Short Paper: Dynamic Online Storage Allocation for Multi-Content Dissemination in Two-Tier Hybrid Mobile Vehicular Networks

Keyvan R. Moghadam*, Maheswaran Sathiamoorthy *, Bhaskar Krishnamachari*, Fan Bai†
*USC Ming Hseih Department of Electrical Engineering
†General Motors Global R&D, ECI Lab

***Abstract*—We present a two-tier hybrid mobile network architecture. In this architecture, the data plane consists of store and forward routing through an intermittently connected mobile network, and the control plane consists of an always-on infrastructure-based wireless network. This architecture aims to enhance bandwidth utilization while providing efficient centralized control. For such an architecture, we formulate and address, from a theoretical perspective, the fundamental problem of disseminating multiple files through storage limited nodes. Given a deadline, we investigate how best to utilize the intermediate nodes (called helper nodes) to disseminate content with the help of the control plane. We examine the formulated optimal policy through theoretical and numerical analysis. The analysis shows interesting counter intuitive facts about the optimal policy.**

I. Introduction

Rapid growth of data usage in everyday life and limited bandwidth resources has raised many concerns on the future of cellular networks as the only means of data transfer to mobile users. On the other hand, with the increasing availability of WiFi-direct capability on mobile devices as well as IEEE 802.11p / WAVE radios for vehicle to vehicle communications, there is a growing interest in exploring the design of large-scale intermittently connected mobile networks (ICMNs) to support such users. This is while, the coexistence of ICMNs with cellular networks in urban areas, provides an opportunity for these networks to work in conjunction with an always-on infrastructure network. This setup gives us the ability to split the data into two categories of delay tolerant and real time traffic and use the ICMN to disseminate the delay tolerant data and increase bandwidth efficiency. Thus, it is of great importance to develop architectures and algorithms for hybrid wireless networks that can combine both distributed and centralized communications.

In this work, we present a two-tier hybrid wireless network consisting of a data plane and a control plane. The data plane consists of nodes that store and forward files to other nodes through an ICMN. The control plane decides which content to store and which content to move by having the mobile devices periodically provide meta-information about their current contents to a centralized server through an always-on cellular network. The central server has a global view of the network and can run a content dissemination scheme by sending the control messages to each of the nodes. These control messages sent by the server are very small compared to files being exchanged by the nodes. We consider the general problem of disseminating multiple files through storage limited nodes in such a two-tier hybrid network. Given a deadline, we investigate how best to utilize the intermediate nodes (called helper nodes) to disseminate content with the help of the control plane.

This work was supported in part by the U.S. National Science Foundation via grant number CNS-1217260.

Most of the prior works in ICMNs are based on a single tier distributed structure and neglect the opportunity of having access to a wireless cellular backbone in urban areas. They usually deal with the dissemination of single files, and in the case of multiple files, the routing schemes are generally heuristics. By introducing our two tier structure, we move on to a more practical set up that leads to optimal solutions. We take a sophisticated, but tractable stochastic model where the contact process is i.i.d and homogeneous with an arbitrary inter-contact duration distribution. Such a contact process can be found in many examples including mobile nodes with random walk, random waypoint and random direction mobility models [2]. Under this assumption the dissemination of multiple files with limited storage can be modeled by a Markov chain in which states represent the number of nodes carrying each file. This formulation yields the number of states being polynomial in the size of the network, albeit exponential in the number of files.

For our Model, we are able to derive the optimal online storage allocation policy for a fixed number of files. This is achieved by solving a dynamic program representing a finite-horizon Markov Decision Process in polynomial time with respect to the size of the network. We examine the formulated optimal policy through theoretical analysis as well as numerical solutions under different conditions such as different deadlines and utility functions. These examinations, reveal interesting structural patterns of the optimal policy which in turn leads to guidelines for devising future scalable heuristics. Our results, also, provide a useful fundamental benchmark against which previous or future heuristics could be tested. One of the counter intuitive results, that we prove, argues that for many scenarios dividing the resources among different parties does not result in the best allocation policy. In fact the optimum policy allocates all of available resources to one party in each step. Here, due to space constraints we omit all the proofs and derivations. All the omitted details may be find in a longer version of this paper in [4].

Related Works: First studies on packet delivery in ICMNs started with Epidemic Routing also known as flooding [8]

where to guarantee delivery, all the mobile nodes participate in relaying the message by maintaining a copy of it and handing it to other nodes in each encounter. At the same time, other structures presented to increase connectivity in ICMNs such as Data MULES [3], and Ferry assisted routing [9].

