Chameleon-MAC: Adaptive and Self-⋆ Algorithms for Media Access Control in Mobile Ad Hoc Networks⋆

Pierre Leone[1], Marina Papatriantafilou[2], Elad M. Schiller[2], and Gongxi Zhu[2]

[1] University of Geneva, (Switzerland)
pierre.leone@unige.ch
[2] Chalmers University of Technology, Sweden
{ptrianta,elad,gongxi}@chalmers.se

Abstract. In mobile ad hoc networks (MANETs) mobile nodes do not have access to a fixed network infrastructure and they set up a communication network by themselves. MANETs require implementation of a wireless Medium Access Control (MAC) layer. Existing MAC algorithms that consider no mobility, solve the problem of eventually guaranteeing every node with a share of the communications bandwidth. In the context of MANETs, we ask: Is there an efficient MAC algorithm when mobility is considered?

MANETs are subject to transient faults, from which self-stabilizing systems can recover. The self-stabilization design criteria, and related concepts of self-⋆, liberate the application designer from dealing with low-level complications, and provide an important level of abstraction. Whereas stabilization criteria are important for the development of autonomous systems, adaptation is imperative for coping with a variable environment. Adapting to a variable environment requires dealing with a wide range of practical issues, such as relocation of mobile nodes and changes to the motion patterns.

This work proposes the design and proof of concept implementation of an adapted MAC algorithm named Chameleon-MAC, which is based on a self-stabilizing algorithm by Leone et al., and uses self-⋆ methods in order to further adapt its behavior according to the mobility characteristics of the environment. Moreover, we give an extensive treatment of the aspects and parameters that can bring the algorithm into the practical realm and we demonstrate documented behavior on real network studies (MICAz 2.4 GHz motes) as well as using simulation (TOSSIM [32]), showing improved overhead and fault-recovery periods than existing algorithms.

We expect that these advantages, besides the contribution in the algorithmic front of research, can enable quicker adoption by practitioners and faster deployment.

⋆ An extended version appears in [31] as a technical report. This work is partially supported by the ICT Programme of the European Union under contract number ICT-2008-215270 (FRONT'S).

S. Dolev et al. (Eds.): SSS 2010, LNCS 6366, pp. 468–488, 2010.

1 Introduction

Mobile ad hoc networks (MANETs) are autonomous and self-organizing systems where mobile computing devices require networking applications when a fixed network infrastructure is not available or not preferred to be used. In these cases, mobile computing devices could set up a possibly short-lived network for the communication needs of the moment; in other words, an ad hoc network. MANETs are based on wireless communications that require implementation of a *Medium Access Control* (MAC) layer [40]. MAC protocols need to be robust and have high bandwidth utilization and low communication delay [38]. The analysis of radio transmissions in ad hoc networks [19] and the relocation analysis of mobile nodes [30] show that MAC algorithms that employ a scheduled access strategy, such as in TDMA, might have lower throughput than algorithms that follow a randomized strategy [such as slotted ALOHA 1, 34]. However, the scheduled approach offers greater predictability, which can facilitate fairness [21] and energy conservation. This work proposes the design and proof of concept implementation of an adapted MAC algorithm named Chameleon-MAC, which is based on a self-stabilizing algorithm [30] and uses new methods and techniques in order to further adapt its behavior according to the mobility characteristics of the environment. Through extensive treatment of the aspects and parameters of the new algorithm, we show that the algorithm in [30] can fit into a practical realm and we demonstrate documented behavior on real network studies (MICAz 2.4 GHz motes) as well as simulation (TOSSIM [32]), showing improved overhead and fault-recovery periods when compared with existing algorithms.

1.1 A Case for Adaptive Self-⋆ in MANETs

The dynamic and difficult-to-predict nature of mobile networks gives rise to many fault-tolerance issues and requires efficient solutions. MANETs, for example, are subject to transient faults due to hardware/software temporal malfunctions or short-lived violations of the assumed settings for modeling the location of the mobile nodes. Fault-tolerant systems that are *self-stabilizing* [10, 11] can recover after the occurrence of transient faults, which can cause an arbitrary corruption of the system state (so long as the program's code is still intact). The self-stabilization design criteria, and related concepts of self-⋆ [4], liberate the application designer from dealing with low-level complications, and provide an important level of abstraction. Consequently, the application design can easily focus on its task – and knowledge-driven aspects.

Whereas stabilization criteria (and the related self-⋆ concepts [4]) are important for the development of autonomous systems, adaptation is imperative for coping with a variable environment. In the context of MANETs, adapting to a variable environment requires dealing with a wide range of practical issues, such as relocation of mobile nodes, and changes to the rate or pattern by which they move [17]. Adaptation has additional aspects in the context of fault-tolerance, such as resource consumption [11, 16]. For example, high rates of churn and concurrent node relocations can significantly increase the overhead.

MANETs require MAC protocols that can provide robustness, high throughput, and low latency [38]. Broadly speaking, existing implementations of MAC protocols are based on carrier sense multiple access with collision avoidance (CSMA/CA) and cannot predictably guarantee these requirements [6, 7]. The design of the Chameleon-MAC algorithm takes a fresh look at time division multiple access (TDMA) algorithms. The Chameleon-MAC algorithm automatically adjusts the choice of timeslots according to the characteristics of a variable environment (just as the Chamaeleonidae lizards adapt their chromatophore cells according to their surrounding colors). The adaptive design of the Chameleon-MAC algorithm offers robustness, a greater degree of schedule predictability than existing CSMA/CA algorithms, and better resource adaptiveness than existing TDMA algorithms. Such advantages allow quicker technological developments and faster industrial deployment of MANETs.

1.2 Relocation Model

While MAC algorithms have been extensively studied in the context of MANETs, both numerically and empirically, analytical study has been neglected thus far. Until now, it has been speculated that existing MAC algorithms (that do not consider mobility of node) would perform well in MANETs. Alternatively, algebraic equations are used for modeling the kinetics of mobile nodes [5]. Kinetic models are difficult to analyze; it is hard to consider arbitrary behavior of mobile nodes and transient faults.

We consider an abstraction of model, named *relocations analysis*, which simplifies the proofs of impossibility results, lower bounds, and algorithms [30]. Models for relocation analysis focus on the location of mobile nodes rather than emphasizing the "movements of the mobile users" [as in 8]. The location of mobile nodes changes in discrete steps that are determined by a small set of parameters, rather than in continuous motion and by complex rules [as in kinetics model of 5]. The analytical results for this abstract model hold for a large set of concrete mobility models that can implement it. For example, a relocation analysis model can be used to estimate the throughput of MAC algorithms in realistic highway scenarios (cf. Section 5.4). Thus the studied model can improve the understanding of MANETs by facilitating an analytical study of its algorithms in system settings that can represent realistic scenarios.

1.3 Our Contribution

Algorithmic properties and systematic studies. We propose the Chameleon-MAC algorithm; an adaptive, self-$\star$ MAC algorithm that takes the scheduled approach and adapts to the variable environment of MANETs. We study the algorithm in an abstract and universal relocation model, and implement it in a real network of wireless MICAz 2.4 GHz motes. The relocation model allows us to compare the Chameleon-MAC algorithm to self-stabilizing MAC algorithms, such as the ones by Leone et al. [30], Herman-Tixeuil [21], slotted ALOHA [1] and p-persistent CSMA/CA (with and without back-off) [40]. This study and proof of concept includes an extensive treatment of the aspects and parameters that can bring the algorithm into the practical realm and demonstrate documented behavior on real network studies (MICAz 2.4 GHz motes) as well as using simulation (TOSSIM [32]). The study also shows that the Chameleon-MAC algorithm maintains higher throughput than [1, 21, 30, 40], while it also reveals other properties of interest.

