Harvey Flooding Rescue in Social Media

Zhou Yang, Long Hoang Nguyen, Joshua Stuve
Department of Computer Science
Texas Tech University
Lubbock, Texas
{zhou.yang, long.nguyen, joshua.stuve}@ttu.edu

Guofeng Cao
Department of Geosciences
Texas Tech University
Lubbock, Texas
guofeng.cao@ttu.edu

Fang Jin
Department of Computer Science
Texas Tech University
Lubbock, Texas
fang.jin@ttu.edu

***Abstract*—Social media provides a significant communication platform for rescuing people when Harvey hits Houston area. In this devastating flooding crisis, the overloaded official emergency institutes are not able to respond quickly due to the burst of call for help in a very short period of time. In this circumstance, many volunteers and people who need help often post their information in social media such as Twitter and Facebook. How to organize volunteers smartly and efficiently to help people is an extremely challenging and significant problem considering the constraints of volunteer's time slots, urgent priorities and etc. In this paper, we propose three rescue scheduling algorithms which are able to provide victims timely help by the volunteers in social media.**

I. Introduction

The emergence of the smartphone, social media platform, and other high technology speed our community entered into a new are, where every aspect is tagged with 'smart'. Social media sites such as Twitter and Weibo are experiencing an explosive level of growth, which makes their function not limited to broadcasting users daily observations [1], [2], serving as indicators for finance market [3], forecasting socio-economic disasters [4]–[6], but also playing a significant role in rescuing people when catastrophic Harvey flooding hits Houston area. From the early morning of August 26, 2017, Hurricane Harvey made landfall, and in the days following, it dumped trillions of gallons of rain on parts of Texas and Louisiana, spawning unprecedented flooding, leaving many people stranded in waist-deep flooding and desperate for help. So many calls came into 911 in one weekend (Aug 26-27) more than 56,000 within 15 hours making the official emergency response system overwhelmed during this crisis [1]. However, social media doesn't have the constraints of the 911 system, which is limited by the number of phone lines and the people available to answer them. Calls for help accumulated on social media platforms such as Twitter where information can flow in real time and is open for anyone to access. Some people posted their address, while others posted the address of a relative or a friend in need of help. As shown in Figure 1, three individuals tweeted the address of locations where people were stranded and needed help.

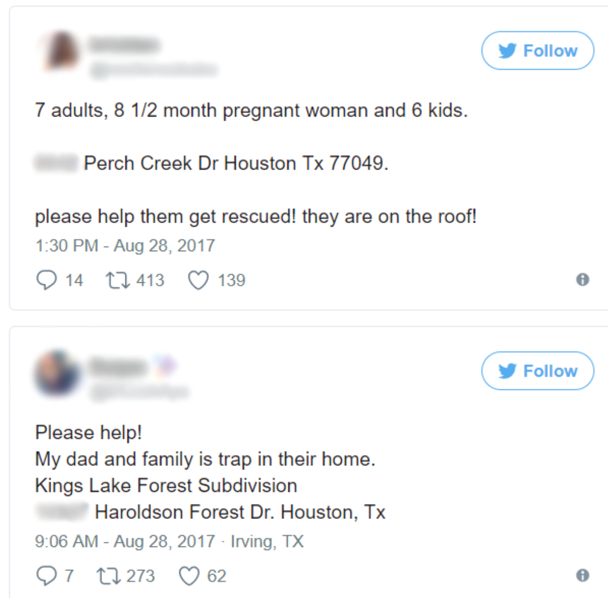

Fig. 1. Example of rescue request tweets for Harvey flooding.

After Harvey flooded out Houston, people sprang into action to help with rescues. Technically, anyone who sees a public plea for help can respond by dispatching a rescuer in a small boat or large vehicle. However, there are also many people who wanted to rescue flood victims, but do not know how and where and even help whom. One reason is their time availability varies. They may be available for a whole day service, or simply can only give nearby people a ride with limited capacity, or even focus more on sick people or babies who have high priority in rescue. The trivial time slot makes it seemingly unqualified to register volunteer in some official associations. How to connect the individual volunteers trivial time slots to schedule rescue work in an efficient way, how to allocate the volunteers task in order to minimize his/her time input, considering calling help location, emergence extent, etc., how to optimize the match among the large group of calling and distributed volunteers, considering time, location, and emergency, is an extreme challenging problem, however, is also a critical segment yet has not been well studied in the disaster relief work.

We propose to design rescue scheduling algorithms, hoping to provide victims with the timely support they needed, filling in gaps that were unable to be filled by the government and other organizations, making the victims connected with

[1]http://www.sandiegouniontribune.com/opinion/the-conversation/sd-hurricane-harvey-5-ways-social-media-helped-rescue-efforts-20170828-htmlstory.html

Fig. 2. (a) Victims and (b) volunteers tweets word cloud for Harvey flooding

volunteers who wish to provide humane care, thus release the flooding threat extensively. We collect Tweets from the very beginning of Harvey floods until the rescue work finished (from Aug 26, 2017 until Aug 30, 2017) in the United States, and conduct experiments using real dataset. As shown in Figure 2, the word cloud shows victims and volunteers' posts. The first work is to identify who and where are the flood victims from social media, in other words, a tweets classifier is required in order to extract Harvey victims' tweets and volunteers' tweets accurately and efficiently. Based on flood victims' location and numbers, we design scheduling algorithms in order to achieve optimal rescue efficiency and maximize utilization of volunteer resource. This approach will enable part-time volunteers to contribute their trivial time, and rescue maximal victims in an efficient way. Experimental results demonstrate such rescue caring algorithms are very practical in real disaster rescue implementations. Our contributions are thus:

- For the first time, we propose that the utilization of a public resource used for disaster rescue can be optimized by incorporating scheduling policies. We demonstrate that this approach is much more efficient than random scheduling using Harvey flooding tweets dataset.
- This scheduling algorithm incorporate considerations to determine whether victims are kids, seniors, pregnant women, among others, in order to give them high rescue priority and extra care. Real data are used to evaluate this algorithm performance and proved it achieved the best performance (less average waiting time, more rescued people, and fewer volunteers).
- We build a classifier as a mechanism to identify flood victims from the social media platform. For those critical missing information such as address, we design crowdsourcing algorithm to estimate their locations to maximize search efforts.

