Decentralised Autonomic Computing: Analysing Self-Organising Emergent Behaviour using Advanced Numerical Methods

Tom De Wolf, Giovanni Samaey, Tom Holvoet and Dirk Roose
Department of Computer Science, KULeuven
Celestijnenlaan 200A, 3001 Leuven, Belgium
{Tom.DeWolf, Giovanni.Samaey, Tom.Holvoet, Dirk.Roose}@cs.kuleuven.ac.be

Abstract

When designing decentralised autonomic computing systems, a fundamental engineering issue is to assess system-wide behaviour. Such decentralised systems are characterised by the lack of global control, typically consist of autonomous cooperating entities, and often rely on self-organised emergent behaviour to achieve the requirements.

A well-founded and practically feasible approach to study overall system behaviour is a prerequisite for successful deployment. On one hand, formal proofs of correct behaviour and even predictions of the exact system-wide behaviour are practically infeasible due to the complex, dynamic, and often non-deterministic nature of self-organising emergent systems. On the other hand, simple simulations give no convincing arguments for guaranteeing system-wide properties. We describe an alternative approach that allows to analyse and assess trends in system-wide behaviour, based on so-called "equation-free" macroscopic analysis. This technique yields more reliable results about the system-wide behaviour, compared to mere observation of simulation results, at an affordable computational cost. Numerical algorithms act at the system-wide level and steer the simulations. This allows to limit the amount of simulations considerably.

We illustrate the approach by studying a particular system-wide property of a decentralised control system for Automated Guided Vehicles and we outline a road map towards a general methodology for studying decentralised autonomic computing systems.

1. Introduction

In an industrial research project [5], we examine the possibilities of a *decentralised autonomic computing* solution for an Automated Guided Vehicle (AGV) warehouse transportation system. A group of AGVs has to transport incoming loads from pick up locations to specific destinations in the warehouse. Experience has shown that the current centralised system has problems with scalability because it cannot handle many AGVs efficiently. Therefore, a centralised solution is not suitable. Also, the current system is not flexible, i.e. it cannot handle frequent changes in the transportation problem and it needs to be customised and optimised each time the system is deployed in another warehouse.

To overcome these difficulties, there have been efforts to design hierarchical control systems (e.g. [6]), in which the goal is to balance control and decentralisation. Our approach, on the other hand, aims at solutions which are completely decentralised to obtain maximal flexibility and scalability. To handle frequent changes, we want the system to adapt itself to each different situation, i.e. to deal with its complexity autonomously – the term autonomic computing has been coined for this system behaviour [9, 10]. To this end, we construct such decentralised autonomic computing systems as a group of interacting autonomous entities that are expected to cooperate. A coherent system-wide behaviour is achieved autonomously using only local interactions, local activities of the individual entities, and locally obtained information. Such a system is called a *self-organising emergent system* [4]. Self-organisation is achieved when the behaviour is maintained adaptively without external control. A system exhibits emergence when there is coherent system-wide or macroscopic behaviour that dynamically arises from the local interactions between the individual entities at the microscopic level. The individual entities are not explicitly aware of the resulting macroscopic behaviour; they only follow their local rules. In the rest of this paper, we will use the terms macroscopic and system-wide interchangeably.

For decentralised autonomic computing, systems that are characterised by both self-organisation and emergence are promising. Such systems promise to be scalable, robust, stable, efficient, and to exhibit low-latency [1]. However, a fundamental problem when building self-organising emergent systems is the lack of an analysis approach that allows to systematically study and asses the system-wide behaviour. This paper aims to address that problem.

Guaranteeing desired system-wide behaviour is an industrial need and requires modelling and analysing the system in a systematic and scientific way. To date, research in autonomic computing (e.g. autonomous robotics [3, 2]) has mainly used so-called individual-based models (e.g. agent-based models), which are mostly analysed by performing a large number of simulation experiments and observing statistical results. At the same time, there is a tendency, for example in the autonomous robotics community [13], to analyse the system behaviour more scientifically (e.g. applying chaos theory).

Traditionally, scientific analysis is done by deriving equation-based models, and using numerical methods to obtain more quantitative results about the system-wide behaviour. However, when it comes to complex systems with multiple autonomously interacting entities, it is often practically infeasible to construct equation-based models that accurately describe the system behaviour. The popular view (e.g. [15]) is that a choice has to be made between an individual-based model, which allows to realistically model complex systems, or an equation-based model, which allows sophisticated numerical algorithms to be employed. Therefore, a gap remains between rigourous scientific analysis and realistic (individual-based) models.

In this paper, we use a recent development in the scientific computing community to bridge this gap: "equation-free" macroscopic analysis [11, 12]. The general idea is that it is possible to use algorithms that are designed for equation-based models, even if the only available model is individual-based. Whenever the numerical algorithms need to evaluate the equation, this evaluation is replaced on the fly by a simulation using the individual-based model. Therefore, the algorithm decides which simulations (e.g. initial conditions, duration, time step, etc.), and how many, are needed to obtain the desired result. This way, more reliable results are obtained about the average system-wide behaviour, compared to mere observation of simulation results, while reducing the computational effort drastically. This approach could become a bridge between complex emergent systems (e.g. decentralised autonomic computing) and traditional numerical analysis.

In section 2, general issues on guarantees for self-organising emergent systems are discussed. Section 3 describes the AGV case study used throughout the paper, a possible self-organising emergent solution, and an example system-wide requirement for which guarantees are needed. In section 4, the "equation free" macroscopic analysis approach and a road map outlining how to apply it are described. Section 5 applies the approach to the AGV case and discusses some results that validate our approach. Finally, we conclude and discuss future work.

2. System-wide Guarantees for Self-Organising Emergent Systems

Before outlining our analysis approach, we need to define the results expected from the analysis of self-organising emergent systems. Self-organising emergent systems promise to be scalable, robust, stable, efficient, and to exhibit low-latency [1], but also behave non-deterministically. Even if the desired system-wide behaviour is achieved, the exact evolution is not predictable [1]. However, self-organising emergent systems exhibit *trends* that are predictable. We define a trend as an average (system-wide) behaviour (i.e. average over a number of runs). Due to the dynamics of self-organising emergent systems, robustness is preferred to an optimal system-wide behaviour. Optimality can only be achieved when the operational conditions remain rather static. But such a static situation will never be reached in the presence of frequent changes. Preferring robustness above optimality implies that modest temporal deviations from the desired behaviour are allowed as long as the desired behaviour is maintained in a trend, i.e. in the average behaviour. Normal temporal deviations are often necessary to explore the space of possibilities, and to counteract the frequent system changes. Therefore, the results we expect from an analysis of self-organising emergent systems are statements that assure the desired evolution of the average system-wide behaviour. We define these statements that guarantee desired trends as *system-wide guarantees*.

