2006－DBS－140（I）（8）
2006／7／12

An Economic Dynamic Replication Model for Mobile-P2P networks

Anirban Mondal[1] Sanjay Kumar Madria[2] Masaru Kitsuregawa[1]

[1] Institute of Industrial Science
University of Tokyo
Japan
{anirban,kitsure}@tkl.iis.u-tokyo.ac.jp

[2]Department of Computer Science
University of Missouri-Rolla
USA
madrias@umr.edu

Abstract

In mobile ad-hoc peer-to-peer (M-P2P) networks, frequent network partitioning leads to typically low data availability, thereby necessitating data replication. This work proposes EcoRep, a novel economic model for dynamic replica allocation in M-P2P networks. EcoRep ensures fair replica allocation, while discouraging free-riding. EcoRep deploys a replica allocation algorithm, which considers relative importance of data items, load and energy issues, user mobility patterns, replica consistency and number of neighbours of a given user. Our performance study demonstrates that EcoRep is indeed effective in improving query response times and data availability in M-P2P networks.

1 Introduction

In a Mobile ad-hoc Peer-to-Peer (M-P2P) network, mobile hosts (MHs) interact with each other in a peer-to-peer (P2P) fashion. Proliferation of mobile devices (e.g., laptops, PDAs, mobile phones) coupled with the ever-increasing popularity of the P2P paradigm strongly motivate M-P2P network applications such as customers in a shopping mall sharing information about the cheapest available 'Levis' jeans and swapping shopping catalogues on-the-fly using mobile devices. During lunch-time, mobile users could share information about the cheapest price of steak across different nearby restaurants and exchange restaurant menus with each other. Visitors to a museum could request images/video-clips of different rooms of the museum to decide which room they will visit first. They could share songs and historical data about the museum. They could even request the museum's path information from other visitors as in virtual reality applications. Such P2P interactions among mobile users are generally not freely supported by existing mobile communication infrastructures. Absolute consistency is not a requirement in such applications [3, 17].

Data availability in M-P2P networks is typically lower than in fixed networks[1] due to frequent network partitioning arising from user movement and/or users switching 'on'/'off' their mobile devices. To improve M-P2P data availability, several replication schemes [10, 11, 23] have been proposed. However, these schemes do not address **fair replica allocation** since they allocate replicas solely based on the read/write access ratio of a data item d without considering the origin of queries for d (e.g., the E-DCG+ approach in [10, 11]). Hence, they regard d as 'hot' and create several replicas of d, even if a single MH M issues a disproportionately large number of (read) queries for d. This is inherently unfair since it favours M, thus these schemes are not able to *fairly* serve requests of multiple MHs. Moreover, they do not combat **free-riding** [9, 12, 18], which is rampant in P2P systems. (Nearly 90% of the peers in Gnutella were free-riders [1].) Since free-rider MHs do not participate in storing replicas, replication opportunities decrease, thereby degrading performance of these schemes.

This work proposes EcoRep, which is a novel economic model for dynamic replica allocation in M-P2P networks. The main contributions of EcoRep are three-fold:

1. It ensures fair replica allocation by considering the origin of queries for any given data item d to compute d's relative importance to the M-P2P network as a whole.
2. It discourages free-riding by requiring MHs to pay a *virtual currency* to the MHs from which they access data items or replicas i.e., an MH has to

[1]Data availability is less than 20% even in a wired environment [21].

contribute to the network for its access requests to be served.

3. It deploys a replica allocation algorithm, which considers relative importance of data items, load and energy of the MHs, user mobility patterns, replica consistency and the number of neighbours of an MH.

Notably, our main focus is on economy-based fair replica allocation, a pleasant side-effect of which is that of discouraging free-riding at no added cost. To manage replication efficiently, EcoRep deploys a super-peer architecture [24]. The super-peer (SP) is an MH, which generally moves within the region and which has maximum remaining battery power and processing capacity at a given time. The M-P2P network covers a relatively small area, hence the total number of MHs cannot be expected to be very large. SP knows the *schedule* of every MH comprising the MH's mobility pattern and the data items that the MH is likely to access at different points of time. Thus, SP is able to determine a near-optimal replica allocation period based on global information of MH schedules, hence it can better manage replica allocation. Our architecture avoids broadcast storm since every MH periodically sends replication-related information and list of data items/replicas stored at itself to SP, and SP broadcasts this information to all MHs. This is in contrast with distributed architectures (e.g., the E-DCG+ approach [10, 11]) that require every MH to periodically broadcast its list of allocated replicas to all MHs in the network, which causes a broadcast storm. Notably, queries need not pass via SP as every MH has adequate information to redirect queries, thus preserving P2P autonomy.

