# DragonNet: A Robust Mobile Internet Service System for Long-Distance Trains

Fung Po Tso, *Member*, *IEEE*, Lin Cui, Lizhuo Zhang, Weijia Jia, *Senior Member*, *IEEE*, Di Yao, Jin Teng, and Dong Xuan, *Member*, *IEEE*

**Abstract**—Wide range wireless networks often suffer from annoying service deterioration due to ever-changing wireless environments. This is especially the case with passengers on long-distance trains (LDT, such as intercity, interprovincial, and international commuter trains) connecting to the Internet. To improve the service quality of wide-area wireless networks, we present the DragonNet system and protocol with practical implementations. The DragonNet system is a chained gateway that consists of a group of interlinked DragonNet routers running the DragonNet protocol for node failure amortization across the long stretching router chain. The protocol makes use of the spatial diversity of wireless signals when not all spots on a surface see the same level of radio frequency radiation. In the case of an LDT of around 500 meters, it is highly possible that some of the DragonNet routers in the gateway chain still see sound signal quality when the LDT is partially blocked from the wireless Internet. The DragonNet protocol fully utilizes this feature to amortize single-point router failure over the whole router chain by intelligently rerouting traffic on failed ones to sound ones. We have implemented the DragonNet system and tested it in real railways over a period of three months. Our results have pinpointed two fundamental contributions of the DragonNet protocol. First, DragonNet significantly reduces the average temporary communication blackout (i.e., no Internet connection) to 1.5 seconds compared with 6 seconds without the DragonNet protocol. Second, DragonNet nearly doubles the aggregate system throughput compared with gateway without running the DragonNet protocol.

**Index Terms**—Long-distance train, mobile Internet, random failure, cascading failure, DragonNet

## 1 Introduction

Witnessing the tremendous popularity of Wi-Fi-capable devices such as laptops, netbooks, and smartphones, rail operators are rushing to deploy high-speed wireless networks in a bid to lure potential passengers to travel by railway [1]. A study revealed that 72 percent of business travelers were more likely to use trains than cars or airplanes if Wi-Fi access was available on trains. Among them, 78 percent of these business travelers would actually use Wi-Fi access if it was made available on trains [2].

Existing infrastructures for providing Wi-Fi to Internet access are realized by relaying WLAN traffic via a cellular network [3], [4], satellite [5], trackside WiMAX [6], or leaky coaxial cable (LCX) [7] to the backbone network. However, there are still some barriers that hinder the use of these technologies. For example, satellite communications are not ideal for high-speed access to trains since satellite links have limited bandwidth and long roundtrip times (RTT). WiMAX access creates an enormous financial burden for the large-scale installation of trackside WiMAX APs and equipment maintenance thereafter, and so does the LCX. On the contrary, the cellular-based infrastructure takes advantage of an existing cellular architecture for reducing the deployment cost. However, handoffs between base stations and drastic fading phenomena can easily cause severe deterioration in signal strength of a certain client device to an unacceptable level, resulting in degraded network performance [8].

*Is there any method we can use to handle the downgraded cellular performance given the characteristics of a long-distance train (LDT)?* First, we may take notice of some everyday practices. Sometimes we may have very poor cellular signals in our office. So, we get out of the office, walk some distance along the corridor, until we get perfect signal strength. The LDT is just like the corridor, except that the LDT is much longer. It is a corridor 500 meters long!

In order to answer the question, we conducted some experiments on a train with velocity ranging from 0 to 100 km/h to see whether significant diversity in signal and networking performance exists among mobile nodes at different locations. The results shown in Fig. 1 clearly demonstrate that not only do the two nodes associated with the same operator's network have significant throughput diversity between them, but they also exhibit strong temporal throughput diversity when they are next to each other. This is because the radio condition varies rapidly at different geographical locations (i.e., spacial diversity). As a consequence, the channel quality seen by each node is very likely to be different from the other due to the distinct radio environment they have. It naturally follows that it is advantageous to exploit the channel diversity along the LDT, so that nodes suffering from a deteriorated or failed link condition can be efficiently mitigated through off-loading to neighboring nodes that have better links.

- *F.P. Tso is with the School of Computing Science, University of Glasgow, Lilybank Gardens, Glasgow G12 8RZ, United Kingdom. E-mail: posco.tso@glasgow.ac.uk.*
- *L. Cui, L. Zhang, W. Jia, and D. Yao are with the Department of Computer Science, City University of Hong Kong, 83 Tat Chee Avenue, Kowloon Tong, Hong Kong. E-mail: lincui2@student.cityu.edu.hk.*
- *J. Teng and D. Xuan are with the Department of Computer Science and Engineering, The Ohio State University, 2015 Neil Avenue, Columbus, OH 43210. E-mail: {tengj, xuan}@cse.ohio-state.edu.*

*Manuscript received 25 Mar. 2012; revised 27 July 2012; accepted 16 Aug. 2012; published online 5 Sept. 2012.*
*For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2012-03-0154. Digital Object Identifier no. 10.1109/TMC.2012.191.*

1536-1233/13/$31.00 © 2013 IEEE Published by the IEEE CS, CASS, ComSoc, IES, & SPS

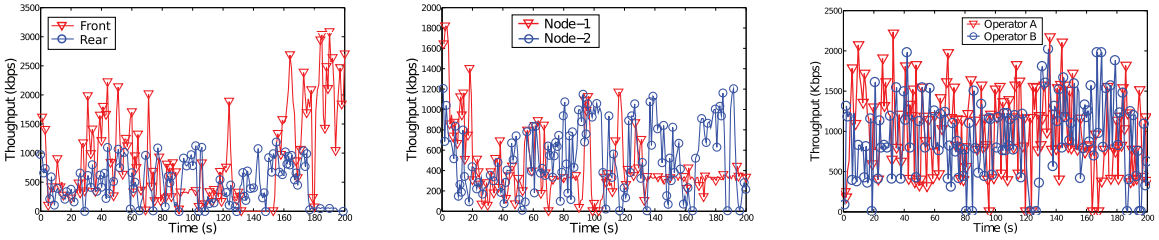



Fig. 1. Channel diversity. Spacial channel diversity for (a) front and rear nodes; temporal diversity for (b) two adjacent nodes of the same operator's network, and (c) two adjacent nodes between two different operators.

Along this line, we present DragonNet, a gateway 500 meters long that aims at handling single-point failure gracefully. DragonNet is formed by a DragonNet router (or D-router; we will use D-router and node interchangeably thereafter in this paper) chain running through the whole length of LDTs. (A typical LDT in China consists of 25 carriages, about 500 meters long. LDT in other countries varies; some are longer, while some are shorter [9].) Each individual D-router in the chain relays local Wi-Fi traffic to the wide-area cellular Internet. The long D-router chain spanning the LDT makes single-point failure manageable. With DragonNet, the failing D-routers can still connect with the Internet through D-routers located in areas of good signal quality. Therefore, communication degradation or even failures at a certain section of LDT can be efficiently amortized. Based on thorough experimental tests, we show that DragonNet can derive significant benefits by exploiting the long stretching feature of LDT. DragonNet operates within cellular coverage; however, cellular wireless is currently the best readily available infrastructure providing high-speed wireless broadband access, and its coverage has been expanding rapidly in recent years. While there is no single best solution for Internet access on LDT, DragonNet can be used as a good complement to other technology, for example, satellite Internet access, once the LDT is out of cellular coverage. Individual passengers may choose to surf the Internet via direct 3G/4G connection, but their cellular devices are not robust against temporary service degradation caused by single-point failure.

DragonNet has been proposed for LDT initially, but it can also be easily extended to other mobile-chained transport systems, such as chained vehicles (buses and trucks), chained ships, and so on. Nevertheless, some may argue that the research issues solved in this paper are not novel because the problem is merely a standard mesh routing problem. In fact, the routing problem that DragonNet intends to solve is essentially different from that of the wireless mesh network (WMN). The WMN focuses more on network formation, while DragonNet concerns itself more with failure recovery. The difference in their starting points dictates that they will employ different approaches and implementations. The mesh routing protocols deal with link failures among mesh routers and mesh clients such that new paths can be established between the source and destination. In comparison, DragonNet primarily copes with external cellular link failures through temporal and spacial networking opportunities so as to maximize the Internet connection time. To this extent, we think DragonNet is a special WMN with two distinct features. The first is the position of the D-routers. They are aligned along a line, which is not typical in a WMN. The second is that DragonNet has special types of failures. We can use the information about the failure to make the failure recovery more efficient. Knowledge of the failure type is not common in WMN, as it normally assumes ad hoc or random link failure. DragonNet is also fundamentally different from the delay-tolerant network (DTN) architecture. DragonNet is devised to provide noninterruptive service for the users, while services provided from a DTN may experience tremendously longer delay.