The goal of our analysis approach is to systematically acquire system-wide guarantees. A systematic approach often implies that one divides the problem into manageable sub-problems. The system-wide behaviour of a self-organising emergent system typically consists of a number of *system-wide properties* that have to be maintained. Therefore, first each of those system-wide properties are considered separately in our analysis approach. Secondly, we focus on the fact that a system-wide guarantee that holds in all possible settings in which a system operates is difficult, if not impossible to give. Therefore our analysis approach considers such settings separately as multiple delimited scenarios, i.e. steady scenarios. A *steady scenario* is defined as a setting for the system in which certain assumptions are made about the possible dynamic changes and the frequency of change. For example, one can consider steady scenarios where the system has a high utilisation load, a low utilisation load, or a scenario where there is a frequent oscillation between high and low utilisation loads. As a consequence, a complete analysis of the system-wide behaviour consists of multiple system-wide guarantees, each for a specific steady scenario and for a specific system-wide property. In this paper we illustrate the analysis approach for one system-wide property and one steady scenario.

3. The Case Study: Automated Guided Vehicles

In an industrial research project [5], our group develops self-organising emergent solutions for AGV warehouse transportation systems. Experience revealed an industrial need for guarantees about system-wide properties. Therefore, the AGV case study is used throughout the paper.

3.1. Problem Description

The automated industrial transport system, that we consider, uses multiple transport vehicles. Such a vehicle is called an AGV and is guided by a computer system (on the AGV itself or elsewhere). The vehicles get their energy from a battery and they move packets (i.e. loads, materials, goods and/or products) in a warehouse. An AGV is capable of a limited number of local activities: move, pick packet, and drop packet. The goal of the system is to transport the incoming packets to their destination in an optimal manner.

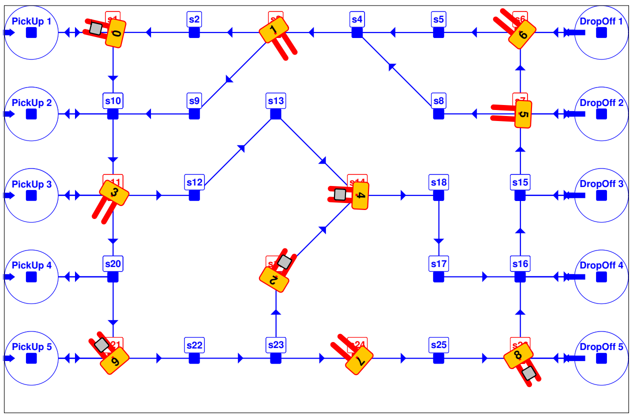


Figure 1. The AGV Simulator.

Because it is too expensive and often infeasible to apply the analysis approach on a real AGV system, we developed a realistic simulator[1] for AGV systems that allows to execute the simulations needed in the analysis approach described in section 4. The screen-shot in figure 1 illustrates the problem setting described above: the locations where packets must be picked up are located at the left of the figure, the destinations are located at the right, and the network in-between consists of stations and connecting segments on which the AGVs can move. Segments are unidirectional, i.e. an AGV can follow a segment in one direction. A bidirectional segment can be constructed with two overlapping unidirectional segments. This facilitates the construction of maps and minimises the possible collision regions. Packets are depicted as rectangular boxes and some of the AGVs shown hold a packet, others do not.

[1] http://www.cs.kuleuven.be/˜distrinet/taskforces/agentwise/agvsimulator/

3.2. A Self-Organising Emergent Solution

The AGV problem is a dynamic problem with non-deterministic features (e.g. packets can arrive at any moment, AGVs can fail, obstacles can appear). The project with our industrial partner [5] has shown that a solution using a central server to control all AGVs cannot handle these frequent changes efficiently. The central server has to constantly monitor the warehouse and each AGV to detect changes and react to them by steering each AGV. Because AGVs are moving constantly, reacting on changes has to occur instantaneously. Therefore, the central server becomes a bottleneck in the presence of frequent changes. As a consequence, such a central solution is not scalable and can only handle a limited number of AGVs. Also, the system is not flexible when it has to be deployed, i.e. the system needs to be customised and optimised each time it is deployed in another warehouse. Therefore, in the context of decentralised autonomic computing for larger and dynamic AGV systems, a self-organising emergent solution in which the AGVs adapt to the changing situations themselves by only using locally obtained information, local interactions, and local activity is promising. In the rest of this section, a self-organising emergent solution is described, which is only one of the large number of possible solutions. We do not state that this is the perfect solution, it is only a test case.

Each AGV needs information to decide autonomously where to move to on the network in order to reach an incoming packet or a destination to drop a packet. This information should be made available locally. To achieve this, we use so-called artificial pheromones that propagate through the network. Such pheromones are pieces of data that are present in the network and, just like real pheromones, they evaporate over time (i.e. older data disappears gradually). When a location where packets arrive contains a packet, that location will actively propagate pheromones through the network. The pheromones are propagated between neighbouring stations (i.e. only local interaction) in the direction opposite to the direction in which the AGVs can move. A pheromone includes information, such as the priority of the packet, that is important for the AGVs to choose a packet. Also, the distance from the station on which the pheromone is located to the location of the packet is important information. Therefore, when a pheromone propagates through a segment, the cost of travelling over that segment is added to the pheromone. This way, the distance is calculated during the propagation process. All the pheromones in the network form a gradient map for the AGVs to follow: on each station, the pheromones indicate which outgoing segment is the “best” to follow, i.e. the AGV chooses the segment in the direction of the closest packet and with the highest priority. Decisions are made one segment at a time, i.e. there is no planning of a fixed path from source to destination.

When an AGV finally reaches a packet, it picks it up, and follows another type of pheromone gradient, one that is propagated from the destination locations. An AGV chooses a segment to follow so that it can reach the destination of the packet it is holding via the shortest path. This shortest path can be found because again the distance is calculated during propagation of the pheromones and added to the pheromone information at each station.

Note that this solution assumes the presence of infrastructure on the stations, that can contain pheromone information and can actively propagate incoming pheromones towards neighbouring stations. Also, the communication bandwidth between the stations should allow this propagation through the entire network. In this paper we assume that this infrastructure is present. Other solutions that do not need this infrastructure can also be analysed with the approach described in section 4.

3.3. Desired Guarantees

There are multiple requirements in an AGV system for which guarantees are needed. To motivate our test case, we consider an essential characteristic of self-organisation which states that certain system-wide properties should be maintained. For example, to achieve a constant throughput of packets, the system should ensure that on average there is always a fraction of the AGVs available to transport new incoming packets, while another fraction is busy transporting packets. In the AGV case, with the lay-out of figure 1, the system-wide property that reflects this requirement is the distribution of the total number of AGVs between two situations: AGVs moving towards the pick up locations without a packet and AGVs holding a packet and moving towards the destination locations. An equal distribution needs to be maintained. Considering the warehouse lay-out of figure 1, we can define two zones in which a certain situation is desired (see figure 2):

- In *Zone A* the AGVs should be moving towards the pick up locations and they should not hold a packet.
- In *Zone B* the AGVs should be moving towards the drop off locations and they should hold a packet.

