Modeling Obstacles in INET/Mobility Framework: Motivation, Integration, and Performance

Hermann S. Lichte, Jannis Weide
University of Paderborn
Paderborn, Germany
{hermann.lichte, jannis}@upb.de

ABSTRACT

Wireless network protocols are commonly evaluated through simulations. The achieved results may vary significantly with the modeled propagation environment. Protocols that use carrier sensing for collision avoidance may not operate properly in the presence of obstacles (e.g. buildings) that shield hosts from each other. These failures may not be recognized when using a long-term stochastic model (e.g. log-normal shadowing), leading to inadequate simulation results. Studying a simple carrier-sensing protocol, we find that when shielding effects are properly modeled, they increase collisions dramatically with increasing transmission power as opposed to the stochastic model. Thus, we formally introduce a model that describes shielding effects and which is still simple enough to be efficiently implemented. We discuss how such implementation looks like, using the Mobility Framework and the INET Framework for OMNeT++ as specific examples. We identify connection-specific caches to be crucial for the run-time performance of the model's implementation.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*; C.4 [**Performance of Systems**]: Modeling Techniques; I.6.5 [**Simulation and Modeling**]: Model Development

General Terms

Design, Performance

1. INTRODUCTION

Nowadays, computer scientists rely on simulations for evaluating new wireless network protocols. Especially, in the area of Wireless Sensor Networks (WSNs) and Mobile Ad Hoc Networks (MANETs), the alternative of building large-scale prototypes for experimentation may not be possible due to resource limitations, e.g. the hardware may be too expensive or the deployment may be impracticable. In contrast, simulations are easily programmed, easily deployed, and – with appropriate computing power – even scale well in numbers that are infeasible for prototyping. Unfortunately, results obtained through simulations can be misleading just as easily if inappropriate models are used. The most common errors are due to overly simplistic radio and environmental models, e.g. [3]:

- Transmission areas are circular.
- All radios have equal range.
- If I can hear you, you can hear me.
- If I can hear you at all, I can hear you perfectly.

In the OMNeT++ community, several frameworks have evolved for computer network simulation, e.g. the Mobility Framework, MiXiM, and the INET Framework. Many people have contributed suitable models that depart from the overly simplistic assumptions mentioned above, e.g. the publicly available ChSim [7] offers an implementation of a time-correlated and frequency-correlated Rayleigh fading model.

In this paper we address the issue of modeling obstacles in the propagation environment. Commonly, a stochastic model is used to capture the effect of obstacles, e.g. log-normal shadowing. Such a model fails to properly describe *specific* obstructions in the propagation environment. Consider Figure 1 as a simple, motivating example. Here, the obstacle in the center of the four hosts a, b, c, and d hinders communication between $a \leftrightarrow d$ and $b \leftrightarrow c$ by severely attenuating signals on these channels. On the other hand, $a \leftrightarrow b$, $a \leftrightarrow c$, $b \leftrightarrow d$, and $c \leftrightarrow d$ are not affected.

Wireless network protocols that rely on carrier-sensing to avoid collisions may fail in scenarios such as the one depicted in Figure 1. Since b and c are shielded from each other, they cannot sense each other's transmission and, if both have data to transmit, they may simultaneously start a transmission which will then collide at d. This paper shows that protocol failures caused by shielding effects cannot be captured properly by a stochastic model. Thus, we describe an alternative obstacle model in Section 2 that properly captures shielding effects to derive more meaningful simulation results. We discuss an implementation of this obstacle model in Section 3 from

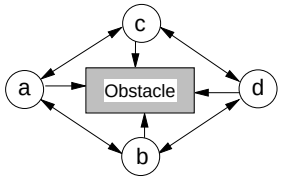


Figure 1: An obstacle in-between four hosts a, b, c, and d may shield hosts $a \leftrightarrow d$ and $b \leftrightarrow c$ from each other.

OMNeT++ 2009, Rome, Italy.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680

start to finish using the Mobility Framework and INET Framework as concrete examples of how the model can be integrated into an existing framework. Especially, in simulations where network sizes may be large, an *efficient* implementation is needed. Hence, we discuss optimization strategies of the implementation and measure its performance in Section 4, finding that caches are crucial for efficiency. Finally, we evaluate an opportunistic wireless network protocol exemplarily in Section 5 using both models. We discuss how the models dominate the results and, thus, the conclusions to be drawn from them. In Section 6, we briefly summarize the related work that this paper is based on, concluding the paper in Section 7.

The complete implementation of our obstacle model for the Mobility Framework is publicly available, but with the model and implementation issues described in this paper, implementations for other frameworks should be straightforward.