While, all the above-mentioned works, are mainly focused on dissemination of single file, the set of works that do consider multiple files are generally heuristics and can be categorized in two main-streams of probabilistic routing or utility-based routing. For example, Single copy routing [6], [7] considers both the probabilistic routing and a set of utility based routing protocols, where the utility is measured in terms of how useful a node will be in delivering the packet to another node. In RAPID, Balasubramanian *et al*. [1] use utility metrics. The work by Reich *et al.* [5] has a similar flavor to ours, where they consider clients and servers (similar to demanding nodes and seeds), but their treatment of the problem are based on a distributed algorithm while not considering the opportunity of using a higher control tier. In fact, all these works by not exploiting the presence of a back bone layer, work on a one tier network and are more focused on developing specific heuristic for different set of assumptions.

II. Model and Problem Setup

Consider a set of N nodes in an intermittently connected mobile network coexisting with an always-on cellular network. Let $\mathcal{V}$ be the set of files on demand with $V = |\mathcal{V}|$. These files have been seeded earlier through the cellular network into some random nodes. We call these nodes as *seed* nodes and indicate their number for a particular file by $n_k(v)$, where $v \in \mathcal{V}$. Furthermore, assume that there are a set of nodes for each file asking to have that file called demands . Demands are not known to the control plane during the seeding phase. Lets $n_d(v)$ denotes the number of demanding nodes for file v. We assume that each seed is dedicated to a single file in $\mathcal{V}$ files and that demanding node requires one of these files.

With this setup, there are $s = N - \left(\sum_{v=1}^{V}(n_k(v) + n_d(v))\right)$ remaining nodes at the beginning that are neither interested nor have any of the V files to begin with. These nodes can be used to increase the connectivity of the network for any particular file by simply copying that file into their memory and handing it to the other nodes opportunistically. Given this logic behind the use of these nodes, we use the name "*helper* nodes" for them. Each of the helper nodes will make a certain storage available on their buffer which can help spread one type of content through the network. The seed nodes or demands may also free some buffer to participate in helping other files. However, without loss of generality, to make it easier for the reader to follow, we do not consider this scenario in our formulation. The goal is to moderate the helping nodes with the help of the control tier of the network in a way that maximizes the satisfaction of the users. To do that we need to measure the overall user satisfaction quantitatively. We will, shortly, define this measure as a utility function which can capture the two major aspects of being satisfied in a social sense. Also we need to tie this utility to a certain deadline, since no subscriber will remain interested in a file forever.

Contact Model: In our problem, contacts may happen between any two nodes of the ICMN layer (data plane). The length of each contact is negligible compared to the inter contact times and they happen in a sequence. The inter-arrival rate between any of two contacts is determined according to a random process, depending on network dynamics. Also, homogeneity is assumed. Such a homogeneity can be found in examples where nodes randomly distributed on a 2D field at the time $-\infty$ and each following a random walk moving pattern.In examples like this the inter-arrival contact times are distributed according to an exponential process [2].

Files can be transferred when two nodes meet each other, referred to as encounters here. The duration of the encounters are assumed to be sufficiently long enough to allow the transfer a complete file. Each encounter can be seen as a probabilistic event separated in time from the other encounters.

Helper Node Allocation: Helper node allocation problem arises when there are more than one files on-demand in the network. In such cases, each helper node has more than one option to help and needs to decide the file it would like to help. This is where the control tier steps in and having a global view of the network will make this decision for helper nodes by assigning each a file to help. The assignment can be done dynamically with two different strategies. One without and the other with rewriting permission. In the first strategy, a helper node will not rewrite its buffer with any other file once it's started helping one. In the second strategy, a helper node may need to rewrite its current buffer with the new assigned file in an encounter.

Choice of Utility Function: When there is only one file on demand, the objective is straightforward: happiness will be maximized when a higher percentage of demanding nodes are satisfied. However, the issue of happiness becomes more complicated when the number of involved parties increases. The first important notion which affects the satisfaction of customers when number of customer parties go beyond one is *fairness* and we try to incorporate it to our function.