II. Related Work

a) Tweets Classification: Text classification focuses on automatically finding and extracting interesting and non-trivial information from unstructured text. Typical examples of text classification tasks include classifying documents into a set of specified topic areas (supervised learning), grouping documents such that each member of each group has a similar meaning (clustering or topic modeling), and extracting information with specific requirements (information retrieval) [7]–[10]. The Naive Bayes classifier is perhaps the simplest and the most widely used classifier. It models the distribution of documents in each class using a probabilistic model assuming that the distribution of different terms is independent of each other. Nearest neighbor classifier is a proximity-based classifier which use distance-based measures to perform the classification. The main idea is that documents which belong to the same class are more likely "similar" or close to each other based on the similarity measure [11], [12]. Support Vector Machines (SVM) are supervised learning classification algorithms where have been extensively used in text classification problems. Support Vector Machines try to find a good linear separators between various classes [13], [14]. A tweet is very shorter, with the limitation of 140 words, which makes conventional clustering algorithms problematic. Most words only occur once in each short text and, as a result, the term frequency-inverse document frequency (TF-IDF) [15] measure cannot work well. We intend to utilize a logistic regression, to help identify whether a tweet is rescue related or not.

b) Scheduling Algorithms: Scheduling algorithms are widely used for distributing limited resource among parties that request them based on certain rules, such as efficiency or fairness. The main purpose of scheduling algorithm is to maximize the utilization of resources and to ensure fairness among the parties using the resource. In this paper, we mainly talked about FCFS [16] scheduling and priority scheduling. Those scheduling techniques have been extensively used in telecommunication, computer system, logistics, military, among others. Uwe and Ramin [16] demonstrate that there are algorithmic issues in job scheduling where theoretical and applied research can both contribute to a solution, and came up with a new method to improve the utilization of FCFS scheduling. Joseph and Jennifer discussed the complexity of determining whether a set of periodic, real-time tasks can be scheduled on $m \geq 1$ identical processors with respect to fixed-priority scheduling, and showed that the problem is NP-hard in all but one special case [17].

III. Information Retrieval from Tweets

In this part, three tasks required in order to get full information from tweets.

A. Data Collection and Geocoding

The study described in this paper uses tweets geolocated to the United States and collected over a period from Aug 26 to Aug 30, 2017. We query Twitter API to collect tweets that also have meta-information including geographical coordinates, Twitter places, user profile location, and 'mentions information' about locations present in the body of the tweet. In cases when no geographical location was found in the tweet text, we proceed to process the geographical coordinates and the self-reported location string in user's profile metadata [6].

B. Tweets classifier - SVM

A tweet classifier is built in order to identify whether a tweet is calling for rescue or not. To reduce the computational complexity, only the tweets description is used as input in the support vector machine (SVM) classifier. The process begins by constructing a bag of words from the training dataset descriptions by deleting meaningless stop-words such as 'the', 'a/an', and 'at'. The resulting bag of words is composed of M words denoted as $[w_1, w_2, ..., w_M]$ [18]. Each tweet description X is considered as a vector of length M. If the word w_i occurs in its description, then $X(i)$ will be assigned a value of 1; otherwise, it will be 0. Each protest in the training dataset is assigned $Y = 1$ if it is a rescue related tweet, or $Y = 0$ if it is a non-rescue related tweet by manually checking the meaning of its description. In this way, each tweet is converted to a corresponding vector based on the bag of words. By combining all the vectors of tweets, a document term matrix is built. Eventually, the classifying decision becomes the solution to an optimization problem:

$$maxL_D(\alpha_i) = \sum_{i=1}^{N} \alpha_i - 1/2 \sum_{i=1}^{N} \alpha_i \alpha_j y_i y_j x_i x_j \qquad (1)$$

such that $\sum_{i=1}^{N} \alpha_i y_i = 0$ and $\alpha_i > 0$. And the decision rule is:

$$f(x) = \sum_{i=1}^{N} \alpha_i K(x_i, x) + b \qquad (2)$$

where $K(x_i, x)$ is a polynomial kernel in our solution. Equation (1) and (2) are explained in details at [19].

a) Classifier Evaluation: 1000 tweets are manually labelled as either rescue or non-rescue tweet. 70% of the dataset was used for training, and the rest used as test data. To ensure that the classification results are trustworthy, the performance is carefully evaluated by cross validation utilizing measurement criteria of precision (positive predictive value), recall (true positive rate) [20], F-measure (a measure that combines precision and recall) and accuracy (the proportion of true results, both true positives and true negatives among the total number of cases examined) [21]. The best classifier is SVM which achieved the best performance, with F-measure of 0.687 and accuracy of 0.93. A couple of well known classification methods (K-nearest neighbor [22], CART [23], and logistic regression [24]) serve as baseline models.

TABLE I
Classification methods comparison.

	Precision	**Recall**	**F_ measure**	**Accuracy**
Log. Regression	0.248	0.658	0.360	0.703
KNN	1.000	0.413	0.584	0.926
CART	0.710	0.579	0.638	0.917
SVM	0.606	0.793	0.687	0.930

C. Priority Determination

Rescue tweets contains some special information, including location, number of people, emergency conditions, and special requirement. To accurately identify flood victims' emergency conditions, some critical features, such as age, health status and situations need to be incorporated. We decide whether one rescue request need to trigger emergency rescue by designing a keywords corpus including terms such as *'grandma, grandpa, senior, old, baby, kid, pregnant, sick, ill, dangerous'*, when a tweet contains those keywords, the request will be considered high priority.

