Modeling Application Spreading using Mobile Ad Hoc Networks

Á. Horváth[*], K. Farkas[*,**]
[*] Institute of Informatics and Economics, University of West Hungary, Sopron, Hungary
[**] Department of Telecommunications, Budapest University of Technology and Economics, Budapest, Hungary
{horvath, farkas}@inf.nyme.hu

***Abstract*—Information spreading via mobile ad hoc connections is a frequently investigated research topic today. However, studying the characteristics of application spreading by exploiting direct connections between the user devices has not got too much attention so far. In this paper, we use Closed Queuing Networks to model application spreading in such a way, and to capture the users' behavior, too. With our model we are able to investigate the following issues: 1) how many pieces of a given application can be sold in a given population; and 2) how much time is needed to reach the state in which there is no more interest in the population to purchase the application. We give analytical approximations on the lower and upper bounds of the number of application purchases and validate our results via simulations. Moreover, we present some simulation results to show trend lines which demonstrate the influence of our model's parameters on the application spreading process.**

I. Introduction

Knowing the characteristics of application spreading is important not only from technical but also from economical point of view. The application provider has to know or at least should be able to assess how many pieces of a given application can be sold in a given population; how much time is needed to reach the state in which there is no more interest to purchase the application; which factors influence the spreading mechanism and how.

The traditional way of application distribution is using a central entity, e.g., an Internet webshop. The users can select their favorite applications, purchase and download the application software. However, there exist modern communication paradigms, which can and will influence the spreading process. For example, mobile devices (e.g. laptops, PDAs, smart phones) can communicate directly between one another by forming an ad hoc network, so the applications can be downloaded directly from a node in that network. The participants of the ad hoc communication can try out applications, which can motivate them to purchase the ones they liked. Since secure communication has been a challenging issue in mobile ad hoc environments yet, purchasing is available only via a traditional way.

Exploiting direct connections between user devices in application spreading is a rarely investigated research topic. Moreover, there are many unexploited factors in mobile ad hoc communication, which also have an effect on application spreading. For example, if a user can try out an application and has a community experience, e.g., with a multi-player game, she will be more motivated in purchasing it than she would have seen only some advertisements. Thus, mobile ad hoc networks can open new ways for sales and marketing processes.

In this paper, we propose the use of Closed Queuing Networks (CQNs) [1] for modeling application spreading aided by mobile ad hoc networks. The CQN is an appropriate stochastic model for describing the rapid appearance and disappearance of a given number of nodes in ad hoc networks. Moreover, we can assign intensities to state transitions, by which the time behavior of the spreading process can be described. With the right setting of our model's parameters an application provider will be able to assess how much profit can be realized in a given time if application spreading takes place via the described way.

Moreover, we give analytical approximations on the lower and upper bounds of the number of application purchases. We show simulation results to validate our analytical approximations and present how our spreading model does work and how its dynamics does look like. We show also trend lines, in which we demonstrate how the parameters of our model do influence the spreading process and the number of application purchases.

The procedures and technical details to describe, discover, deploy, manage and terminate the application software are beyond the scope of this paper. Detailed discussion about these issues can be found in [2]. Similarly, we do not touch security issues in this work, related discussions can be found, e.g., in [3, 4, 5, 6].

The rest of the paper is organized as follows. We give a short overview about the related works in Section II. In Section III, we describe our communication model. In Section IV, we propose a spreading model based on CQNs. We show our analytical and simulation results in Section V and Section VI, respectively. Finally, we give a short summary in Section VII.

II. Related Works

Studying the characteristics of application spreading has not got too much attention so far; however, its importance is increasing with the proliferation of modern communication paradigms. On the contrary, epidemic spreading, spreading of computer viruses and information dissemination are frequently investigated research topics dealing with issues similar to our problem.

