Hindawi Publishing Corporation
International Journal of Distributed Sensor Networks
Volume 2016, Article ID 6968014, 13 pages
http://dx.doi.org/10.1155/2016/6968014

Research Article

CooperSense: A Cooperative and Selective Picture Forwarding Framework Based on Tree Fusion

Huihui Chen,[1,2] Bin Guo,[1] and Zhiwen Yu[1]

[1]*Northwestern Polytechnical University, Xi'an 710129, China*
[2]*Luoyang Institute of Science and Technology, Luoyang 471023, China*

Correspondence should be addressed to Huihui Chen; chenhuihui.cn@gmail.com

Received 14 December 2015; Accepted 29 March 2016

Academic Editor: Diego López-de-Ipiña

Mobile crowd photographing has become a major crowd sensing paradigm, which allows people to use cameras on smart devices for local sensing. In MCP, pictures taken by different people in close proximity or time period can be highly similar and different MCP tasks have diverse constraints or needs to deal with such duplicate data. In order to save the network cost and improve the transmitting efficiency, pictures will be preselected by mobile clients and then uploaded to the server in an opportunistic manner. In this paper, CooperSense, a multitask MCP framework for cooperative and selective picture forwarding, was designed. Based on the sensing context of pictures and task constraints, CooperSense structures sequenced pictures into a hierarchical context tree. When two participants encounter, their mobile clients will just exchange their context trees and at the same time automatically accomplish forwarding high-quality pictures to each other via a tree fusion mechanism. Via virtual or real pruning and grafting, mobile clients learn which picture should be sent to the encounter and which one should be abandoned. Our experimental results indicate that the transmission and storage cost of CooperSense are much lower comparing with the traditional Epidemic Routing (ER) method, while their efficiency is almost the same.

1. Introduction

People using cameras to get pictures is a typical application of mobile crowd sensing. Nowadays, widespread smart devices are driving more and more people to use cameras as visual sensors to relook at this world. It raised a particular type of participatory sensing paradigm: Mobile Crowd Photographing (MCP) [1, 2].

MCP applications collect uploaded pictures and send them to the backend server for further collective knowledge mining. Typical MCP achievements include monitoring the pollution of creeks [3], reposting and sharing fliers distributed in urban areas [4], and gathering pictures of buildings for 3D city modeling [5]. These MCP applications assume that transmission is available anytime and anywhere. However, communication will become unavailable and the internet will become inaccessible in some cases, such as an emergency. For example, 36 cellular base stations were out of service due to the power cut caused by the Tianjin explosion accident in 2015 [6]. Then some MCP applications were developed to report scenes of emergency situations (e.g., disasters or fire) [7–9]. Also, because pictures are of large size (around 200 KB for a useable zoomed-out picture to 3 MB for a full-size picture taken by an iPhone 6S), they cost much network flow and money (about 0.29 RMB for 1 MB on average in China) when the commercial network is connected (e.g., 3/4G). Therefore, some researchers studied opportunistically forwarding pictures [10] with ER (Epidemic Routing) [11] based on some free and high-speed transmissions like Bluetooth, WiFi, and so on. Researches on Delay Tolerant Network (DTN) [9, 12] and Mobile Opportunistic Network (MON) [13, 14] are typical examples.

MON and DTN utilize cooperative Work Nodes (WN) to store, carry, and forward pictures, which has been mentioned by Uddin et al. [7], Weinsberg et al. [12], Jiang et al. [15], Zhao et al. [16], and Yu et al. [17]. Their works leverage the cooperative sensing scheme to eliminate sampling redundancy so as to save energy. Cooperative WNs refer to those participants who join in to collect or forward pictures in this paper.

In MCP, it is possible that participants perform the same task and collect similar pictures. As a result, not all WN-carried pictures are valuable. Then what kind of pictures are valuable? Different applications give different answers. For instance, in the disaster area after an earthquake, pictures of buildings taken from different directions and in different places might be more valuable than those taken with the same spatial context [7]. Valuable pictures refer to those about-to-receive ones that are different from carried ones. Therefore, in order to select valuable pictures, MCP applications will keep additional context information on saved pictures. However, existing picture dissemination schemes in MON/DTN are not sufficient since the utilization of photographing contexts has not been effectively explored.

Based on requirements of forwarding pictures through an opportunistic way, five important constraints for the solution are summarized as follows.

(1) Self-Driven Delivery. Under the submitting permission from participants, the self-driven picture delivery is more applicable comparing with the manual transmission. Therefore, opportunistic forwarding is applicable.

(2) Energy-Saving. Energy limitation has been stated in all mobile applications, and thus our method must keep balance between the energy-saving and performance.

(3) Coverage Assurance. Selecting representative high-quality data and then forwarding or uploading them is a more effective way to decrease the traffic and storage consumption, but only on the condition that the selection method guarantees the coverage of the selected pictures from those raw ones.

(4) Low Delay. People out of the observation area need pictures on the scene to know about the situation, so the receiving delay of these pictures must be as minor as possible.

(5) Useless Epidemic Termination. Whenever WNs encounter, they exchanges their pictures via an epidemic way. Once a picture is uploaded to the data center, those duplications of this picture carried by WNs are useless and thus should not be forwarded any more. In this case, useless epidemic must be terminated as soon as possible.

In order to address these constraints mentioned above, we will use data fusion or aggregation in our picture forwarding method. Picture data fusion in MCP is different from numeric data fusion (e.g., an average value of temperature) [18, 19] or splicing pictures in wireless sensor network [20], and it usually relies on a data selection scheme [7, 9, 15]. In other words, based on some selection constraints, the WN will only forward valuable pictures selected from a picture set union of this WN and others' picture collections.

CooperSense is a novel cooperative and selective picture forwarding framework for varied MCP tasks. It leverages the cooperation of opportunistically encountered participants to decrease the delay of uploading pictures and utilizes epidemic picture routing with fusion to eliminate sampling redundancy. Specifically, the following contributions are mainly made:

(1) Proposing a generic framework based on a cooperative data fusion to address the problem of how to select and forward high-utility picture for MCP tasks; we, especially, are the first to consider useless epidemic termination for energy-saving in its cooperative picture forwarding framework.

(2) Developing a novel and efficient tree model to structure the information of picture collection, named picture tree (PicTree), which satisfies various MCP task needs and constraints for data fusion; photographing contexts as locality, timestamp, and shooting angle are used to build this tree structure.

(3) Proposing a PicTree fusion based method for picture selection; without image processing, PicTree fusion can efficiently select valuable pictures and abandon useless ones.

We have developed a visualized simulator to simulate and analyze forwarding behaviors of WNs, including PicTree fusion based and Epidemic Routing based [11] (ER-based) picture forwarding schemes. Results show that CooperSense saves over 50% traffic and storage space comparing with the traditional ER-based method.

The rest of the paper will be organized as follows. Section 2 outlines the related work on common data fusion and picture selection and fusion methods for MCP. Section 3 presents the CooperSense framework and data models followed by PicTree-based picture set fusion method in Section 4. We will present and discuss experimental results in Section 5, and we conclude the paper and speculate about the future work in Section 6.

2. Related Works

CooperSense well addressed problems in the cooperative picture forwarding procedure in MON/DTN, which utilizes a data fusion method and a data selection method. In this section, we will introduce some related works about these two methods.

2.1. Data Fusion. Studies focusing on the issue of data forwarding in MON/DTN during a disaster/emergency either introduces DTN architecture to an efficient disaster response network [21] or eliminates redundancies in disaster related content [7] for fast message delivery out of the disaster zone. The performance of simple routing protocols (e.g., flooding) for data forwarding is generally poor [10]. Many generic opportunistic forwarding protocols, such as Epidemic Routing (ER) and its variations, for example, Epidemic Routing with Fusion (ERF) [16], have been proposed to route data among mobile nodes.

Picture data cannot be fused as numeric data can. For the visual sensor network, Chow et al. proposed the method to process and combine overlapping portions of pictures in neighbor sensors so as to reduce the energy spent on image transmission [20]. But, in MCP, battery capacity of mobile devices is limited. In order to save the power used on image processing, the representative pictures should be simply selected in the first place by mobile devices as a fusion result. Picture set fusion is regarded as picture selection, and we will introduce some related works in the following.

