EcoRep: An Economic Model for Efficient Dynamic Replication in 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 making data replication a necessity. This work proposes EcoRep, a novel economic model for dynamic replica allocation in M-P2P networks. EcoRep performs replica allocation based on a data item's relative importance, which is quantified by the data item's *price* in terms of a virtual currency. The price of a data item depends on its access frequency, the number of users who accessed it, the number of its existing replicas, its (replica) consistency and the average response time required for accessing it. EcoRep ensures fair replica allocation by considering the origin of queries for data items. EcoRep requires a query issuing user to pay the *price* of his requested data item to the user serving his request. This discourages free-riding and encourages user participation by providing an incentive for users to become service-providers. EcoRep also considers other issues such as load, energy and network topology as replication criteria. Our performance study indicates 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 [11] strongly motivate M-P2P network applications. Some application scenarios, which would facilitate mobile users in sharing information with each other *on-the-fly* in a P2P manner, are as follows:

- Tourists in different sight-seeing buses could share touristic information (e.g., images of castles) with each other. (*inter-vehicular communication*)
- Customers in a shopping mall could share information about the cheapest available 'Levis' jeans. They could also exchange shopping catalogues 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 even request the museum's path information from other visitors as in virtual reality applications.

Notably, such P2P interactions among mobile users are generally not freely supported by existing mobile communication infrastructures. The notion of replica consistency in this work is based on the time of the latest update. For example, for the shopping mall application, a copy of a shopping catalogue, which was updated one hour ago, is considered to be more consistent than one that had been updated two days ago. Note that our application scenarios do not require absolute replica consistency [3, 17], hence we consider tolerance to weaker replica consistency.

Interestingly, every sight-seeing bus generally has a tour guide for facilitating tourists. A tour guide could facilitate data sharing *locally* among the tourists within his own bus. Tour guides in different buses could also collaborate with each other to enable effective data sharing across tourists in different buses, thereby supporting *remote querying*. Similarly, shopping malls usually have administrators such as information guides, who typically move within particular regions of the mall. Administrators in different parts of the mall could interact with each other to facilitate data sharing among mobile customers across different regions of the mall. Furthermore, museums almost always have administrators who supervise particular sections of the museum and provide information to visitors. Such museum administrators generally have the capability to interact with each other. As we shall see in this work, such administrators can be used to facilitate replication in our application scenarios.

Incidentally, data availability in M-P2P networks is typically lower than in fixed networks due to frequent network partitioning arising from user movement and/or users switching 'on'/'off' their mobile devices. (Data availability is less than 20% even in a wired environment [20].) To improve M-P2P data availability, several replication schemes [9, 21] 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 [9]). Hence, these schemes would regard d as 'hot' and create several replicas of d, even if a

International Conference on Management of Data
COMAD 2006, New Delhi, India, December 14–16, 2006

single MH M issues a very large number of (read) queries for d. This is inherently unfair since it favours M, thus these schemes are not able to serve the requests of multiple MHs in a *fair* manner. Moreover, existing schemes do not combat **free-riding** [10, 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. EcoRep performs replica allocation based on a data item's relative importance, which is quantified by the data item's *price* in terms of a virtual currency. The price of a data item depends on its access frequency, the number of users who accessed it, the number of its existing replicas, its (replica) consistency and the average response time required for accessing it. EcoRep requires a query issuing user to pay the *price* of his requested data item to the user serving his request. Hence, a user has to provide service to the network to earn enough currency to be able to issue his own queries.

The main contributions of EcoRep are two-fold:

1. It ensures fair replica allocation by considering the origin of queries for data items to determine their relative importance to the network as a whole.

2. It discourages free-riding and provides an incentive for users to become service-providers by virtue of its economic nature.

EcoRep also considers other issues such as load, energy and network topology as replication criteria. Incidentally, the load of the MH M serving an access request for a data item d can influence the price of d. If M is relatively underloaded, it would be able to provide better service since it can serve the request quickly, thereby implying lower query response time and consequently, higher data item price. However, if M is overloaded, prices of data items accessed at M would decrease due to increased query response times. Notably, our primary focus is **economy-based fair replica allocation**, a pleasant side-effect of which is that of **discouraging free-riding** at no additional cost. EcoRep can also be regarded as an **incentive scheme**, which encourages increased user participation in M-P2P networks essentially due to its economic nature.

To manage replication efficiently, EcoRep deploys a super-peer architecture [23]. The super-peer (SP) is an MH, which generally moves within the region and which has maximum energy and processing capacity at a given time. In the context of our application scenarios, the tour guides in the sight-seeing buses, the administrators in the shopping mall and the museum administrators would act as SP. We assume that the SPs in different regions (in case of the shopping mall and museum applications) or in different buses (for the inter-vehicular application) have the capability to collaborate with each other. This is in consonance with real-world scenarios e.g., administrators in different parts of a shopping mall are generally equipped with mobile devices to interact with each other.

Our architecture facilitates replica allocation and avoids broadcast storm during replica allocation. Intuitively, storing replicas arbitrarily at any MH could adversely impact many MHs due to high communication overheads between MHs, unnecessary delays and querying failures. Hence, replication should be performed carefully based on MH characteristics (e.g., load, energy) as well as network topology. Thus, some *regional knowledge* becomes a necessity. As we shall see later, each MH *periodically* sends a message to SP with information such as its current location, load, available memory space and energy status, thereby equipping SP with such regional knowledge. This helps SP to better manage replication. In contrast, for an architecture without any SP (e.g., the E-DCG+ approach [9]), each MH needs to broadcast its status to all other MHs to make each other aware of the regional status, thereby creating an undesirable broadcast storm during replica allocation. Our architecture avoids such broadcast storm due to SP.