In [7], the authors investigate how a virus propagates in a real network. They propose a model based on the adjacency matrix of the network to determine an epidemic threshold, below which the number of infected nodes decreases exponentially. The threshold is $1 / \lambda_{1,A}$, where

$\lambda_{1,A}$ denotes the first eigenvalue of the adjacency matrix A. In [8], the authors model the spreading of computer viruses with scale-free networks, and also give an epidemic threshold, which is an infection rate. The infections can spread in the network, if their infection rate is above the epidemic threshold. Epidemic spreading is also used for modeling information spreading, such as the susceptible-infected-resistant (SIR) model [9], or other topology-dependent models [10, 11]. In [12], the authors deal with the commercial usage of mobile ad hoc networks and present a radio dispatch system with mobile ad hoc communication, in which the network topology is the key element of information dissemination.

In our case, the network topology is not critical since no real-time information dissemination is needed between the users in the process we model. The above mentioned papers do not deal with the direct commercial use of mobile ad hoc networks, except [12], which considers information dissemination as a tool, and not as a goal. Moreover, none of these papers deals with application spreading and does capture the users' behavior.

III. Communication Model

For modeling the application spreading process, we use the following communication model. The applications are server-client type multi-user applications and each one has two versions, a trial and a full version. In our model, we examine the spreading of a given application in a given population, which is composed of the users, which are interested in the use of the application. We do not consider uninterested individuals as users because they do not influence the spreading process. Henceforth, we refer to the examined population simply as users.

Users communicate with each other using mobile devices. We can categorize them into different classes depending on whether they possess either the full version or the trial version of the application, or none of them. We name the different classes based on the terminology of epidemics, since our model shows similarities with epidemic spreading models. We call the users as *infected* if they possess the full version of the application, *susceptible* if they possess the trial version and *resistant* if they have got none of them, or they have already lost the interest of using it.

The mobile devices of the users form ad hoc networks from time to time, and communicate with each other directly. In such networks, the users can download the trial version of the application and try it out if there is at least one infected user in the network. If the user liked the application, she can buy it using a traditional purchasing way, e.g., via the Internet (this phase is necessary, since the implementation of a secure and reliable payment and licensing method in mobile ad hoc environment is a challenging issue). Later, if the user purchased the application, she can use it or even spread its trial version further.

Users possessing the trial version will be motivated in purchasing the full version only if limitations apply to using the trial. Hence, there is a limit (*leech limit*[1]) that restricts how many nodes possessing the trial version (*leech*[1]) can connect to a node possessing the full version (*seed*[1]). In this sense, the seeds can be considered as servers, which can serve a limited number of clients. A seed is always an infected user, while a leech may be either an infected or a susceptible user. Figure 1 shows what happens when a susceptible user purchases the application. The green devices depict infected users, while the yellow and red ones depict susceptibles. Since the leech limit is two, only two susceptible users (the yellow ones) can connect to the only infected user and two users (the red ones) have to wait. After one of them purchased the full version of the application, they can use it, too.

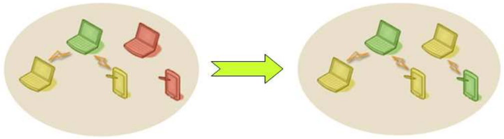

Figure 1. Change of application usage when a susceptible user purchases the application

Furthermore, we define three different user types based on their behavior. *Type*$_A$ users are interested in using the application; therefore, if they liked it they will purchase it. *Type*$_B$ users are also interested in using the application, but if they can find a seed from time to time which they can connect to, they will not purchase the application. In the cases when no free seed is available, *Type*$_B$ users will also purchase the application with a given probability. *Type*$_C$ users never buy the application; however, they influence the spreading process.

IV. Modeling Application Spreading

In this section, we present our spreading model and describe how we can use it.

A. Spreading Model

A CQN is appropriate for modeling stochastic processes by describing the rapidly changing network topology. Moreover, we can define the state transitions of the CQN with transition intensities, what allows us to examine the time behavior of the spreading process.

Figure 2 shows our CQN model, in which the different states represent the whole user population. Each user is in one state depending on her actual user class. Resistant users which possess neither the full version nor the trial version of the application are in state 0. Initially, all nodes are in this state. Susceptible users currently not using the application (passive susceptibles) are in state 1. Susceptible users currently using the application (active susceptibles) are in state 2. Similarly, the passive and active infected users are in state 3 and 4, respectively.