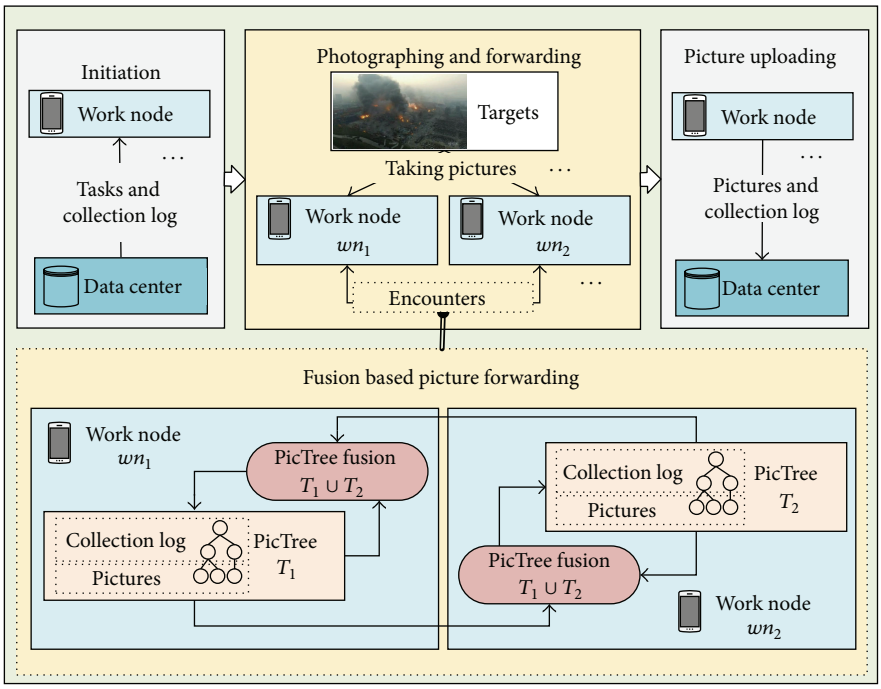


Figure 1: The framework of CooperSense contains three stages.

2.2. Picture Selection. MCP applications can measure similarity of pictures based on different features due to their various knowledge mining purposes. Available features include both visual features and photographing contexts [2], such as the shooting direction, the timestamp, the location, and the ambient light. For example, FlierMeet [4] collects crowdsourced photos of fliers for cross-space public information reposting, tagging, and sharing, and it only selects one picture of every flier based on the visual feature. PhotoCity [5] leverages a game-based incentive way to collect photos of buildings in cities for 3D city modeling, and it selects pictures of buildings taken from different directions. Photonet [7] and the work in [9] are disaster response applications, by using which a group of survivors and first responders may be able to survey the damage and send images to a rescue center without the access to functional communication infrastructure. These two applications focus on collecting visually different pictures based on the location and visual features. Therefore, in order to satisfy various situations of MCP applications, CooperSense will measure similarity of pictures based on the location, the shooting direction, and the timestamp for picture selection.

In order to save traffic and decrease the communication overhead, some MCP applications use a preselection way to forward valuable pictures, for example, PhotoNet [7] and MediaScope [15]. Preselection means that valuable pictures are selected by mobile clients before they are uploaded. However, those applications are application-specific and cannot address varied needs/constraints (e.g., spatiotemporal contexts, single or multiple shooting angles to a sensing target). COUPON [16] addresses data fusion in MON, but it is a generic and formal solution and needs further improvement for MCP. Therefore, CooperSense can address data fusion issue for MCP in MON and utilizes a data fusion for the preselection process. For MCP, data fusion is actually merging picture sets of different WNs.

2.3. Picture Set Fusion. For MCP, most applications will focus on how to efficiently assess similarity of pictures and select diverse pictures. For example, WNs of PhotoNet [7] a priori forward other pictures that can increase content-diversity of others' picture sets. The work in [9] detects critical content (e.g., fires, road blocks) on the images through image processing, and those detected images with critical content can be forwarded faster than those without critical content in order to quickly address critical events. Mediascope [15] also selects different pictures based on their locations. Although they tried to decrease duplicate picture transmission through picture set fusion, the comparing algorithm to select pictures is queue-based, which is time-consuming. In order to improve the efficiency, CooperSense uses a faster tree-based algorithm to aggregate picture sets, which will be introduced in the following sections.

3. CooperSense Framework

3.1. An Overview. CooperSense aims to satisfy the sensing requirements of various MCP tasks. It uses an efficient and effective picture set fusion method and an overhead-saving selective forwarding method based on the picture set fusion. In this section, we will present three data models and the framework of CooperSense.

As shown in Figure 1, CooperSense has three-stage workflow: (1) the *initiation stage*, (2) the *photographing and forwarding stage*, and (3) the *picture uploading stage*. We will introduce the mechanism via the workflow at the WN side.

Firstly, in the *initiation* stage, when a person is entering the observed area, he/she will download task requirements to his/her smart device, namely, the WN. Since some tasks may be long-term, the WN can download requirements of both new tasks and old unexpired tasks. Meanwhile, in order to terminate the epidemic of useless pictures in time, the *collection log* of unexpired old tasks is also downloaded. The collection log is a PicTree structure and contains the information of carried and uploaded pictures. Both MCP server and WNs have collection logs. According to the collection log contributed by other WNs, any WN can delete useless pictures and forward useful pictures to other WNs in the next stage, that is, the photographing and forwarding stage. Here, useless pictures refer to WN-carried duplicates of uploaded pictures, so deleting them means terminating useless epidemic.

Secondly, in the *photographing and forwarding* stage, people take pictures according to tasks' requirements, and WNs automatically forward valuable pictures to other WNs when they encounter. For instance, assuming that the WN wn_1 encounters a WN wn_2, then the forwarding consists of two steps: (i) wn_1 sends its collection log to wn_2 and wn_2 merges wn_1's collection log with its own, named PicTree fusion; (ii) wn_2 can select pictures during the PicTree fusion procedure and send them back to wn_1, named PicTree fusion based forwarding. Also, using this two-step procedure, wn_2 sends pictures to wn_1.

Finally, in the *picture uploading* stage, the WN leaves the observed area and uploads picture collections when the data center becomes accessible. Also, uploading picture is based on PicTree fusion to save traffic.

3.2. Data Modeling

3.2.1. Task Model. An MCP task is denoted by TSK: $\langle tid, whn, whr, ntr\rangle$. *tid* is the identifier of the task. *whn* denotes the start time and the end time of the task. *whr* is a set of geographic coordinates and denotes where the targets are. *ntr* denotes the required number of pictures, and it is used to determine whether to stop an unexpired task or not. By the way, the task posted to participants is described in natural language and TSK is its formalization.

Since pictures taken in close proximity or time period may be highly similar, the implicit task constraints are defined for the picture selection in most MCP applications, for example, PhotoNet [7], Smartphoto [22], and Fliermeet [4]. The task constraint is denoted by TSKC: $\langle tid, c\_loc, c\_tm, c\_agl\rangle$. If two pictures are different, *c_loc* is the minimum geographic distance of their locations, *c_tm* denotes the minimum time span of their picture-taking time points, and *c_agl* is the minimum angle of their shooting directions.

3.2.2. Picture Model. A picture captured by a smart device is denoted with PIC: $\langle wid, tid, loc, tm, agl, img\rangle$. *wid* is the identifier of the WN, *tid* is the identifier of the task that the picture is taken for, *loc* and *tm* denote the coordinate and the timestamp of the photographing, respectively, *agl* denotes the angle between the shooting direction and the north pole, and *img* is the image file.

Table 1: Layers and tree nodes' structures.

Layer	Attribute of the picture	Symbol	Attributes of the tree node
Task	*tid*	L^t	$\langle vLink, tid\rangle$
Location	*loc*	L^l	$\langle vLink, loc, c\_loc\rangle$
Date and time	*tm*	L^d	$\langle vLink, ts, te, c\_tm\rangle$
Shooting angle	*agl*	L^a	$\langle vLink, agl, c\_agl\rangle$
Image	*img*	L^i	$\langle vLink, img\rangle$

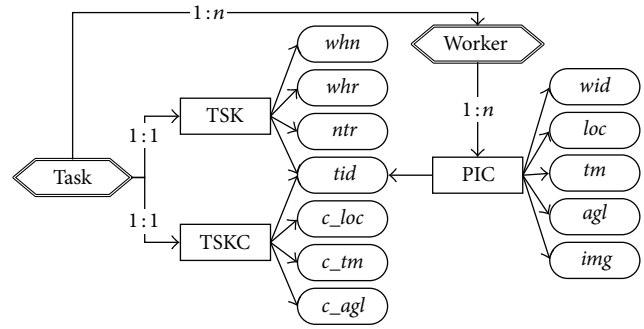


Figure 2: Data model relationship between pictures and a task.