Our architecture does not require local queries to pass via SP, thereby preserving P2P autonomy. This is possible because every MH periodically sends the list of data items/replicas stored at itself to SP, and SP broadcasts this information to all MHs. Thus, every MH has adequate information not only for redirecting queries locally, but also for distinguishing between local and remote queries. An MH issues a remote query by sending the query to the SP in its region; SP stamps the query with its unique identifier and forwards it to other SPs.

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 costs for replication. 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 discusses existing works, while Section 3 discusses the EcoRep economic model. Section 4 details 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, 13, 7, 4, 22] primarily for resource allocation in distributed systems. A competitive micro-economic auction-based bidding model with support for load-balancing has been proposed in [5], while the work in [13] 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. Incidentally, none of these works address the unique issues associated with the M-P2P environment such as frequent network partitioning and mobile resource constraints. Our recent works [16, 15] address issues associated with frequent network partitioning and mobile resource constraints, but they

do not consider economic issues. Recently, the work in [22] has proposed an economic model for M-P2P environments. However, the proposal in [22] aims at data dissemination with the aim of reaching as many peers as possible, while we consider on-demand services. Furthermore, it does not consider replication.

Works concerning free-riding include [6, 10, 14, 18]. The works in [6, 10] propose incentive schemes to combat free-riding. The work in [18] discuss utility functions to capture user contributions, while trust issues are examined in [14]. However, these works are completely orthogonal to replication issues associated with free-riding.

The work in [12] proposes a suite of replication protocols for maintaining data consistency and transactional semantics of centralized systems. P2P replication systems include Clique [19] 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 proposal in [9] present 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** [9] 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 proposal in [9] does not consider economic issues such as incentives and prices of data items. Furthermore, the architecture in [9] is not suitable for our application scenarios since it does not consider 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 dynamic replica allocation in M-P2P networks.

In EcoRep, each data item 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 a data item d stored at an MH M_S, it pays the *price* ρ (in terms of a *virtual currency*) of d to M_S since M_S serves its request. Thus, M_I spends the amount ρ, while M_S earns ρ. 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. EcoRep provides an incentive for MHs to provide service to the network so that they can earn more revenue in order to be able to issue their own queries. An MH can provide service to the network either by storing data items/replicas that are accessed by other MHs or by forwarding messages e.g., queries, query results (i.e., relay functions). The amount Rel earned by an MH each time it performs a relay function is constant, irrespective of the message being forwarded. (Randomness ensures that each MH will generally have to forward comparable number of messages.) We stipulate that the value of Rel should be lower than the price of the lowest-priced data item to ensure that storage of data items and replicas is assigned higher priority than relay functions.

Each MH maintains recent read-write logs (including timestamps) of its own data items and the read-logs of the replicas stored at itself. As we shall see shortly, each MH uses this information for computing the prices of the data items and replicas stored at itself. We shall explain the details concerning the access statistics maintenance at each MH in Section 4. Furthermore, 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 define the **load** $L_{i,j}$ of an MH M_i at time t_j as the job queue length of M_i normalized w.r.t. available bandwidth and service capacity to address heterogeneity.

$$L_{i,j} \;=\; J_{i,t_j} \;\div\; (\; \sigma_i \times B_i \;) \tag{1}$$

where J_{i,t_j} represents the job queue length of M_i at time t_j. Since job queue length is a function of time, load is also a function of time. 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. σ_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. We have used the minimum service capacity among all the MHs as 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 B_{min} = 56 Kbps.

Factors influencing the price of a data item

When an MH M_I accesses a data item d that is stored at an MH M_S, the price ρ, which is spent by M_I and earned by M_S, depends upon the following factors:

- **Access frequency of d during recent time periods:** Higher access frequency of d implies greater importance of d, hence d's price should increase with increasing access frequency.
- **The number of MHs served by d during recent time periods:** The larger the number of MHs served by d, the greater is d's importance to the network as a whole. Hence, d's price should increase as it serves requests originating from more MHs. Thus, unlike existing works, for two data items with equal access frequencies, the price would be higher for the data item that is accessed by a larger number of MHs.
- **The number of existing replicas of d:** The lesser the number of replicas of d in the network, the more difficult it is to obtain d. Hence, d's price should increase as d's number of replicas decreases and vice versa. This is in consonance with the economic principles, which dictate higher prices for rarer items.
- **Consistency of the replicas which answered queries on d:** In some sense, replica consistency may be regarded as being akin to the quality of results. Hence,

higher replica consistency should imply higher price and vice versa. Notably, replica consistency issues do not arise for queries answered by an MH's own data items since such data items are always consistent.

- **Query response time:** Response time τ for a query Q pertaining to d reflects the quality of service provided to the query issuing MH M_I by the query serving MH M_S, hence shorter response times should command higher price. τ equals (T_W + T_D + T_{delay}), where T_W is the waiting time spent by Q in M_S's job queue, T_D is the download time for d, and T_{delay} is the path delay. 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. T_{delay} depends on the delays introduced in the path of the query results during the hops from M_S to M_I due to bandwidth variations, in case M_S and M_I are not 1-hop neighbours. Thus, T_{delay} = ($\sum_{i=1}^{n_{hop}}(R_{Size}/B_i)$), where n_{hop} is the number of 'hops' between M_S and M_I, R_{Size} is the size of the query result and B_i is the bandwidth between the MHs at the i^{th} hop. T_{delay} considers the connectivity of an MH answering an access request.