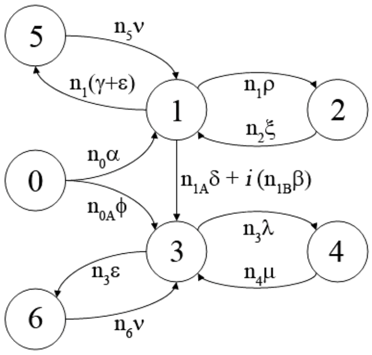

Figure 2. The proposed CQN for modeling application spreading

[1] After the terminology of BitTorrent [13]

Users which possess the trial version of the application, but already lost the interest in using it are in state 5. Similarly, users which possess the full version, but already lost their interest are in state 6. Users in state 5 and 6 are also called resistant users, such as the users in state 0. The Greek letters represent the transition intensities with regard to a single user between the states, n_x represents the number of users in state x, while n_{xA}, n_{xB} and n_{xC} represent the number of Type$_A$, Type$_B$ and Type$_C$ users in state x, respectively ($n_x = n_{xA} + n_{xB} + n_{xC}$). The transition intensity is a real number assigned to the state transition, which denotes how many times a state transition takes place during a given time interval. These transitions are described in Table I.

TABLE I.
DESCRIPTION OF THE STATE TRANSITIONS IN THE PROPOSED CQN

Transitions	Description
1 → 2	A susceptible user starts to run the application and tries to connect to a seed in the network. If it does not succeed, she has to wait.
2 → 1	A susceptible user stops running the application.
3 → 4	An infected user starts to run the application. Thus, either she tries to connect to a seed, or will be a seed herself to which leeches can connect.
4 → 3	An infected user stops running the application.
0 → 1	A resistant user gets the trial version of the application.
3 → 6	An infected user lost the interest in using the application.
1 → 5	A susceptible user lost the interest in using the application. The γ additional intensity represents that susceptible users lose the interest faster because of the waiting probability.
0 → 3	A resistant user purchased the application. It is possible that someone purchases the application without trying out it. In our model, we allow this state transition only to Type$_A$ users, so n_{0A} depicts the number of Type$_A$ users in state 0.
1 → 3	A susceptible user purchased the application. n_{1A} and n_{1B} depict the number of Type$_A$ and Type$_B$ users in state 1, respectively. This transition is enabled only for Type$_B$ users when there is no free seed available in the network which they can connect to. Therefore, the indicator variable i is zero if there is no free seed available, and one otherwise. Type$_C$ users never purchase the application, so this state transition is not allowed to take place for Type$_C$ users. However, Type$_C$ users can also connect to seeds, so their presence increases the probability of application purchasing of Type$_B$ users.
5 → 1	It is possible that a resistant user, which lost the interest in using the trial version, wants to use the application again after a while. If so, her state changes to susceptible.
6 → 3	Similarly, if a resistant user, which lost the interest in using the full version, wants to use the application again her state changes to infected.

The users form ad hoc networks from time to time and change their state depending on 1) whether they have got any version of the application or lost the interest in using it; or 2) whether they start the application or stop using it.

B. *Usage of the Spreading Model*

A system state can be unambiguously described by the user distribution (n_0, n_1, n_2, n_3, n_4, n_5, n_6). The state transition values with regard to a single user (α, β, γ, δ, ε, ϕ, λ, μ, ν, ρ and ξ) are the parameters of our model, which can be set experimentally. The holding time h of each system state can be computed by the following way:

$$h = \frac{1}{\sum\limits_{\forall state\_x} out_x}, \qquad (1)$$

where out_x denotes the sum of intensities for transitions leaving state x. The holding time h shows us how much the expected value of the spent time is in that system state. The next system state can be generated by computing the ratio of the transition intensities. For example, the probability that the next transition in the system will be transition 0 → 1 is $n_0 \cdot \alpha \cdot h$. Upon selecting the transition that takes place, we move one user from the source state to the destination state of the selected transition and compute the holding time of the new system state. The transition intensities, from which we derive the holding time, change after each transition due to the user distribution changes.

