

Pattern-based Management in Dynamic Wireless Networks

Zoltan Lajos Kis (Budapest University of Technology and Economics, Hungary)

Johan Nielsen (Ericsson Research, Sweden)

Abstract—In this paper we present our results of enhancing a promising decentralised network management approach, called pattern-based management, to work efficiently in wireless, dynamic networks. The pattern-based approach has previously proven efficient and useful in wired networks, especially for monitoring network resources. We show here that with our enhancements pattern-based management will also function efficiently in wireless, dynamic networks. We validate these findings with simulations results run in ns-2. We also show that with a proposed modification to how ARP is implemented in simulators and devices today an even higher level of efficiency is achieved.

Index Terms—Pattern-based network management, distributed management

I. INTRODUCTION

The pattern-based management approach [1][2][3] is a decentralised management approach based on the use of graph traversal algorithms to control and coordinate the processing and aggregation of management information inside the network. A key feature of the pattern-based approach is its ability to separate the management instructions or requests being sent into the network from how these instructions or requests are being propagated and aggregated in the network. The paradigm achieves this through the development of two important concepts: the navigation pattern and the aggregator. The former represents the generic graph traversal algorithms that implement distributed control, while the latter implements the computations required to realise the task.

However, almost all previous work performed on pattern-based management have assumed fixed, wireline networks with no or low level of node and link failures. Furthermore, in a wired network each node knows what

active links it has and who its neighbours are.

In this paper we present our work where we have modified and enhanced the pattern-based management approach to be sufficiently robust to work efficiently in wireless, dynamic networks, where nodes and networks move around and links form and break ad hoc. These enhancements includes mechanisms for detecting the neighbours as well as keeping active links alive while waiting for a reply on the instruction or query being sent out into the network. They also include mechanisms to detect when links break, for example if a node moves or dies, and ways to aggregate the response via alternative routes.

We present simulation results validating that our different enhancements greatly improves how pattern-based management will work in a wireless, dynamic environment. During our simulations we detected that the current standard implementation of ARP, designed for wired media, significantly hampered the results in our simulations, but when we implemented the proposed solution in [9] the results improved greatly. Finally we will conclude the results achieved and propose directions for future work.

II. PREVIOUS WORK ON PATTERN-BASED MANAGEMENT

Pattern-based management is an approach to distributed management that aims at overcoming the limitations of centralized management. Its goal is to build scalable, robust and adaptable management systems. The Pattern-based management paradigm is based on the use of graph traversal algorithms to control and coordinate the processing and aggregation of management information inside the network. This aggregation is done in a parallel, asynchronous fashion across the network, whereby all nodes contribute to the computation [4]. From the perspective of a network manager, a pattern provides the means to “diffuse” or spread the computational process over a large set of

This paper describes work undertaken in the context of the Ambient Networks - Information Society Technologies project, which is partially funded by the Commission of the European Union. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the Ambient Networks Project.

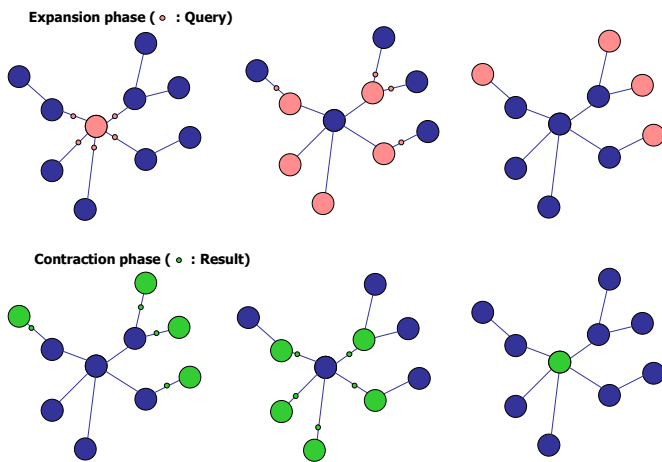


Figure 1 – The two phases of the echo pattern

nodes.

