EPFL Technical Report IC/2003/16

On Optimal Update Policies and Cluster Sizes for 2-Tier Distributed Systems

Prasenjit Dey*, Anwitaman Datta†
{Prasenjit.Dey, Anwitaman.Datta}@epfl.ch
School of Computer and Communication Sciences
École Polytechnique Fédérale de Lausanne (EPFL)
CH-1015 Lausanne, Switzerland

Abstract

We try to analyze a generic model for 2-tier distributed systems, exploring the possibility of optimal cluster sizes from an information management perspective, such that the overall cost for updating and searching information may be minimized by adopting a judiciously lazy updating policy. We do not assume either centralized coordination or decentralization, and since it is an initial work, we only advocate the existence of such optimal policies rather than how such policies may be discovered by the system participants. We put our work in perspective using two examples from diverse domains of distributed systems, namely the wireless cellular networks, which are based on centralized coordination and peer-to-peer systems using clusters (like Kazaa).

Keywords: Cellular Wireless Networks, Peer-to-Peer Systems, Clustering, Lazy Updates, Optimization.

1 Introduction

Distributed systems often have hierarchical structure, with some participants having a specialized role, thus coordinating other participants. We investigate the case of 2-tier hierarchy. Our investigation focuses on determining optimal cluster sizes that could be coordinated by a cluster head, vis-a-vis the update cost incurred by participants and the search cost in such distributed systems, and also explore the possibility of using a lazy update policy such that the overall cumulative cost of updating and searching is optimized. While 2-tier systems are a very special case of the general hierarchical systems, the results nonetheless provide an insight about the possibility of optimal cluster sizes, and such 2-tier systems are indeed often found in some real systems. We put our work in perspective with respect to two such instances, one that of peer-to-peer systems like Kazaa and the other that of mobile wireless communication systems, where the base stations can be thought to be cluster heads. While our work discusses distributed systems, it does not assume decentralization or otherwise, and hence the results are valid for either case. The work is a stepping stone for further work, and thus we identify the possibility of having an optimal update policy and optimal clusters. Discussion of schemes to achieve the optimal update policy or formation of optimal clusters, either by centralized coordination or self-organization in decentralized systems, is beyond the scope of the present work.

Moreover there are several assumptions, particularly of the cost functions, which are only indicative and do not necessarily reflect realistic systems. Thus the results provided in this paper are meant to give a qualitative insight rather than quantitative results.

*Laboratory of Mobile Communication: This work is partly funded by Nokia Research Center, Helsinki.
†Distributed Information Systems Laboratory: The work presented in this paper was supported (in part) by the National Competence Center in Research on Mobile Information and Communication Systems (NCCR-MICS), a center supported by the Swiss National Science Foundation under grant number 5005-67322.

The paper is organized as follows. In Section 2 we introduce 2-tier systems and its motivation in the context of state-of-the-art cellular and P2P systems. We make our case that there exists optimal designs for such 2-tier distributed systems by analyzing a generic model in Section 3. We present our results in Section 4 and draw conclusions in Section 5.

2 2-tier systems

While 2-tier systems are the simplest among the hierarchical systems, some of the popular and well known systems can be modeled as such. These include traditional cellular networks to more recent peer-to-peer systems like Kazaa.

2.1 Cellular wireless networks

The cellular wireless networks are perhaps the most obvious example of a 2-tier system. The *users* are divided into groups called *cells* by their geographic location and each cell is coordinated by a group head called a *base station* as shown in Figure 1. The base station controls and knows whatever happens in a cell based on updates from the user. The users can communicate with each other only through their respective base stations. The base stations in turn communicate with each other, not visible to the users, to provide a communication service. This 2-tier structure of the cellular networks will continue to exist even with the rapid developments in mobile ad-hoc networks because ad-hocness can be interesting in a local area but not at a global scale. The cellular 2-tier system also makes sense because [4],

- Cellular structure of networks is necessary to maximize throughput and optimize battery performance.
- The network has to be hierarchy based to avoid unnecessary updates or find operations through the whole network to track a particular user.
- Memory intensive tasks like directory search operations and transfer of trunk route data will rely on less mobile and reliable nodes higher in the hierarchy and moreover there may not be enough number of nodes between a source and destination to complete a communication session in an ad-hoc fashion.

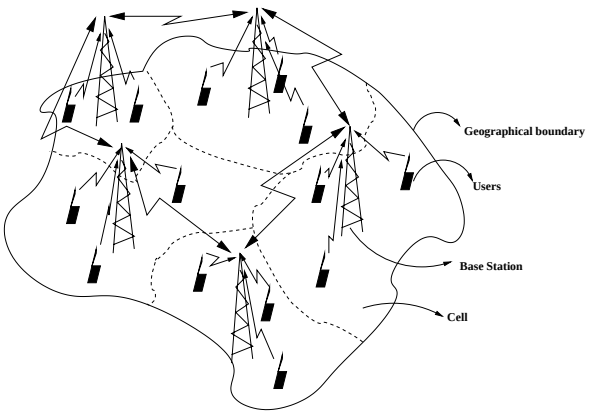

Figure 1: Cellular Architecture

2.2 P2P systems

Napster is one of the most widely known P2P systems. It used a central index for all the content available at all the end-points (peers). Thus for searching information, the central index was required to be queried. Similarly for sharing information the central server was required to be informed.

Napster, shown in Figure 2(a) suffered from the typical drawbacks of centralized solutions, namely single point of failure as well as scalability problems.

Gnutella like systems on the other hand are in the other extreme, where all peers maintain their own information, however for searching, the whole network needs to be flooded with a query. Thus every peer asks all other peers that it is in touch with, in order to perform a search. The system topology is shown in Figure 2(b). Gnutella suffered from the typical drawbacks of unstructured decentralization, namely high overhead and latency for message flooding. Thus to say, flooding based systems are inefficient and expensive.

Chronologically and logically systems like Kazaa evolved, which use a hybrid of both Napster and Gnutella architectures, thus forming small clusters, where the cluster heads centrally store all information about their members in a way similar to Napster, while such cluster heads communicate among themselves in a more ad-hoc way similar to Gnutella. While the actual system topology for Kazaa is not known, it may be speculated to have a topology as has been shown in Figure 2(c).

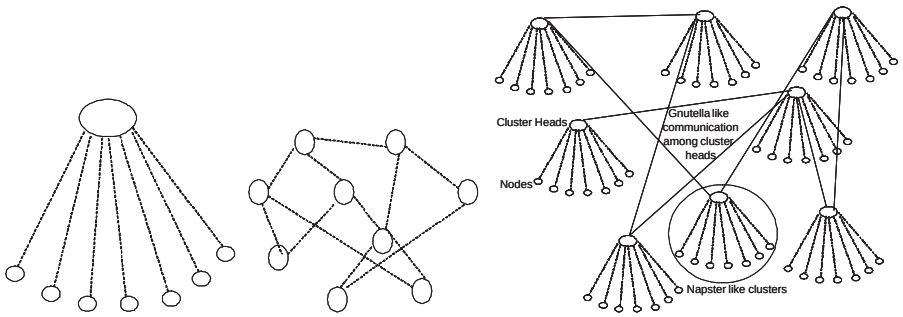

Figure 2: (a)Napster, (b)Gnutella and (c)Kazaa(speculated) system architectures

2.3 Optimal clusters

For a given peer population of n and the number of cluster heads or clusters s, Napster may be thought to be a special case of Kazaa with only one cluster head ($s = 1$) and n members within a single cluster, while Gnutella at the other extreme, with n cluster heads ($s = n$) and single member per cluster. Performance wise, it has been observed that Kazaa is much more efficient than Gnutella, and indeed compares with Napster, on the other hand, Kazaa has the fault tolerant and resilient properties of Gnutella networks.