IV. Scheduling Algorithm

Since 911 can only serve certain callers at a time, with so many people calling for help at the same time, it's extremely difficult for everyone to get through. Thus many people who in need of rescue turn to social media for help. They may use hashtags such as '#HarveySOS', '#HarveyRescue' and '#HoustonRescue', or may post keywords such as 'help Harvey victims'. Meanwhile, there are volunteers who are willing to offer help and have the capacity such as a boat. However, those volunteers are scattered around the city. Without a coordination center with well-organized scheduling policy, it's hard to efficiently assign those volunteers to rescue the people who are in need of help. In this work, we aim to solve this dilemma by applying several scheduling algorithms that will boost the efficiency of this system, and hence speed the rescue work to minimize the hurt to victims.

A. The Queuing System And Scheduling Policies

First of all, some terms used in this paper need to be clarified. There are several key elements in a queuing system.

- Server, which refers to any resource that provides the required services. In our case, volunteer is the server.
- Customer, which refers to someone who requires service. In this paper, it refers to the victim requesting for help via posting rescue tweets.
- Calling population, which refers to the population of potential customers. It may be assumed to be finite or infinite (if arrival rate is not affected by the number of customers being served and waiting).
- System capacity, which refers to a limit on the number of customers that may be in the waiting line or system. In this case, we assume it to be unlimited.

Since the volunteer resource is limited, some of the requests have to wait in line for service, thus the queue is formed in the service center. The foregoing system can be mapped into a queuing system with multiple servers. And it can be described by using Kendall's notation in the form $A/S/C$ [25] [26], where A denotes the distribution of inter-arrival time, S represents the service-time distribution and c represents the number of servers of a queuing system. It has been extended to $A/S/c/K/N/D$ where K is the system capacity discipline and N is the size of the calling population [27] [28] [29]. When the final three parameters are not specified (e.g. M/M/1 queue), it is assumed $K = \infty$, $N = \infty$ and $D = FIFO$ [30].

In our study, this problem can be modeled by a $M/M/c$ queue by the definition of Queuing Theory. A $M/M/c$ queue is a stochastic process whose state space is a set of $\{0, 1, 2, 3, ...\}$ where the value corresponds to the number of customers in the system, including any members waiting for service. Several rules applied in this systems as listed below.

- Arrivals occur at rate according to a Poisson process and move the process from state i to $i + 1$.
- Service times have an exponential distribution with parameter . If there are less than c requests, some of the volunteers will be idle. If there are more than c jobs, some of the requests have to wait in a buffer to be served.
- The buffer is of infinite size, so there is no limit on the number of customers it can contain since there is no physical limitation.

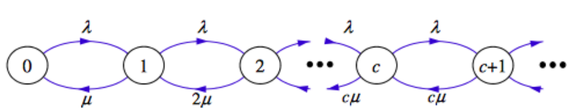

Fig. 3. State Space Diagram for $M/M/c$, λ:arriving rate, μ:service rate

Many theorems in queuing theory can be proved by reducing queues to mathematical systems known as Markov chains [31]. The model can be described as a continuous time Markov chain with transition rate matrix.The state space 0, 1, 2, 3, ..., refers to a set of values that the corresponding process may take. The model is a type of birthdeath process. The state space diagram for this chain is described in Figure 3, which indicates how the states of the process can be inter-converted. In $M|M|c$ queues, the arrival rate remains the same as $M|M|1$ queues but the service rate will depend on the number of servers. The service rate will be n for $n <= c$. As soon as the number of customers exceeds c, the service rate becomes c as shown in equation μ_c. The service rate, μ_c in this case, will be:

$$\mu_c = \begin{cases} n\mu & \text{n <c for n=1,2,...c} \\ c\mu & \text{n >c for n=c,c+1...} \end{cases}$$

The probability of having n customers in the service system can be written in a similar way as we wrote for $M|M|1$ model but with revised service rate.

$$P_n = \left(\frac{\lambda}{\mu_c}\right)^n \times P_0 \tag{3}$$

or

$$P_n = \begin{cases} \left(\frac{\lambda^n}{\mu(2\mu)(3\mu)\cdots\cdots(n\mu)}\right) P_0 \text{ (if n <c)} \\ \left(\frac{\lambda^n}{\mu(2\mu)(3\mu)\cdots\cdots(c\mu)(c\mu)^{n-c}}\right) P_0 \text{ (if n c)} \end{cases}$$

namely,

$$P_n = \begin{cases} \left(\frac{1}{n!}\left(\frac{\lambda}{\mu}\right)^n\right) P_0 \\ \left(\frac{1}{c!}\left(\frac{\lambda}{\mu}\right)^c \left(\frac{\lambda}{c\mu}\right)^{(n-c)}\right) P_0 \end{cases} \tag{4}$$

1) Performance measures of $M|M|c$ Queuing model: First, we will determine the number of customers in the queue, L_q. In the system, there will be no queue formed till the number of customers are less than or equal to the number of servers. The customer will enter in the queue when he finds all the servers busy. Hence, $n - c$ represents the number of customers in the queue. We can write L_q as below.

$$L_q = \sum_{n=c}^{\infty} (n - c)P_n \tag{5}$$

To determine L_q, substitute $j = n - c$ or $n = c + j$ in the above expression as given below.