In short, our contributions in this paper are threefold:

- We have designed and implemented the DragonNet protocol to adaptively form a narrow and long stretching network and efficiently manage both internal and external (cellular) network link failures and achieve high protocol stability.
- We have built the D-router from scratch for running the DragonNet protocol. Though it only accepts USB-interfaced UMTS/HSPA modems for cellular network access for now, it can be easily extended to future LTE access with only a small-scale driver update.
- We have intensively tested the DragonNet prototype on two railways in Hong Kong. The performance results confirm that DragonNet can significantly amortize individual cellular link failure onto the whole chain and give higher aggregated system throughput.

The remainder of this paper is organized as follows: Section 2 presents a survey of related work. We present the design rationale and constraints in Section 3. In Section 4, we give an introduction of the DragonNet architecture. We then describe the DragonNet protocol and its operation in Section 5. Implementation details are presented in Section 6. In Section 7, we extensively evaluate the performance of DragonNet and its supporting protocols. We present the results of a real DragonNet for different applications in various scenarios. Section 8 proposes and discusses failure prediction as a means to enforce proactive counter failure measurements. Section 9 concludes this paper.

## 2 Related Work

Much research work has been done on the integration of mobile networks and the Internet. Aida and Kambori [10]

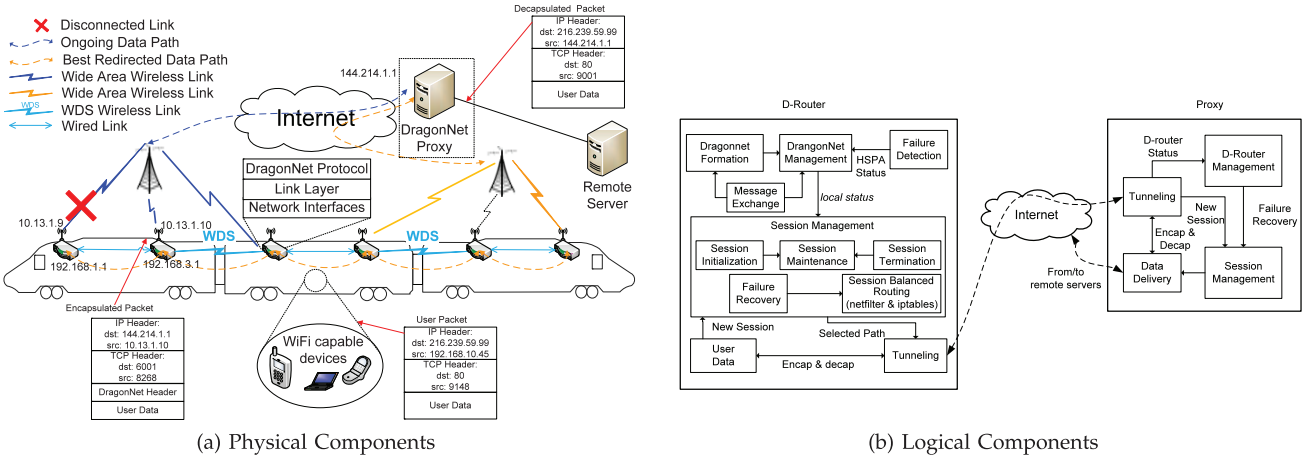
proposed utilizing heterogeneous wireless links that can provide both a continuous slow connection and an intermittent fast connection for broadband access service on a train. Lannoo et al. [11], [12] proposed extensions to the Gavrilovichs [13] moving base stations model. The authors argue, just as in [14], that frequent handoffs greatly reduce the bandwidth available to fast moving users. Consequently, they propose using radio-over-fiber to feed base stations along the rail track. Unlike in the Gavrilovichs model, there are no moving base stations; instead, there is a fiber-distributed antenna network. Ishizu et al. [7] observed that LCX has been used throughout Japan for radio communications on trains; thus, they proposed an architecture for communications on "bullet trains" that consists of a base station with an Ethernet interface and a mobile device. Bianchi et al. [15] stated that it may be expensive to wire a train for network access and rewiring may be needed every time the train is reconfigured. Therefore, they proposed using IEEE 802.11 to construct a wireless network between the train cars. Trains can be connected to the Internet via a satellite link [5]. But, a satellite link has substantially high end-to-end delay. Aguado et al. [16] presented a network architecture based on WiMax for use in railway environments because it can provide mobility support at speeds up to 500 km/h. Kumar et al. [6] introduced an architecture called SWiFT. This architecture consists of IEEE 802.11e access points within train carriages for the on-train network, IEEE 802.16m base stations at the trackside for the access network, and an optical backbone for linking the IEEE 802.16m base stations to the global Internet. Luglio et al. [17] proposed the Noordwijk TCP protocol to resolve obstructions of the line of light in state of the broadband coverage over the railways. However, none of the published works so far provide a solution on the entire infrastructure for a network on a train that can be flexibly implemented in the real world. More relevant to our work are the train-to-wayside communication systems proposed by Bergs et al. [18] and Pareit et al. [19]. The authors proposed a train-to-wayside communication system that manages multiple heterogenous wireless links (combining HSPA with Wi-Fi, satellite, etc.), providing scheduling and transparent IP mobility among these links. However, our work complements their work as we optimize a single connection type (the HSPA link) by using multiple modems.

When a mobile client is in a soft handoff state in wireless networks, a user is connecting simultaneously to more than one cell and, consequently, uses more resources. On the other hand, studies show that users' movements in a cellular network can be predictable and such predictions can help optimize resource allocation. Most of the handoff prediction schemes are based on either the estimate of current position and mobility speed or usual movement patterns [20], [21], [22]. In [20], the authors used software mobile agents to collect users' roaming history to predict upcoming handoff events. Liang and Haas [21] predict the future location by exploiting a dynamic Gauss-Markov model applied to the current and historical movement data. Karimi and Liu [22] utilized trajectory prediction on paths followed in the recent past and on the spatial information of the deployment environment. There are also some prediction schemes that estimate future RSSI to predict the probability of handoff [23], [24]. However, the RSSI prediction schemes can hardly be applied to a cellular network due to complicated geographical environment and weather conditions.

## 3 Design Rationale

*Failure Classification.* Having a long stretching gateway chain, the patterns of connection failures that occur during mobility are completely different from that of a single gateway. We classify the failures into two types: random failures and cascading failures. Random failure means that a node is temporarily screened from the cellular wireless signal in an unorganized manner. For example, hardware fault and temporary signal degradation belong to random failures because they are unpredictable. Cascading failure occurs when many D-routers close in location are temporarily out of coverage consecutively; for instance, while a train is traversing through a (physical) tunnel and cellular coverage black spot. Handoff is commonly believed to happen in the cell boundary while a mobile device is entering the new cell. Given the chain structure of DragonNet, it is natural to anticipate cascading failure during handoff. However, in our field test, we found that handoff does not always happen at the same location because of two reasons. First, the cell boundary varies due to cell breathing. Cell breathing is a common phenomenon in a code-division multiple access (CDMA) cellular network, in which, when a cell becomes heavily loaded, it decreases the coverage of the geographical area, and mobile users at the boundary will then be handed over to the more lightly loaded neighboring cells. Second, a handoff decision is made by the base station and the user device has no knowledge of when handoff is to be carried out. According to our definition, we include handoff in the random failure. As a result, the primary challenge of DragonNet is to efficiently manage these failures so they will not affect ongoing traffic associated with failed cellular connections and evenly spread traffic sessions across all available D-routers. We define a session as the period of time a user is using Internet service(s) and all of the traffic flows incurred during this period of time. The user session begins when the user initiates Internet connection(s) and ends when the user terminates all Internet connections. To overcome the challenges, we devised the DragonNet protocol with a session-based rerouting algorithm to support DragonNet.

*Multiple Layers of NAT.* Apart from failure management and load balancing, DragonNet also needs to deal with NAT transparency. In DragonNet, the D-router creates a NAT for all associated wireless clients and the HSPA network creates another NAT for all D-routers. To this extent, all wireless clients have to traverse at least two layers of NAT to reach servers. Under rerouting, if failed D-routers divert their sessions to other D-routers, servers would simply reset all service sessions because they are perceived as new service requests. This example can be visualized in Fig. 2a, where the traffic leaving interface 10.13.1.9, for instance, is switched to interface 10.13.1.10, but the server treats NATed IP address of interface 10.13.1.10 as a new comer and thus creates a new service session for it. To solve this problem, we introduce the




(a) Physical Components (b) Logical Components

Fig. 2. DragonNet architecture.

proxy for ensuring the service transparency between the end users and servers.

*Design Constraints.* We also list the practical constraints that DragonNet has to satisfy to be practically feasible for LDT besides technical challenges as follows:

- C1: Client devices are off the shelf and cannot be modified.
- C2: Any operation should be transparent to users so that they are not aware of having service interruptions due to temporary network blackout.
- C3: Algorithms and protocols should be feasible to be implemented into commodity Wi-Fi APs.

