Fifty-third Annual Allerton Conference
Allerton House, UIUC, Illinois, USA
September 29 - October 2, 2015

Architecture and Implementation of An Information-Centric Device-to-Device Network

Yu Wu, Matthew Barnard and Lei Ying
Electrical, Computer and Energy Engineering
Arizona State University

ABSTRACT

Today's mobile devices almost exclusively connect to infrastructures for communications and information access; but advances such as AirDrop and WiFi Direct are bringing device-to-device communication to the forefront of mobile computing. Self-organizing ad hoc mobile networks have a wide range of applications in scenarios where infrastructure is not available or with limited bandwidth, such as communications in the aftermath of natural disasters, censorship resistant communications, and battlefield communications. In this paper, we propose an information-centric device-to-device network, called ICD2D. The network is distributed and requires no centralized coordination. For each published item of data, the system creates a metatdata; a publish-subscribe mechanism disseminates the metadata and facilitates filtering information in a distributed fashion. We implemented a full-featured system on NS3. Evaluations show significant improvement in successful information retrieval compared with OLSR (Optimized Link State Routing), a common approach to ad hoc routing.

1. INTRODUCTION

Mobile devices such as wearable tech, smart phones, and tablets have penetrated almost every aspect of our daily lives, becoming extensions of ourselves. Today's mobile devices mainly connect to infrastructure (such as cellular base stations and WiFi access points) for communications and information access; but device-to-device communication technologies, such as AirDrop, WiFi ad hoc, and WiFi Direct, are making the mobile ad hoc network a reality.

Given the popularity of mobile devices and device-to-device communication technologies, the application of a self-organizing ad hoc mobile network is clear. Though the infrastructure network will surely remain dominant, device-to-device mobile networks will fill the gaps where high deployment and administrative costs or impracticality have traditionally kept networks sparse. We next list a few such applications.

- **Communications for disaster recovery:** Natural disasters or attacks of weapons of mass destruction can cause catastrophic failure of network infrastructure, but are less likely to destroy hand-held devices. Automatically discovering and networking mobile devices in an area can make life-saving information (e.g. water, food, or power sources) available for connected survivors.
- **Censorship resistant communications:** During the Arab Spring Egypt, Libya, and Syria each shut down Internet access for an extended period in order to block information about protests and communications among protesters. A self-organizing mobile network can provide an alternative communication network during these interruptions and is resistant to government censorship.
- **Battlefield communications:** In battlefield communications, in particular in hostile environments, limited bandwidth is available via infrastructure-based communications such as satellites. A self-organizing network formed by soldiers' mobile devices can be used for coordination in the battlefield.

In the scenarios above, it is impossible to plan ahead and deploy communication infrastructure a priori. However, timely information sharing and access are imperative in these live-or-death scenarios. Motivated by the applications above, we propose an information-centric paradigm for self-organizing device-to-device networks. The main contributions of this paper include:

- In the applications mentioned above, the primary purpose of networking is information sharing, instead of end-to-end communications. Therefore, our system adopts an *information-centric* network architecture, in which each mobile device periodically synchronizes with its neighbors on metadata of available contents in the network. This architecture eliminates the need to maintain end-to-end routes, a benefit we gain over most existing mobile ad hoc networks.
- The network architecture is completely distributed and requires no centralized coordination. In our system, each item of content is associated with a piece of metadata created by the publisher. Users subscribe

to metadata filters that automatically select matching metadata from the network, allowing decisions on content retrieval to be made manually or automatically. This publish-subscribe paradigm for metadata facilitates distributed information search and is particularly suitable for spontaneous events where new contents are continuously added into the network. From this perspective, our system is also different from most existing information-centric network architectures where a centralized name server is required for content management.

- We designed and implemented a full-featured system on NS3. Our experimental evaluation confirms the novelty of our design and shows significant improvement on fulfilling information requests compared with OLSR.

1.1 Related Work

The concept of information-centric network (or content-centric network) is not new. Starting from the seminal work [1], information-centric networking has emerged as one of research areas that may revolutionize the future Internet. A range of problems have been studied, including naming [2, 3, 4], routing [5, 6, 7, 8], in-network caching [9, 10, 11, 12, 13, 4, 14], etc. However, these solutions are almost exclusively tailored for the wired Internet (at best, they include mobile phones at the edge) and assume the existence of centralized naming server and content store. Fundamentally different problems need to be tackled for a high-performance information-centric D2D network due to node mobility and wireless communications. There have been a series of work studying the congestion control in content centric network. For example, Saino *et al.* [15] propose a receiver-driven congestion control mechanism by predicting the location of content chunks before requesting. Arianfar *et al.* [16] introduce a preliminary protocol based on classic TCP congestion window which however relies on data packets to explicitly enforce the size adjustment of congestion windows. Our work differs from theirs by considering the end to end latencies of data block transmission to dynamically throttle the request rate. Facilitated by an efficient joint optimization based scheduling algorithm, our system can achieve satisfiable content fulfilment ratio, as verified in Sec. 4.

