbers of agents in each state are eliminated and 3. discrete time effects are neglected. The well studied Minority Game (MG) is studied below where quantities characterising emergent behaviours are introduced such that the model can retain relevant fluctuations but otherwise abstract from the microscopic detail (ie mesoscopic modelling) **Mesoscopic Modelling.** We focus on the construction of less microscopic models with equivalent (and engineered guaranteed) macroscopic behaviour. To allow engineering it is necessary to build-in the expected behaviour rather than allowing to to emerge in a less policed way. The mesoscopic model is constructed by introducing variables which describe the emergent structures and applying an averaging process to identify mesoscopic states. Such a model is not macroscopic because it still contains short-time fluctuations.

From Microscopic to Mesoscopic Modelling – A Case Study with Minority Games. The El Farol Bar Problem depicts a bar in Santa Fa which is only enjoyable when it is not too crowded – agents are regularly forced to decide either to go or stay home – based on past attendances. Deductive reasoning breaks down under complication, forcing the agent to employ inductive reasoning as their rationality is bounded and they need to make assumptions about future behaviour of (irrational) opponents. **Stochastic Minority Game**. Each agent keeps a probability to change the go/nogo choice. After making a good choice the probability is adjusted. **Emergent Behaviour**. Specific behaviour emerges such as deterministic-alternation freezing regime or supplier-loyal behaviour which in both cases have agent win every second time. The emergent behaviour cannot be derived from the rate equation. **Mesoscopic Model**. The mesoscopic model will implement states which directly represent the emergent behaviour thereby accounting for correlation effects. In each of six states (loyal, alternating and undetermined for each of go and nogo) the agent counts its wins and losses and transitions to a neighbouring state when a threshold is exceeded. (thought: this is rather like reversing the derivation of a Hidden Markov Model as it replaces hidden agent states with observables rather than the other way around) **Deliberative Implementation**. Bratman developed a theory of practical human reasoning which describes rational behaviour by the notions Belief, Desire and Intention. The BDI model is suitable for developing deliberative agents. (implemented in the JADE platform). **Comparison and Results**. This mesoscopic deliberative MG is found to exhibit equivalent macroscopic behaviour as the microscopic adaptive stochastic MG with infinitely many states.

Conclusions. It would be valuable to examine if methodological ways are possible to derive microscopic states from deliberative mesoscopic models.

Part III: Applications.

Pherome Model: Application to Traffic Conjestion Prediction. (Yasushi Ando, Osamu asutani, Hiroshi Sasaki, Hirotoshi Iwasaki, Yoshiaki Fukazawa, and Shinichi Honiden)

Abstract. Social insects perform complex tasks without top-down style control, by sensing and depositing chemical markers called "pheromone". We have examined applications of this pheromone paradigm towards intelligent transportation systems (ITS). Many of the current traffic management approaches require central processing with the usual risk for overload, bottlenecks and delays. Our work points towards a more decentralized approach that may overcome those risks. In this paper, a car is regarded as a social insect that deposits (electronic) pheromone on the road network. The pheromone represents density of traffic. We propose a method to predict traffic congestion of the immediate future through a pheromone mechanism without resorting to the use of a traffic control centre. We evaluate our method using a simulation based on real-world traffic data and the results indicate applicability to prediction of immediate future traffic congestion. Furthermore, we describe the relationship between pheromone parameters and accuracy of prediction.

Introduction. Technologies attempting to solve traffic congestion suffer from communication delays to drivers and that conditions can change in the time a route takes to traverse. Short term statistical prediction methods do not predict sharp or irregular fluctuations. A pheromone model is an alternative.

Pheromone. Basic Mechanisms. There are both positive and negative pheromones in nature (ie food vs alarm) **Characteristics.** Evaporation and propagation.

Model for Traffic. Electronic pheromone is deposited in inverse proportion to the cars speed. **Brueckners Model**. Is the basis. **Transition functions**. Aggregation, Evaporation and Propagation.

Evaluation. **Data**. Real data for 12000 cars for 2 hours. **Assumptions for simulation**. All cars have speed sensor and communication device, all roads have info servers for each direction and can communicate with neighbouring links. **Simulator**. Animation.

