Pareto Optimal Cross Layer Lifetime Optimization for Disaster Response Networks

Harsha Chenji and Radu Stoleru
Department of Computer Science and Engineering, Texas A&M University, College Station, Texas, USA
cjh, stoleru@cse.tamu.edu

***Abstract*—Disaster Response Networks (DRNs) are designed to assist first responders during the recovery period following a large scale disaster. The system lifetime of deployed DRNs is critical to successful recovery, as are performance metrics such as packet delivery delay. In this paper we investigate the Pareto front between system performance and system energy consumption in such DRNs. The latter is further reduced compared to state of art methods by accepting the least possible performance penalty as a tradeoff. We observe that not all nodes in the network may consume or produce data; such relay nodes can be excluded from the routing process to save energy, but at the cost of decreased system performance. The problem is formulated mathematically using Raven as the underlying routing protocol. The Nondominated Sorting Genetic Algorithm II (NSGA-II) is used to obtain the Pareto-optimal points, and the DRN is made to operate at these points. Extensive performance evaluations demonstrate that a 162 minute increase in system lifetime is possible for a 61 minute increase in the packet delivery delay, while the packet delivery ratio remains almost constant.**

I. Introduction

During the aftermath of a natural or man-made disaster, the lack of usable power and communication infrastructure hampers the disaster recovery process. DRNs by design consist of self-powered wireless networking devices that can be deployed in an ad hoc fashion, creating a communications infrastructure in the process. Since these devices are spread over a large area, leading to a sparse network where disconnections are common, DRNs are also disruption and delay tolerant. DistressNet [1] is such a DRN that was developed by the authors. It's contribution is a "fog computing" paradigm, where first responders can produce, consume and share large amounts of data akin to cloud computing, but while the cloud is spread over a large disconnected network. Because of the mission critical nature of the data as well as the scenario, it becomes essential to ensure both energy efficiency and high system performance.

Energy consumption patterns in delay tolerant networks have been studied extensively [2], [3], [4], [5]. Reducing radio usage saves energy, and this can be achieved by either relaying less packets [6] or by discovering contacts [4] optimally. Because of sparse node density and large node inter-contact times [7], it is imperative that a node not miss any contacts. A hardware approach to saving energy involves the use of external hardware that alerts the node when a contact is in range [8].

This paper aims to not just reduce the energy consumption, but to do so in a Pareto optimal manner by quantifying the effect on system performance simultaneously. The energy consumption can be reduced further compared to state of art methods by accepting the lowest possible system performance penalty for the amount of energy to be saved. In other words this is the "ultimate" optimization that can be performed, in the sense that performance is actively traded off for lifetime. Some relay nodes in a DRN may just relay packets (data waypoints in [1]) and not produce or consume data; energy consumption is reduced by completely excluding these nodes from the routing process. The number of nodes to be excluded and the choice of nodes influences the achievable performance, as do protocol specific parameters (e.g. the number of replicas in [9]). In DistressNet, the data flows (source-destination pairs) are not well known and are to be determined subject to some availability constraints - so there is the additional problem of determining these flows before the set of relay nodes can be determined.

This cross layer approach to optimizing system lifetime and performance in a Pareto optimal fashion is formulated mathematically as a dual objective non-linear programming problem. NSGA-II is used to solve the problem and obtain the set of Pareto optimal points. The ability to operate a DRN (DistressNet in this case) at various Pareto-optimal points means that the users (first responders) can actively control the performance/energy tradeoff. .The contributions of this paper are as follows: 1) the first (to the best of our knowledge) framework that is able to discover the Pareto front between system performance and energy consumption, 2) modeling the approach as a dual objective optimization problem that can be extended to other DRNs, 3) an algorithm to estimate the two objectives in DistressNet when Raven [10] is used as the underlying routing protocol, and 4) an improved chromosome generator that shortens the optimization runtime.

II. Motivation

In this section we motivate this paper by analyzing various methods to save energy in a DRN, positing the existence of a Pareto front, and placing these methods along the Pareto front. In a DRN, mobile vehicles are used to mule data to and from static nodes (deployed at collapsed buildings, for example). Being spread over a large geographical area with few resources, the mobility in a DRN is very sparse. As a result, the node inter-contact time is on the order of tens of minutes or even hours: it is typical for a node to spend only about 20% of its lifetime in contact with at least one other node. This quantity is henceforth called the *contact time (CT)*. This means that a node can afford to sleep in the inter-contact time (about 80%) and still not hamper the performance by missing any contacts.

One of the trivial methods of energy saving in a DRN is to not save any at all - each node stays awake all the time. Network performance is maximal, but so is the energy consumed ("A" in Figure 1). One simple optimization is to have the node sleep in the inter-contact time. Either mobility prediction or hardware assistance can be used to wake up the node. For

978-1-4799-3635-9/14/$31.00 ©2014 IEEE

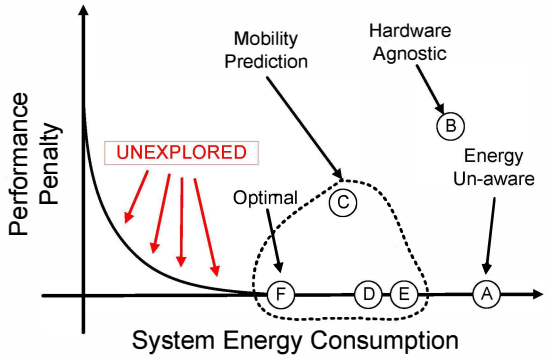


Fig. 1: The Pareto front between system performance and system energy consumption. Points A through F represent state of art approaches. The unexplored stretch of the Pareto front, where energy consumption is reduced by incurring a performance penalty, has not been explored.