The main benefits of pattern-based management are that it separates the semantics of the task from the details of the distributed execution; it enables building scalable management systems; it facilitates management in dynamic environments; and it does not require “a priori” knowledge of the network topology.

Previous work has identified the Echo pattern to be useful for distributed monitoring of network resources [1][2]. The Echo pattern has a two-phase operation (Figure 1). The first phase is expansion phase, where the management query or instruction (called explorer) is propagated from the originator – e.g. a management node – to all nodes of the network on an instantly-built spanning-tree. When a node receives the explorer for a first time it forwards this explorer on its all direct links except the link the explorer arrived on. If a second explorer message arrives at a node that already has received one explorer message, this node sends a reply to it. The reply states that the node has already received an explorer message and hence there will be no further reply sent on this link. The node also performs locally the management operations specified in the explorer message.

Once the explorer message has reached the leaves of the spanning tree, i.e. the explorer message cannot propagate any further, the second phase will start. The second phase is the contraction phase, where management information is retrieved on the branches of the spanning-tree. In this contraction phase a node waits for replies on the explorer message on all links it sent the explorer message on, and when it has received all replies it will aggregate its own result with the result received before it sends the aggregated result to the node from whom it received the first explorer message.

An example of such a management query could be

“give me the top 5 links with the highest traffic load in the network”. This query is propagated in an explorer message to all nodes in the network by using the Echo pattern, and the answer is aggregated back to the initial node. Furthermore, during the aggregation phase, nodes receiving aggregates can further aggregate the information sent upstream by only transmitting the top 5 flows of all the flows this node has received information about. This way the processing load can be distributed over the network as well as the communication load.

For the evaluation of the pattern-based management paradigm, the SIMPSON simulator was developed by Stadler et al. [5]. SIMPSON was originally developed to evaluate the scalability of the paradigm itself. Thus it was written to simulate large-scale wired networks. It has been shown with simulations in [6] that a simple management function can be run in about 18 seconds using the pattern-based management paradigm in an Internet-sized network, while the same task would require more than five days of continuous operation of a centralised management system, where a central node would query all nodes in the Internet sequentially.

III. ROBUST PATTERNS

Our work was focused on how we can adapt the pattern-based management paradigm so it would work in a wireless, dynamic environment, where nodes, links and even networks will appear and disappear on the fly, i.e. to develop patterns that are robust enough to work in such an environment. These robust patterns are enhancements added to the original pattern concept, which will enable them to function in dynamic, wireless networks, on the network-layer.

A. Robustness Requirements

As mentioned above, pattern-based management has been developed for fixed, wired networks, where no or very few link, packet or node errors occur and where every node knows who their actual active neighbours are. However, in a wireless, dynamic environment, nodes and networks are connected to each other using wireless accesses. Furthermore, nodes and networks move around in a highly dynamic manner, breaking network links and setting up new links ad-hoc. Nodes might also die without any warning. This implies that explorer messages or the results might be lost while the explorer message has been transmitted over the link but before the reply message has been transmitted over the link, or that the node or network has moved and changed point of attachment to the rest of the network. Furthermore, since wireless links are used the

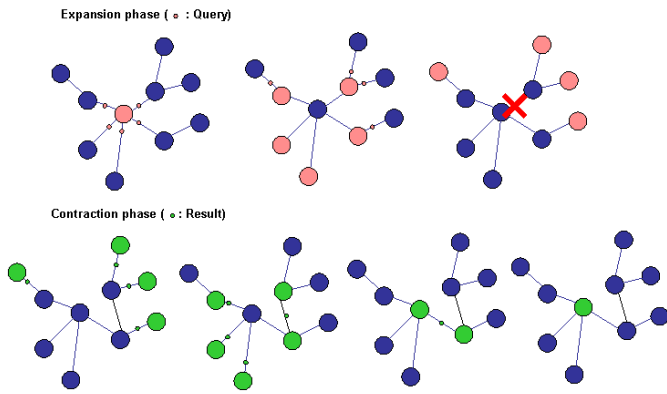


Figure 2: Robust patterns adapting to changing environment

transmitting node might not know what other nodes are within transmission range and hence what nodes can and will receive an explorer message.