If a user reaches state 5 or 6, it means she lost the interest in using the application. However, it is still possible that she will be interested later again. Therefore, we allow users to return from these states, but only with low intensity to ensure that the system will reach sooner or later the state (final system state) in which all users are in either of state 5 and state 6. In the final system state, the holding time is large because ν is low. Hence, further transitions need very much time to take place, which means that the interest to purchase the application is very low. Though the system is not in steady state, our investigations will finish reaching this situation, because all users lost the interest in using the application.

By summing the holding times we can determine how much time is needed to reach the final system state, while the number of transition 1 → 3 and 0 → 3 will show how many pieces of the application software were sold during this time. By implementing our model in a simulator and evaluating the simulation results, we can learn the dynamics of the spreading process and the attitude of the different user types.

V. ANALYTICAL RESULTS

In this section, we show the computation of the total number of application purchases (tp) during the time interval till the system reaches the final system state. It is composed from two parts, the total number of application purchases of Type$_A$ users (tp_A) and the total number of application purchases of Type$_B$ users (tp_B) (the number of application purchases of Type$_C$ users is zero by definition). Thus, $tp = tp_A + tp_B$. We give an analytical estimate for tp_A, and validate our results via simulations in the next section. On the other hand, tp_B is hard to compute analytically, because it depends on the indicator variable i, so we give only an upper bound on it, and derive this value by simulation in the next section.

In our model, there are two ways to purchase the application software. Transition 0 → 3 means that a user purchased the application without trying it out (direct purchase), while transition 1 → 3 means that a user purchased the application after she could try it out (indirect purchase).

Initially, each user is in state 0, which is a source state. Since each user will leave this state (as we mentioned above, we do not consider the uninterested individuals as users) and never return, we can compute the number of direct purchases (*dp*) as follows:

$$dp = n_A \cdot \frac{\phi}{\alpha + \phi}, \tag{2}$$

where n_A denotes the number of Type$_A$ users, α and ϕ are transition intensities with regard to a single user in our model (cf. Figure 2). As we mentioned above, we consider only Type$_A$ users in computing *dp*. Equation (2) means that *dp* users, which are Type$_A$ users, will purchase the application without trying it out, while $n_A \cdot \alpha / (\alpha + \phi) + n_B + n_C$ users will not purchase it without trying it out (n_B and n_C denote the number of Type$_B$ and Type$_C$ users, respectively), so their state will be susceptible.

Since Type$_C$ users do not purchase the application at all, our further investigations are limited only to $n_A \cdot \alpha / (\alpha + \phi) + n_B$ users. They can either purchase the application or lose the interest in using it. Since tp_B is hard to compute analytically, we give an analytical estimate only on $n_A \cdot \alpha / (\alpha + \phi)$, while we give an upper bound on tp_B.

There are two ways how susceptible users can change their state: 1) they can become either resistant by transition 1 → 5, or 2) infected by transition 1 → 3. Since the source of both transitions is state 1, the ratio of these transition intensities determines what the ratio is between the users, which become resistant or infected. It is possible, that users in state 5 return to state 1 and later they will purchase the application and become infected; however, its probability is very small in our model. So, we can estimate the number of indirect purchases of Type$_A$ users (idp_A) as

$$idp_A = n_A \cdot \frac{\alpha}{\alpha + \phi} \cdot \frac{\delta}{\delta + \gamma + \varepsilon}, \tag{3}$$

where δ, γ and ε are single user transition intensities in our model (cf. Figure 2). We can give an upper bound on the indirect purchases of Type$_B$ users (idp_B) as

$$idp_B \le n_B \cdot \frac{\beta}{\beta + \gamma + \varepsilon}, \tag{4}$$

where β is also a single user transition intensity of our model (cf. Figure 2). Equality holds only if the indicator variable *i* is equal to one. Summing (2) and (3) we can estimate tp_A, which is also a lower bound on *tp*, while we can get the upper bound on *tp* by summing (2), (3) and (4):

$$dp + idp_A \le tp \le dp + idp_A + idp_B \tag{5}$$

VI. Simulation Results

In this section, we present the simulations, by which we have investigated the time behavior of the spreading process, and show trend lines to demonstrate the influence of the parameter settings.