2. SYSTEM MODEL

We assume that two hosts H_i and H_j with their respective locations $(x(H_i), y(H_i))$ and $(x(H_j), y(H_j))$ can communicate without error if

$$\mathrm{SINR}(i,j) \geq \nu, \tag{1}$$

i.e. the Signal to Interference plus Noise Ratio (SINR) exceeds a certain threshold ν required for successful reception. The threshold ν typically depends on the modulation and coding used. The SINR for a transmission from i to j is given by

$$\mathrm{SINR}(i,j) := \frac{P_t\, a(i,j)}{N_0 + \sum_{k \neq i} P_k\, a(k,j)} \tag{2}$$

where $a(i,j)$ describes the effect of the wireless channel on the transmitted signal power P_i. All nodes other than i cause interference if they are simultaneously transmitting, contributing to the noise level N_0.

2.1 Propagation

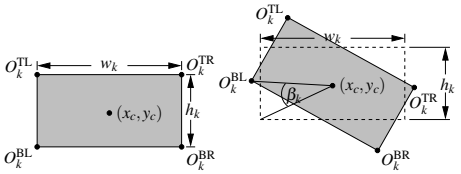
Path loss The well-known Friis equation describes how the signal transmitted by some host i to host j decays over distance d_{ij} in free space [8]:

$$a_{\mathrm{PL}}(i,j) = \frac{G_t G_r \lambda^2}{(4\pi)^2 d_{ij}^2 L}$$

G_t and G_r describe the transmitter and receiver antenna gains, respectively, L captures circuit losses, and λ refers to the wave length. The ratio between received signal power and transmitted signal power is referred to as path loss.

Shadowing Obstructions in the propagation environment cause the signal strength to vary even for equal distances. Measurements indicate that these variations can be described by a stochastic model, e.g. by a zero-mean Gaussian distributed random variable with standard deviation σ (*log-normal shadowing*), expressed by the term $a_{\mathrm{SH}}(i,j)$. This model is an aggregate of several physical effects, e.g. reflection, diffraction, and scattering [8].

Fading In a multi-path propagation environment, slight changes in a host's position may lead to drastic fluctuations in the received signal strength. This is caused by constructive and destructive interference of the signals received on multiple paths with their phase shifts being random. We use the fading model described in [5] in conjunction with the obstacle model proposed here to jointly represent large-scale and small-scale effects. We capture the small-scale effects of fading through the term $a_{\mathrm{FD}}(i,j)$.



(a) Non-rotated (b) Rotated obstacle

Figure 2: Obstacles are modeled as rectangular shapes.

The SINR at the receiver is influenced by all three phenomena, thus

$$a(i,j) = a_{\mathrm{PL}}(i,j)\, a_{\mathrm{SH}}(i,j)\, a_{\mathrm{FD}}(i,j). \tag{3}$$

2.2 Obstacles

Obstacles $O_k \in \mathbf{O}$ are modeled as rectangular shapes as depicted in Figure 2. The following set of parameters characterizes an obstacle:

- Center position $(x_c(O_k), y_c(O_k))$
- Dimension (width w_k and height h_k)
- Angle of rotation β_k
- Attenuation factor a_k

Given two hosts H_i and H_j, an obstacle O_k affects the transmission between the hosts if any of the obstacle's borders intersects with the line segment connecting H_i and H_j. From computational geometry, we use the algorithm ANY-SEGMENTS-INTERSECT(S) that determines whether any of n line segments given by the set S intersect [1]. In this case, we need to specify $S = S_k^{ij}$ such that it contains the line segment that represents the connection between the two hosts, namely $\overline{H_i H_j}$, as well as the four line segments that constitute the borders of the obstacle O_k. For a non-rotated obstacle, the corner points are given using width w_k and height h_k as follows:

$$x(O_k^{\mathrm{TL}}) = x_c(O_k) - w_k/2 \text{ and } y(O_k^{\mathrm{TL}}) = y_c(O_k) - h_k/2$$
$$x(O_k^{\mathrm{TR}}) = x_c(O_k) + w_k/2 \text{ and } y(O_k^{\mathrm{TR}}) = y_c(O_k) - h_k/2$$
$$x(O_k^{\mathrm{BL}}) = x_c(O_k) - w_k/2 \text{ and } y(O_k^{\mathrm{BL}}) = y_c(O_k) + h_k/2$$
$$x(O_k^{\mathrm{BR}}) = x_c(O_k) + w_k/2 \text{ and } y(O_k^{\mathrm{BR}}) = y_c(O_k) + h_k/2$$