Result of Experiments. Prediction Accuracy. Favourable RMS (31.5) compared to moving average (41.0) and persistent prediction (35.2 @ 1 min : 45.3 @ 5min) models. **Relationship between Parameters and Prediction**. Accuracy increased with separate values for the aggregation parameter for commercial vs non-commercial vehicles. **Improvement of Shortest Route Search**. Pheromone (54.3%) was superior to 1 min persistent (36.2%)

Related Works. The launch of mobile agents from unmanned war-planes could avoid danger with alarm pheromone. Pheromone models are better at irregular time fluctuations and use less processing power than statistical and Bayesian models.

Conclusions and Future Works.

How Bee-Like Agents Support Cultural Heritage. (Marti Fabregas, Beatriz Lopez and Josep Masana)

Abstract. Elderly people are a great repository of knowledge, the majority of which has never been gathered by formal means. In this paper we introduce an application of multi-agent systems to support knowledge acquisition from this rich repository knowledge which is only available from elderly and experienced people. Our system provides the opportunity to complement different versions of the same knowledge produced in an extensive geographical and cultural region with the main objective of supporting Cultural Heritage. Users without much technological knowledge can search or leave information about some type of knowledge. Then, the system behaves like a swarm of bees, in this way the bee-like agents process the user contributions and the knowledge emerges from the system. Queen-like agents, honey-bee, drones and foragers have different roles inside the hive: looking for information resemblances, computing information confidence, checking the necessity of knowledge validation, and updating user's reliability. The system's feasibility has been tested on the specific area of ethnobotany, which concerns the ways in which specific societies name and classify plants.

Introduction. Current traditional knowledge is under threat of disappearing due to the dominance of modern information systems with from the younger generation and the alienation of the older generation. There is no information flow from the older generation to the Internet. This paper uses swarm technologies to gather, combine, validate and publish traditional knowledge. The initial application is in ethnobiology which is concerned with how different societies name and classify plants.

Swarm-like Multi-Agent System. Stigmergy is an agent interaction via the environment (such as pheromone). **En_C_Prou General View**. Is a development platform mainly concerned with the web interface and the hive. A user can consult or contribute to the system via a web interface. Each user has a reliability degree calculated based on the usefulness in the information that they have entered. The system searches for similarity between information entered by different agents. **Bee-Like Agents**. Each honey-comb contains knowledge related to a single concept in the application – in the case of ethnobotany, each honey comb is a plant. There are three kinds of bees/agents. **Queens** are responsible for the brood – ensuring that there are the right number of agents for the information in the system. There are two types of **Workers** which convert the information into knowledge (Drones and Honey bees). **Drones** fertilise the information in two ways: 1. by being aware of the quality and quantity of information in the system and influencing the activity to validate the information and 2. the drones look for similarity between different honey combs so that it might be co-referenced. **Honey Bees** are in charge of gathering information about a single concept. Finally, the **Foragers** are in charge of finding new information sources; both the reliability degree of each user and the recent contributions. **Coordination**. Communication between agents is via stigmergic mechanisms. Bee-keepers introduce new information about a subject (plants) via the web-page. Foragers report this new info (nectar) into an info board. Honey bees convert the information (nectar) to knowledge (honey) by searching immediate references provided by the forager and then computes confidence values. The drones find out if information required verification as a result of the new information and identifies any resemblances with the new data. Reports and nectar re the stigma used for agent coordination. **Hive**. Each individual concept is stored in a separate honey comb and each cell in the honeycomb represents an attribute (ie uses, leaves, seeds etc) as a true/false value with confidence based on the knowledge confidence on the attribute. Links are maintained between related concepts.

Emerging Knowledge. Knowledge emerges as the swarm acts with 4 methods**. Resemblance of Honeycombs.** The appropriate technique for testing similarity varies from domain to domain: in ethnobotany, two plant are similar based on the attributes

that they share**. Knowledge Confidence.** User contributions are combined into knowledge using the Mycin certainty factors method. The confidence contribution of a single user is computed based on their reliability and +/- if they agree with the attribute. The contributions are sorted by reliability and interactively combined, two at a time, least reliable first into a single value until the whole list becomes one value. (thus the most reliable contribution has significant weight). **The Degree of Necessity to Check Information**. If knowledge confidence is low and there are a many contributors then the need to check the information is high. **Users Reliability Update**. A contributors reliability is calculated to be high if the majority of their contributions agree with the majority of the contributions.