3.2.3. PicTree Model. The picture set carried by a WN is denoted by P^{wid}, and its corresponding PicTree-based collection log is a PTree structure [2] and presents similarity degree of pictures. Reference [2] proposed PTree, a tree structure to cluster pictures. PTree's layers can be exchanged to increase the clustering efficiency. Since we focus on picture forwarding in DTN or MON, we used a generic fixed layer mapping in this paper; that is, each layer is one-to-one mapped onto the feature of a picture. As shown in Table 1, five layers from top to down are the task layer, the location layer, the date and time layer, the shooting angle layer, and the image layer, denoted by L^t, L^l, L^d, L^a, and L^i, respectively. Image layer is the bottom layer, where all images are stored.

Each layer of the PicTree is mapped onto a feature of the picture, so properties of tree nodes in different layers are denoted by different vectors. As shown in Table 1, most attributes of tree nodes correspond to attributes of both the task and the picture shown in Figure 2. Attributes (*ts*, *te*) in the layer L^d denote a time span. Particularly, each tree node has a common attribute, that is, the virtual link denoted by a Boolean variable *vLink*, to mark whether the tree node is actually live or not. *vLink* is an important attribute for PicTree fusion, and we use an example to explain the usage of it. A WN wn_1 uploads a PIC x to the data server and it then removes x. After that, wn_1 encounters wn_2 who also carries the PIC x; then the result is that wn_2 forwards the uploaded PIC x to wn_1, which is clearly a useless copy operation. In order to avoid such forwarding, our method introduces the parameter *vLink* to mark uploaded pictures. With the information from *vLink*, wn_1 can reject wn_2's sending-data request and also tells wn_2 that x is already available at the server.

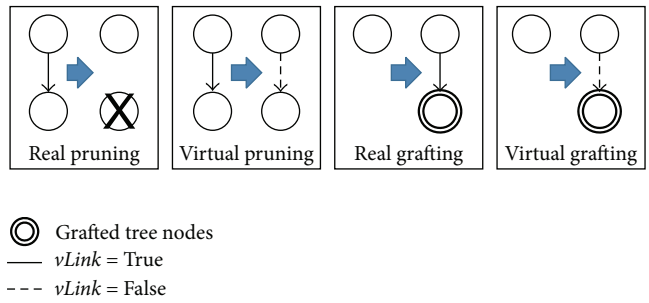


Figure 3: Four kinds of basic operations for PicTree fusion.

Traditional tree fusion utilizes two operations, pruning and grafting. In order to terminate useless epidemic, we proposed four operations for PicTree fusion shown in Figure 3, namely, *virtual* or *real grafting* and *virtual* or *real pruning*. The tree node whose *vLink* is False means that it is *virtual pruned*, and its offspring tree nodes can be removed, that is, *real pruned*. Consequently, if *vLink* of the tree node is False, then, the fact that this node is grafted is called *virtual grafting*, and if not, the grafting is called *real grafting*.

4. Methodology

We use the PicTree structure to store the collection log, including all the context information of carried pictures. Through merging two PicTrees of two WNs, useful pictures can be selected while useless ones can be deleted. Next, we will introduce PicTree fusion followed by PicTree fusion based picture forwarding.

4.1. PicTree Fusion

4.1.1. Tree Node Fusion. PicTree fusion is a procedure of merging one PicTree with another PicTree through merging or grafting tree nodes. The root node is meaningless and tree nodes mentioned in this paper do not include the root node.

In order to merge PicTrees, two methods are proposed as follows.

(1) A method to assess the similarity of two tree nodes: as shown in Table 2, $\langle tid, c\_loc, c\_tm, c\_agl\rangle$ is a TSKC instance, and *tid*, *loc*, *ts*, *te*, and *agl* are attributes of tree nodes. Two tree nodes N_1 and N_2 on the same layer are similar if they satisfy the condition in Table 2 and their parent tree nodes are also similar. Therefore, two pictures are similar only if they are in two similar leaf tree nodes.

(2) A method to value parameters of the merged tree node: if two tree nodes are merged, parameters of the new tree node will be valued with methods shown in Table 3.

Further, there are seven basic fusion rules for tree node fusion as follows:

(1) Any nonleaf tree nodes can be merged if they are similar.

Table 2: Conditions for accessing similarity of two tree nodes N_1 and N_2 on different layers.

Layer	Conditions for two tree nodes being similar
L^t	$N_1.tid = N_2.tid$
L^l	$\|N_1.loc, N_2.loc\| < c\_loc$
L^d	$\|N_1.ts - N_2.ts\| + \|N_1.te - N_2.te\| < c\_tm$
L^a	$\|N_1.agl - N_2.agl\| < c\_agl$

Table 3: Methods to combine tree nodes.

Parameter	Calculation Method
tid	$N_1.tid$ or $N_2.tid$
vLink	$N_1.vLink \wedge N_2.vLink$
loc	$(N_1.loc + N_2.loc)/2$
ts, *te*	$\text{Min}\{N_1.ts, N_2.ts\}$, $\text{Max}\{N_1.te, N_2.te\}$
agl	$(N_1.agl + N_2.agl)/2$

(2) If a picture is uploaded to the data center, its container leaf node's parent node will be *virtually pruned*.

(3) If two leaf nodes are similar, then the older one (i.e., the stored picture's timestamp is earlier) will be abandoned. So the leaf node has no siblings.

(4) Child nodes of a virtually pruned tree node is useless, and they will be really pruned, that is, deleted.

(5) The virtually pruned node (i.e., *vLink* = False) has the strongest epidemic capability.

(6) Any node can be grafted to a PicTree only if it is different from other nodes of this PicTree.

(7) Nodes of two branches are merged from top to bottom like a zip until nodes on a certain layer cannot be merged.

According to the conditions and valuation methods shown in Tables 2 and 3, we will use an example to show the PicTree fusion based on fusion rules mentioned above as well as to explain two operations: *pruning* and *grafting*. As shown in Figure 4, the WN wn_2 and the WN wn_3 combine their PicTrees with the PicTree T_1 of wn_1, respectively, where those same-named tree nodes are similar. There are two kinds of basic fusion modes operating in this figure: pruning and grafting.

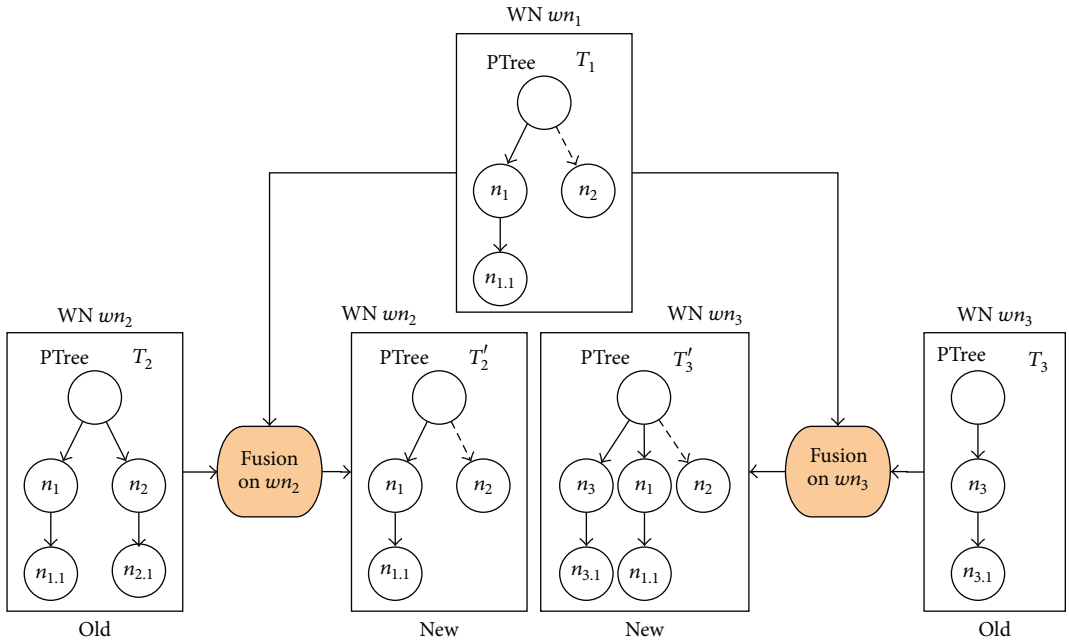


Figure 4: PicTree fusion of CooperSense. The dashed line means *vLink* = False. The left fusion is a pruning process while the right fusion is a grafting process.