Note that there is an overlap between both zones at the left and right of the factory lay-out. Because each AGV can only be counted to be in one situation, the desired situations are used as a guide: an AGV in these overlap regions is counted for zone A if the AGV does not hold a packet, and for zone B if the AGV holds a packet. Note that also AGVs in zone A could be holding a packet, as well as there

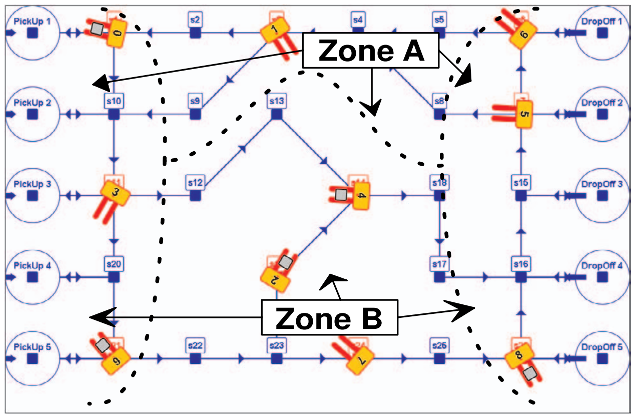


Figure 2. AGV Distribution between zones.

may be AGVs in zone B that aren't. These situations are undesirable, so we wish to avoid them.

Therefore, two guarantees are needed: an equal distribution between the two desired situations and a small number of AGVs in the undesirable situations. Because we prefer robustness to optimality, it is allowed that at certain points in time there is an unequal distribution and that some AGVs are in an undesirable situation, as long as trends in the system-wide behaviour guarantee the two requirements.

The analysis goal is to check if these requirements are guaranteed for a specific solution and a specific steady scenario. In this paper, we limit our analysis to the scenario with a *maximal system load*. Consider the AGV case with the lay-out shown in figure 1, with 10 AGVs, and with the assumption that the pick up locations always contain a packet (i.e. packets that are picked up are replaced immediately). Furthermore, we assume that no other dynamic events can occur (e.g. AGVs failing, AGVs added to the system, no obstacles will appear, etc.). It is generally necessary and interesting to investigate the effect of changes in this setting on the dynamics of the system, but this is outside the scope of the current paper.

To make the required guarantees concrete we quantify them mathematically as follows. Two measures are considered. A first measure (E) represents the distribution of AGVs between the two desired situations and a second measure (S) represents the number of AGVs in the desired situations. These system-wide measures can be calculated from the number of AGVs in each situation:

NB_{AP}**:** AGVs in zone A, holding a packet.

NB_{ANP}**:** AGVs in zone A, not holding a packet.

NB_{BP}**:** AGVs in zone B, holding a packet.

NB_{BNP}**:** AGVs in zone B, not holding a packet.

We define the measure S to be the normalised sum of AGVs in the desired situations. This yields:

$$S = \frac{NB_{ANP} + NB_{BP}}{NB} \tag{1}$$

In this equation, NB is the total number of AGVs. According to the requirements, S needs to be high.

From [8, 14, 2] we know that (spatial) entropy, defined as

$$E = \frac{-\sum\limits_{1 \leq i \leq N} p_i * \log p_i}{\log N}, \tag{2}$$

is suitable to reflect the spatial distribution of entities between different states. Here p_i is the probability that state i occurs and $\sum_{1 \leq i \leq N} p_i = 1$. Dividing by $\log N$ normalises E to be between 0 and 1. Entropy is high (close to 1) when the considered states have an equal probability to occur, and low (close to 0) when only a few of the states have a high probability to occur.

To apply this measure to the distribution of AGVs, the different states for the entropy measure are defined as the desired situations for the AGVs. Consider the AGVs that are already distributed between the desired situations. The probability for such an AGV to be in one specific desired situation at one moment in time is used as the probability in the entropy equation. Then we can define the entropy measure for the AGV example as:

$$E = \frac{-p_{ANP} \log p_{ANP} - p_{BP} \log p_{BP}}{\log 2} \tag{3}$$

with

$$p_i = \frac{NB_i}{NB_{ANP} + NB_{BP}} (i \in \{ANP, BP\}) \tag{4}$$

and

$$p_{ANP} + p_{BP} = 1 \tag{5}$$

According to the requirements, E needs to be high because then there are approximately as many packet-carrying AGVs in zone B as there are packet-less AGVs in zone A.

4. Analysis with Advanced Numerical Methods

4.1 Analytic versus Experimental

To guarantee system-wide requirements, typically two approaches can be used:

- a formal proof: a (mathematical) formal model of the system is constructed and a proof is found analytically;
- an experimental simulation: simulations of the system are performed and the analysis results are obtained by observation of measurements.

The first approach offers hard guarantees, but in the context of decentralised autonomic computing, where complex and dynamical systems are considered, the model and proof are complex or even infeasible. The second approach is also infeasible because simulations of decentralised autonomic systems are expensive and many (long) simulations are necessary to make reliable statements.

This paper uses an *"equation-free" macroscopic analysis* approach [11, 12], combining numerical analysis algorithms and realistic simulation-based modelling. Traditionally, numerical analysis is applied to equation-based models. The system-wide behaviour is modelled by a (macroscopic) evolution equation. Numerical algorithms obtain the desired results, i.e. quantitative statements about system-wide properties. The results are more reliable than mere observation of simulations. However, in complex dynamical systems, deriving a macroscopic equation from a model for the dynamics of each autonomous entity (i.e. the microscopic dynamics) is often not possible, unless the microscopic model is very simple [15].

The "equation-free" approach resolves this issue by replacing the equation-based model by a realistic simulation model. Simulation measurements are analysed, but, in contrast to mere observation, the numerical analysis algorithms acquire the results themselves by steering the simulation process towards the algorithm's goal. The advantage is that the results are calculated on the fly and only those simulations are executed that are actually needed to obtain a specific result. The results are therefore of equal scientific value as the equation-based analysis, while reducing the computational effort drastically compared to mere simulations. And compared to formal proofs, this method is feasible for complex and dynamical systems. Note that numerical algorithms assume a rather smooth behaviour (i.e. a rather continuous evolution over time). As described in section 2, for self-organising emergent systems, we focus on analysing trends which are expected to evolve gradually.