$$L_q = \sum_{j=0}^{\infty} jP_{c+j}$$

P_{c+j} can be written as

$$P_{(c+j)} = \left(\frac{\lambda}{\mu} + \frac{\lambda}{2\mu} + ... + \frac{\lambda}{c\mu}\right)\left(\frac{\lambda}{c\mu}\right)^j P_0$$

or

$$P_{(c+j)} = \frac{(\rho)}{c!c^j}\rho^j P_0$$

Hence,

$$L_q = \sum_{j=0}^{\infty}\left(\frac{(\rho)}{c!c^j}\rho^j P_0\right)$$

$$= \left(\frac{\rho^{c+1}}{c!c}\right) P_0 \sum_{j=0}^{\infty} j\left(\frac{\rho}{c}\right)^{j-1}$$

which can be written as

$$= \left(\frac{\rho^{c+1}}{c! \times c^j}\right) \times P_0 \times \left(\frac{\partial\left(\sum_{j=0}^{\infty}\left(\frac{\rho}{c}\right)^j\right)}{\partial\left(\frac{\rho}{c}\right)}\right)$$

$$= \left(\frac{\rho^{c+1}}{c!c}\right) \times P_0 \times \left(\frac{\partial\left(\sum_{j=0}^{\infty}\left(\frac{\rho}{c}\right)^j\right)}{\partial\left(\frac{\rho}{c}\right)}\right)$$

$$= \left(\frac{\rho^{c+1}}{c!c}\right) \times P \times_0 \left(\frac{\partial\left(\frac{1}{1-\frac{\rho}{c}}\right)}{\partial\left(\frac{\rho}{c}\right)}\right)$$

$$L_q = \left(\frac{\rho^{c+1}}{(c-1)!(c-\rho)^2}\right) P_0 \tag{6}$$

After determining L_q , we can estimate the waiting time in the queue W_q using Littles law as given below [32], [33].

$$W_q = \frac{L_q}{\lambda}$$

Customers waiting in the service system will be the addition of W_q and service time.

$$W = W_q + \frac{1}{\mu}$$

The number of customers L in the service system will be,

$$\begin{aligned} L &= \lambda W \\ &= \lambda W_q + \frac{\lambda}{\mu} \end{aligned} \tag{7}$$

B. Scheduling Policies

The scheduling policy of a queuing system is crucial because it will largely affect the efficiency of a system [34]. However, it is irrational if we only take efficiency into consideration when dealing with social research. Other factors, such as fairness and well-being of a society, are of equal importance. Thus they should be considered as well. Therefore, in this work, we try to take into multiple dimensional elements into consideration and comparisons are made based on those elements. The main purpose is to evaluate the utilization of volunteer resources and explain how the situation can be improved from the viewpoint of scheduling policy.

Specifically, to evaluate the utilization and efficiency of using the public resource (Volunteer) in the Harvey hurricane, we proposed and compared three scheduling methods. The first one is First Come First Serve, which is used to describe how volunteer resource is utilized in the case of Harvey hurricane. On top of that, two scheduling methods are proposed: priority scheduling and hybrid scheduling. Specifically, the priority scheduling method tries to take into consideration the significance and urgency of each request, giving high priority to the old, the sick, the handicapped, kids and women, and at the same time, it will follow first come first serve criteria. Lastly, the hybrid scheduling method was put forward to further improve the whole system.

Algorithm 1: FCFS Scheduling with Single Server

```
Input: sequence of request[1..n], arrivalTime,
       serviceTime
Output: The scheduled sequence of requests[1..n]
Sorting the sequence of request according to their
  arrivalTime;
for every request i ∈ R do
    waitingT[i]=0;
    for every j from 0 to i − 1 do
        waitingTime[i]=waitingTime[i-1]+serviceTime[j];
    end
end
for every request i ∈ R do
    totalT[i]=waitingT[i]+serviceT[i];
    averageWT=averageWT+waitingT[i];
    averageTAT=averageTAT+TAT[i];
end
```

1) First come first serve: First Come, First Served is a service scheduling policy whereby requests of customers are sequentially attended in the order that they arrived, without taking into consideration other preferences or priorities. Generally, it is the most well-accepted public service scheduling policy for processing of queues in which customers wait for service that is not prearranged or booked ahead of time. In the analysis of FCFS policy, we assume that all the tweets are processed according to their post time. Namely, the rescue service for people with the earliest tweets time will be scheduled at first. Priority will not be taken into consideration, though some people may in a critical condition. FCFS scheduling policy has some important details that deserve discussion.

- In the FCFS scheduling, the service for rescue is non-preemptive, no matter how long it takes.
- Though the FCFS scheduling is (intuitively) fair in the human sense, it is unfair in the sense that non-urgent requests prior to the urgent request and a time-consuming request could make a lot of request wait.
- The average waiting time is relatively long.
- It can be embedded within other scheduling policy.

The FCFS scheduling algorithm for the single-server case is quite explanatory as described in Algorithm 1.

2) Static Priority Scheduling: Priority scheduling refers to the method of scheduling request for service based on priority. It involves potential priority assignment to every request, and requests with higher priority will be scheduled first, whereas requests with equal priority are scheduled based on a First-Come-First-Server (FCFS) basis. The priority can be either static or dynamic. Static priorities are assigned upon the coming of a request, whereas dynamic priorities are assigned depending on the specific situation of the request while waiting for service. Priority scheduling can be either preemptive or non-preemptive. A preemptive priority scheduling will preemptive the service resource if the priority of the newly arrived request is higher than the priority of the currently ongoing request, while a non-preemptive priority scheduling will simply put the new request in the queue according to FCFS. Priority scheduling policy also has some essential details deserving note.

- It's non-preemptive scheduling policy.
- Priority scheduling is intuitively unfair since there are queue-jumpers. However, it allows the important or urgent request to be scheduled first.
- Request with lower priority may be postponed if there are many requests with higher priority since everyone trapped in floods has the impetus to overstate their situations.