Constraint C1 is critical to ensure user-friendliness and cost-efficiency. Users are generally reluctant to replace their mobile devices because they have already spent a lot of money to get one. Meanwhile, users are usually loath to install extra softwares and then go through a complicated configuration procedure for receiving Internet services. DragonNet makes sure that users can gain access to mobile Internet as easily as conventional Wi-Fi. Constraint C2 underscores the key idea of DragonNet, transparency. Generally speaking, it means users carry out their Internet-related tasks or services as usual during temporary services blackout of the associated APs, and DragonNet takes care of service recovery without the user intervention. To be more specific, transparency guarantees that both client devices and servers need not modify their service protocols and are blind to DragonNet operations, but users should still get served as usual. Constraint C3, talks about feasibility of the solutions in real-world networking elements. DragonNet's core protocol runs in a set of conventional APs, i.e., D-routers, which collaborate closely with facilitate DragonNet services.

## 4 DragonNet Architecture

In this section, we present the architecture and main components of the DragonNet system along with its supporting protocols.

The DragonNet architecture is depicted in Fig. 2. The DragonNet system consists primarily of a chain of the D-routers that bridges local Wi-Fi LAN to the outside cellular wireless network. The bus-like structured topology can be perceived like a living Dragon. The support protocol coordinates D-routers to perform traffic rerouting to amortize connection degradation or failure at a certain point. Similar to the link-state protocol, the DragonNet protocol collects and exchanges local D-router's status with neighbors regularly. Routing decision is made upon receiving status metrics from neighboring D-router, particularly when failure reports are received.

The D-router has both local and wide-area wireless networking interfaces. The local interface provides wireless access to local users through Wi-Fi, while the wide-area wireless interface maintains the connection with cellular Internet through the modem (a commodity USB dongle). Since the modem is driven by, and interacts with the cellular network, via a standard set of AT commands [25], each D-router is able to associate with a variety of wide-area wireless technologies such as UMTS, HSPA, and future LTE (4G) with a suitable modem. Hence, the DragonNet can be attached to either one or a mixture of these technologies. We primarily choose HSPA as it is currently the most dominant wide-area broadband technology. We also advocate using the mixture of different operators' cellular network to further increase temporal network diversity. We refer to wide-area interfaces as HSPA modems thereafter in this paper.

In practice, the IP addresses of a wide-area interface assigned by the wide-area operator in many cases are in a private range of addresses (e.g., 10.13.1.1). Requests from local users are directed to the D-router and further to the Internet. Based on a particular scheduling policy, a D-router selects a given interface, for example, the neighboring D-router or Internet, for each packet or request. Once the wide-area network interface is selected, client originated packets are source NATed, i.e., both the IP address and port are mapped, using the IP address and port of the wide-area interface. To the external world, the D-router appears as a NAT box. Therefore, the remote server sees requests or packets from the mobile user as from D-router. D-router deNATs packets returning from the remote server and forwards them to the appropriate mobile user.

On the other end of DragonNet, there is a proxy enabling service transparency to servers. As packets originating from a certain client device usually reach servers through

different paths, i.e., cellular interfaces, due to rerouting decisions made by individual D-router, this rerouting decision will force all packets, including TCP packets, to take another path to the destination servers while ongoing D-router undergoes blackout. To this extent, TCP session will be torn down and reset that eventually results in service interruptions to end users. To get rid of service suspensions, we introduce a proxy between DragonNet and servers. Under this circumstance, D-routers first redirect all packets to proxy which then re-encapsulates packets as if they are originated from proxy. Upon receiving response from the servers, the proxy incorporates packets with actual destination addresses and forwards them back to wireless clients. The proxy needs not to be a part of operator's network. In our test bed, we have two proxies located in the university and a commercial data center, respectively.

## 5 DragonNet Protocol

As we discussed above, the DragonNet protocol has two goals. First, it amortizes random and cascading D-router failure onto the whole D-router chain. Second, it aggregates and balances bandwidth along the whole D-router chain.

### 5.1 Operations

The individual D-router's hardware failure may divide the D-router chain into two or more smaller D-router chains. Such division will collapse the DragonNet if centralized management is adopted. But with decentralized management, smaller D-router chain will adaptively form smaller DragonNets to continue providing Internet services to end users. To support the decentralized operation, there are four types of control messages exchanged among D-routers and proxy in the DragonNet protocol:

- *HELLO Message*. Carrying the *local status* information, this message is used for DragonNet formation during the initialization and connection maintenance phase. It is broadcast periodically but any change of the *local status* of the D-routers also triggers an immediate broadcast/unicast of this message. The Hello message is only exchanged among D-routers and restricted to one hop. If a D-router does not receive HELLO from neighbor before time-out (currently 30 seconds), the D-router assumes that the neighbor has hardware failure and partitions the network into smaller DragonNets (later will join back the network again if the D-router sees HELLO from neighbor again).
- *NOTIFY Message*. This message is used to notify proxy about the failure of the D-routers' cellular interface. The failed D-router will broadcast the NOTIFY message to both previous and next neighbors. When this message is received, the neighbors would relay it to the proxy immediately if their cellular interfaces are available; otherwise they just forward it to the next D-routers. Upon receiving this NOTIFY, the proxy will update its local mapping table accordingly.
- *CAscading FAilure (CAFA) Message*. This message is sent out to notify the next D-Routers about the presence of cascading failure when cascading failure is detected.
- *Keep-ALIVE Message*. This message is sent to proxy periodically when there is no traffic on the cellular connection. It acts as a heartbeat for D-router to notify the proxy of its aliveness and prevents time-out of possible network operator's NAT entries between D-routers and proxy, as discussed in Section 3.

To retain a certain level of reliability in the delivery of the messages, if one does not receive an ACK message from the neighbor, it will retry up to 10 times before triggering error handling procedures such as rerouting (assuming neighbors HSPA is down).

On top of these messages, the DragonNet protocol performs the following operations to manage DragonNet and its failures:

- *Local Status Updates.* D-routers periodically broadcast HELLO message to discover their previous and next neighbors. HELLO message should only reach one hop away to avoid message flooding over the DragonNet. On the other hand, D-routers on the chain listen for HELLO messages from interconnecting interfaces. Once Hello messages are received, D-routers bind the arrival interfaces with previous and next D-routers using unique IDs extracted from received Hello messages.
- *Session Creation and Routing.* When a new session joins, the host D-router selects a route for it based on HELLO exchanged with neighbors. The decision can be either to (logically) tunnel this session to proxy directly or to forward this session to neighbor node. If latter option applies, then the procedure should repeat until one of the D-routers (logical) tunnels the new session to the proxy. Once the route is identified, assuming all D-routers are healthy, all packets belonging to this session should follow this route. The path selection processes will be discussed in greater details in Section 5.2.
- *Failure Detection* detects HSPA's backhaul link condition and produces HSPA link status for each D-router. This is a parameter of *local status*, and it is eventually carried by other messages. D-routers instantly broadcast HELLO and NOTIFY message upon detecting failures on HSPA backhaul links. Neighbors then relay the NOTIFY message to the proxy for updating reverse path. This is because when the HSPA interface of a D-router is down and sessions are rerouted to neighbors, the proxy will not know the happening of rerouting before the data are being sent out from client through the affected sessions after rerouting. So the NOTIFY message is needed to let proxy know the changes of rerouting and make some coordination to the affected sessions in a session table. Otherwise, packets from server to client on these sessions will be dropped. Neighboring D-routers examine received HELLO messages and determine which type of failure it should trigger. If the behavior of cascading failure is detected, a CAFA message should be sent out to notify the next D-router.

- *Packet Encapsulation and Decapsulation.* The user packets will be encapsulated at D-routers and forwarded to the HSPA interface through the path according to the session table. Therefore, on the proxy end, upon receiving a packet from a D-router, it decapsulates the packet. If the packet belongs to a new session, the proxy should assign a new free port to this session and then forward the packet to destination. Otherwise, the proxy looks up and forwards the packet through the in-use port number. For the reverse direction, the proxy encapsulates every packet and forwards it to the specified D-router.
- *Heartbeat.* When there are no data flowing through the HSPA interfaces, provided that they are still active and sound, the ALIVE message is sent from their D-router to the proxy periodically to keep the tunneling path alive.

Failure detection is performed by looking at HSPA's physical and logical status. If the kernel reports that the physical link status is set to *down*, this D-router will be omitted until its wide-area interface is fixed or replaced. On the contrary, logical link quality measurement includes periodically collecting physical layer parameters, RSSI (receiver signal strength) and $E_c/I_o$ (ratio of received pilot energy over-the-air to total received energy or the total power spectral density), both of which are good indicators for estimating the dynamic wireless link. Besides, the predictability in LDT's route can greatly help efficiently manage the wireless resources. The LDT route and the position of base stations are known to the DragonNet system. Even if they are unknown, one or two trips are largely enough to feed the system with all necessary information. Given the predictable nature of routes, failure prediction is realized by examining current cell details and comparing them with historical details so as to get DragonNet ready for upcoming predictable failures such as handoffs and cell coverage black holes.