In mobile cellular networks also its not easy to allocate an appropriate load of users on a base station. Huge load of users on a particular base station may result in an enormous processing load on that base station and hence result in poor quality of service or dropping of calls whereas another base station nearby may be operating at below its capacity. It would make sense to optimally assign user load on the base stations i.e finding optimal cluster sizes, so that the overall operating cost is minimized.

Such interesting observations motivate the quest for optimal cluster sizes. In order to do so, we need to understand and quantify the various tradeoffs. As an initial step we use a simple cost model for searching and updating in such systems.

2.4 Search and lazy updates in 2-tier systems

Whenever information is stored in a distributed manner, there are two costs to be incurred, that of searching and that of updating. It is thus desirable to adopt a lazy approach for making updates [7, 5], so that instead of updating every time there is a change locally at a participant, such changes are intimated to the cluster head only if a certain number of updates are locally accumulated. The drawback with such a lazy policy is that if some other participant is searching for the information, then the information obtained from the cluster head is stale, and thus there is a penalty to be paid,

either because a stale information is used, or else in order to obtain the latest information from the original participant who had failed to inform the cluster head.

In the 2-tier systems that we discussed, search cost will actually have two constituents.

Inter-cluster search: If the information being searched is not available within the cluster where the search is initiated, the cluster heads will have to communicate among themselves in order to locate the cluster where the information resides. This inter-cluster communication can possibly be done in a deterministic way, in which case the lower bound for communication is $s-1$, while if it is done in a randomized manner [6, 2], the cost will be $O(s\log(s))$. This is not the lower bound, but is easily achievable, as has been argued in [6]. Under the assumption of uniform data distribution as well as query distribution, inter-cluster searches will occur with a probability of $\frac{s-1}{s}$.

Intra-cluster search: Either if the search is initiated for information resident in the same cluster, or if a search request comes from another cluster head, an intra-cluster search is needed unless the cluster head has up-to-date information. The query may actually be answered by the cluster head provided it has the information, or else it will have to ask its members. The cost incurred in asking members will depend on the cluster population $\frac{n}{s}$.

Thus we see that inter-cluster searches are more expensive with growing number of clusters, and also more probable $(1-\frac{1}{s})$. This implies that smaller number of clusters (small s) is desirable. A fact affirmed by Napster's performance. But this does not account for either load balancing nor the susceptibility of a single point of failure.

On the other hand, intra-cluster searches become cheap with smaller cluster size (thus larger s). These define the tradeoffs, which we will study to determine optimal cluster sizes.

One may argue that intra-cluster searches are not required if the cluster heads are kept up-to-date. It is a well established and accepted fact though that lazy updates are cost efficient for many kinds of information and application domains [5, 3]. If updates are not lazily made, that would mean frequently initiating updates, which is not desirable. For example, in wireless systems, uplink is a scarce and expensive resource in terms of bandwidth and battery life of the handset. Or in the case of P2P systems, these may lead to a lot of unnecessary network traffic because of packet overheads, which prompts to make a more judicious, that is to say, a lazy updating scheme. Another equally important justification of lazy updates is that otherwise the cluster heads may be overwhelmed with update requests.

This determines the tradeoff so that for any cluster size, we need to find an optimal updating policy in order to reduce the cumulative search and update cost over the whole network.

Having thus defined our tradeoffs, the question that we want to answer is whether there is an optimal updating policy and an optimal cluster size, given overall population and search and update characteristics.

We analytically model 2-tier systems in our quest for optimal clusters and lazy update policy, as will be elaborated next.

3 Optimizing the cumulative cost of updates and searches in 2-tier distributed systems

We consider a situation where n nodes are distributed into s *clusters* of equal size. Each cluster is headed by a node called the *cluster head* and rest of the nodes in that cluster are called *cluster members*. Each node's attributes experience changes at discrete time k. All nodes belonging to a cluster report their information of any change of attributes or *updates* to the cluster heads. The changes can occur with the nodes remaining within the given cluster or the change may result in a node moving to a new cluster in which case it must update its information to the new cluster head. We assume reliable communication among cluster heads either in a deterministic or randomized fashion. For searches, the best situation can be if nodes *always* update their changes to their cluster heads in which case the information at the cluster heads is always current. Any *search* for information about

a node's attributes will consume less resource and the information found will be "fresh". Whereas if the nodes *never* update, the search for that information would consume more resource and would return a "stale" information. This means there is a tradeoff between the update and search, which implicitly depends also on the number of clusters. Hence there exists an optimal number of clusters and update policy for each node which optimizes the total resource cost.

3.1 System Model

The events and their notation with their statistic is as follows:

Cluster change: Random variable $v_k \in \{0, 1\}$ for a node denotes the event that the node moves from one cluster to another at time k. It takes a value 1 if the node changes cluster and 0 otherwise.

$$P(v_k) = \begin{cases} \beta & : \quad v_k = 1 \\ 1 - \beta & : \quad v_k = 0 \end{cases} \tag{1}$$

Local change (within a cluster): Random variable $w_k \in \{0, 1\}$ denotes the event that a change occurs in a node within a cluster at time k. It takes a value 1 if a change occurs and 0 otherwise.

$$P(w_k) = \begin{cases} \alpha & : \quad w_k = 1 \\ 1 - \alpha & : \quad w_k = 0 \end{cases} \tag{2}$$

This change can be imagined to be a change of cell by an user in the case of cellular networks or a change of file data or application availability at a node in the case of peer-to-peer networks.

Decision by a node at time k to update: $u_k \in \{0, 1\}$ denotes the decision by a node at time step k to update the changes to the cluster head. The decision to update is denoted by 1 and the decision not to update is denoted by 0.

Staleness counter or state of a node: x_k denotes the number of changes that has happened on a node since it last reported updates to its cluster head.

Search probability: At each time k, a node (or an information it holds) can be searched for with probability γ. It is possible that there are simultaneous searches for same item, in which case further optimization is possible, but we ignore such complications in the model.

3.1.1 State equation

The above events can be combined to write a state equation which gives a description of the next state at time $k + 1$ given the description at time k which is

$$x_{k+1} = [x_k(1 - u_k) + w_k](1 - v_k). \tag{3}$$

The equation above describes the phenomenon because if the user updates ($u_k = 1$) or it changes a cluster ($v_k = 1$), the state level is reset to zero. A change within a cluster ($w_k = 1$) without update results in an increase of the state by a step.