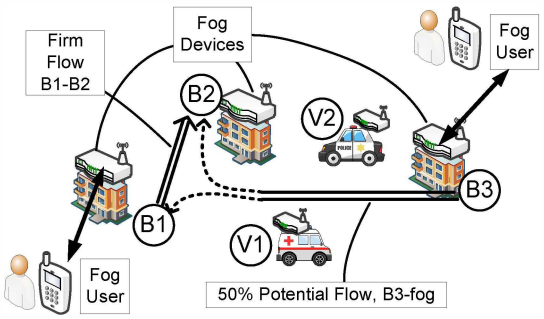


Fig. 2: Illustrating the fog in a deployed DRN. Buildings B1,B2,B3 are Centers (in PDMM parlance) and have a fog device on them. Mobile Agents V1 (ambulance) and V2 (patrol car) travel between the three Centers, and have a wireless router but are not fog devices. Fog users with a smartphone connect to a fog device for access.

example, [8] uses long range auxiliary radios to detect a vehicle and subsequently wake up the node. In such a scheme ("E" in Figure 1), there is a reduction in energy consumed but no loss in performance. These prediction schemes are not completely optimal; replacing it by an oracle [11] ("D" in Figure 1) could possible save a little more energy.

It is important to note that saving energy purely in software (by minimizing the number of transmitted messages at the routing layer) or only on the radio interface (by enabling 802.11 PSM mode) will not save much energy. This is because the energy consumed by the radio interface in a typical WiFi router is much less than that consumed by the base board. Low power sensor networks, however, have the opposite characteristic. Table I compares a WSN mote, a hand held WiFi smartphone and a WiFi router. As we can see, the power consumed by a router's radio (0.72W) can be 5x smaller than the base board (3W). A scheme that aims to minimize the number of relayed messages, as compared to "A", will have low performance but will save a little energy ("B" in Figure 1).

Device	Radio Power Cons.	Base Power Cons.
Sensor	0.06W	0.006W
Smartphone	0.7W	0.2W
Router	0.72W	3W

TABLE I: Power consumption of various device classes. Sensor: based on a 3V Epic mote with 802.15.4. Smartphone - based on 3.7V HTC Evo 4G with 802.11. Router - based on 12V Mikrotik RB433UAH router with 802.11.

Depending on the routing protocol that is used, not all contacts may involve transfer of data. For example, in RAPID [12] a node transfers packets only to a node with higher utility. An optimization that could be performed here is to have a low capacity backhaul link, such as a satellite link, that nodes can use to query the buffer contents of any other node and thus calculate the utility or any similar metric. As a result, a node can infer whether a vehicle has any packets that could be transferred; if not, it can sleep through the contact. Such a scheme ("F" in Figure 1) represents the most energy that can be saved without any performance penalty. The percentage of contact time that involves useful transfer of data in relation to the node lifetime is called the *useful contact time (UCT)*.

Key Idea Given the above background, we investigate in this paper the existence of a Pareto front in a DRN, assuming that a scheme such as "F" is already present (such a scheme does not exist to the best of our knowledge; designing it is a research problem in itself, and not the focus of the paper). The major intuition is that if some nodes can be excluded from the routing process, then those nodes can save energy by sleeping. However there is a performance penalty involved; it is the aim of this paper to quantify this trade off and investigate the feasibility of operating at various points on the Pareto front. It has to be mentioned that knowledge of data flows in the network is essential, and is explained in the next section.

III. Preliminaries and Problem Formulation

In this section we first present the data production and consumption model in DistressNet, followed by a short explanation of Raven, the underlying routing protocol. After these preliminaries, the problem is formulated mathematically. An NSGA-II based solution is proposed to solve this problem, followed by a numerical example to further clarify these concepts.

A. Preliminaries

DistressNet [1], our previous work, is a DRN designed for Urban Search & Rescue first responders. In this system, traditional cloud services like Twitter and Amazon S3 are instantiated in a disconnected DRN called the "fog". Users, in this case the first responders, are able to upload files to the fog, retrieve them later or share with other users. Physically, the fog is comprised of several DistressNet devices which are COTS wireless routers. These routers use delay tolerant networking to synchronize and transfer files among each other. A major feature of the fog is that users can upload files without specifying an IP address or a hostname of a device - unlike uploading to FTP for example. The back end service, in this case the fog service that runs on routers, intelligently chooses a device on which to replicate data. Figure 2 shows the working of the fog in DistressNet. There are three collapsed buildings

B1, B2 and B3 with a COTS wireless router inside. Two other devices on vehicles V1 and V2 serve as data mules for delay tolerant networking, but are not fog devices - fog users cannot access their files on these devices. The heterogeneous nature of data means that each stream may have different QoS requirements. Raven is a QoS aware routing protocol for DRNs that can be tuned to improve the jitter, which is the variance of the packet delivery delay. We now provide a short overview of routing in a fog as well as the Raven routing protocol.

1) Firm and Potential Flows in a Fog: Data flows in the fog are not concretely defined since the fog back end has to choose endpoints optimally. A *firm* flow is one whose source and destination are well known (as IP addresses, host names or a DTN URI for example). For example, a firm flow can exist between B1 and B2 in Figure 2. A *potential* flow's source is well known, but its destination is simply the fog. It is up to the fog service to convert each potential flow into one or more firm flows. The exact number of firm flows created depends on the potential flow's *availability metric* which ranges from 0-100%. It denotes the importance of the data. An availability of 100% means that data will be available on all Fog devices: a firm flow is created from the source to every other fog device. An availability of 25% means that data will be available on a quarter of all fog routers: a subset of devices is chosen and firm flows are created to each of them. For example, a potential flow with 50% availability can exist between B3 to the fog in Figure 2. It is up to fog service to create a flow from either B3 to B2 or B3 to B1. More critical data will have higher availability. This process of choosing a subset of devices has implications on the system's performance - and is solved by the problem formulation below.