### 5.2 Rules and Conditions for Path Selection

Assume there are $n$ D-routers connected in a bus topology to form a DragonNet. Each D-router has three interfaces. Two of them are configured to connect to previous D-router and next D-router, the remaining one is attached to the cellular HSPA network. Let *loading factor* of $i$th D-router be $W_i$, representing aggregated session weights for all traffic sessions in this D-router. The bit rate of the session is quantized to (0,1] to become a weight, $w$, where 1 is referenced as the upper bound of certain HSPA interfaces of D-routers. The weights are recalculated at the same interval as broadcasting Hello message. Suppose the weight of $i$th session is $w_i$ , and there are $n_i$ ongoing sessions on D-router $i$, so the *loading factor* $W_i$ is defined as $W_i = \sum_{j=1}^{n_i} w_j$. D-router $i$ also keeps the aggregated *loading factors*, $WF_i = \sum_{j=1}^{i-1} W_j$ and $WB_i = \sum_{j=i+1}^{n} W_j$, and the number of available D-routers, $NF_i = (i-1)$ and $NB_i = (n-i)$, in the forward and backward directions, respectively.

Therefore, *unbalance factor*, the difference of maximum and minimum *loading factors* in DragonNet, of $i$th D-router is expressed as $d_i = \max(WF_i/NF_i,\ W_i,\ WB_i/NB_i) - \min(WF_i/NF_i, W_i, WB_i/NB_i)$. This factor is used for measuring the level of load difference for the DragonNet; the larger the value, the higher level of load unbalance exists in the DragonNet.

During operation of the DragonNet protocol, the basic units, D-routers, have to exchange their *local status* regularly. The *local status* is defined as ($W_i$, HSPA status, $WF_i$, $NF_i$, $WB_i$, $NB_i$).

*Session path discovery*. Based on the above setup, DragonNet initializes every D-router accordingly at the boot-up phase. But how does DragonNet accept a new session after then? For each new arrival session, a D-router has three choices: forward it to the previous or next node, or send out through the HSPA modem directly. The decision is certainly not made in an unorganized fashion. Instead, the selection of an outgoing node is made by router $i$ based on a set of rules. First of all, let $D_i$ denote the new *unbalance factor* for D-router $i$ after a new session joined, so we have the following rules for handling it:

- *Rule 1*. If the HSPA interface of D-router $i$ is available to handle the newly joined session and resulting *unbalance factor* for this D-router is less than or equal to the original *unbalance factor* $d_i$, i.e., $D_i \leq d_i$, D-router $i$ takes the privilege to accept this session and (logically) tunnel all packets to the proxy.
- *Rule 2*. If D-router $i$ cannot satisfy rule 1, then it also evaluates $D_i$ of two interconnecting ports and chooses the one producing minimal $D_i$ among the three interfaces it has.

This path discovery process should continue in every D-router until a suitable HSPA interface is identified for the newly joined session.

*Session-based Rerouting*. Besides, when accepting new sessions, traffic rerouting also runs in an infinite loop in the background to periodically check every D-router's loading to determine whether or not to trigger a session balancing rerouting process. For instance, if the HSPA interface of a D-router is temporarily screened out from HSPA coverage, its associated traffic has to be rescheduled. DragonNet should begin the rerouting process on router $i$ only when either of the following conditions is satisfied:

- *Condition 1*. If the difference of the loading factor for D-router $i$ and the rest of D-routers, either forward or backward, is larger than or equal to twice of a minimum session weight of D-router $i$, then DragonNet will reroute some or all of D-router $i$ sessions to the side with smaller loading factors. Larger than or equal to twice of minimum session weight is required because we want to avoid unnecessary rerouting oscillation between two neighboring nodes.
- *Condition 2*. If condition 1 is not satisfied, DragonNet checks condition 2. This condition basically checks whether the loading difference between D-router $i$ and its previous one, D-router $i-1$ is larger than or equal to twice of the minimum session weight of D-router $i$. DragonNet will reroute some or all of D-router $i$ sessions to D-router $i-1$.

- *Condition 3*: Similar to condition 2, but this one is primarily concerned about loading difference between D-router $i$ and its next one, D-router $i+1$. If it is larger than or equal to twice of the minimum session weight of D-router $i$, then DragonNet will reroute some or all of D-router $i$ sessions to D-router $i+1$ if this check is valid.

By following a set of rules and conditions, we derived the DragonNet protocol. Although the session-based rerouting is presented with the assumption that there is no random or cascading failure, it can handle the failures with no or merely a little modification to the protocol. In fact, if a random failure occurs, DragonNet simply sets such D-router's status to NULL and then shifts suspended sessions to other node according to the rules and conditions described above. Cascading failure is a very special case in DragonNet decision-making rules. Recall that cascading failure is due to a LDT traveling through a cellular coverage black hole, such as (physical) tunnels. It can be observed that the last node should stay on for the longest period of time and the first node should resume connection at the earliest time. To this extent, DragonNet deals with this special case by pushing all data to the rear node, while the LDT is entering a (physical) tunnel and then switching to the front node when the LDT is leaving the (physical) tunnel.

Given that network connectivity can change quickly, including from being connected to disconnected and back, some may ask how the rerouting protocols deal with such connection "oscillations". As we have discussed above, rerouting can only be triggered under certain conditions. Consequently, although explicit cellular link failure should trigger an immediate rerouting, resuming from disconnection will not necessarily do this if conditions are not met. Thus, the rerouting protocol can efficiently prevent "oscillation" from happening. Moreover, we think another factor that may affect protocol performance is latency for transfer of packets across the DragonNet chain. Nonetheless, we will show in Section 7 that protocol message processing time and RTT for transferring packets along the chain are negligible so that they will not affect the system performance.

## 6 Implementation

We have built a prototype implementation of DragonNet on Linux Operating System (OS) platform. The DragonNet protocol glues and coordinates all components together to provide noninterruptive mobile Internet services on the LDT. The proxy is implemented as a user-level application that is hosted on a machine on the wired network. The DragonNet protocol on D-routers is implemented on OpenWrt (ver. 8.09.1), a Linux distribution (2.6.26) for embedded devices. The programs that run on D-routers and proxy are both written in C/C++.

We built our D-router-based Broadcom's BCM5354, a 802.11b/g Router System-on-Chip, and Qualcomm's MSM7200 HSPA chipset. This Broadcom's chip is equipped with a processor that is powerful enough to accommodate a light weight OS. For this reason, we have ported a comprehensive OpenWrt embedded Linux OS to the D-router platform. Another reason for using this chip is that this chip comes with a USB2.0 host controller. We take the liberty to integrate it with a USB HSPA modem (USB WiMAX modem was also tested in our lab, but unfortunately there is no WiMAX infrastructure in the city) so as to bridge local wireless group to mobile Internet via this interface. The driver in use is *usbserial*, and it has been modified to periodically check the wide-area interface's health status for every 500 ms by sending Hayes AT commands to the HSPA modem.

The DragonNet's traffic is classified into either control or user traffic. DragonNet restricts control messages to be exchanged within the DragonNet such that no modification of client and server applications is required. The DragonNet performs two routine tasks on user traffic: tunneling between D-routers and proxy and route selection among D-routers. That says, first, DragonNet re-encapsulates packets with new a source and destination address so that they can reach the proxy before going to servers and vice versa; second, DragonNet protocol monitors and changes routes of ongoing traffic periodically for failure recovery and load balance. To achieve the objectives, DragonNet includes Netfilter and iptables to intercept and modify packets for routings. Netfilter and iptables are building blocks of a framework inside the Linux $2.4.\times$ and $2.6.\times$ kernel series. This framework enables packet filtering, network address (and port) translation, and other packet mangling.

Packet encapsulation for tunneling requires adding an extra TCP header to each packet. Therefore, the overall length of the packet always exceeds maximum transmission unit (MTU, usually 1,500 bytes). Rather than fragmenting the large packet into a few smaller packets, we found that large packets are actually discarded by the network operator's network elements. In light of this, as long as MSS plus extra header is larger than MTU, DragonNet sets MSS for packets between wireless clients and D-routers, proxy, and servers to the length of MTU minus an additional header (which is 1,420 bytes for our test cases). Consequently, packets exchanged between D-routers and proxy, though with an extra 40-bytes TCP and IP header plus a 12-bytes proprietary header for tunneling, would not be longer than 1,500 bytes. The header consists of 4 bytes each for client and server's IP address: one byte protocol flag (UDP or TCP) and 4 reserved bytes.