3.1.2 Cost function

The cost function comprises of two costs; the update cost and the search cost. Our search for an optimal policy is the one which minimizes the sum of the update and the search cost.

Search cost: The search cost has two components; the cost of searching inside a cluster (f_{intra}) and the cost of searching in a different cluster (f_{inter}) if not found in the given cluster. The cost of *intra* cluster search is inversely proportional to the number of clusters (more clusters means less nodes per cluster and vice versa). The search may yield a stale information which is undesirable and hence a penalty depending on the staleness of the information (x_k) about the

node or resource is imposed. Hence the intra cluster search cost is a function $f_{intra}(x_k, \frac{n}{s})$. We intend to study two kinds of this function. They are

$$f_{intra}(x_k, \frac{n}{s}) = \begin{cases} \frac{ne^{x_k}}{s} & : \text{ exponential cost} \\ \frac{nx_k}{s} & : \text{ linear cost} \end{cases} \tag{4}$$

The *inter* cluster search can result if the queried resource or node is not found by the cluster head in its cluster and hence has to contact other cluster heads to find the resource or the node. It can happen with probability $\frac{s-1}{s}$. The cost of searching $s-1$ cluster heads for the information is $\mathcal{O}(s-1)$ if the communication between the cluster heads is deterministic or it can be $\mathcal{O}(s\log(s))$ if the communication between cluster heads is randomized. Therefore the inter cluster search is a function $f_{inter}(s)$.

Hence the total expected search cost at time k is,

$$S(x_k, s) \quad = \quad \gamma C_1[f_{intra}(x_k, \frac{n}{s}) + \frac{s-1}{s} f_{inter}(s)] \tag{5}$$

where C_1 is a proportionality constant which depends on the actual system.

Update cost: The total update cost at time k is the cost due to an update in the cluster ($u_k = 1$) and the necessary update if there is a change in cluster ($v_k = 1$).

$$U_k \quad = \quad C_2 u_k(1 - v_k) + C_2 v_k \tag{6}$$

where C_2 is the cost per update. Therefore the total expected update cost is,

$$E_{v_k}[U_k] \quad = \quad C_2 u_k(1 - \beta) + C_2\beta. \tag{7}$$

Since our objective is to find an optimal policy that minimizes the total cost over time, we can write the cumulative-optimal-cost function from time k to a finite horizon [1], $J^*_k(x_k, s)$, for a given s as,

$$J^*_k(x_k, s) \quad = \quad \min_{u_k=0,1} E_{w_k,v_k}\{S(x_k, s) + U_k + J^*_{k+1}(x_{k+1}, s)\}. \tag{8}$$

The policy that minimizes the cumulative-optimal-cost function from a finite horizon to a time k is also the optimal policy at each step from time k to the finite horizon because by the *principle of optimality*, if it was not then we could have chosen any other minimizing policy at a time step and minimized the overall cost further. Thus the policy we obtain tells us the optimal thing to do at each time step.

3.2 Analysis

From (3) and (8) we can now set up a dynamic programming problem [1] with state variable x_k, control variable u_k and random disturbances w_k and v_k. The cost function can be defined both in forward or backward recursion but it turns out in our case that its easier to solve our problem by backward recursion starting from a finite horizon and a terminal value.

From (8), we have,

$$J^*_k(x_k, s) \quad = \quad \min_{u_k=0,1} \{S(x_k, s) + E_{v_k}[U_k] + E_{w_k,v_k}[J^*_{k+1}(x_{k+1}, s)]\} \tag{9}$$

Replacing (3) in $E_{w_k,v_k}[J^*_{k+1}(x_{k+1}, s)]$ and from the assumption that w_k and v_k are independent, we have,

$$E_{w_k,v_k}\{J^*_{k+1}(x_{k+1}, s)\} = \bar{\alpha}\bar{\beta}J^*_{k+1}(x_k(1-u_k), s) + \alpha\bar{\beta}J^*_{k+1}(x_k(1-u_k)+1, s) + \beta J^*_{k+1}(0, s) \tag{10}$$

where $\bar{\alpha} = 1 - \alpha$ and $\bar{\beta} = 1 - \beta$.

Replacing (7) and (10) in (9) we have,

$$J^*_k(x_k, s) = \min \begin{cases} S(x_k, s) + \beta C_2 + \tilde{J}(x_k, s) & : \quad u_k = 0 \\ S(x_k, s) + C_2 + \tilde{J}(0, s) & : \quad u_k = 1 \end{cases} \tag{11}$$

where $\tilde{J}(x_k, s) = \beta J^*_{k+1}(0, s) + \alpha\bar{\beta} J^*_{k+1}(x_k + 1, s) + \bar{\alpha}\bar{\beta} J^*_{k+1}(x_k, s)$. From the above equation we see that at each time k, the values of x_k can clearly be divided into two regions where if x_k is lower than some threshold value t, the policy is not to update ($u_k = 0$) and if x_k is above t we decide to update ($u_k = 1$). Since $J^*_k(x_k, s)$ is an increasing function of x_k at time k, (11) will always lead to a threshold kind of a decision rule.

We also see from (11) that the threshold t is given by the solution of the equation where the cost due to decision $u_k = 0$ is equal to the cost due to the decision $u_k = 1$. This is equal to,

$$\beta C_2 + \tilde{J}(x_k, s) \quad = \quad C_2 + \tilde{J}(0, s)$$

or

$$\tilde{J}(x_k, s) - \tilde{J}(0, s) \quad = \quad \bar{\beta} C_2 \tag{12}$$

If we solve (11) numerically as a finite horizon problem, we have the optimal decision regions at each time as shown in Figure 3. Numerical solutions show that asymptotically the threshold always reaches a steady state value. However we do not have an analytical proof of the asymptotic behavior of the threshold.

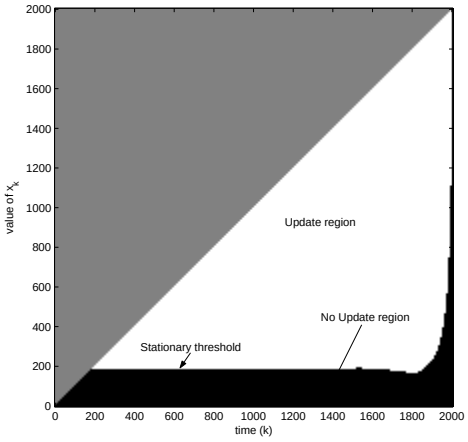

Figure 3: Existence of a stationary threshold

Assuming the steady state nature of the threshold, we can solve for the steady state values of (11) and (12) as shown in Appendix A, and we have the steady state threshold $\tilde{t}$ which is given by,

$$\tilde{t}(s) \quad = \quad \max\left\{ t \quad \middle| \quad S'(t, s) - \delta^{t+1} S'(t, s) - \frac{\beta\delta}{\alpha\bar{\beta}} \sum_{j=1}^{t} \delta^j S'(j, s) \leq \beta C_2, \quad t \in \{1, 2, ...\} \right\} \tag{13}$$

where $S'(x, s) = \alpha S(x + 1, s) + \bar{\alpha} S(x, s) - \alpha S(1, s) - \bar{\alpha} S(0, s)$, $\delta = \frac{\alpha\bar{\beta}}{1-\bar{\alpha}\bar{\beta}}$ and $s \in \{1, 2, \cdots, n\}$.