2. OVERVIEW OF THE SYSTEM

In this section, we present an overview of the system, including data structure, message structure and three key user operations. The system, as shown in Figure 1, adopts an information-centric architecture and consists of three layers.

- **Information layer**: This layer is dedicated to the management of local (locally generated or received) information contents. It handles functionalities such as publishing, ranking, caching and content forwarding.

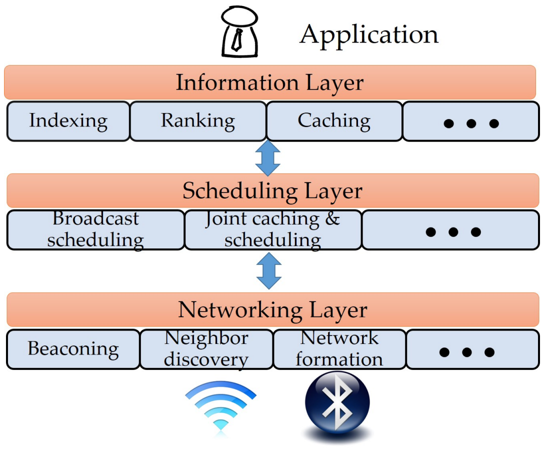

Figure 1: The three layers of an information-centric D2D network

- **Scheduling layer:** This layer is the communication foundation of the system that schedules data transmissions based on the type, freshness, and popularity of a content and based on the state of the network.

- **Networking layer:** This layer is responsible for neighbor discovery and network formation, i.e., maintaining the connectivity of the network.

2.1 Data Structure

The system includes four types of data as described below.

- **Published data**: These are raw data that are generated by end users and will be requested by other users in the system.

- **Data blocks**: A published data is fragmented into a set of data blocks. Each data block contains the name of the corresponding published data and a sequence number that is the position of the block and is needed for resembling the published data. The ID of a data block is the hash value of the data block.

- **Metadata**: During fragmentation, a metadata is created for each content. A metadata includes the name, the name of data blocks, the GPS location when published, a set of associated hashtags and a human readable field describing the other important information of the content. The hash value of the content is also used as the ID of the metadata.

- **Hashtags**: Hashtags are keywords that are used for grouping or filtering metadata. Hashtags can be created when a user publish contents or when a user requests contents.

2.2 Message Structure and Processing

In this paper, we use "messages" to refer to data packets that are broadcast in the network. Note that except the data-block messages, other messages can be grouped into a single packet as the sizes of these messages are small.

- **Beacons**: A beacon message includes node ID, which is the hash value of the node's MAC address, and the GPS location of the node. A mobile device periodically broadcasts beacon messages to let the neighbors be aware of its existence.
- **Interests**: An interest message is a list of (hashtag, TTL) pairs, where TTL stands for time-to-live which is the remaining number of broadcasts allowed for the hashtag. Each mobile device maintains a pending interest table. When receiving an interest message, if a hashtag is not in the pending interest table and the TTL is ≥ 1, the (hashtag, TTL) pair will be added to the table; otherwise, if the TTL in the interest message is larger than the TTL (for the same hashtag) in the table, then the TTL in the table gets updated. At the beginning of every *interest-broadcasting interval*, a mobile device reduces all TTLs in the pending interest table by one and then broadcasts the pending interest table (as one interest message).
- **Catalogs**: A catalog message is a list of (metadata, TTL) pairs where the metadata are all associated with a specific hashtag, and the TTL (time-to-live) in the category message is the amount of time before which the metadata will be removed from the mobile device. Each mobile maintains a catalog library. When receiving a catalog message, *if the hashtag associated with the catalog message is in the node's pending interest table*, then catalog is merged into the catalog library by adding new (metadata, TTL)s to the corresponding catalogs and updating those (metadata, TTL) pairs if the TTL in the catalog message is larger than that in the library. The catalog library is refreshed periodically by removing those (metadata, TTL) pairs with TTL$\leq$ 0. The TTL in the (metadata, TTL) pair is set initially by the mobile device who creates the metadata, and is reset when the data block associated with the metadata is requested to be retrieved. In this way, metadata of popular contents will be kept in the system and metadata of unpopular contents will be removed from the system avoiding congesting the network.
- **Requests**: A request message consists of data block ID, weight, the content location and TTL. Each mobile device maintains a pending request table. When receiving a request message, a mobile device adds (data block ID, weight, TTL) triple to the pending request table if the data block ID does not exist or updates the TTL if the the data block ID exists but the TTL in the table is smaller than that in the request message. During each *scheduling interval*, the mobile device broadcasts all or a subset of data blocks that are in the pending request table. When a data block is successfully broadcast or the associated TTL becomes zero, the corresponding entry is removed from the pending request table. The weight field in the request message indicates the importance of the data block for retrieving the content. For example, in a video file, I-frames are more important than other frames for reconstructing the video.
- **Data blocks**: A data-block message includes a data block. A mobile device broadcasts a data-block message when it receives a data block that is in its pending request table or receives a request message for a data block that is in its cache, directed by the joint scheduling algorithm introduced in Sec. 3.2.