## 7 Evaluation

We have implemented the DragonNet system at the length of 25 nodes. But for the ease of conducting tests on real railways, we have tested DragonNet with four D-routers in two railways. We have also carried out extensive tests for 25 nodes in which each one is embedded with a random and cascading failure generator. As shown in Fig. 4, we can see that processing delay for in-lab and field tests is highly correlated. Even in the most extreme case, the packet processing delay (including queuing delay) does not exceed 900 $\mu$s in both tests. Moreover, we also demonstrate in Fig. 4f that the CDF of RTT for carrying packets over 25-hops D-router is less than 150 ms, which is much smaller than values shown in Fig. 4d where RTT from D-router to server is also included. Fig. 4f also evidences that propagation along the long stretching


Fig. 3. East (blue line) and West (purple line) railways in Hong Kong (based on Google maps). Locations A, B, C, and D are cellular coverage black holes.

DragonNet will not deteriorate the network performance. Furthermore, since the Wi-Fi interface has much greater throughput than HSPA one's, the main throughput bottleneck of DragonNet is on the HSPA interfaces. Through our tests, we found that the maximum throughput of the DragonNet is the aggregate throughput of all HSPA interfaces attached in the network. We observed approximately 2.5 and 13.4 Mbps throughput for four and 25 nodes, respectively. Due to these reasons, we argue that test results for four nodes can safely scale for 25 nodes. In this section, we will present the experimental evaluations of DragonNet.

### 7.1 Testbed and Setup

Fig. 3 describes the field test routes on the East and West railway lines in Hong Kong. The trains in both lines have velocity ranging from 0 to 100 km/h. For the ease of conducting field tests, we have configured the DragonNet to have the length of four battery-powered D-routers across the first four compartments that stretch about 100 m in length, and the D-routers are interconnected with only WDS for these tests. All D-routers were attached to a HSPA network (7.2 Mbps downlink, 2 Mbps uplink) for providing mobile Internet services. Locations A, B, C, and D marked in Fig. 3 are cellular coverage black holes where cascading failures can be observed. We have conducted approximately 20 hours field test during our performance evaluation period.

There are also two proxies and two servers involved in on-site tests. Two identical proxies (running Ubuntu 9.04 with P4 3-GHz CPU and 1-GB RAM) are separately hosted in our lab and a commercial data center to ensure that the result will not be affected by anything specific to one proxy and its path. The two servers with the same configuration as the proxy server are hosted in the university. The client device can be any Wi-Fi-enabled devices. We use netbooks running Ubuntu 9.04 and Windows XP for the tests to make sure OS-dependent issues are dispelled. In mobile tests, we placed a D-router in the middle part of each compartment and interconnected them with WDS as described above. At least one mobile client was associated with one D-router. All end-to-end experiments and measurements have, therefore, occurred in between mobile clients and the servers with proxy servers in the middle.

### 7.2 Protocol Performance

We will evaluate end-to-end delay, including processing delay, for individual D-router and proxy under both lightly and heavily loaded situations.

*Processing Efficiency*. As we have already noted above, we have used the DragonNet composed of four D-routers for those on-site tests for the convenience of coordination among testers. Initially, DragonNet was set to operate with

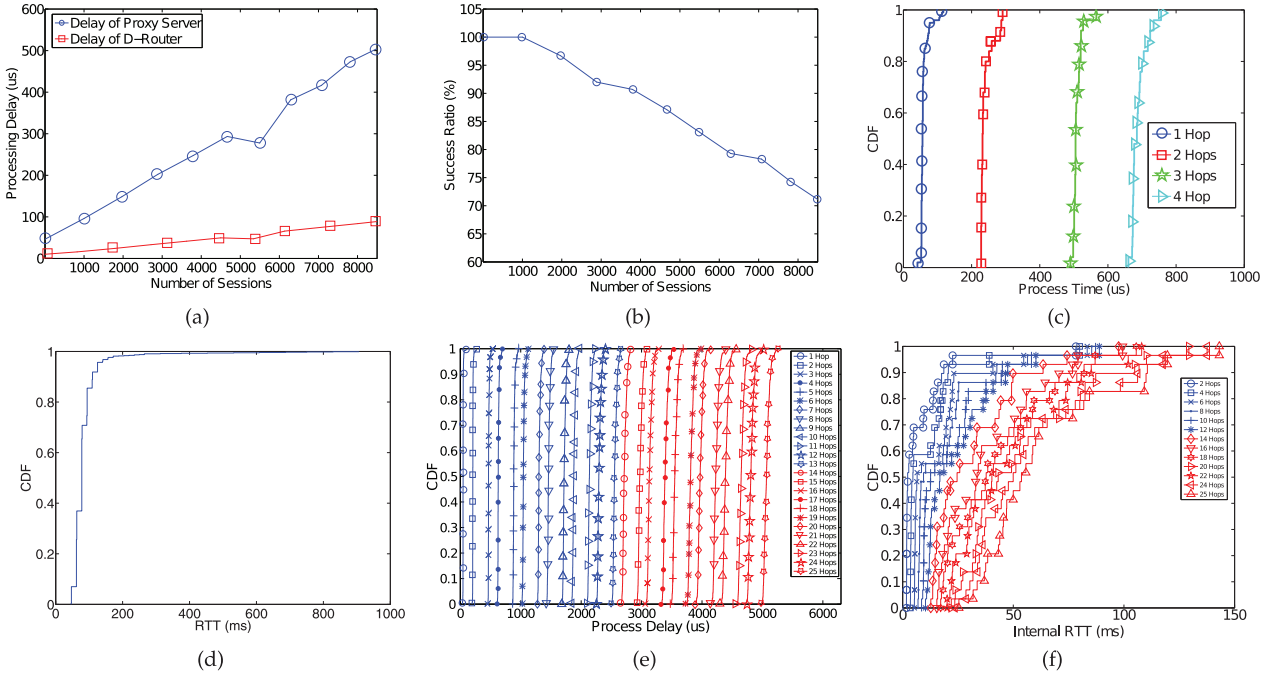



Fig. 4. Field test performance (four nodes): (a) Processing delay under different loadings; (b) success ratio under different load; (c) CDF of processing delay; (d) CDF of round trip time including processing delay introduced by DragonNet protocol and proxy. And in-lab experimental performance (25 nodes): (e) CDF of processing delay, (f) CDF of internal round trip time excluding RTT from DragonNet to servers.

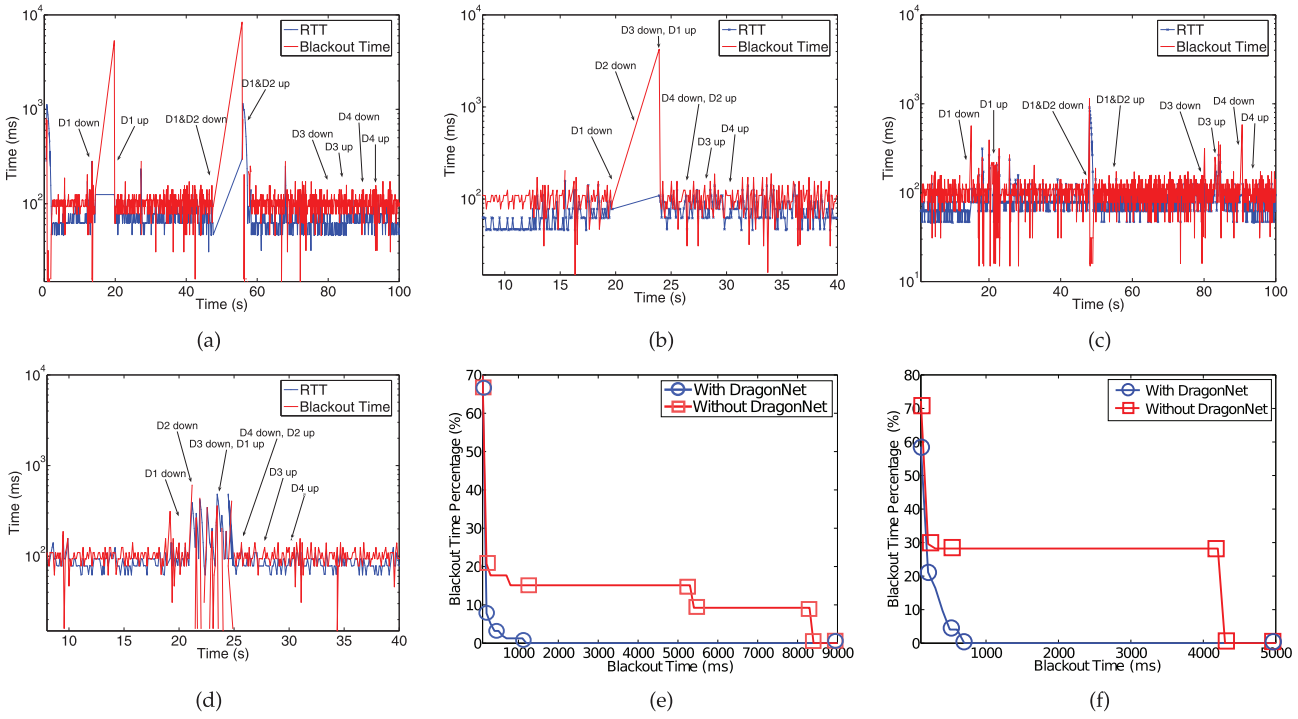



Fig. 5. (a) and (b) are snapshots for random and cascading failure blackout, respectively, without the DragonNet protocol; (c) and (d) are corresponding with the DragonNet protocol; (e) and (f) are distribution of blackout time reduction for random failure and cascading failure, respectively.