Quantifying the relative importance of a data item by its price

Based on the factors discussed above, an MH M_S, which stores a data item d, computes d's price in two steps. First, M_S computes ρ_{rec}, which is the price of d based on the accesses to d at M_S during the most recent replica allocation period. Second, M_S uses moving averages of ρ_{rec} over a fixed number of replica allocation periods to compute the price ρ of d. This is necessary because 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). Table 1 summarizes the notations, which we shall henceforth use in this paper.

Notation	Significance
d	A given data item
M_S	MH that stores a given data item d and serves requests for d
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_{M_S}	Probability of availability of MH M_S
N_R	Number of existing replicas of d
J_{M_S,t_j}	Job queue length at query serving MH M_S during time t_j
σ_{M_S}	Service capacity of query serving MH M_S

Table 1: Summary of notations

Computation of ρ_{rec}: M_S first sorts the MHs in *descending* order of their access frequencies for d during the most recent replica 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, M_S computes ρ_{rec} of d as follows:

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

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. C_i reflects the (replica) consistency with which d was answered for queries by MH i. C_i = 1 for queries answered by M_S's own data items since such queries are always answered with absolute 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. Each MH 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 $T_{\epsilon,C}$, which is replicated at each MH and is the same for each MH.

BA_i equals (T_B/N_a), where T_B is the sum of all the bandwidths that M_S allocated to MH i over each of the times when MH i accessed d at M_S. N_a is the total number of access requests that MH i made for d. Observe how the probability of availability PA_{M_S} of MH M_S influences the price. Equation 2 takes heterogeneity in service capacities of MHs into account by normalizing job queue length of the query serving MH M_S w.r.t. its service capacity. Furthermore, the total 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. To put things into perspective, now let us consider some examples for the computation of ρ_{rec} with N_{MH} = 50. For simplicity, assume that $J_{M_S,t_j}/\sigma_{M_S}$ = 1 (at a given time t_j) for the following examples:

1. *A single MH makes 100 accesses to d; for all i, C_i = 1, BA_i = 50 units; PA_{M_S} = 0.5, N_R = 1:* Here, n_i = 100 when i =1 (and 0 otherwise), hence ρ_{rec} = 25.

2. *A single MH makes 100 accesses to d; for all i, C_i = 1, BA_i = 50 units; PA_{M_S} = 0.5, N_R = 5:* Here, n_i = 100 when i =1 (and 0 otherwise), hence ρ_{rec} = 8.33.

3. *A single MH makes 100 accesses to d; for all i, C_i = 1, BA_i = 10 units; PA_{M_S} = 0.5, N_R = 5:* Here, n_i = 100 when i =1 (and 0 otherwise), hence ρ_{rec} = 1.67.

4. *4 MHs make 25 accesses each to d; for all i, C_i = 1, BA_i = 50 units; PA_{M_S} = 0.5, N_R = 1:* Here, n_i = 25 for i = 1 to 4 (and 0 otherwise), hence ρ_{rec} = 62.5.

5. *50 MHs make 2 accesses each to d; for all i, C_i = 1, BA_i = 50 units; PA_{M_S} = 0.5, N_R = 1:* Here, n_i = 2 for i = 1 to N_{MH}, hence ρ_{rec} = 637.5.

6. *50 MHs make 2 accesses each to d; for all i, C_i = 1, BA_i = 50 units; PA_{M_S} = 1, N_R = 1:* Here, n_i = 2 for i = 1 to N_{MH}, hence ρ_{rec} =1275.

7. *50 MHs make 2 accesses each to d; for all i, C_i = 0.25, BA_i = 50 units; PA_{M_S} = 1, N_R = 1:* Here, n_i = 2 for i = 1 to N_{MH}, hence ρ_{rec} = 318.75.

Observe from examples 1 and 2 above how ρ_{rec} decreases with increase in the number of existing replicas. From examples 2 and 3, notice how ρ_{rec} decreases when average available bandwidth for download is decreased. Across examples 1, 4 and 5, we see how ρ_{rec} increases as d serves more MHs. Observe from examples 5 and 6 how ρ_{rec} increases with increase in the probability of availability of M_S. Examples 6 and 7 indicate how ρ_{rec} decreases with decreasing replica consistency. Finally, even though the total access frequency of d is the same in all the above examples, ρ_{rec} differs in each case.

Computation of the moving average price ρ: Incidentally, ρ_{rec} may not always be able to reflect the true importance of a given data item d to the network. As a single instance, d's access frequency may have been low during the most recent allocation period, even though d could have been heavily accessed during the allocation periods prior to the most recent one. Since replica allocation is an expensive operation, the true importance of a data item should be determined meticulously to avoid unnecessary replica allocation. Hence, M_S computes the moving average price ρ of d. However, simple moving averages give the same weight to the last N replica allocation periods, hence they suffer from 'lag' i.e., they are not capable of reacting quickly to changing access patterns of data items. Given that in M-P2P networks, access patterns typically change dynamically over time, we use the Exponential Moving Average (EMA), which gives higher weights to recent access patterns relative to older access patterns, thereby reducing the 'lag'. M_S computes the price ρ of d as follows:

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

where EMA_{prev} represents the EMA that was computed for the previous replica allocation period, and N represents the number of replica allocation periods over which the moving average is computed. Observe that the EMA approach only requires M_S to store the previous allocation period's value of EMA instead of the values of price data for the entire period being averaged, thereby optimizing memory space usage of SP, which is important in case of M-P2P networks. We performed preliminary experiments to determine a suitable value for N for various scenarios in our applications. The results indicate that N = 5 is a reasonably good value for our application scenarios.