Consider the general case of V files, each file v with demands $n_d(v)$. At the end of the deadline, let $c_d(v)$ be the number of demands satisfied for the corresponding file. The one we considered for out use here is following:

$$U(\mathbf{c_d}) = \beta \left(\frac{\sum_{v=1}^{V} c_d(v)}{\sum_{v=1}^{V} n_d(v)} \right)^2 - (1-\beta) \left(\sum_{v=1}^{V} \left| \frac{c_d(v)}{n_d(v)} - 1/V \sum_{v=1}^{V} \frac{c_d(v)}{n_d(v)} \right| \right)^2 - \beta \quad (1)$$

for $0 \leq \beta \leq 1$. It is not hard to see that this function accounts for the total number of satisfied demands for $\beta > 0$. It also prevents starvation and punishes unfairness to a certain degree (depending on β). Certainly, the best choice of β depends on the application and the user behavior. More detail about the choose of utility function may be found in [4]. Once the utility is designed, the goal of the helper node allocation strategy

would be to maximize the system utility by suitably allocating the helper nodes for each file.

III. Dynamic Allocation Scheme

In this section, we present the Dynamic Allocation Scheme, a Markov Decision Process (MDP) based allocation scheme to allocate the helper nodes to the demanded files. We argue that this scheme is optimal in maximizing the expected user satisfaction utility by the deadline and can be used as a bound against heuristic approaches. We expect the Dynamic allocation scheme to be run on a central server that has global view of the network. Choosing the right number of helper nodes for optimally disseminating existing files depends very much on the number of seeds, demands and the contact model. Since such an adaptive method has to *decide* at each stage how many helper nodes are to be enlisted, it can naturally be modeled as an MDP.

MDP Primer: An MDP problem ρ is represented by a 4-tuple $\{\mathcal{S}, \mathcal{A}_\cdot, p_\cdot(\cdot,\cdot), R_\cdot(\cdot)\}$ where $\mathcal{S}$ is the state space, consisting of a finite number of states. $\mathcal{A}_S$ is a finite set of possible actions in S. $p_{\mathbf{a}_S}(S, S')$ is the transition probability of going from $S \in \mathcal{S}$ to $S' \in \mathcal{S}$ given action $\mathbf{a}_S \in \mathcal{A}_S$ is performed and $R_{\mathbf{a}_S}(S)$ is the expected reward collected at state $S \in \mathcal{S}$ when action $\mathbf{a}_S \in \mathcal{A}_S$ is taken. Given the network state, the goal is to find the associated action which maximizes the total finite horizon reward till the end of time (finite horizon).

State Space: Each state of the state space should capture the "state" of the dissemination process. In our case, due to the i.i.d contact process and the homogeneity of the nodes, we need to track the count of these state identifiers. Thus, Considering that there are total V files in the system, each state S can be represented by a $2V+1$-tuple $\{(c_d(v,S), c_h(v,S))_{v=1}^{v=V}, t\}$. In which, $c_d(v,S)$ and $c_h(v,S)$ are, respectively, the number of satisfied(completed) demands and number of helper nodes for file v and t denotes number of time slots elapsed since the initial time. $t \leq T$ where T is the deadline.

Actions: An action at a state S is defined as vector $\mathbf{a}_S = (a_1, \ldots, a_V)^T$, where a_v, denotes the number of helper nodes assigned in current time slot to help file v. When rewriting is allowed, feasible actions can be represented by a set of vectors $\mathcal{A}_S = \{\mathbf{a}_S = (a_1, \ldots, a_V)^T | 0 \geq a_v, \sum_{v=1}^{V} a_v = s\}$ and when rewriting is not allowed $\mathcal{A}_S = \{\mathbf{a}_S = (a_1, \ldots, a_V)^T | \mathbf{a}_S \geq c_h(v,S)\mathbf{1}_{V\times 1}, \mathbf{1}^T\mathbf{a}_S = s\}$.

Transitions: When the network is in a state S with time stamp t, it will moves to another state with time stamp $t+1$ in the next encounter. The new state depends on the action vector $\mathbf{a}_S$ and the nodes which meet in the encounter. Since the control-tier can not predict the future behavior of the nodes, the next encounter is always unknown at the time of assigning the helpers nodes. As a result, any action taken by the control tier results in moving to a set of possible states with certain probabilities. The server makes its decision with respect to this transitional probabilities. These transitional probabilities are presented in the rest of this section for the case where rewriting is permitted. The detailed derivation of each can be found in [4] which is an extended version of this paper. In following formulations, which demonstrate the transition probabilities, S is the current state, S_0 is the next state in time where nothing changes compare to S, $S_1^{v'}$ is the next state in time where one is added to the satisfied demands of file v', $S_2^{v'}$ is the next state in time where one is added to the helpers of file v' and $S_3^{v',v''}$ is the next state in time where one is added to the helpers of file v' and one is reduces from the helpers of file v''.