boundary of one hop, that says, four chained D-routers can be seen as four DragonNets, to study the performance of DragonNet on an individual D-router. Wireless clients then joined DragonNets through Wi-Fi. The clients start by incrementally adding new sessions (FTP flows), until successful session establishment ratio dropped to an unsatisfactory level. At the same time, we measured the processing time taken by each D-router and proxy for adding a new session. Then DragonNet's boundary was restored back to four to merge the D-routers into one DragonNet for further tests. Apart from process delay for single node, this time we measured processing delay for a session taking multiple hops for studying processing delay of multihopping. Finally, we traced RTT of every packet sent by wireless clients for evaluating overall DragonNet performance.

Fig. 4 shows our filed test results. Fig. 4a reveals processing delays for a D-router and proxy under different loading situations. As we can observe from the figure, the processing time increases linearly from 50 to 550 $\mu$s when new sessions are added incrementally. Even though the heavily loaded one can take a longer time to handle a newly joined session, the process time, which is in the unit of microseconds, is nearly negligible when they are compared with RTT (usually in the magnitude of milliseconds). Fig. 4b unveils the success ratio for a D-router to accept a new session under different traffic loading conditions. The success ratio decreases nearly linearly with increment of the number of sessions. As for a good Internet service system, the success ratio for delivering user's traffic through to servers should be close to a 100 percent, which can be provided by DragonNet when the number of loaded sessions in a D-router is less than 2,000. If we place a D-router in each compartment of a train, and assume that each wireless client can eat up 50-100 sessions, the D-router can have the room to serve about 20-40 passengers in a compartment with robust Internet access service. Fig. 4c unveils the CDF of processing delay of the DragonNet protocol for multi-hopping sessions where a total number of 1,000 sessions were evenly distributed to four D-routers due to the session-based algorithms. The result indicate the larger the number of hops, the longer processing time is required by DragonNet. Moreover, the increment of processing time is nonlinear as CDF lines become flatter and flatter. Nevertheless, we should point out that processing delay given by DragonNet is only the tip of iceberg when they are taken into account of end-to-end delay. Fig. 4d shows the CDF of round trip time for every packet collected during tests. The result is very encouraging for us because most of the time, $\sim 99$ percent, the end-to-end delay is less than 150 ms (RTT $\leq$ 300 ms) which is the favorable value to VoIP.

*DragonNet Blackout Reduction*. One of the main factors that affects end-user experience is the frequent presence of blackout periods in cellular networks. These blackout periods are normally due to interference problems, hardware failures, or loss of connectivity or coverage. We will quantify the performance of DragonNet under blackouts in this section. We define a blackout period as a period of time greater than a given time threshold (500 ms in our test settings) when no data are received.

These tests were done by sending a period of 3,600 seconds worth bursts of data packets back to back from client to the server via HSPA links in each test and measure throughput based on the interarrival times between packets in a burst. For each of these traces, we identified those periods of time where the interarrival time between packets is greater than 1 second as blackout period. The snapshots for random and cascading failure caused blackout for D-routers D1, D2, D3, and D4 with and without the DragonNet protocol in one of the traces are shown in Fig. 5. It can be seen

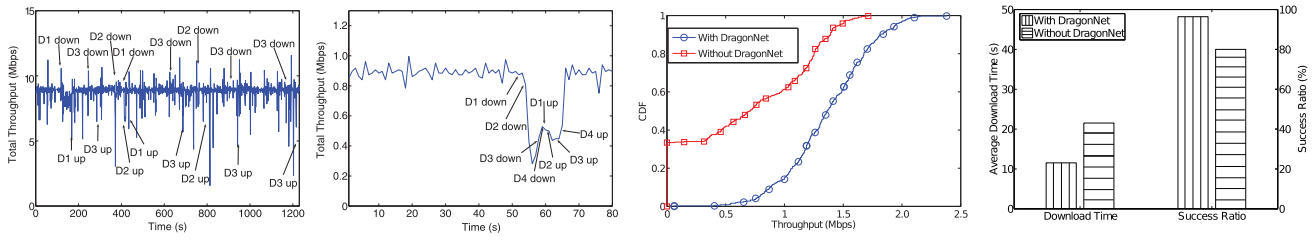



Fig. 6. A typical example of aggregated TCP throughput for DragonNet with (a) random failure; (b) cascading failure; (c) CDF of aggregate HSPA downlink (TCP) throughput with and without the DragonNet protocol; and (e) average HTTP download time with and without DragonNet for front page of cnn.com.

that snapshots, Figs. 5c and 5d, for the one with the DragonNet protocol enabled gives flatter and narrower curves than those without, Figs. 5a and 5b.

As shown in Figs. 5e and 5f, we show the percentage of time spent in a blackout period for a DragonNet system with and without running the DragonNet protocol. From this figure, we can see that the amount of time spent in a blackout can be very significant for a given D-router. This is a natural effect of the fact that under mobile environments, mobile devices suffer frequent cell handoffs, loss of coverage, and sudden disconnections. When comparing the above results with the percentage of time spent by the DragonNet system in a blackout, we see that the DragonNet spends a much smaller portion of the time in a blackout. Thus, the probability that the DragonNet user cannot receive data from any of its D-routers for a period of 1 second or more is almost negligible. Therefore, the DragonNet significantly increases resilience against network failures due to its dynamic rerouting strategy.

*DragonNet Throughput*. Fig. 6 shows a typical time-series example of individual and aggregated throughput for chained D-routers with DragonNet during random and cascading failures. Conventionally, for UDP traffic, all packets flowing through failed D-routers are lost but the session persists. On the contrary, TCP resets its connection if the blackout time is larger than the predetermined timeout threshold. Both are harmful to user experience. As shown in Fig. 6a, the HSPA interfaces of four D-routers failed randomly as time passed, but not all D-routers failed at the same time, while, in Fig. 6b, the D-router failed and recovered consecutively. However, in their without-DragonNet counterparts, the client connections are dropped and the throughput plummeted to zero.

We have also demonstrated in Fig. 6c the CDF distribution of aggregate downlink throughput of all HSPA interfaces for our test system with and without the DragonNet protocol. DragonNet explicitly has a factor of two times higher throughput than the conventional network. Nonetheless, we believe a greater improvement can be achieved if more D-routers are included in DragonNet. In fact, the percentage of improvement varies with respect to the probability of actual link failure and connection blackout. In mobile scenarios we measured in Hong Kong over a period of time, each D-router (i.e., mobile interface) experiences about 10 percent connection blackout time of total journey of 60 minutes. DragonNet takes advantage of channel diversity to relay traffic for broken links. Therefore, it is estimated that the more connection blackouts there are, the greater the improvement in DragonNet will be. Even in such a mobile environment, the impact of highly variable bit rates in DragonNet is not as pronounced as for each individual gateway due to the fact that DragonNet exploits the benefits offered by long stretching geographic channel diversity.

### 7.3 System Performance

In this section, we study the performance of a DragonNet client web browsing's session to illustrate a real user experience in the presence of link failures.

Given the fact that web browsing is the most common Internet service, we adopt web browsing traffic to measure DragonNet's application performance. First of all, we replicated a copy of *cnn.com* front page (dated 9 March 2012) to our test server. This is done to avoid official update so that we can keep our tests fair and consistent. The front page of CNN consists of 89 objects all together of size 1,145 Kbytes. With an average bit rate of 400 Kbps each, the Firefox web browser can often fully download the page within 20 seconds using HSPA access. In light of this, we implemented a Firefox web browser plugin application to randomly refresh it so as to repeatedly download *cnn.com* for these tests. The randomness is chosen in between 30 and 60 seconds to make sure that the time interval is reasonably long to have concurrent access as well as individual access among clients during the tests. Caching is disabled to force web-browser downloading from server when each time *reload* command is triggered. We believe the tests reflect actual behaviors of passengers inside a compartment of commuter train, where they access the Internet in a purely random manner.

In Fig. 6d, we show the average download time taken by the mobile browser for test cases. From this figure, we can clearly see that on average mobile users perceive nearly no disconnection even though their associated D-router has lost connection with the wide-area network.