4.2 "Equation-Free" macroscopic analysis

The "equation-free" macroscopic analysis approach was proposed in [11, 12]. The observation is that most numerical methods have no explicit need for the macroscopic evolution equation; all they need is a routine that *evaluates* these equations for a given value of the macroscopic (measurement, observation) variables. Once we replace these evaluations with a suitable *estimation*, all these algorithms can be readily applied. To this end, the following procedure, called a *"macroscopic time-stepper"* [11], is performed. From the given values of the observation variables, we generate (a set of) consistent initial conditions for the microscopic AGV system (*initialisation operator*); we then *simulate* using the microscopic simulation code, and we fi-

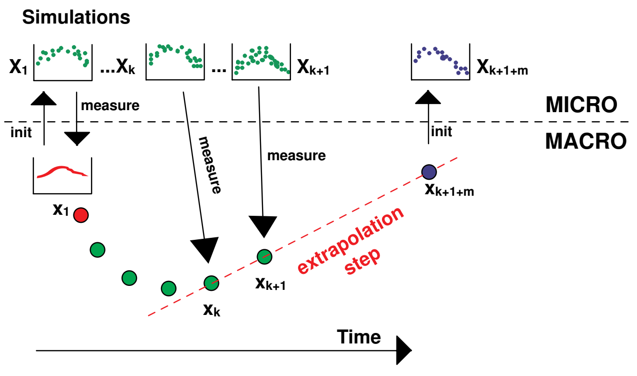


Figure 3. Equation-Free Accelerated Simulation over Time

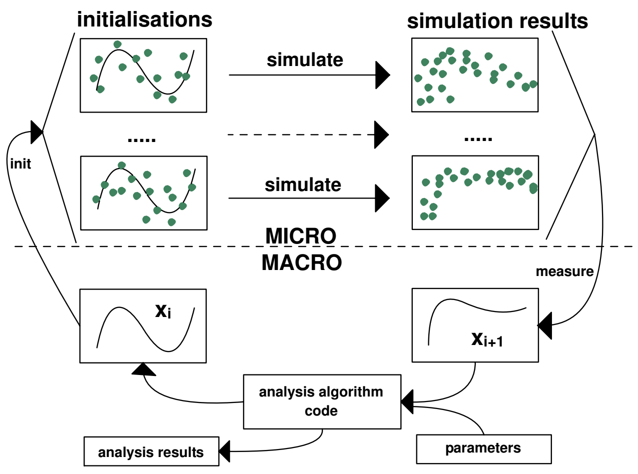


Figure 4. Equation-Free Accelerated Simulation guided by the analysis algorithm

nally *measure* the new values of the observation variables. As such, the equation and its evaluation are replaced by the simulation code and therefore the proposed method constitutes a bridge between classical numerical analysis and microscopic (e.g. agent-based) simulation.

The described macroscopic time-stepper can be used to accelerate the simulations significantly. Figure 3 illustrates the basic idea. First an initial value x_1 for the measured variable is chosen by the analysis algorithm. Using this value, a simulation is initialised with microscopic value(s) X_1 (i.e. `init` in figure 3), consistent with x_1, and executed for a certain duration. At some points in time, one measures the new value for the variable that is analysed (i.e. `measure` in figure 3). This is repeated a number of times, such that enough successive values x_k are available for the analysis algorithm to make the extrapolation step that skips m time steps of simulation. This means that a new value x_{k+1+m} is estimated with extrapolation, using a number of measured values (x_k and x_{k+1} in figure 3). Starting from this new value the process is repeated by initialising with microscopic value(s) X_{k+1+m}. This acceleration over time is called a *projective integration algorithm*. For a detailed stability analysis, which allows to answer questions such as how many observation values x_k are needed to skip m steps, we refer to [7]. Note that if the observation variables behave stochastically, k may need to be increased to avoid amplification of the statistical noise.

Simulations can also be accelerated in other ways. Suppose we want to obtain the steady state behaviour, so we look for values of the observation variables that remain constant as time evolves. Instead of computing the time evolution until the system reaches this steady state, one can use numerical methods that determine steady states in a more direct and efficient way. Denote the macroscopic time-stepper starting from an initial condition x_i and performing a simulation for a fixed duration by $\Phi(x_i)$. Denote the value of the observation variable after this simulation as $x_{i+1} = \Phi(x_i)$. The steady state x^* can then be computed by solving the equation

$$\Phi(x^*) - x^* = 0 \tag{6}$$

numerically, e.g. by Newton's algorithm [16]. This iterative method generates consecutive approximations for the steady state, which typically converge much faster to the steady state than computing the time evolution through consecutive time steps.

Figure 4 summarises a more general view on the approach. First we supply to the analysis algorithm (e.g. Newton's algorithm) initial values (x_i) for all the system-wide variables under study. Then the `init` operator initialises a number of simulations accordingly, i.e. measuring the system-wide variables after initialisation should equal the given initial values. Because there are multiple ways to initialise a simulation for the same value of a system-wide variable, we need to initialise these "degrees of freedom" randomly in multiple initialisations. Then the simulations are executed for a predetermined duration. This duration can be fixed or supplied by analysis algorithm. At the end, one measures the averages of the values of the system-wide variables (x_{i+1}), which are given to the analysis algorithm as a result. Then the analysis algorithm processes the new results to obtain the next initial values (e.g. Newton: a guess for the steady state). The cycle is repeated until the analysis algorithm has reached its goal (e.g. Newton: the steady state). The algorithm decides on the configuration of the simulations such as the initial conditions and the duration of the simulations. Following the same principle, e.g. in the presence of parameters, more general tasks, such as parameter optimisation or control can be performed as well [11].

As such, a more focussed and accelerated simulation-based analysis approach, guided by the analysis algorithm's goal, can be used to obtain more reliable results than mere observation of simulation results.

4.3 Road Map for Equation-Free Macroscopic Analysis

There are a number of steps that have to be performed for the equation-free approach to work. In this section we give an overview of those steps:

1. *Identification of observation variables*. In the context of self-organising emergent systems, the challenge is to find variables which measure the system-wide properties under study. The properties to study are derived from the system requirements. In other words, first a *quantification of the required emergent properties* in terms of measurable variables is needed.

2. *Related system-wide variables*. Are there other variables for system-wide properties that influence the system-wide property under study? If so, then these variables have to be incorporated into the analysis process. Otherwise, the evolution of the system is not correctly and completely represented and analysed. The underlying assumption of the equation-free analysis approach is that a set of measurable variables can be found that offer an adequate description of the macroscopic system dynamics. This means that the chosen measurement variables are indeed the variables that would appear in the macroscopic evolution equation. In other words, when choosing system-wide variables, we strive to reach a set of variables that reflect the macroscopic degrees of freedom.

3. *Micro-variables*. For each system-wide variable, the corresponding variables of the simulation at the level of the individual entity in the system, that influence this system-wide variable need to be identified.

4. *Measurement operator*. We need to know how to calculate the system-wide variables from the micro-variables in the simulation.

5. *Initialisation operator*. We need to define an operator that allows to initialise the microscopic variables of the simulation or multiple simulations to reflect the given values for the system-wide variables. When there are degrees of freedom in the initialisation, these are initialised randomly and multiple simulation are considered to average this randomness.

