EXTENSION OF SIMPLE RECOMMENDATION MODEL

by

Sedat Çiftçi

B.S., Computer Engineering, Boğaziçi University, 2005

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Computer Engineering
Boğaziçi University
2008

EXTENSION OF SIMPLE RECOMMENDATION MODEL

APPROVED BY:

Asst. Prof. Haluk Bingöl (Thesis Supervisor)	....
Assoc. Prof. Yağmur Denizhan	....
Dr. Suzan Üsküdarlı	....

DATE OF APPROVAL: 29.08.2008

ACKNOWLEDGEMENTS

I would like to thank my advisor Asst. Prof. Haluk Bingöl, my committee members Assoc. Prof. Yağmur Denizhan and Dr. Suzan Üsküdarlı for their valuable contributions to my thesis.

I would also like to thank my colleagues, instructors and assistants of Computer Engineering Department of Boğaziçi University for their contributions to my academic knowledge.

Special thanks to my parents Hüseyin Çiftçi and Hadice Çiftçi, my sister Halime Çiftçi and my brother Murat Çiftçi for their continuous support in all areas of my life.

ABSTRACT

EXTENSION OF SIMPLE RECOMMENDATION MODEL

In Simple Recommendation Model (SRM), the effect of memory size on fame with respect to the population size is analyzed. Agents of SRM may learn new agents and may forget some of the agents that they know as a result of recommendations. During recommendations, the agent who will make recommendation, the agent to whom recommendation is made, the agent that will be recommended and the agent that will be forgotten are selected. These selections are random in SRM. The population size is constant in SRM.

In this thesis, SRM is extended by changing the selection method of agents. Also, dynamic population size is introduced. Each extension is simulated and the effect of each extension is analyzed by comparing the simulation results with the results of SRM.

ÖZET

BASİT TAVSİYE MODELİNİN GENİŞLETİLMESİ

Basit Tavsiye Modelinde (SRM), nüfus boyutuna göre hafıza boyutunun şöhret üzerindeki etkisi analiz edilmiştir. SRM'deki ajanlar tavsiyeler sonucunda yeni ajanlar öğrenir ve bildiği ajanları unutur. Tavsiye sırasında tavsiye yapacak ajan, tavsiyenin yapılacağı ajan, tavsiye edilecek ajan ve unutulacak ajan seçilir. SRM'de bu seçimler rasgele yapılır. SRM'de nüfus boyutu sabittir.

Bu tezde, ajanları seçme metodu değiştirilerek Basit Tavsiye Modeli genişletiliyor. Ayrca, dinamik nüfus boyutu tanıtılıyor. Herbir genişleme simule ediliyor ve herbir genişlemenin etkisi simülasyon sonuçlarının SRM'nin sonuçlarıyla karşılaştırılarak analiz ediliyor.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS/ABBREVIATIONS

n	Number of agents
m	Memory size of an agent
ρ	Memory ratio $= m/n$
a_i	The i^{th} agent
M_i	Memory of a_i
k_i	Knownness of $a_i = \mid \{a_j \mid a_i \in M_j\} \mid$
f_i	Fame of $a_i = k_i/n$
v	The average recommendation per agent
a_G	The giver agent
a_T	The taker agent
a_R	The recommended agent
a_F	The forgotten agent
SRM	Simple Recommendation Model

1. INTRODUCTION

We can recommend many things that we like such as a film, music, game, restaurant, etc. to the other people. If the person that we make recommendation to also likes the thing that we recommend, he/she can recommend this thing to the others, too. As a result of these recommendations, the fame of the thing that is recommended is increased in the population.

In Simple Recommendation Model (SRM) [1], this recommendation procedure is modeled by using agents that recommend only agents. As a result of recommendations, agents may learn new agents and may forget some of the agents that they know. In the recommendation procedure, the agent who will make recommendation, the agent to whom recommendation is made, the agent that will be recommended and the agent that will be forgotten are selected randomly. At the end of simulations, the effect of memory size on fame with respect to the constant population size is analyzed in SRM.

In SRM, it is stated the SRM can be used to build more complex models by using different selection methods. In my thesis, I extend the SRM by using different selection methods instead of random selection methods and analyze the effect of each new selection method on SRM. I also introduce dynamic population size concept in my thesis.

In the next chapters, SRM, extension of SRM, and extension of SRM to dynamic population are explained respectively. In the last chapter, a conclusion, that states the summary of the observations seen in the simulation results, is given.

2. SIMPLE RECOMMENDATION MODEL

Simple Recommendation Model (SRM)[1] is a model that is used to investigate the effect of memory size on fame with respect to the population size. Agents may learn new agents and may forget some of the agents that they know as a result of recommendations.

In SRM, there are n agents where each of them has the same memory size, m $(0 \leq m \leq n)$. The memory M_i of an agent a_i is a subset of the agents in the population. An agent a_i *knows* a_j if a_j is an element of M_i. The *knownness* k_i of an agent a_i is the number of agents that know a_i. If $k_i = 0$, then the agent a_i is called *completely forgotten* and if $k_i = n$, then the agent a_i is called *perfectly known* $(0 \leq k_i \leq n)$. The *fame* f_i of an agent a_i is k_i/n $(0 \leq f_i \leq 1)$. The *memory ratio* ρ is m/n $(0 \leq \rho \leq 1)$. Fame and memory ratio are used to make comparison between the simulations with different n and m values since they do not depend on n or m.

The system state is shown as an $n \times m$ matrix in Figure 2.1. Initially, an agent knows its m-neighbors. At each simulation cycle, a *giver* agent a_G selects the *recommended* agent a_R from its memory and recommends a_R to a *taker* agent a_T. If a_T already knows a_R, it does not do anything. Otherwise, a_T *learns* a_R by *forgetting* an agent a_F from its memory (learning an agent means getting it into the memory and forgetting an agent means removing it from the memory). The a_G, a_T, a_R and a_F are randomly selected by using a uniform distribution. The simulation ends when the average recommendation per agent, v, is reached. The algorithm of SRM is given in Appendix A.1.

Minimum fame in the population (f_{min}), maximum fame in the population (f_{max}), average fame of the top 5% of the agents in the population, where the agents are sorted according to their fame values in decreasing order ($f_{5\%}$), and percentage of forgotten agents in the population (u) vs. ρ graphs are investigated at the end of simulations for different combinations of n and ρ.

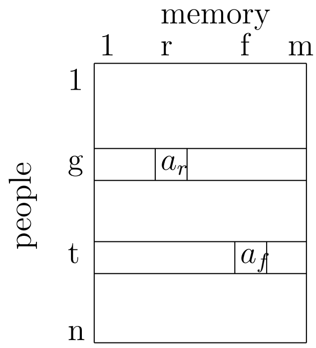


Figure 2.1. $n \times m$ matrix that represents the system state taken from [1]

In the simulations, a $\rho = 0.01 - 0.05$ range with 0.01 increments, $\rho = 0.10 - 0.90$ range with 0.05 increments and an $n = 100 - 1000$ range with 100 increments are used. For the value of v, 10^6 is used.

According to the simulation results of SRM given in Figure 2.2, some of the observations that are stated in [1] are listed below:

- f_{min} decreases linearly when ρ decreases (for $n = 1000$, $f_{min} \approx 1.1\rho - 0.12$)
- u increases linearly when ρ decreases (for $n = 1000$, $u \approx -9.2\rho + 1$) and f_{min} and u complement each other for the same values of n
- f_{max} increases linearly when ρ increases (for $n = 1000$, $f_{max} \approx 0.91\rho + 0.11$)
- $f_{5\%}$ is very similar to f_{max} (for $n = 1000$, $f_{5\%} \approx 0.95\rho + 0.071$)

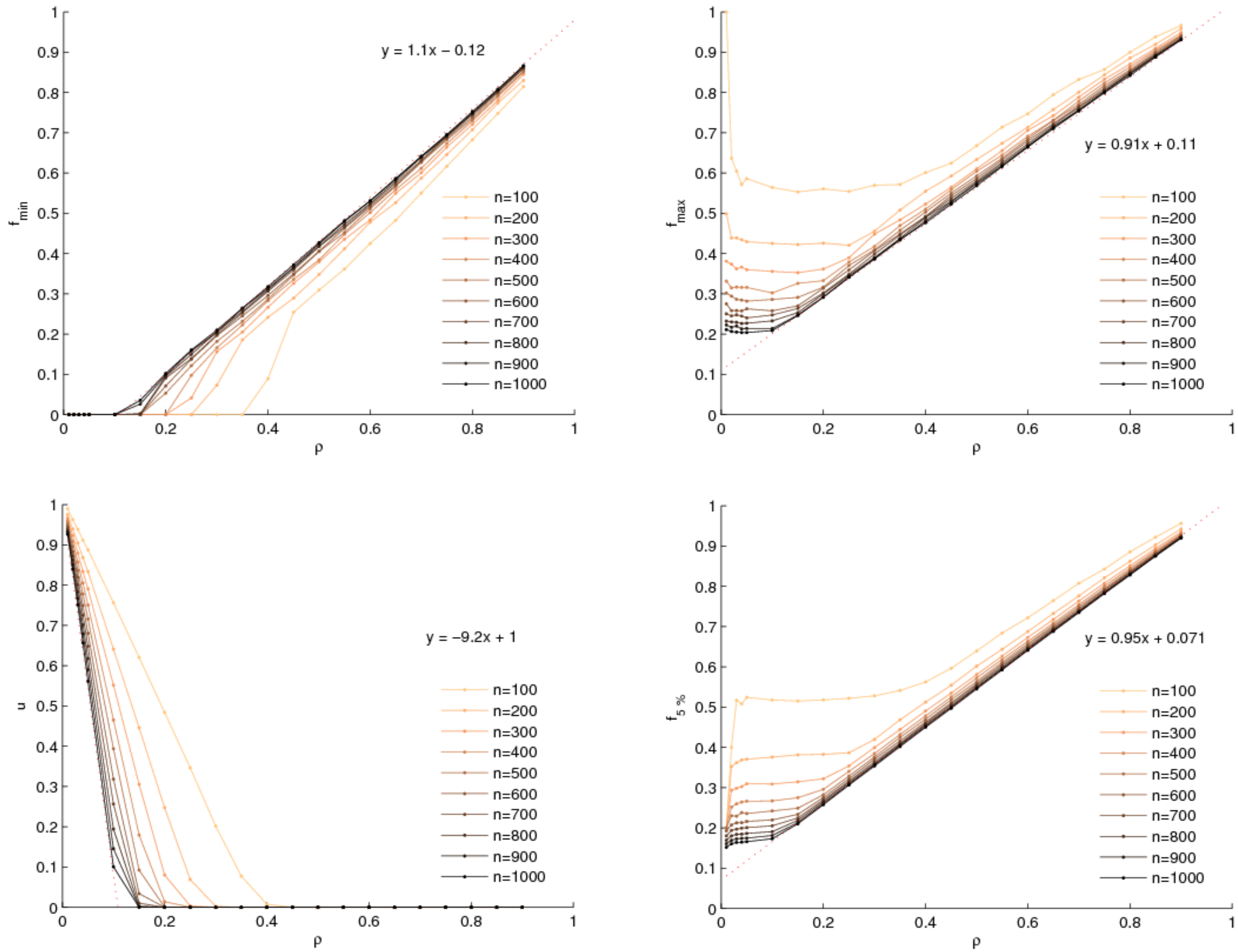


Figure 2.2. Simulation results of SRM taken from [1]

3. EXTENSION OF SRM

As it is stated in SRM, a_G, a_T, a_R and a_F are randomly selected by using a uniform distribution. We can simply extend the SRM by changing the selection method of a_G, a_T, a_R or a_F. In the following subsections, we change the selection method of each a_G, a_T, a_R and a_F one by one and compare the simulation results with the results of SRM. The new selection methods that we use are iterative selection of a_G, selection of a_T from friends, recommendation counter based selection of a_F, recommendation counter based selection of a_R, and advertisement based selection of a_R.

3.1. Iterative Selection of a_G

In one simulation cycle of SRM, an agent is randomly selected from the population as a giver agent a_G. We change the random selection of a_G with iterative selection of a_G.

In one simulation cycle of iterative selection of a_G, each agent is v_C times selected as a_G by iterating over the entire population so that we give equal chance to all of the agents in the population to be selected as a_G. The v_C can be defined as the average recommendation per agent per simulation cycle. The reason of defining new variable v_C is its need in recommendation counter based selection of a_F and extension of SRM to dynamic population. The algorithm is given in Appendix A.2.

In the simulations, v_C is taken as 10. The used n values are in range $100 - 900$ with 200 increments, ρ and v values are the same as the ones in SRM. The result of the simulations with iterative selection of a_G is given in Figure 3.1. The shown results are calculated by taking the average of 10 runs. The lines given in SRM are also shown in the figure for comparison. Simulation results of all runs, where $n = 900$, is given in Appendix B.1.

If the simulation results given in Figure 3.1 are compared with the results of the SRM, it is seen that almost all of the results are the same. Therefore, we can say

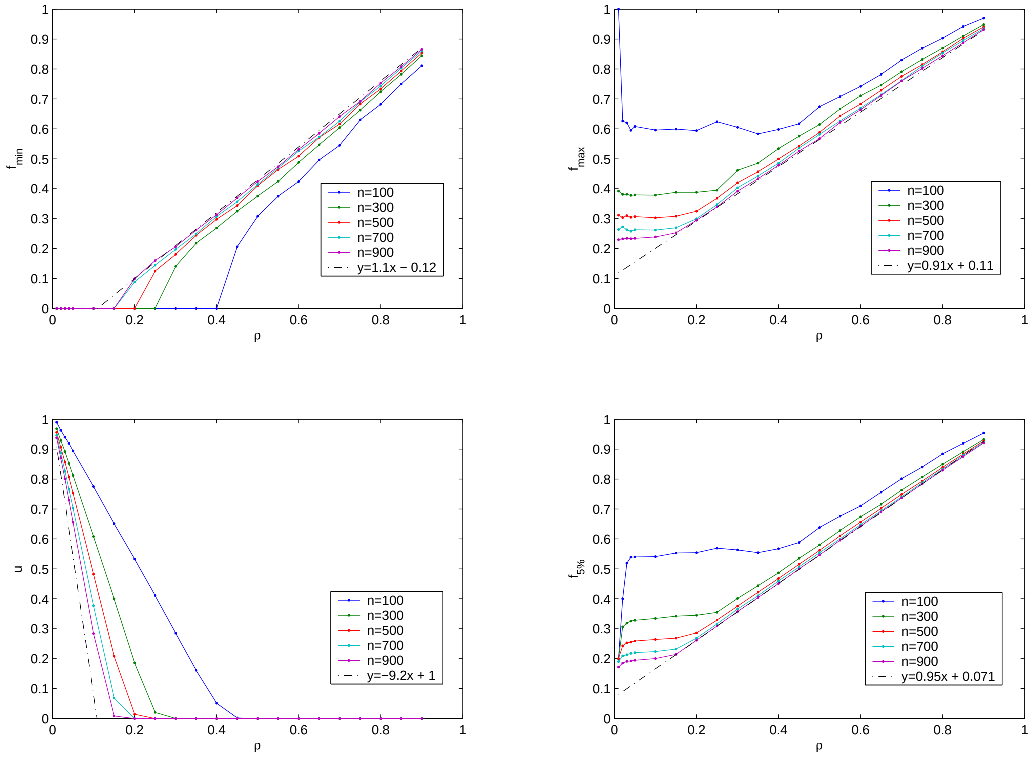


Figure 3.1. Simulation results of iterative selection of a_G

that whether the selection of a_G is random or iterative does not affect the simulation results. The reason behind this observation is due to the very big value of v. When the value of v is very big, each agent will be more likely to be selected as a_G nearly equal since a_G is randomly selected by using a uniform distribution and as a result whether the selection of a_G is random or iterative does not make any difference. On the other hand, if the value of v were very small (such as 10^2) in the simulations of both SRM and SRM with iterative selection of a_G, the results could be much different.

After this point, we will use iterative selection of a_G in our rest of the simulations for the selection of a_G unless the opposite is stated.

3.2. Selection of a_T From Friends

In SRM, giver agent a_G recommends to taker agent a_T. The taker agent is randomly selected from the population. That is, any agent except a_G can be the taker independent of the giver which is an oversimplification as stated in [1]. However, in most of the models, the agents interact with a limited number of agents such as in Bak's sandpile model [2], Axelrod's two-dimensional culture model [3], or Conway's game of life [4].

A more realistic approach could be selecting a_T from the "friends" of a_G. Define the friendship as a graph where agents are at the vertices and there is an edge between two friends. Note that the graph is undirected so that the friendship relation is symmetric which is not the case in real life. We take this graph as a communication graph so that a_T is selected from $N_k(a_G)$ randomly, where $N_k(a_G)$ is defined as the set of vertices at most k distance away from a_G.

In this interpretation, the communication graph used in SRM is a complete graph and a_T is selected from the $N_1(a_G)$. To get an answer for the question "If the underlying communication graph changes, how the simulation results will be changed?", we use different communication graphs, make simulations and compare the simulation results with the results of SRM.

The communication graphs that we used are Erdös and Rényi Random Graph, Small World Network and Graph Constructed By Barabási-Albert Model. In the next subsections, the detail of each graph and the result of each one is given.

3.2.1. Erdös and Rényi Random Graph

Erdös and Rényi proposed two models to generate random graphs, one of them in 1959 [5] and the other one in 1960 [6]. The generated graphs are also called as Erdös and Rényi Random Graph. Those proposed models to generate random graphs are:

1. $G(n, K)$: Initially there are n vertices that are disconnected. The ER Random Graph is constructed by connecting randomly selected two nodes until the number of edges becomes K [5].
2. $G(n, p)$: Initially there are n vertices that are disconnected. The ER Random Graph is constructed by adding each possible edge to the graph with probability

p [6].

In our communication graph based on ER Random Graph, we use the second model to generate our random graph. In the simulations, p is taken as 0.1. The used n values are in range $100 - 900$ with 200 increments, ρ and v values are the same as the ones in SRM. The result of the simulations, where the communication graph is based on ER Random Graph, is given in Figure 3.2(a). The shown results are calculated by taking the average of 10 runs. The lines given in SRM are also shown in the figure for comparison. Simulation results of all runs, where $n = 900$, is given in Appendix B.2.1.