2) The Raven Routing Protocol: We use Raven [10] as the underlying DRN routing protocol. It is able to control QoS metrics in the DRN, especially the variance of the packet delivery delay, by using risk aversion techniques. The mobility in the area is assumed to follow the Post Disaster Mobility Model [13]. In this model, Centers refer to static areas in the disaster area such as the triage or a collapsed building or the fuel depot. There are several types of Mobile Agents which move between Centers according to various patterns. For example, an ambulance chooses a Center at random to travel to but always returns to the Triage before repeating the process. It is assumed that there is a fog device present at each Center. Going back to Figure 2, two Mobile Agent categories are illustrated - ambulance (V1) and patrol car (V2). They move between three Centers which are collapsed buildings B1,B2,B3.

This mobility model is represented by a stochastic multigraph. The edge weights in a stochastic graph are not scalars but are distributions with a mean and variance: the *risk* is computed as a linear combination of mean and variance. It is called a multigraph since multiple edges are possible between vertices. In Raven, each vertex corresponds to a Center whereas each edge corresponds to a Mobile Agent category. The edge weights represent the physical travel delay incurred by a vehicle of that particular category, in traveling between the two incident Centers. Thus, each edge has a unique travel delay distribution depending on the mobility patter as well as the geographic location of its Centers. The stochastic multigraph for the scenario in Figure 2 is shown in Figure 3. Centers

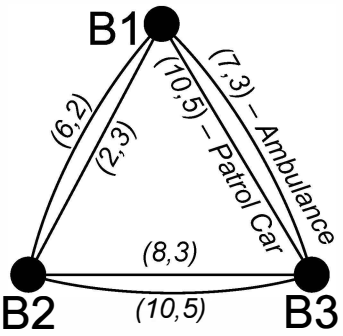


Fig. 3: The stochastic multigraph for the scenario in Figure 2. There are 3 vertices corresponding to 3 Centers, and 2 edges between every pair that correspond to the 2 Mobile Agent categories ambulance and patrol car. Edge weights are the (mean,variance) of the travel delay for that particular Mobile Agent category. This graph is based on mobility only, and is not affected by flows, either firm or potential.

B1,B2,B3 correspond to vertices B1,B2,B3 respectively. Two edges between B1 and B3 correspond to ambulance and patrol categories. The edge weight is the travel time distribution ($\mathcal{N}(7,3)$ minutes) for that category (ambulance) between the two incident centers (B1 to/from B3).

Raven works by using this graph to determine a set of paths between source and destination - i.e., on a *per-flow basis*. The K-Shortest Paths algorithm is modified to handle stochastic weights. The number of paths to be determined, K, as well as the risk aversion coefficient ρ (risk = mean + ρ*variance), are supplied by the user. To summarize, Raven computes K paths for each firm flow; and the performance of the protocol depends on the values of K and ρ as discussed in [10]. For example, for the B1 to B2 firm flow mentioned before, Raven could choose 2 paths B1-ambulance-B2 and B1-ambulance-B3-patrolcar-B2 depending on ρ.

Key Idea: The major insight in this paper is that if a fog device (at a Center) is not on one of the paths chosen by Raven to route data, it can sleep and save energy. Thus, the number of unique nodes present in the set of paths for all flows is an indicator of the energy consumption. This number can be reduced in a variety of ways: in the process of converting potential flows to firm flows (choosing Centers such that flows overlap) or in Raven (by reducing K or by choosing a set of paths other than the shortest). However, these techniques affect the system's performance - and therefore, there is a Pareto frontier between performance and energy consumption. Additionally, the user can control the operating point along this frontier by simply changing system parameters like those used by Raven.

B. Problem Formulation

The disaster area consists of c centers $C_1 \dots C_c$ and is represented by the stochastic multigraph $\mathcal{S}$. There are f firm flows $F_1 \dots F_f$ and their sources/dests $\{F_i^S\}$/$\{F_i^D\}$. p potential flows $P_1 \dots P_p$, their availabilities $A_1 \dots A_p$ and sources $\{P_i^S\}$ but no destinations. Thus the total number of flows is $h = f + \sum_{i=1}^{p} \lceil cA_i \rceil$. Each of these h flows has an associated parameter $k_1 \dots k_h$ that is used by Raven (the number of paths

to compute). The user has specified a global Raven parameter K: as a result, $1 \leq k_i \leq K$. When Raven computes paths for each of these h flows, let the number of unique Centers in the union of these paths be $\bar{c} \leq c$.

Now since $\mathcal{S}$ is stochastic, each path in this graph has an associated mean and variance (equal to the sum of means/variances of the constituent edges), and is hence a distribution. The path weight represents the packet delivery delay because the physical travel delay in a DTN is a major component of the packet delivery delay [7]. When a packet is sent on k paths simultaneously, the expected delay is the minimum of the delays of the k individual paths; it follows that the per-flow packet delivery delay is the minimum of k normally distributed random variables. For the flow numbered i with paths parameter k_i, the delay is $D_i = min\{D_{i1}, D_{i2}, \ldots, D_{ik_i}\}$ where each D is a distribution and not a scalar. A closed form expression for this minimum of several random variables is not trivial. For h flows, the overall packet delivery delay distribution is $D = \frac{\sum_{i=1}^{h} D_i}{h}$ where to re-emphasize, all quantities except h are normally distributed random variables with a mean and a variance. Using the mean-risk probability model, the risk of this distribution D is $risk(D) = E[D] + \rho * \sqrt{V[D]}$.