6. Check the micro-macro *scale separation*. For the equation-free approach to work and to be efficient, the microscopic variables should evolve on a much faster timescale than the macroscopic variables that determine the macroscopic evolution. Thus, changes in the state of the individual entities in the system should be fast compared to the evolution of the overall system behaviour. If this would not be the case, then any error introduced by our initialisation procedure could significantly influence the results and hence create errors.

7. Define the different *steady scenarios* to analyse. As explained earlier, we need to consider the possible steady scenarios separately to allow systematic and useful analysis. This step identifies system parameters that need to be modified in order to cover the range of possible operational conditions for the system.

8. Choose an *analysis algorithm*. Once the previous steps are successfully completed, an analysis algorithm can be chosen. Depending on the kind of data and the goals, e.g. finding steady state behaviour, optimising values of system parameters, controlling the operational modus, one selects a suitable analysis algorithm (e.g. projective integration, Newton's method).

4.4. Underlying Assumptions

An important issue is if the analysis approach is generally applicable to decentralised autonomic computing systems. A thorough study of this issue is outside the scope of this paper. However, at this moment, it seems that only a separation of time scales between the macroscopic and microscopic evolution is needed (see step 6 in road map). When this assumption is valid, it implies that re-initialising and restarting a simulation at arbitrary points produces results that are comparable to a single long simulation. Indeed, in the presence of time-scale separation, initialisation errors disappear quickly compared to the evolution of the system-wide behaviour. For a review of the mathematical principles that support this intuitive explanation, we refer to [12].

The main *practical* issue in applying the road map to an arbitrary self-organising emergent system is to find a set of macroscopic variables that correctly reflect the evolution of the system-wide properties. This is far from trivial because an open issue for emergent systems is to understand how the macroscopic behaviour is accomplished by the individual entities. Finding macroscopic variables, and how they are measured from the microscopic state, requires detailed knowledge of both the system under study and the system-wide requirements. Besides a measurement operator, an initialisation operator is required that initialises a microscopic simulation consistently with a macroscopic variable. However, when these issues are resolved, a successful application of the road map also results in new insights on how the macroscopic behaviour is related to the behaviour of the individual entities. Moreover, in section 5.3, we show how to check if a set of macro-variables completely captures the evolution of the system-wide property.

5. Equation-Free Macroscopic Analysis of the AGV System

5.1. Preliminary simulations

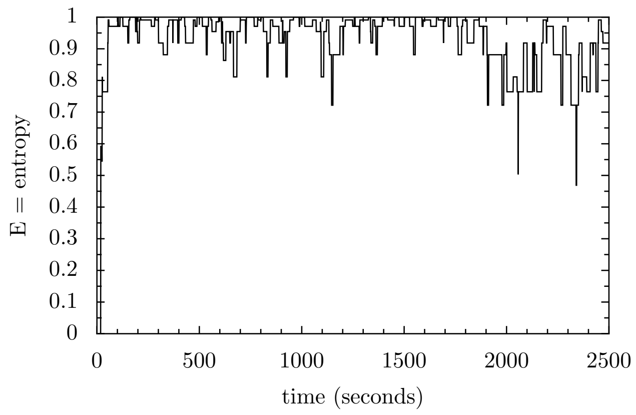


Figure 5. E **in one AGV simulation run.**

We first have to define which system-wide variables are suited for macroscopic analysis (step 1 of road map). This requires some preliminary simulations. We consider the AGV case as described in section 3. The initial condition is given by an equal distribution of AGVs over the two zones, and none is holding a packet. Since there are 10 AGVs in total, the initial distribution is given by $NB_{ANP} = NB_{BNP} = 5$ and $NB_{AP} = NB_{BP} = 0$. This results in an entropy $E = 0$, and a normalised sum $S = 0.5$. (See formulas (1) and (3).) We perform a simulation over 2500 time steps and measure E and S at each time step. The simulator is written such that one time step corresponds to a logical duration of 1 second. The result for E is shown in figure 5. We see a clear non-deterministic effect in this variable, which is also true for S. Moreover, there are discontinuous jumps, because the variable can only take a discrete set of values.

However, if we perform the same simulation a large number of times and average the results, we see that this randomness smoothes out. Consider the same system, and the same initial condition. Now, we perform $N = 1000$ simulation runs instead of just one, and we report on the values averaged over all runs, i.e. E_{avg} and S_{avg} in figures 6(a) and 6(b). It is clear that, despite the non-deterministic nature of one simulation, the averaged behaviour evolves rather smoothly. Therefore, it is possible to make statements about the average long-term values of entropy and normalised sum with numerical methods. In the rest of our analysis, the average entropy E_{avg} and normalised sum S_{avg} are the system-wide observation variables.

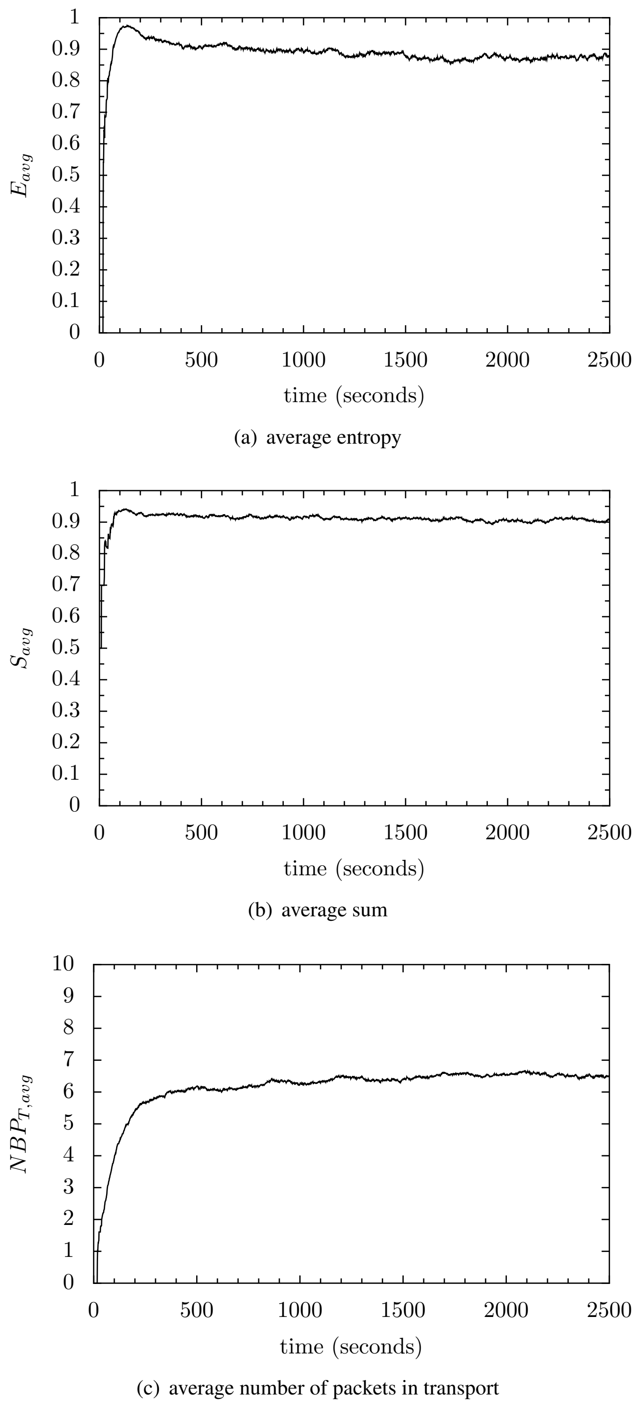


Figure 6. E_{avg}, S_{avg}, **and** $NBP_{T,avg}$ **of a set of** $N = 1000$ **runs of the AGV case as a function of time.**