Based on our analysis, priority scheduling is necessary. First, it provides a good mechanism where the relative urgency of each request may be precisely defined. Thus the relatively urgent request can be scheduled for service first. Second, compared to plenty of requests that ask for request animals, the requests for rescuing human should be scheduled first especially when the resource is limited. The Static Priority algorithm used in this paper can be seen in Algorithm 2.

Algorithm 2: Greedy Algorithm for FCFS Scheduling with Multi-server

Input: R: a sequence of requests, at:arrivalTime, st:serviceTime, iat:interarrivalTime
Input: released, occupied: server
Output: scheduling sequence fo request
Initiation: all the servers are released at beginning
Sort R with regard to starting time of this rescue
for *every* $I \in R$ **do**
  Move all servers in occupied which finished before the start of I into released
  if *released* $\neq \emptyset$ **then**
    $m = select(earliestreleased)$
    Move m from released to occupied
  else
    $m = m + 1$
    create Rm and initialize it to $\emptyset$
  end
  Add I to R_m
end

3) Hybrid Scheduling: The hybrid scheduling policy is proposed to further improve the utilization of resource and enhance the efficiency of the queueing system. Basically, the hybrid scheduling is a combination of FCFS scheduling, priority scheduling and dynamic scheduling. The dynamic scheduling method discussed in this paper is different from the paper of Liu and Layland [35]. One of the difference is that the hybrid scheduling policy is non-preemptive. The hybrid scheduling policy discussed in this paper is different from that discussed in real-time scheduling for CPU scheduling. Some of the differences are listed in the following.

- The hybrid scheduling policy is non-preemptive.
- If the server is idle, incoming requests will be scheduled based on FCFS.
- Priority value is assigned to every request upon its creation. Specifically, if our classification algorithm detects words like "old","kid","child" and "handicap", then this tweet will be given higher priorities.
- If a tweet doesn't have keywords that indicate for higher priority, then the priority of the tweet is assigned according to the estimated time for rescue.
- When a server is available, requests with the highest priority will be scheduled first. If all the requests are with the same priority, then the request with the earliest finish time will be scheduled first.
- When a server is available, the priority of some requests can be reassigned due to urgent situations.

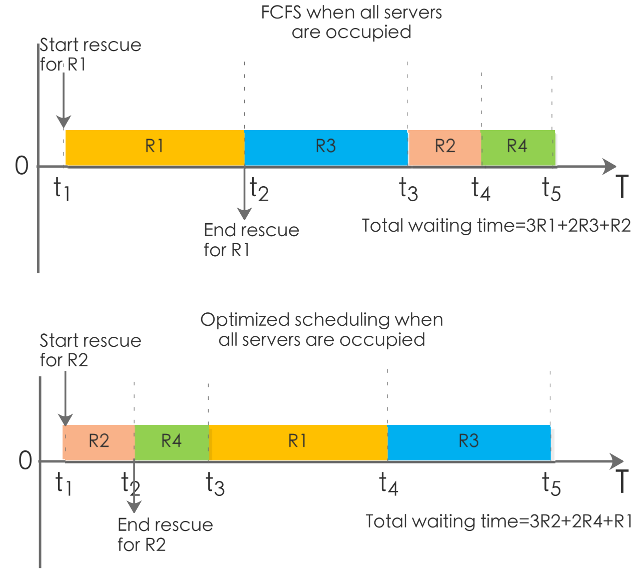

Fig. 4. Hybrid scheduling when all the servers are occupied. The box R_i represents request R_i that arrives in the queue, and the length of a box represents the length of service time. When all the servers are occupied, scheduling the shortest request first could optimize the waiting time for the system

The hybrid scheduling policy discussed in this paper was based on the foregoing rules, and the earliest finish time can be explained by the Figure 4.

4) Performance Evaluation: In order to evaluate the utilization of volunteer resource and performance of the Request-Server queuing system under different scheduling policies, several indexes are used in this study.

- Stationary test was performed to test the stationarity.
- Time-based indexes such as average waiting time in queue and total time in system are evaluated.
- Server utilization

There is always a trade-off between quality and efficiency in a single-server queue. For instance, the calling center where there is only one server, if good service is provided, then people in line have to wait longer. To evaluate the quality of service for such a system, the fraction of customers who have to wait before receiving service, also known as the delay probability, and the average customer waiting time are two important performance measures. Usually, these two measures

```
Algorithm 3: Hybrid Scheduling
Input: R: a sequence of requests, at:arriv-alTime,
       st:serviceTime, iat:inter-arrivalTime
Input: released, occupied: server
Output: scheduling sequence of request
initialize all the servers' state and requests state
sort request accoring to the arrival time
for every request I in R do
    select the request I with the earliest arriving time
    if no one waiting in queue then
        Move all servers in occupied which finished
         before the start of I into released
        if released server ≠ ∅ then
            select(released server)
            update server states
        else
            select a new server
        end
    else
        add request I to the queue
        sort queue with priority and service time
        seledt the first element of queue
        if released server ≠ ∅ then
            select(released server)
            update server states
        else
            select a new server
        end
    end
end
```

should be maintained under certain levels to meet customer expectations [36]. For a single-server $M/G/1$ queue, where the arrival process is assumed to be a homogeneous Poisson process. Let be the arrival rate of the Poisson process, and let m be the mean service time, then, $\rho = \lambda m$ is the traffic intensity of the queue [36]. When $\rho < 1$, it can also be interpreted as the server utilization [36]. By the Poisson property, the delay probability is given by $P_w = 1 - \rho$. When ρ is close to one, almost all customers have to wait before receiving service [36], [37].