Our performance study indicates that EcoRep is indeed effective in improving query response times and data availability in M-P2P networks, while incurring relatively low communication traffic costs. To our knowledge, this is the first work to propose an economic model for data replication in M-P2P networks.

The remainder of this paper is organized as follows. Section 2 reviews existing works, while Section 3 describes the EcoRep economic model. Section 4 discusses the replica allocation algorithm of EcoRep. Section 5 reports our performance study. Finally, we conclude in Section 6 with directions for future work.

2 Related Work

Economic models have been discussed in [5, 14, 7, 4] primarily for resource allocation in distributed systems, which differs from our focus on data replication. A competitive micro-economic auction-based bidding model with support for load-balancing has been proposed in [5], while the work in [14] examines economy-based optimal file allocation. The proposal in [7] uses game-theoritic and trust-based ideas. Although the work in [4] considers economic replication, it does not address fairness in replica allocation and P2P concerns such as free-riding.

Works concerning free-riding include [12, 6, 16, 18, 9, 15]. The works in [12, 6, 16] propose incentive schemes to combat free-riding. The works in [18, 9] discuss utility functions to capture user contributions, while trust issues are examined in [15]. However, these works are completely othogonal to replication issues associated with free-riding.

The work in [13] proposes a suite of replication protocols for maintaining data consistency and transactional semantics of centralized systems. P2P replication systems include ROAM [19], Clique [20] and Rumor [8]. An update strategy, based on a hybrid push/pull Rumor spreading algorithm, for truly decentralized and self-organizing systems has been examined in [3]. The proposals in [10, 11] present three replica allocation methods with periodic and aperiodic updates, which consider limited memory space in MHs for storing replicas, access frequencies of data items and the network topology, to improve data accessibility in mobile ad-hoc networks. The E-DCG+ approach [11] is among the most influential replica allocation approaches. By creating groups of MHs that are biconnected components in a network, E-DCG+ shares replicas in larger groups of MHs to provide high stability. However, the architecture considered in [10, 11] is not suitable for our application scenarios since it does not consider user mobility patterns, load sharing and tolerance to weaker consistency.

3 EcoRep: An Economic Model for Data Replication in M-P2P networks

This section discusses EcoRep, which is an economic model for replica allocation in M-P2P networks. In EcoRep, each data item d has a *price* ρ (in terms of a *virtual currency*) that quantitatively reflects its relative importance to the M-P2P network as a whole. Whenever an MH M_i accesses d stored at an MH M_j, it pays the *price* ρ of d to M_j since M_j serves its request. Thus, M_i spends the amount ρ and M_j earns ρ. In essence, an MH has to provide *service* (i.e., storing data items/replicas that are accessed by other MHs) to the network for its own access requests to be served, which discourages free-riding. We do not regard 'relay functions' as 'service' since randomness ensures that each MH will generally have to forward comparable number of messages. We define the **revenue** of an MH as the difference between the amount of virtual currency that it earns and the amount that it spends. Revenue values may vary significantly across MHs. In EcoRep, when an MH joins the M-P2P network, SP provides the MH with a small amount of revenue to start with.

In EcoRep's super-peer architecture, each MH maintains recent read-write logs (including

timestamps) of its own data items and the read-logs of the replicas stored at itself. Periodically, each MH sends its logs to SP so that SP knows about data accesses. Each data item d is owned by only *one* MH, which can update d *autonomously* anytime; other MHs cannot update d. Memory space of MHs, bandwidth and data item sizes may vary. We assume location-dependent data access [22] i.e., an MH in region X will access data only from MHs in X. We define **load** L_i of an MH M_i as the job queue length of M_i normalized w.r.t. available bandwidth and service capacity to address heterogeneity.

$$L_i \ = \ J_{i,t_j} \ \div \ (\ \sigma_i \times B_i \ ) \qquad (1)$$

where J_{i,t_j} represents the job queue length of M_i at time t_j. σ_i and B_i are the normalized values of the service capacity and the available bandwidth of M_i respectively. σ_i is fixed for a given MH since it is hardware-dependent. $\sigma_i = (\ \sigma_{M_i}/\sigma_{min} \ )$, where σ_{M_i} is the service capacity of M_i and σ_{min} is a low service capacity. This work has used the minimum service capacity among all the MHs as σ_{min}, however any low value of service capacity would suffice for σ_{min}. Similarly, $B_i = (\ B_{M_i} \div B_{min} \ )$, where B_{M_i} represents the available bandwidth of M_i and B_{min} is a low bandwidth e.g., we have used 56 Kbps as the value of B_{min}.

