Greedy Effects in Autonomous Mobility: Experimental and Theoretical Analysis

Natalia Chechina, Peter King, Phil Trinder

Department of Computer Science, Heriot-Watt University, Edinburgh, UK, EH14 4AS

Abstract

Distributed load balancers exhibit thrashing where tasks are repeatedly moved between locations due to incomplete global load information. This paper shows that systems of Autonomous Mobile Programs (AMPs) exhibit the same behaviour, and identifies two types of redundant movement (greedy effect). AMPs are unusual in that, in place of some external load management system, each AMP periodically recalculates network and program parameters and may independently move to a better execution environment. Load management emerges from the behaviour of collections of AMPs.

The paper explores the extent of greedy effects by simulating collections of AMPs and proposes negotiating AMPs (NAMPs) to ameliorate the problem. We present the design of AMPs with a competitive negotiation scheme (cNAMPs), and compare their performance with AMPs by simulation. We establish new properties of balanced networks of AMPs, and use these to provide a theoretical analysis of greedy effects.

Keywords: load balancing, autonomous mobile program, mobile agent

1. Introduction

Autonomous mobile programs (AMPs) are mobile agents that improve execution efficiency by managing load; AMPs are aware of their resource

*Corresponding author. Fax: +44 (131) 451-3327
Email addresses: nc75@hw.ac.uk (Natalia Chechina), P.J.B.King@hw.ac.uk (Peter King), P.W.Trinder@hw.ac.uk (Phil Trinder)
URL: http://www.macs.hw.ac.uk/~nc75 (Natalia Chechina),
http://www.macs.hw.ac.uk/~pjbk (Peter King),
http://www.macs.hw.ac.uk/~trinder (Phil Trinder)

Preprint submitted to Journal of Parallel and Distributed Computing   November 1, 2010
needs, sensitive to the execution environment and periodically seek a better location to reduce execution time [1]. To analyse AMP behaviour on local area networks (LANs) we have constructed a simulation model [2]. Comparing the simulation results and observations of the real system [1] shows that simulated and real AMPs enter the same stable states (i.e. where no AMP can reduce its execution time by moving). The differences between simulated and real AMPs are minor and readily explained.

Like other distributed load balancing systems, e.g. [3, 4, 5], both real and simulated collections of AMPs exhibit greedy effects. These greedy effects are a phenomenon that results in redundant AMP movements during the balancing of loads between locations, and are a result of locally optimal choices made by each AMP.

The aim of the current paper is to examine properties and features of the greedy effects in collections of AMPs and expose their causes. We further aim to reduce the greedy effects, and to estimate the greedy effects in the modified algorithm. The estimation of the greedy effects is implemented both theoretically and by using simulation.

The paper is organised as follows. We discuss AMP related concepts (Section 2), and identify two forms of greedy effects (Section 3). The AMP cost model and simulation are adapted to facilitate an investigation of the greedy effects (Section 4). We examine the AMP greedy effects on initial distribution, rebalancing and AMP execution time (Section 5). Analysis of types of movements shows that the majority of redundant movements occur because an AMP is unaware of the intentions and movements of other AMPs. So, we discuss ways to reduce the greedy effects and propose the concept of negotiating AMPs (NAMPs) that communicate their intentions with the view to reduce redundant moves. While a number of negotiation schemes are possible, we have designed and simulated AMPs with a competitive scheme (cNAMPs) (Section 6). cNAMPs announce their intentions to move and compete with each other for opportunity to transfer to the new location.

An analysis of simulated cNAMP results shows that even this simple negotiation significantly decreases both the number of redundant movements and the time to rebalance. For example, cNAMPs make no redundant movements during initial distribution, which makes initial balancing in the conducted experiments at least three times faster in comparison with AMPs (Section 7). We establish the properties of balanced states, i.e. states where no AMP can gain a greater AMP relative speed by moving [6] (Section 8), and establish the maximum number of redundant movements after a cNAMP termination.
using case analysis (Section 9). A summary of the results and discussion of future work are provided in Section 10. We assemble a set of consistent and precise definitions about the behaviour of collections of AMPs and cNAMPs which appears at the end of the paper (Glossary).

Novelty. The paper goes beyond [7] by making the following additional research contributions:

1. We establish the properties of balanced states to estimate the significance of the greedy effects. We analyse AMPs as the general case and examine a number of properties, such as independent balance, singleton optimality, and consecutive optimality (Section 8).

2. We examine the significance of the greedy effects by predicting the worst case (maximum number) of redundant movements after a cNAMP termination from a network of \( q \) subnetworks (i.e. sets of locations with identical available speeds) using case analyses (Section 9).

3. We define a consistent terminology for reasoning about the behaviour of AMPs and cNAMPs. Many definitions are generalisations of, or more precise versions of, those used in earlier papers [2, 6] (Glossary).

2. Related Work

2.1. Introduction

This section discusses previous work, first exploring the three core AMP technologies: load balancing (Section 2.2), mobile computing (Section 2.3) and autonomous systems (Section 2.4), before covering mobile agent load balancing (Section 2.5).

The novelty of the AMP approach is that it automates the performance prediction process using compile-time analysis to identify potential migration points in the code, and devolves to the AMP the decision as to whether to migrate or not as each migration point is executed. Effective load management results from emergent behaviour of collections of AMPs. AMP behaviour has previously been investigated using mobile languages like Java Voyager [8] and using simulation [2].

2.2. Distributed Load Management

The problem of load balancing in distributed computer systems has been widely studied, e.g. [5, 9, 10]. The main difficulties which load balancing
approaches face are minimizing execution time, minimizing communication delays, and maximizing resource utilization [11]. According to the taxonomy in [12], AMPs are global dynamic load balancers. According to the detailed classification of dynamic load balancers [13], AMPs have a hybrid decision making policy, sender initiated scheme, and use a best fit simple transformation. AMPs take all movement decisions themselves, independently of other AMPs. They use the information collected by load servers (which only collect network state information and are designed to reduce AMP coordination time) to estimate their remaining execution time if they executed at other locations. If the predicted time to complete at the current location is greater than the predicted time at the best available remote location plus the communication time to reach the remote location, then the AMP will relocate.

The way AMPs search a better location may seem similar to iterative algorithms [14, 15]. Indeed, both approaches aim to distribute load among nodes, and the load is estimated in the number of tasks (or programs) per node. There are two distinctions: firstly, in iterative algorithms locations make the movement decisions, but with AMPs programs make themselves. Secondly, while in iterative algorithms the locations aim with each iteration to approach some mean load, AMPs seek only to reduce the program execution time.

In general redundant movements are a result of locally optimal choices. To reduce the number of redundant movements different techniques are used, such as limiting any particular task to a maximum number of migrations [16], calculating the largest difference between the estimated execution time and the interprocess communication cost [17], applying market mechanisms [4], etc. These techniques aim to reduce the redundant movements where locations have a task scheduler or management agents, whereas we aim to reduce them for independent AMPs without a scheduler.

2.3. Mobility

Mobile computations can move between locations in a network and potentially enable better use of shared computational resources [18]. Basically a mobile program can transport its state and code to another location in a network, where it resumes execution [19].

Fuggetta et al. distinguish two forms of mobility supported by mobile languages [20]: weak mobility is the ability to move only code from one location to another; strong mobility is the ability to move both code and
its current execution state. AMPs were constructed using languages with weak and strong mobility. However, a substantial subset of experiments was conducted using Java Voyager [21] which supports only weak mobility.

2.4. Agents and Autonomous Systems

Agent technology is a high-level, implementation independent approach to developing software as collections of distinct but interacting entities which cooperate to achieve some common goal [22]. With the continuing decline in price and increase in speed of both processors and networks, it has become feasible to apply agent technology to problems involving cooperation in distributed environments. In particular, where agents may change location, typically to manipulate resources in varying locations [23].

Autonomous systems are also called autonomic computing systems, and a definition has been given by IBM: “autonomic computing system can manage themselves given high-level objectives from administrators” [24]. Four aspects of self-management are self-configuration, self-optimisation, self-healing, and self-protection. AMPs are primarily self-optimisation systems.