We have developed a Java based simulator software, which works as follows. It computes the holding time of the initial state, stores it and generates the transition that takes place using weighted random numbers. The next system state is created by moving one user from the source to the destination of the selected transition. In the next system state, we compute and store the holding time again, and so on. The whole process runs until we reach the final system state first. We can store statistics in every system state, and combining them with the holding time we can investigate also the time behavior of the spreading process.

A. Dynamics of the Spreading Process

In the following, we describe three scenarios to show the dynamics of the spreading process and validate our analytical results. In every scenario, we ran the simulations with different parameter settings, which we collected in Table II. The dimension of the parameters (Greek letters in Table II) is 1 / hour. In each case, we repeated the simulations 10 times, and we got very similar results. Thus, we picked up randomly one simulation run in every scenario for investigations (cf. Figure 3, 4 and 5).

With these simulations our goal is not to find the correct set of parameters. This can be done only experimentally, because it depends on many things, such as the popularity of the application or the environment where the ad hoc communication takes place. Rather, we try to set the parameters as realistic as possible based on good sense. For example, $\lambda = 2 \cdot 10^{-2}$ / hour means that an infected user starts the application once in every 50 hours on average, while $\mu = 9 \cdot 10^{-1}$ / hour means that she is using it for about 67 minutes on average.

In Simulation 1, we investigate the dynamics of the spreading process, and validate our analytical results. Using (2) and (3), we can estimate *dp* and idp_A, such as

TABLE II.
SIMULATION PARAMETERS

	Simulation 1	Simulation 2	Simulation 3
α	10^{-3}	$3 \cdot 10^{-3}$	$3 \cdot 10^{-3}$
β	10^{-3}	10^{-3}	10^{-3}
γ	10^{-3}	10^{-3}	10^{-3}
δ	$5 \cdot 10^{-3}$	$5 \cdot 10^{-3}$	$5 \cdot 10^{-3}$
ε	10^{-3}	10^{-3}	10^{-3}
ϕ	10^{-5}	10^{-5}	10^{-5}
λ	$2 \cdot 10^{-2}$	$2 \cdot 10^{-2}$	$2 \cdot 10^{-2}$
μ	$9 \cdot 10^{-1}$	$9 \cdot 10^{-1}$	$9 \cdot 10^{-1}$
ρ	$2 \cdot 10^{-2}$	$2 \cdot 10^{-2}$	$4 \cdot 10^{-2}$
ξ	$9 \cdot 10^{-1}$	$9 \cdot 10^{-1}$	$9 \cdot 10^{-1}$
ν	10^{-9}	10^{-9}	10^{-9}
n_A	300	300	300
n_B	400	400	400
n_C	300	0	0
leech limit	2	2	2

dp = 2.97, while idp_A = 212.16, so tp_A = 215.13. In the simulation, dp = 3, while idp_A = 210, so tp_A = 213, which is close to the analytical estimate. We can calculate the upper bound of idp_B using (4), such as $idp_B \leq 133.33$, while the simulation result is in line with this value (idp_B = 71). Figure 3 shows how the size of the different user classes was changing during the spreading process. The duration of the process was 8470 hours, which is almost one year; however, we can see that the interest to purchase the application was very low after 4500 hours.