In practice, MH owners do not move randomly since they have some *schedule*. An MH M's schedule contains information concerning M's location during any given time period T and the data items required by M during T. Each MH owner initially sends his schedule to SP and if later on, his schedule changes significantly, he will keep SP updated about these changes by piggybacking such information onto replica allocation-related messages to SP. Thus, SP is able to exploit MH schedules for replica allocation purposes. SP backs up information using the Internet as an interface to handle failures and we assume that some of the MHs have access to the Internet for backup purposes. If SP fails or network partitioning occurs, these MHs can connect to the Internet to obtain adequate information to act as SP.

Factors influencing the price of a data item

Price of a data item d depends on **d's access frequency**, **number of MHs served by** d, **number of existing replicas of** d, **(replica) consistency of** d and **average response time for queries on** d. Since high access frequency of d generally implies that d is important, its price should increase with increasing access frequency. However, if d serves a large number of requests originating from only a few MHs in the M-P2P network, its price should decrease due to its relatively low importance to the network as a whole. Thus, in contrast with existing works, given two data items with equal access frequencies, the price of the data item that serves a larger number of MHs would be higher.

In consonance with economic principles, which postulate higher prices for rare items, d's price should increase with decreasing number of its replicas and vice-versa. Higher replica consistency of d implies higher price due to better quality of results. Response time τ for a query Q pertaining to d reflects the quality of service provided to the query issuing MH M_Q by the query serving MH M_S, hence shorter response times should command higher price. τ equals $(\ T_W + T_D \ )$, where T_W is the waiting time spent by Q in M_S's job queue and T_D is the download time for d. T_W depends on M_S's job queue length and its service capacity. T_D depends upon the bandwidth allocated by M_S for d's download, which is related to M_S's total bandwidth and the number of concurrent access requests to M_S.

Quantifying price and revenue

Based on the factors discussed above, let us now derive the formula for the price of a data item d. Let ρ_{rec} be the price of d based on the accesses during the most recent allocation period. ρ_{rec} may not always be able to reflect the true importance of d to the network (e.g., when spurious 'spikes' in d's access frequency occur), thereby leading to unnecessary replica allocations. Hence, we use ρ, which is the moving average price of d over a fixed number of allocation periods. As we shall see shortly, both ρ_{rec} and ρ are computed by SP since they need to be computed based on the accesses to d and its replicas across the entire M-P2P network. For the sake of convenience, Table 1 summarizes the notations, which shall henceforth be used in this paper.

For computing ρ_{rec} for d, we sort the MHs in *descending* order of access frequency for d during the most recent allocation period i.e., the first MH in this order made the most accesses to d. Given this order and using the notations in Table 1, SP computes ρ_{rec} of d as follows:

$$\rho_{rec} = \sum_{i=1}^{N_{MH}} (w_i \times n_i \times C_i \times BA_i \times PA_i) \ / \ ((N_R + 1) \times (J_{i,t_j}/\sigma_i)) \qquad (2)$$

where the weight coefficient w_i equals $(\ i/N_{MH} \ )$, thereby ensuring that more the number of MHs served by d, the more its price will be. For simplicity, assume $N_{MH} = 50$, $BA_i = 50$ units, $PA_i = 0.5$, $N_R = 1$ and $J_{i,t_j}/\sigma_i = 1$. If only a single MH makes 100 accesses to d, $\rho = 25$. When 4 MHs make 25 accesses each to d, $\rho = 62.5$. But, if 50 MHs make 2 accesses each to d, $\rho = 637.5$. Observe how ρ_{rec} increases as d serves more MHs, even though d's total access frequency is same.

$C_i = 1$ for queries answered by an MH's own data items since such queries are always answered with ab-

Notation	Significance
d	A given data item
ρ_{rec}	Price of d during most recent allocation period
ρ	Moving Average Price of d across multiple allocation periods
N_{MH}	Number of MHs
w_i	Weight coefficient for MH i for fairness in serving multiple MHs
n_i	Number of access requests for d originating from a given MH i
C_i	Average consistency with which MH i answered queries on d
BA_i	Bandwidth allocated by MH i for d's download
PA_i	Probability of availability of a given MH i
N_R	Number of existing replicas of d
J_{i,t_j}	Length of job queue at MH i during time t_j
σ_i	Service capacity of MH i