2.5. Mobile Agent Load Balancing

AMPs differ from other mobile agent systems designed to balance load in that each AMP is both autonomous and self aware, i.e. it knows key information like remaining execution time and program size. Another difference is that the approach does not split a program into subtasks as in [10, 25, 26]. An AMP is the whole program. Thus, AMPs represent a radical point in distributed decision making when not a location server [27], or a load balancing coordinator [28], or a cluster manager [25] decide where and when to move but each agent itself.

3. Greedy Effects

An optimal rebalancing is a sequence of AMP movements that is the minimum number of AMP movements needed to enter a stable state.

The AMP greedy effects are the result of a non-optimal AMP rebalancing, which differs from the optimal rebalancing in having additional redundant movements, and is a result of the AMP making a locally optimal choice, i.e. AMPs do not possess sufficient and accurate state information to make the optimal movement decision. There are two types of the AMP greedy effects: location thrashing and location blindness. Both location thrashing and blindness are observed in real [8] and simulated [2] AMP experiments.
3.1. **AMP Distribution Scenarios**

To illustrate the greedy effects we first introduce the following AMP scenarios, each of which specifies the number of AMPs and locations, and types of locations:

- **Scenario 1**: 25 AMPs on 15 locations with CPU speeds 3193 MHz (Loc1 – Loc5), 2167 MHz (Loc6 – Loc10) and 1793 MHz (Loc11 – Loc15).
- **Scenario 2**: 20 AMPs on 10 locations with CPU speeds 3193 MHz (Loc1 – Loc5), 2168 MHz (Loc6) and 1793 MHz (Loc7 – Loc10).
- **Scenario 3**: 10 AMPs on 3 locations with CPU speeds 3193 MHz.

The scenarios are the same as were used in real experiments. For all scenarios Loc1 is the root location. By the root location we mean the location where all AMPs start; it is also called either initiating location or first location in [1]. In the experiments we use large and small AMPs. Large AMPs are programs of matrix multiplication of size 1000 × 1000, and small AMPs are programs of matrix multiplication of size 500 × 500.

3.2. **Location Thrashing**

Location thrashing is the greedy effect resulting from an AMP’s lack of information about other AMPs intending to move to the same location. That is, two or more AMPs decide to move on the basis of the same information about the target location, which causes further AMP retransmission. Location thrashing occurs in dynamic load balancing systems; other terms are processor thrashing [29], task thrashing [16], task dumping [3, 30], transmitting dilemma [31].

Location thrashing is illustrated in Figure 1(a), which shows AMP movements in the experiments with the real system [1], based on scenario 3 (Section 3.1). In Figure 1(a) each icon denotes an AMP. The locations are specified on the vertical axis and the horizontal divisions represent time intervals. The time intervals are of different lengths, showing states that the system enters as it attempts to reach a stable state.

After the termination of two AMPs from Loc1 in state S1, state U1 is entered and two AMPs from Loc2 and one AMP from Loc3 discover a better opportunity for execution on Loc1 simultaneously. These three AMPs move to Loc1 (state U2), and then one AMP moves back to Loc2 to enter a stable state (state S2). In this case, an optimal rebalancing from the state U1 to the state S2 can be reached by one AMP movement to Loc1 from each of Loc2 and Loc3 as Figure 1(b) shows. Note that location thrashing incurs two performance penalties, namely the cost of additional communication and the
cost of slower execution. By *additional communication cost* we only mean additional AMP movements during a rebalancing and not communication time they may take.

3.3. Location Blindness

*Location blindness* is the greedy effect resulting from an AMP’s lack of precise information about the remaining execution time of other AMPs. The problem is not with poor runtime predictions, but rather an inability to obtain accurate AMP runtime predictions at distributed locations, i.e. the more accurate information that is required, the more *expensive* it becomes to collect and process the information in a distributed system [32].

Figure 2 shows an example of location blindness in some simulated AMP experiments. The experiment is undertaken on the basis of scenario 1 (Section 3.1). Numbers identify the number of AMPs on a location. After an AMP termination on *Loc14* in state *S1* and the system enters state *U1* an
AMP from Loc5 discovers the opportunity for faster execution first and moves (state U2). Then an AMP from Loc7 discovers the opportunity for faster execution on Loc5 and also moves (state S2). Figure 3 shows optimal AMP rebalancing. In contrast to the location threshing, location blindness only causes redundant communication and causes no additional computation cost. Each AMP will have improved its environment by moving.

Thus, among two types of the AMP greedy effects the location thrashing is more harmful, because it causes an increase in AMP execution time, and hence decreases AMP efficiency. To analyse the number of redundant movements and time to balance we investigate the greedy effects with simulated AMPs.

4. Simulating the Greedy Effects

Earlier simulation experiments of AMPs on a LAN showed that the simulation closely models real AMPs on LANs, and, hence, is an effective tool to analyse AMP behaviour [2]. We provide essential calculations and discuss properties and features of AMPs (Section 4.1). The minor changes we propose to the previous simulation model [2] which allow the simulation of
the greedy effects can be found in [33]. These changes include bounds on information transfer times and delays on the transfer of state information.

4.1. Cost Model and Parameterisation

The key equations of the AMP cost model defined in [6, Section 3.3.2] are repeated here, and the parameter values are given in Table 1.

Available speed is the execution speed of a single AMP on a location, i.e.

\[
S = (CPU\text{ speed}) \cdot (1 - \text{non AMP load}),
\]

is used to differentiate the total resources of a location from the resources available for AMPs. The current research, like the previous AMP investigations [1], assumes that all resources of a location are available for AMPs, and the CPU speed coincides with the available speed, except the case of the root location where the external load is higher.

An AMP relative speed, \( R \), is available speed equally divided between the
Table 1: Parameter Definitions

<table>
<thead>
<tr>
<th></th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d$</td>
<td>Dimension of square matrix</td>
</tr>
<tr>
<td>$\text{gran}$</td>
<td>Fragment of work, which must be executed between searches for a better location</td>
</tr>
<tr>
<td>$N$</td>
<td>Number of locations in a network</td>
</tr>
<tr>
<td>$O$</td>
<td>Overhead</td>
</tr>
<tr>
<td>$R$</td>
<td>AMP relative speed</td>
</tr>
<tr>
<td>$S$</td>
<td>Available speed</td>
</tr>
<tr>
<td>$T_{\text{comm}}$</td>
<td>Time for single communication</td>
</tr>
<tr>
<td>$T_{\text{coord}}$</td>
<td>Coordination time in the load server architecture</td>
</tr>
<tr>
<td>$T_{\text{gran}}$</td>
<td>Execution time of fragment of work $\text{gran}$</td>
</tr>
<tr>
<td>$T_h$</td>
<td>Execution time on the current location</td>
</tr>
<tr>
<td>$T_t$</td>
<td>Execution time on the new location</td>
</tr>
<tr>
<td>$T_{\text{renew}}$</td>
<td>Time of total state information renewing by load server</td>
</tr>
<tr>
<td>$T_{\text{req}}$</td>
<td>Time to send a request to a remote load server and receive a response</td>
</tr>
<tr>
<td>$T_{\text{send}}$</td>
<td>Time to transmit an AMP to the new location</td>
</tr>
<tr>
<td>$x_{\text{loc}}$</td>
<td>Number of AMPs on a location</td>
</tr>
</tbody>
</table>

AMPs at the location, $x_{\text{loc}}$, i.e.\(^1\)

$$R = \frac{S}{x_{\text{loc}}}.$$ \quad (2)

An AMP executes for a time $T_{\text{gran}} = T_{\text{coord}} \frac{O}{S}$ before it tests the relative speeds of other locations to see if a move will improve its completion time. The time parameters for our simulation are taken from Java Voyager AMP measurements on a LAN [6], as validated in [2]. The coordination time, $T_{\text{coord}}$, is determined experimentally to be 0.011s for a load server architecture. For an overhead $O$ of 5%, $T_{\text{gran}}$ is 0.22s.

After executing for $T_{\text{gran}}$ an AMP makes a request to the load server of the current location about states of other locations in the network. If an AMP decides to stay on the current location, then it continues execution for a further $T_{\text{gran}}$ seconds, otherwise it moves to a new location taking $T_{\text{send}} = 0.029 + 5.07 \times 10^{-6} \cdot d^2$ seconds. The AMP studied here performs square matrix multiplication and $d$ is the dimension of the matrix. Experiments with other programs, e.g. coin counting or ray tracing, show consistent behaviour, and in the following we use matrix multiplication.