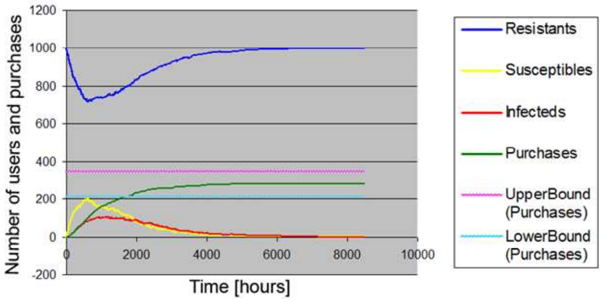

Figure 3. The size of user classes and the number of purchases – Simulation 1

In Simulation 2, we tripled the value of transition intensity α, which means that the trial version of the application will be spread faster, and we expect that the speed of the whole spreading process will increase. In addition, we set the number of Type$_C$ users to zero to demonstrate how Type$_C$ users influence idp_B. Since we set $\alpha >> \phi$, $\alpha / (\alpha + \phi)$ can be estimated by 1 and $\phi / (\alpha + \phi)$ by ϕ / α, so we expect that dp will drop to the third of the value we got in Simulation 1, while the idp_A will be nearly the same. The upper bound of idp_B is exactly the same as in Simulation 1; however, we expect that idp_B will decrease, because Type$_B$ users will find a seed more frequently than in Simulation 1 (there are no Type$_C$ users in this scenario). Figure 4 shows that the simulation terminated after 5330 hours, however, the interest to purchase the application decreased extremely after 1500 hours. The value of dp and idp_A are close to the expected values (dp = 1, idp_A = 214), while idp_B decreased to 62. The decreasing value of idp_B can be explained by the absence of Type$_C$ users.

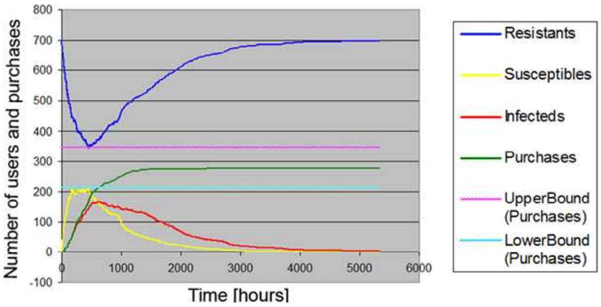

Figure 4. The size of user classes and the number of purchases – Simulation 2

In Simulation 3, we doubled the value of transition intensity ρ, from which we expect that Type$_B$ users will less frequently find a free seed to connect to, so idp_B will increase compared to Simulation 2. The results verified our expectations and idp_B increased to 81 (Figure 5).

We collected the results of these simulations in Table III. The expected values are in parentheses.

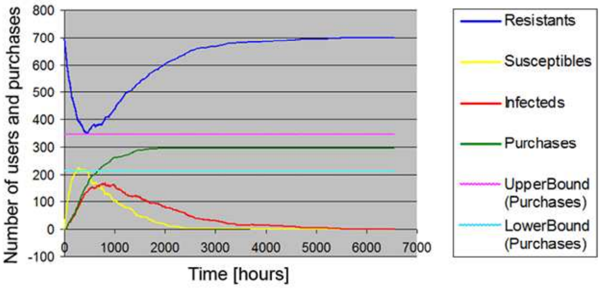

Figure 5. The size of user classes and the number of purchases – Simulation 3

TABLE III.
SIMULATION RESULTS

	Simulation 1	Simulation 2	Simulation 3
tp_A	213 (215.13)	215 (214.56)	214 (214.56)
dp	3 (2.97)	1 (0.99)	1 (0.99)
idp_A	210 (212.16)	214 (213.57)	213
idp_B	71 (≤ 133.33)	62 (≤ 133.33)	81 (≤ 133.33)
duration	8470 hours	5330 hours	6541 hours

B. Tendencies

In our further investigations, we show trend lines to understand how idp_B does change by changing those parameters, which do not influence the analytically computed upper bounds (such as leech limit, the number of Type$_C$ users and the transition intensities regarding to the usage of the application).

In the following simulations, we use the parameters of Simulation 1, except that we want to investigate. In these simulations, each result is the rounded average of 10 individual runs, and the figures show also the 95% confidence intervals.