Table 1: Summary of notations

solute consistency. For queries answered by replicas, we consider three different levels of replica consistency, namely *high*, *medium* and *low*. C_i is assigned values of 1, 0.5 and 0.25 for high, medium and low consistency respectively. SP maintains a table $T_{\epsilon,C}$, which contains the following entries: (x%, high), (y%, medium), (z%, low), where x, y, z are error-bounds, whose values are application-dependent and pre-specified by the system at design time. Thus, C_i is computed using $T_{\epsilon,C}$. BA_i equals (T_B/N_a), where T_B is the sum of all the bandwidths that MH i allocated for each time when d was downloaded from itself and N_a is the total number of access requests for d at MH i. Notably, SP will know the probability of availability PA_i of MH i over a period of time by keeping records of such MH availability information in its log files. Number of copies of d in the M-P2P network equals the number of replicas in addition to the original data item itself, which explains the term $(N_R{+}1)$ in Equation 2. MH job queue lengths are normalized by service capacities to address service capacity heterogeneity.

After computing ρ_{rec}, SP computes the moving average price ρ of d. Since simple moving averages give equal weight to the last N reallocation periods, they are not able to react quickly to dynamically changing access patterns, which are characteristic of M-P2P networks. Hence, we use the Exponential Moving Average (EMA), which gives higher weights to recent access patterns relative to older access patterns. SP computes the price ρ of d as follows:

$$\rho = (\rho_{rec} - EMA_{prev}) \times 2/(N+1)) + EMA_{prev} \quad (3)$$

where EMA_{prev} is the EMA computed for the previous replica allocation period, and N is the number of replica allocation periods over which the moving average is computed. SP only needs to store the previous allocation period's value of EMA instead of the respective prices for the previous N allocation periods. This optimizes memory space usage of SP. Based on preliminary experiments, $N = 5$ is a reasonably good value for our application scenarios.

Suppose MH M stores p data items of its own and q replicas. Let ρ_i be the price of the i^{th} data item/replica. Let ne_{d_i} and ne_{r_i} be the access frequencies of the i^{th} data item and the i^{th} replica respectively. M's earning E follows.

$$E = \sum_{i=1}^{p} (\rho_i \times ne_{d_i}) + \sum_{i=1}^{q} (C_i \times \rho_i \times ne_{r_i}) \quad (4)$$

where C_i indicates the average consistency with which queries on replicas at M were answered. The first term of Equation 4 does not contain C_i since it concerns M's own data items, which are always absolutely consistent.

Now suppose M accesses p original data items and q replicas. Let ρ_i be the price of the i^{th} data item/replica. Let ns_{d_i} and ns_{r_i} be the access frequencies of the i^{th} data item and the i^{th} replica respectively. The amount S spent by M is computed as follows:

$$S = \sum_{i=1}^{p} (\rho_i \times ns_{d_i}) + \sum_{i=1}^{q} (C_i \times \rho_i \times ns_{r_i}) \quad (5)$$

In the above equation, the significance of C_i is the same as that of Equation 4. M's revenue is simply (E - S). Observe how EcoRep's economy-based paradigm of replication encourages MHs to store replicas so that they can increase their revenues, thereby ensuring that they obtain better service from the M-P2P network.

4 AReL: An Adaptive Revenue-Load-based Replica Allocation Algorithm for EcoRep

This section discusses the AReL (Adaptive Revenue-Load) replica allocation algorithm deployed by EcoRep. Since AReL considers both revenue and load of an MH for replica allocation, let us first explore the interaction between revenue and load of an MH M. Incidentally, an MH M may earn high amounts of virtual currency by serving only a few requests for

some high-priced data items, while not issuing any access requests of its own, the implication being that M's revenue would be high, even though M is underloaded. On the other hand, M could be serving a large number of access requests for low-priced data items, thereby implying that M's revenue would be low in spite of its high load. Even if M earns high amounts, M's revenue could still be low if M issues several access requests for high-priced data items. In essence, there is no direct correlation between the revenue and load of an MH. AReL uses a parameter λ that can be tweaked to adjust the relative importance of revenue and load during replica allocation. Thus, AReL is capable of *adapting* to the needs of different types of applications.

Computation of λ involves calculating the normalized values of revenue and load since revenue and load are measured in different units. Normalization is necessary to correctly reflect the relative weights of revenue and load. As a single instance, if an MH's revenue equals 10000 revenue units and its load equals 10 load units, its revenue value would always dominate, irrespective of the weights assigned to revenue and load by AReL. We define the normalized revenue of an MH M as $M_{Rev}/Total_{Rev}$, where M_{Rev} is the revenue of M and $Total_{Rev}$ is the summation of the revenues of all the MHs in the network. Similarly, normalized load of an MH M is defined as $M_{Load}/Total_{Load}$, where M_{Load} is the load of M and $Total_{Load}$ is the summation of the loads of all the MHs in the network. For the sake of convenience, we shall henceforth designate the normalized revenue value of an MH as R and the normalized load of an MH as L. Moreover, for every MH, we normalize further to make $(R + L) = 1$. This can be easily performed by multiplying the value of $(R + L)$ of every MH by a real number k, whose value may differ across MHs. For the sake of convenience, we shall use $R + L = 1$ to reflect the above normalization.