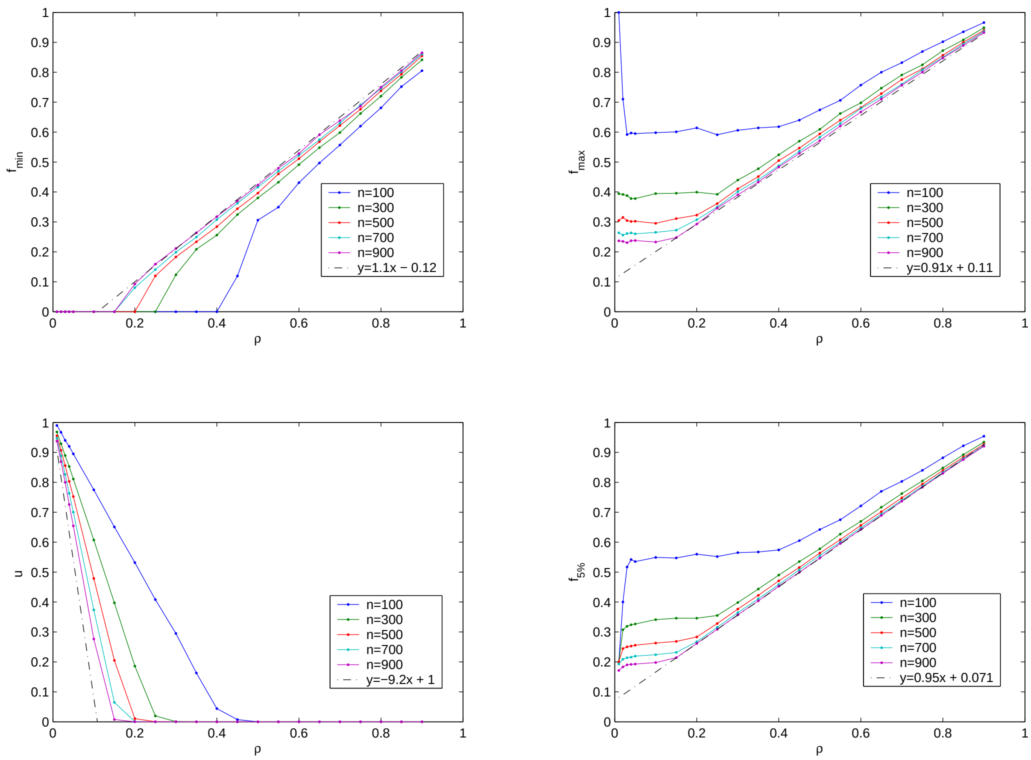
If we compare the simulation results given in Figure 3.2(a) with the results of the SRM, it is seen that almost all of the results are the same. That is, if the communication graph is based on ER Random Graph, it nearly does not affect the simulation results. This is a very interesting observation. Although we limit the number of agents, that a_G can make recommendation to, to the number of friends a_G has, the simulation results are nearly not affected.

We also make simulations with different p values to see whether the results that we got are nearly the same as the results of the SRM due to the value of p that we used, or not. The simulation results given in Figure 3.2(b) are done with $p = 0.1, 0.2, 0.3, 0.4, 0.5$ for $n = 100$. The shown results are calculated by taking the average of 10 runs. If we look at those results, it can be seen that p does not affect the simulation results. As a result, if we change the underlying communication graph of SRM from complete graph to ER Random Graph, the simulation results are almost not affected.

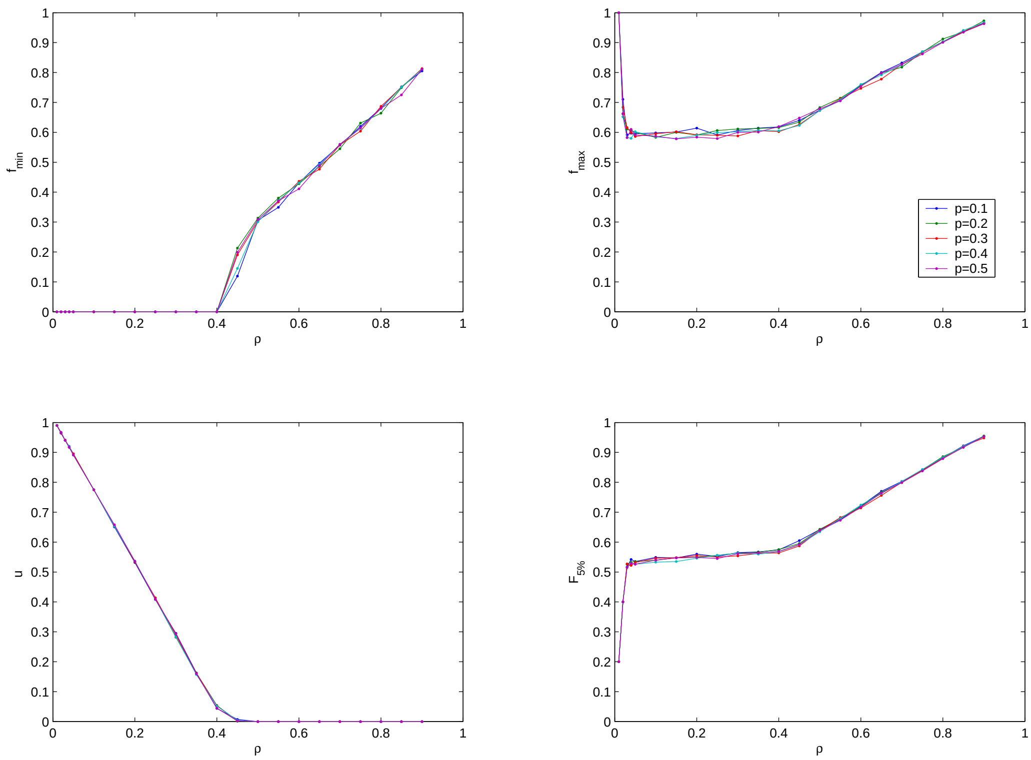
3.2.2. Small World Network

The term "Small World" network is introduced by Watts and Strogatz [7]. The reason for why they use the term "Small World" is the similarity between their model and the small-world phenomenon [8], [9].

A small world network begins with a ring lattice of n vertices. Each vertex

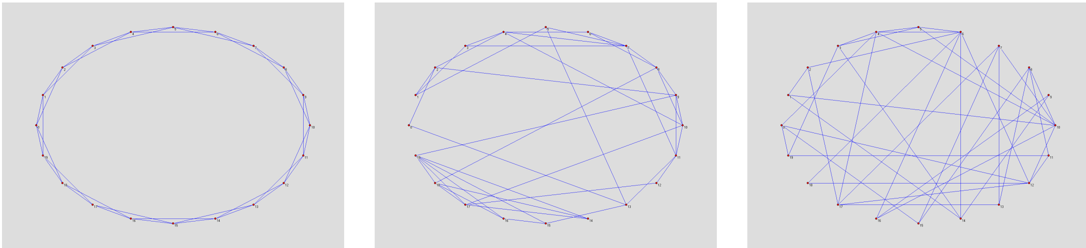


(a)



(b)

Figure 3.2. Simulation results of selection of a_T from friends where the underlying communication graph is an ER random graph, $G(n, p)$. (a) Various n values for $p = 0.1$, and (b) various p values for $n = 100$.



(a) (b) (c)

Figure 3.3. Sample created networks for $n = 20$ and $d = 4$. (a) $p = 0$ (regular), (b) $p = 0.5$ (small world), and (c) $p = 1$ (random).

has a connection with its d-nearest neighbors, that is each vertex has d edges. Then, rewiring procedure takes place. In rewiring procedure, a vertex and the edge that connects to its nearest neighbor in the clockwise direction is chosen. According to a probability p, this edge is

- either removed and a new edge is introduced between this vertex and a randomly chosen vertex among all of the vertices in the ring by avoiding duplicate edges
- or this edge is preserved.

This process is done for each vertex in the ring by iterating vertices in the clockwise direction around the ring. One lap is completed when all vertices are iterated. In the second lap, the edges with the second-nearest neighbors are rewired as in the first lap. After $d/2$ laps, the rewiring procedure ends.

The resultant network depends on the selected p. If $p = 0$, then the ring lattice do not change. If $p = 1$, then each edge is rewired randomly. For the other intermediate values of p, the graph is called small world network. In Figure 3.3, sample created networks for $n = 20$ and $d = 4$ with different p values are shown. In [7], it is stated that $n \gg d \gg ln(n) \gg 1$ is needed and $d \gg ln(n)$ guarantees the connectivity of a random graph [10].

In our communication graph based on small world network, each agent is a vertex in small world network. The initial ring is created by connecting a_i to $a_{(i+j) \bmod n}$ and

$a_{(i-j) \bmod n}$ for $j = \{1, ..., k/2\}$. Then the rewiring procedure is applied.

In the simulations, p is taken as 0.1 and d is taken as 10. The used n values are in range $100 - 900$ with 200 increments, ρ and v values are the same as the ones in SRM. The result of the simulations, where the communication graph is based on a small world network, is given in Figure 3.4(a). The shown results are calculated by taking the average of 10 runs. The lines given in SRM are also shown in the figure for comparison. Simulation results of all runs, where $n = 900$, is given in Appendix B.2.2.

If we compare the simulation results given in Figure 3.4(a) with the results of the SRM, it is seen that almost all of the results are the same. That is, if the communication graph is based on a small world network, it nearly does not affect the simulation results.

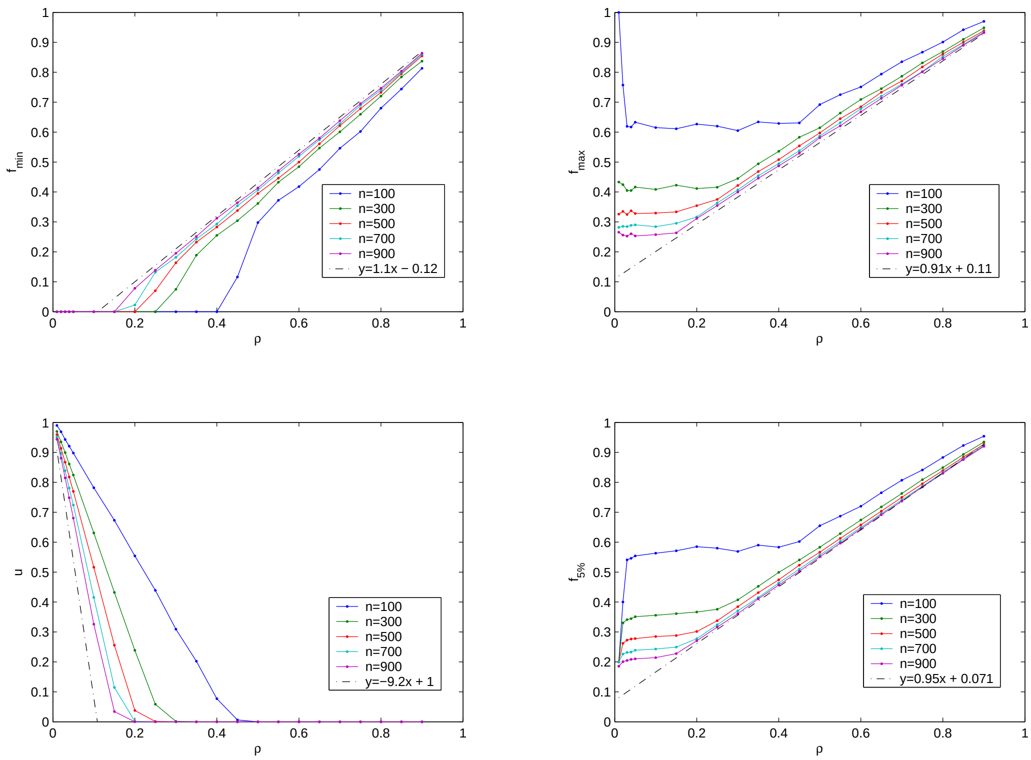
We also make simulations with different p values to see the effect of p. The simulation results given in Figure 3.4(b) are done with $p = 0.1, 0.2, 0.4, 0.6, 0.8$ for $n = 100$. The shown results are calculated by taking the average of 10 runs. If we look at those results, it can be seen that p does not affect the simulation results.

3.2.3. Graph Constructed By Barabási-Albert Model

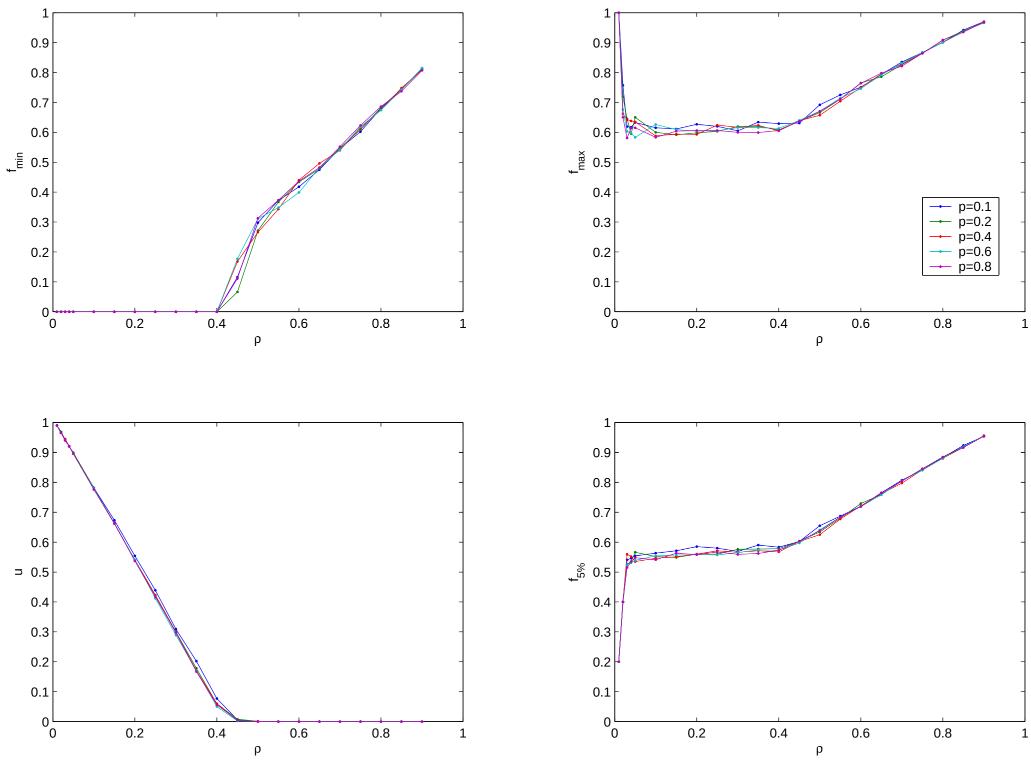
In this case, the communication graph is based on a graph constructed by Barabási-Albert (BA) Model [11]. In the BA model, a connected graph that contains d_0 vertices is used as a base graph. At each step, a new vertex is added to the graph by connecting it to the d ($\leq d_0$) different vertices among the existing vertices. The selection of a vertex for connecting to the newly added vertex depends on the degree of the existing vertices, also known as preferential attachment. The probability p_i that a new vertex is connected to vertex i is calculated as

$$p_i = \frac{degree_i}{\sum_j degree_j}$$

where the summation of degrees is done over all existing vertices.



(a)



(b)

Figure 3.4. Simulation results of selection of a_T from friends where the underlying communication graph is a small world network with $d = 10$. (a) Various n values for $p = 0.1$, and (b) various p values for $n = 100$.

In our simulations, the initial network is a complete graph constructed by the first $(d + 1)$ agents added to the communication graph. The remaining agents are added to the communication graph one by one by using the BA model.

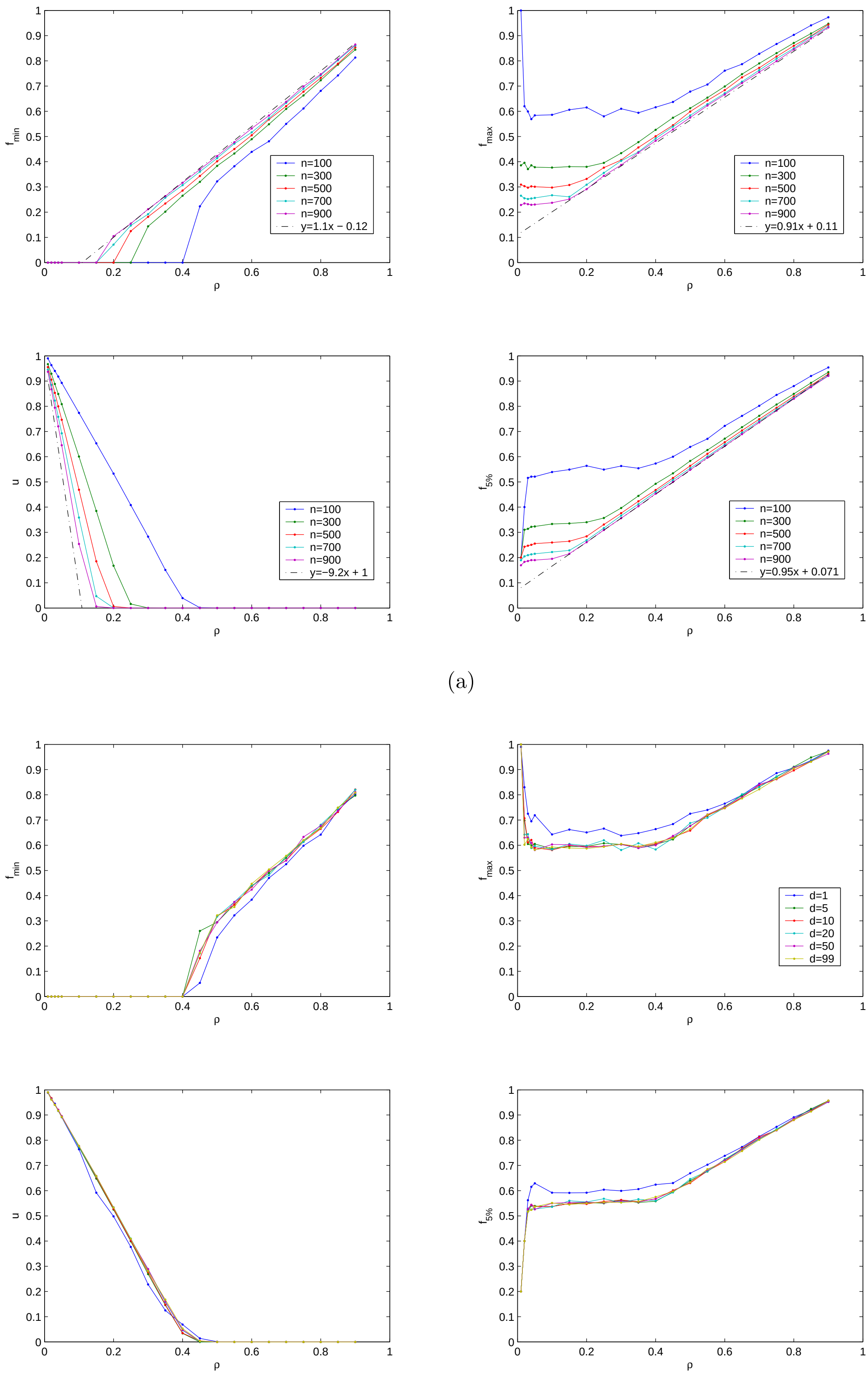
In the simulations, d is taken as 10. The used n values are in range $100 - 900$ with 200 increments, ρ and v values are the same as the ones in SRM. The result of the simulations, where the communication graph is based on a graph constructed by Barabási-Albert (BA) Model, is given in Figure 3.5(a). The shown results are calculated by taking the average of 10 runs. The lines given in SRM are also shown in the figure for comparison. Simulation results of all runs, where $n = 900$, is given in Appendix B.2.3.

If we compare the simulation results given in Figure 3.5(a) with the results of the SRM, it is seen that almost all of the results are the same. That is, if the communication graph is based on a graph constructed by Barabási-Albert (BA) Model, it nearly does not affect the simulation results.

After seeing that the simulation results are nearly the same as the results of SRM, we suspect that this result may be due to the value of d we select. To analyze the effect of d, we run the simulations with different d values. The used d values are 1, 5, 10, 20, 50 and 99. The simulation results shown in the Figure 3.5(b) are calculated as the average of 10 runs of each d value for $n = 100$. Again, the simulation results are also nearly the same as the result of SRM for different values of d.