Once we have the optimal-stationary-policy thereshold function $\tilde{t}(s)$ from (13), we assume that it is used in the network. When the value of x_k is below $\tilde{t}(s)$, we do not update and therefore we do not incur any update cost but we incur the search cost penalty $S(x_k, s)$. But if x_k is above $\tilde{t}(s)$ we update, which from (3) resets x_k to zero. Thus we incur a minimum expected cost, $MinCost(s)$, for a given s, which is equal to,

$$MinCost(s) \quad = \quad E_{\{x_k | x_k \leq \tilde{t}(s)\}}[S(x_k, s)] \tag{14}$$

This $MinCost(s)$ tells us that given a cluster size $\frac{n}{s}$, we have an optimal threshold $\tilde{t}(s)$, where, if x_k is below or equal to $\tilde{t}(s)$, we do not update and if x_k is above $\tilde{t}(s)$, we decide to update, and thus incur the minimum cost $MinCost(s)$.

Since the state transition of x_k in our problem is independent of any x_k, at any given time, all possible values of x_k are equally likely. Therefore the expectation in (14) becomes an average,

$$MinCost(s) \quad = \quad \frac{1}{\tilde{t}(s) + 1} \sum_{x_k=0}^{\tilde{t}(s)} S(x_k, s)$$

$$= \frac{1}{\tilde{t}(s)+1}\sum_{x=0}^{\tilde{t}(s)} S(x,s) \tag{15}$$

Now the optimal number of clusters is the one which incurs the minimum of this object function $MinCost(s)$. Therefore, the optimal number of clusters s^* is given by,

$$s^* = \left\{ s \mid MinCost(s) = \min_{m\in\{1,2,\cdots,n\}} MinCost(m) \right\} \tag{16}$$

Once we have the optimal number of clusters s^*, we can use the function $\tilde{t}(s)$ found from (13) to find the optimal threshold t^* for this optimal number of clusters.

Thus we have an optimal update policy given by,

$$u_k = \begin{cases} 0 & : \quad x_k \leq t^* \\ 1 & : \quad x_k > t^* \end{cases} \tag{17}$$

and an optimal cluster size $\frac{n}{s^*}$ for a given number of nodes n.

4 Result

To give an example of our framework, we take some reasonable values of the parameters α and β. We assume $\alpha = 0.5$ which means that there is a high rate of change activity at the nodes. We assume that the event that a node changes cluster is relatively rare which is why we choose $\beta = 0.005$. This makes sense in the context of both mobile cellular networks and typical peer-to-peer systems. For example in case of a cellular network a user is likely to reamin around its area of residence or place of work. In the case of a peer-to-peer system, the nodes change cluster only if they do not get the desired quality of service or a node goes offline and then comes up again. Both of these events are relatively rare as compared to other events at the nodes. We choose some arbitrary values for γ and n as well as the constants $C1$ and $C2$. The actual values would depend on the real system characteristics.

We find the optimal thresholds and cluster sizes for different cases of intra-cluster cost penalty for lazy update and inter-cluster cost of search propagation. The expressions for steady state threshold for various cases of intra-cluster cost penalty for lazy update and inter-cluster cost of search propagation are shown in Appendix B, using the general expression in (13). Using the expressions for the different cases in Appendix B, the plot of $MinCost(s)$ against s is shown in Figure 4 as below.

(a) Linear-Cost penalty and Randomized search propagation.

(b) Exponential-Cost penalty and Randomized search propagation.

(c) Linear-Cost penalty and Deterministic search propagation.

(d) Exponential-Cost penalty and Deterministic search propagation.

From this plot we find the optimal s^* which gives the minimum cost. Once we have s^* we reuse (13) to find the optimal threshold t^* for the various cases.

When penalty for stale information is exponential, the threshold for lazy updates is much lower than when the penalty is linear, as expected. We also see that the number of clusters, when the inter-cluster search is propagated in a randomized manner, is significantly smaller than when search is (can be) propagated deterministically. Thus, depending on the feasibility issues, like the infrastructural requirements or the (im)possibility of deterministic communication among cluster heads, system designer can choose appropriate policies to develop an efficient 2-tier distributed system.

We also infer that the cumulative cost can indeed be minimized for each of the communication policies (for search propagation) and for different penalties for stale information at cluster heads on account of lazy updates, provided judicious threshold for laziness is chosen. The clustering itself can be done optimally in order to globally reduce the cost over all possibilities of cluster sizes.

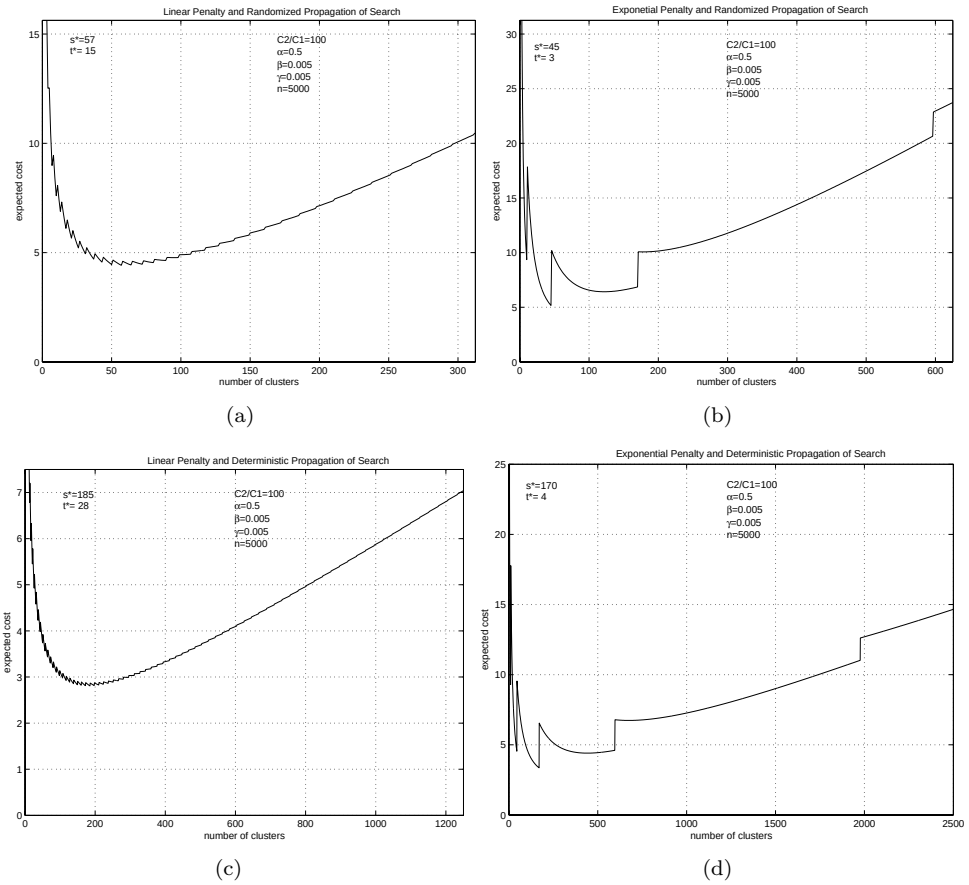

Figure 4: Plots of minimum cost and their corresponding cluster size and threshold.