As soon as an AMP makes a decision to move to a new location, the load server of the current location decreases its number of AMPs. In turn, the

---

\(^1\)In [1] the \textit{available speed} is called \textit{relative speed}, and the \textit{AMP relative speed} is called \textit{average relative speed}. 

10
Table 2: AMP Greedy Effect Experiment Summary

<table>
<thead>
<tr>
<th>Config</th>
<th>E1: Initial distribution</th>
<th>E2: Rebalancing after an AMP termination</th>
<th>E3: Large AMP execution time, sec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean No. of redun. moves</td>
<td>Mean time, sec</td>
<td>Mean No. of redun. moves</td>
</tr>
<tr>
<td>Scenario 1</td>
<td>64</td>
<td>60.4</td>
<td>6</td>
</tr>
<tr>
<td>Scenario 2</td>
<td>43</td>
<td>50.5</td>
<td>11</td>
</tr>
<tr>
<td>Scenario 3</td>
<td>13</td>
<td>26.8</td>
<td>6</td>
</tr>
</tbody>
</table>

new location increases its number of AMPs after the full AMP has arrived. To obtain state information from other locations in a network, a load server sends requests to locations in a sequential order following [6]. The time taken to send a request to a remote Java process and receive a response, $T_{req}$, using Java Voyager has been measured, and is equal to 0.25s [6, p. 80]. Thus, a load server completely renews state information about $N$ locations in a network every $T_{renew}$ seconds:

$$T_{renew} = T_{req}(N - 1).$$

(3)

The main rule on the basis of which AMPs make a decision to move to a new location is whether execution time on the current location, $T_h$, exceeds execution time on the next location, $T_n$, and communication delay, $T_{comm}$:

$$T_h > T_n + T_{comm}.$$  

(4)

If condition (4) is satisfied, then an AMP moves. Here, communication time is time to transfer an AMP, i.e. $T_{comm} = T_{send}$.

5. AMP Greedy Effect Experiments

This section investigates frequency and significance of greedy effects in the three scenarios from Section 3.1 on homogeneous and heterogeneous networks. We further classify the redundant movements and identify the primary cause.

A homogeneous network is a set of locations with the same available speeds, except the root location, which may be different, because of the overheads of initiating the remote processes. A heterogeneous network is a set of locations with different available speeds.

5.1. Experiments and Results

Experiment 1 (E1): Initial distribution. This experiment investigates the greedy effects as large AMPs distribute over the network from a single
location in each scenario. Column Initial distribution in Table 2 shows the mean number of redundant movements and the time required for the system to enter a stable state. Clearly the optimal number of AMP movements to reach a stable state would occur if each AMP moved a maximum of once, directly to the location it will occupy in the stable state. The column shows that with the increase of the number of locations and AMPs the number of average movements per AMP also increases.

Experiment 2 (E2): Rebalancing after an AMP termination. The experiment measures the number of movements and time required for a system to rebalance after an AMP termination in each scenario. Column Rebalancing after an AMP termination in Table 2 shows the mean number of redundant movements and rebalancing time.

Experiment 3 (E3): Large AMP execution time. This experiment estimates large AMP execution time and measures its variability in each scenario. The total number of AMPs corresponds to the relevant scenario. All AMPs, two of which are small and the rest are large, start on the location 1. The results are presented in column Large AMP execution time in Table 2.

The simulation experiment results in Table 2 show that an increase of the number of AMPs and/or locations causes an increase in the number of
Figure 5: Rebalancing after an AMP Termination (Scenario 1)

redundant movements per AMP during initial distribution. The dependence of the number of redundant movements on the number of locations and the total number of AMPs after an AMP termination is less obvious. However, additional experiments where AMPs have large execution times show that the number of redundant movements is directly proportional to the number of locations and AMPs.

5.2. Analysing Greedy Effects

To identify opportunities to reduce AMP redundant movements we analyse AMP movements during initial distribution and after an AMP termination on Figures 4 and 5 respectively. The Figures show movements in larger scale systems of AMPs, i.e. Scenario 1, than Figures 1 and 2. Figure 4 shows the initial AMP distribution between locations as the system rebalances from initial (unstable) state U1 to stable state S. As all AMPs move from Loc1 in state U1 to Loc2 – Loc5 in state U2, we do not indicate them with lines. There are 88 movements in total, but there would only be 24, if each AMP moved directly to the location it reaches in state S, i.e. the system makes 64 redundant movements. Figure 5 shows 12 AMP movements which the system makes to enter a new stable state S2 after an AMP termination on
The analysis of AMP movements allows them to be classified into the following main types of redundant movements: 1) two or more AMPs move from one location to another, and then some of them rebalance; 2) two or more AMPs move from a location, and then some AMPs move back to the location; 3) two or more AMPs move from different locations to one location, and then some of them immediately move again. Therefore, we conclude that redundant AMP movements are mainly caused by AMP ignorance of intentions and actions of other AMPs in the network and, hence, lack of information to make an efficient decision, i.e. location thrashing.

6. Negotiating AMPs and cNAMPs

As the main reason for poor AMP movement decisions is a lack of communication between AMPs via the load servers, we propose to use negotiation to reduce the greedy effects. Possible methods of negotiation between AMPs and/or load servers are discussed in Section 6.1. We introduce a simple negotiation scheme in which AMPs announce their movement intentions and compete for opportunity to transfer. AMPs using this scheme are called cNAMPs. The modifications to AMP algorithm to provide this behaviour are described in Section 6.2.

6.1. Negotiating AMPs

The analysis of the greedy effects in the simulated AMP experiments in Section 5 shows that the majority of redundant movements occur because an AMP makes a decision on the basis of currently available information and is unaware of impending movements of other AMPs.

To analyse and understand AMPs better we draw an analogy between AMPs and human society, where an AMP acts as an individual. Then AMP behaviour corresponds to autistic behaviour. Such an AMP does not request information about other AMPs, does not provide information itself, and does not communicate with other locations to make efficient movement decision. To make a movement decision AMPs only rely on the current information and are not concerned about actions of other AMPs on the same information about the target location.

In order to reduce the greedy effects AMPs must negotiate with each other, i.e. communicate more information. There can be different types of negotiation, such as malicious, honest, etc. A malicious strategy would be for
Table 3: cNAMP and Load Server Pseudocode

<table>
<thead>
<tr>
<th>cNAMP</th>
<th>Load Server</th>
</tr>
</thead>
<tbody>
<tr>
<td>while work remains to execute</td>
<td>forever do</td>
</tr>
<tr>
<td>{if outstanding request &amp; positive response</td>
<td>case local cNAMP sent a request to Loc_i:</td>
</tr>
<tr>
<td>inform local load server about movement</td>
<td>lock information about Loc_i:</td>
</tr>
<tr>
<td>move to target location</td>
<td>{renew and unlock information about Loc_i:</td>
</tr>
<tr>
<td>else if no local cNAMP awaits a response</td>
<td>if positive response</td>
</tr>
<tr>
<td>{for n from 1 to total number of loc-s</td>
<td>reduce actual and committed loads</td>
</tr>
<tr>
<td>find minimum of T_n + T_comm</td>
<td>case arrival notification from remote cNAMP:</td>
</tr>
<tr>
<td>if T_h &gt; minimum</td>
<td>increase committed load</td>
</tr>
<tr>
<td>{send request to L_n</td>
<td>case cNAMP arrived:</td>
</tr>
<tr>
<td>inform local load server about request sent }</td>
<td>increase actual load</td>
</tr>
<tr>
<td>continue execution }</td>
<td></td>
</tr>
</tbody>
</table>

A load server to misrepresent the load so that other AMPs were deterred from moving to a location. An honest strategy requires AMPs and load servers to share information to reduce wasted movements and is more effective for load balancing. A range of honest AMP negotiation tactics are possible, e.g. competitive where AMPs compete with each other to move to the target location, queuing where each AMP has a sequence number in a queue, and moves to the new location only if an earlier AMP in the queue rejected to move, probabilistic where an AMP makes a decision to move on the basis of calculating the probability of simultaneous AMP movements from other locations, relationship where a network is logically divided into groups, locations first share information within their group, a location can be a member of more than one group, thus information is spread like a rumour.

