Efficient SINR Queries For CSMA/CA Simulation

Alexander Kröller
Braunschweig University of Technology
Mühlenpfordtstr. 23
38106 Braunschweig
Germany
a.kroeller@tu-bs.de

Max Pagel
Braunschweig University of Technology
Mühlenpfordtstr. 23
38106 Braunschweig
Germany
pagel@ibr.cs.tu-bs.de

Dennis Pfisterer
University of Lübeck
Ratzeburger Allee 160
23538 Lübeck, Germany
pfisterer@itm.uni-luebeck.de

ABSTRACT

We propose an efficient simulation technique for the CSMA medium access protocol. It is based on the well-established model of using signal to interference and noise ratio (SINR), which is very accurate and allows for highly realistic predictions of collisions. However, traditional implementations require $O(n)$ time to compute interference values. We evaluate how to speed this up by using efficient data structures such as k-d-trees or geometric hash tables in our implementation. There are different levels of accuracy, some of which allow for $O(\sqrt{n} + a)$ lookups, where a is the number of nearby senders. We demonstrate the achievable speedup and discuss accuracy tradeoffs for different settings using the Shawn simulator.

Categories and Subject Descriptors

I.6.3 [**Simulation and Modeling**]: Applications; C.2.1 [**Network Architecture and Design**]: Wireless communication.

General Terms

Algorithms, Performance

Keywords

Simulation, Shawn, Wireless Networks, Signal-to-Noise-Ratio.

1. INTRODUCTION

It is expected that in a future Internet of Things, Wireless sensor networks (WSNs) will play a major role and the devices will outnumber the hosts on the Internet by several orders of magnitude. As such networks are not available today, simulations are imperative to test novel algorithms and protocols. Any such simulation tool needs to deal with certain aspects of wireless communication. One important aspect here is *interference*, which causes otherwise unrelated communication to fail because the wireless signals interfere with each other. In real networks, a nodes' MAC protocol deals with this issue. Therefore a simulation needs models to detect when interferences occur, and need to provide communication that behaves as if running a matching MAC protocol.

MSWiM'10, October 17–21, 2010, Bodrum, Turkey.

We discuss an implementation of such a communication simulation. It simulates the CSMA/CA MAC protocol based on a well-established interference model, the SINR inequality. We present ways to speed up the simulation of this model. We argue that there is some loss in accuracy using our methods, but they are little compared to the valuable benefit in simulation speed. As a proof of concept, we have implemented our methods for the discrete event simulator Shawn [4]. The code will be available to the public through regular updates to Shawn. As it is based on C++ templates, it is easily portable to other simulators such as Ns-2 [7].

This paper is organized as follows: In the next section, we describe fundamentals underlying this work. In Section 3, our approach is presented and discussed. Afterwards, in Section 4, we evaluate the implementation of these methods using a number of test scenarios and compare them to a highly accurate simulation, as well as to the existing CSMA/CA simulation in Shawn. The final Section 5 summarizes our finding and sketches future improvements.

2. FUNDAMENTALS

This section introduces the simulated protocol and the signal propagation model and following, discusses the properties of two geometric data structures that we use to speed up the simulation.

2.1 CSMA/CA & MAC Layer

In wireless environments, collision detection is not possible for senders and the hidden terminal problem (where the signals of two senders superimpose only at the receiver and cause a collision) further complicates this issue. To improve this situation, mechanisms have been developed to reduce collision probability. The most frequently used algorithm is CSMA/CA (Carrier Sense Multiple Access/Collision Avoidance), which exists in two flavors: *slotted* and the *unslotted*, as described in the IEEE 802.15.4 Standard [2]. In ad-hoc and sensor networks, the unslotted variant dominates. Consequently, we consider the unslotted variant.

2.2 Signal-to-Interference-and-Noise Ratio

Accurate modelling of signal propagation is a key issue in simulating wireless communication. There exists a wide variety of models, ranging from purely combinatorial ones [8], that abstract from influences of the physical environments to realistic propagation models [6]. For a simulation of CSMA/CA, we obviously not only require models defining whether a sender's signal reaches a receiver, but also models for whether communication fails due to interference with other senders. We decided to use the well-established Signal-to-Interference-and-Noise Ratio (SINR) [1] for that matter.