Abstract-study-model properties and possibilities. Thus far, the designers of fault-tolerant algorithms for MANETs have considered a plethora of mobility models [8] for modeling the location of mobile nodes. Therefore, it is difficult to compare different MAC algorithms for MANETs. Our study model is an abstract relocation model used to analytically compare MAC algorithms [as in 30] or to conduct experimental study and evaluation as we do in this work. In [30], the properties of the model are briefly explored. The present work demonstrates further that the relocation model is sufficiently abstract to describe a variety of situations, and detailed enough to allow comparative studies of different algorithms.

Concrete-study-model scenarios and conclusions for practice. Based on the abstract relocation model, we show how to describe concrete mobility models for Vehicular Ad-Hoc Networks (VANETs) that, due to the nodes' mobility, have regular and transient radio interferences. We also systematically study the role of the local estimation of global parameters, because mobile nodes do not have direct knowledge about the model's global parameters and their impact on the throughput. Furthermore, we use the aforementioned models to study the algorithms with varying mobility parameters and demonstrate that the Chameleon-MAC algorithm adapts in variable environments that have radio interferences. Namely, the study shows that the Chameleon-MAC algorithm quickly recovers from radio interferences that occur due to the nodes' mobility in VANETs. Moreover, even in scenarios with no mobility, the Chameleon-MAC algorithm's throughput is higher than the one of existing implementations [1, 21, 30, 40]. We present measurements, on a real network of MICAz 2.4 GHz motes, that validate this observation, which we first obtain through TOSSIM [32] implementations.

2 Preliminaries

The system consists of a set of communicating entities, which we call *(mobile) nodes* (or *(mobile) motes*). Denote the set of nodes by P (processors) and every node by $p_i \in P$.

Synchronization. We assume that the MAC protocol is invoked periodically by synchronized *common pulses*. The term *(broadcasting) timeslot* refers to the period between two consecutive common pulses, t_x and t_{x+1}, such that $t_{x+1} = (t_x \bmod T) + 1$, where T is a predefined constant named the *frame size*, i.e., the number of timeslots in a TDMA frame (or broadcasting round).

Communication model. We consider a standard radio interference unit (such as `CC2420` [36]) that allows sensing the carrier and reading the energy level of the communication channel. Sometimes, we simplify the description of our algorithms and relocation models by considering concepts from graph theory. Nevertheless, the simulations consider a standard physical layer model [27].

At any instance of time, the ability of any pair of nodes to directly communicate is defined by the set, $N_i \subseteq P$, of *neighbors* that node $p_i \in P$ can communicate with directly. Wireless transmissions are subject to collisions and we consider the potential of nodes to interfere with each others' communications. We say that nodes $A \subseteq P$ *broadcast simultaneously* if the nodes in A broadcast within the same timeslot. We denote by $\mathcal{N}_i = \{p_k \in N_j : p_j \in N_i \cup \{p_i\}\} \setminus \{p_i\}$ the set of nodes that may interfere with p_i's communications when any nonempty subset of them, $A \subseteq \mathcal{N}_i : A \neq \emptyset$, transmit simultaneously with p_i. We call $\mathcal{N}_i$ the *interference neighborhood* of node $p_i \in P$, and $|\mathcal{N}_i|$ the *interference degree* of node $p_i \in P$.

3 Models for Relocation Analysis

We enhance the abstract relocation model of [30] using geometric properties that can limit the node velocity, unlike the earlier model [30]. In order to exemplify concrete models that can implement the proposed abstract one, we consider two mobility models that are inspired by vehicular ad hoc networks (VANETs).

3.1 Abstract Model Definitions

In [30], the authors use relocation steps, r_t, that relocate a random subset of nodes, P_{r_t}, and require that the number of nodes that relocate at time t is at most $\alpha|P|$, where $\alpha \in [0, 1]$ (*relocation rate*) and time is assumed to be discrete. The relocation steps of [30] are random permutations of the locations of the nodes in P_{r_t}. Namely, within one relocation step, a mobile node can relocate to any location. Thus, the model in [30] does not limit the node velocity.

This work looks into different scenarios in which each mobile node randomly moves in the Euclidian plane $[0, 1]^2$. Initially, n vertices are placed in $[0, 1]^2$, independently and uniformly at random. The relocation steps in which a bounded number of nodes, $p_i \in P_r$, change their location, $p_i(t) = \langle x_i(t), y_i(t)\rangle$, to a random one, $p_i(t+1)$, that is at a distance of at most $\beta \geq \text{distance}(p_i(t), p_i(t+1))$, where t is a discrete time instant, $\beta \in [0, 1]$ is a known constant named *(maximal) relocation distance* and $\text{distance}(p_i, p_j) = \sqrt{(y_j - y_i)^2 + (x_j - x_i)^2}$ is the geometric distance. We note that β limits the node speed in concrete mobile models that implement the proposed abstract model.

3.2 Concrete Model Definitions

Relocation analysis considers an abstract model that can represent several concrete mobility models. We consider two concrete mobility models that are inspired by vehicular ad hoc networks (VANETs). The models depict parallel and unison motion of vehicles that move at a constant speed and in opposite lanes. The first model assumes that the mobile nodes are placed on a grid and thus the radio interferences follow regular patterns. The second model considers vehicle clusters that pass by each other and thus their radio interferences are transient.

Regular radio interference. This model is inspired by traffic scenarios in which the vehicles are moving in parallel columns. We consider m rows, $r_0, r_1, \ldots, r_{m-1}$, of m nodes each ($m = 20$ in our experiments). Namely, $r_j = p_{j,0}, p_{j,1}, \ldots p_{j,m-1}$ is the j-th row. In this concrete model, the axes of the Euclidian plane $[0,1]^2$ are associated with scales that consider $\frac{1}{200}$ as its *distance units*. We assume that, at any time, the location, (x_i, y_i), of a mobile node, p_i, is aligned to the axes' scale. Namely, $200x_i$ and $200y_i$ are integers. Moreover, at any time, the distance between $p_{j,k}$ and $p_{j,k+1}$ (where $k, j \in [0, m-2]$) and the distance between $p_{j,k}$ and $p_{j+1,k}$ (where $j \in [0, m-2]$) is ψ distance units ($\psi = 10$ in our experiments). In the initial configuration, the nodes are placed on a $m \times m$ matrix (of parallel and symmetrical lines). At each relocation step, the nodes in the even rows move a constant distance, $speed > 0$, to the right. Moreover, when the nodes move too far to the right, they reappear on the left. Namely, the rightmost node in the rows, $p_{j,m-1}$, becomes the leftmost node in the row when the location of the $p_{j,m-2}$ is not to the right of the vertical line, $\ell_{up,down}$, that can be stretched between the location of that nodes $p_{up,m-1}$ and $p_{down,m-1}$, where $up = j - 1 \bmod m$ and $down = j + 1 \bmod m$. In Section 5, we show that the radio interferences of this model follow a regular pattern.

Transient radio interference. This model is inspired by two vehicle clusters that pass each other while moving in opposite lanes. The clusters are formed in a process that assures a minimal distance of ψ units between any two neighboring nodes ($\psi = 10$ in our experiments). Namely, we start by placing the nodes on a $m \times m$ matrix (of parallel and symmetrical lines), and let the nodes move toward their neighbors in a greedy manner that minimizes the distances among neighbors ($m = 20$ in our experiments). Once the clusters are formed, the cluster on the right moves towards the one on the left in a synchronized manner. At each relocation step, they reposition themselves to a location that is a constant number of units distance, $speed > 0$, from their current location. In Section 5, we show that the radio interferences of this model are transient.