Price ρ of a data item w.r.t. a region: Recall that our application scenarios involve multiple SPs such that each SP is responsible for a specific region in space. For example, in case of the shopping mall application, a region could be the area covered by one floor of the mall. For the tourist bus application, it could be either the area of the bus, or the area of some castle, museum or garden which the tourists of a bus are currently visiting. As we shall see in Section 4, the price of a data item d should be computed w.r.t. each such region to determine the region(s) in which d should be replicated. Hence, each MH *periodically* sends the prices of the data items and replicas stored at itself to the corresponding SP S in its region. S collates such price information from all the MHs in its region to compute the price ρ_{Region} of a data item d w.r.t. the region covered by itself. ρ_{Region} is the sum of the prices of d at all the MHs within S's region. Given that η is the number of MHs in the region covered by a particular SP, and ρ_i is d's price ρ at the i^{th} MH, ρ_{Region} equals ($\sum_{i=1}^{\eta}$ ρ_i).

Revenue of an MH: In EcoRep, a MH M earns virtual currency from accesses to its own data items and replicas that are stored at itself. 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. The amount E (of virtual currency) earned by M follows:

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

In the second term of Equation 4, C_i is a parameter which indicates the average consistency with which queries on the replicas (stored at M) were answered. C_i does not occur in the first term of Equation 4 as the first term concerns M's own data items, which are always absolutely consistent.

M spends its virtual currency when it accesses data items and/or replicas stored at other MHs. 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 follows:

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

where C_i has the same significance as in Equation 4. Using Equations 4 and 5, the *revenue* ω of M follows.

$$\omega = E - S \quad (6)$$

Thus, revenue of an MH is simply the difference between the amount that it earns and the amount that it spends. 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.

When an MH joins the M-P2P network, SP provides the MH with a small amount of revenue to start with so that the MH can issue a few queries. However, once this initial rev-

enue is exhausted, the MH will eventually have to provide service to the network, otherwise it will not be able to issue any further queries. However, it is possible for a malicious MH to join the network, obtain the initial amount of revenue from SP and issue a few queries. Then it can leave the network and re-join the network after some time. To mitigate the effect of such free-riding, the unique device identifier of a mobile device could be used to uniquely identify an MH; SP could maintain a log containing MH identifiers and a summary of MH behaviours during recent time intervals. Understandably, our aim is to encourage MH participation in the network, hence we do not want to exclude MHs by deploying policies that are too strict against free-riding.

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. We first explore the interaction between revenue and load of an MH.

Interaction between revenue and load of an MH

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. Thus, M's revenue could 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 could be low in spite of its high load. Even if M earns high amounts of virtual currency, 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. 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 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 a function f as follows.

$$f = R \times L = R \times (1 - R)$$

Using the product rule of differentiation, we differentiate f 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 λ= (R + L) in this case.

Case 2: Revenue is assigned higher weight than load: AReL computes the following function f.

$$f = R^2 \times L = R^2 \times (1 - R)$$

Now we differentiate f w.r.t. R.

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

To find the maximum value of f, we set the derivative (dp/dR) to zero. Since $R \neq 0$, $-3R + 2 = 0 \Rightarrow R$ = (2/3). Hence, L = 1/3. Thus, the maximum value of f occurs when R =2L, hence λ = 2R + L.

Case 3: Revenue is assigned lower weight than load: In this case, we compute f as follows.

$$f = R \times L^2 = R \times (1 - R)^2$$

Differentiating f w.r.t. R, we obtain

$$\begin{aligned} df/dR &= R(2)(1 - R)(-1) + (1)(1 - R)^2 \\ &= (1 - R)(1 - 3R) \end{aligned}$$

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

Notation	Significance
$data_{id}$	Identifier of a given data item
M_I	Identifier of the query issuing MH
M_S	Identifier of the MH serving the query request
SP_I	Identifier of the SP in whose region M_I is currently moving
t	Time of query issue
D_{int}	List summarizing internal accesses to M_S's own data items at M_S
D_{ext}	List summarizing external access to M_S's own data items at M_S
R_{int}	List summarizing internal accesses to replicas at M_S
R_{ext}	List summarizing external access to replicas at M_S

Table 2: Notations for access statistics maintenance

Maintenance of access statistics at each MH

Table 2 summarizes the notations for access statistics maintenance at each MH. Each MH M_S distinguishes between accesses made to its own data items from within the re-

gion of its corresponding SP (i.e., *internal accesses*) and accesses to its own data items from MHs that are moving within the region of other SPs (i.e., *external accesses*). D_{int} and D_{ext} are lists in which M_S summarizes the internal accesses and external accesses respectively to its own data items. D_{int} guides M_S in selecting its own data items that should be replicated within the region of its corresponding SP, while D_{ext} facilitates M_S in determining its own data items that should be replicated at regions covered by other SPs. Using Table 2, each entry in D_{int} is of the form ($data_{id}$, M_I), while entries in D_{ext} are of the form ($data_{id}$, SP_I, M_I). M_S uses the entry of SP_I in D_{ext} to decide the SP, within whose region the given data item should be replicated. (Every query issuing MH includes the identifier of its corresponding SP in its query.)