2.3 Key User Operations

From the perspective of end users, the system supports three primitive operations: publish, subscribe and retrieve.

- **Publish:** To publish a content, the content is first fragmented into data-blocks [1], and a metadata is created for the new content. Each metadata is assigned with a set of hashtags by the user , and is added to the corresponding catalog.
- **Subscribe**: Since published data are generated in the network in real time, users do not have prior knowledge of available contents. To search interesting contents in the network, a user periodically broadcasts "interest" messages to their neighbors. As described in Section 2.2, these interest messages will be disseminated in the network and the mobile user is expected to receive all catalogs associated with the hashtags in the interest message. Metadata received will be further filtered according to local rules defined by the mobile user, ranked according to a ranking algorithm that takes into consideration the popularity of contents and other factors, and then sent to user interface for display.
- **Retrieve:** From the ranked metadata, a mobile user can select one or multiple contents to retrieve by sending out request messages. As described in Section 2.2, when a node receives a request and finds the corresponding data block, it will potentially broadcast the

[1]The block does not have to be equal-sized, but maximal size should not exceed the Maximal Transmission Unit (MTU) on the MAC layer.

data block, depending on the scheduling result to be introduced in Section 3.2; otherwise, it will broadcast the request message to its neighbors.

3. INFORMATION-CENTRIC CONGESTION CONTROL AND SCHEDULING

Due to space constraint, we only present the design of two key components of ICD2D: information-centric congestion control and information-centric scheduling.

3.1 Congestion Control

In ICD2D, the traffic load in the network is determined by multiple factors including users' request rates, content sizes and content locations, so the design is much more complex than in traditional communication networks. To develop a tractable solution, we consider a simple model as shown in Figure 2.

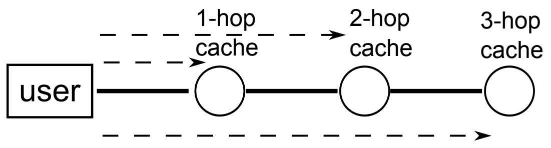

Figure 2: A simple model for information-centric congestion control

Let x_r denote the request rate of user r and $p_{r,h}$ the probability that the content requested by user r is h-hop away. Then we consider the following utility maximization problem:

$$\begin{aligned} &\max_{\{x_r\}} \textstyle\sum_r U_r(x_r) \\ \text{subject to: } &\textstyle\sum_r x_r p_{r,h} \le c_h \quad \forall h, \end{aligned}$$

where $U_r(\cdot)$ is the utility function of user r and c_h denotes the capacity of the network for retrieving one-hop contents. We remark that the network capacity of retrieving h-hop contents is c_h. A more accurate model is to consider a network capacity region $\mathcal{C}$ and constraint $\mathbf{c} = (c_1, c_2, \ldots)^t \in \mathcal{C}$, where the capacity region is determined by the PHY layer and scheduling algorithms, and could be difficult to characterize. We therefore only consider the simple model above and will study a more practical model in an extended version of the paper.

Now to solve the network utility maximization problem above, we consider the following Lagrangian dual

$$\begin{aligned} &\max_{x_r} \textstyle\sum_r U_r(x_r) + \sum_h \gamma_h \left(c_h - \sum_r x_r p_{r,h}\right) \\ = \; &\max_{x_r} \textstyle\sum_r \left(U_r(x_r) - x_r \sum_h \gamma_h p_{r,h}\right) + \sum_h \gamma_h c_h, \end{aligned}$$

where γ_h is the Lagrangian multiplier associated with hop constraint h. Since $\sum_h \gamma_h c_h$ is a constant when γ_h's are given, the congestion algorithm should maximize

$$U_r(x_r) - x_r \sum_h \gamma_h p_{r,h}. \qquad (1)$$

In the Internet congestion control literature, it is well-known that the Lagrangian multiplier γ_h can be measured using latency. Therefore, $\sum_h \gamma_h p_{r,h}$ can be approximated using the average latency experienced by user r.

As an example, assume $U_r(x_r) = w_r \log x_r$, then the optimal solution to (1) is

$$x_r = \frac{w_r}{\sum_h \gamma_h p_{r,h}}.$$

An adaptive control algorithm is

$$\dot{x}_r = w_r - x_r \sum_h \gamma_h p_{r,h}.$$

A key design consideration is the method to estimate the average latency. We will study this in the performance evaluation using NS3.