If the obstacle is rotated at an angle β_k, then, for every corner point (x,y) of the non-rotated obstacle, the corresponding corner point (x',y') after rotation is given by:

$$x' = x_c(O_k) + (x - x_c(O_k))\cos\beta_k - (y - y_c(O_k))\sin\beta_k$$
$$y' = y_c(O_k) + (x - x_c(O_k))\sin\beta_k + (y - y_c(O_k))\cos\beta_k$$

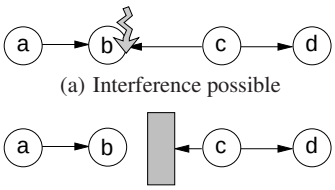
Altogether, this defines the set of line segments S_k^{ij} as

$$S_k^{ij} := \left\{ \overline{H_i H_j}, \overline{O_k^{\mathrm{TL}} O_k^{\mathrm{TR}}}, \overline{O_k^{\mathrm{TR}} O_k^{\mathrm{BR}}}, \overline{O_k^{\mathrm{BR}} O_k^{\mathrm{BL}}}, \overline{O_k^{\mathrm{BL}} O_k^{\mathrm{TL}}} \right\}$$

If the test ANY-SEGMENTS-INTERSECT(S_k^{ij}) succeeds, the attenuation a_k of the obstacle further reduces the strength of the received signal. The *total* attenuation caused by all obstacles in the simulated propagation environment is then given as

$$a_{\mathrm{OB}}(i,j) = \prod_{O_k \in \mathbf{O}} a_k^{b_k^{ij}} \tag{4}$$

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680



(a) Interference possible

(b) No interference due to shielding

Figure 3: An obstacle that obstructs the line segment connecting two hosts may prevent them from interfering.

where the exponent b_k^{ij} indicates whether the obstacle O_k intersects, i.e.

$$b_k^{ij} = \begin{cases} 1 & \text{if ANY-SEGMENTS-INTERSECT}(S_k^{ij}) \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

We obtain the SINR at the receiver in presence of obstacles by multiplying (2) with (4) and inserting the resulting term into (1). An efficient implementation of (4) is crucial for the run-time of simulations. In Section 4 we discuss both a trivial approach and a more efficient implementation using caching and compare their run-times.

2.3 Connectivity

Theoretically, a signal sent out by one host affects all other hosts. Due to attenuation, the received power at hosts very far away from the sender may be low enough to be negligible in simulations. Thus, many wireless frameworks for OMNeT++ define a Maximum Interference Distance (MID) up to which hosts can possibly disturb each other. This is practically motivated as it reduces the run-time of simulations by avoiding unnecessary processing at far away hosts. The MID is given by solving the Friis equation for distance and assuming that a minimum power at the receiver $P_{\min}$ is required for interference and the maximum power being transmitted is $P_{\max}$.

$$d_I = \sqrt[\alpha]{\frac{P_{\max}}{P_{\min}} \left(\frac{\lambda}{4\pi}\right)^2} \tag{5}$$

The presence of obstacles in the propagation environment may shield two hosts from interfering with each other as shown in Figure 3. This is because the additional attenuation imposed by the obstacle affects the MID. The obstacle's shielding effect may cause two hosts to be disconnected, increasing the probability of the hidden node problem [4]. Thus, shielding effects must be considered when computing the MID as they have direct consequences on the behavior of wireless network protocols, illustrated in detail in Section 5. Considering obstacles for computing the network's connectivity makes the MID connection-specific. It is modeled by (6) where the maximum transmission power decays proportionally to the additional attenuation imposed by all obstacles that obstruct the connection.

$$d_I^{ij} = \sqrt[\alpha]{a_{\text{OB}}(i,j) \frac{P_{\max}}{P_{\min}} \left(\frac{\lambda}{4\pi}\right)^2} \tag{6}$$

3. INTEGRATION

In this section we describe how the obstacle model integrates into the Mobility Framework and the INET Framework. We assume the reader to be familiar with the basic concepts of the OMNeT++ simulator and these frameworks.

3.1 New modules and classes

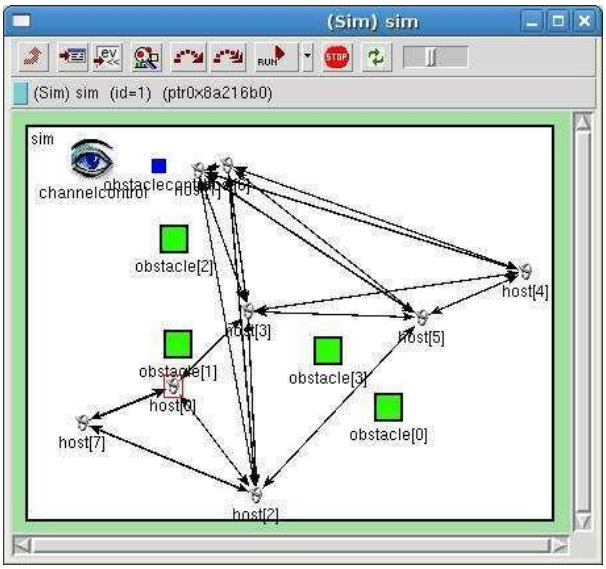


Figure 4: Obstacles may cause hosts to be disconnected.