M_S also differentiates between its own data items and the replicas that are stored at itself. Thus, in Table 2, R_{int} and R_{ext} are lists in which M_S summarizes the internal accesses and external accesses respectively to the replicas stored at itself. Using Table 2, each entry in R_{int} is of the form ($data_{id}$, M_I), while entries in R_{ext} are of the form ($data_{id}$, SP_I, M_I). Notably, here $data_{id}$ refers to the identifier of the *replica* stored at M_S, while for the lists D_{int} and D_{ext}, $data_{id}$ represented the identifier of M_S's own data item. Besides this difference, the data structures of R_{int} and R_{ext} are essentially similar to that of D_{int} and D_{ext} respectively. R_{int} and R_{ext} guide M_S in computing replica prices w.r.t. different SPs. Notably, the lists D_{int}, D_{ext}, R_{int} and R_{ext} are *periodically* refreshed to reflect *recent* access statistics. This is performed by periodically deleting all the existing entries from these lists and then re-populating them with fresh information from the recent queries. Such refreshing is especially important due to the dynamic changes in access patterns in M-P2P networks.

Selection of candidate data items for replication

Using its D_{int} and R_{int} respectively, each MH computes the price of each of its items (i.e., its own data items and replicas stored at itself), which were accessed by MHs from within the region of its corresponding SP S. Since D_{int} and R_{int} summarize the *internal accesses*, these prices are w.r.t. S. Similarly, from its D_{ext} and R_{ext} respectively, each MH calculates the price of each of its items, which were accessed by MHs that are outside the region of S. In this case, MHs from the respective regions corresponding to multiple SPs may have accessed a particular item, hence the prices of data items and replicas are computed w.r.t. each SP *separately*. *Periodically*, each MH sends all these prices to to its corresponding SP S. Upon receiving these prices from all the MHs in its region, each SP S sums up the price of each item (w.r.t. each of the other SPs) from each MH within its region, thereby computing the total price of each item w.r.t. each SP.

Intuitively, internally accessed items should be replicated at MHs within a given SP S's region, while the externally accessed items need to be replicated at MHs in the regions of other SPs. Hence, when selecting candidate items for replica allocation, S distinguishes between internally accessed and externally accessed data items. For the internally accessed items, S sorts these items in descending order of their prices. S considers those 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 items and ρ_j is the price of the j^{th} item. Observe how S prefers items with relatively higher prices for replica allocation due to the higher importance of these items.

For the externally accessed items, S computes the price of each data item d w.r.t. every (external) SP from whose region at least one access was made for d. Then S creates a list $L_{Suggest}$ of these items, each entry of which is of the form ($data_{id}$, ρ, SP_{id}), where $data_{id}$ is the identifier of the item, and ρ is the price of the item w.r.t. SP_{id}, which is the identifier of a given external SP. Then S sorts the items in $L_{Suggest}$ in descending order of ρ. S considers items (of $L_{Suggest}$), whose prices exceed the threshold α, as candidates for replication. (The remaining items are deleted from $L_{Suggest}$.) α equals ($(1/N_d) \sum_{k=1}^{N_d} \rho_j$), where N_d is the total number of items accessed by external SPs, and ρ_j is the price of the j^{th} data item w.r.t a given external SP.

Observe the similarities in determining the candidate data items for replication for internally and externally accessed data items, the difference being that the case for externally accessed data items is more complicated due to the computation of data item prices w.r.t. multiple SPs. Furthermore, S does not participate in allocating replicas for the selected candidate items in $L_{Suggest}$. Instead, for each candidate item, S just sends a message to the *relevant* external SP, which will perform the actual replica allocation at some MH within its region. Given $data_{id}$, the relevant external SP is the corresponding SP_{id} in $L_{Suggest}$. Just as S suggests external SPs to replicate items, the external SPs also suggest S to replicate items that have been accessed at these external SPs by the MHs of S's region. We shall henceforth refer to the list of items, for which S needs to allocate replicas, as Rep. Thus, Rep comprises two types of items: (a) items that are stored at the MHs within its own region R (i.e., *internal items*) (b) items which are stored at MHs outside R (i.e., *external items*). External items are recommended to S by the other (external) SPs.

The AReL algorithm

Each SP performs replica allocation within the region that it covers. *Periodically*, each MH sends its current (x,y) coordinates, its revenue value ω, the prices of items stored at itself, its load, energy and available memory space status to the corresponding SP in its region. SP collates the (x,y) coordinate information of all the MHs in its region to estimate the network topology during the time of replica allocation.

Figure 1 depicts the AREL replication algorithm, which is executed by a given SP S for allocating replicas at MHs within its own region. The list Rep in Figure 1 comprises items that are candidates for replica allocation by S. Line 1 of Figure 1 indicates that AREL allocates replicas starting

from the highest-priced data item, thus preferring higher-priced items since these items have higher importance.

Algorithm *AReL*

Rep: List of data items that are candidates for replication
(1) Sort data items in Rep in descending order of ρ
(2) for each data item d in Rep
(3) FLAG_R = FALSE
(4) Identify list L_A of MHs which have recently accessed d
(5) for each MH M in L_A
(6) Compute the no. ϕ of M's 1-hop neighbours that accessed d
(7) Sort the MHs in descending order of ϕ into a list L_B
(8) while (FLAG_R != TRUE)
(9) for each MH M in L_B
(10) Add M and its 1-hop neighbours to a list L_C
(11) Delete MHs with inadequate memory space from L_C
(12) Delete MHs with low remaining energy from L_C
(13) Delete overloaded MHs from L_C
(14) Delete MHs with low probability of availability from L_C
(15) if (L_C is not an empty list)
(16) From L_C, select the MH with lowest λ for storing the replica of d
(17) Delete all entries from L_A, L_B and L_C
(18) Recompute ρ of d /* ρ depends on number of replicas */
(19) FLAG_R = TRUE
(20) **break**
end

Figure 1: AReL replica allocation algorithm

Lines 4-7 indicate that S identifies MHs that have recently accessed a given data item d. Among these MHs, it tries to replicate d starting from the MH M, which has maximum number of 1-hop neighbours that have accessed d or at one of M's 1-hop neighbours. (S knows the 1-hop neighbours of M due to its knowledge of network topology.) This facilitates bringing d nearer to the origin of most of the requests for d, which is in consonance with EcoRep's objective of serving the maximum possible number of MHs in a fair manner. Line 10 shows that we also consider the 1-hop neighbours of M since M may not have adequate available memory space and/or remaining energy.