(i) *Pruning*. The node n_2 on the PicTree T_2 of wn_2 (denoted by $T_2.n_2$) is infected to be virtually pruned because the node $T_1.n_2$ is virtually pruned and it has the highest epidemic capability. Then the leaf node $T_2.n_{2.1}$ will be really pruned because its parent node is virtually pruned. Pruning is used for useless epidemic termination.

(ii) *Grafting*. Nodes $T_1.n_1$, $T_1.n_2$ and their corresponding offspring nodes are grafted to the PicTree T_3 of the WN wn_3. Grafting is used to forward pictures.

Tree node fusion is a fundamental process in PicTree fusion. Next we will explain depth-first PicTree fusion algorithm.

4.1.2. Depth-First PicTree Fusion Algorithm. PicTree fusion uses the depth-first search. Given two PicTrees T_1 and T_2, the new PicTree T_1' is the result of merging T_1 and T_2. The main algorithm is shown in Algorithm 1. Functions used in this algorithm are *Combine*() and *CombineNode*(), which are shown in Algorithms 2 and 3, respectively. Function *similar*(,) in Algorithm 3 used to assess two nodes' similarity is calculated according to Table 2.

4.1.3. Fusion-Based PicTree Generation. PicTree fusion can be used not only for multi-WNs but also for single WN. Each WN has a PicTree, which is initially copied from the data center and further grows during the picture-taking and picture forwarding procedures.

Given a task (TSK) *tsk* and its task constraints (TSKC) *tc*, when a picture PIC p of this task is taken, a single-branch PicTree will be generated. Tree nodes of this new one-branch

Input: PicTree T_1 and PicTree T_2
Output: New PicTree T_1
$rn_1 = RootNodeOf(T_1)$;
$rn_2 = RootNodeOf(T_2)$;
$rn_1 = CombineNode(rn_1, rn_2)$;
Return rn_1.

Algorithm 1: Depth-first node search and PicTree fusion.

Input: Two tree nodes n_1, n_2
Output: New tree node n_1
$n_1.vLink = n_1.vLink \wedge n_2.vLink$;
Calculate other parameters of n_1 according to Table 3;
$n_1 = CombineNode\ (n_1, n_2)$;
Return n_1.

Algorithm 2: **Function** *Combine* (n_1, n_2).

PicTree are initiated according to Table 1. *vLink*s of all tree nodes are valued True, $L^d.ts = p.tm$, and $L^d.te = p.tm$. Other parameters of tree nodes are equal to attributes of *tsk*, *tc*, and p, such as $L^l.loc = p.loc$, $L^l.c\_loc = tc.c\_loc$. The PicTree will grow by merging the old PicTree and the new one-branch PicTree. If there are multitasks and more than two pictures, a complex multibranch PicTree will be generated through the PicTree fusion. As shown in Figure 5, the PicTree is generated with $\{p_1, p_2, p_3, p_4\}$. Because picture p_2 is similar to p_3, p_2 will be deleted and replaced by p_3.

Input: Two tree nodes n_1, n_2
Output: The new tree node n_1
  If n_1 is on the bottom layer L^i **Then**
    return n_1;
  End if
  $CN_1 = AllChildNodesOf\ (n_1)$;
  $CN_2 = AllChildNodesOf\ (n_2)$;
  For $\forall nn_2 \in CN_2$ **do**
    $haveCombined = False$;
    For $\forall nn_1 \in CN_1$ **do**
      If $similar(nn_1, nn_2) = True$ **Then**
        $nn_1 = Combine\ (nn_1, nn_2)$;
        $haveCombined = True$;
        Break;
      End if
    End for
    If $haveCombined = False$ **Then**
      nn_2 is grafted as a child node to the parent node of nn_1;
    End if
  End for
Return n_1.

Algorithm 3: **Function** *CombineNode* (n_1, n_2).

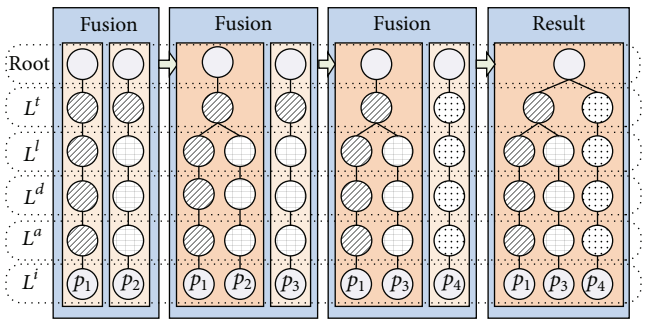


Figure 5: A PicTree is generated through three fusions from left to right with four pictures. Same colored nodes in the same layer are similar nodes.

4.2. Fusion-Based Picture Forwarding. In this section, how to utilize PicTree fusion to forward useful picture and terminate useless epidemic will be explained.

4.2.1. A PicTree Fusion Example. Both virtual/real grafting and virtual/real pruning are utilized for three purposes. (i) The picture file will be forwarded only if there is a real grafting and the picture is sent from the subbranch provider to the subbranch receiver. (ii) The accomplished task and copies of uploaded pictures will be eliminated through the virtual grafting and the virtual pruning. (iii) The storage space is saved through the real pruning.

Figure 4 basically explained the PicTree fusion, but how to utilize this fusion to forward pictures is still a question. Another PicTree fusion example was shown in Figure 6. wn_1 and wn_2 are two working WNs. Picture sets carried by them are $P^{wn_1} = \{p_1, p_2\}$ and $P^{wn_2} = \{p_3, p_4, p_5\}$. The WN wn_3 is a new WN; it downloads collection logs from the data center and enters into the observed area, so wn_3 took no pictures; that is, $P^{wn_3} = \{\}$. There are two forwarding procedures in Figure 6. (1) The WN wn_2 encounters wn_1. The PicTree fusion results are that wn_2 receives a picture p_2 from wn_1, and wn_1 receives two pictures p_4 and p_5 from wn_2. Pictures p_1 and p_3 are similar, so they are not forwarded. (2) The WN wn_2 encounters wn_3. Some branches of PicTree in wn_2 are virtually pruned and some are really pruned, which make wn_3 receive only one picture p_2.

4.2.2. Picture Forwarding. When two WNs encounter, they would like to receive pictures that they do not have. In order to select pictures, each WN costs a very small amount of traffic to receive the collection log from the other WN, and according to the collection log, the WN knows which picture needs be sent to and received from the other.

As shown in Figure 6, before these three WNs encounter, the effective pictures carried by wn_1, wn_2, and wn_3 are $\{p_1, p_2\}$, $\{p_3, p_4, p_5\}$, and $\{\}$, respectively. After wn_2 encounters wn_1 and wn_3, pictures carried by wn_1, wn_2, and wn_3 are $\{p_1, p_2, p_4, p_5\}$, $\{p_2, p_3, p_4, p_5\}$, and $\{p_2\}$, respectively. Pictures carried by each WN are different and should be uploaded.