Figure 2 shows one such occasion, where a link breaks between two nodes while the explorer message is still propagating through the spanning tree but before the answer has been sent back to the original node. Our work presented here focuses on how the pattern-based paradigm can be made robust to be able to cope and work in this wireless, dynamic environment.

B. Pattern Enhancements

In the original patterns when a node receive the same explorer message for a second or consecutive time this node replies to the sender and tells the sender that this node already has received this explorer message and the sender node should hence not expect an answer from this node. In a wireless environment, where a node might not know of its neighbours, we do the opposite. We broadcast the explorer message, and if a node receives the explorer message for the first time it replies with an acknowledgement to the sending node that it has received the explorer message and that the sending node can expect an answer from this node. This way a sending node will know from whom it can expect answers from.

If a node receives the same explorer message for a second or consecutive time it will not acknowledge the reception of this explorer message, and hence this second sender node will not expect a reply from this node. However, the receiving node might store the existence of this secondary sender node, to be used as a potential alternative route back to the source node if the link between the first sender node and this node breaks before the result message has been transmitted. We will describe this further below.

Since nodes and networks might be mobile, and nodes and links might go up and down at any time, a link that

was valid when the explorer message was transmitted over it might not exist when the result is being gathered the node expecting an answer must be able to identify whether it should wait for an answer or if the link has gone down and no reply will ever come over this nonexistent link. We solve this by transmitting a keep-alive message regularly between the sender and receiver nodes. If a node does not receive some of these keep-alive messages the node will realise that the connection is broken. If this node is waiting for replies it will realise that there is no idea to wait for a reply and will send its aggregated reply to its parent.

If, on the other hand, the node realising it has lost its link is the receiver node that is waiting to transmit a result packet upstream it will realise that the link is lost and try to find alternative routes to send the result packet to the originating node. This is where it can be useful to store nodes who have sent copies of the same explorer message, in this situation the node will try to contact these alternative nodes to see if they (one of them) are still within transmission range and if so ask this node if it can forward the result to the originating node. In Figure 2 this is what happens, the link breaks after the explorer message has passed but before the result has been aggregated over the link. Here the upstream node realises no answer will ever come and transmits its aggregated result to its upstream node, while the downstream node realises that it has lost its upstream link and starts to look for alternative routes back to the originating node through which it can transmit its aggregated result.

If this node cannot setup a link to any of its alternative upstream nodes it will try to find any node within its transmission range that can forward its aggregated result message to the originating node. This might be the case if a network has moved a significant distance during the course of the pattern. However, this should be seen as a last resort since it is preferable if the node finds another node that has already seen the same explorer message and knows where to send it.

Please note that if the result is sent back another way than the explorer message was sent out over, these messages must be properly marked to allow the originating node to realise that this message has taken an alternative path and decide whether the information can still be used or not. Also, since messages that take an alternative route back might cause multiple messages to be transmitted over the same link, this might also affect the result of the previous aggregations made in the upstream intermediate nodes.