First, we have investigated how idp_B does depend on the leech limit. Figure 6 shows the results of these simulations. We have investigated also the case when leech limit is zero, which means that a susceptible node cannot try out the application. Though, this situation is in contradiction with our model, we can see that the result (idp_B = 137) is very close to the analytical upper bound of idp_B, since this situation corresponds to the one in which a Type$_B$ user never finds an available seed which she can connect to. The value of idp_B does not linearly decrease with the increasing value of the leech limit. Even if we set the leech limit to 9, we get that 37 Type$_B$ users purchased the application. We can also recognize that idp_B does not decrease very much if the leech limit is over three (idp_B is 49 in case of the leech limit is 4).

In the next investigation, we changed the number of Type$_C$ users to see how Type$_C$ users do influence idp_B

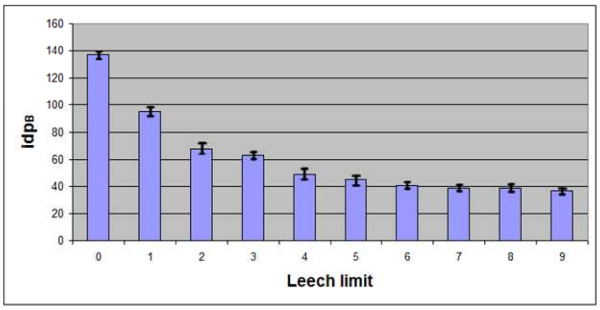

Figure 6. Variation of the idp_B value depending on the leech limit

(Figure 7). In Simulation 2, we have already investigated the situation when the number of Type$_C$ users is zero. Then idp_B was 62, which is higher than we got in this simulation (idp_B was 54). The only difference between the two simulations is that the single user transition intensity α was three times higher in Simulation 2, which made the spreading of the trial version faster. Therefore, the average number of susceptible users was greater, and the contention for an available seed was fiercer. Hence, there were more susceptible users, which could not find a seed to connect to. In the case, when the number of Type$_C$ users is 300, the situation is exactly the same as in Simulation 1. The results of the two simulations are very close to each other (idp_B = 71 and idp_B = 72). Figure 7 shows that idp_B increases with the increase of the number of Type$_C$ users. If the number of Type$_C$ users exceeds the number of Type$_B$ users, idp_B approximates the upper bound we computed analytically.

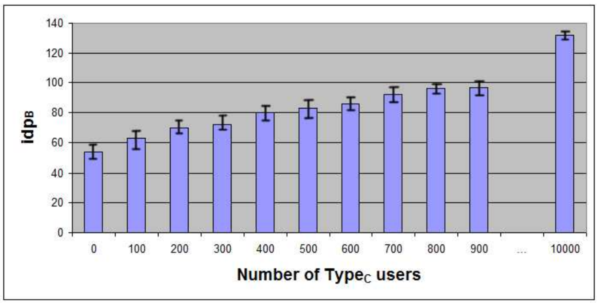

Figure 7. Variation of the idp_B value depending on the number of Type$_C$ users

Finally, we have investigated how the usage of the application influences the value of idp_B. There are four parameters (ρ, ξ, λ and μ), which describe how users do use the application. In fact, $r_1 = \rho / (\rho + \xi)$ shows what the ratio of the active users is among the susceptible users, and $r_2 = \lambda / (\lambda + \mu)$ determines what the ratio of the active users is among the infected users. Note that these values are valid when the CQN is in steady state. The system will converge to them and it can be interesting to see the tendencies. The ratio of r_1 and r_2 determines whether a susceptible user finds an available seed or not. If r_1 / r_2 is greater than the leech limit, a susceptible user which wants to start the application will not find any available seed. Therefore, we investigate how idp_B does change depending on the ratio of r_1 and r_2 (Figure 8). In Simulation 1, we investigated the case when r_1 / r_2 = 1. If r_1 / r_2 decreases, the value of idp_B also decreases, and if r_1 / r_2 increases, the value of idp_B increases, as well. The case when r_1 is much greater than r_2 means that susceptible users almost never find an available seed to connect to. Therefore, the upper bound of idp_B is approximated, when Type$_B$ users are always motivated to purchase the application.