5.2. Macroscopic time-stepper

Once we have identified the system-wide variables of interest, here E_{avg} and S_{avg}, we need to determine other variables that might influence their values (step 2 from road map). To achieve a one-to-one correspondence between E, S and the variables NB_{AP}, NB_{ANP}, NB_{BP} and NB_{BNP}, we add the number of packets in transport NBP_T to the set of system-wide variables. Recall that a packet is in transport if it is held by an AGV. The simulations from section 5.1 show that the average of NBP_T, denoted by $NBP_{T,avg}$, is also smooth (see figure 6(c)) and thus suitable for our algorithms. Therefore, we add the averaged variable $NBP_{T,avg}$ to our set of observation variables. Note that the total number of AGVs is chosen to be constant. So, for a set of N runs, these macroscopic variables are completely determined by the variables NB_{AP}, NB_{ANP}, NB_{BP} and NB_{BNP}, averaged over all runs. The precise positions and internal states of each AGV are degrees of freedom (steps 3 and 4 from road map).

We recall that the current state of each individual run is limited to a discrete set of values for the entropy E, the normalised sum S and the number of packets in transfer NPB_T, because changes in these values reflect that one AGV has changed zone, or has picked up or delivered a packet. We denote all possible states by $x_i = (E_i, S_i, NBP_{T,i})$. A complete initialisation operator (step 5 from road map) consists of two steps: first we need a procedure to find the values of NB_{AP}, NB_{ANP}, NB_{BP}, NB_{BNP}, given a particular state x_i; and second, when average values E_{avg}, S_{avg}, $NBP_{T,avg}$ and a number of runs N are given, we need to find numbers N_i that indicate how often state x_i will be chosen as initial condition so that the averages over all N runs equal the given average values E_{avg}, S_{avg}, and $NBP_{T,avg}$. For a realistic simulation, the distribution of the initial conditions over the possible states should reflect the probability for the occurrence of that state.

Initialisation of one simulation. From equation (3) and (5) one can derive:

$$E = \frac{-p_{BP}\log p_{BP} - (1-p_{BP})\log\left[1-p_{BP}\right]}{\log 2} \qquad (7)$$

It can easily be checked that this equation has two possible solutions for p_{BP}, i.e. p_{BP1} and $p_{BP2} = 1 - p_{BP1}$. However, only one will fulfill the required constraints:

- $NBP_T = NB_{BP} + NB_{AP}$
- $NB = NB_{AP} + NB_{BP} + NB_{ANP} + NB_{BNP}$
- $NB_i \geq 0 (i \in \{ANP, AP, BNP, BP\})$
- equation (1) and equation (3).

One solution p_{BP} can be found numerically. If this solution does not comply with the constraints, then the other solution, given by $p_{BP2} = 1 - p_{BP1}$ will. Using the complying solution for p_{BP} and $p_{ANP} = 1 - p_{BP}$, we can calculate the NB_i values:

$$\begin{aligned} NB_{BP} &= p_{BP} * NB * S && (8)\\ NB_{ANP} &= p_{ANP} * NB * S && (9)\\ NB_{AP} &= NBP_T - NB_{BP} && (10)\\ NB_{BNP} &= NB - NB_{BP} - NB_{ANP} && \\ &\quad - NB_{AP} - NB_{BNP} && (11) \end{aligned}$$

Equation (10) assigns the remaining packets to the AGVs in zone A that hold a packet. Then equation (11) just assigns the number of remaining AGVs to the last variable NB_{BNP}. Once the NB_i values are calculated, there are multiple initialisations possible because the exact AGV positions and the AGVs holding a packet are not determined. These degrees of freedom are initialised randomly.

Initialisation of a set of runs. To define the initialisation operator `init` completely, we need to solve the following inverse problem: given values for E_{avg}, S_{avg}, $NBP_{T,avg}$, find a set of N states, such that the averages are as specified. In general, there are many solutions to this problem. For a correct "equation-free" simulation, it is important to pick a solution that is as realistic as possible.

First, we determine the probability of occurrence of each state x_i. To this end, we perform a long simulation of 50000 steps. At each time step, we record the current state x_i, and obtain a histogram of occurrences. Then, the probability p_i of the system being in one of these states is approximated by the number of occurrences of state x_i, divided by the number of time steps. These p_i can be assumed to be the fractions of time that the system is expected to be in state x_i. We then obtain a probability distribution for the macroscopic states. Note that we assume that the states x_i cover the whole spectrum of possible states and that, as time approaches infinity, the probabilities p_i converge to a constant limit. Our simulations confirm this as a valid assumption.

We need to choose the initial conditions for a simulation as closely as possible to this distribution; otherwise, we probably do not capture the true average behaviour. Indeed, suppose that we initialise with a set of runs that has the correct average, but that is initialised mainly with states that are improbable. The results would be biased towards the improbably initialised behaviour and thus meaningless to interpret the real average behaviour. Therefore, we compute weights w_i, associated to each of the possible states of the system, as follows. We require that the weights w_i are chosen such that they are as close as possible to the probabilities p_i. This can be written as a minimisation problem,

$$\min_{w_i \in [0..1]} \sum_i |w_i - p_i|,$$

which is subject to the following constraints,

$$\sum_i w_i E_i = E_{avg}; \sum_i w_i S_i = S_{avg}$$
$$\sum_i w_i NBP_{T,i} = NBP_{T,avg}; \sum_i w_i = 1.$$

These constraints indicate that the weighted averages should be consistent with what we prescribed, and the weights should be normalised. Then, we have assured that we have a fraction w_i of initial conditions for each of the possible discrete states, such that both the distribution of initial conditions is close to the p_i-distribution, and the average initial entropy, sum and number of packets in transport are as prescribed. Next, we need to assign initial conditions to N number of runs. This is done by assigning N_i runs with each initial condition/state x_i such that

$$\min_{N_i} \sum_i |N_i - w_i N|,$$

subject to the constraint

$$\sum_i N_i = N.$$

The minimisation requires that the fraction of runs initialised with each state N_i/N is as close as possible to the fraction w_i, while fulfilling the constraint that their sum equals N. Finally, in order that this set of runs has the correct averages exactly, we weigh them with a factor $\alpha_i = w_i N/N_i$. This is easily seen by combining the constraints from the two above minimisation problems. The two minimisation problems can easily be solved using standard linear programming methods [17].