### 7.4 Performance at Different Times of the Day

The performance of the cellular network varies across at different times of a day in accordance with the number of subscribers in the current cell. For example, cells in CBD (central business district) become crowed during the day and largely underutilized at nights because people squeeze into CDB for work during the day and leave for home at night. Such performance variance can potentially lead to performance degradation of DragonNet system at different time of a day. In light of this, we therefore listed wide-area interfaces' throughput readings along the same route at

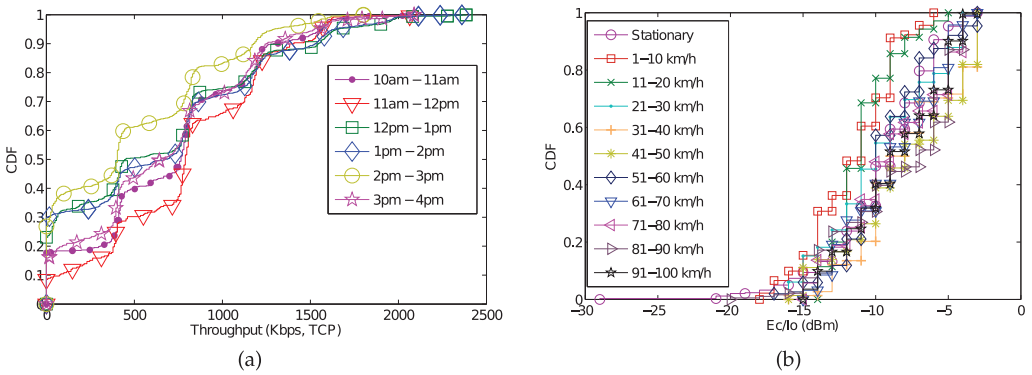



Fig. 7. Measured wide-area throughput (a) at different time of a day (b) for different velocity.

different times in a day to see if such variables could affect DragonNet's performance. Clearly, as shown in Fig. 7a, there is no explicit correlation between throughput at different times of a day because the train traverses through a large span of areas crossing many cells. Nonetheless, the throughput decreases slightly from 12 to 3 pm; it is possibly because people use their cell phone more often during the lunch time than other period of a day. However, the degradation is not too significantly large to lose network connectivities to the wide-area network since DragonNet relies on the network spacial and temporal networking opportunities to achieve robustness along the gateway chain. As long as the network some D-routers are able to maintain network connectivities, other D-router experiencing temporary network outage can leverage networking opportunities of sound ones.

### 7.5 Performance at Different Velocities of the Train

The velocity of a LDT would potentially impose both internal and external impacts on the DragonNet. Internally, the DragonNet periodically detects failures with a control loop. If the train moves at very high speed, such that each single node passes through the coverage black hole before single failure detection cycle, such failures will not be detected. Obviously, the DragonNet protocol can adapt to the failure better if we have a smaller control loop. In practice, we found that the control loop of 500 ms works well. However, if we decrease the control loop time to less than 500 ms, the DragonNet protocol will falsely react to some wide-area interfaces that are experiencing temporal bad signal quality. Nonetheless, as majority of the traffic generated by mobile users is TCP, temporary disruption of TCP connections of 1 second is not noticeable to end users, whereas UDP streaming can be remedied by buffering.

Externally, cellular throughput performance is believed to decrease with increased mobility because cellular throughput performance is affected by cell reselection and routing area update procedures, particularly for fast-moving UEs. Increasing mobility levels will lead to a higher number of cell reselections and routing area updates, and will also increase the amount of packet outage time. To understand the impact of train speed on the system performance, we have captured and analyzed every wide-area interface's signal quality, $E_c/I_o$, with respect to the train's moving speed. The data are split into different mobility groups with interval of 10 km/h per group, as shown in Fig. 7b. We separated signal quality readings at static scenario, i.e., when train stops at stations, as references. Fig. 7b clearly shows that train's moving speed does not affect average wide-area interface performance because the received signal quality does not monotonically decrease with increased mobility level. For example, 80 km/h could have better signal quality readings than 70 km/h group.

## 8 Failure Prediction and Predictability

In previous settings, DragonNet reactively handles all type of network failures. Nevertheless, network outage due to tunnels and cellular coverage black holes is usually fixed in locations and is highly predictable. One or more tracking history can provide DragonNet with sufficient information about these failure locations. If DragonNet can cope with these outage before they actually happen, the DragonNet can provide more resilient and smoother mobile Internet services to passengers. In this section, we present a simple but efficient technique to predict failures that cause cascading failures and handoff events.

Failure spots can be captured by using global positioning system (GPS) [26] receivers installed on the train. Using the GPS to capture network outage locations in fast moving trains is certainly nontrivial. First, a naive use of the off-the-shelf GPS receiver in mobile situations can lead to significant errors due to infrequent measurement report interval and large delay between the satellites and the moving objects [27]. Second, legacy GPS merely tracks and reports location measurement to moving objects, but in DragonNet we need to combine the location readings into the network readings to approximate actual failure locations.

We have examined six GPS tracking history along our measurement routes and found that readings of failure locations do not converge to one specific location with some deviations up to 1 km. We believe that this is due to the delay between a location being reported as failure and the corresponding reading received from GPS. Therefore, we need to calibrate a reference point out of a group of measured points to be used by DragonNet as actual network outage spot. The calibration process consists of two procedures. First, all points that have no neighbors in their 100-m radius range and points with a dilution of precision (DOP) [28] value below minimum accuracy requirement are to be eliminated because they are potentially inaccurate samples. Second, since all remaining

TABLE 1
Prediction Performance

| Attribute | Accuracy | False Positive | False Negative |
|---|---|---|---|
| Coverage Blackholes | 93.5% | 6.5% | 0% |
| Handoff | 62.8% | 29.1% | 51.3% |

readings in the set are close in location with good DOP, we advocate using the idea of center of mass to determine a reference point:

$$R = \frac{\sum m_i r_i}{\sum m_i}$$

where $m$ is defined as weight of each measured point. The actual weight is calculated as $\frac{(5-DOP)}{4}$ because values of DOP are in between $[1, 5)$ (readings equal or greater than 5 will be discarded and should avoid divide by zero problem). In our tests, we found that setting prediction threshold to 5 seconds ahead of predicted failures works well.

In this test, we first enabled a D-router to compute reference failure locations from history traces and then repeated the same routes for 10 times to test the performance of our prediction technique. We use three metrics for accuracy namely, *accuracy*, the percentage of failures at reference locations predicted correctly; *false-positive rate*, the percentage of actual nonoutage mistakenly predicted as outage; and *false-negative rate*, the percentage of actual outage events in the test set wrongly classified as nonoutage. Table 1 shows that our prediction technique exhibits high accuracy for predicting coverage back holes that lead to cascading failures. On the contrary, accuracy of handoff prediction is just above satisfaction, at 62.8 percent with false-negative rate at 51.3 percent out of all measured handoffs. However, predicting handoff is difficult as handoff decisions are independently determined by base station based on channel condition reported by mobile devices. Despite inconspicuous predicting accuracy, the gain of our light-weighted prediction technique is significant as compared to a more sophisticated history-based technique proposed in [29].

## 9 Conclusion

In this paper, we discussed the limitations of the current wireless access systems in the commuter train due to dynamic nature of wireless communication. To that end, we made a case for exploiting the length of LDTs for networking opportunity. We introduced DragonNet, a unique system formed by a chain of D-routers that utilizes multiple healthy cellular wireless links to amortize deteriorated links and thus provides local users with a more reliable access network than which can typically be provided by a single cellular gateway. As a result, the supporting protocol was devised to support DragonNet. The DragonNet protocol manages the DragonNet, detects, predicts, and reacts to failures. With field test analysis, we showed that DragonNet can provide more stable and sustainable data rates for applications such as web browsing. The benefits can all be achieved without requiring users to perform any software or configuration updates on their mobile devices. We will further optimize and evaluate the proactive failure prediction in the next step.

## References


[1] "Wi-Fi—Coming to a Station near You," http://www.railway-technology.com/features/feature1150, 2013.

[2] BBC News, "Wi-Fi May Tempt Train Travellers," http://news.bbc.co.uk/2/hi/technology/3729583. stm, 2013.

[3] Icomera, http://www.icomera.com, 2013.

[4] P. Rodriguez, R. Chakravorty, J. Chesterfield, I. Pratt, and S. Banerjee, "MAR: A Commuter Router Infrastructure for the Mobile Internet," *Proc. ACM MobiSys*, pp. 217-230, 2004.

[5] X. Liang, F.L.C. Ong, P.M.L. Chan, R.E. Sheriff, and P. Conforto, "Mobile Internet Access for High-Speed Trains via Heterogeneous Networks," *Proc. 14th IEEE Int'l Symp. Personal, Indoor and Mobile Radio Comm.*, vol. 1, pp. 177-181, 2003.