3.2 Information-Centric Scheduling

This module is the "brain" of the system and schedules all network activities of the corresponding mobile device. Specifically, this module makes the following decisions, aiming at maximizing utility for network welfare: how to efficiently (1) efficiently transmit/forward messages, (2) utilize the limited cache and (3) broadcast the data blocks. We answer these questions through a joint optimization framework.

We assume a limited amount of system RAM is reserved for the "cache" usage. The cached blocks should be broadcast to the neighborhood, subject to the bandwidth capacity capped by the link capacity and transient network condition. We therefore jointly consider the caching and forwarding problems. We define $\mathcal{N}$ to be the set of neighboring nodes. We assume each content is fragmented to data blocks of equal size as in most real-life applications, and each data block can be transmitted within one scheduling interval (e.g., 100 ms. Let $\mathcal{C}$ denote the set of data blocks received by time slot t, including newly received data blocks and those in the cache, $w_{i,c}$ denote the weight of neighbor node i for block c, t_c denote the TTL, $y_{c,t}$ denote whether the node caches content c at time t, S and B denote the storage capacity and link capacity, respectively. The problem is formulated into an optimization problem shown in Eqn. 2.

$$\max \sum_{c\in\mathcal{C}} \sum_{j\in\mathcal{N}} w_{j,c} \times (y_{c,0} - y_{c,t_{j,c}+1}) \qquad (2)$$

subject to:

$$\begin{aligned} (a) \quad & \sum_{c\in\mathcal{C}} y_{c,t} \le S, \quad \forall t_0 \le t \le \max_{j\in\mathcal{N}} t_{j,c} + 1 \\ (b) \quad & \sum_{c\in\mathcal{C}} (y_{c,t} - y_{c,t+1}) \le B \quad \forall t_0 \le t \le \max_{j\in\mathcal{N}} t_{j,c} \\ (c) \quad & y_{c,t} \ge y_{c,t+1}, \quad \forall c \in \mathcal{C}, t_0 \le t \le \max_{j\in\mathcal{N}} t_{j,c} \end{aligned}$$

The objective function maximizes the overall weight contributed by the considered node by caching and broadcasting data block at scheduling interval t_0, and any cached data block needs to be broadcast before their TTLs expire. Constraint (a) is the storage capacity constraint, (b) is the bandwidth constraint, and constraint (c) implies that any data block can only be cached in the scheduling interval at which it is received, and be removed before broadcasting.

We can see the problem (2) is a linear integer problem, which is NP-complete (The proof is by a reduction from the satisfiability problem [17]). However, the problem size is limited because the scheduling process occurs at a rather short interval (*i.e.*, 100 ms). During one interval, the number of received data blocks is limited, even under heavy traffic regime.

The solution of the optimization problem results in both caching and broadcasting such that:

$y_{c,0} = 0 \ \rightarrow$ `unfreeze` block c

$y_{c,0} = 1 \ \rightarrow$ `freeze` block c

$y_{c,0} = 1$, $t^* = argmin_t\{y_{c,t} = 0\} \ \rightarrow$ `broadcast` block c at time $t^* - 1$.

It is worthy pointing out that the the joint optimization framework supports preemptive scheduling. Newly-arrived requests with higher weights may cancel the already-scheduled frozen data blocks. We leave the performance analysis of this scheduling algorithm to the extended version of the paper.

4. PERFORMANCE EVALUATIONS ON NS3

We implemented a full-featured system on NS3 with key functionalities at all three layers. We mimicked a battlefield communication scenario with 100 soldiers in a $1.5 \times 1.5 km^2$ area of open space. Each soldier moves at a speed of 2 meters per second and is equipped with a WiFi-enabled wearable sensing device supporting 1 Mbps data rate. We assume 20 out of the 100 soldiers continuously collect data at an average rate of 128 kbps and publish to the system. Different unique hashtags are attached to contents generated by those patrolling devices, so a soldier can switch to a channel by explicitly submitting the “interest” message to the corresponding hashtag. Each solder is allowed to switch to a channel to request a content per second. For each request, the TTL is randomly chosen from 400 ms to $30,000$ ms, and the weight is randomly selected from $[0.1, 1]$. By default, the cache size is 256K bytes, realistic for the configuration of general sensor devices. The block size is chosen to be 1 K byte.

We conducted our evaluations on our NS3 by simulating 802.11b WiFi interfaces and configured each device to the ad-hoc mode. We applied the the constant speed delay model and the friis propagation loss model [18] for wireless channels, and some key settings are shown in Table 1.