Here, we describe modules and classes that represent fundamental concepts of the extension that implements the obstacle model. We believe that they are independent of the frameworks and need to be implemented in any case.

Matter module Captures the characteristic properties of obstacles introduced in Section 2.2. The associated Matter class computes the corner points of the obstacle during initialization and offers methods for querying these points.

Attenuation class Objects of this class store attenuation factors for different frequencies, making it possible to define frequency-specific attenuation, i.e. $a_k(f)$, as well as a single attenuation value a_k for all frequencies.

Obstacle module Defines an obstacle to be placed on the *playground* where all hosts reside. This two-fold implementation of Matter/Obstacle module was chosen as it allows existing simulation objects (e.g. hosts) to be given physical attributes as well. We give an illustrative example when discussing the Mobility Framework integration.

ObstacleControl class This core class implements the geometrical algorithm ANY-SEGMENTS-INTERSECT(S). Further, it provides a method for returning the additional attenuation for transmission between two hosts H_i and H_j given by (4). In a trivial implementation, this method always iterates over all registered obstacles $O_k \in \mathbf{O}$, invoking the algorithm ANY-SEGMENTS-INTERSECT(S_k^{ij}) to obtain the attenuation factors. In Section 4 we see that this imposes tremendous run-time and one can do better by employing appropriate caching strategies.

3.2 Mobility Framework

Figure 4 shows a TkEnv window of a simple wireless network modeled with the proposed extension. Some hosts are disconnected due to obstacles intersecting their connections, e.g. host[0] and host[5] are shielded from each other by obstacle[3]. We now discuss the peculiarities of a Mobility Framework integration.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680

3.2.1 Making obstacles mobile

The Mobility Framework provides many mobility models for moving hosts during a simulation. For this, every host module contains a Mobility sub-module that controls the movement of that host independently from other hosts. It was a primary goal of the integration to be able to re-use the existing mobility models for obstacles as well. In our implementation of extended mobility, two new cases need to be considered.

Physical hosts Hosts in the Mobility Framework are modeled as points, whereas in our integration a Matter sub-module may be added to a Host module, thereby giving the host a physical dimension with the antenna at its center. The existing mobility models may still be used to move the host and the matter associated with it.

Movable obstacles Similarly, an Obstacle module that contains a Matter sub-module allows a Mobility sub-module to be included as well. Note that it is not feasible to have Matterless obstacles.

For the existing mobility models to work, collisions with the playground's border and other objects need to be handled. Border handling is already part of the original Mobility Framework. It is implemented in the super-class of all mobility models, BasicMobility. For hosts modeled as points, a collision with the border is detected by testing whether, after the move, the point is outside the playground. This has been adapted to obstacles by testing whether any of the obstacle's corner points are outside the playground. The Mobility sub-module defines a parameter called *border policy* that indicates how a collision with the border is handled (e.g. reflection). With mobile obstacles, collisions between obstacles and hosts with physical dimensions become possible, thus requiring further collision detection and handling. Collisions between obstacles can be detected by checking if one of the line segments that belong to the moving obstacle intersects with any segment belonging to obstacles that are in the vicinity of the move. Unfortunately, even if no intersection is detected, a collision may still have occurred if, after the move, the obstacle is completely contained in another. Thus, an additional check is necessary. The above algorithm is implemented in a new method of BasicMobility called checkCollision(). Similar to border handling, we define a *collision policy* per obstacle that defines how to resolve collisions individually.

3.2.2 Additional attenuation

The additional attenuation caused by obstacles needs to be considered in two cases.

Connectivity Connectivity (see Section 2.3) between hosts is determined in ChannelControl. Here, updateConnections() must compute for every pair of hosts the connection-specific interference distance. For this, calcInterfDist() queries the ObstacleControl module for additional attenuation values whenever connections need to be updated instead of using a constant interference distance.

Reception During transmission, the SnrEval module collects the SINR values and stores them in an SnrList. This list is passed on to the Decider module for checking whether a frame was received correctly, e.g. using the model given by (1). The method calcRcvdPower() computes the power of the received signal considering the effects of the channel, as stated by (3). For the obstacle model, we additionally use $a_{\mathrm{OB}}(i,j)$ given by (4), thus the ObstacleControl module needs to be queried in calcRcvdPower() for the additional attenuation.