4.2.3. Useless Epidemic Termination. The information of both accomplished and unaccomplished tasks is downloaded by WNs. In order to mark the accomplished task, a leaf tree node in the layer L^t corresponding to the task is created and its *vLink* is set to False. By the aid of the PicTree fusion, other WNs in the observed area will prune the branch mapped with this accomplished task. Furthermore, the WN will download the PicTree of uploaded pictures, whose *vLink*s of all tree nodes on the L^a layer are valued False. Here those virtually pruned nodes mean that the picture with the context

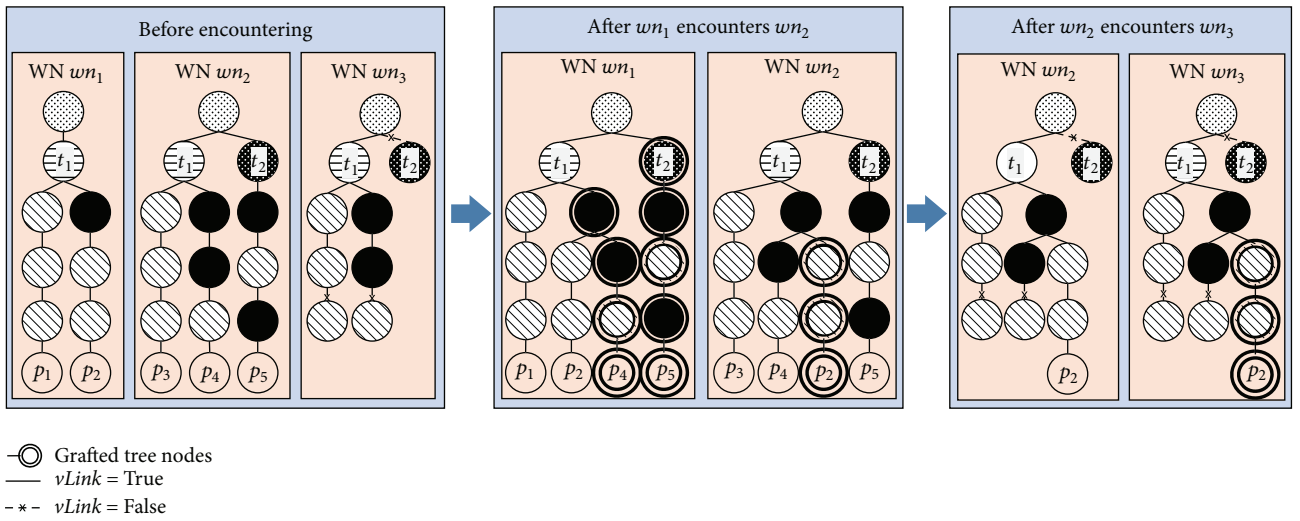


FIGURE 6: PicTree fusion of CooperSense. There are 3 WNs $\{wn_1, wn_2, wn_3\}$, 5 pictures $\{p_1, p_2, \ldots, p_5\}$, and 2 tasks $\{t_1, t_2\}$. wn_2 encounters wn_1 first and then encounters wn_3. Same colored nodes in the same layer are similar nodes.

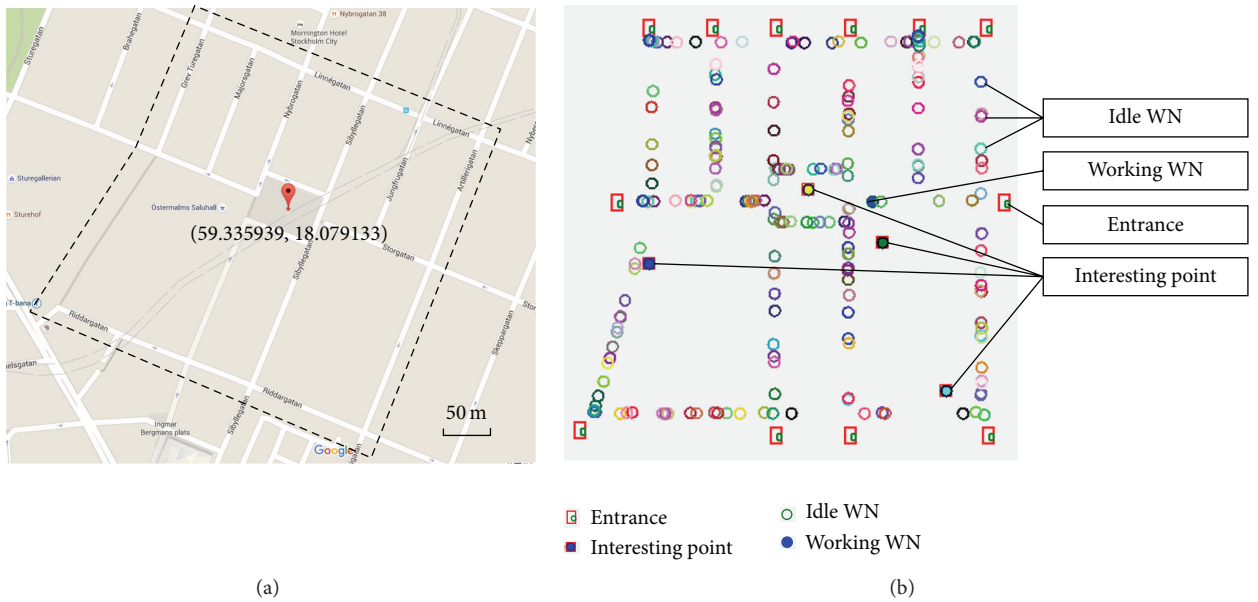


FIGURE 7: Simulation scenarios. (a) The electronic map of the observed area. It is Ostermalm area in downtown Stockholm. (b) Simulated moving WNs and interesting points. And medium-density WNs are simulated and shown in the map.

(denoted by tree nodes of this branch) has been uploaded to the data center already.

As shown in Figure 6, the task t_2 in the WN wn_3 is accomplished and the branch is virtually pruned, and after the WN wn_2 combines its PicTree with PicTree of wn_3, the task t_2 in the WN wn_2 is marked accomplished via a virtually pruned branch.

5. Performance Evaluation

In this section, we will evaluate the performance of CooperSense from different facets, including overhead, efficiency, traffic, and storage.

5.1. Simulation Setup. The Legion Studio [23] traces we use in this study are available at CRAWDAD [20], which are microsimulation of pedestrian mobility Contributed in the Web. They are simulated data set by the work in [13]. These traces are from a simulation of walkers in a part of downtown Stockholm in which several parameters are varied. Sparse and medium-density dataset are used here. The Ostermalm area consists of a grid of interconnected streets shown in Figure 7 and 4 MCP tasks are defined with 4 interesting points. Twelve entrances connect the observed area with the outside world. The active area of the outdoor scenario is 5,872 m^2.

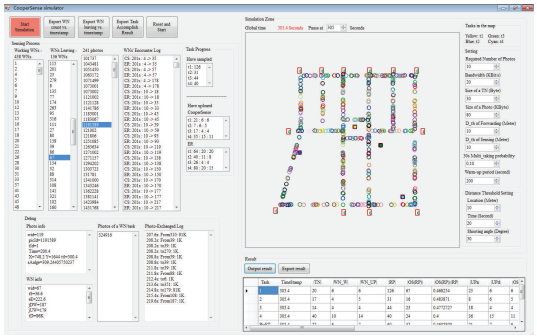


Figure 8: The interface of the simulator of CooperSense (CS: CooperSense; ER: Epidemic Routing). Medium-density WNs are used.

We started collecting data after a warm-up period because the system starts empty. After that we kept collecting observations until no WN forwarded pictures any more. The reason for that is a WN will not take and forward pictures when a WN learns that the task is accomplished from others through exchanging collection logs, and some WNs will still perform tasks when the data center has already got plenty of pictures for the task. The simulator of CooperSense is shown in Figure 8. In the simulation zone, the WN denoted by a color circle is moving according to the trajectory dataset of CRAWDAD [24].

In order to simulate the forwarding behavior of the WNs, we define four rules as follows:

(1) The data and task centers are connected with the outside world, so the data center can receive task requirements from the task center.

(2) If a WN is close to the interesting point (i.e., *whr* of a task) enough, then it will take pictures as a working WN. A WN might take multiple pictures from different positions for one task, and shooting angles will be computed based on this WN's trajectory with the range $[0, 2\pi)$, which is illustrated in Figure 7(b).

(3) When enough pictures have been uploaded to the data center, the corresponding task will be marked as accomplished, but the WN might still upload pictures.

(4) Once two WNs are close enough, only if the link duration time is larger than the transmission time, then data delivery can be finished.