The underlying propagation model is based on perfectly circular signal spreading with exponential decay. Consider a sender at position x using sending power $P(x)$. A receiver at y will pick up the signal with strength

$$p(x,y) := P(x)d(x,y)^{-\alpha} , \tag{1}$$

where $d(\cdot,\cdot)$ denotes the Euclidean distance. The evironmental parameter α is usually assumed to be in the range $[2,4]$, where the default $\alpha = 2$ corresponds to free space propagation. A message can be decoded if and only if the signal strength at the receiver is above a hardware-dependent level β, i.e.,

$$p(x,y) \geqslant \beta . \tag{2}$$

Note that the previous definition of p implies that signal strength isolines are perfect circles, which is believed to be unrealistic to some extent. A trivial remedy would be to incorporate random influences into p, such as the Radio Irregularity Model [9]. However, this is outside the scope of this work, which focusses on efficient evaluation and approximation of these functions, rather than the functions themselves. It does pose a useful addition to the system though, and it is planned to be implemented in future work.

Given p, the SINR defines whether a message can be successfully deciphered at a receiver, or is lost due to interference or ambient noise. Consider a message of a sender at x using sending power $P(x)$. Assume there is a set S of other simultaneous senders, where each $s \in S$ uses power $P(s)$. The SINR inequality defines that the message from x is received at y if and only if

$$\mathrm{SINR}_S(x,y) := \frac{p(x,y)}{N + \sum_{s\in S} p(s,y)} \geqslant \gamma , \tag{3}$$

where the parameter γ defines the minimum signal quality for the receiver's circuitry. N is a parameter reflecting ambient noise.

2.3 2D Neighbor Queries

To simulate signal powers and the SINR of simulated messages, the simulation needs to keep track of all sending and receiving nodes. It must be able to efficiently answer neighbor queries, i.e., reporting all receivers within a given range from some sender, and vice versa all senders within range of a receiver. Our algorithm maintains two data structures: S holds all sending nodes, and R holds nodes currently receiving messages, i.e., all nodes within communication range of at least one sender. As S and R are rapidly changing during the simulation, they must also provide efficient insert and delete. There are several data structures available for 2D neighbor queries, of which we implemented two:

k-d-Trees. The first implementation is based on k-d-Trees [3], which are standard for range queries. They employ a recursive partition of the plane, resulting in a search tree. The tree can be used to report all points within an axis-parallel rectangle in time $O(\sqrt{n}+a)$, where n is the total number of points in the data structure, and a is the size of the output. For circular queries, the tree is queried for a bounding box, and the result is then filtered to the desired output. For dynamic input, the tree needs a rebalancing strategy, which tracks imbalance in the tree and rebuilds it whenever neccessary.

Geometric Hash Tables. The second implementation uses geometric hash tables [5]. These are based on a novel hash function: The plane is tesselated into triangles, and each point is assigned three hashes corresponding to the three corners of the triangle containing the point. Given triangles whose size depends on a fixed query range R, the hashes can be used to answer neighbor queries. One can show that every point within distance R of a query point shares at least one hash with it. Therefore it is sufficient to add every point with all three hashes into a hash table. Queries can then be answered by running three lookups in the table and postfiltering the result. The complexity of these operation equal the hash table's complexities, as the hashing overhead is constant.

3. OUR CONTRIBUTION

Generally, simulating a message transmission is done in three steps, which correspond to the phases of CSMA/CA. When the transmission simulation is invoked with a new message from some node s, the following steps are performed: **1. "CCA":** Simulate the backoffs. This is done by estimating the noise level $N + \sum_{s'\in S} p(s',s)$ at the new sender. If this exceeds a certain threshold β_{CCA}, simulate a backoff by registering a re-evaluation event at a suitable point in the future. **2. "Transmit":** Start transmitting the message: Add s to S. Find additional nodes that receive the new signal and add them to R. For all nodes in R, estimate whether the SINR inequality (3) is violated. For each such receiver, mark the transmission as failure. Finally, enqueue an event for the time when the transmission of s is finished. **3. "Deliver":** In the event of s finishing its transmission, remove s from S. For each potential receiver of the message, see if the message was marked as failed in the meantime. All surviving messages get passed to the message handlers of the recipients. Remove all nodes no longer receiving signals from R.

For the simulation, a balance between accuracy of the simulation and runtime needs to be achieved. We have developed two separate models. Both allow Shawn to base message transmission on actual SINR evaluation, thereby providing an important step to bring simulations closer to reality.