[6] R. Kumar K, P. Angolkar, D. Das, and R. Ramalingam, "Swift: A Novel Architecture for Seamless Wireless Internet for Fast Trains," *Proc. IEEE Vehicular Technology Conf.*, pp. 3011-3015, 2008.

[7] K. Ishizu, M. Kuroda, and H. Harada, "Bullet-Train Network Architecture for Broadband and Real-Time Access," *Proc. 12th IEEE Symp. Computers and Comm.*, pp. 241-248, July 2007.

[8] F.P. Tso, J. Teng, W. Jia, and D. Xuan, "Mobility: A Double-Edged Sword for HSPA Networks," *Proc. ACM Int'l Symp. Mobile Ad Hoc Networking and Computing*, pp. 81-90, Sept. 2010.

[9] Wikipedia, "Longest Trains," http://en.wikipedia.org/wiki/Longest_trains, 2013.

[10] H. Aida and S. Kambori, "Effective Use of Heterogeneous Wireless Links in High Speed Railways By Predictive Scheduling," *Proc. Int'l Symp. Applications and the Internet*, pp. 459-462, 2008.

[11] B. Lannoo, D. Colle, M. Pickavet, and P. Demeester, "Radio-over-Fiber-Based Solution to Provide Broadband Internet Access to Train Passengers," *IEEE Comm. Magazine*, vol. 45, no. 2, pp. 56-62, Feb. 2007.

[12] B. Lannoo, D. Colle, M. Pickavet, and P. Demeester, "Extension of the Optical Switching Architecture to Implement the Moveable Cell Concept," *Proc. 31st European Conf. Optical Comm.*, vol. 4, pp. 807-808, Sept. 2005.

[13] C.D. Gavrilovich, "Broadband Communication on the Highways of Tomorrow," *IEEE Comm. Magazine*, vol. 39, no. 4 pp. 146-154, Apr. 2001.

[14] F.D. Greve, B. Lannoo, L. Peters, T.V. Leeuwen, F.V. Quickenborne, D. Colle, F.D. Turck, I. Moerman, M. Pickavet, B. Dhoedt, and P. Demeester, "Famous: A Network Architecture for Delivering Multimedia Services to Fast Moving Users," *Int'l J. Wireless Personal Comm.*, vol. 33, nos. 3/4 pp. 281-304, 2005.

[15] G. Bianchi, N. Blefari-Melazzi, E. Grazioni, S. Salsano, and V. Sangregorio, "Internet Access on Fast Trains: 802.11-Based on-Board Wireless Distribution Network Alternatives," *Proc. 12th IST Mobile and Wireless Comm. Summit*, pp. 15-18, 2003.

[16] M. Aguado, O. Onandi, P.S. Agustin, M. Higuero, and E.J. Taquet, "WiMax on Rails," *IEEE Vehicular Technology Magazine*, vol. 3, no. 3, pp. 47-56, Sept. 2008.

[17] M. Luglio, C. Roseti, G. Savone, and F. Zampognaro, "TCP Noordwijk for High-Speed Trains," *Proc. First Int'l Conf. Advances in Satellite and Space Comm.*, pp. 102-106, July 2009.

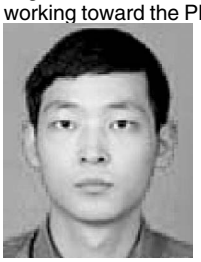
[18] J. Bergs, E.V. de Velde, D. Pareit, D. Naudts, M. Rovcanin, I.D. Baere, W.V. Brussel, C. Blondia, I. Moerman, and P. Demeester, "Design and Prototype of a Train-to-Wayside Communication Architecture," *Proc. Fourth Int'l Conf. Comm. Technologies for Vehicles*, pp. 137-150, 2012.

[19] D. Pareit, E.V. de Velde, D. Naudts, J. Bergs, J. Keymeulen, I.D. Baere, W.V. Brussel, C. Vangeneugden, P. Hauspie, G.D. Vos, I. Moerman, C. Blondia, and P. Demeester, "A Novel Network Architecture for Train-to-Wayside Communication with Quality of Service over Heterogeneous Wireless Networks," *EURASIP J. Wireless Comm. and Networking*, vol. 114, pp. 1687-1499, 2012.

[20] P. Bellavista, A. Corradi, and C. Giannelli, "Resource Allocation Based on Handoff Prediction in WCDMA," *Proc. Vehicular Technology Conf.*, vol. 1, pp. 127-131, 2002.

[21] B. Liang and Z.J. Haas, "Predictive Distance-Based Mobility Management for Multidimensional PCS Network," *IEEE/ACM Trans. Networking*, vol. 11, no. 5, pp. 718-732, Oct. 2003.
[22] H.A. Karimi and X. Liu, "A Predictive Location Model for Location-Based Services," *Proc. Int'l Workshop Advances in Geographic Information Systems (GIS)*, 2003.
[23] P. Bellavista, A. Corradi, and C. Giannelli, "Adaptive Buffering-Based on Handoff Prediction for Wireless Internet Continuous Services," *Proc. First Int'l Conf. High Performance Computing and Comm.*, pp. 1021-1032, 2005.
[24] S.-T. Sheu and C.-C. Wu, "Using Grey Prediction Theory to Reduce Handoff Overhead in Cellular Communication Systems," *Proc. 11th IEEE Int'l Symp. Personal, Indoor and Mobile Radio Comm. (PIMRC)*, vol. 2, pp. 782-786, 2000.
[25] Wikipedia, "Hayes Command Set," http://en.wikipedia.org/wiki/Hayes_command_set, 2013.
[26] Wikipedia, "Global Positioning System," http://en.wikipedia.org/wiki/Global_Positioning_System, 2013.
[27] D. Hadaller, "Mitigating GPS Error in Mobile Environments," technical report, David R. Cheriton School of Computer Science, Univ. of Waterloo, 2008.
[28] "Writing Your Own GPS Applications: Part 2," http://www.developerfusion.co.uk/show/4652/2, 2013.
[29] U. Javed, D. Han, R. Caceres, J. Pang, S. Seshan, and A. Varshavsky, "Predicting Handoffs in 3G Networks," *Proc. Third ACM SOSP Workshop Networking, Systems, and Applications on Mobile Handhelds (MobiHeld)*, 2011.

**Fung Po Tso** received the PhD degree in computer science from the City University of Hong Kong in 2011. He is currently a SICSA research fellow at the School of Computing Science, University of Glasgow, United Kingdom. His research interests include cloud data center (DC) networks, DC management, mobile computing, distributed computing, and cyber-physical systems. He is a member of the IEEE.



**Lin Cui** received the bachelor's degree from Shandong University in 2007 and the master's degree from the Harbin Institute of Technology in 2009. He is currently working toward the PhD degree in the Department of Computer Science, City University of Hong Kong. His general research interests include mobile computing systems and wireless ad hoc/sensor networks.

**Lizhuo Zhang** received the doctoral degree from the Central South University, China. He is currently a senior research associate at the City University of Hong Kong. His research interests include multimedia communication, wireless network, and mobile systems.



**Weijia Jia** received the PhD degree from the Polytechnic Faculty of Mons, Belgium, in 1993. He is currently a full professor in the Department of Computer Science and the director of the Future Networking Center, Shenzhen Research Institute, City University of Hong Kong (CityU). He joined the German National Research Center for Information Science (GMD) in Bonn (St. Augustine) from 1993 to 1995 as a research fellow. In 1995, he joined the Department of Computer Science, CityU, as an assistant professor. He is a senior member of the IEEE.

**Di Yao** received the BSc and MSc degrees from Northeast Dianli University and the National University of Defense Technology in 2005 and 2008, respectively. She is currently a resident advisor at the City University of Hong Kong. Her research interests include the design and application of wireless sensor network systems, system performance measurement, and optimization.

**Jin Teng** received the BS and MS degrees in electronic engineering from Shanghai Jiao Tong University, China, in 2006 and 2009, respectively. He is currently working toward the PhD degree in the Department of Computer Science and Engineering at the Ohio State University. He served as a resident advisor with the City University of Hong Kong from July 2008 to August 2009. His research interests mainly include wireless communication architecture, QoS of wireless networks, network coverage in WSN, and cyberspace security.

**Dong Xuan** received the PhD degree in computer engineering from Texas A&M University in 2001. He is currently an associate professor in the Department of Computer Science and Engineering, the Ohio State University (OSU). He was on the faculty of Electronic Engineering at Shanghai Jiao Tong University from 1993 to 1997. His research interests include distributed computing, computer networks, and cyberspace security. He received the US National Science Foundation CAREER Award in 2005 and the Lumley Research Award from the College of Engineering, OSU, in 2009. He is a member of the IEEE.

▷ **For more information on this or any other computing topic, please visit our Digital Library at** www.computer.org/publications/dlib.