What we do here with finding alternative routes back

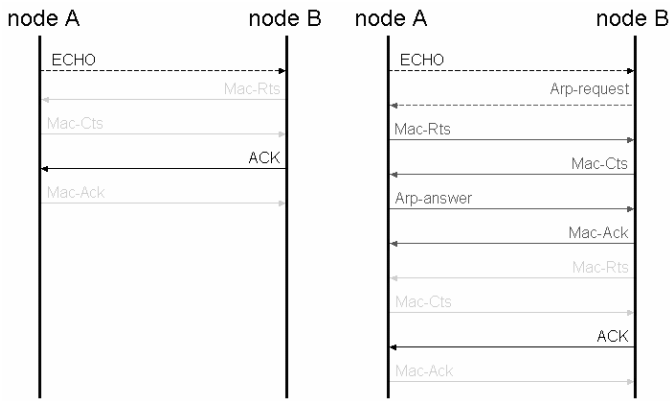


Figure 3 – Comparison of expected (left) and implemented (right) working of the echo propagation on the MAC layer level

to the originating node is to minimise as far as possible the impact of intermittent links in a wireless, dynamic environment, we try to provide the originating node with as much information as possible, but it is up to the originating node to decide whether this information still is valid or not.

C. Basic Robust Pattern

Our basic robust pattern is based on our robustness predefinitions, which means that our nodes do not know their neighbours initially, thus the Echo propagation has to be sent with a broadcast message – instead of unicast messages as in the original echo pattern. The neighbours receiving this echo have to send back an extra acknowledgement message, so the originator node can get a temporary list of neighbours. Because nodes do not know whether they are leaf nodes, they have to discover this by not receiving any acknowledgement messages. Finally, packets can be lost anytime, so watchdog timers have to be set up. This basic robust echo pattern will be referred to as *basic* in our results.

D. Application-level Enhancements

We also identified further application-level enhancements that we can use to achieve increased performance with our robust echo pattern. These enhancements are well known in other fields of wireless communication; however they have not been previously used within the pattern-based management context. Such an enhancement introduces a random back-off mechanism into the propagation of the echo expansion. Using this technique we can avoid broadcast messages sent at the same time to collide with each other. We also added this feature to the basic echo pattern. Since the random back-off periods increase the observed jitter on the end-to-end propagation delay, we refer to this version of the robust echo pattern as *jitter pattern*.

Further enhancement to the *jitter pattern* is possible

by introducing the broadcast counting mechanism. This mechanism had been previously proposed in the wireless broadcast storm problem field [8]. In an ordinary scenario a node would receive the same echo explorer message from multiple broadcast messages. It can be proven by elementary means that with every new broadcast of the same echo received, the effect of the node receiving the broadcasts sending out its own broadcast lessens. Because, the chance that this new broadcast message can reach a node not reached by any received broadcasts decreases. Thus, using this enhancement we can cancel some unnecessarily sent broadcast, decreasing the chance of broadcast collision.

Another enhancement is to introduce a means of retransmissions of messages to ensure their arrival to their destination. In our case this means that the echo propagates are periodically broadcast as long as the node is waiting for aggregates from its acknowledged neighbours. Also a minimal number of echoes have to be broadcast before announcing the node to be a leaf. When a neighbour receives such an echo, it can answer with another acknowledgement message that means it is still active and is still waiting for its aggregates to arrive. This solution minimizes the chance of both not receiving an acknowledgement as well as not receiving an aggregate. The robust echo pattern that incorporates both the jitter and this enhancement is referred to as *periodic pattern*.

E. ARP Problems

During the evaluation of our modification we identified that the current ARP implementation do not work the way it is expected to. During the echo pattern's expansion phase, the explorer is always propagated in a broadcast message, which is replied by a unicast