5 Conclusion

Our analytical model of 2-tier distributed systems adds qualitative understanding as to why a system like Kazaa enjoys the good features of both centralized systems like Napster or totally decentralized system like Gnutella. Similar conclusion about performance and resilience can be made about cellular networks as well. While in our initial work we do not address the issue of forming optimal clusters or achieving optimal thresholding policies, we however show that such optimal choices can be made. The quest for searching such solution is the next logical step, thus determining the future direction of our research.

Previously, in the context of efficiency and reliability of mobile ad-hoc routing, clustering had been advocated [8]. Efficient information management can be a complementary reason for clustering in mobile ad-hoc networks and our present work may be extended to that end.

6 Acknowledgment

We gratefully acknowledge our discussions with Prof. Emre Telatar for some parts in the analysis.

Appendix

A Proof of (13)

From (12) we have that the threshold t_k at time k, is the solution to the equation,

$$\tilde{J}(x_k, s) - \tilde{J}(0, s) \quad = \quad \bar{\beta} C_2$$

From (11) and (12),

$$\beta J^*_{k+1}(0,s) + \alpha\bar{\beta} J^*_{k+1}(x_k+1,s) + \bar{\alpha}\bar{\beta} J^*_{k+1}(x_k,s) - \beta J^*_{k+1}(0,s) - \alpha\bar{\beta} J^*_{k+1}(1,s) - \bar{\alpha}\bar{\beta} J^*_{k+1}(0,s) \quad = \quad \bar{\beta} C_2$$

or

$$\alpha J^*_{k+1}(x_k+1,s) + \bar{\alpha} J^*_{k+1}(x_k,s) - \alpha J^*_{k+1}(1,s) - \bar{\alpha} J^*_{k+1}(0,s) \quad = \quad C_2. \tag{18}$$

We can remove the index of x_k and write a new function at time k as,

$$J'_k(x,s) \quad = \quad \alpha J^*_{k+1}(x+1,s) + \bar{\alpha} J^*_{k+1}(x,s) - \alpha J^*_{k+1}(1,s) - \bar{\alpha} J^*_{k+1}(0,s) \tag{19}$$

Therefore the threshold at time k is given by the solution of the equation,

$$J'_k(x,s) \quad = \quad C_2. \tag{20}$$

The steady state form of $J'_k(x,s)$ will give the solution of the steady state threshold. To find the steady state form of $J'_k(x,s)$, we have to find the recursion of $J'_k(x,s)$ and then set the functions at the two time indexes to be the same i.e. $J'_{k+1}(x,s) = J'_k(x,s) = J'(x,s)$ to get the steady state expression $J'(x,s)$ which then can be used to find the steady state threshold.

Towards that end, from (11) we have,

$$J^*_{k+1}(x+1,s) = S(x+1,s) + \beta J^*_{k+2}(0,s) + \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(x+2,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(x+1,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \tag{21}$$

$$J^*_{k+1}(x,s) = S(x,s) + \beta J^*_{k+2}(0,s) + \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(x+1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(x,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \tag{22}$$

$$J^*_{k+1}(1,s) = S(1,s) + \beta J^*_{k+2}(0,s) + \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(2,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(1,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \tag{23}$$

$$J^*_{k+1}(0,s) = S(0,s) + \beta J^*_{k+2}(0,s) + \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \tag{24}$$

From (19), (21), (22), (23) and (24) we have,

$$\begin{aligned}
J'_k(x,s) \quad &= \quad \alpha J^*_{k+1}(x+1,s) + \bar{\alpha} J^*_{k+1}(x,s) - \alpha J^*_{k+1}(1,s) - \bar{\alpha} J^*_{k+1}(0,s) \\
&= \quad \alpha S(x+1,s) + \alpha\beta J^*_{k+2}(0,s) + \alpha \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(x+2,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(x+1,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \\
&+ \quad \bar{\alpha} S(x,s) + \bar{\alpha}\beta J^*_{k+2}(0,s) + \bar{\alpha} \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(x+1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(x,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \\
&- \quad \alpha S(1,s) - \alpha\beta J^*_{k+2}(0,s) - \alpha \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(2,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(1,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right. \\
&- \quad \bar{\alpha} S(0,s) - \bar{\alpha}\beta J^*_{k+2}(0,s) - \bar{\alpha} \min \left\{ \begin{array}{r} \beta C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \\ C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s) \end{array} \right.
\end{aligned} \tag{25}$$

Adding and subtracting $\alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s)$ and βC_2 in each of the *min* expressions and using the definition of $J'_{k+1}(x,s)$ from (19), we have,

$$\begin{aligned}
J'_k(x,s) \quad &= \quad [\alpha S(x+1,s) + \bar{\alpha} S(x,s) - \alpha S(1,s) - \bar{\alpha} S(0,s)] \\
&+ \quad \alpha[\beta C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s)] + \alpha \min \left\{ \begin{array}{r} \bar{\beta} J'_{k+1}(x+1,s) \\ \bar{\beta} C_2 \end{array} \right. \\
&+ \quad \bar{\alpha}[\beta C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s)] + \bar{\alpha} \min \left\{ \begin{array}{r} \bar{\beta} J'_{k+1}(x,s) \\ \bar{\beta} C_2 \end{array} \right. \\
&- \quad \alpha[\beta C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s)] + \alpha \min \left\{ \begin{array}{r} \bar{\beta} J'_{k+1}(1,s) \\ \bar{\beta} C_2 \end{array} \right. \\
&- \quad \bar{\alpha}[\beta C_2 + \alpha\bar{\beta} J^*_{k+2}(1,s) + \bar{\alpha}\bar{\beta} J^*_{k+2}(0,s)] + \bar{\alpha} \min \left\{ \begin{array}{r} 0 \\ \bar{\beta} C_2 \end{array} \right.
\end{aligned} \tag{26}$$

or

$$\begin{aligned} J'_k(x,s) &= S'(x,s) + \alpha \min\left\{ \begin{array}{r} \bar{\beta} J'_{k+1}(x+1,s) \\ \bar{\beta} C_2 \end{array} \right\} + \bar{\alpha} \min\left\{ \begin{array}{r} \bar{\beta} J'_{k+1}(x,s) \\ \bar{\beta} C_2 \end{array} \right\} - \alpha \min\left\{ \begin{array}{r} \bar{\beta} J'_{k+1}(1,s) \\ \bar{\beta} C_2 \end{array} \right\} \\ &= S'(x,s) + \alpha\bar{\beta} \min\left\{ \begin{array}{r} J'_{k+1}(x+1,s) \\ C_2 \end{array} \right\} + \bar{\alpha}\bar{\beta} \min\left\{ \begin{array}{r} J'_{k+1}(x,s) \\ C_2 \end{array} \right\} - \alpha\bar{\beta} \min\left\{ \begin{array}{r} J'_{k+1}(1,s) \\ C_2 \end{array} \right\} \end{aligned} \tag{27}$$

where $S'(x,s) = \alpha S(x+1,s) + \bar{\alpha} S(x,s) - \alpha S(1,s) - \bar{\alpha} S(0,s)$.