3.2.4. Analysis of Existence of a Relation Between Degree of an Agent And Its Effects on Its Neighbors

In the real world, the number of friends a person has can be different from person to person. Some of the people have lots of friends whereas the others have less friends. Can we say which of those people may have more effect on their friends, the ones that have lots of friends or the ones that have less friends? According to us, the ones that have lots of friends may have more effect on their friends than the ones that have less



(a)

(b)

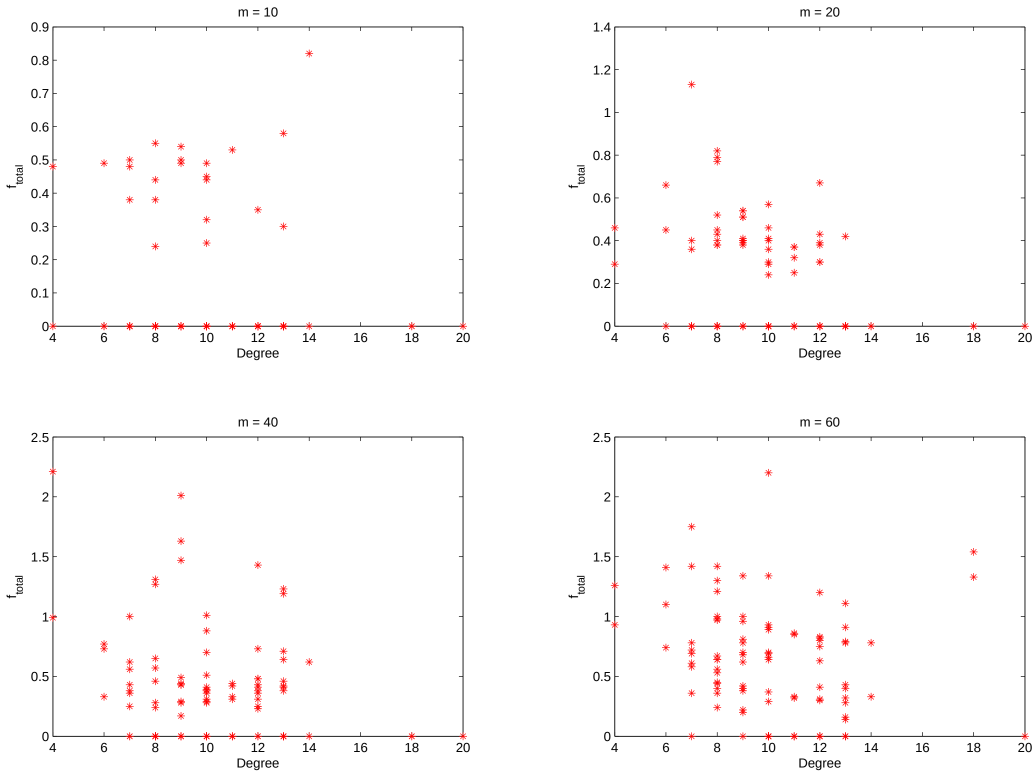
Figure 3.5. Simulation results of selection of a_T from friends where the underlying communication graph is constructed by BA model. (a) Various n values for $d = 10$, and (b) various d values for $n = 100$.

friends. We analyze the simulation results, where the underlying communication graph used are ER random graph and the graph constructed by BA Model, to see if there exists a relation between the degree of an agent and its effects on its neighbors.

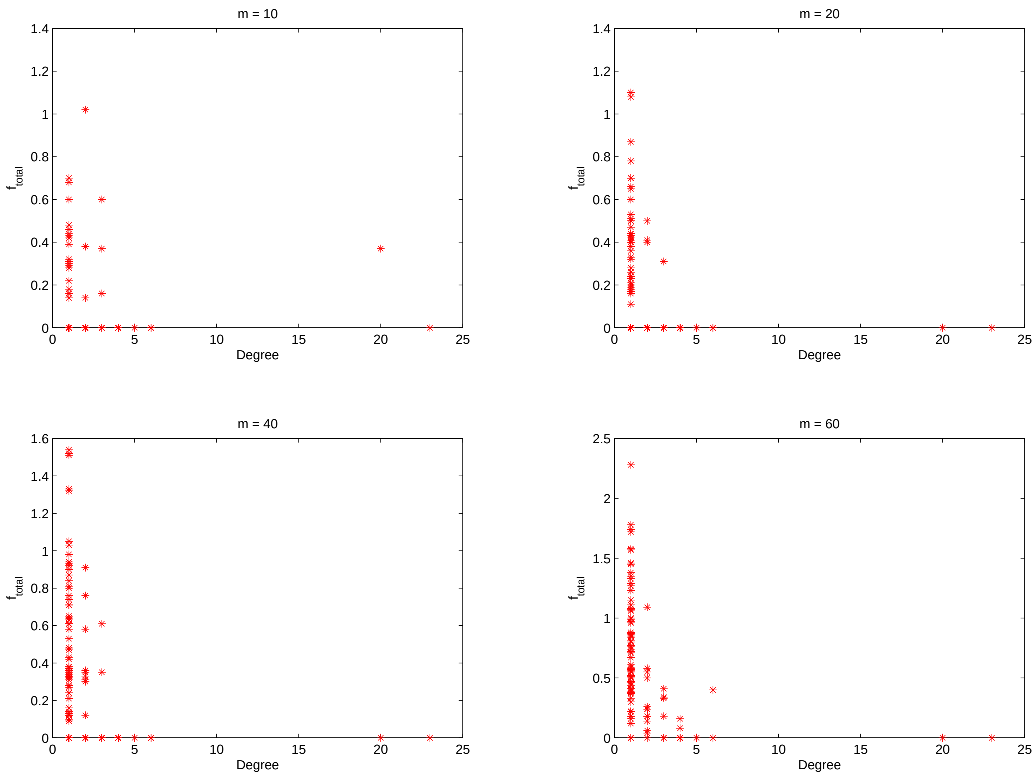
In SRM, agents recommend agents. That is *recommending items* are agents. In order to analyze the existence of a relation between the degree of an agent and its effects on its neighbors, we use different recommending items other than agents in our simulations. We use $n \times m$ unique recommending items. Initially, we share those recommending items equally among the agents so that each one knows m of the recommending items and none of any two agents know the same recommending item. Each agent is said to be the owner of the recommending items that it knows at the initial step. After the initial step, the recommendations take place until the end of simulation as it is in SRM. At the end of a simulation, we can analyze the effect of an agent on the population by looking at the fame of the recommending items owned by this agent. In the simulations, the generated communication graph is used in all runs and in all different m values to get meaningful simulation results.

The first thing that we look at is the total fame of the recommending items owned by an agent, f_{total}, vs. the degree of the owner agent. At the end of a simulation, we prepare an $n \times n$ item matrix, IM, where $IM_{i,j}$ represents the number of items in the M_j that are owned by a_i. That is, $IM_{i,j}$ denotes the number of items owned by a_i and learned by a_j at the end of the simulation. The total fame of the recommending items owned by a_i, f_{total_i}, is $\frac{1}{n}\sum_{j=1}^{n} IM_{i,j}$.

In Figure 3.6(a) and Figure 3.6(b), the results for f_{total} vs. the degree of the owner agent are shown. The communication graph used in first one is ER random graph with $p = 0.1$ whereas in the second one the used communication graph is constructed by BA Model with $d = 1$. There are 4 different m values used in the simulations: 10, 20, 30 and 40. The results are calculated by taking the average of 10 runs. In the simulations, we take $v = 10^5$ and $n = 100$. If we look at those results, there are some overlapping points since there can be more than one same f_{total} value for the same degree. Because of those points, any comment that we can make will be error prone.



(a)



(b)

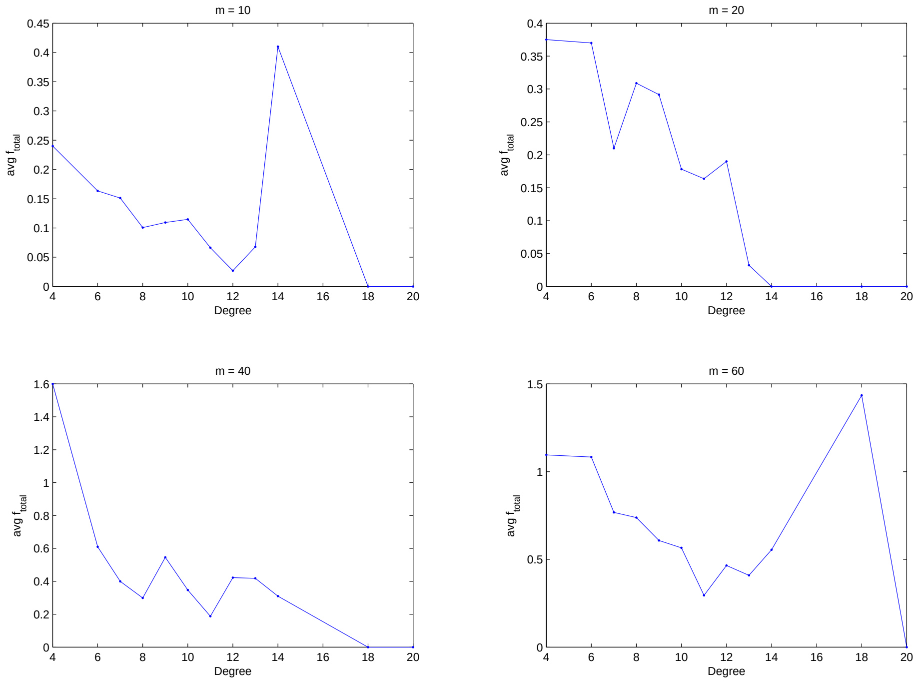
Figure 3.6. Simulation results for f_{total}. (a) The underlying communication graph is an ER random graph $G(n, p)$ where $p = 0.1$ is used, and (b) the underlying communication graph is constructed by BA model where $d = 1$.

The second thing that we look is the $avgf_{total}$ vs. the degree of the owner agents, where $avgf_{total}$ represents the average value of the f_{total} for the agents with the same degree. We calculate the $avgf_{total}$ value for any existing d by simply adding the f_{total} values of agents whose degree are d and divide the result of this addition by the number of agents whose degree are d. We get the results shown in Figure 3.7(a) and Figure 3.7(b) by using the simulation results shown in Figure 3.6(a) and Figure 3.6(b) respectively. However, we can not observe any pattern that applies for the simulation results of any used communication graph with different m values.

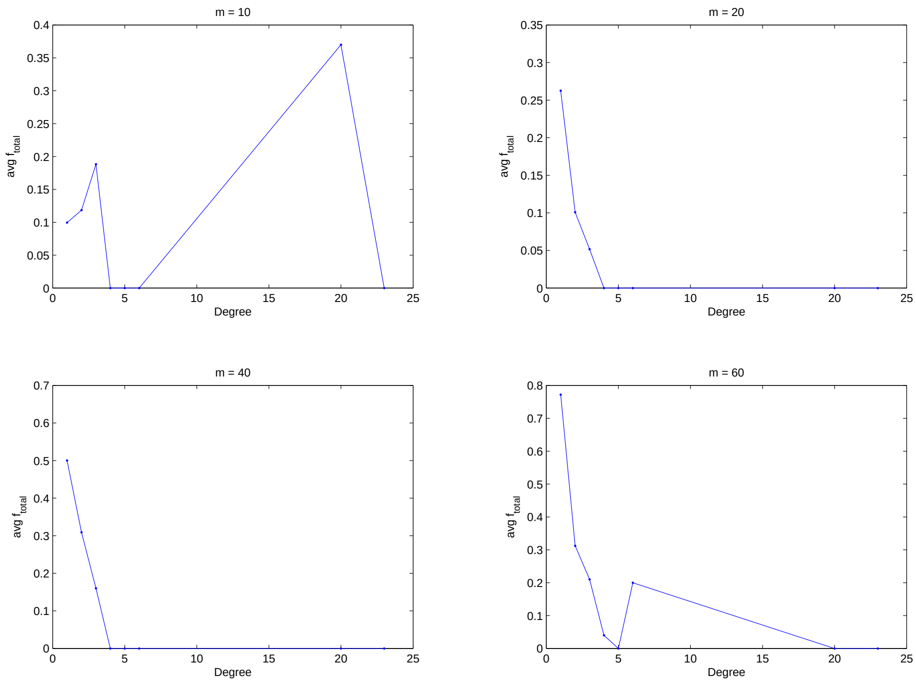
After that, we look at the effect of an agent on its neighbors which are k-distance away from it. Let Adj be the $n \times n$ adjacency matrix representation of the communication graph used. Lets define Adj_k to be the matrix representation of k-distance away neighbors of agents. Adj_k can be calculated as Adj^k, that is the matrix product of k number of Adj matrix. The matrix IM_k, that represents the effect of agents on k-distance away neighbors, can be calculated as $Adj_k \& IM$ where & means logically and two matrices element by element. The value of IM_k will remain constant after a value of k which satisfies that all of the elements of Adj_k is none zero. The total fame of the recommending items owned by a_i over the k-distance away neighbors of a_i, $f_{k\_total_i}$, is $\frac{1}{n}\sum_{j=1}^{n} IM_{k_{i,j}}$.

We calculate $avgf_{k\_total}$ by taking the average value of the f_{k_total} for the agents with the same degree. In Figure 3.8(a) and Figure 3.8(b), the results for the $avgf_{k\_total}$ vs. the degree of the owner agents are shown. We get the results show in Figure 3.8(a) and Figure 3.8(b) by using the simulation results shown in Figure 3.6(a) and Figure 3.6(b) respectively. There are 5 different k values used in the simulations: 1, 2, 3, 4, and 5. It can be see that the value of $avgf_{k\_total}$ is always increasing as k is increasing until the value of k which satisfies that all of the elements of Adj_k is none zero. Except this result, any common pattern that applies for the simulation results of any used communication graph with different m values can not be observed.

Note that there can be more than one path from one vertex to another in a graph with different distances. For example, an agent can be 1-distance away neighbor and

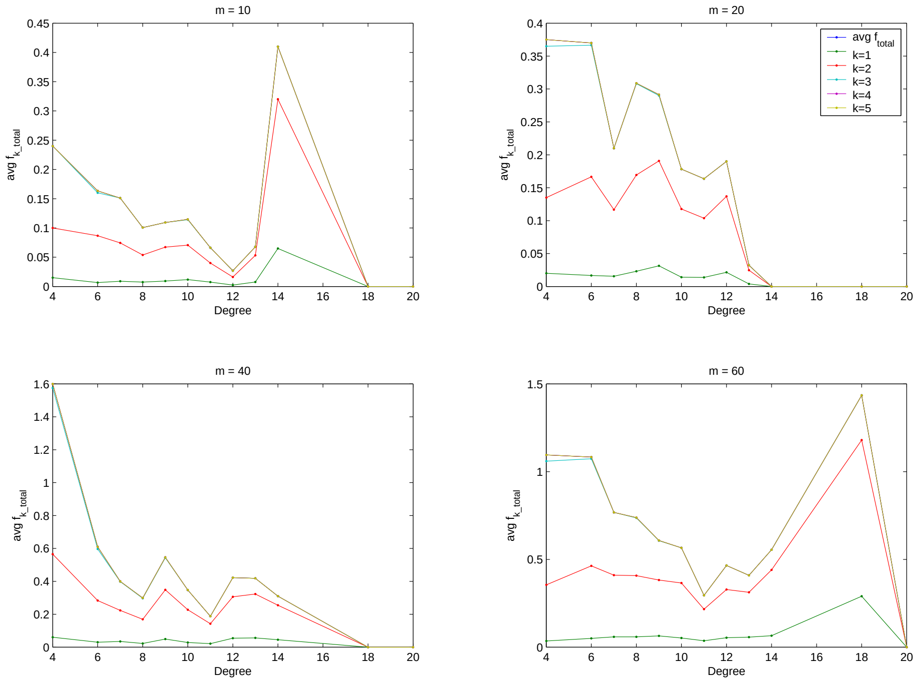


(a)

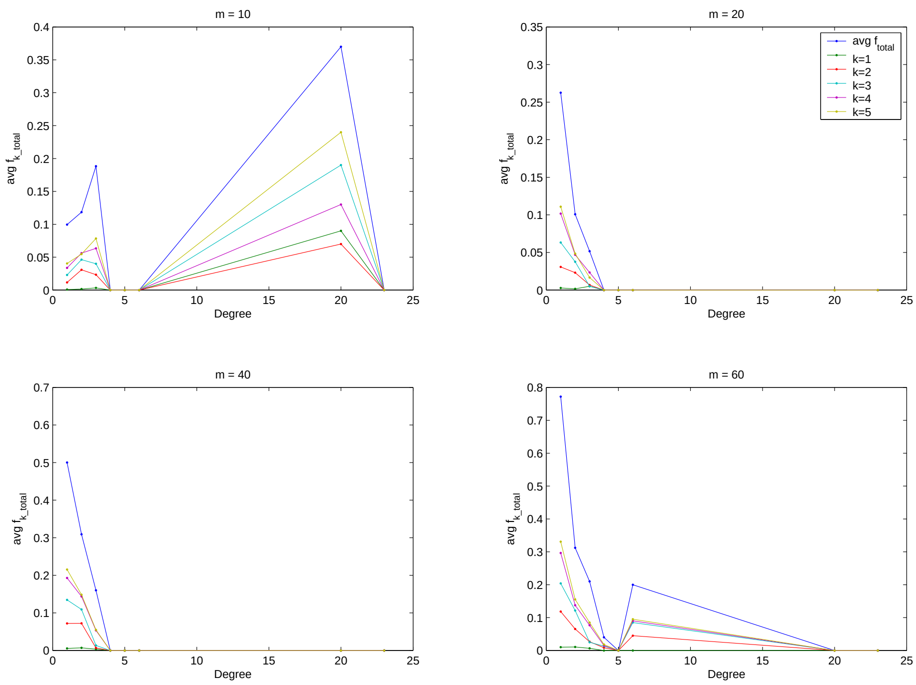


(b)

Figure 3.7. Simulation results for $avgf_{total}$. (a) The underlying communication graph is an ER random graph $G(n, p)$ where $p = 0.1$ is used, and (b) the underlying communication graph is constructed by BA model where $d = 1$.