Separation of scales. It is important for the equation-free approach that the randomly initialised microscopic properties of the system, evolve much faster than the system-wide variables (step 6 from road map). In our case, these variables include the exact position each AGV, whether they are holding a packet or not, as well as the pheromone gradient from the source and destination locations. It is clear that the AGV movement, their pick ups and drop offs occur faster than changes in E_{avg}, S_{avg}, or $NBP_{T,avg}$. We also expect the pheromone gradient to evolve quickly to its natural state, so that this indeterminacy does not create artifacts.

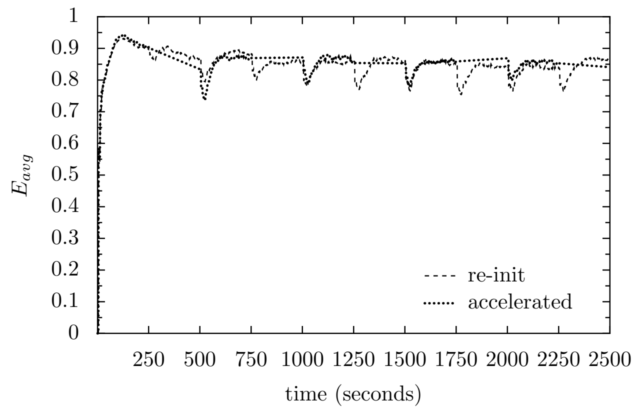
5.3. Simulation with Re-initialisation

We will now check the validity of the macroscopic time-stepper, i.e. validate that the chosen macroscopic variables completely reflect the evolution of the system-wide behaviour. To this end, we perform a simulation using $N = 1000$ runs, which have been initialised such that $E_{avg} = 0$, $S_{avg} = 0.5$ and $NBP_{T,avg} = 0$. Note that all simulations in this paper use the steady scenario described in section 3 (step 7 from road map). The simulation is performed over 250 time-steps, after which E_{avg}, S_{avg} and $NBP_{T,avg}$ are measured. To reduce the statistical noise, we measure these variables as being averaged over the last 50 time-steps as well as over all runs. We then re-initialise the simulation immediately, following the lines of the previous section.

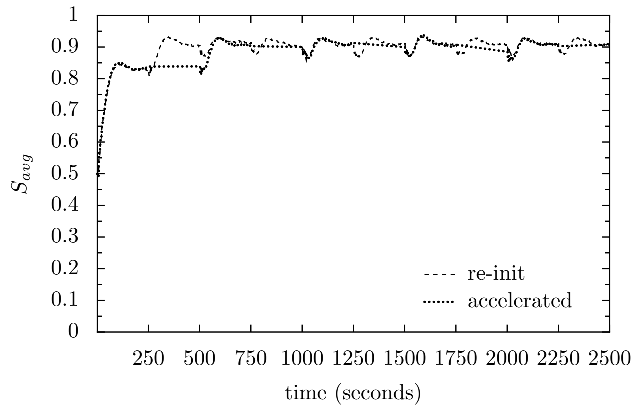
To compare with the preliminary results of section 5.1, we performed 10 of these macroscopic time steps, such that the total simulated time is again 2500 seconds, such as in section 5.1. We remark that, at this stage, the simulations are not yet accelerated. The only difference with the simulations in section 5.1 is that after each 250 time-steps the simulation has been re-initialised. Figures 7(a), 7(b), and 7(c) show that these re-initialisations do not affect the results significantly compared to the preliminary simulation results (figures 6(a), 6(b), and 6(c)). The figures show the values of E_{avg}, S_{avg} and $NB_{PT,avg}$ at each time-step. After each re-initialisation, we first see a steep transient. This is due to the fact that the re-initialisation only ensures that the macroscopic variables have been initialised consistently. The microscopic states that are not accounted for are initialised randomly. Therefore, an artificial fast evolution of E_{avg}, S_{avg}, and $NBP_{T,avg}$ is introduced which equilibrates quickly. This results in a sharp initial transient, which disappears well before the end of the macroscopic time step of size 250. Therefore, we conclude that the values of the variables E_{avg}, S_{avg} and $NBP_{T,avg}$ suffice to predict the evolution of the system. This confirms that the macroscopic time-stepper as constructed above accurately simulates the system-wide behaviour.

5.4. A First Accelerated Result

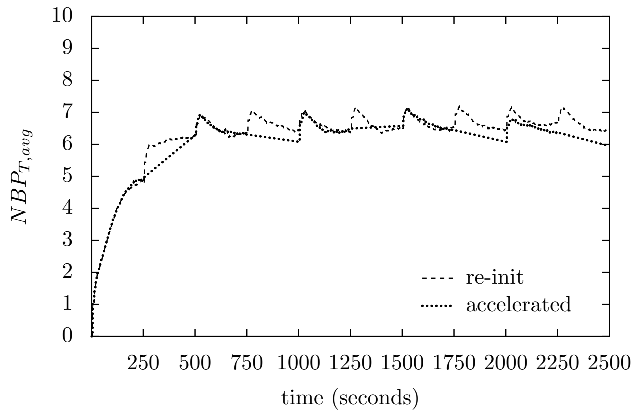
We accelerate the simulations by using projective integration as an analysis algorithm (step 8 from road map). The macroscopic variables that we introduced vary smoothly over time, such that the extrapolation step can be done reliably. Due to the extrapolation steps, the computational effort is reduced. We consider the following set-up: we perform a simulation run of 250 time-steps, as in the previous section, after which we obtain the new values of the variables E_{avg}, S_{avg} and $NBP_{T,avg}$ by averaging the simulation results over the last 50 time steps (steps 201 to 250). We also measure the macroscopic variables after 200 time-steps



(a) average entropy



(b) average sum



(c) average number of packets in transport

Figure 7. E_{avg}, S_{avg}, and $NBP_{T,avg}$ of a set of $N = 1000$ runs obtained with simple re-initialisation (`re-init`) and with projective integration (`accelerated`).

by averaging over steps 151 to 200. It is clear from figure 7 that the macroscopic time-step of 250 seconds is sufficiently large, such that these measurements are not contaminated by the re-initialisation transient. Using these two measurements, we extrapolate over a time interval of length 250 and re-initialise. This way, we have halved the computational effort, since the simulations will now only be performed in half of the time interval for which we obtain results. The accelerated results are shown in figure 7.

Several observations can be made. First, it is clear that the increased efficiency comes at a penalty in the accuracy. In an extrapolation step, some accuracy is lost. It is important to note that this extrapolation error is mainly due to the non-deterministic nature of the underlying simulation. We expect that advanced techniques for variance reduction, e.g. [18], can significantly improve these results, and we are actively pursuing research in this direction. However, even with this simple averaging strategy, we are already able to halve the simulation time.