The steady state solution of $J'_k(x,s)$ is obtained by putting $J'_{k+1}(x,s) = J'_k(x,s) = J'(x,s)$. Then we have from (27),

$$J'(x,s) = S'(x,s) + \alpha\bar{\beta} \min\left\{ \begin{array}{r} J'(x+1,s) \\ C_2 \end{array} \right\} + \bar{\alpha}\bar{\beta} \min\left\{ \begin{array}{r} J'(x,s) \\ C_2 \end{array} \right\} - \alpha\bar{\beta} \min\left\{ \begin{array}{r} J'(1,s) \\ C_2 \end{array} \right\} \tag{28}$$

Let the steady state threshold be $\tilde{t}(s) = \tilde{t}$. Then replacing $x = \tilde{t}$ in (28) and using (20) and assuming $\tilde{t} > 1$, we have,

$$\begin{aligned} C_2 &= J'(\tilde{t},s) \\ &= S'(\tilde{t},s) + \alpha\bar{\beta}C_2 + \bar{\alpha}\bar{\beta}C_2 - \alpha\bar{\beta}J'(1,s) \end{aligned}$$

or

$$J'(1,s) = \frac{1}{\alpha\bar{\beta}} S'(\tilde{t},s) - \frac{\beta}{\alpha\bar{\beta}} C_2 \tag{29}$$

Now replacing $x = \tilde{t} - 1$ in (28), we have,

$$J'(\tilde{t}-1,s) = S'(\tilde{t}-1,s) + \alpha\bar{\beta}C_2 + \bar{\alpha}\bar{\beta}J'(\tilde{t}-1,s) - \alpha\bar{\beta}J'(1,s)$$

or

$$J'(\tilde{t}-1,s) = \frac{\delta}{\alpha\bar{\beta}} S'(\tilde{t}-1,s) + \delta C_2 - \delta J'(1,s) \tag{30}$$

where $\delta = \frac{\alpha\bar{\beta}}{1-\bar{\alpha}\bar{\beta}}$.

Now if $x < \tilde{t} - 1$ the recursion of (28) has a general form,

$$J'(x,s) = S'(x,s) + \alpha\bar{\beta}J'(x+1,s) + \bar{\alpha}\bar{\beta}J'(x,s) - \alpha\bar{\beta}J'(1,s)$$

or

$$J'(x,s) = \frac{\delta}{\alpha\bar{\beta}} S'(x,s) + \delta J'(x+1,s) - \delta J'(1,s) \tag{31}$$

Replacing $x = \tilde{t} - 2$ in (31), we have,

$$J'(\tilde{t}-2,s) = \frac{\delta}{\alpha\bar{\beta}} S'(\tilde{t}-2,s) + \delta J'(\tilde{t}-1,s) - \delta J'(1,s) \tag{32}$$

Replacing $x = \tilde{t} - 3$ in (31), we have,

$$\begin{aligned} J'(\tilde{t}-3,s) &= \frac{\delta}{\alpha\bar{\beta}} S'(\tilde{t}-3,s) + \delta J'(\tilde{t}-2,s) - \delta J'(1,s) \\ &= \frac{\delta}{\alpha\bar{\beta}} S'(\tilde{t}-3,s) - \delta J'(1,s) + \delta J'(\tilde{t}-2,s) \end{aligned} \tag{33}$$

Replacing (32) in (33), we have,

$$J'(\tilde{t}-3,s) = \frac{\delta}{\alpha\bar{\beta}} [S'(\tilde{t}-3,s) + \delta S'(\tilde{t}-2,s)] - J'(1,s)[\delta + \delta^2] + \delta^2 J'(\tilde{t}-1,s) \tag{34}$$

Replacing $x = \tilde{t} - 4$ in (31), we have,

$$\begin{aligned} J'(\tilde{t}-4,s) &= \frac{\delta}{\alpha\bar{\beta}}S'(\tilde{t}-4,s) + \delta J'(\tilde{t}-3,s) - \delta J'(1,s) \\ &= \frac{\delta}{\alpha\bar{\beta}}S'(\tilde{t}-4,s) - \delta J'(1,s) + \delta J'(\tilde{t}-3,s) \end{aligned} \tag{35}$$

Replacing (34) in (35), we have,

$$J'(\tilde{t}-4,s) = \frac{\delta}{\alpha\bar{\beta}}[S'(\tilde{t}-4,s) + \delta S'(\tilde{t}-3,s) + \delta^2 S'(\tilde{t}-2,s)] - J'(1,s)[\delta+\delta^2+\delta^3] + \delta^3 J'(\tilde{t}-1,s) \tag{36}$$

and so on.

Since we are only interested to find another equation for $J'(1,s)$ so that we can use (29) to solve for $\tilde{t}$, we consider the form of $J'(1,s) = J'[\tilde{t}-(\tilde{t}-1),s]$ if the recursion in (36) is followed. This is given by,

$$\begin{aligned} J'(1,s) &= J'[\tilde{t}-(\tilde{t}-1),s] \\ &= \frac{\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-2}\delta^{j-1}S'(j,s) - \delta J'(1,s)\sum_{j=1}^{\tilde{t}-2}\delta^{j-1} + \delta^{\tilde{t}-2}J'(\tilde{t}-1,s) \end{aligned} \tag{37}$$

Now replacing (30) in (37) we have,

$$J'(1,s) = \frac{\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-2}\delta^{j-1}S'(j,s) - \delta J'(1,s)\sum_{j=1}^{\tilde{t}-2}\delta^{j-1} + \delta^{\tilde{t}-2}[\frac{\delta}{\alpha\bar{\beta}}S'(\tilde{t}-1,s) + \delta C_2 - \delta J'(1,s)]$$

or

$$\frac{1-\delta^{\tilde{t}}}{1-\delta}J'(1,s) = \frac{\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-1}\delta^{j-1}S'(j,s) + \delta^{\tilde{t}-1}C_2 \tag{38}$$