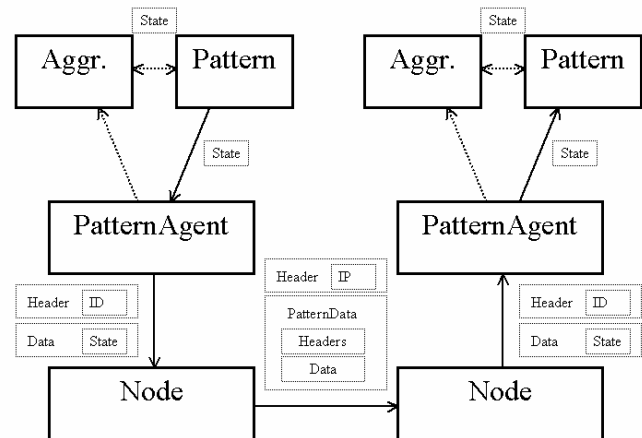


Figure 4 – Pattern-based extension to ns-2

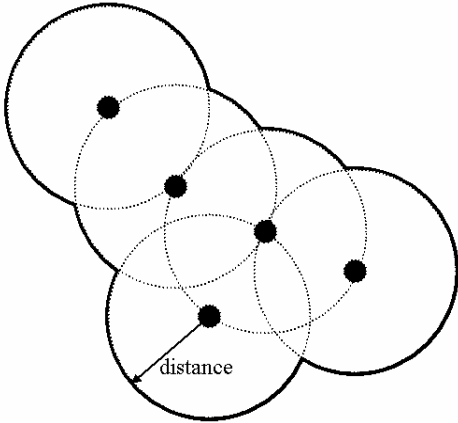


Figure 6 – Possible places for the sixth node in random scenario

message. After receiving the broadcast message, the address of the originator should already be known. In contrast to this, the ARP address resolution was invoked prior to sending the reply messages in our simulations.

The reason is that the current ARP implementations do not store IP address – MAC address mappings for received broadcast messages in the ARP table. Thus an ARP resolution is initiated prior to sending the reply to the originator as its MAC address is unknown at that point. The messages sent during the expected and the actual propagation and reply is shown in Figure 3. In the figure dotted and solid lines indicate broadcast and unicast messages; while the black and grey colours represent messages of the echo pattern, packets of the RTS/CTS mechanism and extra packets sent by the ARP implementations consecutively.

This implies that during the extraction phase numerous ARP resolutions take place at the same time. As the ARP address resolution is started with a broadcast packet – not guarded by the RTS/CTS mechanism – a number of resolutions in progress are set back by broadcast packets lost due to packet collisions. Furthermore the current ARP implementation can only store one address resolution packet at a time. This means that if a node has an address resolution in progress, while it receives another ARP packet, its own address resolution process would be dropped in favour of the incoming process. Dropped ARP resolutions mean dropped unicast messages, because those messages would never find their destination address.

On the whole the current ARP implementation causes lost reply messages

to the pattern. To solve this problem we modified the ARP implementation so that it inserts an entry into the ARP table whenever receiving broadcast messages. This solution was also proposed by [9] as a cross-layer enhancement for MANET routing protocols.

IV. SIMULATION SETUP

Previous work on pattern-based management has been done using the SIMPSON simulator [5]. That simulator has a limited capability and supports only the investigation of link failures and node mobility. Thus it is not usable in the evaluation of our proposed solutions. Therefore we decided to use the widely-accepted ns-2 network simulator [7].

A. Simulation Environment

To set up a simulation environment, we extended the ns-2 functionality with a pattern-based platform (Figure 4). The pattern-based management is simulated as a transport protocol agent that sits on top of the IP layer, and distributes pattern messages among patterns running on the node. For our wireless simulations we used the built-in MAC 802.11 medium access protocol and propagation model.

To implement actual aggregators and navigation patterns, one has to extend the functionality provided by the created stub objects. These objects implement the common functionalities required by the pattern-based management from navigation patterns and aggregators.

For the evaluation of our proposed modifications we implemented a simple aggregator that implements the management task of counting the nodes present in the network. Even though this is quite a simple management task, this fact does not interfere with the evaluation of the navigation pattern itself. This task provides us with a simple way of measuring the performance of the robust pattern, which is to compare the number of nodes