Entry	Description
sim.numObstacles	Number of obstacles on playground
sim.obstacle[k]	Module representing obstacle O_k
.mobility	Mobility sub-module
.x	Placement on x-axis $x_c(O_k)$
.y	Placement on y-axis $y_c(O_k)$
.speed	Speed of movement
.updateInterval	Time interval to update the position
.collisionPolicy	Policy for collisions between obstacles
.borderPolicy	Policy for collisions with the border
.matter	Matter sub-module
.attenuation	Attenuation factor(s) $a_k(f)$
.width	Width w_k
.height	Height h_k
.angle	Angle of rotation β_k

Table 1: OMNeT++ configuration of the obstacle extension.

Since ObstacleControl is the core of the implementation, we briefly describe its interface.

calcObstacleDecrease() Given either the IDs of two hosts or their coordinates, this method returns the total attenuation of obstacles intersecting the connection of the given hosts.

getIntersectingObstacles() Given the coordinates of two hosts, this method stores pointers to obstacle modules in a vector for any obstacle intersecting the connection of the given hosts. This method is useful if a developer needs to perform some operation on some or all obstacles obstructing the connection.

Additionally, the following helper methods are available and also used internally.

testIntersect() Returns a boolean value indicating whether two line segments, each given by their endpoints, intersect. This implements the algorithm ANY-SEGMENTS-INTERSECT(S) with $|S| = 2$, but it can be easily extended to arbitrary S.

getIntersection() Given two intersecting line segments as its arguments, this method returns the coordinates of the point where the line segments intersect.

getIntersectionLength() Given a line segment and a pointer to an obstacle, this method returns the length of the line segment that lies within the obstacle. This is useful if the attenuation caused by an obstacle shall be made proportional to its thickness.

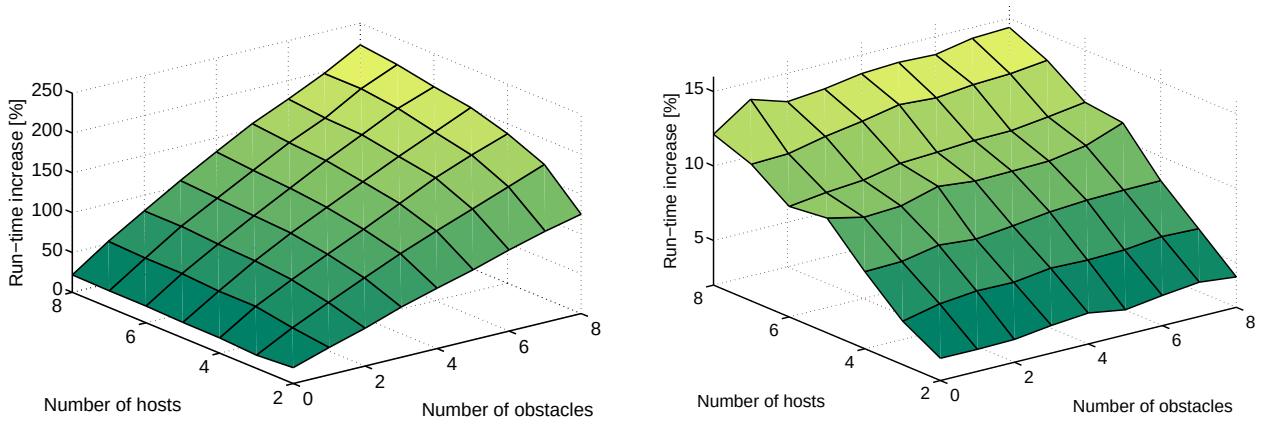
3.2.3 Configuration

Obstacles are defined in the configuration file similar to hosts. Table 1 shows how the characteristic properties of obstacles introduced in Section 2.2 are mapped to module parameters. Note that the obstacle extension need not be deterministic. The attenuation factor a_k is a volatile parameter so it can be specified using a random variable in the configuration, allowing for varying attenuation when an obstacle intersects.

3.3 INET Framework

An integration into the INET Framework is straight forward, since the INET Framework is based on the Mobility Framework. It shares the ChannelControl class as well as the mobility models so that the integration described above applies in the same way. A

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680



(a) Run-time increase of a trivial extension

(b) Reduced run-time increase with caching

Figure 5: Comparing the increase in run-time without (left) and with (right) caching for small networks with static obstacles.

notable difference is that the power received at some host is calculated in an IReceptionModel sub-class. For computing the received power, the coordinates of the hosts need to be known, thus requiring a modification of this interface.