Table 1: Parameter configuration on NS3

Parameter	Value
TxPower	0.1
TxGain	1.0
RxGain	1.0
RSS	-86 dBm
phyMode	DsssRate1Mbps
propagation loss model	Friis

We consider two representative mobility model, *i.e.*, Random Waypoint (RW) and Reference Point Group (RPG). In RW model, a node randomly chooses a destination and moves with a random velocity between $[0, V_{max}]$ for a certain duration of time. The whole process repeats until the the simulation ends. RW is widely used in research community, as a sufficiently generic model covering most cases. In RPG model, a group leader determines the group's motion behavior, while the other nodes randomly move in the neighborhood of the group leader, with both direction and velocity correlating to those of the group leader. Specifically, $V_m(t) = V_l(t) + rand() \times SDR \times V_{max}$, $\theta_m(t) = \theta_l(t) + rand() \times ADR \times \theta_{max}$. $V_m(t)$ and $\theta_m(t)$ represent the speed and direction of node m, while SDR and ADR indicate the speed and direction deviation ratio against the leader l , respectively. Our simulation results were under RW mobility model, unless otherwise explicitly specified.

We ran the system for 600 seconds, and evaluated scheduling complexity, network latency, content fulfilment rate and user utility under different parameter configurations, *e.g.*, cache size, different algorithms.

4.1 Scheduling Complexity and Effectiveness

We examined the computational complexity of our scheduling algorithm, in terms of the amount of time used to solve the optimization problem (Eqn. 2), referred as *scheduling complexity.* The scheduling decision was made every 100 milliseconds in our system. Scheduling complexity should be less than the scheduling interval of the system, i.e., 100 milliseconds.

Fig. 3 shows the scheduling complexity of a randomly chosen device over a duration of 200 seconds (the scheduling complexity of other devices is similar). We can see that the maximal time is smaller than 6 milliseconds, much smaller than 100 milliseconds. This result demonstrates the feasibility of our proposed scheduling algorithm for larger scale problem. We further varied cache size from 64K bytes to 1024K bytes and found the increase of the scheduling complexity is minor. The average scheduling delays are shown in Fig. 4. Therefore, it is promising that our system could be used in real-time scenarios where short scheduling intervals.

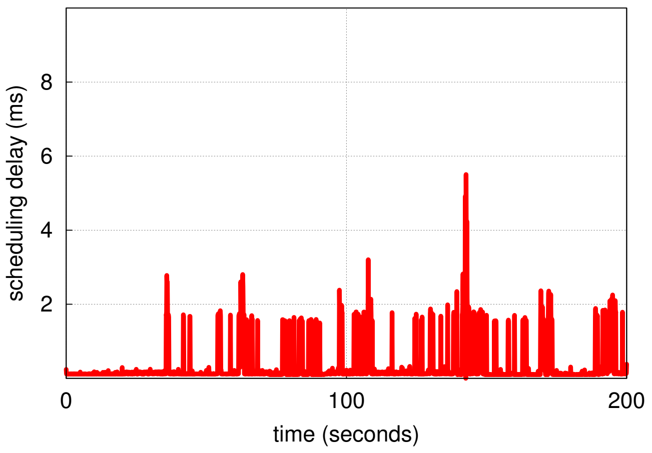

Figure 3: Scheduling complexity over time

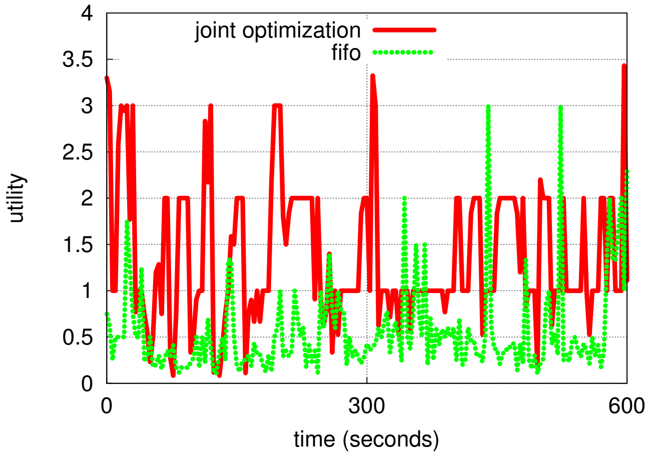

Figure 5: Utility contributed to the network by a user over time

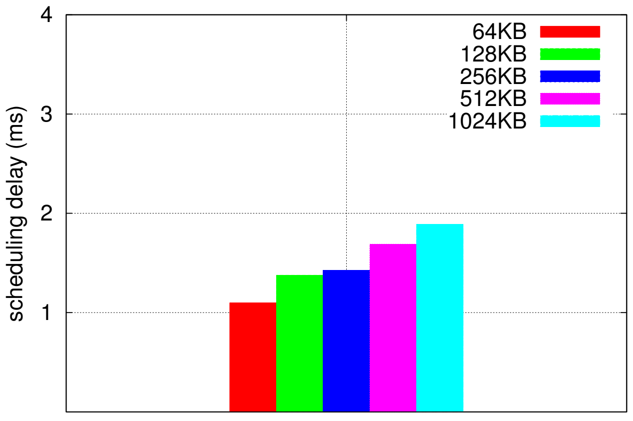

Figure 4: Scheduling complexity over time

To further evaluate the effectiveness of the scheduling algorithm based on the optimization framework, we randomly selected a user, and plotted the utility value over time as shown in Fig. 5. The red curve is the total utility of the broadcast data-blocks under the proposed scheduling algorithm based on the optimization framework (Eqn. 2), and the green curve is the total utility based on a best-effort scheduling which operates in a first-come-first-served (FCFS) manner, adopted by general ICN systems. We can see that our design yields significantly higher utility. The time-average utility of our design is 1.427 while that of the FIFO is 0.579.