6.2. The Design of cNAMPs

cNAMPs are negotiating AMPs with a competitive scheme, which announce their intentions to move and compete with each other for opportunity to transfer to the new location. In the context of cNAMPs the negotiation is a simple coordination among competitive and self-interested agents [34]. cNAMPs do not negotiate directly with each other, but only by means of a load server.

cNAMPs are designed only to reduce location thrashing. Eradicating location thrashing eliminates redundant movements during initial distribution and significantly reduces the number of redundant movements during rebalancing (Table 4 in Section 7). In addition, reduction of location blindness requires that cNAMPs and load servers possess even more information about
locations and cNAMPS of the network. In Section 9 we show that both the probability of redundant movements and their number are very small.

Unlike an AMP, a cNAMP first sends a request to the selected target location declaring the intentions to move. The request is also an agent which can be seen as a cNAMP representative. On arrival to the target location the representative recalculates parameters, informs the target load server if the decision is positive and moves back. Only if the representative confirms the movement decision does the cNAMP actually move; otherwise the cNAMP continues execution locally. When a request informs the target load server about a cNAMP movement the load server reports the load as if the transferring cNAMP had already arrived. Each load server maintains two values for the load: 1) the actual load which is the number of executing cNAMPS and is used for local cNAMP calculations; 2) the committed load which represents the actual load of a location together with the cNAMPS that confirmed their transferring to the location, and is used by remote load servers.

Thus, the target location updates its load twice: when it gets notification that a cNAMP intends to come (committed load), and when the cNAMP arrives (actual load). Pseudocode for cNAMP and load server implementations
Table 4: Comparative Summary of AMP and cNAMP Greedy Effects

<table>
<thead>
<tr>
<th>Configuration and type of experiment</th>
<th>Initial distribution</th>
<th>Rebalancing after an AMP/cNAMP termination</th>
<th>Large AMP/cNAMP execution time, sec</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Time, sec</td>
<td>Mean number of redundant movements</td>
<td>Time, sec</td>
</tr>
<tr>
<td>AMPs</td>
<td>60.4</td>
<td>64</td>
<td>22.5</td>
</tr>
<tr>
<td>cNAMPs</td>
<td>14.7</td>
<td>-</td>
<td>5.9</td>
</tr>
<tr>
<td>Reduction</td>
<td>4.11</td>
<td>64 moves</td>
<td>3.81</td>
</tr>
<tr>
<td>Scenario 1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AMPs</td>
<td>50.5</td>
<td>43</td>
<td>28.2</td>
</tr>
<tr>
<td>cNAMPs</td>
<td>12.4</td>
<td>-</td>
<td>7.8</td>
</tr>
<tr>
<td>Reduction</td>
<td>4.07</td>
<td>43 moves</td>
<td>3.62</td>
</tr>
<tr>
<td>Scenario 2</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AMPs</td>
<td>26.8</td>
<td>13</td>
<td>14.1</td>
</tr>
<tr>
<td>cNAMPs</td>
<td>8.5</td>
<td>-</td>
<td>5.6</td>
</tr>
<tr>
<td>Reduction</td>
<td>3.15</td>
<td>13 moves</td>
<td>2.52</td>
</tr>
<tr>
<td>Scenario 3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

are presented in Table 3. The way cNAMPs confirm a movement decision with the target location has some small similarities to a two-phase commit protocol [35].

7. Comparative cNAMP and AMP Performance

We compare the greedy effects exhibited by AMPs and cNAMPs in Table 4 using the experiment design presented in Section 5. For each scenario the first and the second rows show results of AMP and cNAMP experiments respectively.

The results show that even simple negotiation in cNAMPs significantly reduces the number of movements and time to rebalance. cNAMPs do not make redundant movements during initial distribution, and all scenarios show at least three times faster initial balancing in comparison with AMPs, e.g. dropping from 60.4s to 16.5s in Scenario 1.

During rebalancing after an AMP/cNAMP termination, cNAMPs make far fewer redundant movements. The vast majority of experiments in scenarios 1 and 3 show that cNAMPs do not make redundant movements to rebalance, and the cNAMP rebalancing takes less than half of the time of AMP rebalancing, e.g. to rebalance 19 AMPs take 28.2s, and 19 cNAMPs take 7.8s in Scenario 2.
Table 5: AMP/cNAMP and Request/Response Messages

<table>
<thead>
<tr>
<th>Conf.</th>
<th>AMPs</th>
<th></th>
<th></th>
<th>cNAMPs</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No. Moves</td>
<td>Size, Mb</td>
<td>No. Req/Resp</td>
<td>No. Moves</td>
<td>cNAMPs</td>
<td>Total No. Move</td>
<td>Req/Resp Size, Mb</td>
</tr>
<tr>
<td>Sc. 1</td>
<td>106</td>
<td>954</td>
<td>78</td>
<td>35</td>
<td>113</td>
<td>36</td>
<td>315</td>
</tr>
<tr>
<td>Sc. 2</td>
<td>77</td>
<td>693</td>
<td>66</td>
<td>28</td>
<td>94</td>
<td>30</td>
<td>252</td>
</tr>
<tr>
<td>Sc. 3</td>
<td>28</td>
<td>252</td>
<td>26</td>
<td>11</td>
<td>37</td>
<td>12</td>
<td>99</td>
</tr>
</tbody>
</table>

cNAMPs require less execution time than AMPs. Mean cNAMP execution time is at least 1.6 times less than mean AMP execution time, e.g. mean execution time of 10 AMPs is 232s, whereas mean execution time of 10 cNAMPs is 142s in Scenario 3.

Figure 6 shows initial cNAMP distribution and system rebalancing after a cNAMP termination. Arrows show cNAMP movements, and as before we do not show the cNAMP movements from state U1 to state S1. Figure 6 should be compared with Figures 4 and 5 in Section 5. More examples of cNAMP movements are provided in [33]. The results show that cNAMPs only display location blindness (Section 3), which does not increase cNAMP execution time.

7.1. cNAMP Overhead

Table 5 presents the overhead of cNAMPs compared with AMPs, and we see that although cNAMPs send more messages (columns 2 and 6), most of them are small request/response messages. Therefore, overall AMP communication is larger than overall cNAMP communication (columns 3 and 9). As a consequence cNAMPs execute faster than AMPs (column 7 in Table 4).

8. Properties of Balanced Networks

To analyse the significance of greedy effects we need to establish the properties of balanced states. The properties and definitions developed in this section are essential for the proofs in Section 9, and are summarised in the Glossary. Note that the properties of balanced states apply to all AMPs, not only cNAMPs. The balanced state checker uses identical logic to the simulated and to real AMPs. Hence, it is not surprising that the predicted balanced states exactly reproduce simulated results for all scenarios considered. A homogeneous network is analysed as a special case of a heterogeneous network where the root location is taken as a singleton subnetwork, i.e. a subnetwork with one location.
So, we first describe empirical balanced state checker in Section 8.1 which is used to investigate balanced states, then present the *independent balance property* in Section 8.2, and discuss optimal and near-optimal balanced states in Sections 8.3 and 8.4 respectively. Furthermore, we characterise balanced states for homogeneous networks in Section 8.5 and heterogeneous networks in Section 8.6.

There is a similarity between AMPs balanced states and Nash equilibria. Participants in both Nash equilibrium and AMPs aim to get as much *profit* as possible. However, in game theory participants have a choice to play cooperatively or not, whereas AMPs/cNAMPs have no such choice. Though cNAMPs communicate more information than AMPs, cNAMPs are still *selfish*, and they never *care* about others’ profit but only about decreasing their own execution time. Another difference is that AMPs/cNAMPs do not predict or estimate other AMP/cNAMP behaviour, i.e. as soon as an AMP/cNAMP finds a location where it can reduce its execution time it moves.

### 8.1. Balanced State Checker

The properties of balanced states were investigated using a program to explore the states entered as AMPs are added to the system one by one. The program assumes that communication time is negligibly small relative to computation time, i.e. $T_{\text{comm}} \ll T_{\text{comp}}$. The sequence of stable states that are entered are all balanced.

The algorithm of the balanced state checker is as follows. To allocate a new AMP when a distribution of $x$ AMPs is given ($0 \leq x \leq \infty$), we add one AMP to each location. Then we calculate AMP relative speeds, and remove one AMP from each location except the one with the highest AMP relative speed. The resulting distribution is the distribution $x + 1$ AMPs. More detailed description and pseudocode are given in [33].