(a)



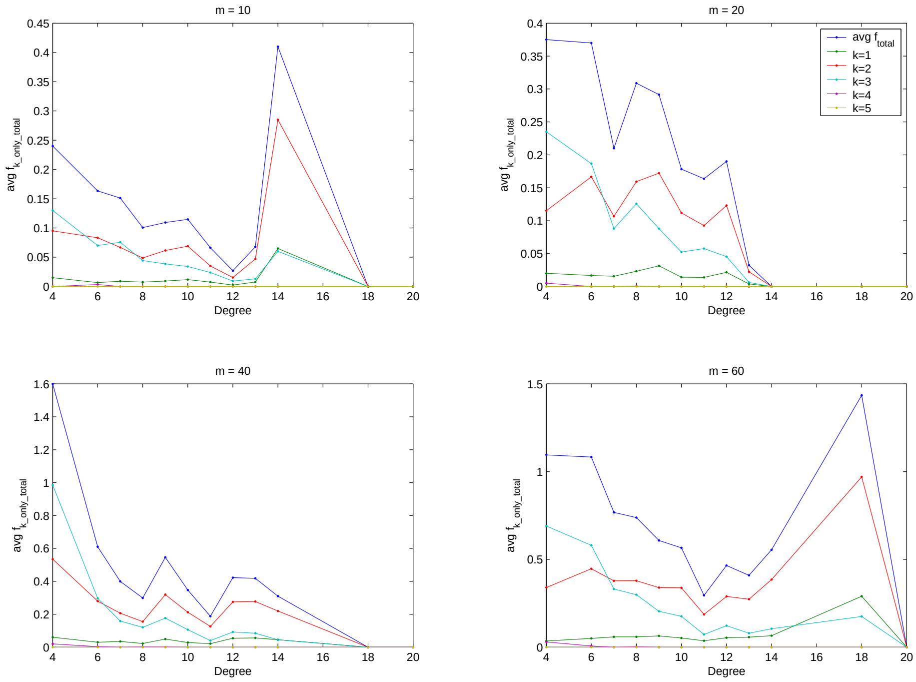
(b)

Figure 3.8. Simulation results for $avgf_{k\_total}$. (a) The underlying communication graph is an ER random graph $G(n, p)$ where $p = 0.1$ is used, and (b) the underlying communication graph is constructed by BA model where $d = 1$.

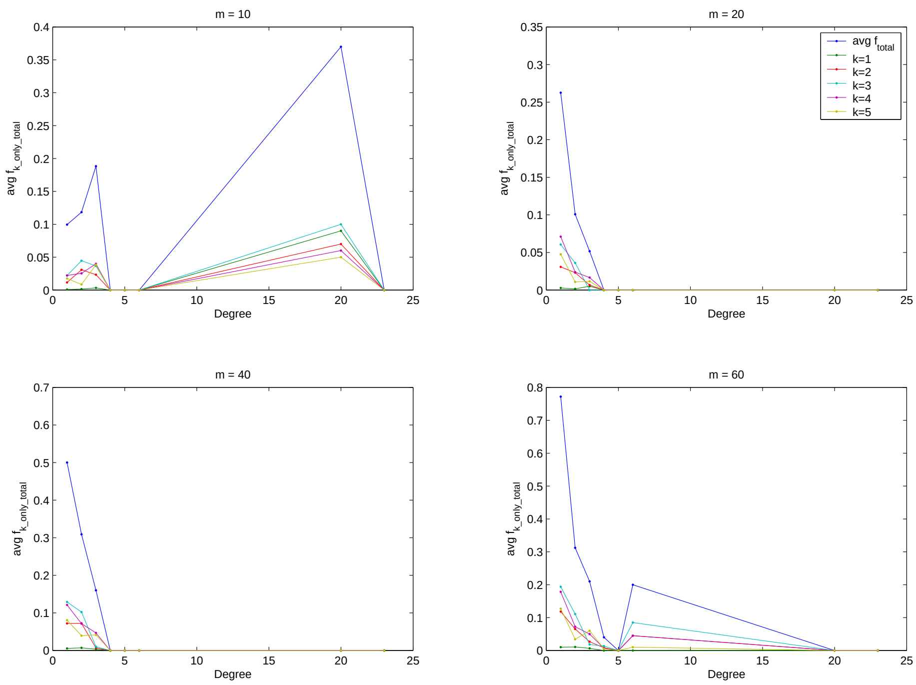
3-distance away neighbor of an agent at the same time. Because of this reason, the value of $avgf_{k\_total}$ is always increasing as k increases. We have to remove those duplications from Adj_k so that we calculate the effect of an agent on its neighbor only once. We call the resulting matrix Adj_{k_only} and it can be calculated as $Adj_k \& \bigwedge_{i=1}^{k-1} \neg Adj_i$ where & means logically and two matrices element by element, $\bigwedge$ means logically and all matrices, and $\neg$ means take logical not of each element of the matrix. The matrix IM_{k_only}, that represents the effect of agents on k-distance away neighbors where the duplications are removed, can be calculated as $Adj_{k\_only} \& IM$ where & means logically and two matrices element by element. The total fame of the recommending items owned by a_i over the k-distance away neighbors of a_i where the duplications are removed, $f_{k\_only\_total_i}$, is $\frac{1}{n}\sum_{j=1}^{n} IM_{k\_only_{i,j}}$.

We calculate $avgf_{k\_only\_total}$ by taking the average value of the $f_{k\_only\_total}$ for the agents with the same degree. In Figure 3.9(a) and Figure 3.9(b), the results for the $avgf_{k\_only\_total}$ vs. the degree of the owner agents are shown. We get the results shown in figure 3.9(a) and 3.9(b) by using the simulation results shown in 3.6(a) and 3.6(b) respectively. There are 5 different k values used in the simulations: 1, 2, 3, 4, and 5. However, again we can not observe any pattern that applies for the simulation results of any used communication graph with different m values.

After looking at all of those results, we can not find any relation between the degree of an agent and its effects on its neighbors. At the beginning of this section, we said that the people that have lots of friends may have more effect on their friends than the ones that have less friends. If we thing about our model, the agents that have higher degrees can be more affected by their friends since we give equal chance to all of the agents to be selected as a_G. If a degree based selection of a_G, where we give more chance to agents that have higher degrees, is used instead of iterative selection of a_G, then simulation results may support our statement. Another possible change that may support our statement can be recommending a_R to all of the friends of a_G, in other words broadcasting the a_R, instead of recommending to only one friend.



(a)



(b)

Figure 3.9. Simulation results for $avgf_{k\_only\_total}$. (a) The underlying communication graph is an ER random graph $G(n, p)$ where $p = 0.1$ is used, and (b) the underlying communication graph is constructed by BA model where $d = 1$.

3.3. Recommendation Counter Based Selection of a_F

Consider two agents a_1 and a_2. They are recommended to agent a_3 and learned by a_3. Suppose a_3 keeps getting recommendations of a_1 but no further recommendation of a_2. In SRM, this situation is ignored. That is, when the time comes to forget, a_1 and a_2 have equal chances. However, if we think about some real life examples, such as advertisements that people see on television, internet, etc., the ones which are more seen will be more probably not forgotten and the ones which are less seen will be more probably forgotten.

We improved forgetting mechanism as follows:

1. A recommended agent, that is kept in memory, is not forgotten for a period, $t_{keep\_duration}$. Only after that period it becomes a candidate to forget.
2. If an agent gets further recommendations, it becomes harder to forget it.

A function of recommendation counter, learned time of an agent and the duration agent is kept in memory is used to implement improved forgetting mechanism.

When an agent a_R is learned by an agent a_T, the $recommendation\_counter_{rt}$ is set to 1 and $learned\_time_{rt}$ is set to current simulation time. After a_R is learned by a_T, for each further recommendation of a_R to a_T, the $recommendation\_counter_{rt}$ is incremented by one.

When agent a_T has to forget an agent (that is an agent that a_T do not know is recommended to a_T), the candidate agents for a_F are the agents a_i with $(current\_time - learned\_time_{it}) \geq t_{keep\_duration}$, where $a_i \in M_T$. If there is not any candidate agent, then a_T does not learn the recommended agent. Otherwise, among the candidate agents the one with the smallest $recommendation\_counter_{it}$ value is selected as a_F. If there are more than one candidate agents with the same smallest recommendation counter value, then among those agents the one with the smallest $learned\_time_{it}$ value is selected as a_F. The reason of selecting the agent with the smallest $learned\_time_{it}$ value

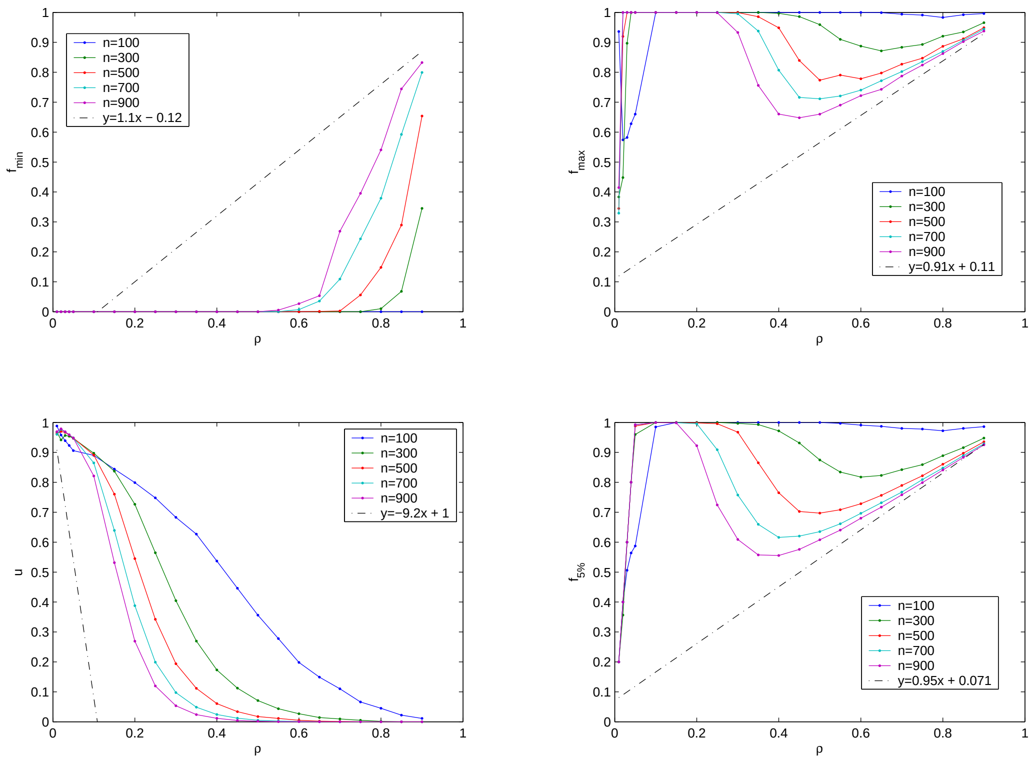


Figure 3.10. Simulation results of recommendation counter based selection of a_F

among the candidate agents with the same smallest recommendation counter value is the fact that although this agent is learned by a_T before than the others, the number of recommendations of this agent to a_T is the same as the number of recommendations of other agents to a_T. The algorithm is given in Appendix A.3.

In the simulations, one simulation cycle is the unit of time and the $t_{keep\_duration}$ is taken as 1 simulation cycle. The used n values are in range $100 - 900$ with 200 increments, ρ and v values are the same as the ones in SRM. The result of the simulations, where the improved forgetting mechanism is used, is given in Figure 3.10. The shown results are calculated by taking the average of 10 runs. Simulation results of all runs, where $n = 900$, is given in Appendix B.3.

The simulation results are different than the results of SRM. The f_{min} values are lower whereas the f_{max}, $f_{5\%}$ and u values are higher than the ones in SRM. That is, "rich gets richer" theory is seen as a result of the improved forgetting mechanism. This forgetting mechanism speeds up the emergence of fame as it is expected. However, as

n increases, for the bigger values of ρ, the f_{max} and $f_{5\%}$ values tends to fit on a line that has nearly the same slop with the f_{max} and $f_{5\%}$ lines of SRM respectively. The improved forgetting mechanism is more effective for lower n and ρ values.

To analyze the effect of the $t_{keep\_duration}$ parameter, we run the simulation again for different $t_{keep\_duration}$ values. The simulation results given in Figure 3.11 are calculated as the average of 10 runs of each $t_{keep\_duration}$ value for $n = 100$. The simulation result for iterative selection of a_G is also shown for $n = 100$ in the same figure in order to perform comparison more easily.

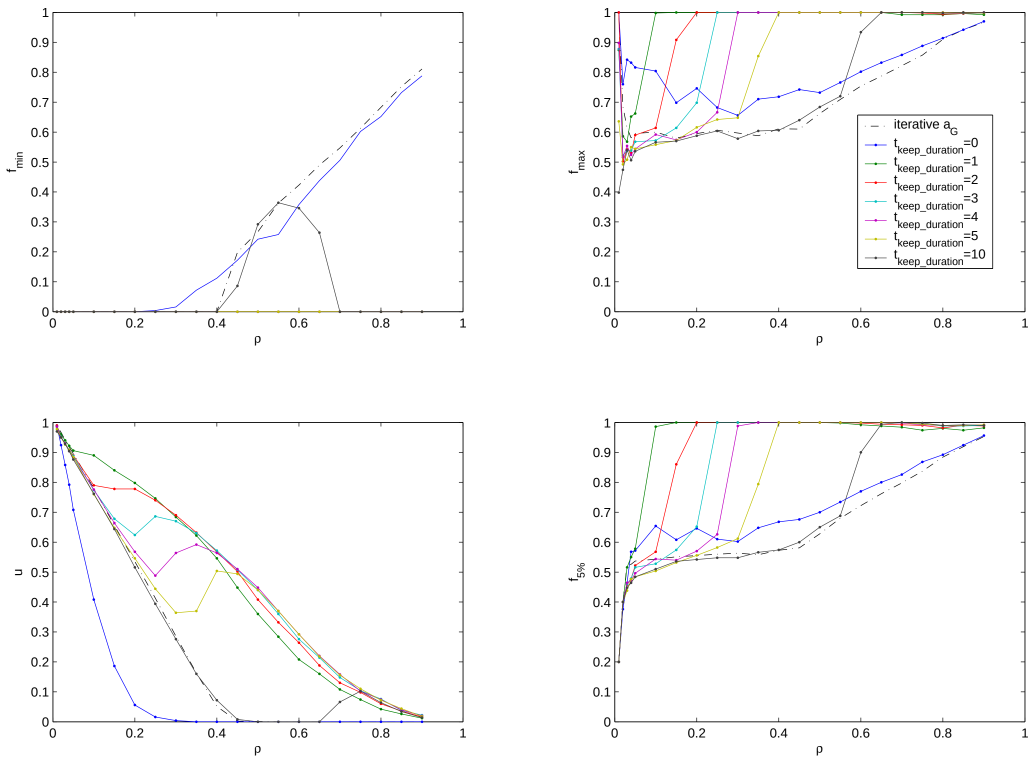
The obvious result that can be seen in Figure 3.11 is the maximum and cumulative fame values increase for $t_{keep\_duration} \leq 10$ whereas the maximum and cumulative fame values decrease for $t_{keep\_duration} \geq 30$. At the end of simulations for $t_{keep\_duration} = 10^5$, which is equal to the simulation end time, the initial fame of the agents in the system does not change as it is expected.

3.4. Recommendation Counter Based Selection of a_R

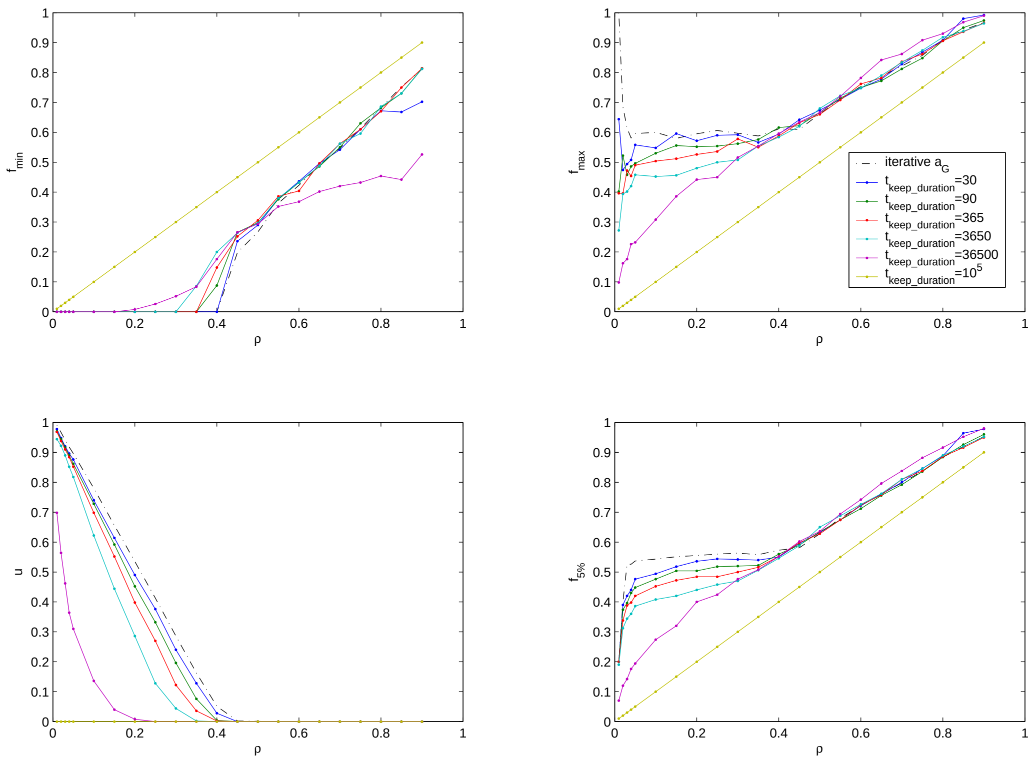
Consider two agents a_1 and a_2. They are recommended to agent a_3 and learned by a_3. Suppose a_3 keeps getting recommendations of a_1 but no further recommendation of a_2. In SRM, when a_3 is selected as a_G, a_1 and a_2 have equal chances to be selected as a_R. Now, lets think about some real life examples. Let there are two restaurants one of which is recommended to you more than the other one by your friends. If you have to recommend a restaurant to someone, which of them would you prefer to recommend?

In recommendation counter based selection of a_R, if an agent gets further recommendations, its chance to be selected as a_R is increased. A function of recommendation counter and learned time of an agent is used to implement improved recommending mechanism.

When an agent a_R is learned by an agent a_T, the $recommendation\_counter_{rt}$ is set to 1 and $learned\_time_{rt}$ is set to current simulation time. After a_R is learned



(a)



(b)

Figure 3.11. Simulation results of recommendation counter based selection of a_F where $n = 100$. (a) $t_{keep\_duration} \in \{0, 1, 2, 3, 4, 5, 10\}$, and (b) $t_{keep\_duration} \in \{30, 90, 365, 3650, 36500, 10^5\}$.