Experiments and Results. The Mycin, necessity to check information and the reliability update methods were all checked and verified as stable but with a low number of users.

Related Work. There are many applications of swarm techniques and a few using swarm techniques for information gathering (inc Wiki)

Conclusions. Principal is demonstrated **–** more large scale testing required.

Sift and Sort: Climbing the Semantic Pyramid. (H.V.D Parunak, Peter Weinstein, Paul Chiusano, and Sven Brueckner)

Abstract. Information processing operations in support of intelligence analysis are of two kinds. They may sift relevant data from a larger body, thus reducing its quantity, or sort that data, thus reducing its entropy. These two classes of operation typically alternate with one another, successively shrinking and organizing the available data to make it more accessible and understandable. We term the resulting construct, the "semantic pyramid." We sketch the general structure of this construct, and illustrate two adjacent layers of it that we have implemented in the Ant CAFE.

Introduction. Moving from raw data to finished policy recommendations has two fundamental problems: 1) data volume reduction and 2) each data item has multiple interpretations and must be related to other data to increase semantic content. This results in an iterative alternating sift and sort process.

The Semantic Pyramid. As intricacy of analysis increases, the volume must reduce to enable processing. **Computational complexity**. Complexity theory classifies various algorithms based on processing time and storage space requirements and Intelligence analysis must operate across the complete range of complexity. **Climbing the Pyramid**. Four broad levels can be identified – Filter, Index, Search, Analyse – but there are many sub-levels – all fundamentally sift/sort alternations.

Sorting Ant Sifting in Ant CAFÉ. The Ant CAFÉ Feedback Loop. There are two main components – the Analyst Modelling Environment (AME) and the Ant Hill. The AME comprises Community Models (representing the interests of a group of human analysts), Tasking models (representing a specific task), Analyst Models (represent the state of an analysts interests) and Query Models (reflecting the immediate info that an analyst wishes to access). The Ant Hill uses techniques inspired by insect societies. They are intrinsically distributed and decentralised,

dynamic processing (not query driven), answer queries with progressively better responses, and stochastic (driven by random sampling). **Dynamical Granularity Ant Clustering**. (DGAC) repeatedly travels down its hierarchical structure looking for nodes (sets of documents) that are internally cohesive, and moving these nodes to locations where they fit better. The resulting hierarchy reduces the complexity of subsequent retrieval. This approach avoids some limitations of previous hierarchical clustering such as the need for a centralised similarity matrix, restarts for new data and for new similarity functions, inability to dissolve nodes and a final answer at the end of processing. Clustering runs continuously to organise the data structure. **Information Foraging**. The similarity of documents under each node increases as one descends the tree from roots to leaves. A query generates a team of foragers which descend the tree to a leaf, evaluate the relevance of that document, deposits the relevance score at the leaf and propagates back up the tree combining with other relevance deposits from other foragers representing the same model diminishing in strength with each step. Subsequent foragers select their path stochastically based on the deposits.

Experimental Results. **The Effectiveness of Foraging**. Is comparable to random until relevance paths develop then the rate of accumulating relevance accelerates dramatically. **Effectiveness of Clustering**. There is a play-off between flexibility and perfect hierarchy. Better algorithms are under study.

Conclusion.

Agent-Based Control of Spatially Distributed Chemical Reactor Networks. (Eric Tatara, Micheal North, Cindy Hood, Fouad Teymour and Ali Cinar)

Abstract. Large-scale spatially distributed systems provide a unique and difficult control challenge because of their non-linearity, spatial distribution and generally high order. The control structure for these systems tend to be both discrete and distributed as well and contain discrete and continuous elements. A layered control structure interfaced with complex arrays of sensors and actuators provides a flexible supervision and control system that can deal with local and global challenges. An adaptive agent-based control structure is presented whereby local control objectives may be changed in order to achieve the global control objective. Information is shared through a global knowledge environment that promotes the distribution of ideas through reinforcement. The performance of the agent-based control approach is illustrated in a case study where the interaction front between two competing autocatalytic species is moved from one spatial configuration to another. The multi-agent control system is able to effectively explore the parameter space of the network and intelligently manipulate the network flow rates such that the desired spatial distribution of species is achieved.