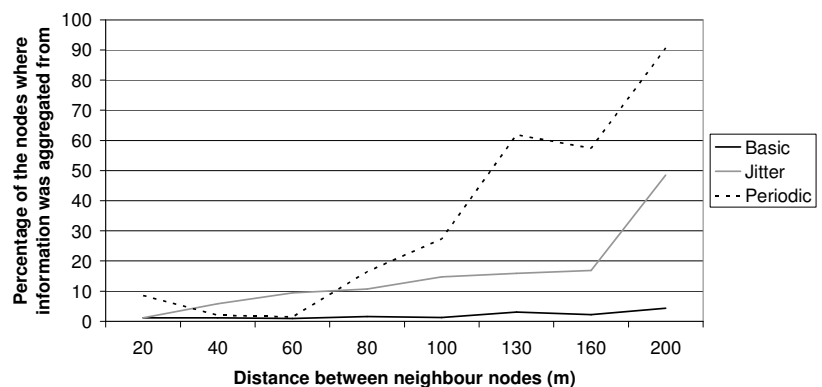


Figure 5 – Comparison of application level solutions

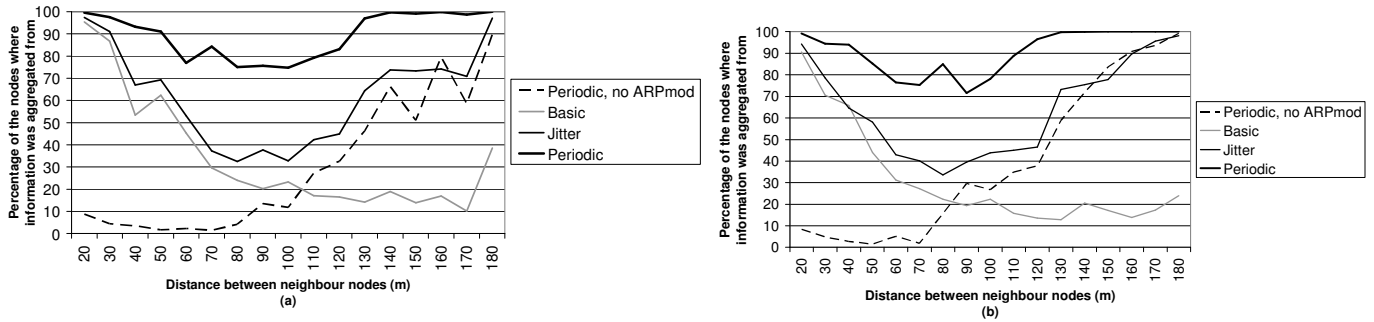


Figure 7 – Comparison of application level solutions using MAC enhancement

aggregated during the simulation with the actual number of nodes.

B. Simulation Scenarios

For MAC layer the MAC802.11 was used in the simulation, and for physical layer and channel the basic 250m effective radius transmission was used. Two scenarios were set up for the simulation of different patterns. In the first scenario (the *grid* scenario) one hundred nodes were set up in a 10x10 grid, and the distance between the nodes varied between different simulation runs, allowing the nodes to have different number of neighbours between different runs.

For the second scenario (the *random* scenario) a topology generator script was created to place nodes at random in the scenario, for more life-like simulations. The placement of nodes is done incrementally, whereas the place of the next node is chosen randomly. Possible places of the next nodes are constrained by the principle of keeping the differences introduced by using different distance values (as in the grid scenario) as much as possible. For an example see [Figure 6], where the dotted circles represent 'distance' from the five nodes already placed down, and the solid line represents possible places for the sixth node (points that are 'distance' away from exactly one node and farther from the other nodes).

Movement was simulated by randomly changing speed and destinations for each node. As this might result in scenarios where the patterns have even no theoretical chance to aggregate all nodes, the results are good for the comparison of different solutions, but the optimal results are unknown. However, the results do show how the patterns would act in real-life scenarios.

These scenarios and movement models are only the first steps towards the full evaluation of wireless protocols. Nevertheless, in our current state of research these simple scenarios are enough to evaluate the feasibility and scalability of the robust pattern-based

management in wireless networks.