4. IMPROVING RUN-TIME BY CACHING

This section investigates the increase in run-time caused by the obstacle extension and introduces an effective caching strategy to speed up simulations significantly. The time-consuming part of the extension is the algorithm that checks for intersections, e.g. calcObstacleDecrease() and getIntersectingObstacles() in our integration. These methods are called on two occasions.

1. When hosts or obstacles move, the connections must be updated.
2. During transmission of a frame, the additional attenuation caused by obstacles must be considered.

We first estimate the worst-case run-time for a network consisting of n hosts and m obstacles.

Connections We need to check for all connections whether an obstacle intersects or not. With n hosts there can be up to $\frac{1}{2}n(n-1)$ connections, where $\frac{1}{2}$ accounts for symmetry. For every connection, m obstacles need to be checked for intersection, leading to a worst-case run-time of

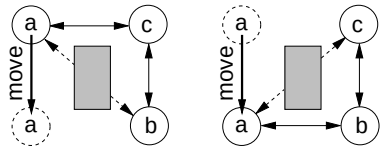
$$\frac{1}{2}mn(n-1) = O(mn^2).$$

Transmission In a fully connected network, a host can transmit to at most $n-1$ other hosts. With m obstacles, the worst-case run-time for a single host is

$$m(n-1) = O(mn).$$

We first investigate the increase in run-time of a trivial implementation where the test of intersection is executed whenever obstacles' attenuation factors need to be queried. We define the increase g in run-time by

$$g = \frac{t_{\text{enabled}} - t_{\text{disabled}}}{t_{\text{disabled}}}$$



(a) *Before* moving (b) *After* moving

Figure 6: All connections of a moving host (here host a) need to be invalidated.

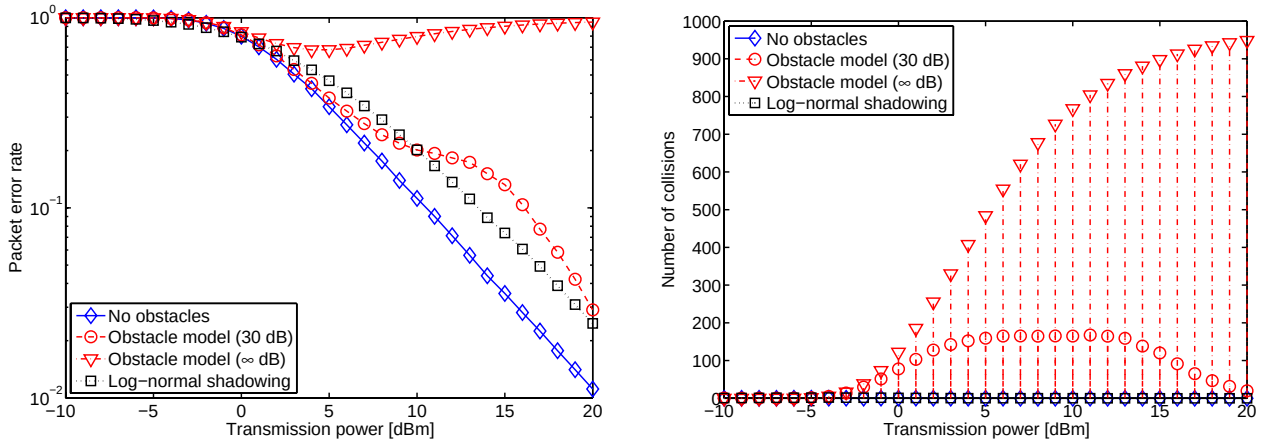
where t_{enabled} denotes the run-time when the obstacle extension is enabled and t_{disabled} when the extension is disabled. Figure 5(a) shows the increase in run-time for varying numbers of hosts and obstacles. We see that for very small networks (up to 8 hosts) with few obstacles (up to 8) the increase can already be as high as 250 %. Clearly, a trivial implementation is infeasible, so we discuss how an efficient caching strategy can improve the run-time of the obstacle extension.

We introduce a cache C for attenuation factors. This cache stores for every connection $\overline{H_iH_j}$ the total attenuation of all obstacles intersecting it (which may be zero). If the cache is valid, instead of executing the intersection tests to compute $a_{\text{OB}}(i,j)$ given by (4), the cached value $C(i,j)$ is returned.

Unfortunately, the value stored in the cache is not valid forever. If a host or an obstacle changes its position, the cache needs to be invalidated. It is important to invalidate only those cache entries which are affected by the movement. Otherwise, every movement would invalidate the entire cache, leading to the worst-case run-time of $O(mn^2)$ again. Therefore, we illustrate what caching strategies can improve for moving hosts and obstacles.