Given the above notation, our objective is to minimize the delay (which is called risk when variance is taken into account, i.e., $risk(D) := E[D]$ when $\rho = 0$) as well as $\bar{c}$. There are three parts to this problem: (1) the conversion of potential flows to firm flows, (2) applying Raven to each of the firm flows so that the delay distribution D can be estimated, and (3) tuning Raven's K parameter so that energy and $risk(D)$ are minimized. The above three problems are solved in a single optimization problem as follows. To convert a potential flow P_i into $\lceil cA_i \rceil$ firm flows, consider a binary vector V of length c. The ith element V_i corresponds to Center C_i: $V_i = 0$ if the Center is not chosen as a destination, and $V_i = 1$ otherwise. The sum of this bit vector should be $\lceil cA_i \rceil$. Repeating this procedure for p potential flows, the length of V becomes pc. Because Raven needs a K parameter for each flow, V is augmented with h more integers. Thus, the vector V of length $(pc + h)$ can now be used as input. It is a dual objective non-linear program:

$$\min_{V} \quad \text{RISK(V), UNIQ(V)} \tag{1}$$

$$\text{s.t.} \quad \sum_{j=1}^{c} V_{(i-1)c+j} = \lceil cA_i \rceil, \; i = 1 \ldots p \tag{2}$$

$$1 \leq V_i \leq K, \; i = (pc+1) \ldots (pc+h) \tag{3}$$

Constraint 2 deals with potential flows. It stipulates that the total number of firm flows created (by setting a Center's bit) for each potential flow equals $\lceil cA_i \rceil$. Constraint 3 makes sure that the K parameter needed by Raven cannot exceed the global value of K specified by the user. Equation 1 involves two procedures RISK and UNIQ which calculate $risk(D)$ and $\bar{c}$ respectively.

Procedure 1 details the calculation of the two objectives. Steps 1-7 involve the creation of firm flows from potential flows: for each potential flow (Step 1), if the bit corresponding to a Center is set (Step 3), a new firm flow is created (Step 4). Now that all h firm flows have been created, Raven is applied to each of these (Step 9) along with the Raven parameter K (Step 10). The resulting set of k_i paths are collected and duplicates are removed (Step 11). The number of unique vertices in these paths is nothing but the number of unique Centers (Step 13) since each vertex in the stochastic multigraph corresponds to a Center. The delay distribution, which is the path weight of each of these paths is calculated (Step 14), averaged (Step 15) and the risk is calculated (Step 16).

Procedure 1 RISK(V) and UNIQ(V)

Input: vector V, firm flows $\{F\}$, K, ρ

1: **for** $i := 1 \ldots p$ **do**
2: **for** $j := 1 \ldots c$ **do**
3: **if** $V_{(i-1)c+j} == 1$ **then**
4: $F \leftarrow F \cup$ (a new firm flow between P_i^S and C_j)
5: **end if**
6: **end for**
7: **end for**
8: $paths \leftarrow \Phi$
9: **for** $i := 1 \ldots h$ **do**
10: $\{Q_k\} \leftarrow$ (Raven with s/d F_i^S & F_i^D, k-parameter V_{pc+i})
11: $paths \leftarrow paths \cup \{Q_k\}$
12: $D_i \leftarrow \min\{Q_1, Q_2, Q_3 \ldots Q_K\}$
13: **end for**
14: UNIQ(V) $= \bar{c} \leftarrow$ unique nodes in $paths$
15: RISK(V) $\leftarrow (\sum_{i=1}^{h} E[D_i] + \rho * \sqrt{V[D_i]})/h$
16: **return** RISK(V),UNIQ(V)

C. Solution

Equation 1 is a dual objective, non-linear optimization problem. Because of its complexity, a stochastic optimization approach is preferred, as opposed to a deterministic one. Evolutionary algorithms are specially suited to solve multi-objective problems - and genetic algorithms (GAs) are the most popular variety of evolutionary algorithms. We use the NSGA-II algorithm to solve Equation 1, owing to its speed and low complexity. It is also able to handle disconnected Pareto fronts.

The input to the algorithm, vector V (Equation 2), is referred to as a "chromosome" in GA parlance. It is a string of numbers, either binary or real valued (in this case, integer valued). Through multiple GA operations like crossover, selection and mutation, new candidate solutions are generated in a stochastic fashion. The current set of candidate solutions (the "population") is evaluated (the "fitness" is calculated) and filtered to retain only non-dominated solutions. NSGA-II gives preference to Pareto optimal points that are situated far away, so as to capture both extremes of the front. The crossover operator used is Simulated Binary Crossover (SBX), since the chromosome is real valued. Mutation occurs according to the Polynomial operator. Selection happens in a Binary Tournament fashion.

D. Example

The above approach to solving Equation 1 is illustrated using Figure 3. Suppose that there is a firm flow from B1 to B2, and a potential flow from B3 with an availability of 0.5. The task is now to convert this potential into either a firm flow from B3 to B1, or B3 to B2. For each choice, a K value is to be determined for each of the flows including the firm flow. The first step is to construct vector V which has a total

length of 3 (one potential flow, three centers) plus 2 (two firm flows in total) $= 5$ elements. Let an example chromosome be $V = [0\ 1\ 0\ 2\ 2]$ (input in Procedure 1). Assuming that the first three elements stand for B1, B2 and B3, the vector indicates (Step 3) that the potential flow is to be converted into a firm flow from B3 to B2 (Step 4). The two firm flows B1-B2 and B3-B2 each have $K = 2$ (Step 10). For the B1-B2 flow, there are six possible routes, two from B1 to B2 directly for each vehicle category, and four from B1 to B2 through B3. Applying Raven will determine the two best paths from these 6. Similarly for the B3-B2 flow, two best paths will be determined, resulting in two sets of two normal distributions. Step 12 reduces each set to one distribution (the minimum distribution), and Step 15 takes the average of these two minimum distributions, resulting in a single number $\text{RISK}(V) = 5$ (for example). $\text{UNIQ}(V)$ in this case is 3 since all three vertices have to be involved in data transfer. The resulting point is $(5, 3)$ which is then evaluated by NSGA-II to determine if it's a Pareto Point or not. The chromosome then mutates to, say $V = [1\ 0\ 0\ 3\ 1]$ and the whole procedure is repeated.