V. EVALUATION OF THE PERFORMANCE OF ROBUST PATTERNS

During the evaluation we focused on the robustness factor of our enhancements. The problems introduced by the wireless scenario results in a non-complete aggregation of the overall state of the network, because of lost messages both during expansion and contraction phases. Therefore the measure we choose was the percentage of nodes a successful aggregation was carried out from. E.g. a pattern with a result of 67 means that during the aggregation phase the resulting aggregate was calculated from data received from 67% of the nodes present in the network. As the collisions are caused by overlapping effective radiuses of the nodes, we tested the patterns in different density networks.

As already mentioned we focused on the success measure only, so the regular measures (message numbers, total execution time) were not tested, as they are always below the centralized approach's results.

A. Robust Enhancements to the Echo Pattern

Our basic robust echo pattern and application level enhancements were tested in the grid scenario. Figure 5 shows how these patterns behave on the grid scenario (using 10 x 10 nodes). On the X-axis the distance between nodes of the grid is shown. The effective radius of the nodes is set to 250m so the density of the grid varies accordingly. The Y-axis represents the percentage of nodes aggregated by the pattern and the counting aggregator function. As it can be seen the periodic enhancement outperforms the other two solutions in "not-too-dense" networks, and gives acceptable performance. However on dense networks none of the above mentioned application level solutions give viable results.

As already described in section III.E, we identified that the results were mainly caused by the ARP

implementation. After evaluating the possibility of modifying the ARP used (and verifying the possibility of making the same enhancements to actual devices), we modified the implementation.

B. ARP Modification

The results of the ARP modification can be seen on (Figure 7), running the simulation on the grid scenario (a), and on the random scenario (b). For a comparison of the results with the non ARP modified values, the results of the periodic robust echo pattern without the modification are also displayed (marked as “no ARPmod”).

From these plots it is obvious that the current ARP implementation really destroys the performance of robust patterns, and it is also obvious that robust patterns work much better with the enhanced address-mapping described above. Furthermore the application level enhancements also give better performance, the periodic receiving the highest aggregate percentage, however with the highest overhead using the most messages during its work.

The performance figures of the robust patterns have noticeable elevation (higher values on the Y axis) in very dense and very sparse networks. In the former case the network is close to a one-hop network. Thus RTS/CTS messages are heard by every node, and assure interference-free transmission channels. In the sparse network scenario all nodes can see their closest neighbours only – i.e. four neighbours at most in the grid scenario. Thus a relatively low frequency of RTS/CTS messages have to share the rest of the traffic. In the case where nodes are not almost all within one hop away, but not as far as not to interfere with a reasonable high number of neighbours, much more RTS/CTS events are not respected. This causes a higher number of collisions, thus a lower success rate.

Finally, even though we did not implement any measures of counter-action for mobility, we tested the current robust echo patterns in mobile environments. If the network nodes are moving during the extraction phase, the spanning-tree built by the expansion might be broken due to the mobility, hence leaving some potential aggregate information not finding their way back to the originator node. The results of this scenario can be seen on (Figure 8).

Despite the lack of counter-actions against mobility the performance of the robust pattern is still satisfactory, but lower than those found in (Figure 7). The *periodic* pattern has the best performance in this scenario as well, except for very sparse networks, where the *jitter* pattern outperforms is. In the sparse network scenario nodes have a higher chance to fall out from each other’s transmission range as in the denser scenarios. But the nodes are not aware of mobility and they consider that the answer is not received only because of collisions and interference. Thus the *periodic* pattern sends many broadcast echoes to already departed neighbours, without the chance of ever getting a response. This raises the chance of collisions with other echoes, or with RTS/CTS messages, thus obstructs the correct behaviour of the nodes.

VI. CONCLUSION AND FUTURE WORK

The pattern-based management paradigm has been previously proven to be a viable concept for distributing network management functions. The aim of present work was to test the paradigm on wireless networks, to propose enhancements and to evaluate the performance of the enhancements by simulations.