$$p_{\mathbf{a}_S}(S, S_1^{v'}) = \frac{2(n_k(v') + c_d(v',S) + c_h(v',S))(n_d(v') - c_d(v'))}{N(N-1)}$$

$$p_{\mathbf{a}_S}(S, S_2^{v'}) = \begin{cases} \frac{2(n_k(v')+c_d(v',S)+c_h(v',S))(a_{v'}-c_h(v',S))}{N(N-1)}, & \text{if} a_{v'} > c_h(v',S) \\ 0, & \text{Otherwise.} \end{cases}$$

$$p_{\mathbf{a}_S}(S, S_3^{v',v''}) = \begin{cases} \frac{2(n_k(v')+c_d(v',S)+c_h(v',S))k}{N(N-1)} & \text{if} a'_{v'} > 0, a'_{v''} < 0 \\ 0, & \text{Otherwise.} \end{cases}$$

where k is the fraction of helpers to v'' which may be overwritten by v', $a'_v = a_v - c_h(v,S)$ is the number of helpers for file v which do not have the file yet, and n_0 is the number of all helper nodes which have not downloaded any file into their buffer yet. Finally, $p_{\mathbf{a}_S}(S, S_0)$ can be calculated knowing that the probability of all possible events should add up to 1.

The Bellman Equation: For the terminating states we can calculate the rewards based on the utility function from Eq 1, $J(S) = U(\mathbf{c}_d)$. The rewards for non terminating states can be calculated by propagating the results from the terminating states using bellman equations, $J(S) = \max_{\mathbf{a}_S \in \mathcal{A}_S} \sum_{S' \in \mathcal{S}} \{p_{\mathbf{a}_S}(S, S') J(S')\}$. This concludes the MDP formulation. Having the MDP formulated in this way, we can prove the following Theorem:

Theorem 1. *The solution of the formulated MDP is the optimal solution for the problem of maximizing the expected user satisfaction by the deadline.*

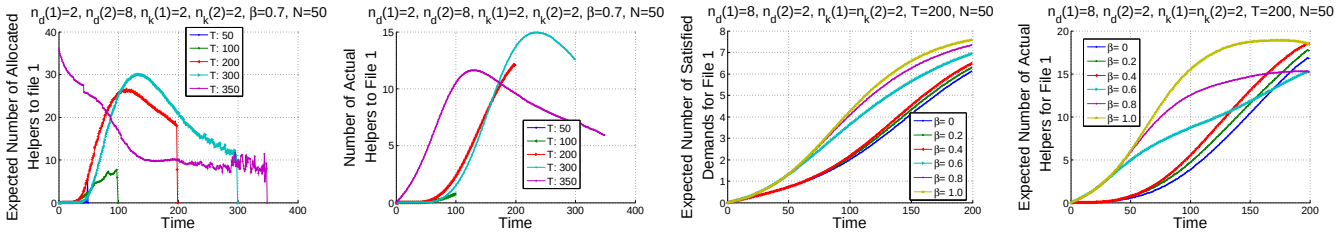
Detailed complexity analysis for getting the solution of this MDP problem is discussed in [4]. In the rest of the paper we study this optimal solution to get to structural pattern of optimal policy. These patterns can be used to develop more precise heuristics.

IV. Analysis

Dissemination without overwriting permission In this case there is no need to search the entire $\mathcal{A}_S$. In fact we prove this interesting property of optimal solution that can reduce the search space into V specific points which are the vertices of $\mathcal{A}_S$. This can be stated as follows:

Theorem 2. *Given MDP formulation ρ of the problem, when overwriting is not allowed, the optimal policy for any state S allocates all the available nodes to help with one file.*

Dissemination with overwriting permission In this case, we investigate the behavior and dependencies of the optimum



(a) (b) (c) (d)

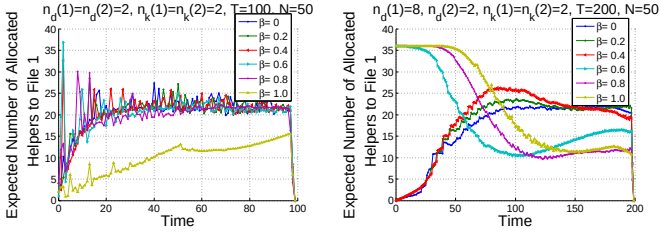
Fig. 1: Expected actions, satisfied demands and helper nodes at each time slot for different deadlines and β.

dynamic allocation scheme by solving the MDP for different settings. For simplicity and ease of exposition, we focus on the dissemination of only 2 files. The number of helper nodes that get allocated to the files are a_1 and a_2 and since $a_2 = s - a_1$, a_2 is automatically inferred when a_1 is specified. The number of nodes is set to be $N = 50$ for all the experiments. We solved the MDP for may different settings from which we present only a few here due to space constrains.