Introduction. There exist chemical reactors which comprise multiple interconnected vessels that can be adjusted to deliver various different products on a continuous basis. However, the centralised modelling and control of these systems with mathematically complex models is challenging. These highly non-linear systems might be better controlled with a multi-agent approach.

Agent-Based Control Framework. **Design process**. In industrial control there are process responsibilities and safety responsibilities. The model to control the chemical reactors includes observation agents for each reactor, actuator agents to control flow between vessels, decision agents with local control objectives and arbitrator agents which serve as a

communication and dispute resolution channel between decision agents and they issue commands to a few or very many actuator agents. The top level supervision agents set the operating conditions and objectives for the whole network. **Global Knowledge Environment**. The non-linearity of the chemical reactors makes it difficult to predict the outcome of changes. The decision agents are provided with a set of heuristics in the form of rules to guide their decisions away from known instabilities and the agents are allowed to make small exploratory changes. There is local flow of information between decision agents via arbitration agents but there is also a global workspace which agents can read/write to asynchronously. In this way successful strategies can be shared, tested widely, and strengthened if broadly successful. Time-critical, first response actions remain with traditional controllers and are strictly outside the domain of higher level agents.

Network Model. The complexity of a reactor system grows geometrically with the number of species and the number of vessels so analytical solutions for more than 3 vessels and 2 species become intractable.

Case Study: Product Grade Transition in a Chemical Reactor Network. A 7x7 grid of reactor vessels with three autocatalytic species. One can produce multiple grades of product by altering the network connectivity and spacial configuration of the species as a result. The dominant species in a reactor generally have an order of magnitude greater concentration than the other species. The transition from one grade to another is very difficult for traditional controllers but a multi-agent system can handle it well. **RePast implementation**. RePast is a Java based toolkit offering event scheduling and visualisation tools.

Conclusion. The multi-agent approach handles this complex situation well. Wow!

A Study of System Nervousness in Multi-Agent Manufacturing Control System. (Hadeli, Paul Valckenears, Paul Versraete, Bart Saint Germain, and Hendrick Brussel)

Abstract. This paper discusses a study on system nervousness in a multi-agent manufacturing control system. Manufacturing control systems, built along this approach, are able to generate short-term forecasts that predict both resource loads and order routings. These forecasts become known throughout the multi-agent system with some time delay. If the agents make their decisions based on these forecasts, proper measures need to be taken to account for these delays, especially when disturbances (rush orders, machine breakdowns) occur. If agents react too eagerly and swiftly, the forecasts become unreliable. This paper studies this issue and the measures in the control system design that address the problem. More precisely, the agents behave in a socially acceptable manner that reconciles adaptation to changed circumstances with predictability.

Introduction. The art here is to design and implement a proper balance or so called social acceptable behaviour.

The Design of the Multi-Agent Manufacturing Control System. Product agents (how to make a product of sufficient quality) Order Agents (the product state and logistics) and Resource Agents (a physical entity in a manufacturing system). **Coordination between Agents**. All indirect communication via pheromone implementing and **Exploration mechanism** (ensure that all order find a way thru the system) and an **Intention Propagation Mechanism** (Registers an intention to use each appropriate resource – and is allocated a time slot – but the intention evaporates). **The Reinforcement of the Forecast Information**. The Intention agent places an intention then the Resource agent which self forecasts its own load then the Exploration agent forecasts order performance and then the Order agent updates it intentions. **The Forecast as an Emergent Property**. The solution list for each order agent and a resource load forecast at every resource agent

appear despite none of the agents being directly aware of each others activity or the global objective.