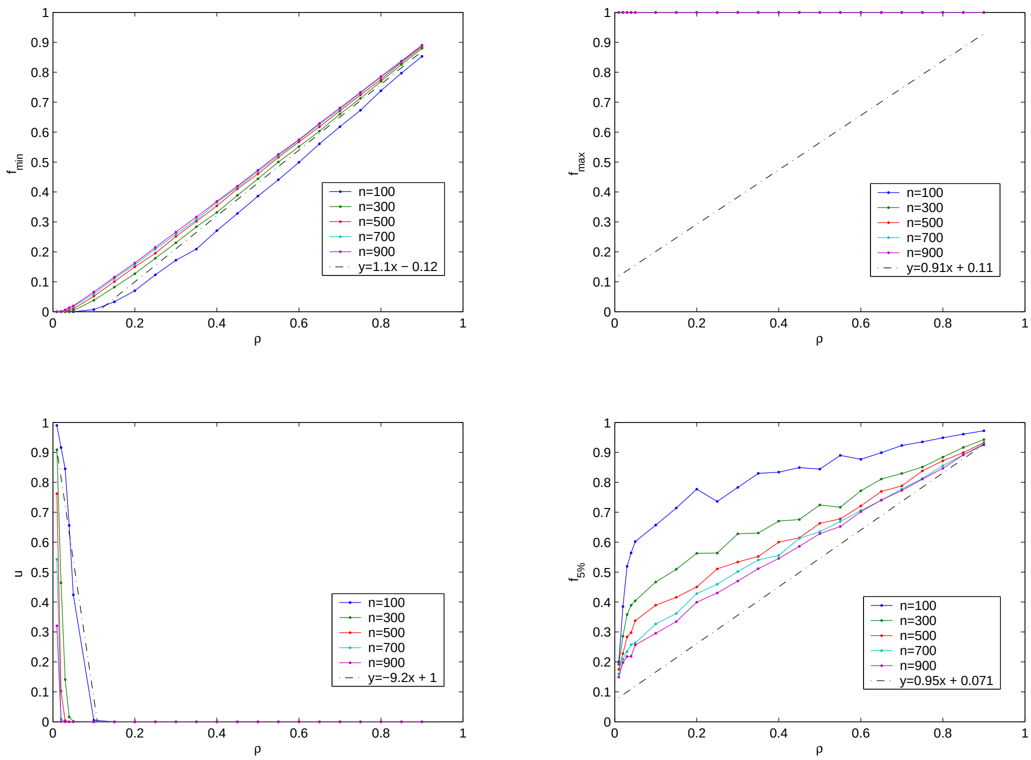


Figure 3.12. Simulation results of recommendation counter based selection of a_R

by a_T, for each further recommendation of a_R to a_T, the $recommendation\_counter_{rt}$ is incremented by one.

When agent a_T has to recommend an agent, the agent with the biggest recommendation counter value is selected as a_R from the memory of a_T. If there are more than one agents with the same biggest recommendation counter value, then among those agents the one with the biggest learned time value, that is the recent learned one, is selected as a_R. The algorithm is given in Appendix A.4.

In the simulations, the used n values are in range $100 - 900$ with 200 increments. We use the same ρ values that are used in SRM. The value of v that we used is 10^5. In Figure 3.12, the result of the simulations, where the improved recommending mechanism is used, is given. The shown results are calculated by taking the average of 10 runs. Simulation results of all runs, where $n = 900$, is given in Appendix B.4.

If we compare the simulation results with the results of SRM, it can be seen that

f_{min} (except for $n = 100$), f_{max}, and $f_{5\%}$ values are higher whereas u values are lower than the ones in SRM. This recommending mechanism speeds up the emergence of fame as it is expected. Also, this recommending mechanism slows down the forgetting of agents.

3.5. Advertisement Based Selection of a_R

Another extension that we make to SRM is using advertisement based selection of a_R. We divide the agents in the population into two sets, A_{adv} and $A_{noneAdv}$, where A_{adv} is the set of agents that will be advertised and $A_{noneAdv}$ is the set of agents that contains remaining agents in the population. The number of agents in A_{adv} is n_{adv} ($\leq n$). The advertisement ratio, θ, is n_{adv}/n ($0 \leq \theta \leq 1$). The probability to select from the advertised agents is p_{adv} ($0 \leq p_{adv} \leq 1$).

For the advertisement based selection of a_R, before the simulation starts, we put the first n_{adv} agents in the population into the set A_{adv} and the remaining agents into the set $A_{noneAdv}$. Then, the selection of a_R based on advertisement works as follows:

- If the memory of the giver agent a_G contains agents from both of the sets (A_{adv} and $A_{noneAdv}$), then which set to be used is based on the probability p_{adv}. With probability p_{adv}, the set A_{adv} and with probability $1 - p_{adv}$, the set $A_{noneAdv}$ is selected. Once the set is selected, a_R is randomly selected from the elements of $\{$selected set $\bigcap M_G\}$.
- Else, the memory of a_G contains agents from only A_{adv} or $A_{noneAdv}$. Thus, an agent from the memory of a_G is selected randomly as a_R as it is in SRM. The algorithm is given in Appendix A.5.

We make simulations with different combinations of θ and p_{adv} to analyze the effect of our proposed selection of a_R based on advertisement. We investigate the change in average fame f_{avg}, fame of top 5% $f_{5\%}$, and percentage of the forgotten agents u for each set A_{adv} and $A_{noneAdv}$ with respect to θ. We have to define f_{avg}, $f_{5\%}$ and u for two sets:

- f_{avg} of a set is calculated as the average fame of the agents in the set
- $f_{5\%}$ of a set is (sum of fames of top 5% agents in the set according to their fame values) $/0.05n$
- u of a set is the percentage of the forgotten agents in the set

We concentrate on the following graphs to see the effect of our proposed advertisement based selection of a_R:

- $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ vs. θ
- $(f_{5\%^{Adv}} - f_{5\%^{NoneAdv}})$ vs. θ
- $(u_{Adv} - u_{NoneAdv})$ vs. θ

In simulations, we take $n = 500$, ρ values as $\{0.01, 0.05, 0.1, 0.2\}$, θ and p_{adv} values as $\{0.01, 0.02, 0.03, 0.04, 0.05, 0.1, 0.2, 0.3, 0.4, 0.45, 0.49, 0.5, 0.51, 0.55, 0.6, 0.7, 0.8, 0.9, 0.95, 0.99\}$. For each ρ value, we make simulations with all combinations of θ and p_{adv}. The graphs we concentrate on are drawn on separate sub-graphs for each p_{adv} value and the result for each ρ value is drawn separately on those sub-graphs. Additionally, $x = p_{adv}$ line is drawn on all plots. The shown results are calculated by taking the average of 10 runs.

In Figure 3.13, $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ vs. θ results are shown. According to those results, advertised agents are in fact promoted whenever $\theta < p_{adv}$ independent of the value of ρ. If $\theta > p_{adv}$, then its in favor of agents that are not advertised in terms of fame, therefore none-advertised agents are indirectly promoted. The indicators of this interpretation are

- The intersection of lines of different ρ values with the line $x = p_{adv}$ is always at the point where $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ is 0.
- $(f_{avg^{Adv}} - f_{avg^{NoneAdv}}) > 0$ for $\theta < p_{adv}$ is seen.
- $(f_{avg^{Adv}} - f_{avg^{NoneAdv}}) < 0$ for $\theta > p_{adv}$ is also seen.

Also, it can be said that the ρ value that is used does not affect this behavior, but it affects the value of $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$. The difference for bigger ρ values is bigger than the smaller ρ values.

Another observation that we make for Figure 3.13 is the symmetry seen among some sub-graphs. The $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ values of two sub-graphs, whose total of used p_{adv} values is 1, are mirror view of each other with respect to the point (0.5, 0). In other words, the promotion that advertised agents gain for the given θ and p_{adv} is the same as the indirect promotion that none-advertised agents gain for the θ as 1−(given θ) and p_{adv} as 1−(given p_{adv}).

In Figure 3.14, $(f_{5\%^{Adv}} - f_{5\%^{NoneAdv}})$ vs. θ results are shown. The observations that are stated for Figure 3.13 are also observed in Figure 3.14 with some exceptions especially for $\rho = 0.01$. The reason for those exceptions may be due to the fact that the number of advertised agents becomes smaller as θ decreases, the number of none-advertised agents becomes smaller as θ increases and we use 5% of those agents to calculate $f_{5\%}$. Also, the memory capacity of the agents decreases as ρ decreases.

In Figure 3.15, $(u_{Adv} - u_{NoneAdv})$ vs. θ results are shown. Again, the intersection of lines of different ρ values with the line $x = p_{adv}$ is always at the point where $(u_{Adv} - u_{NoneAdv})$ is 0. Now, $(u_{Adv} - u_{NoneAdv}) < 0$ for $\theta < p_{adv}$ and $(u_{Adv} - u_{NoneAdv}) > 0$ for $\theta > p_{adv}$ is seen. Those results lead to the same interpretation stated for $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ vs. θ, that is advertised agents are in fact promoted if $\theta < p_{adv}$ since $(u_{Adv} - u_{NoneAdv}) < 0$ for $\theta < p_{adv}$. If $\theta > p_{adv}$, then its in favor of agents that are not advertised in terms of fame, therefore none-advertised agents are indirectly promoted.

The ρ value that is used again does not affect the behavior stated above, but it affects the value of $(u_{Adv} - u_{NoneAdv})$. The symmetry property is also observed in Figure 3.15.

After examining those results, we want to explain the intersection of lines of different ρ values with the line $x = p_{adv}$ is always at the point $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ is 0 in Figure 3.13 and at the point $(u_{Adv} - u_{NoneAdv})$ is 0 in Figure 3.15. Now, lets think a giver agent of memory size n and its memory contains all of the agents in the population. Let p_i is the probability of selecting an advertised agent a_i in M_G as a_R and p_j the probability of selecting a none-advertised agent a_j in M_G as a_R. The p_i and p_j can be formulated as

$$p_i = p_{adv}\frac{1}{n_{adv}}$$

$$p_j = (1 - p_{adv})\frac{1}{n - n_{adv}}$$

Since $\theta = n_{adv}/n$, we can replace n_{adv} with θn in the formula of p_i and p_j. The rewritten formulas for p_i and p_j are:

$$p_i = p_{adv}\frac{1}{\theta n}$$

$$p_j = (1 - p_{adv})\frac{1}{n - \theta n} = (1 - p_{adv})\frac{1}{n(1-\theta)}$$

If we subtract p_j from p_i, we get

$$p_i - p_j = \frac{p_{adv}}{\theta n} - \frac{1 - p_{adv}}{n(1-\theta)} = \frac{p_{adv} - \theta}{n\theta(1-\theta)}$$

If we look at the last equation for $p_i - p_j$,

- $p_i - p_j = 0$ for $\theta = p_{adv}$
- $p_i - p_j > 0$ for $\theta < p_{adv}$
- $p_i - p_j < 0$ for $\theta > p_{adv}$

The conditions above are seen in our simulation results. Thus, advertised agents are in fact promoted whenever $\theta < p_{adv}$. If $\theta > p_{adv}$, then its in favor of agents that are not advertised in terms of fame, therefore none-advertised agents are indirectly promoted. Note that, for simplicity we get memory size as n ($\rho = 1$). ρ has also effects on our simulations but we ignore it while formulating our problem.

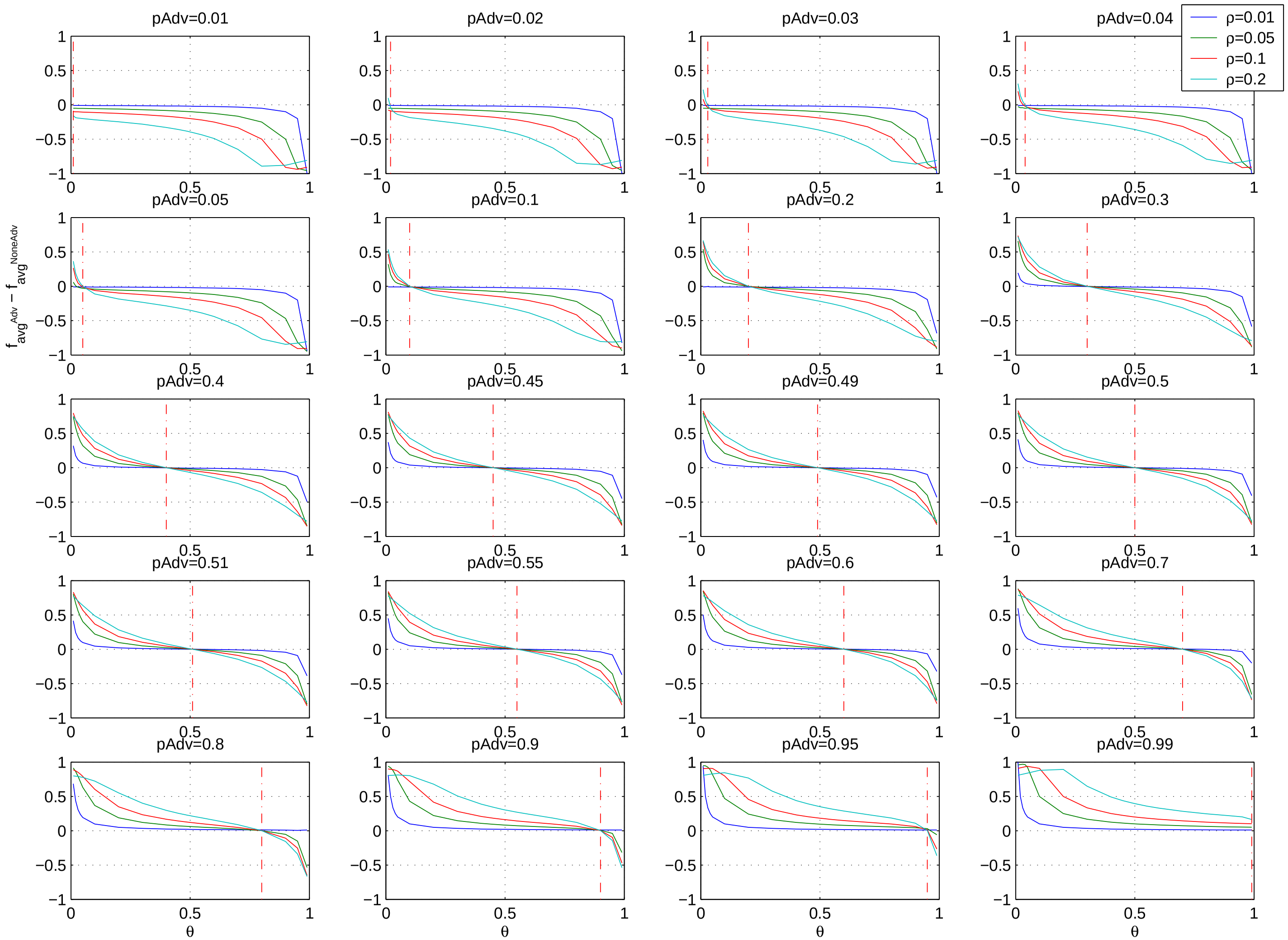


Figure 3.13. Simulation results of advertisement based selection of a_R for $(f_{avg^{Adv}} - f_{avg^{NoneAdv}})$ vs. θ where $n = 500$

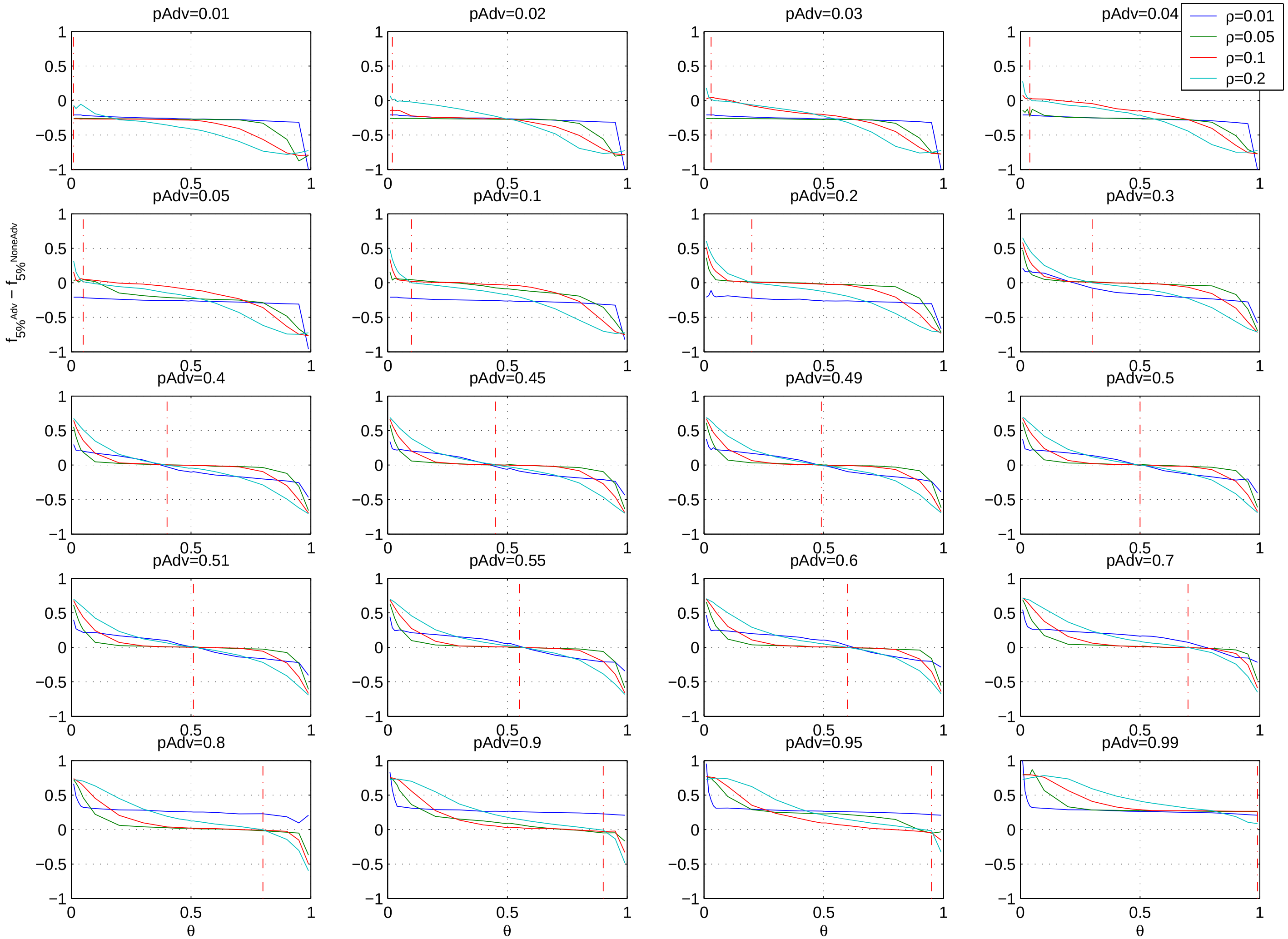


Figure 3.14. Simulation results of advertisement based selection of a_R for $(f_{5\%^{Adv}} - f_{5\%^{NoneAdv}})$ vs. θ where $n = 500$