Computation of λ for different cases follows.

Case 1: Revenue and load are both assigned equal weight: AReL computes the function $f = R \times L = R \times (1 - R)$ (since $R + L = 1$). Using the product rule of differentiation, f is differentiated w.r.t. R.

$$df/dR = R(-1) + 1 - R = 1 - 2R$$

To find f's maximum value, the derivative (df/dR) is set to zero. Hence, $1 - 2R = 0 \Rightarrow R = 1/2$. Since $R + L = 1$, we obtain $L = 1/2$. Thus, f's maximum value occurs when $R = L = 1/2$. Hence, AReL computes $\lambda = (R + L)$ in this case.

Case 2: Revenue is assigned higher weight than load: AReL computes a function $f = R^2 \times L = R^2 \times (1 - R)$. Using the product rule , we differentiate f w.r.t. R.

$$df/dR = R^2(-1) + 2R(1 - R) = R(-3R + 2)$$

To find f's maximum value, we set the derivative (df/dR) to zero. Since $R \neq 0$, $-3R + 2 = 0 \Rightarrow R = (2/3)$. Hence, $L = 1/3$. Thus, $R = 2L$, hence $\lambda = 2R + L$ in this case.

Case 3: Revenue is assigned lower weight than load: AReL computes a function $f == R \times L^2 = R \times (1 - R)^2$. Similar to Case 2, by differentiating f w.r.t. R, we obtain $L = 2R$, hence $\lambda = R + 2L$ for this case.

Now we shall describe the AReL algorithm. The key features of AReL follow. (a) It is executed by SP and it requires every MH to *periodically* send its load status, its revenue value ω, available memory space status, read-write logs of its own data items and read-logs of the replicas stored at itself. SP uses these logs to compute the price ρ of each data item. (b) It prefers higher-priced data items since these items have higher importance to the network. (c) It checks MH schedules to identify MHs that are likely to access a given data item d. Among these MHs, it replicates d at the MH M, which has maximum number of k-hop neighbours that are likely to access d or at one of M's k-hop neighbours. This facilitates serving multiple MH requests in a fair manner. (d) It performs replica allocation based on constraints such as available memory space at the MHs, load status of the MHs and the probability of availability of the MHs. (SP is able to estimate probability of availability of an MH by maintaining availability information about the MH over a period of time.) (e) It considers those data items, whose prices exceed the average price ψ, as candidates for replication. ψ equals ($(1/N_d) \sum_{k=1}^{N_d} \rho_j$), where N_d is the total number of data items in the M-P2P network and ρ_j is the price of the j^{th} data item.

Figure 1 depicts the AReL algorithm. Observe how AReL allocates replicas starting from the highest-priced data item, thus preferring higher-priced data items. Line 15 of Figure 1 indicates that AReL allocates replicas of relatively higher-priced data items to MHs with low values of λ. This facilitates both revenue-balance and load-balance since low value of λ implies relatively lower MH revenue and lower MH load. Revenue-balancing becomes a necessity because gross imbalance across revenues of MHs may result in undesirably low revenues for some of the MHs, which could potentially prevent them from obtaining their desired services (i.e., issuing access requests) from the network. This would decrease the overall participation in the M-P2P network, which is undesirable. On the other hand, load-balancing is required to optimize query response times by preventing queries from incurring long waiting times in the job queues of overloaded MHs. Hence, AReL does not replicate at overloaded MHs since such MHs would not be able to provide good service due to their large job queues.