IV. Performance Evaluation

In this section we present the performance evaluation of our energy saving scheme. First, the existence of the Pareto front is confirmed by implementing NSGA-II and solving Equation 1, for a given scenario with Centers, Mobile Agents, potential and firm flows. Using a DTN simulator, the mathematical modeling is validated by running Raven at the Pareto points and verifying that the delay decreases when energy increases and vice-versa. Two of these points (the extremes) are chosen, and the Raven protocol is then evaluated at each of these Pareto optimal points, and compared with state of art DTN routing protocols such as MaxProp, RAPID and Prophet. Metrics used for comparison are packet delivery delay (PDD), the packet delivery ratio (PDR), the number of relayed messages (REL), and the total awake time (TAT). REL refers to the total number of packet transfers in the network *across all nodes*. TAT is the sum total of the awake time across *Centers only*, assuming that nodes stay awake only during those contacts where is transfer of data and that Mobile Agent nodes are powered by the vehicle's battery. The former assumption is justified by the existence of a low bitrate backhaul link which allows nodes to determine each other's buffer contents, as explained in Section 2.1. We have chosen REL in addition to TAT for fair comparison: only Raven is optimized for Center-only energy savings (TAT) while other protocols aim to reduce REL across all nodes. TAT is a better metric that UCT since the TAT represents all nodes of interest (namely Centers), whereas UCT is an average across all nodes.

A. Obtaining the Pareto Front

On the Helsinki street map, an EOC, a Triage and 25 Centers (collapsed buildings) were setup, their locations chosen randomly. A firm flow was setup from Building 1 to 9 as well as a potential with availability of 0.1 originating at the EOC, for a total of 4 $(= 1 + \lceil 27 \times 0.1 \rceil)$ flows. Three ambulances, three supply vehicles and 10 volunteers moved according to their respective mobility models. The stochastic multigraph for this scenario was computed.

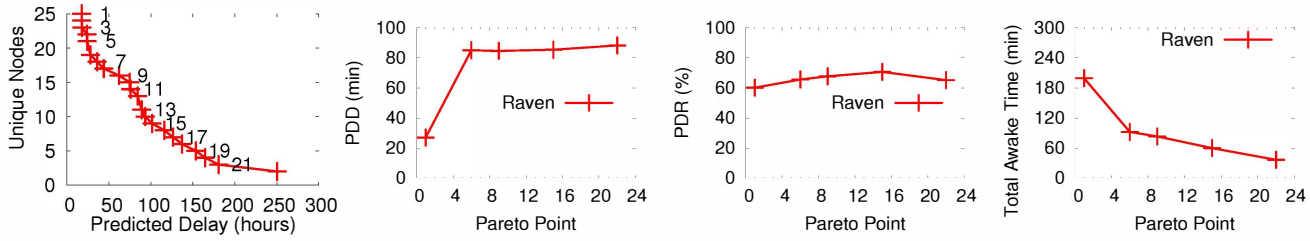
A Java implementation of NSGA-II was provided by the jMetal package: the parallel version where each chromosome is evaluated in a separate thread was chosen. After implementing Algorithm 1 within the jMetal framework, a Pareto front with 22 points was obtained and is shown in Figure 4a. As expected, energy consumption (Y axis) can be minimized only at the cost of increased delay (X axis). Pareto Point 1 utilizes all the 27 Centers with K values of [197 164 196 144] for each of the 4 flows, while Point 22 used only 4 unique Centers with K values of [2 4 3 4].

Optimization: This setup initially had long run times since the constraint handling extensions of NSGA-II were not implemented. The probability of obtaining a randomly generated chromosome that satisfied (Equation 2) was low, and decreased as the number of Centers increased. We developed an improved chromosome generation subroutine that produced high quality initial solutions. First, elements $1 \ldots pc$ of the input vector V were set to zero. Then, for each sub-vector of V corresponding to each of the p potential flows, $\lceil cA_i \rceil$ elements were randomly chosen and set to unity, satisfying Equation 2. As a result, each generation (iteration) in NSGA-II had valid chromosomes that were not discarded, leading to evaluation of more and more chromosomes, and the discovery of many more Pareto-optimal points. The chromosome generator was additionally biased to produce values near the lower/upper bounds as it was found that the ends of the Pareto front were not probed sufficiently even with $20,000$ iterations. The run time was reduced dramatically, resulting in quicker experimentation.