4.2 Request Fulfilment Ratio and Latency

We define "fulfilment ratio" as the fraction of content requests that are successfully retrieved before their deadlines (i.e., before the TTL specified in the corresponding request). We first vary the cache size from 64K bytes to 1024K bytes, as shown in Fig. 7. We can see a larger cache size indeed yields better performance, as the opportunistic caching with a larger cache increases the chances of local hits. In our settings, we assigned higher popularity on fresh contents, and old contents fade out from the system as time goes on. As a result, larger cache size only yields marginal rate growth after the cache size becomes sufficiently large. For example, a 1024K byte sized cache yields 76.812% fulfilment rate, as compared to 75.58% for a 512K byte sized cache.

We also compared the performance of our algorithm with and without congestion control, and further compared them with the routing-based solution, the Optimized Link State Routing Protocol (OLSR), by periodically computing the optimal paths at each device. "ICD2D-congest" refers to our algorithm with congestion control, "ICD2D-non-congest" refers to our algorithm without congestion control, and "olsr-congest" represents the classic OLSR algorithm. Fig. 8 plots the fulfilment ratio achieved by these three algorithms. We can see that, "ICD2D-congest" significantly outperforms "ICD2D-non-congest". In fact, "ICD2D-congest" requests 34.89% less contents compared to "ICD2D-non-congest", throttled by the the congestion control module. However, 20.855% more contents are successfully received before the deadlines as shown in Fig. 6. The phenomenon stresses the importance a well designed congestion control algorithm in such a distributed system.

"olsr-congest" has the worst result, because of two reasons: (1) the control traffic triggered by the OLSR protocol occupied a significant portion of the bandwidth, which causes more network contention; (2) the lack of opportunistic caching increases the number of hops (so the number of transmissions) for content retrieval.

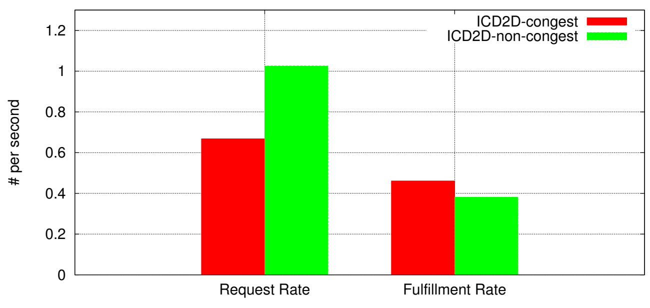

Figure 6: Comparison of request rate and fulfilment rate

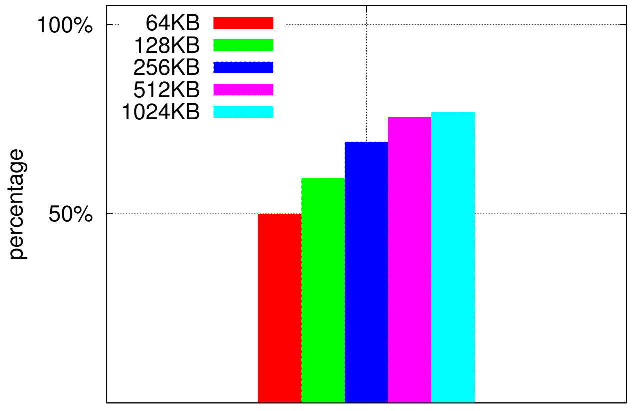

Figure 7: Fulfilment ratio achieved with different cache sizes.

We further studied the retrieval latency with different cache sizes and algorithms. Since the latency will be continuously gauged and fed into our congestion control module (Sec. 3.1). We evaluate the average per-block latency, which is the time at which a content block is requested by the congestion controller to the time at which the block is returned. Fig. 9 plots the cumulative distribution function (CDF) with different cache sizes. We can see larger cache size achieves smaller latency due to increased local hits. But the improvement becomes marginal when the cache size reaches a certain size. Fig. 10 plots the CDF of three different algorithms introduced in Sec. 4.2. "ICD2D-congest" and "ICD2D-non-congest" have much smaller latency compared to "olsr-congest" due to in-network caching. The results echo our observations in Sec.4.2.