3.1 The "Simple" Model

The first approach aims at high simulation speed at the cost of reduced accuracy. Note that, by Equation (2), no message can be transmitted over distances exceeding $(P_{\max}/\beta)^{1/\alpha}$. For evaluating interference, we extend this range using a predefined and constant factor $\varphi > 1$, resulting in the *Noise Range* $\varrho := \varphi(P_{\max}/\beta)^{1/\alpha}$.

The simulation restricts all SINR evaluations to this range, assuming signals traveling more the ϱ to be too weak to have a substantial impact on the system, so we use the signal propagation function

$$p'(x,y) := \begin{cases} p(x,y) & \text{if } d(x,y) \leqslant \varrho \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

instead of p. Simulating CSMA/CA based on p' can be done efficiently. We only require neighbor queries using disks of radius ϱ, and hence both k-d-Trees and Geometric Hash Tables can be used to maintain S and R

To be precise, the following procedures are used: i) In the CCA phase, the noise level is computed over a disk of radius ϱ around the new sender, and ii) in the Transmit phase, the set R' of affected nodes is computed using a neighbor query. All nodes in R store their current noise level, which needs to be increased by the additional noise caused by s. New receivers, i.e., nodes in $R' \setminus R$, need to compute an initial noise cache value using a neighbor query.

Note that the actual noise values are only needed for the initialization of new receivers in $R' \setminus R$. In other cases, the evaluation only checks whether the level exceeds β_{CCA} resp. γ, allowing to sometimes abort the summation prematurely.

3.2 The "Extended" Model

The previous model is a highly efficient means to approximate CSMA/CA, but it does sacrifice accuracy. In reality, p does not

drop to zero within a fixed range, and hence a sufficiently large set of senders can cause interference over arbitrary large distances. The "Extended" model accounts for this by evaluating noise levels over potentially all nodes, while attempting to reduce simulation complexity by splitting up and sorting the summations based on the current overall distribution of S and R. The resulting algorithm needs to run neighbor queries with varying radii, which is impossible using Geometric Hash Tables. Hence it is limited to k-d-Trees.

In the CCA phase, the simulation needs to check whether the new sender currently receives a noise level exceeding β. For that matter, it computes the minimum distance δ at which $|S|$ senders using power $P_{\max}$ could be placed without preventing the sender from starting the transmission. This distance is computed as

$$\delta := \left(\frac{|S|P_{\max}}{\beta - N}\right)^{1/\alpha} + \varepsilon \tag{5}$$

for arbitrary small $\varepsilon > 0$. Now the simulation runs a query over range δ to check for senders within that range from s. If there are none, the noise level must be below β_{CCA}, and the transmission is started. If the query finds a sender, the CCA test is evaluated over the whole sender set S (with premature abort if possible).

Once the sender starts transmitting, the simulation needs to evaluate which other transmissions fail due to the new signal. Instead of checking all of them, we restrict this to those receivers where a failure seems likely. For that matter, we compute the smallest distance ν that satisfies the following: It is possible for a receiver to decode a signal with strength β despite $|S| - 1$ senders at distance ν sending interference with full sending power $P_{\max}$. According to Equation (3), we can compute this as

$$\nu := \left(\frac{(|S| - 1)P_{\max}}{\gamma/\beta - N}\right)^{1/\alpha} + \varepsilon \tag{6}$$

for an arbitrary small $\varepsilon > 0$. Now, using a query around s, we compute the set R_ν of receivers within distance ν from s. While this does not necessarily represent all receivers whose transmission will fail due to s sending, it is a reasonable approximation.

Next, the simulation adds R_ν to R and updates the noise level for nodes in R_ν. This uses cached noise values with a re-evaluation procedure: i) If the new SINR estimate exceeds γ, the transmission is sure to fail and can be marked as such. ii) Compute a worst-case SINR estimate, based on cached values and estimates for added noise from other senders. We assume all senders that are further than R_ν from the receiver to be just infinitesimally further. If this estimate does not reach γ, the message cannot fail no matter where the senders are located, and abort the procedure. iii) If neither of the two cases apply, we extend the search range. We compute a range such that the existence of a single sender within this extended range would guarantee message failure and re-evaluate the two cases above. This is iterated until the range eventually covers the whole network. To speed up the evaluation, we use incremental range queries. This is done using rectangular queries instead of disks, see Figure 1. It is not guaranteed that the CSMA/CA simulation follows the SINR inequalities accurately. Especially the recomputation of ν causes a "breathing effect" in the range queries. However, we are convinced that the outcome is reasonably close to an exact simulation but considerably more efficient. This is evaluated in the following section.