System Nervousness. Accuracy and reliability are the most important forecasting properties. System nervousness is the degree to which elements in the system react on the internal and external stimuli. **The Appearance of System Nervousness**. Caused by the inherent delays in asynchronous communication. Negative impacts to both reliability and accuracy. **Mechanisms to Have Those Agents Live in Harmonious Ways**. When order agent encounter a better solution they must probabilistically wait for a while and observe the outcomes of the disturbance before reacting. **Stabilise the System: socially acceptable behaviour**. Order agents must not change their intention without sufficient reason. Ie 1) ensure that the new solution is better to the old one 2) apply a probabilistic method each time an order agent is willing to change intention and 3) limit the frequency of change intentions. The Detection of Nervousness. Three indicators 1) frequencies of intention changes 2) performance value of each order agent and 3) Fluctuation of resource load.

Implementation and Experimental Results. It works for a simple example.

Discussion.

Key Insights by paper ……..

- **T-Man: Gossip-Based Overlay Topology Management.** That topology can be constructed locally and incrementally and that topology has a critical influence on efficiency
- **Basic Approach to Emergent Programming.** That cooperation between software components and between the system and the environment can be used to guide system adaption in the absence of a heuristic.
- **Emergent Timetabling by Cooperative Self-Organisation.** That software agents can negotiate and communicate on behalf of a human (who provides broad policy and constraints) with a large number of other agents in a continuous and high bandwidth manner.
- **Self-Adaption and Dynamic Environment Experiments with Evolvable Virtual Machines.** That useful hierarchical software can be constructed dynamically from basic software components.
- **Evolution at the Next Level in a Peer-to-peer Network.** That Selfish agents can organise themselves into clusters that can jointly perform some useful function.
- **Exchange Values and Self-regulation of Exchanges in Multi-agent Systems.** That mathematicians and accountants should not be allowed to breed outside of captivity.
- **A New Protocol to Share Critical Resources by Self-organized Coordination.** That a balance can be struck between centralised and decentralised modes by elaborating agent contracts locally and validating them centrally.
- **Information-Driven Phase Changes in Multi-agent Coordination.** Communication delays between agents can introduce performance phase changes – but they can be managed locally by an agent optimising performance while avoiding the phase change.
- **Self Organising Applications Using Lightweight Agents.** That development architectures like DIET are available to model multi-agent systems in a scalable and flexible manner.
- **Solving Dynamic Distributed Constraint Satisfaction Problems with a Modified Weak-Commitment Search Algorythm.** Dunno!
- **Development of Self-Organising Emergent Applications with Simulation-Based Numerical Analysis.** Simulation of multi-agent systems can be combined with numerical analysis techniques to explore and understand the behaviour of a multi-agent systems without having to determine global equations and complete mathematical proofs.
- **On the Role of Simulations in Engineering Self-organising MAS.** Both π-calculus and the human immune system are worth following up and understanding.
- **Mesoscopic Modelling of Emergent Behaviour.** Mesoscopic modelling can provide a valuable middle-ground between the overwhelming detail/processing of multiple-agents and macroscopic Rate Equations which abstract away the critical time based dynamics of the system.
- **Pheromone Model: Application to Traffic Congestion Prediction.** Multi-agent techniques are well matched to multi-agent applications. The authors seem either limited in their intellectual and innovative sophistication (junior or Japanese or both???) or alternatively just applaudably pragmatic.
- **How Bee-Like Agents Support Cultural Heritage.** The (overly?) tight analogy with a social insect model is unsettling but does introduce an very interesting approach to knowledge capture and management. This technique is interesting in that it may well be sensitive to and propagate human memes.
- **Sift and Sort: Climbing the Semantic Pyramid.** The agent model provides for robust continuous processing. This is another very interesting approach to structuring knowledge in an “organic” manner.
- **Agent-Based Control of Spatially Distributed Chemical Reactor Networks.** The Agent model can offer some

apparently stark simplicity to intractable problems.

- **A Study of System Nervousness in Multi-Agent Manufacturing Control System.** Communication and negotiation bandwidth of humans can be increased by using agents as proxies. Job Shop schedulers still seem to operate without considering the profitability of the operation. A biding process will deliver superior financial results and customer satisfaction.