As part of the work a platform independent pattern-based management API has been designed that can be used to program patterns and aggregators independent of the underlying simulation environment. This API was fully implemented in the ns-2 network simulator environment, and also the SIMPSON simulator has been modified to be compatible with it.

We proposed both application level enhancements and an ARP enhancement for the original navigation patterns. Even though we used the echo pattern for our simulations, the proposed enhancements can be used more generally as common wireless enhancements for the pattern-based management paradigm itself. We

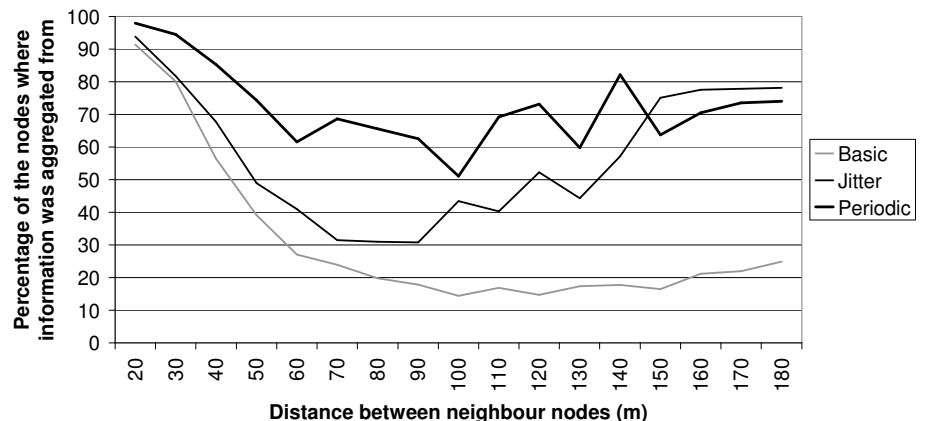


Figure 8 – Comparison of application level solutions using MAC enhancement with moving nodes

investigated these enhancements on different scenarios, and found that they enable the patterns to achieve acceptable performance on wireless networks, even in dynamic environments.

In our future work we plan to simulate the patterns on larger scale wireless networks, and also plan to propose novel enhancements that can cope with the dynamic nature of wireless networks. Such enhancements can be to find alternate routes for aggregates that cannot be sent back to their originator node and also trying to maintain a spanning-tree for multiple pattern executions.

REFERENCES

- [1] K.S. Lim and R. Stadler: "Developing pattern-based management programs", 4th IFIP/IEEE International Conference on Management of Multimedia and Network Services (MMNS'01), Chicago, Illinois, October/November 2001, pp. 345-358.
- [2] K.S. Lim and R. Stadler; "Weaver: realizing a scalable management paradigm on commodity routers", 8th IFIP/IEEE International Symposium on Integrated Network Management , 24-28 March 2003, Colorado Springs, Colorado, March 2003, pp:409 – 424
- [3] C. Adam, R. Stadler, "Patterns for Routing and Self-Stabilization", in Proc. of Network Operations & Management Symposium (NOMS 2004), Seoul, Korea, April 19.23, 2004
- [4] K.S. Lim, C. Adam and R. Stadler; "Decentralizing Network Management", submitted to IEEE electronic Transactions on Network and Service Management
- [5] The SIMPSON Simulator: <http://web.it.kth.se/~stadler/>
- [6] R. Stadler: "Decentralized and adaptable management based on active networking technology," The First International Workshop on Active Network Technologies and Applications (ANTA 2002), Tokyo, Japan, March 25-26, 2002.
- [7] The Network Simulator – ns2: <http://www.isi.edu/nsnam/ns/>
- [8] S. Ni, Y. Tseng, Y. Chen, J. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network," Wireless Networks, Vol. 8, March 2002
- [9] S. Perur, L. Chandra-Wadia, S. Iyer, "Improving the Performance of MANET Routing Protocols Using Cross-Layer Feedback," Conference on Information Technology, Bhubaneswar, December 2003.