4 The Chameleon-MAC Algorithm

An adaptive and self-⋆ MAC algorithm for MANETs is presented. The algorithm is based on a non-adaptive, yet self-stabilizing, MAC algorithm for MANETs [30]. Leone et al. [30] explain how mobile nodes are able to learn some information about the success of the neighbors' broadcasts and base their algorithm on vertex-coloring; nodes avoid broadcasting in the timeslots in which their neighbors successfully broadcast. Namely, the algorithm assigns each node a color (timeslot) that is unique to its interference neighborhood.

The algorithm by Leone et al. [30] is self-stabilizing; however it is not adaptive. For example, the algorithm allows each node to broadcast at most once in every broadcasting round and assumes that the number of timeslots in the broadcasting rounds, T, is at least as large as the maximal size of the interference neighborhoods. Before presenting the Chameleon-MAC algorithm, we explain some details from [30] that are needed for understanding the new Chameleon-MAC algorithm.

4.1 Self-stabilizing MAC Algorithm for MANETs

Keeping track of broadcast history is complicated in MANETs, because of node relocations and transmission collisions. The algorithm by Leone et al. [30] presents a randomized solution that respects the recent history of the neighbors' broadcasts based on information that can be inaccurate. However, when the relocation rate or maximal relocation distance are not too great, the timeslots can be effectively allocated by the Leone et al. algorithm [30].

This is achieved using a randomized construction that lets every node inform its interference neighborhood on its broadcasting timeslot and allows the neighbors to record this timeslot as an occupied/unused one. The construction is based on a randomized competition among neighboring nodes that attempt to broadcast within the same timeslot. When there is a single competing node, that node is guaranteed to win. Namely, the node succeeds in informing its interference neighborhood on its broadcasting timeslot and letting the interference neighborhood mark its broadcasting timeslot as an occupied one. In the case where there are $x > 1$ competing nodes, there might be more than one "winner" (hence causing collisions). However, the expected number of winners will decrease after each subsequent round, because of the randomized construction. Why this procedure is guaranteed to converge is shown in [30]. For self-containment, we include the pseudo-code description of [30] in Fig. 1.

4.2 The Chameleon-MAC Algorithm

The Chameleon-MAC algorithm adapts to a variable environment in which relocation parameters, as well as the size of the interference neighborhoods, can change. Moreover, the algorithm adapts its behavior according to the state of the allocated resources, i.e., nodes adjust their timeslot allocation strategy according to the distribution of assigned timeslots. In order to do that, the algorithm employs new methods and techniques that achieve self-$\star$ properties [4].

Definition 1 is required for the presentation of the Chameleon-MAC algorithm.

Definition 1 (Timeslot properties). *Consider the interference neighborhood, $\mathcal{N}_i : p_i \in P$, and its timeslot assignment in $\mathcal{N}_i$, which we denote by $[\nu_{i,s}]$, where $\nu_{i,s} = \{p_j \in \mathcal{N}_i : s_j = s\}$, where s_j is the timeslot used by processor $p_j \in \mathcal{N}_i$. We say that timeslot $s \in [0, T-1]$ is:*

- *empty; if no neighbor in $\mathcal{N}_i$ transmits in timeslot s, i.e., $|\nu_{i,s}| = 0$,*

```
Variables and external functions
  MaxRnd = number of rounds in the competition
  s:[0, T-1] ∪ {⊥} = next timeslot to broadcast or null (⊥)
  competing: boolean = competing for the channel
  unique: boolean = indicates a unique broadcasting timeslot
  unused[0,T-1]: boolean = marking unused timeslots
  MAC_fetch()/MAC_deliver(): layer interface
  broadcast/receive/carrier_sense/collision(): media primitives

Upon timeslot(t)
  if t = 0 then (* On the first timeslot perform a test *)
    (* Was the previous broadcast unsuccessful? *)
    if ¬ unique ∨ s = ⊥ then
      (* Choose again the broadcasting timeslot *)
      s ← get_random_unused()
    unique ← false (* reset the state of unique *)
  unused[t] ← true (* remove stale information *)
  (* Get a new message and sent it if everything is ok *)
  if s ≠ ⊥ ∧ t = s then send(MAC_fetch())

Upon receive(m) do MAC_deliver(m)

Function get_random_unused() (* selects unused timeslots *)
  return select_random({ k ∈ [0,T-1] |unused[k] = true })

Function send(m)
  (competing, unique, k) ← (true, true, 1) (* start competing *)
  while k ≤ MaxRnd ∧ competing = true (* stop competing? *)
    with probability 2^(−MaxRnd+k) do
      broadcast(m) (* try acquiring the channel *)
      competing ← false (* quit the competition *)
    with probability 1 − 2^(−MaxRnd+k) do
      wait until the end of the competition time unit
    k ← k + 1

(* Stop competing when a neighboring node starts transmitting *)
Upon carrier_sense(t) (* a neighbor is using timeslot t *)
  if competing = true then unique ← false
  (competing, unused[t]) ← (false, false)

(* Collisions indicate unused timeslots *)
Upon reception_error(t) do unused[t] ← true
```

Fig. 1. The self-stabilizing MAC algorithm for MANETs by Leone et al. [30], code of processor p_i

- congested; *if more than one neighbor transmits in timeslot s, i.e., $|\nu_{i,s}| > 1$,*
- well-used or unique; *if exactly one neighbor transmits, i.e., $|\nu_{i,s}| = 1$, and*
- unused; *if no neighbor or more than one neighbor transmits, i.e., $|\nu_{i,s}| \neq 1$ and thus timeslot s is not properly assigned.*

Variable relocation rate and distance. MAC algorithms that follow the scheduled approach spend some of the communication bandwidth on allocating timeslots. We consider such algorithms and relocation models with constant parameters, in order to explain the existence of a trade-off between the throughput and the convergence time. Then, we explain how to balance this trade-off by adapting the algorithm's behavior according to the relocation parameters.

Obviously, there is a trade-off between the throughput, τ, and the communication overhead, h, because $\tau + h$ is bounded by the communication bandwidth. The throughput of MAC algorithms that follow the scheduled approach is guaranteed to converge within a bounded number of broadcasting rounds, ϱ. The rate by which they converge depends not only on the model's parameters, but also on the communication overhead, h. For example, the more bandwidth that the Leone et al. algorithm [30] is spending on competing for timeslots, the greater recovery each broadcasting round provides, and consequently, the shorter the convergence period is. This is a trade-off between the throughput, τ, and the convergence time, ϱ, which is settled by the communication overhead, h. Complementing the Leone et al. algorithm [30], the Chameleon-MAC algorithm balances this trade-off.

The Chameleon-MAC algorithm lets nodes adjust the communication overhead that they spend on competing for their timeslot according to an estimated number of unique timeslots (cf. Definition 1). The intuition behind the balancing technique is that the energy level becomes low as the nodes successfully allocate unique timeslots, because there are fewer collisions. Thus, after a convergence period, the nodes can reduce the communication overhead, h, by using less of the communication bandwidth on competing for their timeslots. Namely, they adjust the value of $MaxRnd$ of Figure 1. The eventual value of h depends on the parameters of the relocation model, because the nodes should not stop dealing with relocations. Namely, in order to cope with relocations, the nodes should always deal with events in which their broadcasting timeslots stop being unique due to the relocation of mobile nodes. Such relocations cause message collisions and higher energy level in the radio channels. The Chameleon-MAC algorithm copes with such changes by letting each node gradually adjust the amount of communication bandwidth it spends competing for timeslots.

Thus, when the relocation parameters are constants, the communication overhead, h, is set to a value that balances the trade-off between the throughput, τ, and convergence period, ϱ.