4.3 Simulation under RPG model

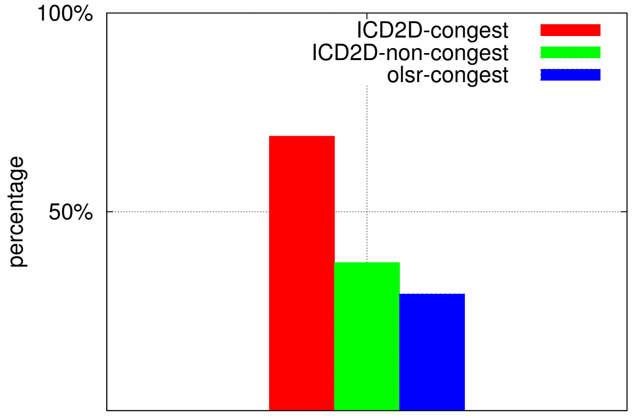

Figure 8: Fulfilment ratio achieved by different algorithms.

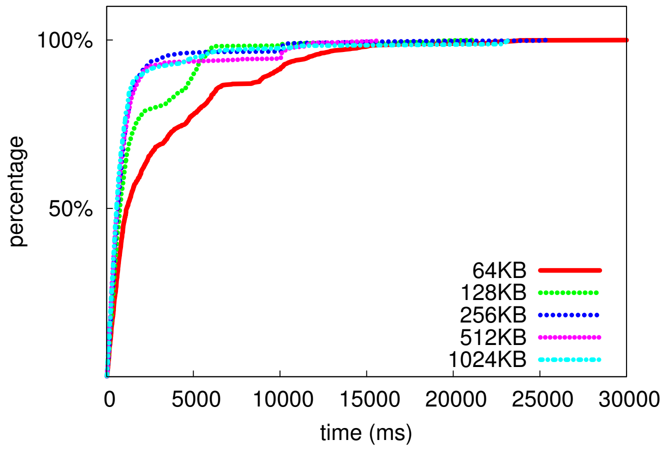

Figure 9: CDF of network latency with different cache sizes.

We finally reran the experiment under a RPG mobility model, where $SDR = ADR = 0.3$. Similar results were witnessed as those under RW pattern. Under the same traffic pattern, the "fulfilment rate" increases to 86.177% from 69.02%. We believe the reason is that soldiers are tending to receive more meta data from other members in the same group, and therefore chances of receiving the contents are higher. We also consider the retrieval latency in this scenario, and the measured result is 1219.46 ms, compared to 1319.5 ms under the RW model. The CDF of the network latency is shown in Fig. 11.

5. CONCLUSIONS

In this paper, we presented a novel design of an information-centric mobile network based on device-to-device com-

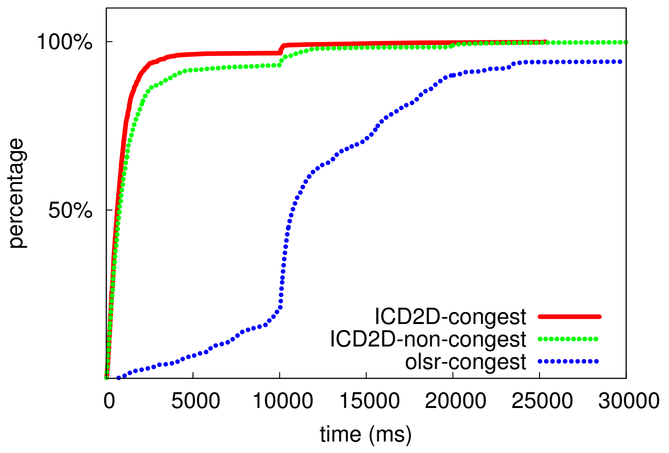

Figure 10: CDF of network latency by different algorithms.

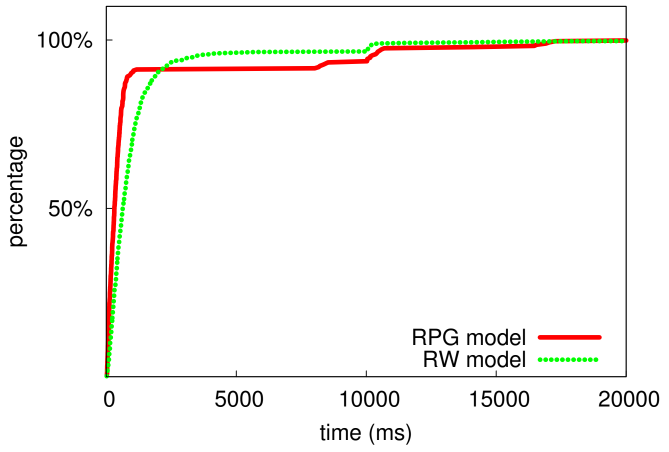

Figure 11: CDF of network latency under RW and RPG model.

munications. The network is distributed and self-organizing. In the system, metadata are disseminated in the network using a publish-subscribe mechanism, which makes users be aware of available contents that are generated in the network in real time. Users can request the contents they are interested by sending request messages and contents are retrieved using a protocol similar to reverse path forwarding. To improve network performance, we developed novel information-centric congestion control and scheduling algorithm to improve network performance. Implementation and evaluations on NS3 demonstrate the superiority of our design.