From Lines 11-14 observe how that S performs replica allocation based on constraints such as available memory space at the MHs, energy, 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.) In particular, AREL avoids replica allocation at overloaded MHs (see Line 13) primarily because such MHs would not be able to provide good service due to their large job queues, which would force queries to incur long waiting times and consequently, higher response times.

Line 16 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 of revenues across 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.

In Line 18 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 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

Our experiments consider *five* different regions. Each region has 50 MHs and 1 SP. MHs in each region move according to the *Random waypoint model* [2] within the region. Each region's area is 1000 metre $\times$1000 metre. The *Random Waypoint Model* is appropriate for our application scenarios, which consider random movement of users. As a single instance, tourists in a shopping mall generally move randomly i.e., they do not follow any specific mobility pattern. SPs move within their respective regions. SPs are able to communicate with each other. Each region 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 either a local data item or a remote one. We had performed experiments with different percentages of remote and local queries, the results indicating increasing query response times with increasing percentage of remote queries. Since the results exhibited similar trends, here we present the results of experiments in which 60% of the queries were remote ones, while the other 40% were local queries.

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 [9]. 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 3 summarizes the parameters used in our performance evaluation.

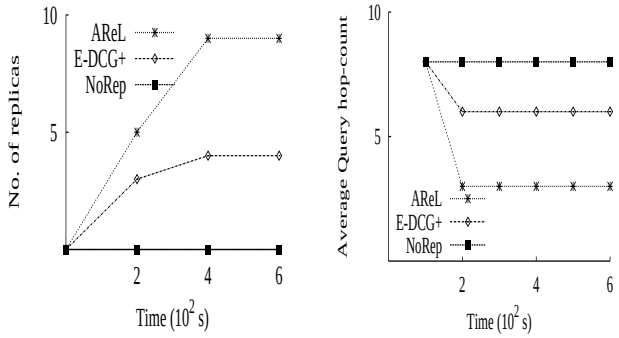
Performance metrics are **average response time** (**ART**) of a query, **data availability** (**DA**) and **traffic** (**TR**) for replica allocation. We compute ART as follows:

$$ART = (1/N_Q) \sum_{i=1}^{N_Q} (T_f - T_i) \quad (7)$$

where T_i is the time of query issuing, T_f is time of the

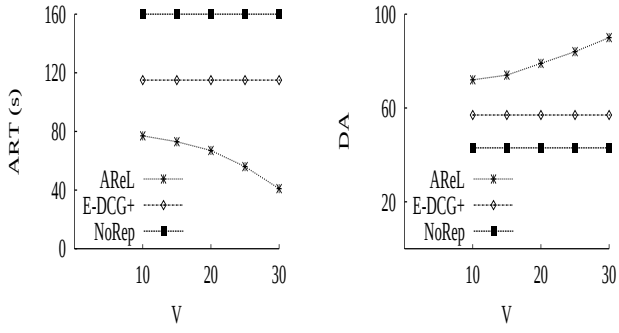
Parameter	Default value	Variations
No. of MHs (N_{MH})	50	10, 20, 30, 40
Zipf factor (ZF)	0.9	0.1, 0.3, 0.5, 0.7
Allocation period TP (10^2 s)	2	1, 3, 4, 5, 6
Queries/second	20	
Bandwidth between MHs	28 Kbps to 100 Kbps	
Probability of MH availability	50% to 85%	
MH service capacity	1 to 5 service capacity 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 3: Performance Study Parameters



(a) No. of replicas for a data item (b) Average hop-count of queries

Figure 2: Effect of fair replica allocation



(a) Average Query Response Time (b) Data Availability

Figure 3: Effect of revenue threshold

query result reaching the query issuing MH, and N_Q is the total number of queries. DA is computed as follows:

$$DA \;=\; (\; N_S/N_Q \;) \;\times\; 100 \qquad (8)$$

where N_S is the number of queries that were answered successfully and N_Q is the total number of queries. 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 course of the experiment.

As reference, we adapt the **E-DCG+** approach [9] (discussed in Section 2) to our scenario. Notably, the E-DCG+ approach is the closest to our scheme since it addresses *dynamic* replica allocation in M-P2P environments. Moreover, none of the proposals on economic issues addresses *dynamic* replica allocation in M-P2P networks. E-DCG+ is executed at every replica allocation period. As a baseline, we also compare our proposed AReL algorithm 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., λ = R + L, as discussed in Section 4).