In Line 17 of Figure 1, the price ρ of a data item d is recomputed after replica allocation since ρ depends upon the number of existing replicas. If there is still some available memory space at some MHs after the

Algorithm *AReL*

D: List of data items that are candidates for replication
(1) Sort data items in D in descending order of ρ
(2) for each data item d in D
(3) FLAG_R = FALSE
(4) Check MH schedules to identify list L_A of MHs likely to access d
(5) Compute the number ϕ of M's k-hop neighbours for each MH in L_A
(6) Sort the MHs in descending order of ϕ into a list L_B
(7) while (FLAG_R != TRUE)
(8) for each MH M in L_B
(9) Add M and its k-hop neighbours to a list L_C
(10) Delete MHs with inadequate memory space from L_C
(11) Delete MHs with low remaining energy from L_C
(12) Delete overloaded MHs from L_C
(13) Delete MHs with low availability from L_C
(14) if (L_C is not an empty list)
(15) From L_C, select the MH with lowest λ for storing d's replica
(16) Delete all entries from L_A, L_B and L_C
(17) Recompute ρ of d
/* ρ depends on the number of replicas */
(18) FLAG_R = TRUE
(19) **break**
end

Figure 1: AReL replica allocation algorithm

AReL algorithm has been executed for all the candidate data items for replication, the algorithm is executed multiple times until none of the MHs have adequate memory space for storing replicas.

5 Performance Evaluation

MHs move according to the *Random waypoint model* [2] within a 1000 metre ×1000 metre area. The entire network contains 200 data items that are uniformly distributed among 50 MHs i.e., each MH owns 4 data items. Each query is a request for one of the data items. *Periodically*, every TP seconds, SP decides whether to perform replica allocation. Network topology does *not* change significantly during replica allocation since it requires only a few seconds [11]. In all our experiments, 20 queries/second are issued in the network, the number of queries directed to each MH being determined by the Zipf distribution. Communication range of all MHs (except SP) is a circle of 100 metre radius. Table 2 summarizes the performance study parameters.

Performance metrics are **average response time** (**ART**) of a query, **data availability** (**DA**) and **traffic** (**TR**) for replica allocation. ART = $(1/N_Q)\sum_{i=1}^{N_Q}(T_f - T_i)$, where T_i is the time of query issuing, T_f is time of the query result reaching the query issuing MH, and N_Q is the total number of queries. DA = (N_S/N_Q)*100, N_S being the number of queries that were answered successfully. Each query has a 'hops-to-live' i.e., queries that are not answered within n hops are dropped. Preliminary results of our experiments indicated that $n = 4$ is a reasonable value for our application scenarios. We define TR as the total hop-count for replica allocation during the experiment. As reference, we adapt the **E-DCG+** approach [11] to our scenario. E-DCG+ is executed at every replica allocation period. As a baseline, we compare with an approach **NoRep**, which does not perform replica allocation. Incidentally, AReL showed comparable performance for different values of λ, hence we present here the results of AReL corresponding to equal weight for both revenue and load (i.e., $\lambda = R + L$, as discussed in Section 4).

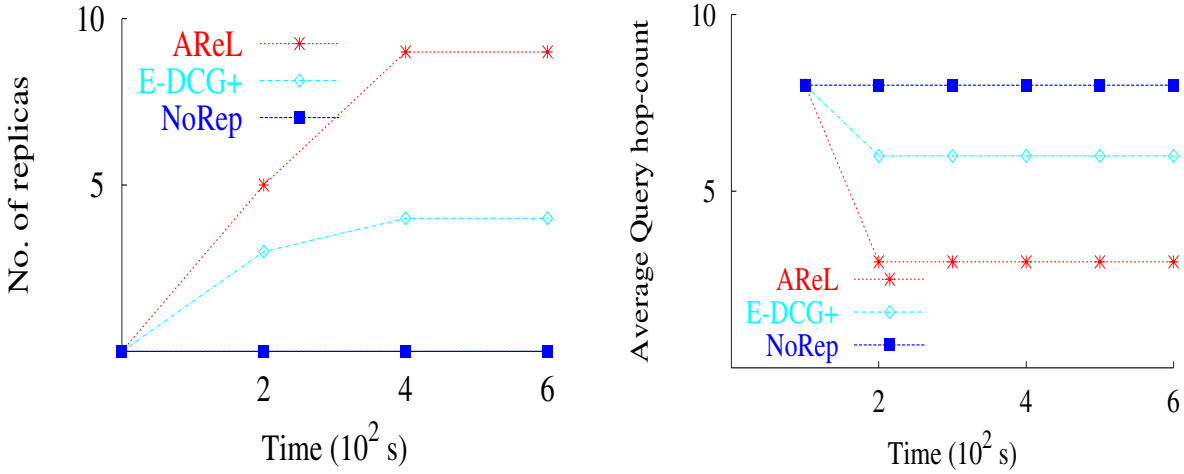
Parameter	Default value
No. of MHs (N_{MH})	50
Zipf factor (ZF)	0.9
Allocation period TP (10^2 s)	2
Queries/second	20
Bandwidth between MHs	28 Kbps to 100 Kbps
Probability of MH availability	50% to 85%
MH service capacity	1 to 5 service capacity units
Size of a data item	50 Kb to 350 Kb
Memory space of each MH	1 MB to 1.5 MB
Speed of an MH	1 metre/s to 10 metres/s
Size of message headers	220 bytes