Variable size of interference neighborhoods. The Chameleon-MAC algorithm follows the scheduled approach because of its greater predictability compared to the random access approach. Namely, the nodes are allocated unique timeslots from the TDMA frame (cf. Definition 1). The Chameleon-MAC algorithm considers frames of a fixed size and adapts the timeslot allocation according to the size of interference neighborhoods complementing the Leone et al. algorithm [30].

The presentation of the techniques in use is simplified by considering the two possible cases: (1) the size of p_i's interference neighborhood, D_i, is not greater than the TDMA frame size, T, i.e., $D_i \leq T$ and (2) $D_i > T$ (where $p_i \in P$ is a node in the system). Notice that the procedures are concurrently executed and do not require explicit knowledge about D_i.

∘ $D_i \leq T$. In this case, the Chameleon-MAC algorithm allocates to each node at least one timeslot. Node p_i publishes (in its data-packets) the number of timeslots, $T_i \in [0, T]$, that it uses in order to facilitate fairness requirements (i.e., attach this information to message m send in line 30 of Figure 1). The fairness criterion is that $|T_i - M_i| \leq 1$, where M_i is the median of timeslots allocated to the neighbors of p_i. The nodes apply this fairness criterion when deciding on the number of timeslots to use.

∘ $D_i > T$. The algorithm employs two methods for dealing with cases in which there are more nodes in the neighborhood than TDMA timeslots; one for facilitating *fairness* and another for *contention control*.

Fairness can be facilitated by letting the nodes transmit at most ℓ consecutive data-packets before allowing other nodes to compete for their (already allocated) timeslots, where ℓ is a predefined constant.

The contention control technique is inspired by the p-persistent CSMA algorithm [40]. It assures that, at any time, an expected fraction of $p \in (0, 1)$ mobile nodes that do not have unique timeslots would compete for an unused one and $(1-p)$ of them would defer until a later broadcasting round (cf. Definition 1). i.e., each node that needs to change its broadcasting timeslot decides, with probability

p, to attempt to use the new timeslot in the broadcasting round that immediately follows [as in the Leone et al. algorithm 30], and with probability $(1-p)$ it skips that broadcasting round. Namely, the function get_random_unused() in lines 23 to 24 of Figure 1, returns $\perp$ with probability $(1-p)$.

This process can be repeated for several timeslots until the node decides, with probability p, to attempt to use one of the unused timeslots.

The combination of the methods for facilitating fairness and contention control aims at allowing nodes to eventually acquire a unique timeslot. The successful reservation of unique timeslots for the duration of ℓ broadcasting rounds depends, of course, on the nodes' relocation.

Adaptive timeslot allocation strategy. The Chameleon-MAC algorithm employs two methods for adjusting the timeslot allocation according to the allocation's distribution.

◦ *Luby-algorithm-inspired method.* Luby [33] presents a round-based vertex-coloring algorithm. In each round, every uncolored vertex selects, uniformly at random, an unused color (cf. Definition 1). In Luby's settings, each vertex can accurately tell whether its color is well-used. This is difficult to achieve deterministically in wireless radio communications. However, the nodes can discover with probability $\Pr_{test} \in [\frac{1}{4}, \frac{1}{2}]$ whether their broadcasting timeslots are well-used [see 30, Section 4]. Therefore, a *non-uniform* selection of unused timeslots is the result of letting the nodes whose broadcasting timeslot is congested select, uniformly at random, an unused timeslot (as in line 24 of Figure 1). This is because a ratio of $1 - \Pr_{test}$ nodes cannot detect that their broadcasting timeslot is congested, and hence do not change their broadcasting timeslot.

The Luby-algorithm-inspired method prefers empty timeslots to congested ones when selecting a new broadcasting timeslot from the set of unused ones.

◦ *Žerovnik-algorithm-inspired method.* Žerovnik [41] accelerates the stabilization of vertex-coloring algorithms by favoring colors that are less represented in the neighborhood. Inspired by Žerovnik's technique, the Chameleon-MAC algorithm employs a method that aims at favoring timeslots that are less frequently used in the neighborhood when nodes are required to select a new timeslot. The heuristic favors the selection of congested timeslots with low energy level of the radio channel over congested timeslots with high level (cf. Definition 1).

In more detail, when node $p_i \in P$ considers a new timeslot, s_i, and s_i happens to be congested, then with probability $\Pr(s_i)$, p_i indeed uses timeslot s for broadcasting and with probability of $1 - \Pr(s_i)$ it does not change its broadcasting timeslot, where $\Pr(s_i) = 1 - \mathcal{O}(\exp(timeslot\_energy\_level[s_i]))$, the array $timeslot\_energy\_level_i[]$ stores the channel energy level that node p_i recorded during the latest broadcasting round. We note that the value of $timeslot\_energy\_level_i[t]$ is negative in our settings, because of the low energy used for transmission.

5 Experimental Evaluations

Throughput is a basic measure of communication efficiency. It is defined as the average fraction of time that the channel is employed for useful data propagation [40]. We study the relationship between the eventual throughput, τ, of the studied algorithms and the model parameters. The study considers: (1) simulated throughput, $\tau_{\text{simulated}}$, obtained by simulation (TOSSIM [32]), (2) estimated throughput, $\tau_{\text{estimated}}$, which is a MATLAB interpolation of the simulation results and (3) measured throughput, τ_{measured}, obtained by executing the Chameleon-MAC algorithm in a real network of wireless MICAz 2.4 GHz motes. The results of the experiment suggest that the throughput, τ, is a function that depends on the model parameters.

5.1 System Settings

The simulations were conducted in TOSSIM/TinyOS `2.1v` [32] and considered 400 motes. We have considered the default values for TOSSIM's radio model, which is based on the `CC2420` radio, used in the MICAz 2.4 GHz and telos [36] motes. In our simulations, we use timeslots of 2.5 ms.

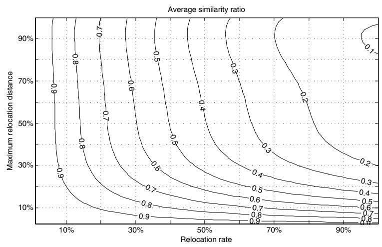

Fig. 2. Average similarity ratio is depicted as the function $ASR(\alpha, \beta)$ of the relocation rate, α (x-axis), and the maximal relocation distance β (y-axis). 5th polynomial interpolation was used.

Radio interference model. TOSSIM/TinyOS `2.1v` simulates the ambient noise and RF interference a mote receives, both from other radio devices as well as outside sources. In particular, it uses an SNR-based (Signal-to-Noise Ratio) packet error model with SINR-based (Signal to Interference-plus-Noise Ratio) interference [32]. Thus, common issues, such as the hidden terminal problem and the exposed terminal problem, are considered.

Simulation details. The simulation experiments considered 400 mobile nodes. Before the system execution, each node had selected its broadcasting timeslot uniformly at random from the set $[0, T-1]$ of timeslots. We then let the simulation run for a sufficiently long period (100 broadcasting rounds) during which the throughput, τ, eventually converges.

Empirical test-bed. This work demonstrates a proof of concept of the Chameleon-MAC algorithm using a real network of 35 MICAz 2.4 GHz motes, which are composed of the `ATmega128L` microcontroller and the `CC2420` radio chip. The motes were placed on a 5×7 matrix, the distance between two neighboring motes was 22 cm and the radio admission level was -90dBm [9, Section 6.3]. Throughout this particular experiment, the motes were not moved.