### 8.2. Independent Balance Property

The analysis of balanced states shows the following property.

**Property 1** (independent balance). For a balanced state, the relationship between the number of AMPs $x_i$ and $x_j$ on locations in any two subnetworks $i$ and $j$ is independent of the number of locations in those subnetworks and independent of the presence or absence of other subnetworks, subject only to the sum $x = x_i + x_j$ being constant.
Table 6: AMP Distribution in a Pair of Subnetworks for a Given Sum of AMPs

(a) Scenario A

<table>
<thead>
<tr>
<th>Available speed of locations</th>
<th>Number of AMPs</th>
<th>Available speed of locations</th>
<th>Number of AMPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>8</td>
<td>$S_1$</td>
<td>8</td>
</tr>
<tr>
<td>$S_2$</td>
<td>3</td>
<td>$S_2$</td>
<td>3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$S_i$</td>
<td>$x_i$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(b) Scenario B

<table>
<thead>
<tr>
<th>Available speed of locations</th>
<th>Number of AMPs</th>
</tr>
</thead>
<tbody>
<tr>
<td>$S_1$</td>
<td>8</td>
</tr>
<tr>
<td>$S_2$</td>
<td>3</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>$S_i$</td>
<td>$x_i$</td>
</tr>
</tbody>
</table>

(c) Scenario C

Independent balance property holds in all scenarios investigated and can be observed in the following experiments.

1) Distribute 11 AMPs over a heterogeneous network of two singleton subnetworks. Without loss of generality, assume that the distribution places 8 AMPs on the location of subnetwork 1 and 3 AMPs on the location of subnetwork 2 as Table 6(a) shows.

2) Add further singleton subnetworks to the system and also add enough AMPs to the new system so that balanced state is achieved and there are 11 AMPs distributed between singleton subnetworks 1 and 2. The distribution between singleton subnetworks 1 and 2 will always be 8 AMPs in subnetwork 1 and 3 AMPs in subnetwork 2 (Table 6(b)). It will never be distributed 7 AMPs in subnetwork 1 and 4 AMPs in subnetwork 2, or 9 AMPs in subnetwork 1 and 2 AMPs in subnetwork 2.

3) Adding any number of locations to a subnetwork will not change the distribution between a pair of locations from subnetworks 1 and 2 so long as the system is in a balanced state and a sum of AMPs in the pair of locations from subnetworks 1 and 2 is 11 (Table 6(c)).

Let there be two subnetworks, with the available speeds $S_1$ and $S_2$, and let each location of subnetworks 1 and 2 have $x_1$ and $x_2$ AMPs in a balanced state respectively. Then the following inequalities hold in any balanced state:
We distinguish two balanced states which a system can enter: optimal and near-optimal balanced states. In Sections 8.3 and 8.4 we investigate their properties and generalise some of the definitions given in [6].

8.3. Optimal Balance

An optimal balanced state is a state when locations with the same available speed have equal numbers of AMPs. The total number of AMPs $x$ in an optimally balanced network with $q$ subnetworks is: $x = \sum_{i=1}^{q} x_i N_i$, where there are $N_i$ locations in subnetwork $i$ and each location has $x_i$ AMPs.

Property 2 (singleton optimality). All balanced states which a network of singleton subnetworks enters are optimally balanced.

The singleton optimal property is a direct corollary of the optimal balanced state definition. Furthermore, from optimal balanced state definition and independent balance property it follows that:

1) finding optimal balance states of arbitrary heterogeneous networks only requires the finding of the optimal balance states for networks with the same number of singleton subnetworks:

$$k = \sum_{i=1}^{q} x_i, \quad k \in [1; +\infty).$$

(6)

2) any optimally balanced network is a composition of optimally balanced pairs of subnetworks.

Solving inequations (5) for a heterogeneous network of two subnetworks with available speeds $S_1$ and $S_2$, and $k = x_1 + x_2$, each location has the following number of AMPs:

$$x_i \approx \frac{S_i(k + 1)}{S_1 + S_2} - \frac{1}{2}, \quad i = 1, 2.$$

(7)

Here, by $\approx$ we mean rounding to the nearest value. If the fraction part of $x_i$ possesses the value exactly .5, then it is rounded up for a location with
a higher available speed, and it is rounded down for a location with a lower available speed. To indicate the rounding to the nearest value in the paper we use double square brackets, \([\square]\).

This analysis of balanced and optimal balanced states enables testing of an assumption made in [6], i.e. it is said that AMPs tend to have equal AMP relative speed in optimal balance. However, the observations show the following results:

**Lemma 3.** An AMP relative speed on a faster location tends to be slightly lower than the mean AMP relative speed; and an AMP relative speed on a slower location tends to be slightly higher than the mean AMP relative speed.

The proof of Lemma 3 is presented in Appendix A.

8.4. Near-Optimal Balance

For any optimal balanced state, the *optimal number of AMPs* for a subnetwork is the number of AMPs on each location in the subnetwork.

*Nearest upper* (lower) *optimal balanced state* is the optimal balanced state which the system enters by adding (removing) the minimum number of AMPs.

We define a *near-optimal balanced network* be the network where all subnetworks except for one have their optimal number of AMPs. The locations of this subnetwork have either the optimal number of AMPs or one less than the optimal number. This subnetwork is uniquely determined by being the subnetwork with slowest AMP relative speed in the nearest upper optimal balanced state.

**Property 4** (consecutive optimality). *If a system with a total of x AMPs is optimally balanced, and the subnetwork with the highest AMP relative speed is a singleton, then the system with a total of x + 1 AMPs is also optimally balanced.*

The consecutive optimal property is derived from the near-optimal balance definition as a corollary.

On the basis of the independent balance property (Section 8.2) we conclude that *a system has no balanced states other than optimal and near-optimal balanced states*. That is, at most one subnetwork can be in a near optimal balanced state. If two networks were simultaneously in a near optimal balance state, then there would be an immediate transfer of AMPs from the slower to the faster subnetwork (in terms of AMP relative speed).
As the near-optimally balanced subnetwork is uniquely determined for each optimally balanced state, we can identify a subnetwork which changes its number of AMPs when we add/remove an AMP:

1) when we add an AMP to an optimally balanced network, the number of AMPs in the subnetwork which would have the highest AMP relative speed if we add one AMP to each location increases by one;
2) when we remove an AMP from an optimally balanced network, the number of AMPs in the subnetwork with the slowest AMP relative speed decreases by one.

8.5. Characterizing Balanced States in a Homogeneous Network

Let a homogeneous network have \( N \) locations, the available speed of non-root locations be \( S \). As only a part of the root location capacity is available for an AMP execution, let the load factor, \( 0 < f < 1 \), define available resources for AMPs on the root location, i.e. available speed of the root location is \( f \cdot S \).

In an optimal balanced state on the basis of (7) the system has the following numbers of AMPs on the root, \( x_{rt} \), and non-root, \( x_{nrt} \), locations:

\[
\begin{align*}
    x_{rt} & \approx \frac{f(2k + 1) - 1}{2(f + 1)}, \\
    x_{nrt} & \approx \frac{2k - f + 1}{2(f + 1)}.
\end{align*}
\]

(8)

(9)

In a near-optimal balanced state the root location has \([\left\lfloor \frac{f(2k+1)-1}{2(f+1)} \right\rfloor]\) AMPs and non-root locations have either \([\left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor]\) or \([\left\lfloor \frac{2k-3f-1}{2(f+1)} \right\rfloor]\) AMPs. For further distinguishing non-root locations with a different number of AMPs, those which have an optimal number of AMPs, i.e. \([\left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor]\), we call heavy locations, and those which have one or two AMPs less than a heavy location, i.e. \([\left\lfloor \frac{2k-3f-1}{2(f+1)} \right\rfloor]\) and \([\left\lfloor \frac{2k-5f-3}{2(f+1)} \right\rfloor]\), we call light and very light locations respectively.