4. EVALUATION

To evaluate the efficiency and accuracy of our proposed methods, we conducted simulations on two different scenarios. For each, we varied the networks size to investigate the scalability. Every configuration was ran several times to even out stochastic outliers. We used five CSMA/CA implementations: i) the three ones presented in this paper (Simple with k-d-Trees and Geometric Hash Tables), ii) the standard CSMA/CA simulation of Shawn, and iii) a model using perfect SINR evaluation (This is achieved by running Simple with parameter $\varphi = \infty$). We expect Shawn's existing implementation to be the fastest (because of it's simplified model) and the perfect model to be the most accurate. This serves as the extremes between which the new algorithms are to be placed. For each scenario, we randomly sampled points from a square region and ran the following protocols on the resulting network:

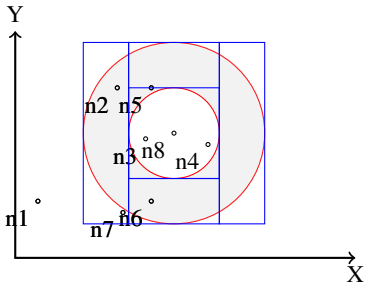

Figure 1: Extending range queries. When computing the SINR for n_8 in the extended model, four rectangular queries are used to expand the query range in each iteration.

"Hello World": Each node broadcasts a single message, letting CSMA/CA handle collisions. As all nodes start sending at the same time, it leads to the maximal number of concurrent senders. This produces large amounts of noise and generally the most collisions.

"Tree Routing": One node is selected as a sink for all routing messages. It floods the network with tree creation messages to build up a tree. Afterwards, 10 randomly selected nodes send a message to the sink. As all routing messages are sent to the same sink at the same time, they are likely to produce collisions in the sink's area.

There are some parameters in the simulation, for which we used values as defined by the IEEE 802.15.4 standard [2]:

β	at least $-85\text{dBm} = 3.16 \cdot 10^{-9}\text{mW}$
β_{CCA}	$\beta + 10\text{dB} = 3.16 \cdot 10^{-8}\text{mW}$
γ	$4\text{dB} = 2.5$
$P_{\max}$	$0\text{dBm} = 1\text{mW}$

The "Simple" model requires the noise range factor φ as a parameter. Obviously it has a large impact on this model, as setting $\varphi = 0$ results in not having message collisions at all, whereas $\varphi = \infty$ turns Simple into an exact SINR evaluation. We ran prestudies to calibrate φ to apparantly useful values. These are $\varphi = 17$ for the "Hello World" scenarios and $\varphi = 8$ for the other two.

To determine the simulation accuracy, we evaluated the collision probability for all messages that are sent by a protocol, see Figures 2 and 3. The results match our expectations. Shawn's existing simulation is far from accurate, as the model is very simple. This is clearly visible with the "Hello World" protocol. It is roughly a factor 2 from the reference for the other protocols, and therefore of limited use. For massive-scale simulation, where simulation speed is crucial and sacrificing accuracy is feasible, the model may still be useful. Simple (in both incarnations) is always better, sometimes even as good as Extended. The difference between Simple and Extended in the "Hello World" protocol evaluation (Figure 2)

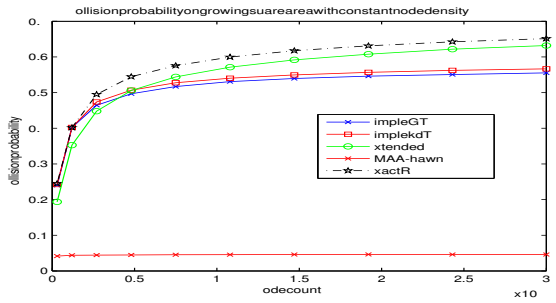

Figure 2: Message collision probabilities for "Hello World".