Table 2: Performance Study Parameters

Effect of fair replica allocation

We conducted an experiment to observe the number of replicas created by AReL and E-DCG+ for a single 'hot' data item d over a period of time. This data item was selected randomly from the top 10% hottest data items. Figure 2a depicts the results. For both AReL and E-DCG+, the number of replicas increases over time since more MHs start participating in providing service. However, the number of replicas does not increase indefinitely over time and eventually plateaus after some time due to competition among replicas for MH memory space. AReL creates more replicas than E-DCG+ because AReL's economic model encourages more MHs to participate in storing replicas, hence total available memory space and bandwidth are more for AReL than for E-DCG+.

Figure 2b indicates the average number of hop-counts required for querying the same data item d during different periods of time. These results were averaged over a total of 1200 queries. Initially, before replica allocation had been performed, all three approaches required comparable number of hops for querying d. After replica allocation has been performed, AReL requires lower number of hops than E-DCG+ to answer queries on d since AReL creates more



(a) No. of replicas for a data item

(b) Average hop-count of queries

Figure 2: Effect of fair replica allocation

replicas for d as discussed for Figure 2a. More replicas generally decrease the querying hop-count since it increases the likelihood of queries being answered within lower number of hops. E-DCG+ requires lower number of querying hop-counts than NoRep essentially due to replication.

Performance of AReL

We conducted a simulation experiment using default values of the parameters in Table 2. Figure 3 depicts the results, which can be explained partly by the explanations for Figure 2. Additionally, AReL creates larger number of replicas for many different data items depending upon data item prices. Thus, AReL would create a replica for a data item d, which is accessed by a large number of MHs, even if d's total access frequency is low, in which case E-DCG+ would not create any replica. Furthermore, AReL allocates replicas only to underloaded MHs, while it is possible for E-DCG+ to allocate replicas to overloaded MHs. The performance gap between AReL and E-DCG+ keeps increasing over time due to to more MH participation in case of AReL. Incidentally, during replica allocation, E-DCG+ requires every MH to broadcast its RWR values to every MH, thereby incurring $O(N_{MH}^2)$ messages, while AReL requires each MH to send only one message to SP and SP to send a message to each MH, thus incurring $O(N_{MH})$ messages, which explains the results in Figure 3c.

6 Conclusion

We have proposed EcoRep, which is a novel economic model for dynamic replica allocation in M-P2P networks, the aim being to improve data availability. EcoRep ensures fair replica allocation, while discouraging free-riding. EcoRep's replica allocation algorithm AReL considers relative importance of data items in terms of price, load and energy issues, user mobility patterns, replica consistency and number of neighbours of a given user. Our performance evaluation indicates that EcoRep is indeed effective in improving data availability in terms of reduced query response times and higher data availability in M-P2P networks. We plan to extend this work by considering replication in the presence of obstacles in M-P2P networks.

References

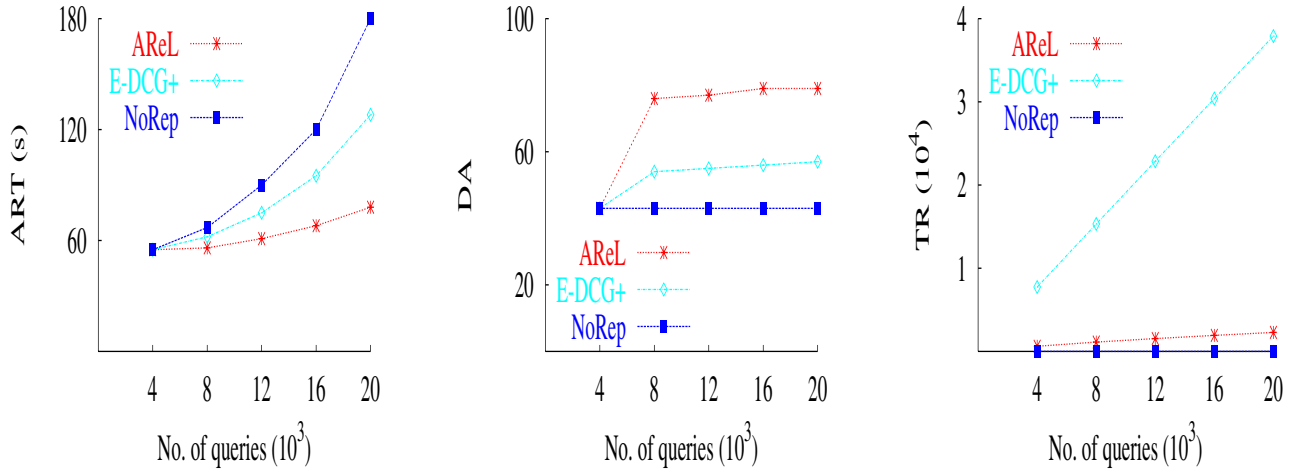
[1] E. Adar and B. A. Huberman. Free riding on Gnutella. *Proc. First Monday*, 5(10), 2000.