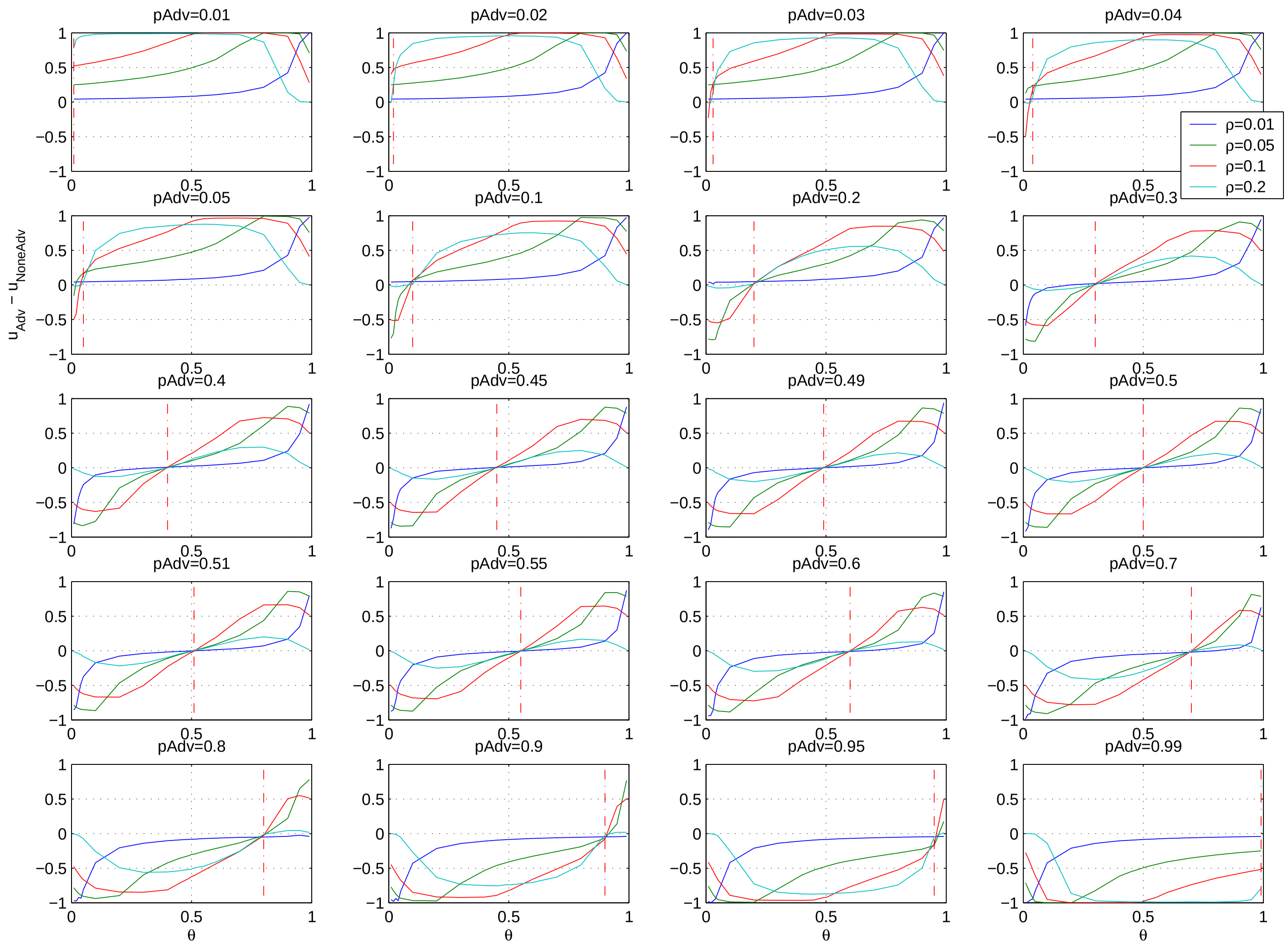


Figure 3.15. Simulation results of advertisement based selection of a_R for $(u_{Adv} - u_{NoneAdv})$ vs. θ where $n = 500$

4. EXTENSION OF SRM TO DYNAMIC POPULATION

The population size, n, is static in SRM. That is no births or deaths exist. Why not to try to extend the SRM to simulate fame in a world where agents born and die, in other words n becomes dynamic.

4.1. Dynamic n

In dynamic n, the population size, n, becomes dynamic by adding the *birth* and *death* of agents to the system. In dynamic n, every agent has a *life-span* $t_{life-span}$ that comes from a normal distribution with mean $t_{life-span-mean}$ and standard deviation $t_{life-span-deviation}$. The agent is *removed* from the population at $t_{death} = t_{birth} + t_{life-span}$, where t_{birth} is the time when agent is *added* to the population. The agents added to the population are called *living* agents whereas the agents removed from the population are called *dead* agents. Note that even if an agent is dead, it can still be remembered by some living agents.

In order to introduce the birth to the system, some addition properties are also added to the system. These are gender, reproduction period, and marriage.

The *gender* of an agent is selected randomly from the set {*male, female*}. The *reproduction period* of an agent starts at $t_{birth} + t_{reproductive}$ and ends at t_{death} where a uniform distribution in the range $[t_{reproductive-min}, t_{reproductive-max}]$ is used to select $t_{reproductive}$. An agent is called a *reproductive* during its reproductive period.

In order to have a child agent a_C, there must be a *marriage* between two reproductive single agents of opposite gender. In the system, there is a singles list for each gender that is initially empty. At each simulation cycle, a single reproductive agent a_S checks the opposite genders singles list to find a partner agent a_P. If there is one or more agents in this list, one of them is selected randomly as a_P, a_S and a_P get married and a_P is removed from the singles list of its gender. Otherwise, a_S puts

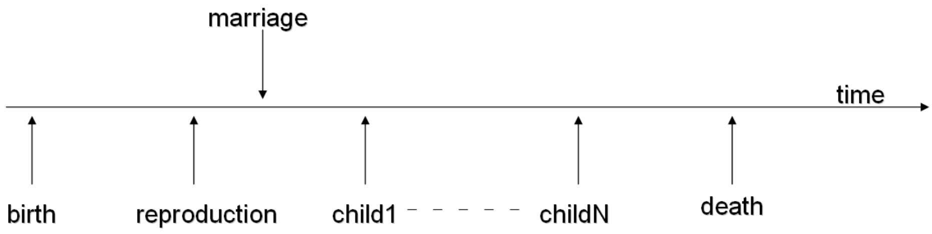


Figure 4.1. The life of an agent shown on a time line

itself at the end of the singles list of its gender. Note that if a single agent dies, it is removed from the singles list of its gender.

If two agents get married, they stay married as long as they both live. If one of the partners dies, the other partner becomes single again. If marriage takes place at $t_{current}$,the time t_{birth} to get a child is calculated as $t_{birth} = t_{current} + t_{child}$ where t_{child} comes from a uniform distribution $[t_{child-min}, t_{child-max}]$. At the birth of a child, time to get the next child is calculated similarly.

When child agent a_C is born, its gender, $t_{life-span}$ and $t_{reproductive}$ are determined and it is added to the population. Half of the initial memory content of a_C comes from one parent and the other half comes from the other parent. Recommendation process from a parent to the child is used to supply the initial memory content. Now, a_C knows some agents in the population, but nobody, including its parents, knows a_C. Again, the recommendation process is used to introduce a_C to the population. This time, each parent recommends their child and this is repeated $m/2$ times. Notice that parents may not know their child at the end of this process.

Note that, the initial memory content of the agents who born in the simulation startup is the same as that of SRM. That is, every agent who born in the simulation startup knows its m-neighbor. The algorithm of dynamic population is given in Appendix A.6. The birth time, starting time of reproduction, marriage time, time to get child and death time of an agent are shown on a time line in Figure 4.1. The death time of an agent can be at any time after its birth time which depends on its life span.

Table 4.1. Values of the parameters related with dynamic n used in the simulations

Parameter	Value
$t_{reproductive-min}$	20 year
$t_{reproductive-max}$	30 year
$t_{child-min}$	1 year
$t_{child-max}$	11 year
$t_{life-span-mean}$	60 year
$t_{life-span-deviation}$	20 year

The definition of the fame is needed to be updated for dynamic n due to the fact that the population size n changes. If we call n_t as the number of living agents at time t, the fame f_i of an agent a_i at time t becomes k_i/n_t.

The expected behavior for dynamic n is some of the dead agents are still well-known by the living agents and some of the living agents are also well-known by the other living agents as in the real world.

The simulation parameters that are related with dynamic n are listed in Table 4.1. One year corresponds to 365 simulation cycle.

In all simulations related with dynamic n, the initial population size n_0 is taken as 50. The used ρ values are the same as the ones in SRM. Simulation results are calculated by taking the average of 10 runs. Each simulation run ends at the end of 300 year. In other words, each simulation takes $300 \times 365 = 109500$ simulation cycles. At each simulation cycle, each agent in the population performs 10 recommendations ($v_C = 10$). Therefore, in the simulations related with dynamic n, the average recommendation per agent is $60 \times 365 \times 10 = 2.19 \times 10^5$ since $t_{life-span-mean}$ is 60 year.

The graphs for simulations related with dynamic n are again f_{min}, f_{max}, $f_{5\%}$, and u vs. ρ from left to right respectively. Living and dead agents are considered separately. The upper graphs are for living agents whereas the lower graphs are for dead agents. The values of those graph are calculated as follows:

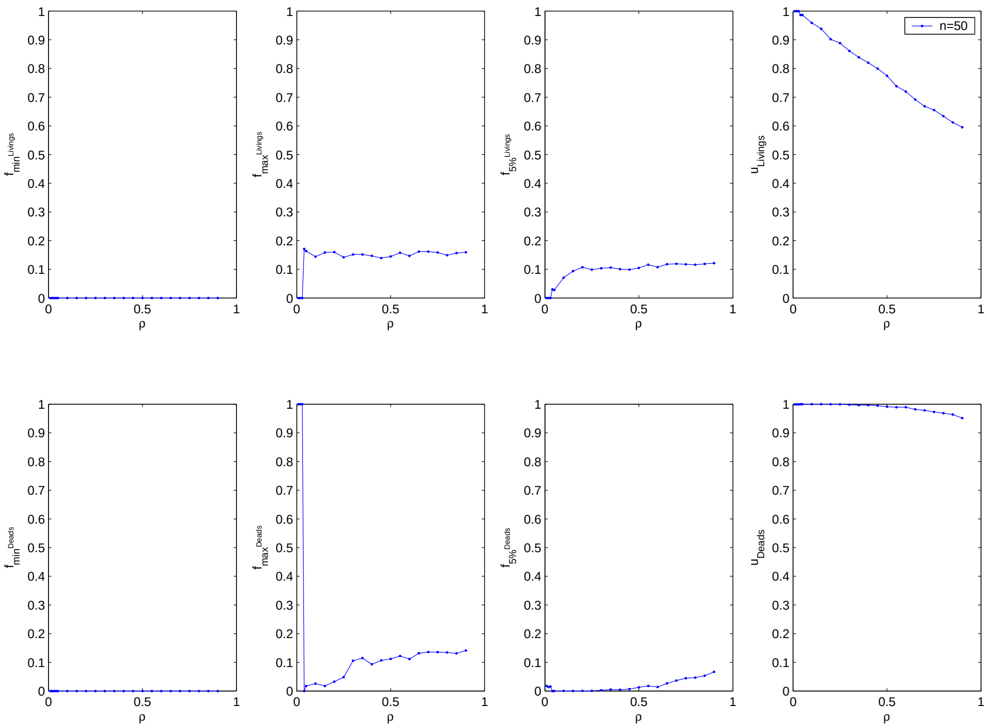


Figure 4.2. Simulation results of dynamic n

- $f_{min^{Livings}}$ is the minimum fame among the living agents whereas $f_{min^{Deads}}$ is the minimum fame among the dead agents.
- $f_{max^{Livings}}$ is the maximum fame among the living agents whereas $f_{max^{Deads}}$ is the maximum fame among the dead agents.
- $f_{5\%^{Livings}}$ is the average fame of the top 5% of the living agents whereas $f_{5\%^{Deads}}$ is the average fame of the top 5% of the dead agents according to their fame values.
- $u_{Livings}$ is the percentage of the forgotten living agents whereas u_{Deads} is the percentage of the forgotten dead agents.

In Figure 4.2, the simulation results related with dynamic n are shown. According to the simulation results, $f_{max^{Livings}}$ and $f_{5\%^{Livings}}$ values are bigger that $f_{max^{Deads}}$ and $f_{5\%^{Deads}}$ values in general. However, for $\rho \leq 0.03$, $f_{max^{Deads}} = 1$ whereas $f_{max^{Livings}} = 0$. But, at $\rho = 0.04$, $f_{max^{Deads}}$ drops to zero and then slightly increases as ρ increases. In addition to those observations, almost all of the dead agents are forgotten whereas $u_{Livings}$ values are lower than u_{Deads} values. Note that, simulation results of all runs is given in Appendix B.5.1.

Table 4.2. Population statistics related with the simulations for dynamic n

Property	Minimum	Average	Maximum	PRB
Births/1000 people/year	18.28	24.35	26.29	21
Deaths/1000 people/year	8.89	10.29	13.50	9
Avg. pop. growth rate %	0.58	1.38	1.65	1.2

Depending on dynamic n parameters, population can over explode or die out. We try to tune the parameters so that reasonable population statistics can be obtained. We take population statistics of Population Reference Bureau for world as reference [12]. The tuned parameters shown in Table 4.1 provide the statistics given in Table 4.2 which are close to the Population Reference Bureau (PRB) values.

4.2. Dynamic n with Recommendation Counter Based Selection of a_F

In this section, we use the recommendation counter based selection of a_F for dynamic n and analyze the results of the simulations. In the simulations, $t_{keep\_duration}$ is taken as 1 simulation cycle.

In Figure 4.3, the simulation results are shown and simulation results of all runs is given in Appendix B.5.2. If we compare those results with the results of dynamic n only, the difference in f_{max}, $f_{5\%}$ and $u_{Livings}$ can be seen immediately. Since most of the dead agents are added to the population before the most of the living agents, our recommendation counter based selection of a_F allows to increase fame values of dead agents and as a result dead agents dominate the living agents in terms of fame.

4.3. Dynamic m in Dynamic n

In SRM, the memory size of an agent is $m = n \times \rho$. All agents have the same memory capacity. If we change the definition of memory size of an agent as $m = n_t \times \rho$ for dynamic n, then m increases as the number of agents in the population increases.

Some changes have to be made to resolve ambiguities in dynamic n for dynamic

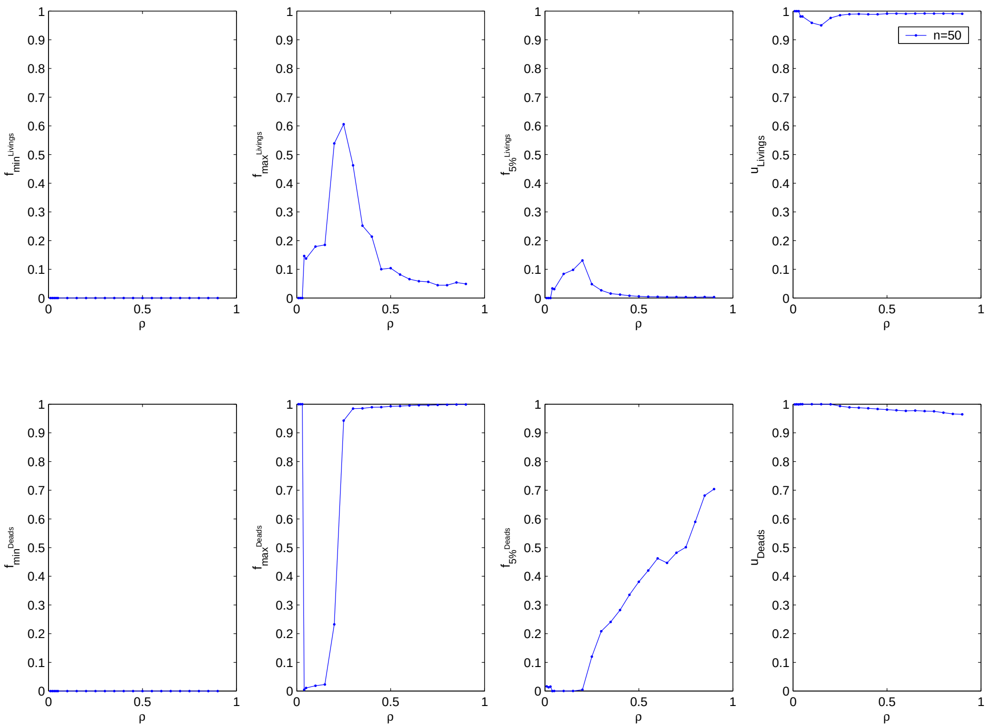


Figure 4.3. Simulation results of dynamic n with recommendation counter based selection of a_F

m:

- In dynamic n, it was said that "Half of the initial memory content of a_C comes from one parent and the other half comes from the other parent". We change this as "Each parent makes recommendations to its child, and the number of recommendation that a parent agent a_P makes is half of the memory size of a_P".
- In dynamic n, it was said that "The recommendation process is used to introduce a_C to the population. This time, each parent recommends their child and this is repeated $m/2$ times". We change this as "The recommendation process is used to introduce a_C to the population. This time, each parent makes recommendations where a_R is the a_C, and the number of recommendation that a parent agent a_P makes is half of the memory size of a_P"

In Figure 4.4, the simulation results are shown and simulation results of all runs is given in Appendix B.5.3. These results resembles the results of SRM except the slop values of the lines. The fame values are nearly equally shared between living and dead

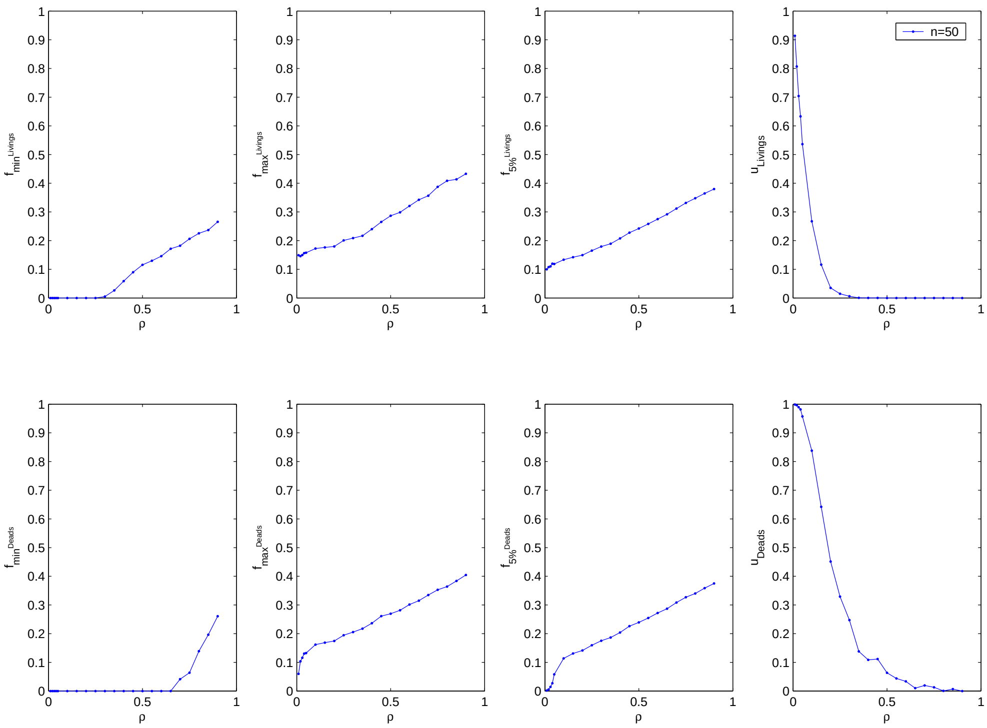


Figure 4.4. Simulation results of dynamic m in dynamic n

agents.

4.4. Dynamic m in Dynamic n with Recommendation Counter Based Selection of a_F

In this section, we use the recommendation counter based selection of a_F for dynamic m in dynamic n and analyze the results of the simulations. In the simulations, $t_{keep\_duration}$ is taken as 1 simulation cycle.

In Figure 4.5, the simulation results are shown and simulation results of all runs is given in Appendix B.5.4. Again, our recommendation counter based selection of a_F leads to "rich gets richer" theory. Most of the living and dead agents are forgotten whereas fame values of some living and dead agents increase sharply and make peak.