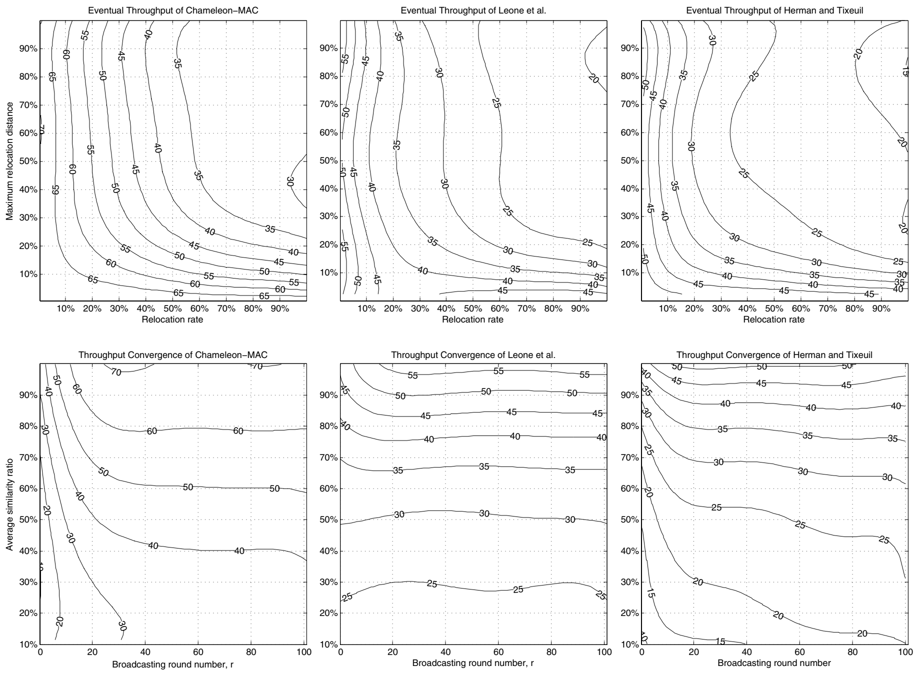

Fig. 3. Throughput of the Chameleon-MAC algorithm, τ, is compared to those of Leone et al. [30] and Herman-Tixeuil [21]. On the top row, the throughput, $\tau(\alpha, \beta)$, is depicted as a function of the relocation rate, α (x-axis), and the maximal relocation distance β (y-axis). On the bottom row, the throughput, $\tau(r, asr)$, is depicted as a function of the broadcasting round, r (x-axis), and ASR (y-axis). E.g., suppose that the ASR is 50%, then the throughput of the Chameleon-MAC algorithm will be about 20%, 30% and 40% within about 5, 10 and 20 broadcasting rounds, whereas the throughput of the Herman-Tixeuil [21] will be about 20% and 25% within about 10 and 60 broadcasting rounds. 5th polynomial interpolation was used.

The Chameleon-MAC algorithm assumes that each node has access to an accurate clock that can facilitate a common pulse. In our test-bed, the common pulse was emulated by starting every timeslot with a beacon message sent by the central mote, $p_{3,5}$.

5.2 Presentation

The relocation rate and distance are global parameters that define the relocation model. It is not clear how they can be estimated locally by the mobile nodes. Therefore, we consider the average similarity ratio (ASR) parameter, which can be calculated locally by the nodes [31].

The definition of the *average similarity ratio* (ASR) considers the unit disk graph (UDG); given the disk radius, $\chi \in [0, 1]$, we define $G_t = (P, E_t)$ as the UDG that the mobile nodes induce in time t, where $E_t = \{(p_i, p_j)|\text{distance}(p_i, p_j) \leq \chi\}$ is the set of edges at time t and $\text{distance}(p_i, p_j) = \sqrt{(y_j - y_i)^2 + (x_j - x_i)^2}$ is the geometric distance. The ASR is defined by a (non-aggregated) similarity ratio, $ASR_i = \frac{|N_i(G_t) \cap N_i(G_{t+1})|}{|N_i(G_t)|}$, that considers the neighbors that a mobile node maintains when relocating to a new neighborhood, where $N_i(G)$ is the set of p_i's neighbors in graph G. The nodes that are placed near the plane's borders have a lower (non-aggregated) similarity ratio upon relocation. Hence, $ASR = \text{average}(\{ASR_i | p_i \in P(t)\})$, considers the set of nodes, $P(t) = \{p_i \in P | \langle x_i(t), y_i(t) \rangle \in [\frac{1}{5}, \frac{4}{5}]^2\}$.

$ASR(\alpha, \beta)$ depicts ASR as a function of the relocation rate and distance; see Fig. 2. *Contour charts* present two parameter functions, e.g., $ASR(\alpha, \beta)$. They are often used in geographic maps to join points of the same height above sea level. Contour lines in Fig. 2 connect values of $ASR(\alpha, \beta)$ that are the same (see the text tags along the line).

The term *percentage of potential throughput* (PPT) is used in the presentation of the throughput: $\tau_{\mathsf{simulated}}$, $\tau_{\mathsf{estimated}}$ or τ_{measured}. It considers a TDMA frame in which only data-packets are sent and they are all received correctly. We define $bit_{potential}$ as the sum of bits in all the payloads of such a frame. Given a broadcasting round, r, and a node, $p_i \in P$, we define $bit_{actual}(r, i)$ as the number of bits in the payloads that were sent correctly by p_i or received correctly by p_i in r. Given a broadcasting round, r, we define PPT as $\frac{1}{|P(t_r)|} \sum_{p_i \in P(t_r)} \frac{bit_{actual}(r,i)}{bit_{potential}}$, where t_r is the time in which the broadcasting round begins and $P(t) = \{p_i \in P | \langle x_i(t), y_i(t) \rangle \in [\frac{1}{5}, \frac{4}{5}]^2\}$. We note that in our settings, the *maximum potential throughput* MPT is 76%, because of the time required for multiplexing and the transmission of the packet header/footer.

5.3 Throughput

The results of the experiments allow us to compare the Chameleon-MAC algorithm to Leone et al. [30], Herman-Tixeuil [21], slotted ALOHA [1] and p-persistent CSMA/CA (with and without back-off) [40] (Fig. 3, Fig. 4 and Fig. 5).

The simulated throughput values are compared as a function, $\tau(\alpha, \beta)$, of relocation rate and distance in Fig. 3-top. The charts show that the Chameleon-MAC algorithm has greater throughput than Leone et al. [30] and Herman-Tixeuil [21].

The simulated throughput values are depicted as a function, $\tau(r, asr)$, of the broadcasting rounds and the ASR

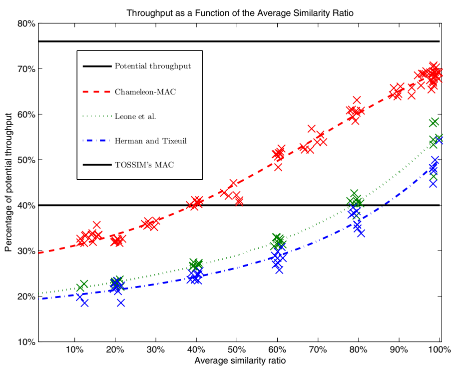

Fig. 4. Throughput of the Chameleon-MAC algorithm, τ, is compared to those of Leone et al. [30], Herman-Tixeuil [21], and CSMA/CA with back-off

in Fig. 3-bottom. The charts show that the CHAMELEON-MAC algorithm converges within 30 broadcasting rounds, whereas the Herman-Tixeuil [21] converge period may take more than 100 rounds.