[2] J. Broch, D.A. Maltz, D.B. Johnson, Y.C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocol. *Proc. MOBICOM*, 1998.

[3] A. Datta, M. Hauswirth, and K. Aberer. Updates in highly unreliable replicated peer-to-peer systems. *Proc. ICDCS*, 2003.

[4] D.F. Ferguson, C. Nikolaou, and Y. Yemini. An economy for managing replicated data in autonomous decentralized systems. *Proc. International Symposium in Autonomous Decentralized Systems*, pages 367–375, 1993.

[5] D.F. Ferguson, Y. Yemini, and C. Nikolaou. Microeconomic algorithms for load balancing in distributed computer systems. *Proc. ICDCS*, pages 491–499, 1988.



(a) Average Query Response Time (b) Data Availability (c) Replica Allocation Traffic

Figure 3: Performance of AReL

[6] P. Golle, K.L. Brown, and I. Mironov. Incentives for sharing in peer-to-peer networks. *Proc. Electronic Commerce*, 2001.

[7] C. Grothoff. An excess-based economic model for resource allocation in peer-to-peer networks. *Proc. Wirtschaftsinformatik*, 2003.

[8] R. Guy, P. Reiher, D. Ratner, M. Gunter, W. Ma, and G. Popek. Rumor: Mobile data access through optimistic peer-to-peer replication. *Proc. ER Workshops*, 1998.

[9] M. Ham and G. Agha. Ara: A robust audit to prevent free-riding in p2p networks. *Proc. P2P*, pages 125–132, 2005.

[10] T. Hara. Effective replica allocation in ad hoc networks for improving data accessibility. *Proc. IEEE INFOCOM*, 2001.

[11] T. Hara and S. K. Madria. Dynamic data replication using aperiodic updates in mobile ad-hoc networks. *Proc. DASFAA*, 2004.

[12] S. Kamvar, M. Schlosser, and H. Garcia-Molina. Incentives for combatting freeriding on p2p networks. *Proc. Euro-Par*, 2003.

[13] B. Kemme and G. Alonso. A new approach to developing and implementing eager database replication protocols. *Proc. ACM TODS*, 25(3), 2000.

[14] J. F. Kurose and R. Simha. A microeconomic approach to optimal resource allocation in distributed computer systems. *Proc. IEEE Trans. Computers*, 38(5):705–717, 1989.

[15] S. Lee., R. Sherwood, and B. Bhattacharjee. Cooperative peer groups in Nice. *Proc. INFOCOM*, 2003.

[16] N. Liebau, V. Darlagiannis, O. Heckmann, and R. Steinmetz. Asymmetric incentives in peer-to-peer systems. *Proc. AMCIS*, 2005.

[17] E. Pitoura and B. Bhargava. Maintaining consistency of data in mobile distributed environments. *Proc. ICDCS*, 1995.

[18] L. Ramaswamy and L. Liu. Free riding: A new challenge to P2P file sharing systems. *Proc. HICSS*, page 220, 2003.

[19] D. Ratner, P.L. Reiher, G.J. Popek, and G.H. Kuenning. Replication requirements in mobile environments. *Proc. Mobile Networks and Applications*, 6(6), 2001.

[20] B. Richard, D. Nioclais, and D. Chalon. Clique: A transparent, peer-to-peer replicated file system. *Proc. MDM*, 2003.

[21] S. Saroiu, P.K. Gummadi, and S.D. Gribbler. A measurement study of P2P file sharing systems. *Proc. MMCN*, 2002.

[22] G. Tsuchida, T. Okino, T. Mizuno, and S. Ishihara. Evaluation of a replication method for data associated with location in mobile ad hoc networks. *Proc. ICMU*, 2005.

[23] O. Wolfson, S. Jajodia, and Y. Huang. An adaptive data replication algorithm. *Proc. ACM TODS*, 22(4):255–314, June 1997.

[24] B. Yang and H. Garcia-Molina. Designing a super-peer network. *Proc. ICDE*, 2003.