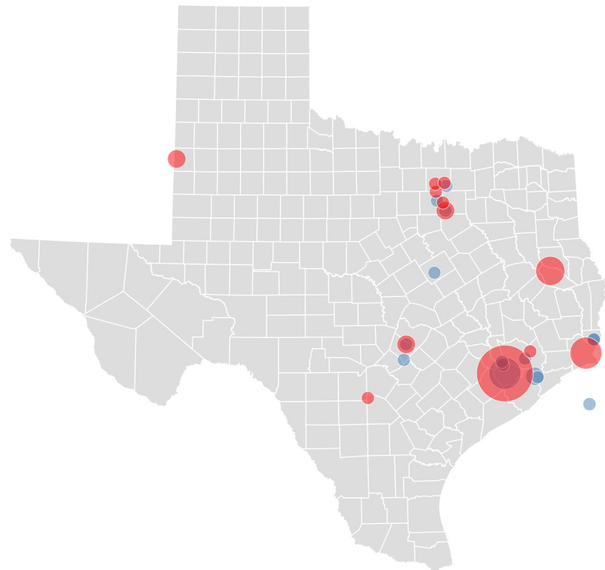

Fig. 5. Scatters of Harvey flooding victims and volunteers from Tweets in Texas, August 30, 2017. Red denotes victims who request help from Twitter and blue represents volunteers with boats.

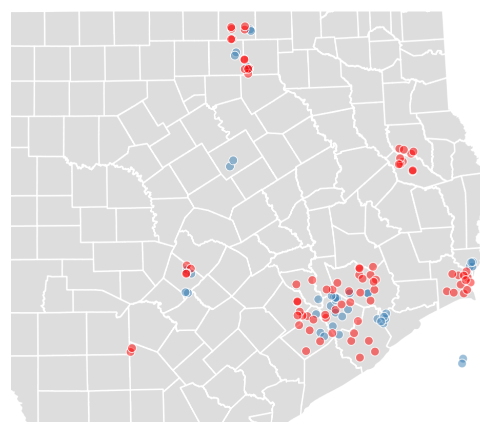

Fig. 6. A close look of Harvey flooding victims and volunteers from Tweets in Houston area, August 30, 2017. Red denotes victims who request help from Twitter and blue represents volunteers with boats.

V. Experimental Results

1) Results of Goodness-of-fit Test: The data about tweets used in the paper is from Aug. 26 to Aug. 30. It is reasonable to test if all the data is governed by the same distribution with the same parameter, since that the data distribution is affected by human living habit. Specifically, at late night the inter-arrival time of tweets is larger than that of daylight since most people fall asleep. Moreover, all the formulas and equations for the queuing system are held on the condition that the system is stationary. Therefore, the distribution test for data and stationary test for the queue process should be implemented.

The goodness-of-fit test is used to fit data into distribution [38]. First, we assume that inter-arrival time of rescue tweets from four successive days are governed by the exponential distribution with the same λ. However, the result of the goodness-of-fit test shows that the hypothesis is rejected by the small p-value (less than 5%). Second, we assume that the inter-arrival time on a daily basis is governed by the exponential distribution with different λ value. The goodness-of-fit test shows that null hypothesis is accepted and provide different λ value, and the estimated λ for the four successive days is 72,41,38 and 32, respectively. And figre 7 how the data is distributed.

There are two points behind the statistics should be explained. One is that the data from four successive days is not governed by the same exponential distribution. The reason is that the data used in this paper is just a sample of the huge information about Harvey hurricane. With some information

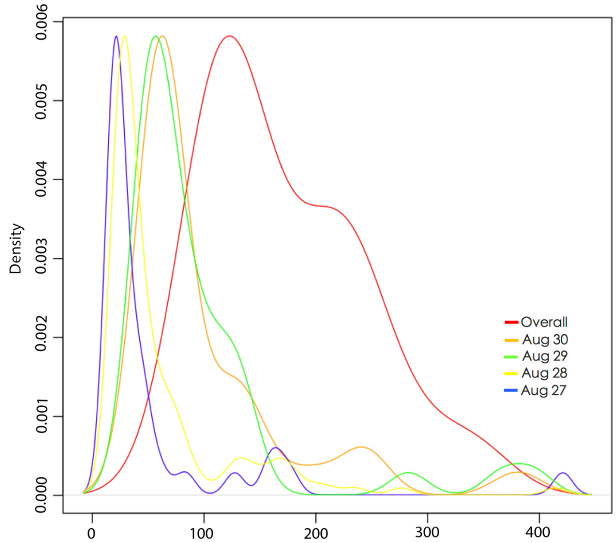

Fig. 7. The estimated probability density distribution of inter-arrival time, from Tweets data collected from Aug 27 to Aug 30, 2017.

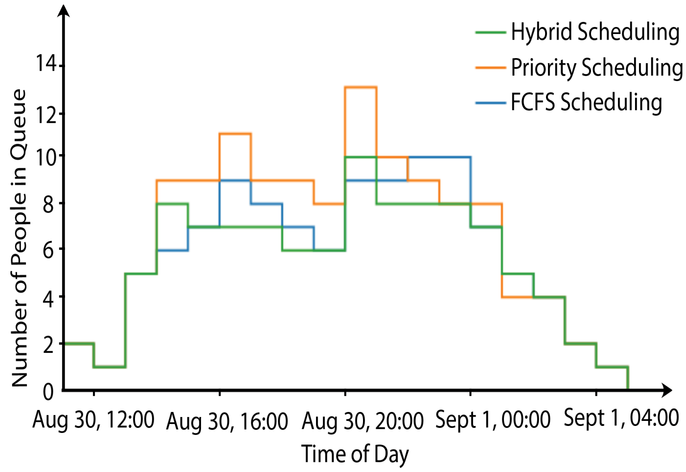

Fig. 8. Number of request (people) waiting in the queue. The increment in y-axis indicates requests entering the queue, and the decrements in y-axis signifies requests leaving the queue. The y-value at a specific time represents the length of the queue.

unavailable, there isn't enough evidence to support the hypothesis that the data is governed by the same distribution with the same parameters. The other one is that the inter-arrival time for the rescue tweets become smaller as Harvey hurricane land Houston. It can be explained since the rainfall on a daily basis is different. As a result, there are a few requests asking for help in the first day of Harvey hurricane. As the Harvey hurricane escalated, more and more people were flooded, consequently, the inter-arrival time of tweets that asking for help gets smaller.