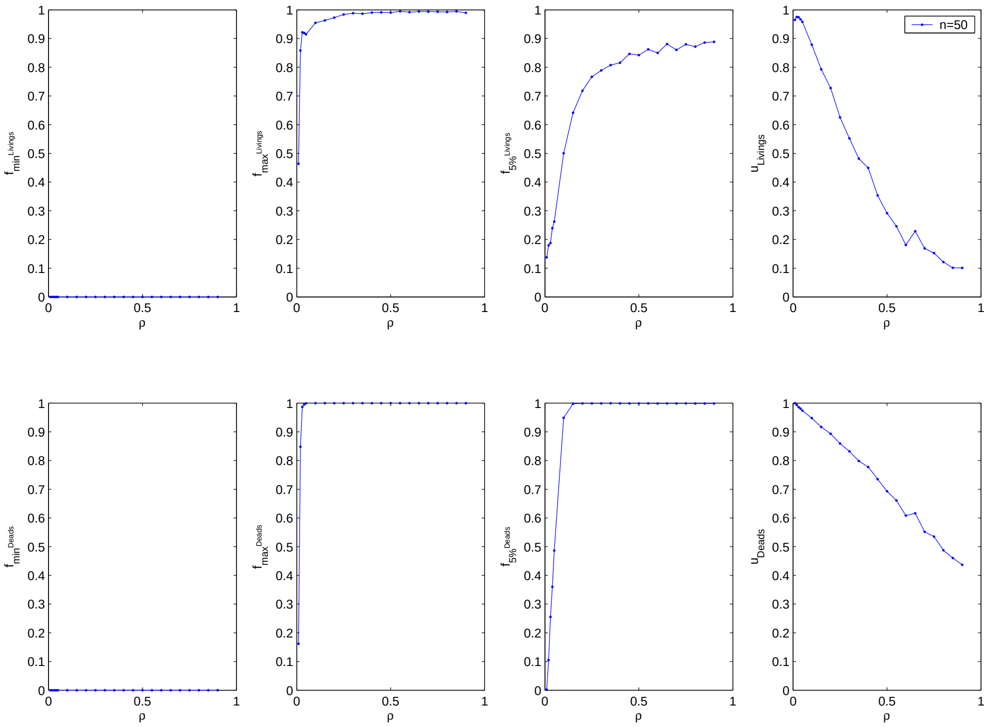


Figure 4.5. Simulation results of dynamic m in dynamic n with recommendation counter based selection of a_F

5. CONCLUSIONS

In this thesis, we extend the Simple Recommendation Model and analyze the effect of each extension. In addition to the extensions that we made, many different extensions can also be applied to SRM as it is stated in [1].

The first extension that we made is changing the random selection of the giver agent a_G to iterative. As a result of this extension, the simulation results of SRM are not changed. The reason of why results are not changed is due to the very big value of v that we used. If the value of v were very small, then the results could be much different. In the rest of our extensions, we use iterative selection of a_G.

The second extension that we made is changing the selection of the taker agent a_T from the population to selection of the taker agent a_T from the friends of a_G. The types of friendship graph that we used are Erdös and Rényi Random Graph, Small World Network and Graph Constructed By Barabási-Albert Model. Although we limit the number of agents that a_G can make recommendation to, the simulation results of SRM are not affected from any type of the friendship graph that we used. Therefore, fame emerges independent of the communication graph that we used. We also make an analysis to see if there exists a relation between the degree of an agent and its effects on its neighbors. However, we can not find any pattern that shows the relation between the degree of an agent and its effects on its neighbors by looking at the simulation results.

The third extension that we made is related with the selection of the forgotten agent a_F. Instead of randomly selecting a_F, we introduce a recommendation counter based selection of a_F. As a result of the simulations, we observe that our proposed forgetting mechanism leads to "rich gets richer" theory so that f_{min} values are lower whereas the f_{max}, $f_{5\%}$ and u values are higher than the ones in SRM. Thus, emergence of fame speeds up with the usage of our proposed forgetting mechanism. The proposed forgetting mechanism is more effective for lower n and ρ values.

The fourth extension that we made is related with the selection of the recommended agent a_R. We use a recommendation counter based selection of a_R instead of randomly selecting a_R. We see that this recommending mechanism speeds up the emergence of fame as it is in recommendation counter based selection of a_F. In addition to this result, another observation that we see is this recommending mechanism slows down the forgetting of agents.

The fifth extension is advertisement based selection of the recommended agent a_R. After analyzing many simulation results related with this extension, we found that advertised agents are in fact promoted if $\theta < p_{adv}$. If $\theta > p_{adv}$, none-advertised agents are indirectly promoted. For $\theta = p_{adv}$, advertised agents and none-advertised agents have equal chance in terms of fame.

The last extension that we made is related with dynamic population. After making four different simulations related with dynamic n, we see that we can not make too many comments about the results of those simulations because of so many parameters that we add to simulations related with dynamic n. Also, our simulations end at the end of 300 year that results in lower number of generations. Therefore, a more simplified version of the proposed extension of SRM to dynamic population can be taken as a starting point in order to investigate the emergence of fame in a world where population is dynamic. For example, mitosis separation of an agent can be used to introduce birth into the system so that the complexities that are added to the system by gender and marriage properties can be omitted.

APPENDIX A: Algorithms

A.1. Algorithm of SRM

Algorithm 1 Algorithm of SRM

Require: a_i knows its m-neighbor

for $i = 1$ to $n \times v$ **do**
    $a_G \Leftarrow$ random agent from the population
    $a_T \Leftarrow$ random agent from the population
    $a_R \Leftarrow$ random agent from M_G
    // begin: Perform Recommendation
    if $a_R \notin M_T$ **then**
        // a_T does not know a_R
        $a_F \Leftarrow$ random agent from M_T
        replace a_F with a_R in M_T // a_F is forgotten and a_R is learned by a_T
    else
        // a_T knows a_R; do nothing
    end if
    // end: Perform Recommendation
end for

A.2. Algorithm of Iterative Selection of a_G

Algorithm 2 Algorithm of Iterative Selection of a_G

Require: a_i knows its m-neighbor

for $i = 1$ to v/v_C **do**
    // One step is called as simulation cycle
    for $j = 1$ to v_C **do** // a_G is iteratively selected
        for $l = 1$ to n **do**
            $a_G \Leftarrow a_l$
            $a_T \Leftarrow$ random agent from the population
            $a_R \Leftarrow$ random agent from the M_G
            // Perform recommendation as it is done in SRM
        end for
    end for
end for

A.3. Algorithm of Recommendation Counter Based Selection of a_F

Algorithm 3 Algorithm of Recommendation Counter Based Selection of a_F

Require: $minRecCounter$, $learnedTimeOfMinRecCounter$ are set to max. integer value and a_F is set to $null$

if $a_R \notin M_T$ **then** // a_T does not know a_R
  for $i = 1$ to m **do**
    if $(current\_time - learned\_time_{it}) \geq t_{keep\_duration}$ **then**
      if $recommendation\_counter_{it} < minRecCounter$ **then**
        $a_F \Leftarrow M_i$
        $minRecCounter \Leftarrow recommendation\_counter_{it}$
        $learnedTimeOfMinRecCounter \Leftarrow learned\_time_{it}$
      else if $recommendation\_counter_{it} = minRecCounter$ **then**
        if $learned\_time_{it} < learnedTimeOfMinRecCounter$ **then**
          $a_F \Leftarrow M_i$
          $minRecCounter \Leftarrow recommendation\_counter_{it}$
          $learnedTimeOfMinRecCounter \Leftarrow learned\_time_{it}$
        end if
      end if
    end if
  end for
  if $a_F \neq null$ **then**
    replace a_F with a_R in M_T // a_F is forgotten and a_R is learned by a_T
    $recommendation\_counter_{rt} \Leftarrow 1$
    $learned\_time_{rt} \Leftarrow current\_time$
  end if
else // a_T knows a_R
  $recommendation\_counter_{rt} \Leftarrow recommendation\_counter_{rt} + 1$
end if

A.4. Algorithm of Recommendation Counter Based Selection of a_R

Algorithm 4 Algorithm of Recommendation Counter Based Selection of a_R

Require: $maxRecCounter$, $learnedTimeOfMaxRecCounter$ are set to min. integer value and a_R is set to $null$

for $i = 1$ to m **do**
  if $recommendation\_counter_{it} > maxRecCounter$ **then**
    $a_R \Leftarrow M_i$
    $maxRecCounter \Leftarrow recommendation\_counter_{it}$
    $learnedTimeOfMaxRecCounter \Leftarrow learned\_time_{it}$
  else if $recommendation\_counter_{it} = maxRecCounter$ **then**
    if $learned\_time_{it} > learnedTimeOfMaxRecCounter$ **then**
      $a_R \Leftarrow M_i$
      $maxRecCounter \Leftarrow recommendation\_counter_{it}$
      $learnedTimeOfMaxRecCounter \Leftarrow learned\_time_{it}$
    end if
  end if
end for

A.5. Algorithm of Advertisement Based Selection of a_R

Algorithm 5 Algorithm of Advertisement Based Selection of a_R

1. **if** $(\exists a_i \mid a_i \in A_{adv} \wedge a_i \in M_G) \wedge (\exists a_j \mid a_j \in A_{noneAdv} \wedge a_j \in M_G)$ **then**
2. $\quad randNumber \Leftarrow$ random value between 0 and 1
3. $\quad$ **if** $randNumber \leq p_{adv}$ **then**
4. $\quad\quad a_R \Leftarrow$ random $a_i \mid a_i \in A_{adv} \wedge a_i \in M_G$
5. $\quad$ **else**
6. $\quad\quad a_R \Leftarrow$ random $a_i \mid a_i \in A_{noneAdv} \wedge a_i \in M_G$
7. $\quad$ **end if**
8. **else**
9. $\quad a_R \Leftarrow$ random $a_i \mid a_i \in M_G$
10. **end if**

A.6. Algorithm of Dynamic n

Algorithm 6 Algorithm of Dynamic n

Require: Initially there are n agents of memory size m. Their gender, reproduction period and life-span are determined. Each agent knows its m-neighbor as it is in SRM

```
for i = 1 to v/v_C do
    // One step is called as simulation cycle, 365 cycle = 1 year
    Call DynamicPopulationOperations Method
    for j = 1 to v_C do // a_G is iteratively selected
        for l = 1 to sizeOfPopulation do
            a_G ⇐ a_l
            a_T ⇐ random agent from the population
            a_R ⇐ random agent from the M_G
            // Perform recommendation as it is done in SRM
        end for
    end for
end for
```

A.6.1. DynamicPopulationOperations Method

Algorithm 7 Dynamic Population Operations Method

for $l = 1$ to $sizeOfPopulation$ **do**
if a_i is reproductive **then**
if a_i has partner **then**
if partner of a_i dies **then**
put a_i into the singles list of its gender
else if its the birth time of the new child agent **then**
Determine the gender, reproduction period and life-span of the child agent a_C
a_i and its partner makes recommendation to a_C (the number of recommendation each one makes is half of its own memory size)
a_i and its partner recommends their child to randomly chosen living agents (the number of recommendation each one makes is half of its own memory size)
determine the birth time of the next child agent
end if
else // a_i has not any partner
$a_P \Leftarrow$ a randomly selected agent from the single list of opposite gender
if $a_P = null$ **then**
put a_i into the singles list of its gender
else
a_i and a_P get married (a_P is removed from the singles list)
determine the birth time of the new child agent
end if
end if
end if
end for
remove dead agents from population
add new born agents to the population

APPENDIX B: All Runs Of Some Simulation Results

B.1. Iterative Selection of a_G

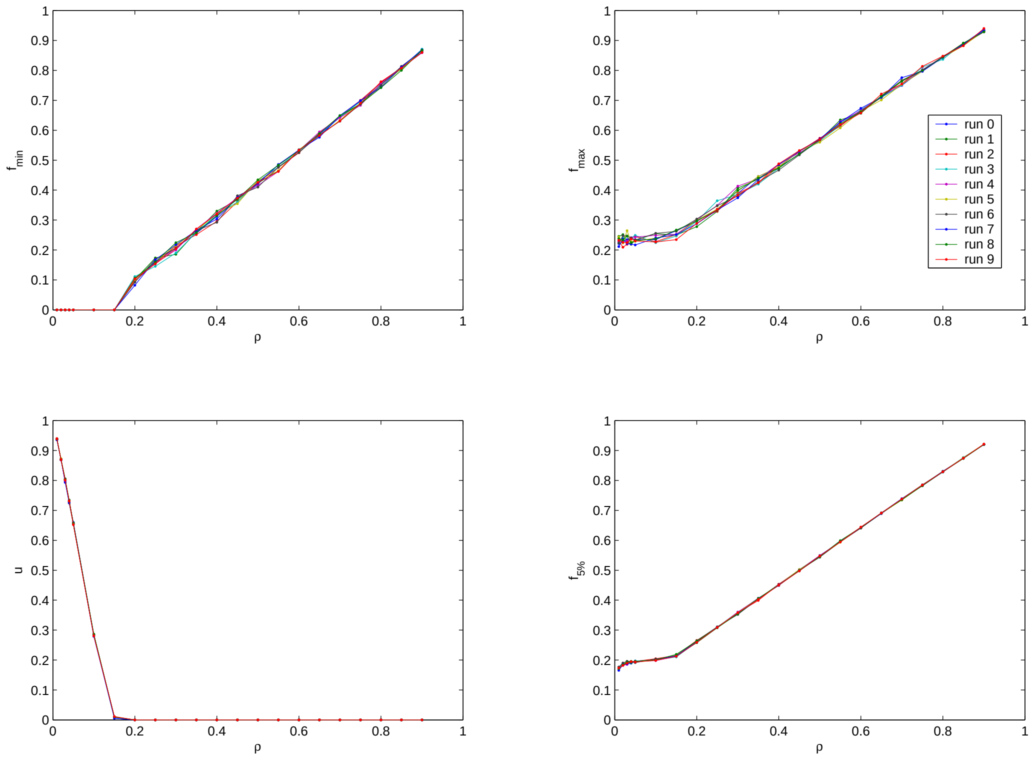


Figure B.1. Simulation results of all runs of iterative selection of a_G where $n = 900$

B.2. Selection of a_T From Friends

B.2.1. Erdös and Rényi Random Graph

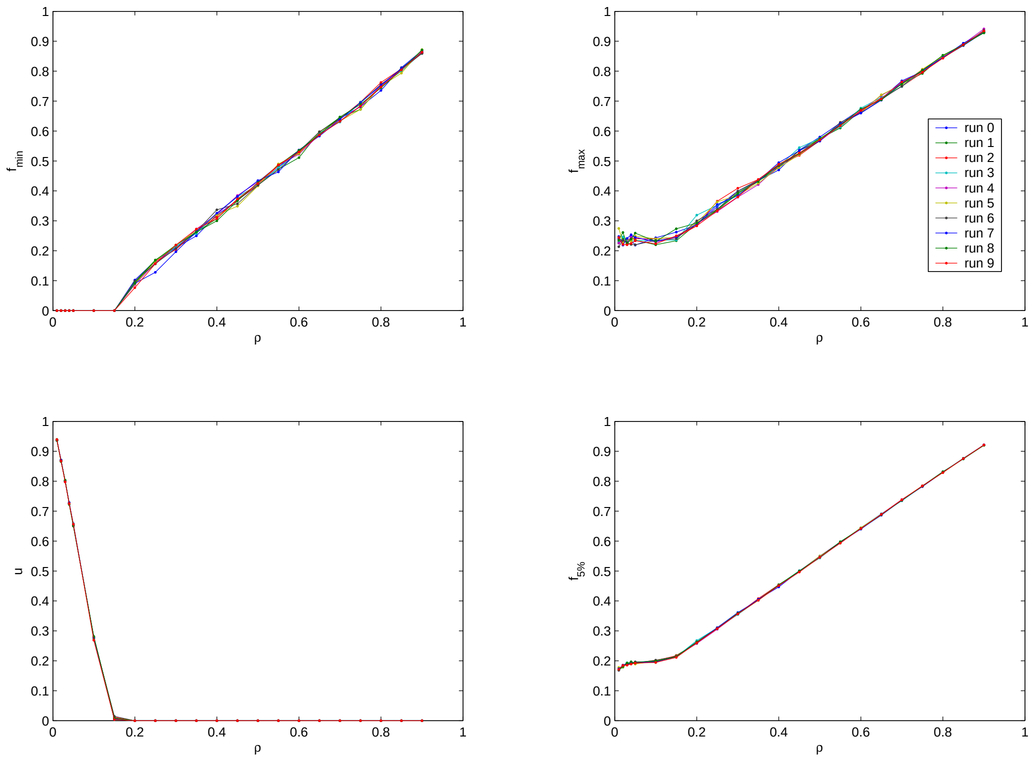


Figure B.2. Simulation results of all runs of selection of a_T from friends where $n = 900$ and the underlying communication graph is an ER random graph with $p = 0.1$

B.2.2. Small World Network

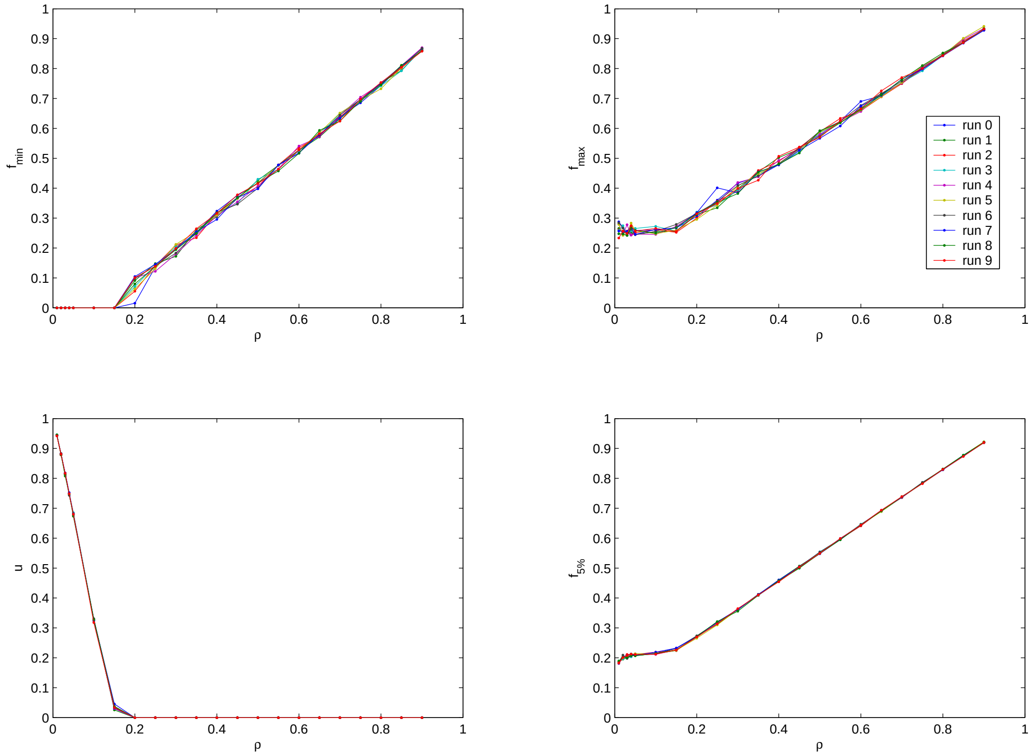


Figure B.3. Simulation results of all runs of selection of a_T from friends where $n = 900$ and the underlying communication graph is a small world network with $p = 0.1$