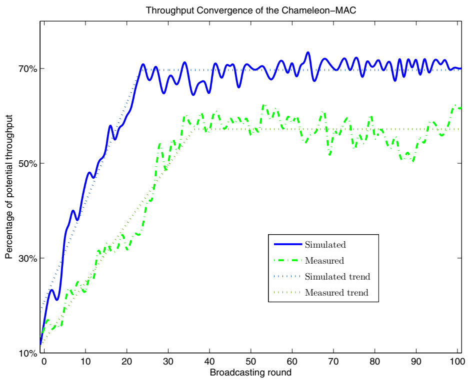

Fig. 5. Simulated and measured throughput (upper solid line and lower dash-dot line, respectively) together with their trends (dotted lines). The x-axes consider the broadcasting round number, r. The y-axes consider the percentage of potential throughput (PPT). The simulated throughput's trend is $\tau_{simulated\_trend}(r) = 0.02r + 0.19$ when $r < 25$, and 0.57 when $r \geq 25$. The measured throughput's trend is $\tau_{measured\_trend}(r) = 0.01r + 0.11$ when $r < 37$ and 0.70 when $r \geq 37$.

The simulated throughput values are depicted as a function, $\tau(asr)$, of the ASR in Fig. 4. The chart shows that around $asr = 40\%$, there is a critical threshold, above which the throughput of the CHAMELEON-MAC algorithm is higher than TOSSIM's native MAC (CSMA/CA with back-off). We let TOSSIM's native MAC (CSMA/CA with back-off) represent the MAC algorithms that follow the randomized approach, such as slotted ALOHA [1] and p-persistent CSMA/CA without back-off [40], because TOSSIM's native MAC has greater throughput than slotted ALOHA [1] and p-persistent CSMA/CA without back-off [40]. The results show that the maximal eventual throughput of the CHAMELEON-MAC algorithm is $\tau = 70.4\%$ when the relocation model considers no mobility, which is 92.6% of the maximum potential throughput (MPT). In general, the interpolated function $\tau_{estimated}(asr) = 0.41(asr)^{1.353} + 0.29$ can be used for estimating the eventual throughput of the CHAMELEON-MAC algorithm when the ASR is constant.

The simulated and measured throughput values are compared in Fig. 5. We note that the eventual measured throughput is about 82% of the simulated one with standard deviations of 2.93% and 1.33%, respectively, which are similar to the throughput deviation of TOSSIM's native MAC (CSMA/CA with back-off). Moreover, the measured convergence is 59%, slower than the simulated one. We attribute these differences in throughput, stability and convergence to the lack of detail in TOSSIM's radio interference model [32].

5.4 Validation of the Abstract Relocation Analysis in Concrete Mobility Models

We consider the two concrete mobility models for vehicular ad hoc networks (VANETs) that were presented in Section 2. They validate the abstract relocation model by showing that, in scenarios that exclude unrealistic situations, the

Chameleon-MAC algorithm can maintain throughput that is greater than the studied algorithms [1, 21, 30, 40] in the presence of regular and transient radio interference that occurs due to the nodes' mobility. Moreover, we show that the throughput of the Chameleon-MAC algorithm in VANETs is correlated to the function, $\tau_{estimated}(asr)$, that interpolates eventual throughput (cf. Section 5.3).

Regular radio interference. This model is inspired by traffic scenarios in which vehicles are moving in parallel columns. The experimental results are presented in Fig. 6. The experiment considers $speed \leq 20$ distance units per broadcasting round, because we exclude unrealistic situations.

The figure shows that within fewer than 30 broadcasting rounds, the throughput of the Chameleon-MAC algorithm is greater than TOSSIM's native MAC (CSMA/CA with back-off) and within about 50 broadcasting rounds, the throughput of the Chameleon-MAC algorithm converges to a value that is less than 5% from the eventual value and 50% greater than TOSSIM's native MAC.

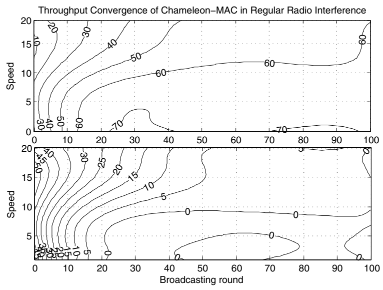

Fig. 6. Simulated and estimated throughput in VANETs with regular interference. The top chart presents the throughput, $\tau_{\mathsf{simulated}}(r, speed)$, as a function of the broadcasting rounds, r (x-axis), and the *speed* (y-axis) in unit distance per broadcasting round. The bottom chart presents the residual, $\tau_{\mathsf{simulated}}(r, speed) - \tau_{\mathsf{estimated}}(ASR(r))$, as a function of the broadcasting rounds, r (x-axis), and the *speed* (y-axis) in unit distance per broadcasting round.

Transient radio interference. This model is inspired by two vehicle clusters that pass each other while moving in opposite lanes. Fig. 7 considers such transient radio interference. The results show that, in scenarios that exclude unrealistic situations, the Chameleon-MAC algorithm maintains greater throughput than the studied MAC algorithms [1, 21, 30, 40] in the presence of transient radio interference and it can quickly recover from such faults.

Fig. 7 considers the simulated throughput, $\tau_{simulated}(r)$, estimated throughput, $\tau_{estimated}(r) = \tau_{estimated}(ASR(r))$ (cf. the interpolated eventual throughput for a constant ASR in Section 5.3), average similarity ratio, $ASR(r)$, and the residual, $residual(r) = \tau_{simulated}(r) - \tau_{estimated}(r) + 1$ (the addition of the constant 1 allows the reader to visually compare between $residual(r)$ and $ASR(r)$). The figure depicts an interference period during the broadcasting rounds $r \in [20, 40]$.

6 Discussion

Though some fundamental ad hoc networking problems remain unsolved or need optimized solutions, it is believed that ad hoc networks are not very far from

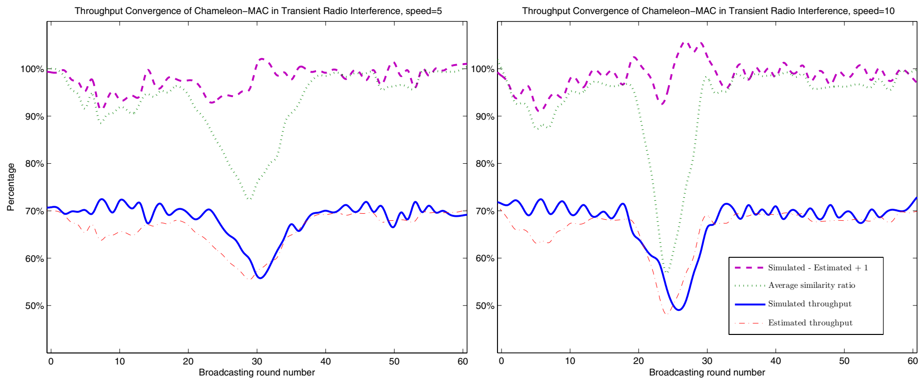

Fig. 7. Simulated throughput, $\tau_{simulated}(r)$, estimated throughput, $\tau_{estimated}(r)$, average similarity ratio, $ASR(r)$, and the residual, $residual(r) = \tau_{simulated}(r) - \tau_{estimated}(r) + 1$, are depicted by the solid line, dash-dot line, dotted line, and dashed line, respectively in a VANETs with transient interference. The left and right charts consider $speed = 5$ and $speed = 10$, respectively, of distance units per broadcasting round. The x-axes consider the broadcasting round number, r. The y-axes consider the percentage of the aforementioned functions.

being deployed on a large-scale commercial basis. For example, the TerraNet system allows voice and data communications via a peer-to-peer mobile mesh network comprised of modified handsets that were originally designed for cellular networks [39]. Vehicular ad hoc networks (VANETs) are a type of mobile network used for communication among mobile vehicles [22]. Nowadays, major automakers are developing future applications for vehicular communications on roads and highways. In sensor MANETs, the motes have joint monitoring tasks and the nodes' mobility can help achieve such tasks. Moreover, mobility can facilitate logistical services, say, by providing network connectivity between the MANETs' nodes and other communication endpoints, such as command-and-control units. Such services and applications can be constructed using the Virtual Node (VN) abstraction for MANETs [14, 15] without the use of fixed or stationary infrastructure.