8.6. Characterizing Balanced States in a Heterogeneous Network

Solving equations (6) and (7) for three or more subnetworks gives the following approximation of AMP distribution in optimal balanced state:

\[
\begin{align*}
    x_i & \approx \frac{S_i(2k + q)}{2\sum_{j=1}^{q} S_j} - \frac{1}{2}, \\
    i &= 1, 2, ..., q.
\end{align*}
\]

(10)
here, we round to the nearest value. According to discussion in Section 8.4, numbers of AMPs on locations in a near-optimal balanced state coincides with the numbers in the nearest upper optimal balanced state, except the subnetwork with the slowest AMP relative speed. Locations of this subnetwork have either number of AMPs given by (10) or one AMP less.

9. cNAMP Greedy Effect Analysis

This section examines the cost of cNAMP location blindness, by predicting the maximum number of redundant movements in homogeneous networks (Section 9.1) and heterogeneous networks (Section 9.2). To estimate the number of redundant movements, we analyze the conditions under which cNAMPs transfer. Figure 7 shows the cNAMP cost model. A network with AMPs, and hence with cNAMPs, enters one of two balanced states: optimal and near-optimal balance. We analyse the maximum number of movements and its probability after a cNAMP termination from optimally and near-optimally balanced homogeneous and heterogeneous networks.

9.1. Homogeneous Network

**cNAMP Termination in an Optimally Balanced Network.** From (8) and (9), in an optimal balanced state the root location has \[ \left\lfloor \frac{(2k+1)-1}{2(f+1)} \right\rfloor \] cNAMPs and every non-root location has \[ \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor \] cNAMPs.

**Theorem 5.** There is no greedy effect when a cNAMP terminates in an optimally balanced homogeneous network.

The proof is given in Appendix B. The proof shows that the number of cNAMPs on locations with the same available speed must differ by at least two before cNAMPs will move between them. We call this condition the minimum difference criterion.

**cNAMP Termination in a Near-Optimally Balanced Network.** In near-optimal balance a system has \[ \left\lfloor \frac{(2k+1)-1}{2(f+1)} \right\rfloor \] cNAMPs on a root location, and either \[ \left\lfloor \frac{2k-3f-1}{2(f+1)} \right\rfloor \] or \[ \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor \] cNAMPs on non-root locations (Section 8.5).

**Theorem 6.** The greedy effect causes at most one redundant movement, when a cNAMP terminates in a near-optimally balanced homogeneous network.
Cost Model Components:

\[ T_h = \frac{W_r}{R_h} \]  \hspace{1cm} (11)
\[ T_n = \frac{W_r}{R_n} \]  \hspace{1cm} (12)

- \( f \) - the load factor on the root location;
- \( R_h \) - the current location relative speed;
- \( R_n \) - the new location relative speed;
- \( T_h \) - execution time on the current location;
- \( T_n \) - execution time on the new location;
- \( W_r \) - the work remaining;
- \( x \) - the total number of cNAMPs;
- \( x_{nrt} \) - the number of cNAMPs on a non-root location;
- \( x_{rt} \) - the number of cNAMPs on the root location;

Figure 7: cNAMP Cost Model Components

Proof of Theorem 6 follows directly from Lemma 7.

**Lemma 7.** A redundant movement occurs only in a case of a cNAMP termination in near-optimal balance on the root location which is discovered first by a cNAMP from a light location.

Recall that a light location has one cNAMP less than the optimal number (Glossary). The proof of Lemma 7 again proceeds by case analysis on the location where termination occurs, and where the first movement is initiated [33].

**Probability of the Greedy Effect after a cNAMP termination.** The greedy effect occurs only when a cNAMP terminates from the root location in a near-optimal balanced state, and a cNAMP from a light location discovers better opportunity for execution first. We present the probability calculation and experiment description in Appendix C. It shows that the probability of the greedy effect increases as the number of cNAMPs, \( k \), increases. It also increases as the number of locations, \( N \), decreases. For example, a network of three locations where \( k = 200 \) cNAMPs and the load factor \( f = 0.45 \) the probability does not exceed 19%; in a network of more than 10 locations with the same parameters the probability does not exceed 10%. 

25
Summary. In the analysis of a cNAMP movement on homogeneous networks, we have defined dependence between number of cNAMPs and locations in optimal and near-optimal balanced states. The analysis shows the following results of cNAMP behaviour after a cNAMP termination in an optimal balanced network:

1) there is no greedy effect (Theorem 5);
2) cNAMPs do not move, when a system is in a near-optimal balanced state (minimum difference criterion).

When a cNAMP terminates in a near-optimal balanced network the greedy effect only occurs after a cNAMP termination at the root location, this happens in one case and causes only one redundant movement (Theorem 6). The probability of this movement is less than 10% in a network of more than 10 locations.

9.2. Heterogeneous Network

cNAMP Termination in Optimally and Near-Optimally Balanced Networks. The analysis is made for a heterogeneous network of $q$ subnetworks one of which is the root location singleton subnetwork.

Theorem 8. The number of redundant movements in a heterogeneous network after a cNAMP termination does not exceed $q - 1$.

Lemma 9. A system makes at most $q - 2$ redundant movements after a cNAMP termination from an optimally balanced heterogeneous network.

Lemma 10. A system makes at most $q - 1$ redundant movements after a cNAMP termination from a near-optimally balanced heterogeneous network.

The proofs of Theorem 8, Lemma 9 and Lemma 10 again proceed by case analysis on the state (i.e. optimal or near-optimal balanced) and location where termination occurs, and then where the first movement is initiated [33].

Probability of the Greedy Effect after a cNAMP Termination. The calculation of the probability of the maximum number of redundant movements after a cNAMP termination from optimally balanced network in [33] shows that the median probability of $q - 2$ redundant movements after a cNAMP termination from optimally balanced heterogeneous network does not exceed 1%.
As it is difficult to estimate mean and maximum values of the probability for a near-optimally balanced subnetwork, the results of conducted experiments show that the probability is less than 30%, and it rapidly decreases as the number of subnetworks increases [33].

Summary. The analysis of cNAMP movements after a cNAMP termination from optimally balanced heterogeneous network shows that: 1) cNAMPs never move from locations which in optimal balance have higher cNAMP relative speed to locations which in optimal balance have lower cNAMP relative speed [33]; 2) a system makes at most \( q - 2 \) redundant movements to rebalance (Lemma 9); 3) the probability median value of maximum number of redundant movements does not exceed 1%.

Results of analysis after a cNAMP termination from near-optimal balance are as follows: 1) to rebalance a system makes at most \( q - 1 \) redundant movements (Theorem 8); 2) in the experiments the probability of the maximum number of movements does not exceed 30% and rapidly decreases with the increase of a number of subnetworks.

10. Conclusion and Future Work

10.1. Summary

In the current paper we have identified two types of greedy effects in AMP systems: location thrashing causes additional movements and increase in AMP execution time; location blindness causes only additional movements, as all transferred AMPs improve their execution environment. Both greedy effects appear in the load balancing literature (Section 3).

We have simulated the greedy effects in an AMP implementation (Section 4), and shown that each AMP makes on average five redundant movements during execution for the scenarios considered. Although greedy effects have limited impact on networks with a small number of AMPs, few locations, or small AMPs, their effects increase as any of these factors scale. The analysis of the redundant movement types and the reasons they occur have showed that redundant movements are mainly caused by location thrashing (Section 5).

To reduce location thrashing we have introduced the concept of negotiating AMPs, described and implemented AMPs that negotiate with a competitive scheme, so called cNAMPs. By negotiation we only mean a coordination among competitive and self-interested agents. The key differences between
cNAMPs and AMPs are as follows: before the movement a cNAMP sends a request to the target location; and each location has two values for the number of cNAMPs, one that is used by local cNAMPs, and another that is published to other locations (Section 6).

cNAMP simulation results show that cNAMPs exhibit only the location blindness. cNAMPs do not make redundant movements during initial distribution, and all scenarios show at least three times faster initial balancing in comparison with AMPs, e.g. dropping from 60.4s to 16.5s in Scenario 1. During rebalancing after an AMP/cNAMP termination, cNAMPs make far fewer redundant movements, and the cNAMP rebalancing takes less than half of the time of AMP rebalancing, e.g. to rebalance 19 AMPs take 28.2s, and 19 cNAMPs take 7.8s in Scenario 2. cNAMPs require less execution time than AMPs. Mean cNAMP execution time is at least 1.6 times less than mean AMP execution time, e.g. mean execution times of 10 AMPs and 10 cNAMPs in Scenario 3 are 232s and 142s respectively (Section 7).