B.2.3. Graph Constructed By Barabási-Albert Model

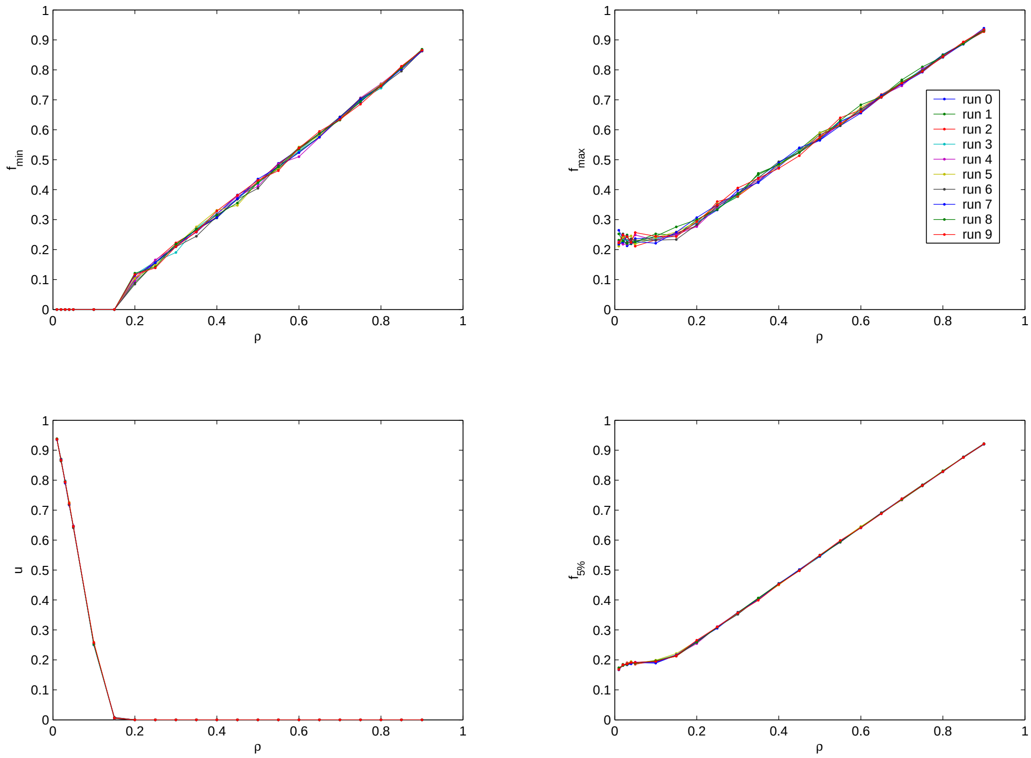


Figure B.4. Simulation results of all runs of selection of a_T from friends where $n = 900$ and the underlying communication graph is constructed by BA model with $d = 10$

B.3. Recommendation Counter Based Selection of a_F

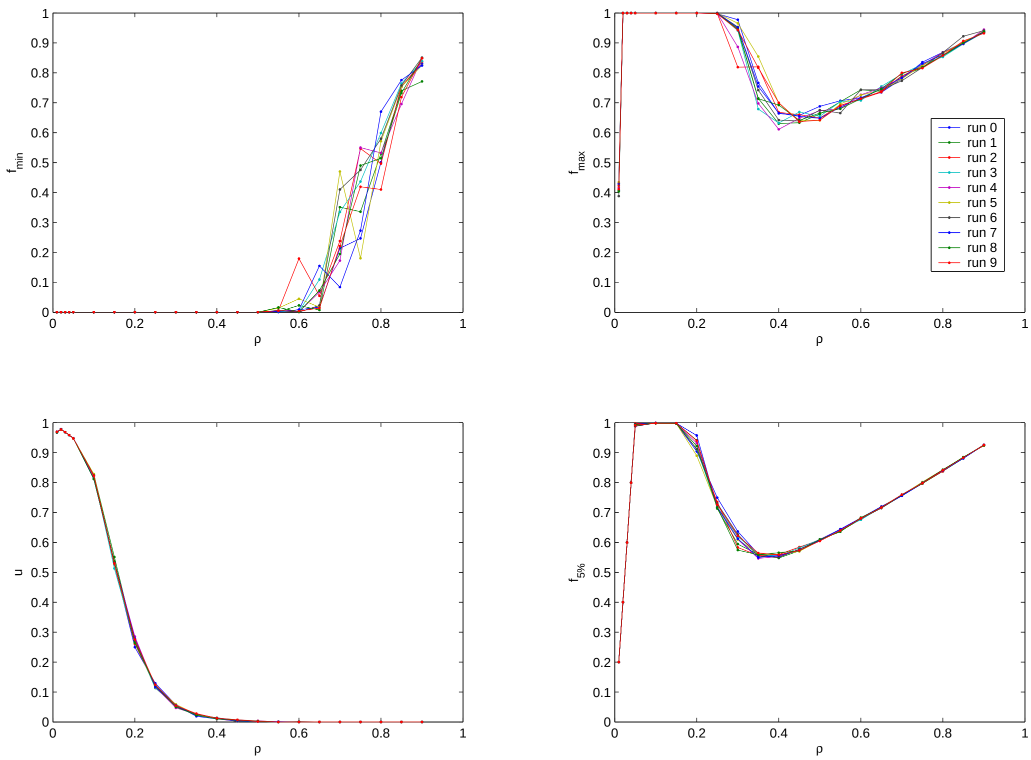


Figure B.5. Simulation results of all runs of recommendation counter based selection of a_F with $t_{keep\_duration} = 1$ where $n = 900$

B.4. Recommendation Counter Based Selection of a_R

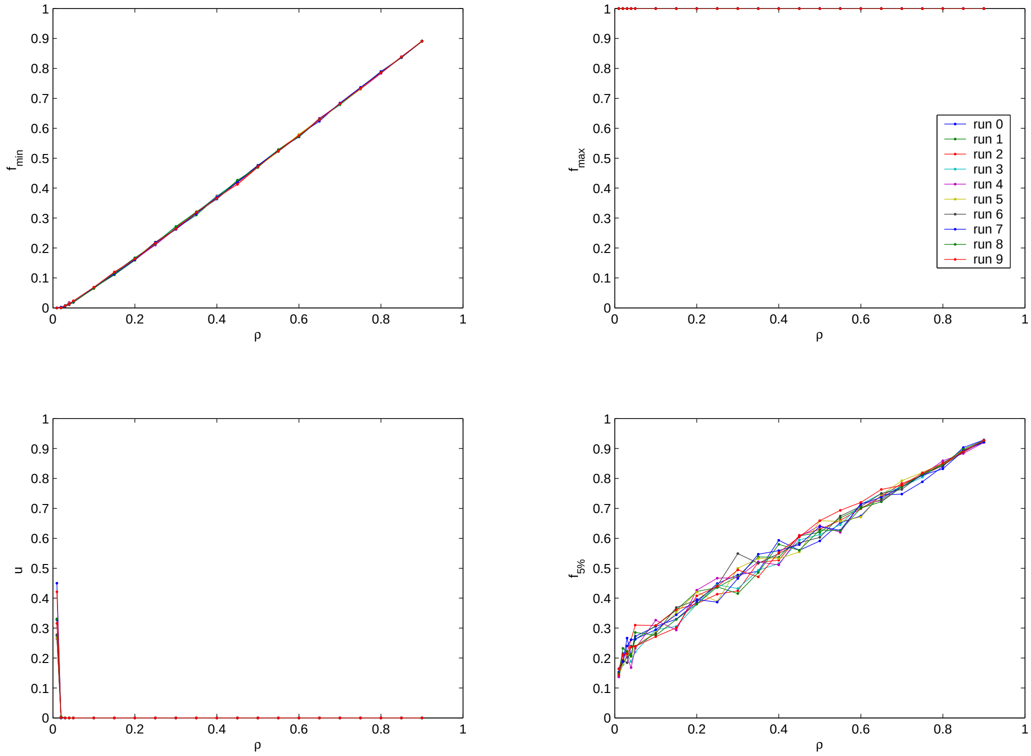


Figure B.6. Simulation results of all runs of recommendation counter based selection of a_R where $n = 900$

B.5. Dynamic Population

B.5.1. Dynamic n

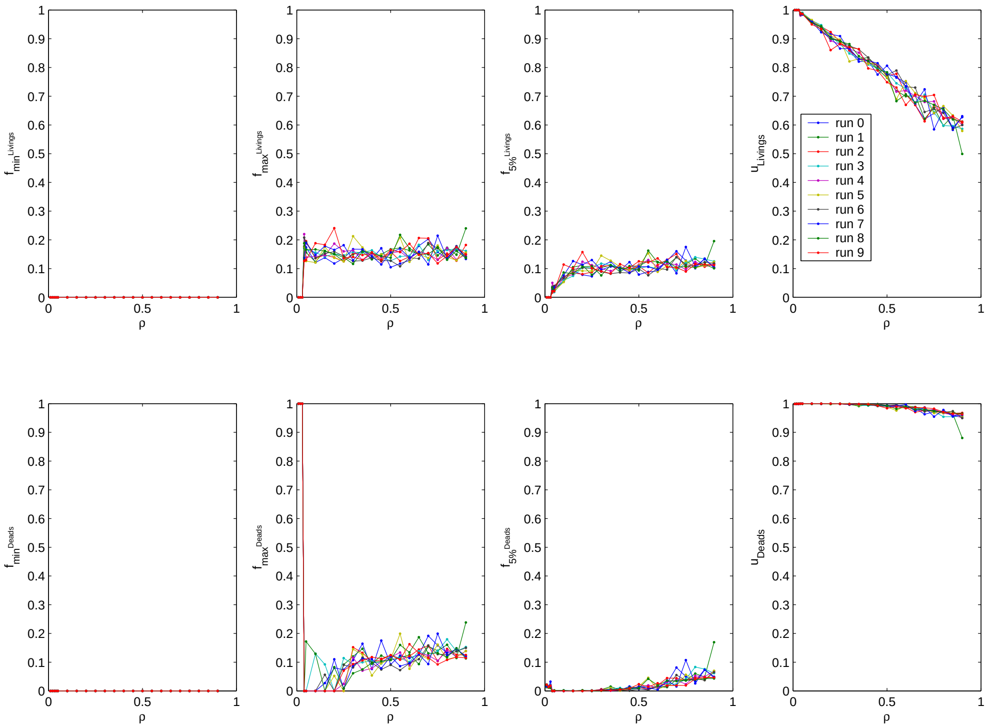


Figure B.7. Simulation results of all runs of dynamic n

B.5.2. Dynamic n with Recommendation Counter Based Selection of a_F

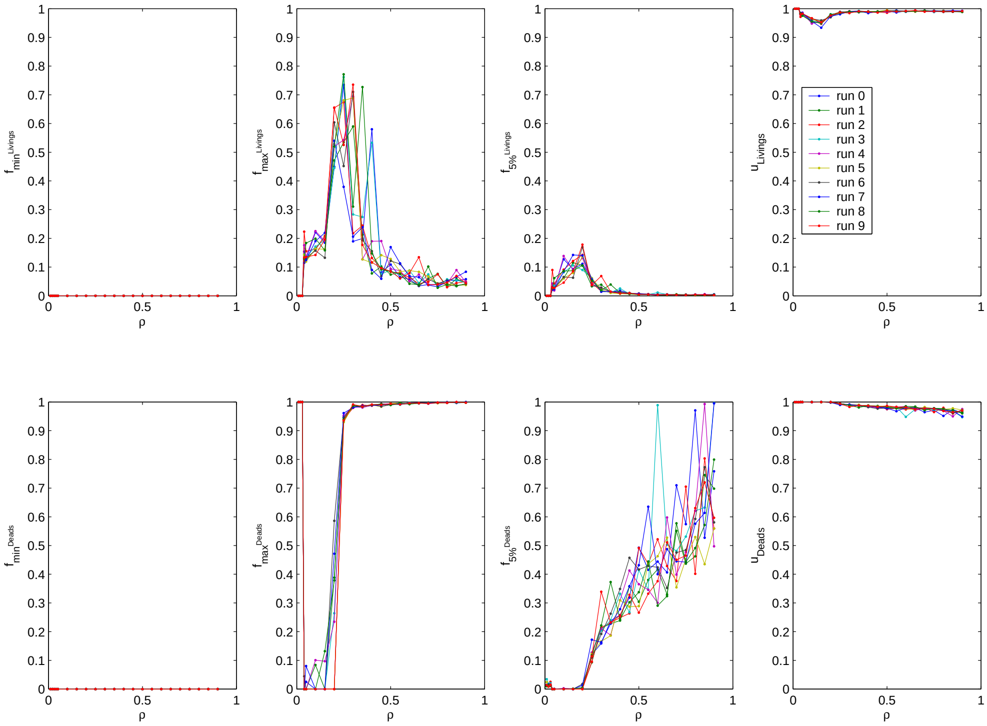


Figure B.8. Simulation results of all runs of dynamic n with recommendation counter based selection of a_F where $t_{keep\_duration} = 1$

B.5.3. Dynamic m in Dynamic n

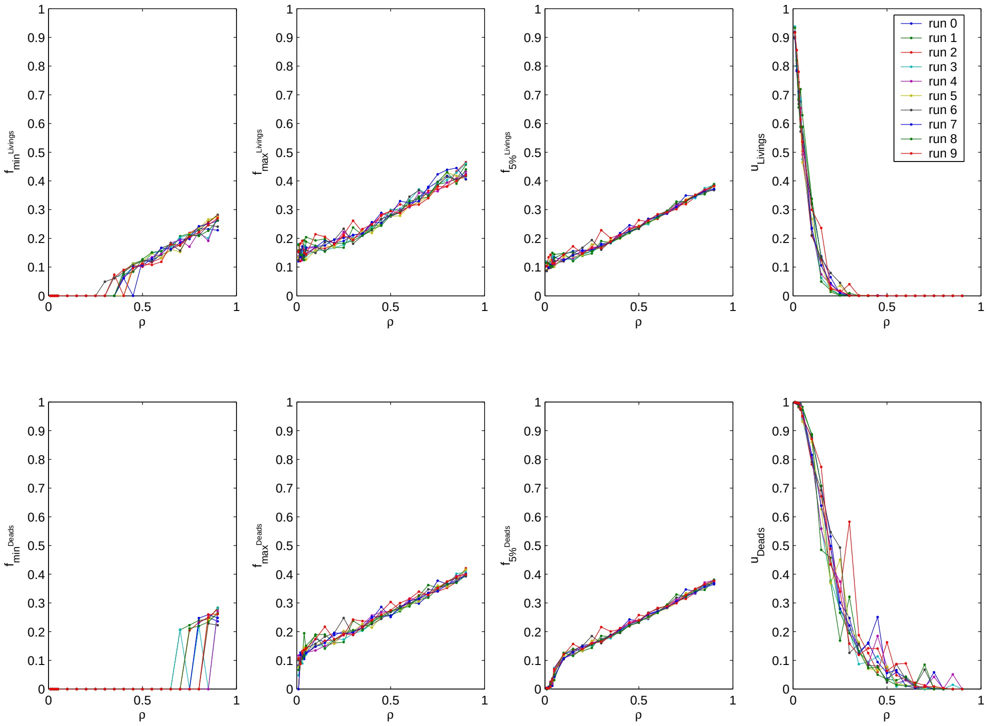


Figure B.9. Simulation results of all runs of dynamic m in dynamic n

B.5.4. Dynamic m in Dynamic n with Recommendation Counter Based Selection of a_F

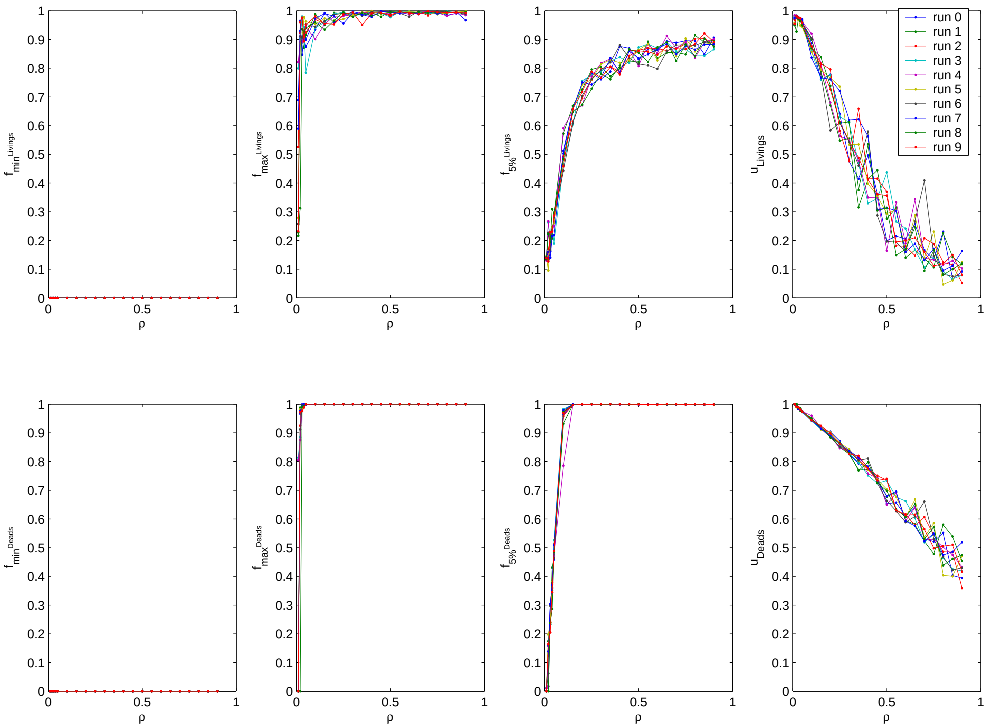


Figure B.10. Simulation results of all runs of dynamic m in dynamic n with recommendation counter based selection of a_F where $t_{keep\_duration} = 1$

REFERENCES

1. Bingol, H., Fame Emerges as a Result of Small Memory, Physical Review E 77, 036118 (2008)

2. Bak, P., How Nature Works, Copernicus, New York (1996)

3. Axelrod, R., The dissemination of culture, J. Conflict Resolut. 41, 203 (1997)

4. Berlekamp, E., J. Conway, and R. Guy, Winning Ways for Your Mathematical Plays, Academic, New York, Vol. 2 (1982)

5. Erdös, P., and A. Rényi, On Random Graphs, Publicationes Mathematicae 6, p. 290297 (1959)

6. Erdös, P., and A. Rényi, On The Evolution of Random Graphs, Publications of the Mathematical Institute of the Hungarian Academy of Sciences 5, 17 (1960)

7. Watts, D. J., and D. H. Strogatz, Collective dynamics of small-world networks, Nature, London, 393, 440442 (1998)

8. Milgram, S., The small world problem, Psychol. Today 2, 6067 (1967)

9. Kochen, M., The Small World, Ablex, Norwood, NJ (1989)

10. Bollabas, B., Random Graphs, Academic, London (1985)

11. Barabási, A.-L., and R. Albert, Emergence of Scaling in Random Networks, Science, 286, 509 (1999)

12. Population Reference Bureau, World Population Data Sheet (2007)