6.1 Related Work

The `IEEE 802.11` standard is widely used for wireless communications. Nevertheless, the research field of MAC protocols is very active and has hundreds of publications per year. In fact, the `IEEE 802.11` amendment, `IEEE 802.11p`, for wireless access in vehicular environments (WAVE), is scheduled for November 2010. It was shown that the standard's existing implementations cannot guarantee channel access before a finite deadline [6, 7]. Therefore, VANETs' real-time applications cannot predictably meet their deadlines. The design of the Chameleon-MAC algorithm facilitates MAC protocols that address important issues, such as predictably [6, 7], fairness [21] and energy conservation.

The *algorithmic* study of MAC protocols is a research area that considers the analysis of distributed MAC algorithms using theoretical models that represent the communication environment. The scope of this work includes distributed MAC algorithms for wireless ad hoc networks and adaptive self-stabilization.

MAC **algorithms.** ALOHAnet and its synchronized version Slotted ALOHA [1] are pioneering wireless systems that employ a strategy of "random access". Time division multiple access (TDMA) is another early approach where nodes transmit one after the other, each using its own timeslot, say, according to a defined schedule. The analysis of radio transmissions in ad hoc networks [19] and the relocation analysis of mobile nodes [30] show that there are scenarios in which MAC algorithms that employ a scheduled access strategy have lower throughput than algorithms that follow the random access strategy. However, the scheduled approach offers greater predictability, which can facilitate fairness [21] and energy conservation.

◦ *Non-self-stabilizing MAC algorithms for wireless ad hoc networks.* Wattenhofer's fruitful research line of *local algorithms* considers both theoretical [18, 20, and references therein] and practical aspects of MAC algorithms [43, and references therein] and the related problem of clock synchronization [28, and references therein]. For example, the first partly-asynchronous self-organizing local algorithm for vertex-coloring in wireless ad hoc networks is presented in [37]. However, this line currently does not consider MANETs. An example of a self-organizing MAC protocol that considers a single hop and no mobility is [35].

◦ *Non-self-stabilizing MAC algorithms for MANETs.* An abstract MAC layer was specified for MANETs in [24]. The authors mention algorithms that can satisfy their specifications (when the mobile nodes very slowly change their locations). MAC algorithms that use complete information about the nodes' trajectory are presented in [42, and references therein] (without considering the practical issues that are related to maintaining the information about the nodes' trajectories). A self-organizing TDMA algorithm that maintains the topological structure of its communication neighborhood is considered in [6, 7]. The authors use computer simulations to study the protocol, assuming that nodes have access to a global positioning system, and that transmission collisions can be detected.

◦ *Self-stabilizing MAC algorithms for wireless ad hoc networks.* Two examples of self-stabilizing TDMA algorithms are presented in [21, 23]. The algorithms are based on vertex-coloring and consider ad hoc networks (in which the mobile nodes may move very slowly). Recomputation and floating output techniques [11, Section 2.8] are used for converting deterministic local algorithms to self-stabilizing ones in [29]. However, deterministic MAC algorithms are known to be inefficient in the context of MANETs, as shown in [30]. There are several other proposals for self-stabilizing MAC algorithms for sensor networks [such as 2, 3, 25, 26]; however, none of them considers MANETs.

Adaptive self-stabilization. In the context of self-stabilization, adaptive resource consumption was considered for communication [16] and memory [cf. Update algorithm in 11]. Namely, after the convergence period, the algorithm's resource consumption is asymptotically optimal [as in 11] or poly-logarithmic times the optimal [as in 16]. In the context of MANETs, the idea of self-stabilizing Virtual Node (VN) [13–15] and tokens that perform random walks in order to adapt to the topological changes [17] was widely adopted by the theory and practice of MANETs. These concepts can implement a wide range of applications, such as group communication services [17], and traffic coordination and safety [44].

6.2 Conclusions

The large number of MAC algorithms and protocols for ad hoc networks manifests both the interest in the problem, as well as the lack of some commonly accepted and well understood methods that enable balancing the large number of trade-offs. The proposed adaptive and self-⋆ Chameleon-MAC algorithm and the systematic study presented here shed light on and bridge this situation: The extensive study, using the TOSSIM [32] simulation and the actual MICAz 2.4 GHz mote platform, of the aspects and parameters that are of interest in the practical realm, offer a gnomon to facilitate adoption and deployment in practice.

The study includes a wide, multi-dimensional range of mobility situations and demonstrates that the Chameleon-MAC algorithm is an alternative with improved overhead and fault-recovery periods compared to existing algorithms in the literature and in practice [1, 21, 30, 40]. To highlight a small example, in scenarios that exclude unrealistic situations, the Chameleon-MAC algorithm maintains greater throughput than the studied ones [1, 21, 30, 40], including TOSSIM's native MAC (CSMA/CA with back-off).

As a side-result, this work demonstrates that the relocation model is sufficiently abstract to describe a variety of situations (from limited mobility scenarios to concrete mobility models for VANETs) and detailed enough to allow comparative studies of different algorithms.

Another contribution in the paper, which was an intermediate step in our study and which can be of independent interest, is the analysis of the role of the local estimation (by the nodes) of the global model parameters and their impact on the algorithms. Interestingly, we discovered the average similarity ratio (ASR) that can estimate well the throughput of the studied algorithms.

References

[1] Abramson, N.: Development of the ALOHANET. IEEE Transactions on Information Theory 31(2), 119–123 (1985)

[2] Arumugam, M., Kulkarni, S.: Self-stabilizing deterministic time division multiple access for sensor networks. AIAA Journal of Aerospace Computing, Information, and Communication (JACIC) 3, 403–419 (2006)