5.2. Baseline and Metrics. ER (Epidemic Routing) [11] is a popular opportunistic forwarding approach, and we will compare CooperSense with it. In the simulation, each WN of ER has a set of pictures and a set of tasks. The flag of the accomplished task is set to False, all unexpired tasks are downloaded like CooperSense, and each WN forwards pictures and tasks' flags.

Different metrics are defined in this paper to evaluate the method, including traffic-saving, storage-saving, efficiency, effectiveness, and coverage. Detailed metrics are defined in the following sections along with the experimental results.

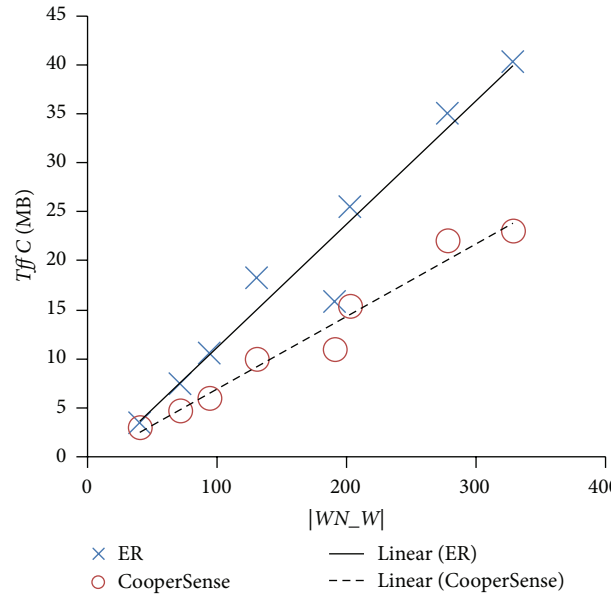


Figure 9: Experimental result of comparing the traffic-saving performance of CooperSense and ER. All tasks are accomplished.

5.3. Experimental Result

5.3.1. Energy-Saving Evaluation. Data transmission is the most energy-consuming process, so we use traffics of forwarding pictures (denoted by *TffC*) and ratio of extra uploaded pictures (denoted by *ExUpR*) to evaluate the energy-saving performance. *ExUpR* denotes the ratio of extra uploaded pictures calculated with (1), where *UPd* denotes the set of uploaded pictures, *UPd_op* denotes the optimal diversified subset [2] of *UPd* that contains valuable pictures, and *tsk* is a task TSK:

$$ExUpR = \frac{|UPd\_op| - tsk.ntr}{tsk.ntr}. \quad (1)$$

WN_W denotes the set of working WNs in the observed area. Because different WN enters and leaves uncertainly, *WN_W* at different time points are different. So $|WN\_W|$ in the experiment has an average value.

As shown in Figure 9, *TffC* of ER is much higher than that of CooperSense, especially when $|WN\_W|$ is high. This is due to the fact that more pictures will be forwarded when the density of *WN* increases and ER creates more useless forwarding.

ExUpR are utilized to evaluate the extra overhead. Epidemic termination is the way to decrease useless forwarding, which is evaluated by the metric *ExUpR*. As shown in Figure 10, most *ExUpR*s of both ER and CooperSense are close and less than 50%, but sometimes *ExUpR* of CooperSense is much lower than that of ER.

Through comparisons of *TffC* and *ExUpR* between ER and CooperSense, it is obvious that the latter can save more traffic overhead.

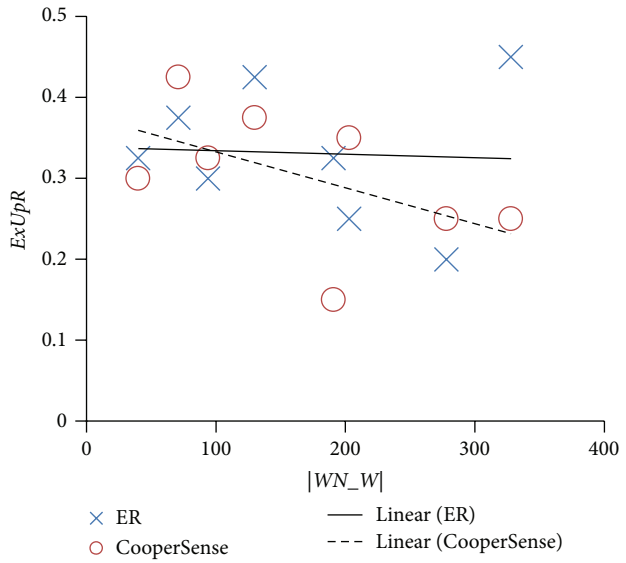


Figure 10: Experimental result of comparing the traffic-saving performance of CooperSense and ER. All tasks are accomplished.

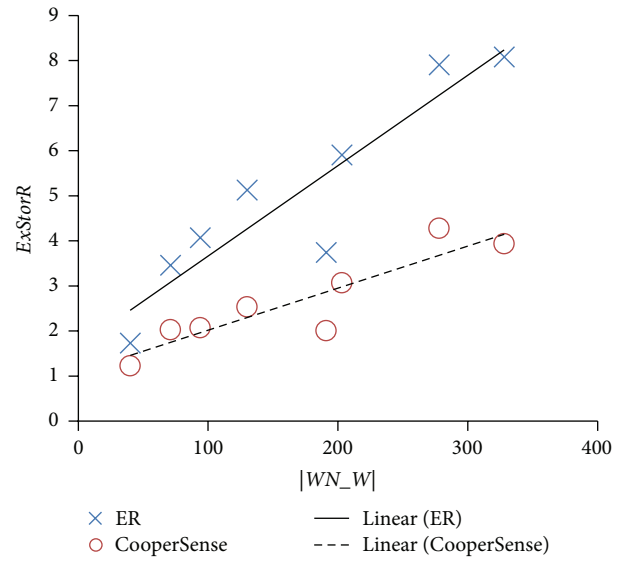


Figure 11: Experimental result of comparing the storage-saving performance of CooperSense and ER. All tasks are accomplished.

5.3.2. Storage-Saving Evaluation. *ExStorR* denotes extra storage overhead ratio calculated with (2), where *RP* refers to the set of raw pictures taken by cameras and *CP* denotes to the set of pictures (including copies) carried by all WNs:

$$ExStorR = \frac{|CP| - |RP|}{|RP|}. \quad (2)$$

As shown in Figure 11, *ExStorR* of ER is much higher than that of CooperSense. Therefore, CooperSense saves storage overhead more effectively. CooperSense uses PTree-fusion to stop forwarding redundant pictures, so the storage overhead and the traffic overhead are saved. Although ER can also use variables to mark already uploaded pictures and find duplicate pictures, for MCP, its traffic overhead is still higher than CooperSense's because WNs of ER must send complete PICs while WNs of CooperSense only need to send a certain part of PICs according to the PTree-fusion context.

5.3.3. Efficiency Evaluation. Delay of uploading pictures is used to evaluate the efficiency of opportunistic forwarding method. *AvgDelay* refers to the average delay of uploaded pictures calculated with (3), where *tArr* denotes the timestamp of the picture being uploaded to the data center. *TmC* denotes average time-consumption of a task:

$$AvgDelay = \frac{\sum_{p \in UPd} (p.tArr - p.tm)}{|UPd|}. \quad (3)$$

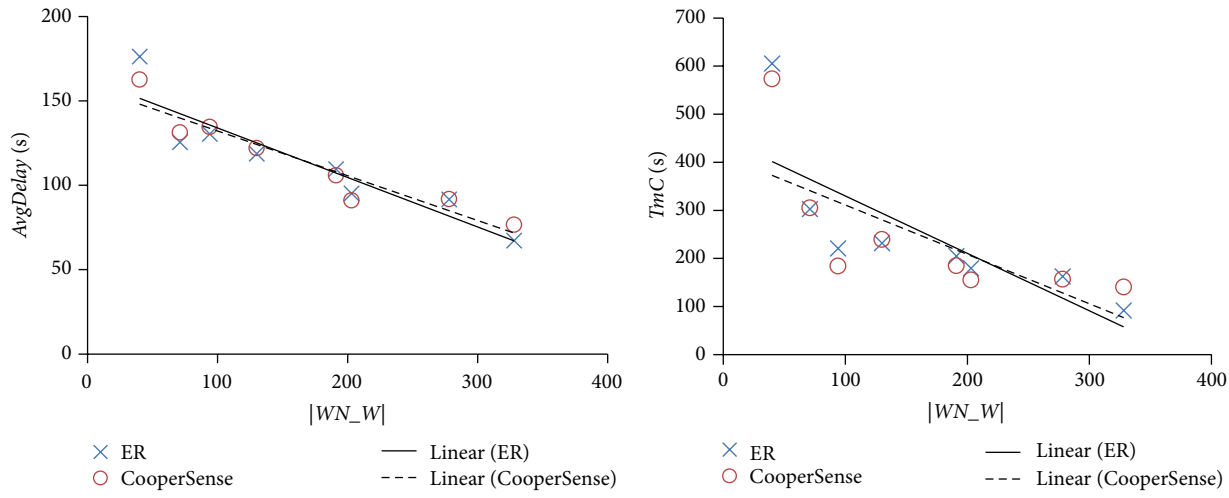
Efficiency evaluation is shown in Figure 12, where both *AvgDelay* and *TmC* of ER and CooperSense are close. Generally, ER is an efficient way to upload pictures, but the result shows that CooperSense provides a similar efficiency as ER can do. On the one hand, although WNs of CooperSense forward fewer pictures via the opportunistic network than those of ER do, CooperSense's efficiency for task performing is not influenced. On the other hand, both *AvgDelay* and *TmC* decline as $|WN\_W|$ increases, so the efficiency of opportunistic forwarding method cannot be improved by increasing the number of PIC copies but by involving more WNs. Hence the selective forwarding mode utilized by CooperSense consumes less resource on the promise of efficiently accomplishing tasks.

5.3.4. Effectiveness Evaluation. In order to evaluate the effectiveness of our method, we define metrics to indicate how much opportunity of uploading pictures our method can utilize. *OppUpR* denotes the ratio of opportunistically uploaded pictures to the data center calculated by (4), where the picture set *UPo* ($UPo \subseteq UPd$) consists of pictures that are not uploaded by their photo-takers:

$$OppUpR = \frac{|UPo|}{|UPd| - |UPo|}. \quad (4)$$

Effectiveness evaluation result is shown in Figure 13. *OppUpR* of them have no clear difference; therefore, CooperSense and ER have similar effectiveness of opportunistically uploading pictures. Additionally, but when $|WN\_W|$ increases, *OppUpR* of them increases because more people could opportunistically carry pictures.

5.3.5. Coverage Evaluation. Coverage evaluation is based on four metrics: $|UPa|$ is the number of those pictures that are carried out of the observed area, $|UPd|$ is the number of uploaded pictures, *DR* denotes the ratio of delivered pictures



(a) Average delay of uploading one picture

(b) Average time-consumption of a task

Figure 12: Experimental results of the efficiency of CooperSense and ER.

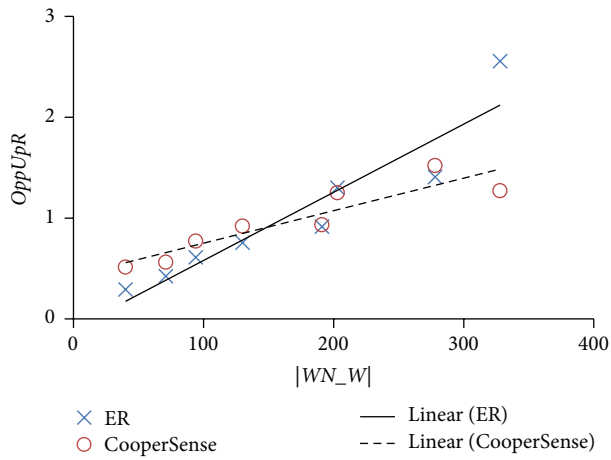


Figure 13: Comparison between ER and CooperSense on the ratio of opportunistically uploaded pictures.

calculated with (5), and $OpUpR$ denotes the ratio of optimal uploaded pictures calculated by (6):

$$DR = \frac{|UPd|}{|RP|}, \tag{5}$$

$$OpUpR = \frac{|UPd\_op|}{|UPd|}. \tag{6}$$

The comparison results are shown in Figure 14. $|UPa|$ of ER are much larger than that of CooperSense and $|UPd|$ are similar, which proofs that CooperSense can save storage overhead comparing with ER. Additionally, since $|UPa|$ of CooperSense is less than that of ER but $OpUpR$ of CooperSense is a little higher than that of ER, this means that UPd of CooperSense is more diverse than that of ER.

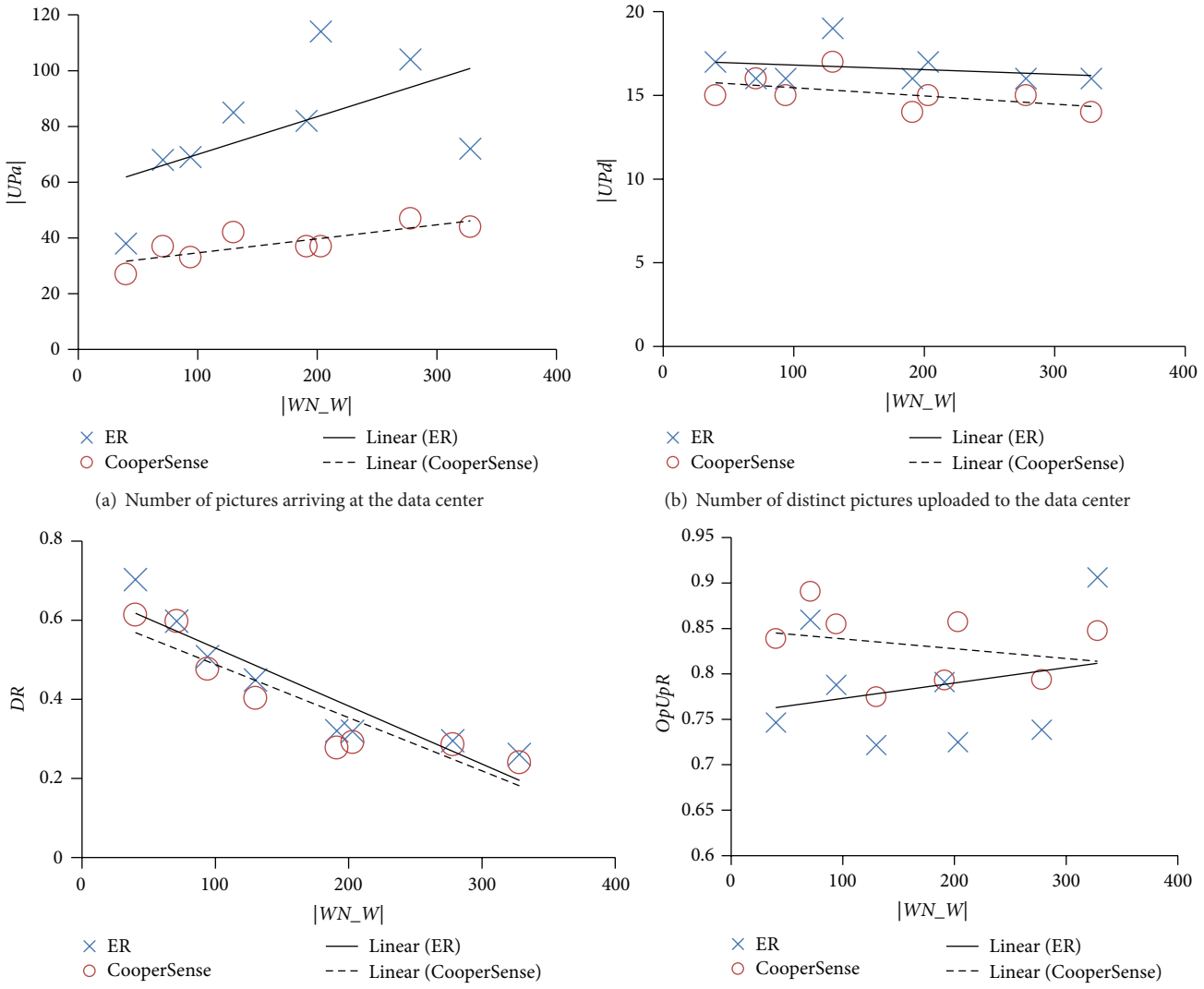
As experimental results show, CooperSense almost has the same picture collection capability of ER while it saves much traffic and storage overhead comparing to ER. The PTree-fusion method utilizes a tree structure to store picture collection logs and PTree-fusion is an efficient way to exchange different data among WNs for MCP in MON. Therefore, our method is flexible, which can selectively forward valuable pictures without complex computations.