Second, as can be seen from the measurements in figure 6(a), the macroscopic variables approach a steady state. This indicates that a numerical algorithm to determine this steady state directly is preferably used (step 8 from road map). The Newton algorithm (see section 4.2) is such an algorithm which takes an arbitrary initial guess for the steady state as input and converges to the real steady state. However, due to the rather large variance on the measurements, we were unable to converge successfully, unless the error tolerance for the Newton algorithm was set very low (2 digits of accuracy). The reason is that the extrapolation steps used in Newton's algorithm require a higher degree of smooth behaviour than a projective integration over time as described previously. We expect to remedy this in the near future by using a suitable variance reduction technique, and will report on this in a future publication.

6. Conclusion and Future Work

This paper describes a promising analysis approach that allows to address a fundamental engineering issue for autonomic decentralised systems, i.e. assessing the system-wide behaviour of self-organising emergent systems. Due to the complex, dynamic, and non-deterministic nature of such systems, a formal or analytic proof and even predictions of the exact behaviour are not feasible. Therefore, an alternative road map to study trends in the system-wide behaviour is proposed, based on so-called "equation-free" macroscopic analysis techniques. Numerical analysis algorithms are used to steer and accelerate the simulation process. The main assumption is the presence of a separation in time scales between the microscopic and macroscopic evolution. Initial steps in the AGV case study show that this technique can yield more reliable results about the system-wide behaviour, compared to mere simulation observation results, at an affordable computational cost.

Several questions remain open, and constitute the basis for future work. First, it is a challenge to alter the numerical procedure in such a way that it uses suitable variance reduction techniques to obtain the level of smoothness that the numerical algorithms require. This is necessary to successfully apply more advanced algorithms, such as bifurcation tools to study parameter sensitivity. For example, the influence of a specific parameter on the system-wide behaviour, e.g. the number of AGVs on the factory floor can significantly influence the guarantees that can be given. Such a parameter can then be used to control the system-wide behaviour by making sure that it stays in the range of values that lead to desired behaviour. Secondly, additional steady scenarios to analyse need to be considered (e.g. low utilisation load, frequent failure of AGVs, frequent occurrence of obstacles, etc.). In general, one needs to identify the parameters that determine the conditions in which a system is used, and analyse steady scenarios to cover the full range of possibilities. Thirdly, we are actively investigating how additional system-wide properties in the AGV case, such as throughput, can be incorporated in the analysis. The identification of a set of problem-specific "key variables" still needs to be done on a case-by-case basis. Once the procedure has been successfully applied to a range of systems, we expect to obtain a set of useful guidelines for this crucial step. Finally, the biggest challenge for the future is to see to which extent this analysis approach can be integrated into a systematic engineering approach for self-organising emergent systems.

7. Acknowledgements

G. Samaey is a Research Assistant of the Fund for Scientific Research – Flanders (FWO). This work is supported by grants FWO G.0130.03, IUAP/V/22 (Belgian Federal Science Policy Office) (G. Samaey, D. Roose), and by the K.U.Leuven research council as part of the Concerted Research Action on Agents for Coordination and Control - AgCo2 (T. De Wolf, T. Holvoet).

References

[1] R. J. Anthony. Emergence: A paradigm for robust and scalable distributed applications. In *Proceedings of IEEE International Conference on Autonomic Computing (ICAC'04)*, pages 132–139, New York, May 2004.

[2] T. Balch. Hierarchic social entropy: An information theoretic measure of robot group diversity. *Autonomous Robots*, 8:209–237, 2000.

[3] T. Balch and R. C. Arkin. Communication in reactive multiagent robotic systems. *Autonomous Robots*, 1(1):27–52, 1995.

[4] T. De Wolf and T. Holvoet. Emergence and Self-Organisation: a statement of similarities and differences. In *Proceedings of the Second International Workshop on Engineering Self-Organising Applications*, pages 96–110, July 2004.

[5] Egemin and DistriNet. Emc^2: Egemin modular controls concept,. IWT-funded project with participants: Egemin (http://www.egemin.be) and DistriNet (research group of K.U.Leuven). Started on 1 March 2004, ending on 28 February 2006.

[6] M. P. Fromherz, L. S. Crawford, C. Guettier, and Y. Shang. Distributed adaptive constrained optimization for smart matter systems. In *Proceedings of AAAI Spring Symposium on Intelligent Embedded and Distributed Systems*, 2002.

[7] C. Gear and I. Kevrekidis. Projective methods for stiff differential equations: problems with gaps in their eigenvalue spectrum. *SIAM Journal on Scientific Computing*, 24(4):1091–1106, 2003.

[8] S. Guerin and D. Kunkle. Emergence of constraint in self-organizing systems. *Nonlinear Dynamics, Psychology, and Life Sciences*, 8(2):131–146, April 2004. (available at http://www.redfish.com/).

[9] IBM. *Autonomic Computing Manifesto: IBM's Perspective on the State of Information Technology*. IBM, 2001. (online at http://www.research.ibm.com/autonomic/manifesto/).

[10] J. O. Kephart and D. M. Chess. The vision of autonomic computing. *IEEE Computer Magazine*, 36(1):41–50, January 2003.

[11] I. G. Kevrekidis, C. W. Gear, and G. Hummer. Equation-free: The computer-assisted analysis of complex, multiscale systems. *AIChE Journal*, 50(7):1346 – 1355, 2004.

[12] I. G. Kevrekidis, C. W. Gear, J. M. Hyman, P. G. Kevrekidis, O. Runborg, and C. Theodoropoulos. Equation-free, coarse-grained multiscale computation: enabling microscopic simulators to perform system-level analysis. *Communications in Mathematical Sciences*, 1(4):715 – 762, December 2003. (available online at http://www.intlpress.com/CMS/).

[13] U. Nehmzow. Quantitative analysis of robot-environment interaction - towards scientific mobile robotics. *J. Robotics and Autonomous Systems*, 44:55–68, 2003.

[14] H. V. D. Parunak and S. Brueckner. Entropy and Self-Organization in Multi-Agent Systems. In J. P. Müller, E. Andre, S. Sen, and C. Frasson, editors, *Proceedings of the Fifth International Conference on Autonomous Agents*, pages 124–130, Montreal, Canada, 2001. ACM Press. (available at citeseer.ist.psu.edu/parunak01entropy.html).

[15] H. V. D. Parunak, R. Savit, and R. L. Riolo. Agent-based modeling vs. equation-based modeling: A case study and users' guide. In *MABS*, pages 10–25, 1998. (online at http://www.erim.org/˜vparunak).

[16] W. H. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, Cambridge, 2nd edition, 1992.

[17] W. Winston. *Operations research: applications and algorithms*. International Thomson Publishing, Belmont, CA, 3rd edition, 1994.

[18] Y. Zou and R. Ghanem. A multiscale data assimilation with the ensemble kalman filter. *Multiscale modelling and simulation*, 3(1):131–150, 2004.