Replacing (29) in (38), we have,

$$\frac{1-\delta^{\tilde{t}}}{1-\delta}\left[\frac{1}{\alpha\bar{\beta}}S'(\tilde{t},s) - \frac{\beta}{\alpha\bar{\beta}}C_2\right] = \frac{\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-1}\delta^{j-1}S'(j,s) + \delta^{\tilde{t}-1}C_2$$

or

$$\frac{1}{\beta\delta}S'(\tilde{t},s) - \frac{\delta^{\tilde{t}}}{\beta\delta}S'(\tilde{t},s) - \frac{\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-1}\delta^{j-1}S'(j,s) = \frac{1}{\delta}C_2$$

or

$$S'(\tilde{t},s) - \delta^{\tilde{t}}S'(\tilde{t},s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-1}\delta^{j}S'(j,s) = \beta C_2 \tag{39}$$

Adding and subtracting $\frac{\beta\delta}{\alpha\bar{\beta}}\delta^{\tilde{t}}S'(\tilde{t},s)$ in (39), we have,

$$S'(\tilde{t},s) + \frac{\beta\delta}{\alpha\bar{\beta}}\delta^{\tilde{t}}S'(\tilde{t},s) - \delta^{\tilde{t}}S'(\tilde{t},s) - \frac{\beta\delta}{\alpha\bar{\beta}}\delta^{\tilde{t}}S'(\tilde{t},s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}-1}\delta^{j}S'(j,s) = \beta C_2$$

or

$$S'(\tilde{t},s) + \left[\frac{\beta\delta}{\alpha\bar{\beta}} - 1\right]\delta^{\tilde{t}}S'(\tilde{t},s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}}\delta^{j}S'(j,s) = \beta C_2$$

or

$$S'(\tilde{t},s) - \delta^{\tilde{t}+1}S'(\tilde{t},s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{\tilde{t}}\delta^j S'(j,s) \quad = \quad \beta C_2 \tag{40}$$

We know that our values of x can only be integers and so should be the values of $\tilde{t}$. Now the $\tilde{t}$ found by solving (40) may not be an integer. Therefore we define our threshold as $\lfloor\tilde{t}\rfloor$. Using this definition of the stationary threshold and using (40), the stationary threshold $\tilde{t}(s)$ can be equivalently written as,

$$\tilde{t}(s) \quad = \quad \max\Big\{t \quad \Big| \quad S'(t,s) - \delta^{t+1}S'(t,s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}\delta^j S'(j,s) \leq \beta C_2, \quad t \in \{1,2,...\}\Big\} \tag{41}$$

□

B Stationary threshold expressions for various intra-cluster and inter-cluster penalties

(a) Linear-Cost penalty and Randomized search propagation

For this case, from (4) and (5), we have

$$\begin{aligned} S(x,s) &= \gamma C_1[f_{intra}(x,\frac{n}{s}) + \frac{s-1}{s}f_{inter}(s)] \\ &= \gamma C_1[\frac{nx}{s} + \frac{s-1}{s}s\log(s)] \\ &= \gamma C_1\frac{nx}{s} + \gamma C_1(s-1)\log(s) \end{aligned} \tag{42}$$

Therefore from (42) and (13),

$$\begin{aligned} S'(x,s) &= \alpha S(x+1,s) + \bar{\alpha}S(x,s) - \alpha S(1,s) - \bar{\alpha}S(0,s) \\ &= \frac{n\gamma C_1 x}{s} \end{aligned} \tag{43}$$

Now from (43) and (13),

$$\begin{aligned} \beta C_2 \geq S'(t,s) - \delta^{t+1}S'(t,s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}\delta^j S'(j,s) &= \frac{n\gamma C_1 t}{s} - \delta^{t+1}\frac{n\gamma C_1 t}{s} - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}\delta^j\frac{n\gamma C_1 j}{s} \\ &= \frac{n\gamma C_1 t}{s} - \delta^{t+1}\frac{n\gamma C_1 t}{s} - \frac{\beta\delta n\gamma C_1}{\alpha\bar{\beta}s}\sum_{j=1}^{t}j\delta^j \end{aligned}$$

or

$$\begin{aligned} \frac{s\beta C_2}{n\gamma C_1} &\geq t - t\delta^{t+1} - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}j\delta^j \\ &= t - t\delta^{t+1} - \frac{\beta\delta}{\alpha\bar{\beta}}\left[\frac{\delta-\delta^{t+1}}{(1-\delta)^2} - \frac{t\delta^{t+1}}{(1-\delta)}\right] \\ &= t - t\delta^{t+1} - \frac{\beta\delta^2}{\alpha\bar{\beta}(1-\delta)^2} + \frac{\beta\delta^{t+2}}{\alpha\bar{\beta}(1-\delta)^2} + \frac{\beta t\delta^{t+2}}{\alpha\bar{\beta}(1-\delta)} \\ &= t - t\delta^{t+1} - \frac{\alpha\bar{\beta}}{\beta} + \frac{\beta\delta^{t+2}}{\alpha\bar{\beta}(1-\delta)^2} + \frac{\beta t\delta^{t+2}}{\alpha\bar{\beta}(1-\delta)} \qquad \text{[From definition of } \delta] \end{aligned}$$

or

$$t - t\delta^{t+1} + \frac{\beta\delta^{t+2}}{\beta\delta(1-\delta)} + \frac{\beta t\delta^{t+2}}{\beta\delta} \quad \leq \quad \frac{s\beta C_2}{n\gamma C_1} + \frac{\alpha\bar{\beta}}{\beta} \qquad \text{[because } \alpha\bar{\beta}(1-\delta) = \beta\delta]$$

or

$$(1-\delta)t + \delta^{t+1} \quad \leq \quad \frac{(1-\delta)s\beta C_2}{n\gamma C_1} + \delta \tag{44}$$

Therefore, for the case of linear-cost penalty and randomized search propagation, we see from (44), that (13) becomes equal to,

$$\tilde{t}(s) \quad = \quad \max\Big\{t \quad \Big| \quad (1-\delta)t + \delta^{t+1} \leq \frac{(1-\delta)s\beta C_2}{n\gamma C_1} + \delta, \quad t \in \{1,2,...\}\Big\} \tag{45}$$

(b) Exponential-Cost penalty and Randomized search propagation

For this case, from (4) and (5), we have

$$\begin{aligned} S(x,s) &= \gamma C_1[f_{intra}(x,\frac{n}{s}) + \frac{s-1}{s}f_{inter}(s)] \\ &= \gamma C_1[\frac{ne^x}{s} + \frac{s-1}{s}s\log(s)] \\ &= \gamma C_1\frac{ne^x}{s} + \gamma C_1(s-1)\log(s) \end{aligned} \tag{46}$$

Therefore from (46) and (13),

$$\begin{aligned} S'(x,s) &= \alpha S(x+1,s) + \bar{\alpha}S(x,s) - \alpha S(1,s) - \bar{\alpha}S(0,s) \\ &= \frac{n\gamma C_1}{s}(\alpha e + \bar{\alpha})(e^x - 1) \end{aligned} \tag{47}$$