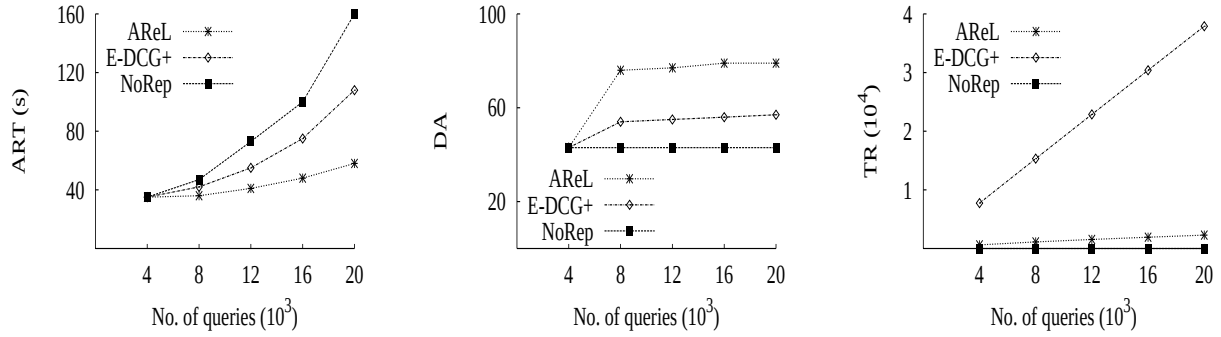
Effect of fair replica allocation

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 of d 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 of d than E-DCG+ because AReL's economic model encourages more MHs to participate in storing replicas, hence total available memory space and available 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 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. Observe that E-DCG+ requires lower number of querying hop-counts than NoRep essentially due to replication.

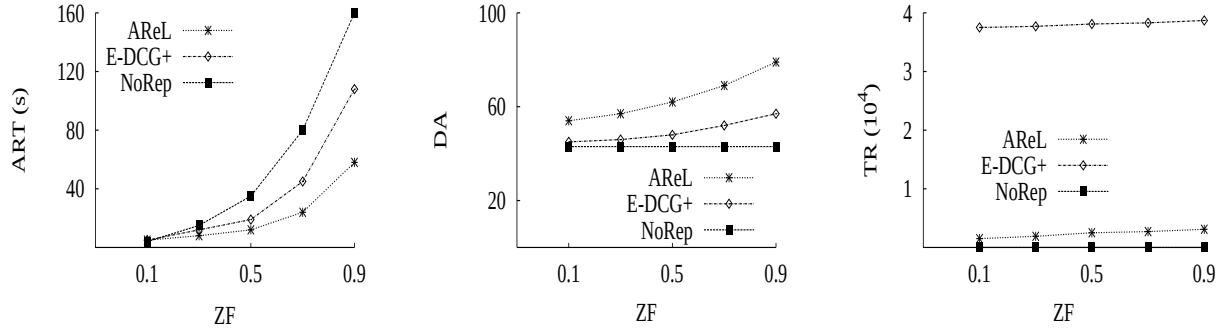
Effect of variations in the number of MHs above threshold revenue

Threshold revenue γ is defined as the ratio of the total revenue of the system to the total number of MHs. In other words, γ is the average revenue in the system. Figure 3 depicts the results concerning the effect of variations in the number of MHs above γ. In Figure 3, the 'V' on the x-axis refers to the 'number of MHs above the threshold revenue γ'. The results indicate that when the revenue of more MHs exceed γ, ART decreases and data availability increases.



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

Figure 4: Performance of AReL



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

Figure 5: Effect of variations in the workload skew

This is due to more MHs participating in providing service as MH revenues increase, thereby implying more opportunities for replication, more memory space and available bandwidth, and multiple paths for locating a replica. E-DCG+ and NoRep show relatively constant ART and DA as these approaches are independent of revenue. AReL outperforms the reference approaches due to the reasons explained for Figure 2 i.e., larger number of replicas owing to more MHs providing service in case of AReL.

Performance of AReL

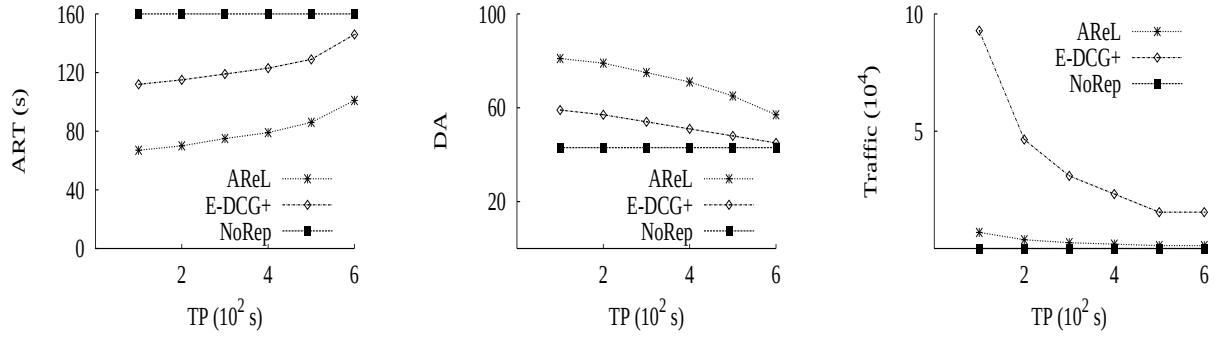
We conducted an experiment using default values of the parameters in Table 3. Figure 4 depicts the results, which can be explained partly by the reasons discussed for Figures 2 and 3. Additionally, AReL creates larger number of replicas for many different data items depending upon data item prices. Thus, 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 4c.