Host moves Whenever a host moves, all its connections need to be invalidated. Consider Figure 6 for an example. Host a has two connections $\overline{ab}$ and $\overline{ac}$ which are both affected by the move. An obstacle obstructs $\overline{ab}$ which after the move of a becomes unobstructed. For $\overline{ac}$, the situation is vice versa. The connection $\overline{bc}$ is not affected since it does not involve a. In general, the movement of some host causes all its connections to be invalidated, whose number is at most $n-1$. Since for every connection all obstacles have to be checked,

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680



(a) Packet error rate (b) Number of collisions at host d

Figure 7: Evaluation of an opportunistic wireless network protocol that relies on sensing frames for coordinating transmissions. The stochastic model fails to capture shielding effects, leading to better performance results than when the obstacle model is used (left). Missing coordination due to shielding increases the number of collisions at the destination d dramatically (right).

the overhead of this strategy is only $O(mn)$ as opposed to $O(mn^2)$.

Obstacle moves An obstacle shielding two hosts from each other obstructs their connection. When the obstacle moves, these connections might become free and, thus, need to be invalidated. Unfortunately, there is no easy relation to identify the connections that might become obstructed. If all connections would need to be checked, this again leaves us with a run-time of $O(mn^2)$.

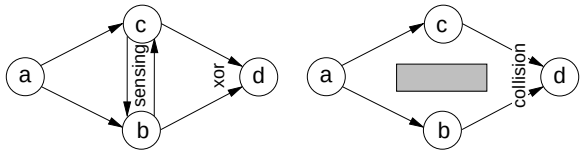
Figure 5(b) shows the increase in run-time when caching is used and obstacles are static. Comparing to the previous results, now with 8 hosts and 8 obstacles the run-time increase is only 15 % as opposed to 250 % without caching. This indicates that caching is vital for an efficient obstacle extension and developers should take the effort in implementing it.

5. EVALUATION

The quality of the propagation model chosen for a simulation significantly impacts the quality of its results and, when an inappropriate model is used, may lead to wrong conclusions. We now illustrate the usefulness of the obstacle model by using an opportunistic protocol in a toy topology as an example.

5.1 A simple opportunistic protocol

Consider the topology shown in Figure 8(a) where hosts communicate on orthogonal channels that are separated in time. Host a wants to send data to host d with host c as an intermediate hop. The transmission of DATA frames must be acknowledged by the receiver using ACK frames. Another host b in the vicinity of a, c, and d can overhear frames between a and c. If b overhears DATA from a but not the associated ACK from c, it retransmits DATA in the hope that either c or d can decode it [6]. If d decodes, the data has reached the final destination using an *opportunistic* transmission via b. If d cannot decode, there is still a chance that the opportunistic retransmission helped c to decode now and, thus, c can continue the transmission to d. Overhearing the ACK frame is essential for coordinating the transmission between b and c.



(a) Unobstructed (b) Obstructed

Figure 8: An obstacle in-between four hosts a, b, c, and d may shield hosts $a \leftrightarrow d$ and $b \leftrightarrow c$ from each other, leading to collisions at d due to missing coordination between b and c.

In presence of obstacles, this simple protocol suffers from increased collisions. Refer to Figure 8(b) for an example where an obstacle shields hosts $a \leftrightarrow d$ and $b \leftrightarrow c$ from each other. In this case, the coordination fails. Assume that host c has successfully decoded DATA from a and so did b. Now c will acknowledge it and continue the transmission to d. In presence of the obstacle, the ACK frame never reaches b, causing b to retransmit the DATA frame as well since it must assume that c did not receive it. With this failed coordination, a collision at host d is possible because the DATA transmitted by c interferes with the DATA transmitted by b. In fact, the correct reception of DATA at both b and c will, without coordination, inevitably result in a collision at d and, thus, in a packet error.

This behavior arises from the design of the protocol where broadcasts are exploited for coordination. Unfortunately, if the performance of such a protocol is evaluated using simple stochastic shadowing models (e.g. log-normal shadowing), the adverse effects of shielding do not become obvious.

5.2 Performance results for both models

We evaluate the example protocol in a fading scenario. Figure 7(a) shows the end-to-end Packet Error Rate (PER) from host a to d for varying transmission powers. Without any obstacle model (i.e. $a_{\mathrm{SH}} = a_{\mathrm{OB}} = 1$) the PER constantly decreases when the transmission power increases, showing best performance. With log-normal shadowing enabled, more packets are lost due to higher

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680

variations in received power. These additional losses are limited within a constant offset as compared to the obstacle-less case, depending on the standard deviation σ. Again, the PER diminishes for high powers. However, using the obstacle extension with a perfectly shielding (i.e. the attenuation is infinite) obstacle instead causes almost all packets to be lost even for high transmission powers. This occurs because the number of collisions at *d* increases dramatically as shown in Figure 7(b). An infinite attenuation causes $a(i, j) = 0$ and, thus, the SINR to become zero for any power P_i. No coordination is possible between *b* and *c* and the protocol will fail if both hosts receive the DATA frame. This especially happens at high power when it is very likely that both *b and c* can sucessfully decode DATA frames.