A similar goodness-of-fit test is performed for the service time, which shows that the service time is governed by the exponential distribution with service time $\mu = 54$ minutes.

2) Results of Stationary Test: The stationary test shows the similar pattern with Goodness-of-Fit test. Namely, if we combine all the data as input for the stationary test, the null hypothesis will be rejected. That is, the system is not stationary since we mixed all the data origins from different sources. However, the tests based on daily data show that the process is stationary. Therefore, the fundamental assumption upon which the formula 1 to formula 5 is justifiable. And the formula 1 to formula 5 can be used to estimate the whole system and its queueing system. Specifically, the average number of requests in the system, the average number of request in the queue and the probability of having n request in the system can be obtained from formula 1 to formula 5.

3) Comparison of Scheduling Policy: To evaluate the utilization and efficiency of using the public resource (volunteers) in the Harvey hurricane, we compared the three scheduling methods here. Generally speaking, the hybrid scheduling algorithm achieved the best performance. First of all, the hybrid scheduling policy has the smallest average waiting time, which is 35 minutes less than the FCFS scheduling policies and 39 minutes less than the priority scheduling. Namely, victims will wait for less time before getting rescued if hybrid scheduling policy is used. Second of all, the system with the hybrid scheduling method has the smallest average queue size. Lastly, the system with the hybrid scheduling method has the smallest ratio of waiting, which indicates that fewer people will wait in line before getting service. And the comparisons can be summarized as following.

- The FCFS algorithm slightly over-performs the static priority scheduling. The static priority has the largest average waiting time of 3.73 hours among the three algorithms, because request with higher priority cut in line and keeps other request waiting.
- The static priority algorithm is more rational since it takes into consideration the significance and urgency of a request.
- The hybrid algorithm has the best performance, with the smallest average waiting time of 3.08 hours, which is less than the average waiting of 3.67 hours in the FCFS scheduling and 3.73 hours in the static priority scheduling. Moreover, the volunteer resource is fully used while efficiency is guaranteed by having the smallest average time of 3.08 hours in the system. And the performance comparisons on average waiting time can be visualized through figure 9.

The results can be explained with the following reasons. FCFS over-performed static priority because when all the volunteers are busy, if a request with higher priority cut in line, it will make all the people in the line waiting. Similarly, the hybrid scheduling minimized such waiting time by scheduling the request with the shortest service time first. As a result, the system as a whole can be optimized with minimal waiting time, and the average number of people waiting in line are the smallest. However, this improvement in performance is at the cost of keeping requests with longer service time waiting.

The implementation of algorithms on the dataset shows that volunteer resource is insufficient for the rescue. As is shown in figure 9, the average waiting time keeps increasing at the end

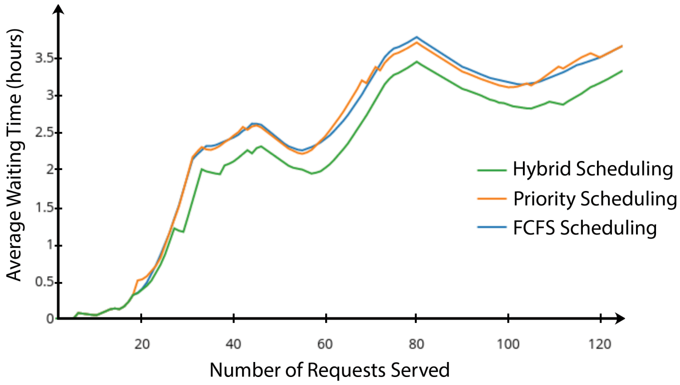

Fig. 9. Average waiting time in the queue, including all the requests that have been served in the system.

of every day. Two solutions can be used to solve this problem. One is to increase the service rate of the volunteer. Another way is to increase the number of volunteers in the system. This conclusion is substantiated with the increasing number of tweets looking for volunteers.

VI. Discussion

In this paper, we did a study to utilize a public social media platform (Twitter) for Harvey flood rescue work. We train a tweets classifier, which is able to identify flood victims and volunteers, based on which we design a series of scheduling algorithms: first come first serve, static priority scheduling, and hybrid scheduling. From extensive experiment study, we demonstrate that the hybrid scheduling approach is much more efficient than random scheduling for Harvey flood rescue. Based on the comparison of three algorithms' performance, we conclude that the average waiting time in queue and average time in system can be obviously reduced by implementing the hybrid scheduling policy. Moreover, with the help of text classification techniques and scheduling algorithm, this strategy can be transferred to other disaster rescue work when the public resources are needed.

References

[1] F. Jin, W. Wang, L. Zhao, E. Dougherty, Y. Cao, C.-T. Lu, and N. Ramakrishnan, "Misinformation propagation in the age of twitter," *Computer*, vol. 47, no. 12, pp. 90–94, 2014.

[2] F. Jin, E. Dougherty, P. Saraf, Y. Cao, and N. Ramakrishnan, "Epidemiological modeling of news and rumors on twitter," in *Proceedings of the 7th Workshop on Social Network Mining and Analysis*. ACM, 2013, p. 8.