B. Verfifying the Pareto Front

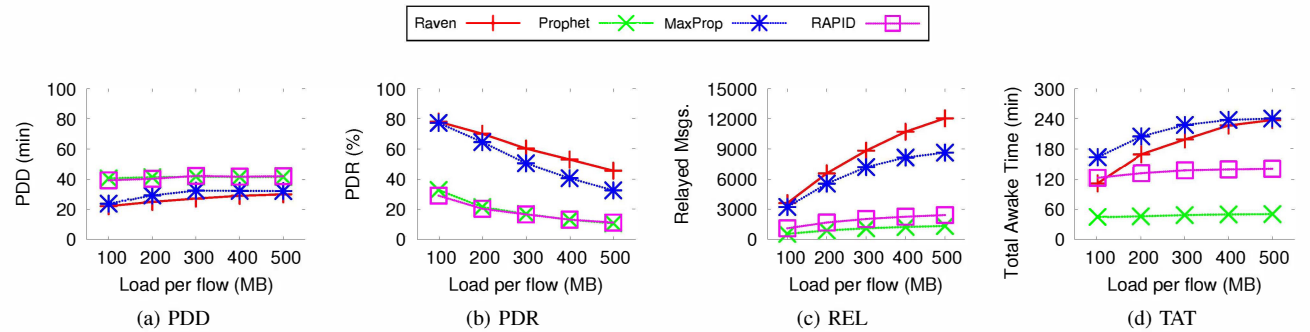
The performance of Raven as it is made to operate at various Pareto points is shown in Figure 4. Data for this and subsequent sections was obtained using TheONE, a Java based opportunistic network emulator at the packet level and *not* NSGA-II. The data workload per flow was 300MB, all created at $t = 0$ and the radio bitrate was 8MBps. The entire simulation lasted for 166.67 minutes and each data point is averaged over 200 random runs. For flows with high K values (> 140), Raven was made to flood packets generated on that particular flow. This was because $K = \inf$ chooses all possible paths in the network and is equivalent to flooding. The K-Safest Paths algorithm of Raven suffers increasing runtime as K increases - this optimization was performed to reduce the simulation time.

Point 1, as defined in Figure 1 optimizes the PDD at the cost of high energy consumption. This is confirmed in simulation since the PDD for Point 1 (Figure 4b) is the lowest, whereas the TAT (Figure 4d) is the highest. The TAT represents the system-wide energy consumption of the nodes at Centers - thus a high TAT means high energy consumption. A similar observation holds for Point 22, confirming the correctness of the problem formulation presented in Equation 1. Surprisingly, the PDR (Figure 4c) is fairly constant across all Pareto points. This can be explained by the fact that each potential flow results in a different set of firm flows for each Pareto point, suggesting that PDR depends on the physical location of the involved Centers. The TAT decreases by about 81% as we move towards the energy-optimal end of the Pareto front, while the PDD increases by $2.2x$. To summarize, the NSGA-II based optimizer provides the user with a variety of Pareto-optimal points, each of which represents a unique balance



(a) (b) PDD (c) PDR (d) TAT

Fig. 4: (a) The Pareto front and the effect of operating at different Pareto optimal points upon (b) packet delivery delay, (c) packet delivery ratio and (d) the total awake time.



(a) PDD (b) PDR (c) REL (d) TAT

(e) PDD (f) PDR (g) REL (h) TAT

Fig. 5: **Delay optimal Raven:** effect of increasing workload per flow (top row) and increasing radio bitrate (bottom row) on the performance of several DRN routing protocols: (a/e) PDD (b/f) PDR (c/g) number of relayed messages and (d/h) total awake time. Top row: Raven operated at Pareto Point 1, bit rate was 8MBps. Bottom row: Raven operated at Pareto Point 1, workload was 300MB.

between performance and energy consumption: for a ~61 minute increase in PDD, the total awake time of Centers decreases by ~162 minutes when moving along the Pareto front.

C. Delay Optimal Raven

We first operate Raven at the delay optimal Pareto point (Point 1 in Figure 4a), and compare it with state of art protocols. Increasing the data workload per flow will saturate the network while the contact opportunities remain the same, resulting in higher PDD and lower PDR, while increasing radio bitrate allows more data to be transferred per contact - the expected result is that PDD should be lower and PDR higher. The results are shown in Figure 5.

1) Effect of workload: Across all workload sizes, the PDD of Raven is the lowest compared to other protocols (Figure 5a) while it delivers *the most* number of packets (Figure 5b). Due to the flooding nature of Raven at high K values, it relays the most packets (Figure 5c). MaxProp replicates packets till an ACK is received. However, the TAT of MaxProp is larger (Figure 5d), suggesting that static Centers tend to relay more packets than the mobile vehicles. In general, the PDD and the number of relayed packets increase with increasing workload, as expected. Both Prophet and RAPID perform selective forwarding, relaying packets to only those nodes with a higher probability of reaching the destination (Prophet) or based on the marginal utility of relaying a packet (RAPID). Since RAPID makes a relaying decision based on the decreasing order of marginal utilities for each packet in the buffer, it relays more packets than Prophet and less packets than MaxProp/Raven. To summarize, Raven has the lowest PDD and highest PDR, but it also consumes more energy (considering relayed packets) than MaxProp.

2) Effect of bitrate: Increasing the radio bit rate allows more packets to be transferred *per* contact, while keeping the total amount of contact time constant. In general, all protocols