Acknowledgement

This work was supported in part by the National Science Foundation (NSF) under Grants ECCS-1547294, CNS-1264012, and CNS-1262329 and the Defense Threat Reduction Agency (DTRA) under Grant HDTRA1-13-1- 0030.

6. REFERENCES

[1] V. Jacobson, D. K. Smetters, J. D. Thornton, M. F. Plass, N. H. Briggs, and R. L. Braynard, “Networking named content,” in *Proceedings of the 5th international conference on Emerging networking experiments and technologies*. ACM, 2009, pp. 1–12.

[2] T. Vu, A. Baid, Y. Zhang, T. D. Nguyen, J. Fukuyama, R. P. Martin, and D. Raychaudhuri, “Dmap: A shared hosting scheme for dynamic identifier to locator mappings in the global internet,” in *Proceedings of the 2012 IEEE 32Nd International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2012, pp. 698–707.

[3] T. Koponen, M. Chawla, B.-G. Chun, A. Ermolinskiy, K. H. Kim, S. Shenker, and I. Stoica, “A data-oriented (and beyond) network architecture,” in *ACM SIGCOMM Computer Communication Review*, vol. 37, no. 4. ACM, 2007, pp. 181–192.

[4] G. García, A. Beben, F. J. Ramón, A. Maeso, I. Psaras, G. Pavlou, N. Wang, J. Sliwinski, S. Spirou, S. Soursos *et al.*, “Comet: Content mediator architecture for content-aware networks,” in *Future Network & Mobile Summit (FutureNetw), 2011*. IEEE, 2011, pp. 1–8.

[5] M. Varvello, I. Rimac, U. Lee, L. Greenwald, and V. Hilt, “On the design of content-centric manets,” in *Wireless On-Demand Network Systems and Services (WONS), 2011 Eighth International Conference on*. IEEE, 2011, pp. 1–8.

[6] M. Heissenbüttel, T. Braun, M. Wälchli, and T. Bernoulli, “Optimized stateless broadcasting in wireless multi-hop networks.” in *INFOCOM*, 2006.

[7] A. Siddique, A. M. Hanashi, I. Awan, and M. Woodward, “Performance evaluation of dynamic probabilistic flooding using local density information in manets,” in *Network-Based Information Systems*. Springer, 2007, pp. 288–297.

[8] M. Amadeo, A. Molinaro, and G. Ruggeri, “E-chanet: Routing, forwarding and transport in information-centric multihop wireless networks,” *Computer Communications*, vol. 36, no. 7, pp. 792–803, 2013.

[9] K. Cho, M. Lee, K. Park, T. T. Kwon, Y. Choi, and S. Pack, “Wave: Popularity-based and collaborative in-network caching for

content-oriented networks," in *Computer Communications Workshops (INFOCOM WKSHPS), 2012 IEEE Conference on.* IEEE, 2012, pp. 316–321.

[10] H. Wu, J. Li, T. Pan, and B. Liu, "A novel caching scheme for the backbone of named data networking," in *Communications (ICC), 2013 IEEE International Conference on.* IEEE, 2013, pp. 3634–3638.

[11] S. Saha, A. Lukyanenko, and A. Yla-Jaaski, "Cooperative caching through routing control in information-centric networks," in *INFOCOM, 2013 Proceedings IEEE.* IEEE, 2013, pp. 100–104.

[12] J. M. Wang, J. Zhang, and B. Bensaou, "Intra-as cooperative caching for content-centric networks," in *Proceedings of the 3rd ACM SIGCOMM workshop on Information-centric networking.* ACM, 2013, pp. 61–66.

[13] *Named Data Networking*, http://www.named-data.net.

[14] H. Xie, G. Shi, and P. Wang, "Tecc: Towards collaborative in-network caching guided by traffic engineering," in *INFOCOM.* IEEE, 2012, pp. 2546–2550.

[15] L. Saino, C. Cocora, and G. Pavlou, "Cctcp: A scalable receiver-driven congestion control protocol for content centric networking," in *Communications (ICC), 2013 IEEE International Conference on.* IEEE, 2013, pp. 3775–3780.

[16] S. Arianfar, J. Ott, L. Eggert, P. Nikander, and W. Wong, "A transport protocol for content-centric networks."

[17] G. Ausiello, *Complexity and approximation: Combinatorial optimization problems and their approximability properties.* Springer Science & Business Media, 1999.

[18] M. Stoffers and G. Riley, "Comparing the ns-3 propagation models," in *Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS), 2012 IEEE 20th International Symposium on.* IEEE, 2012, pp. 61–67.