[3] F. Jin, W. Wang, P. Chakraborty, N. Self, F. Chen, and N. Ramakrishnan, "Tracking multiple social media for stock market event prediction," in *Industrial Conference on Data Mining*. Springer, 2017, pp. 16–30.

[4] F. Jin, F. Chen, R. P. Khandpur, C.-T. Lu, and N. Ramakrishnan, "Absenteeism detection in social media," in *Proceedings of the 2017 SIAM International Conference on Data Mining*. SIAM, 2017, pp. 606–614.

[5] F. Jin, "Algorithms for modeling mass movements and their adoption in social networks," Ph.D. dissertation, Virginia Tech, 2016.

[6] F. Jin, R. P. Khandpur, N. Self, E. Dougherty, S. Guo, F. Chen, B. A. Prakash, and N. Ramakrishnan, "Modeling mass protest adoption in social network communities using geometric brownian motion," in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2014, pp. 1660–1669.

[7] H. Topi and A. Tucker, *Computing handbook: Information systems and information technology*. CRC Press, 2014, vol. 2.

[8] J. Solka, "Text data mining: theory and methods," *Statistics Surveys*, vol. 2, pp. 94–112, 2008.

[9] A. McCallum and K. Nigam, "A comparison of event models for naive bayes text classification," 1998.

[10] K. Nigam, A. K. Mccallum, S. Thrun, and T. Mitchell, "Text classification from labeled and unlabeled documents using em," in *MACHINE LEARNING*, 1999, pp. 103–134.

[11] J. M. Keller, M. R. Gray, and J. A. Givens, "A fuzzy k-nearest neighbor algorithm," *Systems, Man and Cybernetics, IEEE Transactions on*, vol. SMC-15, no. 4, pp. 580–585, July 1985.

[12] Y. Liao and V. Vemuri, "Use of k-nearest neighbor classifier for intrusion detection," vol. 21, no. 5, 2002.

[13] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "A brief survey of text mining: Classification, clustering and extraction techniques," *arXiv preprint arXiv:1707.02919*, 2017.

[14] J. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Processing Letters*, vol. 9, no. 3, pp. 293–300, Jun 1999.

[15] C. Manning, P. Raghavan, and H. Schütze, "Introduction to information retrieval/christopher d," 2009.

[16] U. Schwiegelshohn and R. Yahyapour, "Analysis of first-come-first-serve parallel job scheduling," in *SODA*, vol. 98, 1998, pp. 629–638.

[17] J. Y.-T. Leung and J. Whitehead, "On the complexity of fixed-priority scheduling of periodic, real-time tasks," *Performance evaluation*, vol. 2, no. 4, pp. 237–250, 1982.

[18] J. Sivic and A. Zisserman, "Efficient visual search of videos cast as text retrieval," vol. 31, no. 4, pp. 591–606, April 2009.

[19] R. Berwick, "An idiots guide to support vector machines (svms)," *Retrieved on October*, vol. 21, p. 2011, 2003.

[20] P. Perruchet and R. Peereman, "The exploitation of distributional information in syllable processing," *Journal of Neurolinguistics*, vol. 17, no. 2, pp. 97–119, 2004.

[21] M. Ghosh, "Statistical decision theory and bayesian analysis," *Journal of the American Statistical Association*, vol. 83, no. 401, March 1988.

[22] K. Beyer, J. Goldstein, R. Ramakrishnan, and U. Shaft, *When Is "Nearest Neighbor" Meaningful?* Berlin, Heidelberg: Springer Berlin Heidelberg, 1999, pp. 217–235.

[23] R. A. Berk, *Classification and Regression Trees (CART)*. New York, NY: Springer New York, 2008, pp. 1–65.

[24] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013, vol. 398.

[25] D. G. Kendall, "Stochastic processes occurring in the theory of queues and their analysis by the method of the imbedded markov chain," *The Annals of Mathematical Statistics*, pp. 338–354, 1953.

[26] H. Tijms, "Algorithmic analysis of queues," *A First Course in Stochastic Models*, 2003.

[27] H. A. Taha, *Operations research: an introduction*. Macmillan, 1992.

[28] R. P. Sen, *Operations research: algorithms and applications*. PHI Learning Pvt. Ltd., 2010.

[29] N. Prabhu, "A bibliography of books and survey papers on queueing systems: Theory and applications," *Queueing Systems*, vol. 2, no. 4, pp. 393–398, 1987.

[30] N. Gautam, *Operations Research and Management Science Handbook*. CRC, 2007.

[31] A. A. Markov, "Extension of the law of large numbers to dependent quantities," *Izv. Fiz.-Matem. Obsch. Kazan Univ.(2nd Ser)*, vol. 15, pp. 135–156, 1906.

[32] A. Leon-Garcia and A. Leon-Garcia, *Probability, statistics, and random processes for electrical engineering*. Pearson/Prentice Hall 3rd ed. Upper Saddle River, NJ, 2008.

[33] A. O. Allen, *Probability, statistics, and queueing theory*. Academic Press, 2014.

[34] X. Wu, R. Srikant, and J. R. Perkins, "Scheduling efficiency of distributed greedy scheduling algorithms in wireless networks," *IEEE Transactions on Mobile Computing*, vol. 6, no. 6, 2007.

[35] C. Liu and J. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," vol. 20, no. 1, pp. 46–61, January 1973.

[36] J. Dai and S. He, "Queues in service systems: Customer abandonment and diffusion approximations," in *Transforming Research into Action*. INFORMS, 2011, pp. 36–59.

[37] R. W. Wolff, *Stochastic modeling and the theory of queues*. Pearson College Division, 1989.
[38] *Formal Goodness Of Fit: Summary Statistics And Model Selection*. New York, NY: Springer New York, 2007, pp. 123–175.