For medium transmission powers (0-10 dBm), the PER shows a slight decrease. This is because the protocol works as intended if only one intermediate hop, namely *b or c* but not both, receive data. Then the other host remains silent and no coordination is needed. This happens for medium powers when correct reception over *both* $\overline{ab}$ and $\overline{bc}$ is unlikely, but as soon as both receive (with increasing power), this inevitably results in a collision.

Even if the obstacle's attenuation is finite (e.g. 30 dB), the PER performance varies significantly as opposed to the log-normal shadowing model. As soon as the number of collisions increases, the slope of the PER declines. But due to the finite attenuation, for very high powers, the PER approaches that of the obstacle-less case again.

6. RELATED WORK

Jardosh et al. [2] describe an obstacle mobility model that captures the effect of obstacles on the movement behavior of nodes and on the obstruction of transmissions. They apply the same model of obstacles, i.e. reducing the signal power when a transmitted signal passes an obstacle. Unlike this work, they focus more on the suitability of their mobility model in presence of obstacles rather than on the propagation characteristics of the obstacle model compared to stochastic models. Also, they do not discuss challenges of the implementation, which we do thoroughly for OMNeT++.

In [4] we already envisioned this obstacle model to be implemented for MiXiM, but due to the early state of that framework at the time, no implementation was carried out and, consequently, no performance measurements have been made. So the feasibility and performance of the envisioned approach and its impact on protocol evaluation is first shown in this paper.

7. CONCLUSION

By simulating a wireless network protocol with OMNeT++ we have shown how simulation results are greatly dominated by the propagation models used. Particularly, protocols that rely on carrier-sensing for coordinating transmissions require a proper model of shielding effects. Otherwise, the performance of such a protocol may be overestimated in environments where shielding occurs.

The proposed obstacle model and its implementation in OMNeT++ is suitable for investigating such adverse effects while still being simple enough to be efficient. The model is suitable to describe large-scale shielding effects, e.g. those caused by buildings, but let alone cannot capture small-scale effects in the same way. However, considering all details that influence propagation in the smaller scale as realistic as possible, e.g. scattering, refraction, and diffraction, requires raytracing-based simulations with a precise description of the propagation environment and impracticable runtimes. This is when stochastic models are still of value. In fact, when our obstacle model for describing large-scale shielding effects is combined with stochastic models that describe the smaller scale, both large-scale and small-scale effects can be adequately reflected in simulations while still keeping complexity low enough for simulating large networks in feasible amount of time.

We contribute our implementation to the OMNeT++ community in the hope that with the availability of more sophisticated yet efficient models of the propagation environment, the existing frameworks, e.g. the INET Framework, will become invaluable tools for research on wireless network protocols that are open to anyone.

8. REFERENCES

[1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. MIT Press, 2nd edition, 2001.

[2] A. P. Jardosh, E. M. Belding-Royer, K. C. Almeroth, and S. Suri. Real-world environment models for mobile network evaluation. *IEEE J. Sel. Areas Commun.*, 23(3):622–632, Mar. 2005.

[3] D. Kotz, C. Newport, and C. Elliott. The mistaken axioms of wireless-network research. Technical report, Dartmouth College, July 2003.

[4] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. Parker, O. Visser, H. S. Lichte, and S. Valentin. Simulating wireless and mobile networks in OMNeT++: The MiXiM vision. In *Proc. 1st Intl. Workshop on OMNeT++*, Mar. 2008.

[5] H. S. Lichte and S. Valentin. Implementing MAC protocols for cooperative relaying: A compiler-assisted approach. In *Proc. 1st Intl. Conf. on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools)*, Mar. 2008.

[6] H. S. Lichte, S. Valentin, H. Karl, I. Aad, L. Loyola, and J. Widmer. Design and evaluation of a routing-informed cooperative MAC protocol for ad hoc networks. In *Proc. Ann. Joint Conf. of the IEEE Computer Societies (INFOCOM)*, Apr. 2008.

[7] T. Pawlak and S. Valentin. Chsim – a wireless channel simulator for OMNeT++. Project website `wwwcs.upb.de/cs/chsim`, 2005.

[8] T. S. Rappaport. *Wireless Communications – Principles and Practice*. Prentice Hall, 2nd edition, 2002.

Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2009.5680
http://dx.doi.org/10.4108/ICST.SIMUTOOLS2009.5680