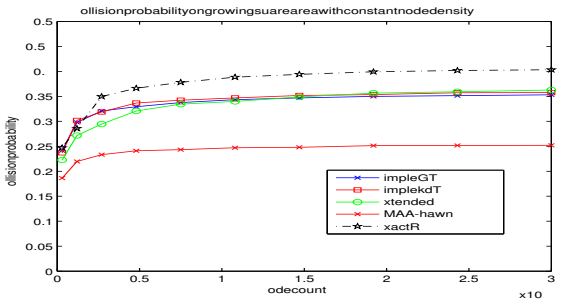

Figure 3: Message collision probabilities for "Tree Routing".

indicates that one should choose Extended over Simple when a massive amount of simultaneous transmissions is to be expected, whereas the accuracy of Simple should suffice in all other scenarios. Figures 4 to 5 show the runtimes. All models are considerably faster (roughly 60%) than exact SINR evaluation. They even outperform CSMA/CA-Shawn in the "Tree Routing" scenarios, where the efficient geometric data structures lead to very fast simulations. This is mostly due to the few collisions, where spatial dispersion of concurrent senders allows for quick SINR decisions.

5. CONCLUSION AND FUTURE WORK

We have presented means to speed up the evaluation of SINR-based simulation of CSMA/CA as used in the IEEE 802.15.4 standard. This allows a simulator to reduce simulation accuracy by a small amount to reduce the total runtime of a simulation by up to 60%, compared to an exact evaluation of the associated formulae. This runtime reduction stems from the usage of efficient geometric data structures as well as simplifications in the actual computations.

We implemented the procedures for the discrete event simulator

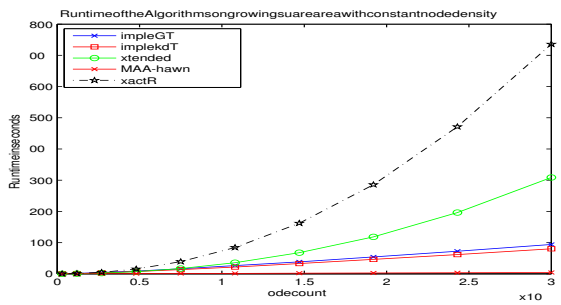

Figure 4: Total simulation runtime for "Hello World".

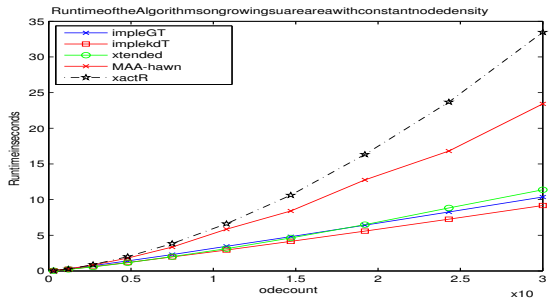

Figure 5: Total simulation runtime for "Tree Routing".

Shawn, and ran exhaustive tests to evaluate the accuracy. We find that the models are usually very close to an exact simulation. This indicates that it is feasible to use the models for all simulations where message collisions are not the focus, but rather just one of several effects that have an impact on the simulation outcome.

Acknowledgements

This work has been partially supported by the EU within the 7th Framework Programme under contract 215270 (FRONTS).

6. REFERENCES

[1] P. Gupta and P. Kuma. The capacity of wireless networks. *IEEE Trans. on Information Theory*, 46(2).

[2] IEEE Computer Society. Wireless medium access control (MAC) and physical layer (PHY) specifications for low-rate wireless personal area networks (WPANs). specification, IEEE Computer Society, 2006.

[3] R. Klein. 2 edition.

[4] A. Kröller, D. Pfisterer, C. Buschmann, S. P. Fekete, and S. Fischer. In *Design, Analysis, and Simulation of Distributed Systems (DASD' 05)*, Apr.

[5] T. Neylon. A locality-sensitive hash for real vectors. In *Proc. Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms (SODA '10)*.

[6] V. Sridhara and S. Bohacek. Realistic propagation simulation of urban mesh networks. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 51(12), 2007.

[7] University of Southern California. Ns-2: Network simulator-2. http://www.isi.edu/nsnam/ns/.

[8] H. Wen, C. Lin, Z.-J. Chen, H. Yin, T. He, and E. Dutkiewicz. An improved Markov model for IEEE 802.15.4 slotted CSMA/CA mechanism. *Journal of computer science and technology*, (3), May.

[9] G. Zhou, T. He, S. Krishnamurthy, and J. A. Stankovic. In *Proc. 2nd International Conf. on Mobile Systems, Applications, and Services (MOBISYS'04)*.