To analyse the significance of the greedy effect we have established the properties of balanced states. Here, we have analysed AMPs as the general case and described properties including independent balance, singleton optimality, consecutive optimality, and characterised optimal and near-optimal balanced states for homogeneous and heterogeneous networks (Section 8).

We have established the significance of the greedy effect by predicting the worst case (maximum number) of redundant movements after a cNAMP termination from a network of $q$ subnetworks. The results show that a difference in the number of cNAMPs needs to be at least two before a movement will occur between locations of a same subnetwork. A system with $q$ subnetworks makes at most $q - 2$ redundant movements after a cNAMP termination from optimally balanced network with median probability less than 1% (Lemma 9). The number of movements after a cNAMP termination in a network of $q$ subnetworks does not exceed $q - 1$ (Section 9).

We have assembled a set of consistent and precise definitions about the behaviour of collections of AMPs and cNAMPs. These definitions are essential for reasoning about AMP/cNAMP behaviour, and often generalise the definitions given in earlier papers [2, 6] (Glossary).

10.2. Future Work

The current paper assumes that the network is totally reliable, and has a flat structure, i.e. all location are equidistant from one another in a fully connected graph (a LAN). The available speed at a location is constant,
and is shared equally by the AMPs or cNAMPs at that location. AMP and cNAMP simulating models are constructed and analysed on a LAN. Other directions include relaxing the assumptions, e.g. network reliability, and using game theoretic techniques to analyse AMP behaviour. We also plan to adapt the cNAMP cost model and investigate cNAMP properties in a wide area networks where the flat communication assumption does not hold.

Appendix A. Proof of Lemma 3

**Lemma 3.** An AMP relative speed on a faster location tends to be slightly lower than the mean AMP relative speed; and an AMP relative speed on a slower location tends to be slightly higher than the mean AMP relative speed.

**Proof.** This can be observed from the following analysis. First, it is important to recall that though a system is equilibrium in an optimal balanced state, AMP relative speeds on locations from different subnetworks are not the same, because the number of AMPs on a location is integer.

According to the assumption in [6], i.e. \( \frac{S_1}{x_1} \approx \frac{S_2}{x_2} \), and the optimal balance property a location of subnetwork 1 must have \( \left\lceil \frac{S_1 k}{S_1 + S_2} \right\rceil \) AMPs. Hence, AMP relative speed on the basis of (2) must be

\[
R_1' = \frac{S_1}{\left\lceil \frac{S_1 k}{S_1 + S_2} \right\rceil}. \tag{A.1}
\]

However, the number of AMPs on a location is given by (10), and an AMP relative speed on a location of subnetwork 1 is

\[
R_1 = \frac{S_1}{\left\lceil \frac{S_1 k}{S_1 + S_2} + \frac{S_1 - S_2}{2(S_1 + S_2)} \right\rceil}. \tag{A.2}
\]

As \( S_1 > S_2 > 0 \) and \( \frac{S_1 - S_2}{S_1 + S_2} < 1 \), then \( 0 < \frac{S_1 - S_2}{2(S_1 + S_2)} < 0.5 \). Thus, \( \frac{S_1 - S_2}{2(S_1 + S_2)} \) increases the number of AMPs assumed in [6] by nought or one during the rounding. Therefore, an AMP relative speed on a location of subnetwork 1 either coincides with the AMP relative speed assumed in [6] or is slightly slower. The same analysis sequence is made to investigate AMP relative speed in subnetwork 2.

The difference between AMP relative speeds on locations from different subnetworks is less when the difference in the available speeds of locations is small; and it decreases with the increase of the total number of AMPs. \( \Box \)
Appendix B. Proof of Theorem 5

Theorem 5. There is no greedy effect when a cNAMP terminates in an optimally balanced homogeneous network.

Proof. The proof proceeds by case analysis on the location where termination occurs, and where the first movement is initiated.

1) Termination at a Non-root Location. From (8) and (9) after the cNAMP termination, the system has \( \left\lfloor \frac{f(2k+1)-1}{2(f+1)} \right\rfloor \) cNAMPs on the root location, \( \left\lfloor \frac{2k-3f-1}{2(f+1)} \right\rfloor \) cNAMPs on the non-root location where termination occurred (the light location), and \( \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor \) cNAMPs on the remaining non-root locations (heavy locations). Hence, the system is in the near-optimal balanced state. We analyse possible movements which may occur from the root and heavy locations to the light location.

- **cNAMP movement from the root to the light location.** The main rule on the basis of which cNAMPs make the decision about a movement is represented in (4). According to it, to make the decision the cNAMP needs to know its execution time on the current location and its execution time on the new location after a cNAMP arrival.

First, on the basis of (11) and (2) we calculate cNAMP execution time on the root location before a cNAMP movement:

\[
T_h = \frac{W_r}{S} \cdot \left\lfloor \frac{f(2k+1)-1}{2(f+1)} \right\rfloor . \tag{B.1}
\]

The light location becomes a heavy location after a cNAMP arrival, and has \( \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor \) cNAMPs. According to (12) and (2) execution time is

\[
T_n = \frac{W_r}{S} \cdot \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor . \tag{B.2}
\]

Substituting (B.1) and (B.2) in (4) gives

\[
\frac{W_r}{S} \cdot \left\lfloor \frac{f(2k+1)-1}{2(f+1)} \right\rfloor > \frac{W_r}{S} \cdot \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor + T_{comm},
\]

30
which can be simplified to

\[
\frac{W_r}{S} \left( \left\lfloor \frac{2kf + f - 1}{2(f + 1)} \right\rfloor - \left\lfloor \frac{2k - f + 1}{2(f + 1)} \right\rfloor \right) > T_{\text{comm}}. \tag{B.3}
\]

To satisfy (B.3) at least \( \left\lfloor \frac{2kf + f - 1}{2(f + 1)} \right\rfloor - \left\lfloor \frac{2k - f + 1}{2(f + 1)} \right\rfloor \) must be positive. However, \( 2kf < 2k \) and \( f - 1 < 1 - f \), which means that \( \left\lfloor \frac{2kf + f - 1}{2(f + 1)} \right\rfloor - \left\lfloor \frac{2k - f + 1}{2(f + 1)} \right\rfloor \leq 0 \). Thus, cNAMPs from the root location never move to a non-root location in a near-optimal balanced state.

- \textit{cNAMP movement from a heavy to the light location.} cNAMP execution time on a heavy location is

\[
T_h = \frac{W_r}{S} \cdot \left\lfloor \frac{2k - f + 1}{2(f + 1)} \right\rfloor. \tag{B.4}
\]

After a cNAMP movement to the light location, the light location becomes heavy and execution time on it is (B.2). Substituting (B.2) and (B.4) in (4) gives the following cNAMP movement condition:

\[
\frac{W_r}{S} \cdot \left\lfloor \frac{2k - f + 1}{2(f + 1)} \right\rfloor > \frac{W_r}{S} \cdot \left\lfloor \frac{2k + f - 1}{2(f + 1)} \right\rfloor + T_{\text{comm}}
\]

or

\[
T_{\text{comm}} < 0. \tag{B.5}
\]

In fact, (B.5) shows that the number of cNAMPs on locations with the same available speed must differ by at least two before cNAMPs will move between them. We call this condition the minimum difference criterion.

We conclude that after a cNAMP termination at a non-root location, there are no cNAMP movements between locations.