Effect of Deadline: Fig. 1a and Fig. 2a show the optimum policy for the case with unequal and equal number of demands respectively. Fig. 1a shows the number of helper nodes allocated for file 1 (a_1) at each encounter. Except the case with the largest deadline, the optimal policy allocates all the nodes to the file with higher number of demands file at the beginning. In other words the optimal policy, always attempts to first compensate for the fairness factor and then moves for increasing the incentive factors. The case with $T = 350$ can be justified by knowing that for long enough deadlines there is enough time to have both files completely satisfied *i.e.* a wide range of possible actions is optimal. Fig. 1b, shows expected number of nodes actually helping file 1 when applying the optimal policy. It can be seen that the optimal policy does not overwrite until all the helpers have full buffers. As a result, the optimal decision is the same as the case where overwriting is not allowed at the beginning phase of data dissemination.

Effect of β: Figure 2a depicts the expected action for the case where files have equal initial conditions. It can be seen that as long as $\beta < 1$, the expected optimal action fluctuates from one extreme to another. This is interpreted as the attempt of optimal policy to remain fair with respect to both files. These results are consistent with Theorem 2 and our previous observation that the optimal solution with overwriting permission is the same as the optimal allocation without overwriting permission when there are helpers with free buffers available. Fig. 2b is the expected optimal allocation policy for the case where demands are unequal. No more fluctuations is seen as a result of the asymmetry in the files' initial condition. Depending on the weight we consider for each of incentive or fairness, the two different aspect of user happiness in the utility function, at each state, the optimum policy goes for the file which is more helpful in achieving the maximum utility and dedicates all the helper nodes to that file.

Figs. 1c and 1d show the expected number of satisfied demands and actual helpers of file 1 for such scenario. As it can be seen the policy never uses its overwriting permission for helpers to either of files. More detailed discussion can be found in [4].



(a) (b)

Fig. 2: Expected number of allocated helpers to file 1 at each time slot for the cases with equal and unequal demands.

V. Conclusion and Future Work

In this work, we have advocated using an always-on cellular infrastructure as controlling tier in conjuncture with ICMNs. This resulted in a hybrid two tier architecture which suits urban areas. A mathematical approach is followed through the paper towards an optimal solution of resource allocation in such architectures. The optimal solution considers both fairness and incentive in dissemination of multiple files through the network. The analysis of optimal solution revealed many different properties for different scenarios.

References

[1] A. Balasubramanian, B. Levine, and A. Venkataramani. Dtn routing as a resource allocation problem. In *ACM SIGCOMM Computer Communication Review*, volume 37, pages 373–384. ACM, 2007.

[2] R. Groenevelt, P. Nain, and G. Koole. The message delay in mobile ad hoc networks. *Performance Evaluation*, 62(1):210–228, 2005.

[3] David Jea, Arun Somasundara, and Mani Srivastava. Multiple controlled mobile elements (data mules) for data collection in sensor networks. In *Distributed Computing in Sensor Systems*, pages 244–257. Springer, 2005.

[4] K. R. Moghadam, M. Sathiamoorthy, B. Krishnamachari, and F. Bai. Dynamic online storage allocation for multi-content dissemination in icmns, 2013. http://anrg.usc.edu/www/publications/papers/SocialReplication.pdf.

[5] J. Reich and A. Chaintreau. The age of impatience: optimal replication schemes for opportunistic networks. In *Proc. of ACM the 5th international conference on Emerging networking experiments and technologies (CoNEXT)*, pages 85–96, 2009.

[6] T. Spyropoulos, K. Psounis, and C.S. Raghavendra. Single-copy routing in intermittently connected mobile networks. In *Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004. 2004 First Annual IEEE Communications Society Conference on*, pages 235–244. IEEE, 2004.

[7] T. Spyropoulos, K. Psounis, and C.S. Raghavendra. Efficient routing in intermittently connected mobile networks: The single-copy case. *IEEE/ACM Transactions on Networking (TON)*, 16(1):63–76, 2008.

[8] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, CS-2000-06, Duke University, 2000.

[9] Farzana Yasmeen, M Huda, M Haque, Michihiro Aoki, and Shigeki Yamada. Using ferry access points to improve the performance of message ferrying in delay-tolerant networks. *Proc. ICCNMC, France*, pages 173–179, 2011.