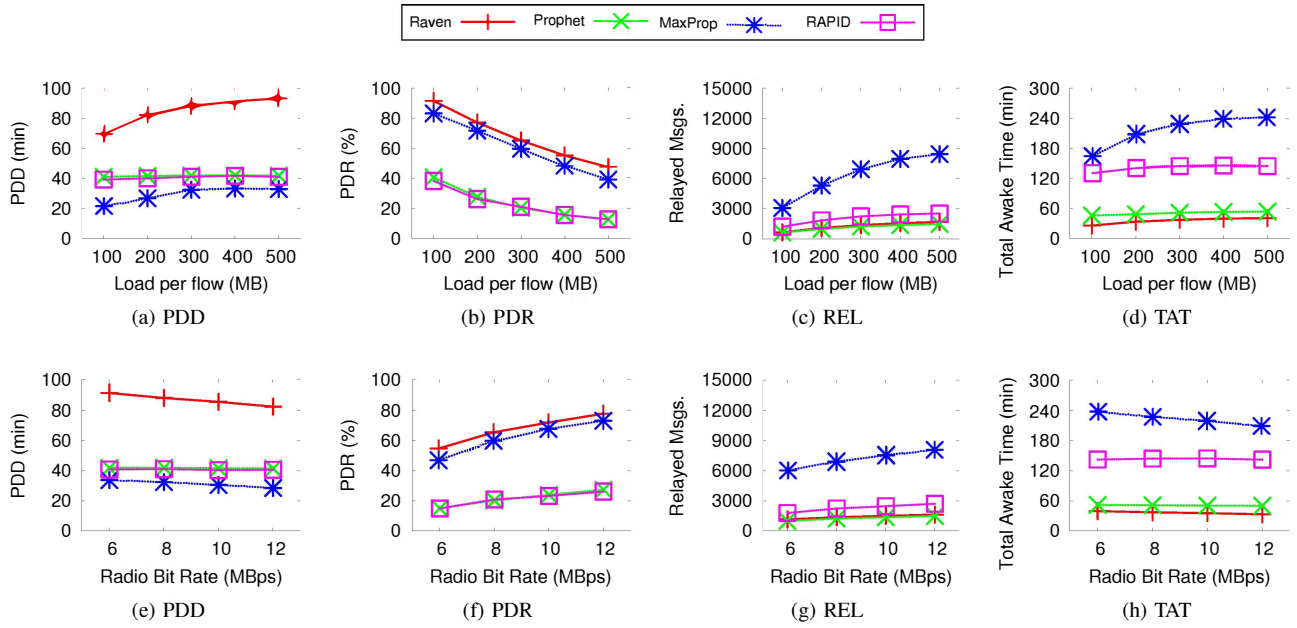


Fig. 6: **Energy optimal Raven:** effect of increasing workload per flow (top row) and increasing radio bitrate (bottom row) on the performance of several DRN routing protocols: (a/e) PDD (b/f) PDR (c/g) number of relayed messages and (d/h) total awake time. Top row: Raven operated at Pareto Point 22, bit rate was 8MBps. Bottom row: Raven operated at Pareto Point 22, workload was 300MB.

show reduced PDD (Figure 5e), increased PDR (Figure 5f) and increased number of relayed packets (Figure 5g). The decrease in TAT (Figure 5h) suggests that Centers save more energy even when the number of relayed packets increases. Since each node only transfers those packets that the other node does not have, Centers tend to collect all the unique packets within a short amount of time. This is because the probability of a vehicle meeting a Center is very high (due to their mobility models compulsorily involving Centers) compared to the probability of meeting another vehicle. With increasing radio bit rate, nodes are able to disseminate all of the generated data (note that no new data is generated after $t = 0$). Therefore, towards the end of the deployment, the number of contacts which involve transfers of unique packets decreases - thus decreasing the TAT. To summarize, Raven has the lowest PDD, delivers and relays more packets than other protocols, while showing a *reduction* in Center energy consumption.

D. Energy Optimal Raven

Next, we operate Raven at the Pareto point which guarantees the highest energy savings at the cost of PDD (Point 22 in Figure 4a), and compare it with state of art protocols. Compared to the previous subsection in which Raven operated at Point 1, the endpoints chosen for conversion of potential flows to firm flows are different *but still satisfy the availability constraints*. State of art protocols are evaluated with the same final firm flows as Raven for fairness. Results are shown in Figure 6. The axes in each of the figures is identical to Figure 5 for ease of comparison.

1) Effect of workload: Raven has the least energy consumption, both in terms of the number of relayed packets (Figure 6c), as well as the total awake time metric (Figure 6d), across *all data workloads*. Additionally it delivers the most packets (Figure 6b) compared to other protocols. However, the delay is higher than the others (Figure 6a), since the number of unique Centers used is only 4. Other protocols do not have this constraint and are free to relay packets to any node. Raven has only a limited number of paths it can use to relay packets, leading to congestion on these paths, resulting in a high PDD. Compared to Figure 5a, the average delay for Raven is higher by about an hour. This is a fairly small price to pay for having the least energy consumption *while delivering the most packets*. MaxProp, an epidemic-like routing protocol has the least delay and second best PDR, but at the cost of high energy consumption. For comparison, MaxProp relays 5x as many packets as Raven and based on TAT, has 5x the energy consumption. Even then, it does not deliver as many packets as Raven can.

2) Effect of bitrate: Once again, compared to the delay optimal variant (Figure 5), Raven has the least energy consumption both in terms of the number of relayed packets (Figure 6g) as well as the total awake time of Centers (Figure 6h), while delivering the *most* packets (Figure 6f). Similar trends are observed, such as increasing PDR, decreasing PDD, increasing REL and decreasing TAT with an increase in radio bitrate. Raven is found to consume almost 80% less energy than MaxProp, while delivering about 10% more packets. To summarize, it is possible to achieve very low energy consumption while keeping the PDR constant, but only at the cost of increased PDD. A high bit rate can reduce the energy consumption even further.

Summary: Operating on the delay optimal extreme of the Pareto front demonstrates the lowest PDD, highest PDR and high REL/TAT; the energy optimal extreme demonstrates the lowest REL/TAT at the same PDR but at the cost of increased PDD. Increasing the workload per flow hampers the PDD.

V. Related Work

Energy efficiency is an important concern to a DRN's user: high or non-uniform energy consumption can lead to network failure, low performance or violation of user constraints. These concerns are doubly important in highly challenged environments such as the post-disaster recovery process. Here we describe some recent research in the area of energy aware delay tolerant networking. For an overview of delay tolerant routing, the reader is referred to our previous work [10].