[3] Arumugam, M., Kulkarni, S.S.: Self-stabilizing deterministic TDMA for sensor networks. In: Chakraborty, G. (ed.) ICDCIT 2005. LNCS, vol. 3816, pp. 69–81. Springer, Heidelberg (2005)
[4] Berns, A., Ghosh, S.: Dissecting self-* properties. In: SASO, pp. 10–19. IEEE Computer Society, Los Alamitos (2009)
[5] Bettstetter, C.: Smooth is better than sharp: a random mobility model for simulation of wireless networks. In: Meo, M., Dahlberg, T.A., Donatiello, L. (eds.) MSWiM, pp. 19–27. ACM, New York (2001)
[6] Bilstrup, K., Uhlemann, E., Ström, E.G., Bilstrup, U.: Evaluation of the IEEE 802.11p MAC method for vehicle-to-vehicle communication. In: VTC Fall, pp. 1–5. IEEE, Los Alamitos (2008)
[7] Bilstrup, K., Uhlemann, E., Ström, E.G., Bilstrup, U.: On the ability of the 802.11p MAC method and STDMA to support real-time vehicle-to-vehicle communication. EURASIP Journal on Wireless Communications and Networking 2009, 1–13 (2009)
[8] Camp, T., Boleng, J., Davies, V.: A survey of mobility models for ad hoc network research. Wireless Communications and Mobile Computing 2(5), 483–502 (2002)
[9] Chipcon Products from Texas Instruments, Texas Instruments, Post Office Box 655303, Dallas, Texas 75265. 2.4GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver (2008)
[10] Dijkstra, E.W.: Self-stabilizing systems in spite of distributed control. ACM Commun. 17(11), 643–644 (1974)
[11] Dolev, S.: Self-Stabilization. MIT Press, Cambridge (2000)
[12] Dolev, S. (ed.): ALGOSENSORS 2009. LNCS, vol. 5804. Springer, Heidelberg (2009)
[13] Dolev, S., Gilbert, S., Lynch, N.A., Schiller, E., Shvartsman, A.A., Welch, J.L.: Brief announcement: virtual mobile nodes for mobile ad hoc networks. In: Chaudhuri, S., Kutten, S. (eds.) PODC, p. 385. ACM, New York (2004)
[14] Dolev, S., Gilbert, S., Lynch, N.A., Schiller, E., Shvartsman, A.A., Welch, J.L.: Virtual mobile nodes for mobile ad hoc networks. In: Guerraoui, R. (ed.) DISC 2004. LNCS, vol. 3274, pp. 230–244. Springer, Heidelberg (2004)
[15] Dolev, S., Gilbert, S., Schiller, E., Shvartsman, A.A., Welch, J.L.: Autonomous virtual mobile nodes. In: DIALM-POMC, pp. 62–69 (2005)
[16] Dolev, S., Schiller, E.: Communication adaptive self-stabilizing group membership service. IEEE Trans. Parallel Distrib. Syst. 14(7), 709–720 (2003)
[17] Dolev, S., Schiller, E., Welch, J.L.: Random walk for self-stabilizing group communication in ad hoc networks. IEEE Trans. Mob. Comput. 5(7), 893–905 (2006)
[18] Goussevskaia, O., Wattenhofer, R., Halldórsson, M.M., Welzl, E.: Capacity of arbitrary wireless networks. In: INFOCOM, pp. 1872–1880. IEEE, Los Alamitos (2009)
[19] Haenggi, M.: Outage, local throughput, and capacity of random wireless networks. Trans. Wireless. Comm. 8(8), 4350–4359 (2009)
[20] Halldórsson, M.M., Wattenhofer, R.: Wireless communication is in APX. In: Albers, S., Marchetti-Spaccamela, A., Matias, Y., Nikoletseas, S., Thomas, W. (eds.) ICALP 2009. LNCS, vol. 5555, pp. 525–536. Springer, Heidelberg (2009)
[21] Herman, T., Tixeuil, S.: A distributed TDMA slot assignment algorithm for wireless sensor networks. In: Nikoletseas, S.E., Rolim, J.D.P. (eds.) ALGOSENSORS 2004. LNCS, vol. 3121, pp. 45–58. Springer, Heidelberg (2004)
[22] Holfelder, W., Johnson, D.B., Hartenstein, H., Bahl, V. (eds.): Proceedings of the Third International Workshop on Vehicular Ad Hoc Networks, VANET 2006. ACM, New York (2006)

[23] Jhumka, A., Kulkarni, S.S.: On the design of mobility-tolerant TDMA-based media access control (MAC) protocol for mobile sensor networks. In: Janowski, T., Mohanty, H. (eds.) ICDCIT 2007. LNCS, vol. 4882, pp. 42–53. Springer, Heidelberg (2007)

[24] Kuhn, F., Lynch, N.A., Newport, C.C.: The abstract MAC layer. In: Keidar, I. (ed.) DISC 2009. LNCS, vol. 5805, pp. 48–62. Springer, Heidelberg (2009)

[25] Kulkarni, S.S., Arumugam, M.U.: Sensor Network Operations, chapter SS-TDMA: A self-stabilizing MAC for sensor networks. IEEE Press, Los Alamitos (2006)

[26] Lagemann, A., Nolte, J., Weyer, C., Turau, V.: Mission statement: Applying self-stabilization to wireless sensor networks. In: Proceedings of the 8th GI/ITG KuVS Fachgespräch "Drahtlose Sensornetze" (FGSN 2009), pp. 47–49 (2009)

[27] Lee, H., Cerpa, A., Levis, P.: Improving wireless simulation through noise modeling. In: Abdelzaher, T.F., Guibas, L.J., Welsh, M. (eds.) IPSN, pp. 21–30. ACM, New York (2007)

[28] Lenzen, C., Locher, T., Sommer, P., Wattenhofer, R.: Clock Synchronization: Open Problems in Theory and Practice. In: van Leeuwen, J., Muscholl, A., Peleg, D., Pokorný, J., Rumpe, B. (eds.) SOFSEM 2010. LNCS, vol. 5901, pp. 61–70. Springer, Heidelberg (2010)

[29] Lenzen, C., Suomela, J., Wattenhofer, R.: Local algorithms: Self-stabilization on speed. In: Guerraoui, R., Petit, F. (eds.) SSS 2009. LNCS, vol. 5873, pp. 17–34. Springer, Heidelberg (2009)

[30] Leone, P., Papatriantafilou, M., Schiller, E.M.: Relocation analysis of stabilizing MAC algorithms for large-scale mobile ad hoc networks. In: Dolev [12], pp. 203–217

[31] Leone, P., Papatriantafilou, M., Schiller, E.M., Zhu, G.: Chameleon-MAC: Adaptive and stabilizing algorithms for media access control in mobile ad-hoc networks. Technical Report 2010:02, Chalmers University of Technology (2010), ISSN 1652-926X

[32] Levis, P., Lee, N., Welsh, M., Culler, D.E.: TOSSIM: accurate and scalable simulation of entire tinyos applications. In: Akyildiz, I.F., Estrin, D., Culler, D.E., Srivastava, M.B. (eds.) SenSys, pp. 126–137. ACM, New York (2003)

[33] Luby, M.: Removing randomness in parallel computation without a processor penalty. J. Comput. Syst. Sci. 47(2), 250–286 (1993)

[34] Metzner, J.J.: On Improving Utilization in ALOHA Networks. IEEE Transactions on Communications 24(4), 447–448 (1976)

[35] Patel, A., Degesys, J., Nagpal, R.: Desynchronization: The theory of self-organizing algorithms for round-robin scheduling. In: SASO, pp. 87–96. IEEE Computer Society, Los Alamitos (2007)

[36] Polastre, J., Szewczyk, R., Culler, D.E.: Telos: enabling ultra-low power wireless research. In: IPSN, pp. 364–369. IEEE, Los Alamitos (2005)

[37] Schneider, J., Wattenhofer, R.: Coloring unstructured wireless multi-hop networks. In: Tirthapura, S., Alvisi, L. (eds.) PODC, pp. 210–219. ACM, New York (2009)

[38] Specifications ASTM, E2213-03. Standard Specification for Telecommunications and Information Exchange between Roadside and Vehicle Systems - 5 GHz Band Dedicated Short Range Communications Medium Access Control and Physical Layer (September 2003)

[39] Stuedi, P., Alonso, G.: Wireless ad hoc VoIP. In: Workshop on Middleware for Next-generation Converged Networks and Applications, Newport Beach, California, USA (November 2007)

[40] Takagi, H., Kleinrock, L.: Throughput analysis for persistent CSMA systems. IEEE Transactions on Communications 33(7), 627–638 (1985)

[41] Žerovnik, J.: On the convergence of a randomized algorithm frequency assignment problem. Central European Journal for Operations Research and Economics (CEJORE) 6(1-2), 135–151 (1998)
[42] Viqar, S., Welch, J.L.: Deterministic collision free communication despite continuous motion. In: Dolev [12], pp. 218–229
[43] Wattenhofer, R.: Theory Meets Practice, It's about Time. In: 36th International Conference on Current Trends in Theory and Practice of Computer Science (SOFSEM), Czech Republic (2010)
[44] Wegener, A., Schiller, E.M., Hellbrück, H., Fekete, S.P., Fischer, S.: Hovering data clouds: A decentralized and self-organizing information system. In: de Meer, H., Sterbenz, J.P.G. (eds.) IWSOS 2006. LNCS, vol. 4124, pp. 243–247. Springer, Heidelberg (2006)