Effect of variations in the workload skew

Figure 5 depicts the results when the zipf factor (ZF) is varied. For high ZF values (i.e., high skew), AReL and E-DCG+ perform better in terms of ART and DA due to more replica allocations in response to load-imbalance conditions. The performance gap between AReL and E-DCG+ decreases with decreasing skew since lowly skewed workloads do not require replica allocations. AReL outperforms E-DCG+ and NoRep due to the reasons explained for Figures 2, 3 and 4. The explanation for Figure 5c is essentially similar to that of Figure 4c.

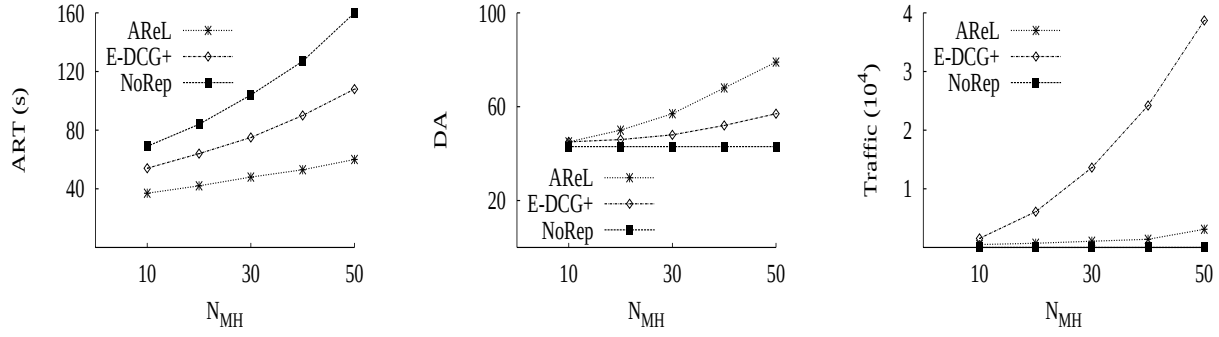
Effect of variations in the replica allocation period

Recall that every TP seconds, SP decides whether to allocate replicas. Figure 6 depicts the results of varying TP. At lower values of TP, more number of replica allocation periods occur, hence load imbalances are corrected quickly in response to changing access patterns, thereby improv-



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

Figure 6: Effect of variations in the replica allocation period TP



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

Figure 7: Effect of variations in the number of MHs

ing ART and DA for both AReL and E-DCG+. As TP increases, load imbalances are corrected less frequently, hence performance degrades for both AReL and E-DCG+. For NoRep, ART and DA remain relatively constant because they depend only upon probability of MH availability. The explanation for the results in Figure 6c is essentially similar to that of Figure 4c. In particular, replica allocation traffic decreases dramatically with increasing TP due to decreased number of replica allocation periods.

Effect of variations in the number of MHs

To test AReL's scalability, we varied the number N_{MH} of MHs, keeping the number of queries proportional to N_{MH}. Figure 7 depicts the results. At high values of N_{MH}, AReL outperforms E-DCG+ due to the reasons explained for Figures 2, 3 and 4. As N_{MH} decreases, the performance gap decreases due to limited replication opportunities. Replica allocation traffic for E-DCG+ dramatically decreases with decreasing N_{MH} due to reduced broadcast traffic.

Discouraging free-riding

Figure 8 depicts the percentage of service providers for AReL in the M-P2P network during different time periods. An MH is regarded as a service provider during a time period T if it stores a data item/replica that is accessed at least once during T. Initially, the MHs have little revenue, but as more queries are issued, MH revenues increase, thereby increasing MH participation levels upto the point where the majority of the MHs are providing service to the network essentially due to AReL's economic replication model.

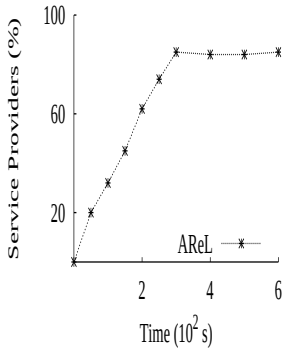


Figure 8: Discouraging free-riding

6 Conclusion

We have proposed EcoRep, which is a novel economic dynamic replica allocation model for improving the typically limited data availability of M-P2P networks. EcoRep allocates replicas based on a data item's relative importance, which is quantified by the data item's *price*. EcoRep ensures fair replica allocation by considering the origin of queries for data items. EcoRep requires a query issuing user to pay the price of his requested data item to the user serving his request, which discourages free-riding. EcoRep also considers load, energy and network topology as replication criteria. Our performance study demonstrates that EcoRep is indeed effective in improving query response times and data availability 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*, 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.

[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] T. Hara and S.K. Madria. Data replication for improving data accessibility in ad hoc networks. *IEEE Transactions on Mobile Computing*, 5(11):1515–1532, Nov 2006.

[10] S. Kamvar, M. Schlosser, and H. Garcia-Molina. Incentives for combatting free-riding on P2P networks. *Proc. Euro-Par*, 2003.

[11] Kazaa. http://www.kazaa.com/.

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

[13] 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.

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

[15] A. Mondal, S.K. Madria, and M. Kitsuregawa. CADRE: A collaborative replica allocation and deallocation approach for mobile-p2p networks. *To appear in Proc. IDEAS*, 2006.

[16] A. Mondal, S.K. Madria, and M. Kitsuregawa. CLEAR: An efficient context and location-based dynamic replication scheme for mobile-p2p networks. *Proc. DEXA*, pages 399–408, 2006.

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

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

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

[22] O. Wolfson, B. Xu, and A.P. Sistla. An economic model for resource exchange in mobile peer to peer networks. *Proc. SSDBM*, 2004.

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