Software based approaches aim to minimize radio usage by reducing neighbor discovery overhead and/or reducing data transfers. The number of transmitted messages has been a metric in the performance evaluation of many DTN routing protocols [9] [13]. The scheme in [4] saves energy by adjusting the contact probing frequently optimally. For data transfers, epidemic routing is a simple protocol where each packet is replicated to every encountered node; this energy-inefficient approach has been the concern of recent research. The authors of [5] model energy efficient epidemic routing as an optimal control problem, while the authors of [6] propose that a packet be transmitted only when the number of neighbors reaches a threshold so as to reduce energy. Markov chains are used to model message dissemination performance of two routing protocols [3], and an optimal dynamic forwarding policy is derived. Yet another approach is to limit the maximum number of replicas a packet can have in the network [9]. Network coding [14] reduces the number of bits transmitted and hence the energy used by the radio. Forwarding packets to socially close nodes may increase performance, but simultaneously causes rapid energy depletion [15]. Intercontact routing [13] is perhaps the most similar to this work in the sense that it uses the post disaster mobility model as well as graphs with stochastic weights; however the objective of this paper is to quantify the effect of excluding certain nodes from routing on the performance (i.e., a hardware based approach), and is not concerned with the number of transmitted messages.

Hardware based approaches to saving energy in DTNs are far and few. In [8], the authors propose the use of a multi-tier platform that involves a long range, low bitrate radio that can wake up a short range, high bitrate radio. An optimization problem maximizes the number of bytes transferred while meeting a power consumption constraint. Using traditional low power, short range radios like 802.15.4 has been shown to be un-optimal for sparsely populated DTNs [2]. Energy efficient MAC protocols are useful for traditionally dense networks like MANETs, but inefficient for sparse DTNs with high inter-contact times. A different approach involves controlling the mobility of nodes to alter network performance [16], but in this paper we assume that node mobility is externally controlled. While these approaches aim to be awake for *every* contact, this paper has a completely different idea of *intentionally sleeping during node contacts* and examining the effect on routing performance. The work in [17] formulates the delay-energy tradeoff as a control problem, but there is no mention of Pareto optimality. It is also not known if the approach valid for non-probabilistic forwarding based protocols (i.e., where a packet is forwarded to a node based on probability p, a system parameter).

VI. Conclusions

We have presented a framework to characterize the Pareto front between system performance and energy consumption. Certain nodes in the network which only relay data but do not produce or consume it, can be excluded from the routing process to save energy - at the cost of performance. A dual objective non-linear program is formulated with Raven as the underlying routing protocol, and is then solved using NSGA-II. A set of Pareto optimal points is found, along with the respective network settings (potential flow endpoints and K values for each firm flow). For a 61 minute increase in delivery delay, a 81% decrease in energy consumption is achieved while the packet delivery ratio remains almost constant. The setup is evaluated using TheONE simulator and compared against state of art protocols.

Acknowledgements: This work was funded in part by NSF grants #1127449, #1145858, #0923203.

References

[1] H. Chenji, W. Zhang, R. Stoleru, and C. Arnett. Distressnet: A disaster response system providing constant availability cloud-like services. *Ad Hoc Networks*, 2013.

[2] H. Jun, M. H. Ammar, M. D. Corner, and E. W. Zegura. Hierarchical power management in disruption tolerant networks with traffic-aware optimization. CHANTS 2006.

[3] Y. Li, Y. Jiang, D. Jin, L. Su, L. Zeng, and D. Wu. Energy-efficient optimal opportunistic forwarding for delay-tolerant networks. *IEEE Transactions on Vehicular Technology*, 2010.

[4] W. Wang, M. Motani, and V. Srinivasan. Opportunistic energy-efficient contact probing in delay-tolerant applications. *IEEE/ACM Transactions on Networking*, 2009.

[5] M. Khouzani, S. Eshghi, S. Sarkar, N. B. Shroff, and S. S. Venkatesh. Optimal energy-aware epidemic routing in dtns. MobiHoc 2012.

[6] X. Lu and P. Hui. An energy-efficient n-epidemic routing protocol for delay tolerant networks. NAS 2010.

[7] H. Zhu, L. Fu, G. Xue, Y. Zhu, M. Li, and L. Ni. Recognizing exponential inter-contact time in vanets. INFOCOM 2010.

[8] N. Banerjee, M. Corner, and B. Levine. Design and field experimentation of an energy-efficient architecture for dtn throwboxes. *IEEE/ACM Transactions on Networking*, 2010.

[9] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. WDTN 2005.

[10] H. Chenji, L. Smith, R. Stoleru, and E. V. Nikolova. Raven: Energy aware QoS control for DRNs. WiMob 2013.

[11] S. Jain, K. Fall, and R. Patra. Routing in a delay tolerant network. SIGCOMM '04.

[12] A. Balasubramanian, B. N. Levine, and A. Venkataramani. Replication Routing in DTNs: A Resource Allocation Approach. *IEEE/ACM Transactions on Networking*, 2010.

[13] M. Y. S. Uddin, H. Ahmadi, T. Abdelzaher, and R. Kravets. Intercontact routing for energy constrained disaster response networks. *IEEE Transactions on Mobile Computing*, 2013.

[14] J. Widmer and J.-Y. Le Boudec. Network coding for efficient communication in extreme networks. WDTN 2005.

[15] C. Chilipirea, A. Petre, and C. Dobre. Energy-aware social-based routing in opportunistic networks. WAINA 2013.

[16] W. Zhao, M. Ammar, and E. Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. INFOCOM 2005.

[17] E. Altman, T. Baar, and F. D. Pellegrini. Optimal monotone forwarding policies in delay tolerant mobile ad-hoc networks. *Performance Evaluation*, 2010.