2) \textit{cNAMP Termination at the Root Location.} After the termination the root location has \( \left\lfloor \frac{f(2k-1)-3}{2(f+1)} \right\rfloor \) cNAMPs, and non-root locations have \( \left\lfloor \frac{2k-f+1}{2(f+1)} \right\rfloor \) cNAMPs, again from (8) and (9). Applying a similar analysis we find that only one movement may occur, and hence there is no greedy effect.
Appendix C. Probability of the Greedy Effect after a cNAMP Termination from Homogeneous Network

The greedy effect occurs only when a cNAMP terminates from the root location in a near-optimal balanced state, and a cNAMP from a light location discovers better opportunity for execution first, which means that the following condition from (4) must be satisfied:

$$\frac{W_r}{S} \cdot \left[ \frac{2k - 3f - 1}{2(f + 1)} \right] > \frac{W_r}{S} \cdot \left[ \frac{f(2k + 1) - 1}{2(f + 1)} \right] + T_{\text{comm}}. \quad (C.1)$$

Thus, the greedy effect probability in a homogeneous network is a product of the probability of a cNAMP termination from the root location and the probability of a discovery a better opportunity for execution first by a cNAMP from a light location:

$$P = P_{\text{termR}} \cdot P_l. \quad (C.2)$$

To calculate the probability of a cNAMP termination at the root location, assume that cNAMP execution time on locations follows an exponential distribution. The mean cNAMP execution time on the root, heavy and light locations is given by:

$$T_{\text{loc}} = \frac{W \cdot x_{\text{loc}}}{S_{\text{loc}}} = \frac{W}{R_{\text{loc}}}. \quad (C.3)$$

Hence, the rate of cNAMP terminations at the root, heavy and light locations is:

$$\nu_{\text{loc}} = \frac{R_{\text{loc}}}{W}. \quad (C.3)$$

Assume, that there are $N_l$ light locations and $N_h$ heavy locations in the system. Then probability that a cNAMP terminates at the root location is

$$P_{\text{termR}} = \frac{\nu_{\text{root}}}{\nu_{\text{root}} + \nu_h + \nu_l}$$

or substituting (C.3)

$$P_{\text{termR}} = \frac{R_{\text{root}}}{R_{\text{root}} + N_h R_h + N_l R_l}. \quad (C.4)$$

cNAMPs from non-root locations have an equal probability of discovering a better opportunity for execution first. The total number of cNAMPs on non-root locations is

$$x_{\text{nonroot}} = N_h x_h + N_l x_l. \quad (C.5)$$
The probability that a cNAMP from a light location is the first to discover a better opportunity for execution is the ratio of the total number of cNAMPs on light locations to the total number of cNAMPs on non-root locations:

\[ P_l = \frac{N_l x_l}{N_l x_l + N_h x_h}, \quad \text{(C.6)} \]

Substituting (C.4) and (C.6) in (C.2), we get the following probability of the greedy effect after a cNAMP termination from a near-optimally balanced homogeneous network:

\[ P = \frac{R_{\text{root}}}{R_{\text{root}} + N_h R_h + N_l R_l} \cdot \frac{N_l x_l}{N_l x_l + N_h x_h}, \quad \text{(C.7)} \]

To investigate the range of values that \( P \) can take, we calculate \( P \), changing the total number of locations from 3 to 50, the number of light locations from 1 to \( N - 2 \), the load factor, \( 0.45 < f < 0.55 \), and the number of cNAMPs, \( k \), from 1 to 200. In the calculation we consider only cases when \( N_h \cdot N_l \neq 0 \), because a system must have both heavy and light locations in order that the greedy effect can occur. The root and light locations must have at least one cNAMP.

The probability of the greedy effect increases as the number of cNAMPs, \( k \), increases. It also increases as the number of locations, \( N \), decreases. The maximum probability in our calculations is 19% and occurred in a homogeneous network of three locations where \( k = 200 \) cNAMPs and the load factor \( f = 0.45 \). In a homogeneous network of more than 10 locations with the same parameters the probability never exceeded 10%.

Glossary

Table C.7 assembles the main notions defined in the paper. In column Source 1 denotes a definition from [6], 2 a generalisation of a definition from [6], and 3 a more precise definition than in [6]. In column Related Concept we provide lists of related concepts to clarify the difference and similarity of the concepts. Column Section gives the numbers of the sections where the notions are introduced in the paper.


33
The actual load is the number of executing cNAMPs on a location. It is used for local cNAMP calculations.

An AMP relative speed, \( R \), is an available speed, \( S \), equally divided between the AMPs at a location, \( x_{\text{loc}} \), i.e. \( R = \frac{S}{x_{\text{loc}}} \).

Autonomous mobile programs (AMPs) are mobile agents that improve execution efficiency by managing load balancing; AMPs are aware of their resource needs, sensitive to the execution environment and periodically seek a better location to reduce execution time.

The available speed of a location is the execution speed of a single AMP on that location, i.e. \( S = (\text{CPU speed}) \cdot (1 - \text{non AMP load}) \).

In a balanced state no AMP can gain a greater AMP relative speed by moving.

cNAMPs are negotiating AMPs with a competitive scheme, which announce their intentions to move and compete with each other for opportunity to transfer to the new location.

The committed load represents the actual load of a location, together with the cNAMPs that have received confirmation to transfer to the location. It is used by remote load servers.

A communication cost is the number of AMP movements during a rebalancing.

Consecutive optimal property. If a system with a total of \( x \) AMPs is optimally balanced, and the subnetwork with the highest AMP relative speed is a singleton, then the system with a total of \( x + 1 \) AMPs is also optimally balanced.

---

1 Termed average relative speed in [6].
2 Termed relative speed in [6].

Greedy effects are the result of a non-optimal AMP rebalancing which differs from the optimal rebalancing in having additional redundant movements, and is a result the AMP making a locally optimal choice.

A heavy location is a location with the optimal number of AMPs.

A heterogeneous network is a set of locations with different available speeds.

A homogeneous network is a set of locations with the same available speed, except the root location, which may have different, because of the communication with the remote processes shipping from it.

Independent balance property. For a balanced state, the relationship between the number of AMPs \( x_i \) and \( x_j \) on locations in any two subnetworks \( i \) and \( j \) is independent of the number of locations in those subnetworks and independent of the presence or absence of other subnetworks, subject only to the sum \( x = x_i + x_j \) being constant.

A light location is a location which has one AMP less than the optimal number of AMPs in a heavy location.

A load server on a location collects network state information to reduce AMP coordination time.

Location blindness is the greedy effect resulting from an AMP’s lack of precise information about the remaining execution time of other AMPs.

Location thrashing is the greedy effect resulting from an AMP’s lack of information about other AMPs intending to move to the same location.

Minimum difference criterion. The number of cNAMPs on locations with the same available speed must differ by at least two before cNAMPs will move between them.

---


In near-optimal balance all subnetworks except for one have their optimal number of AMPs. The locations of this subnetwork have either the optimal number of AMPs or one less than the optimal number. This subnetwork is uniquely determined by being the subnetwork with the highest relative speed in the nearest lower optimal balanced state.

<table>
<thead>
<tr>
<th>Description</th>
<th>Location</th>
<th>Available Speed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nearest upper (lower) optimal balanced state is the optimal balanced state which the system enters by adding (removing) the minimum number of AMPs.</td>
<td>near-opt. balance, optimal balance</td>
<td>8.4</td>
</tr>
<tr>
<td>In optimal balance locations with the same available speed have equal numbers of AMPs.</td>
<td>balanced state, near-opt. balance</td>
<td>8.3</td>
</tr>
<tr>
<td>For any optimal balanced state, the optimal number of AMPs for a subnetwork is the number of AMPs on each location in the subnetwork.</td>
<td>heavy location, optimal balance, subnetwork</td>
<td>8.4</td>
</tr>
<tr>
<td>An optimal rebalancing is a sequence of AMP movements that are the minimum number of AMP movements needed to enter a stable state.</td>
<td>greedy effects</td>
<td>3</td>
</tr>
<tr>
<td>The root location is the location where all AMPs start.</td>
<td>heavy location, light location</td>
<td>3.1</td>
</tr>
<tr>
<td>Singleton optimal property. All balanced states which a network of singleton subnetworks enters are optimally balanced.</td>
<td>balanced state, optimal balance, singleton subnetwork</td>
<td>8.3</td>
</tr>
<tr>
<td>A singleton subnetwork is a subnetwork with one location.</td>
<td>subnetwork</td>
<td>8</td>
</tr>
<tr>
<td>In a stable state no AMP can reduce its execution time by moving.</td>
<td>balanced state</td>
<td>1</td>
</tr>
<tr>
<td>A subnetwork is a set of locations with identical available speeds.</td>
<td>heterogen. network, homogen. network</td>
<td>1</td>
</tr>
<tr>
<td>A very light location is a location which has two AMPs less than the corresponding heavy location.</td>
<td>heavy location, light location</td>
<td>8.5</td>
</tr>
</tbody>
</table>

*It is either called initiating location or first location in [6]*