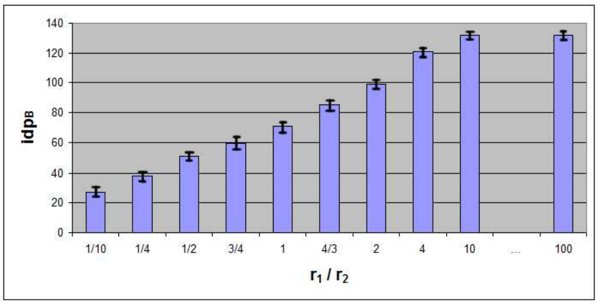

Figure 8. Variation of the idp_B value depending on r_1 / r_2

VII. Summary

In this paper, we proposed a model for application spreading aided by mobile ad hoc connections. We showed how our spreading model can be used. Furthermore, we gave analytical approximations on the lower and upper bounds of the number of application purchases, and validated our results via simulations. Finally, we presented some simulation results to demonstrate how the different parameters of our model do influence the spreading process.

In the future, we plan to elaborate on setting the model parameters as realistic as possible and investigate other tools, such as Stochastic Petri Nets, to be able to refine our spreading model.

Acknowledgment

This work has been partially supported by the Hungarian Scientific Research Fund (OTKA, PD 72984).

References

[1] T. G. Robertazzi, *Computer Networks and Systems: Queuing Theory and Performance Evaluation,* 1990 Springer-Verlag New York, Inc., New York, USA, 1994.

[2] B. Plattner and K. Farkas, “Supporting Real-Time Applications in Mobile Mesh Networks”, MeshNets 2005 Workshop, Budapest, Hungary, July 2005.

[3] S. Čapkun, L. Buttyán, and J.-P. Hubaux, “Self-Organized Public-Key Management for Mobile Ad Hoc Networks”, IEEE Transactions on Mobile Computing, Vol. 2 No. 1, January-March 2003.

[4] Y.-C. Hu, D. B. Johnson and A. Perrig, “Secure Efficient Distance Vector Routing in Mobile Wireless Ad Hoc Networks”, 4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA), Callicoon, New York, USA, June 2002.

[5] Y.-C. Hu, A. Perrig and D. B. Johnson, “Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks”, 8th ACM International Conference on Mobile Computing and Networking (MobiCom), Atlanta, Georgia, USA, September 2002.

[6] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields and E. M. Belding-Royer, “A Secure Routing Protocol for Ad Hoc Networks”, 10th IEEE International Conference on Network Protocols (ICNP), Paris, France, November 2002.

[7] Y. Wang, D. Chakrabarti, C. Wang and C. Faloutsos, “Epidemic Spreading in Real Networks: An Eigenvalue Viewpoint”, 22nd International Symposium on Reliable Distributed Systems (SRDS’03), 2003, Florence, Italy, pp.25-34.

[8] R. Pastor-Satorras and A. Vespignani, “Epidemic Spreading in Scale-Free Networks”, in *Phys. Rev. Lett.*, 86:3200, 2001.

[9] F. Fu, L. Liu and L. Wang, “Information Propagation in a Novel Hierarchical Network”, 46th IEEE Conference on Decision and Control, New Orleans, USA, 2007.

[10] A. Khelil, C. Becker, J. Tian and K. Rothermel, “An Epidemic Model for Information Diffusion in MANETs”, 5th ACM International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems, Atlanta, Georgia, USA, 2002.

[11] O. Sekkas et al., “Probabilistic Information Dissemination for MANETs: the IPAC Approach”, 20th Tyrrhenian Workshop on Digital Communications, Pula, Italy.

[12] E. Huang, W. Hu, J. Crowcroft and I. Wassel, “Towards Commercial Mobile Ad Hoc Network Application: A Radio Dispatch System”, 9th Annual International Conference on Mobile Computing and Networking, San Diego, USA, 2003.

[13] B. Cohen, “Incentives Build Robustness in BitTorrent”, 1st Workshop on Economics of Peer-to-Peer Systems, UC Berkeley, USA, 2003.