Now from (47) and (13),

$$\begin{aligned} \beta C_2 \quad &\geq \quad S'(t,s) - \delta^{t+1}S'(t,s) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}\delta^j S'(j,s) \\ &= \quad \frac{n\gamma C_1}{s}(\alpha e + \bar{\alpha})(e^t - 1) - \delta^{t+1}\frac{n\gamma C_1}{s}(\alpha e + \bar{\alpha})(e^t - 1) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}\delta^j\frac{n\gamma C_1}{s}(\alpha e + \bar{\alpha})(e^j - 1) \end{aligned}$$

or

$$\begin{aligned} \frac{s\beta C_2}{n\gamma C_1(\alpha e + \bar{\alpha})} &\geq (e^t - 1) - \delta^{t+1}(e^t - 1) - \frac{\beta\delta}{\alpha\bar{\beta}}\sum_{j=1}^{t}\delta^j(e^j - 1) \\ &= (e^t - 1) - \delta^{t+1}(e^t - 1) - \frac{\beta\delta}{\alpha\bar{\beta}}\left\{\frac{\delta e[1-(\delta e)^t]}{1-\delta e} - \frac{\delta[1-\delta^t]}{1-\delta}\right\} \\ &= e^t - 1 - e^t\delta^{t+1} + \delta^{t+1} - \frac{\beta\delta^2 e[1-(\delta e)^t]}{\alpha\bar{\beta}(1-\delta e)} + \delta - \delta^{t+1} \end{aligned}$$

or

$$e^t - e^t\delta^{t+1} - \frac{\beta\delta^2 e[1-(\delta e)^t]}{\alpha\bar{\beta}(1-\delta e)} \quad \leq \quad \frac{s\beta C_2}{n\gamma C_1(\alpha e + \bar{\alpha})} + (1-\delta)$$

or

$$e^t - e^t\delta^{t+1} - \frac{(1-\delta)\delta e[1-(\delta e)^t]}{(1-\delta e)} \quad \leq \quad \frac{s\beta C_2}{n\gamma C_1(\alpha e + \bar{\alpha})} + (1-\delta) \tag{48}$$

Therefore, for the case of exponential-cost penalty and randomized search propagation, we see from (48), that (13) becomes equal to,

$$\tilde{t}(s) \quad = \quad \max\Big\{t \quad \Big| \quad e^t - e^t\delta^{t+1} - \frac{(1-\delta)\delta e[1-(\delta e)^t]}{(1-\delta e)} \leq \frac{s\beta C_2}{n\gamma C_1(\alpha e + \bar{\alpha})} + (1-\delta), \quad t \in \{1,2,...\}\Big\} \tag{49}$$

(c) Linear-Cost penalty and Deterministic search propagation

For this case, from (4) and (5), we have

$$\begin{aligned} S(x,s) &= \gamma C_1[f_{intra}(x,\frac{n}{s}) + \frac{s-1}{s} f_{inter}(s)] \\ &= \gamma C_1[\frac{nx}{s} + \frac{s-1}{s}(s-1)] \\ &= \gamma C_1 \frac{nx}{s} + \gamma C_1 \frac{(s-1)^2}{s} \end{aligned} \tag{50}$$

Therefore from (50) and (13),

$$\begin{aligned} S'(x,s) &= \alpha S(x+1,s) + \bar{\alpha} S(x,s) - \alpha S(1,s) - \bar{\alpha} S(0,s) \\ &= \frac{n\gamma C_1 x}{s} \end{aligned} \tag{51}$$

We see that the expression of $S'(x,s)$ is same as the case of linear-cost penalty and randomized-search propagation. Therefore the expression of the stationary threshold also remains the same which is equal to,

$$\tilde{t}(s) = \max\Big\{t \;\Big|\; (1-\delta)t + \delta^{t+1} \leq \frac{(1-\delta)s\beta C_2}{n\gamma C_1} + \delta, \quad t \in \{1,2,...\}\Big\}. \tag{52}$$

However note that the expression of the quantity $MinCost(s)$ for this case will be different from the linear-cost penalty and randomized-search propagation case. This will result in a different s^* and so a different $t^* = \tilde{t}(s^*)$.

(d) Exponential-Cost penalty and Deterministic search propagation

For this case, from (4) and (5), we have

$$\begin{aligned} S(x,s) &= \gamma C_1[f_{intra}(x,\frac{n}{s}) + \frac{s-1}{s} f_{inter}(s)] \\ &= \gamma C_1[\frac{ne^x}{s} + \frac{s-1}{s}(s-1)] \\ &= \gamma C_1 \frac{ne^x}{s} + \gamma C_1 \frac{(s-1)^2}{s} \end{aligned} \tag{53}$$

Therefore from (53) and (13),

$$\begin{aligned} S'(x,s) &= \alpha S(x+1,s) + \bar{\alpha} S(x,s) - \alpha S(1,s) - \bar{\alpha} S(0,s) \\ &= \frac{n\gamma C_1}{s}(\alpha e + \bar{\alpha})(e^x - 1) \end{aligned} \tag{54}$$

We see that the expression of $S'(x,s)$ is same as the case of exponential-cost penalty and randomized-search propagation. Therefore the expression of the stationary threshold also remains the same which is equal to,

$$\tilde{t}(s) = \max\Big\{t \;\Big|\; e^t - e^t\delta^{t+1} - \frac{(1-\delta)\delta e[1-(\delta e)^t]}{(1-\delta e)} \leq \frac{s\beta C_2}{n\gamma C_1(\alpha e + \bar{\alpha})} + (1-\delta), \quad t \in \{1,2,...\}\Big\} \tag{55}$$

However, note that in this case too, the expression of the quantity $MinCost(s)$ will be different from the exponential-cost penalty and randomized-search propagation case. This will result in a different s^* and so a different $t^* = \tilde{t}(s^*)$.

References

[1] D. P. Bertsekas. *Dynamic Programming and Optimal Control*. Athena, 1995.

[2] A. Datta, M. Hauswirth, and K. Aberer. Updates in Highly Unreliable, Replicated Peer-to-Peer Systems. In *The Proceedings of ICDCS 2003 (to appear)*, 2003.

[3] P. Dey, B. Rimoldi, and E. Telatar. Optimal policy to track a mobile user. In *Winter School on Coding and Information Theory, Monte Verita, Switzerland*, Feb. 2003.

[4] P. Dey, B. Rimoldi, and E. Telatar. Protocol Overheads in Mobile Ad-Hoc Networks, 2003. Technical report, Nokia Research Center, Helsinki.

[5] F. Ferrandina, T. Meyer, and R. Zicari. Lazy Database Updates Algorithms: a Performance Analysis, 1994. Technical report, J.W. Goethe Universitat, Robert-Mayer-Str. 11–15, D-60054 Frankfurt am Main.

[6] R. M. Karp, C. Schindelhauer, S. Shenker, and B. Vöcking. Randomized rumor spreading. In *FOCS*, 2000.

[7] A. Labrinidis and N. Roussopoulos. Update propagation strategies for improving the quality of data on the web. In *The VLDB Journal*, pages 391–400, 2001.

[8] M.Jiang, J.Li, and Y.C.Tay. Cluster Based Routing Protocol (CBRP) Function Specifications, 1999. IETF Draft.