5.4. Discussion. CooperSense forwards pictures through exchanging PicTree of encountered worker nodes, and there are still some issues that are not explained in this paper, some of which are listed below:

(1) The visual feature is not used during the PicTree fusion. In this paper, we do not include the visual feature of pictures to assess the similarity because the visual similarity measurement is highly related to the application. However, the image layer can be easily inserted when it is needed in real applications because of the flexibility of PicTree construction.

(2) Some works on opportunistic forwarding leverage the prediction of people's position to select the right people to carry and forward data, which is more adaptable for delay tolerant tasks. Since reducing delay is the goal of this scenario, we do not use this method in this paper, but it may be utilized in our future work.

(3) Although an emergency scenario is used to test our work, cooperative crowd sensing is also applicable in some crowed spots, such as in a thousand-attendee conference, a football match, or a singer show. Defining a task and its task constraint, using CooperSense, people can receive various pictures shared by others nearby who have better or different views without using the limited and narrow-bandwidth cellular communication.



(a) Number of pictures arriving at the data center

(b) Number of distinct pictures uploaded to the data center

(c) Ratio of delivered pictures

(d) Ratio of optimal uploaded pictures

Figure 14: Experimental result of comparing the performance of CooperSense and ER.

6. Conclusion and Future Work

Above all, CooperSense is a novel cooperative and selective data forwarding framework for MCP. In this paper, the Pic-Tree model and its fusion method were proposed for participant collaboration and selective data forwarding. Experiment results indicate that CooperSense has higher performance in terms of storage and network cost than ER. It is planned to be used in city-sense project for large-scale user study, and all the methods will be further improved accordingly.

Competing Interests

The authors declare that they have no competing interests.

Acknowledgments

This work was partially supported by the National Basic Research Program of China (no. 2015CB352400) and the National Natural Science Foundation of China (no. 61332005, 61373119, 61222209).

References

[1] B. Guo, Z. Wang, Z. Yu et al., "Mobile crowd sensing and computing: the review of an emerging human-powered sensing paradigm," *ACM Computing Surveys*, vol. 48, no. 1, article 7, 2015.

[2] H. Chen, B. Guo, S. Krishnaswamy, Z. Yu, and L. Chen, "Crowdpic: a multi-coverage picture collection framework for mobile crowd photographing," in *Proceedings of the UIC*, pp. 68–76, IEEE, 2015.

[3] S. Kim, C. Robson, T. Zimmerman, J. Pierce, and E. M. Haber, "Creek watch: pairing usefulness and usability for successful citizen science," in *Proceedings of the 29th Annual CHI Conference on Human Factors in Computing Systems (CHI '11)*, pp. 2125–2134, ACM, Vancouver, Canada, May 2011.

[4] B. Guo, H. Chen, Z. Yu, X. Xie, S. Huangfu, and D. Zhang, "FlierMeet: a mobile crowdsensing system for cross-space

public information reposting, tagging, and sharing," *IEEE Transactions on Mobile Computing*, vol. 14, no. 10, pp. 2020–2033, 2015.

[5] K. Tuite, N. Snavely, D.-Y. Hsiao, N. Tabing, and Z. Popović, "PhotoCity: training experts at large-scale image acquisition through a competitive game," in *Proceedings of the 29th Annual CHI Conference on Human Factors in Computing Systems (CHI '11)*, pp. 1383–1392, ACM, Vancouver, Canada, May 2011.

[6] Miit: Tianjin explosions have no serious impact on local communications network, http://en.people.cn/n/2015/0813/c98649-8935580.html.

[7] M. Y. S. Uddin, H. Wang, F. Saremi, G.-J. Qi, T. Abdelzaher, and T. Huang, "PhotoNet: a similarity-aware picture delivery service for situation awareness," in *Proceedings of the 32nd IEEE Real-Time Systems Symposium (RTSS '11)*, pp. 317–326, IEEE, Vienna, Austria, December 2011.

[8] A. Zaslavsky, P. P. Jayaraman, and S. Krishnaswamy, "ShareLikesCrowd: mobile analytics for participatory sensing and crowdsourcing applications," in *Proceedings of the IEEE 29th International Conference on Data Engineering Workshops (ICDEW '13)*, pp. 128–135, IEEE, Brisbane, Australia, April 2013.

[9] J. T. B. Fajardo, K. Yasumoto, and M. Ito, "Content-based data prioritization for fast disaster images collection in delay tolerant network," in *Proceedings of the 7th International Conference on Mobile Computing and Ubiquitous Networking (ICMU '14)*, pp. 147–152, Singapore, January 2014.

[10] M. Conti, S. Giordano, M. May, and A. Passarella, "From opportunistic networks to opportunistic computing," *IEEE Communications Magazine*, vol. 48, no. 9, pp. 126–139, 2010.

[11] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," Tech. Rep. CS-200006, Duke University, 2000.

[12] U. Weinsberg, Q. Li, N. Taft et al., "CARE: content aware redundancy elimination for challenged networks," in *Proceedings of the 11th ACM Workshop on Hot Topics in Networks (HotNets '12)*, pp. 127–132, ACM, Redmond, Wash, USA, October 2012.

[13] B. Guo, Z. Yu, D. Zhang, and X. Zhou, "Cross-community sensing and mining," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 144–152, 2014.

[14] H. Ma, D. Zhao, and P. Yuan, "Opportunities in mobile crowd sensing," *IEEE Communications Magazine*, vol. 52, no. 8, pp. 29–35, 2014.

[15] Y. Jiang, X. Xu, P. Terlecky, T. Abdelzaher, A. Bar-Noy, and R. Govindan, "Mediascope: selective on-demand media retrieval from mobile devices," in *Proceedings of the 12th International Conference on Information Processing in Sensor Networks (IPSN '13)*, pp. 289–300, ACM, Philadelphia, Pa, USA, 2013.

[16] D. Zhao, H. Ma, S. Tang, and X.-Y. Li, "COUPON: a cooperative framework for building sensing maps in mobile opportunistic networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 2, pp. 392–402, 2015.

[17] Z. Yu, H. Xu, Z. Yang, and B. Guo, "Personalized travel package with multi-point-of-interest recommendation based on crowdsourced user footprints," *IEEE Transactions on Human-Machine Systems*, vol. 46, no. 1, pp. 151–158, 2016.

[18] M. Soltani, M. Hempel, and H. Sharif, "Utilization of convex optimization for data fusion-driven sensor management in WSNs," in *Proceedings of the International Wireless Communications and Mobile Computing Conference (IWCMC '15)*, pp. 1224–1229, IEEE, Dubrovnik, Croatia, August 2015.

[19] N. T. Baranasuriya, S. L. Gilbert, C. Newport, and J. Rao, "Aggregation in smartphone sensor networks," in *Proceedings of the 9th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS '14)*, pp. 101–110, IEEE, Marina Del Rey, Calif, USA, May 2014.

[20] K.-Y. Chow, K.-S. Lui, and E. Y. Lam, "Efficient on-demand image transmission in visual sensor networks," *Eurasip Journal on Advances in Signal Processing*, vol. 2007, Article ID 95076, 1 page, 2007.

[21] F. Luqman, F.-T. Sun, H.-T. Cheng, S. Buthpitiya, and M. Griss, "Prioritizing data in emergency response based on context, message content and role," in *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief (ACWR '11)*, pp. 63–69, ACM, December 2011.

[22] Y. Wang, W. Hu, Y. Wu, and G. Cao, "Smartphoto: a resource-aware crowdsourcing approach for image sensing with smartphones," in *Proceedings of the 15th ACM International Symposium on Mobile ad hoc Networking and Computing (MobiHoc '14)*, pp. 113–122, ACM, Philadelphia, Pa, USA, August 2014.

[23] Legion studio, http://www.legion.com/.

[24] S. T. Kouyoumdjieva, Ó. R. Helgason, and G. Karlsson, *CRAWDAD Data Set kth/Walkers (v.2014-05-05)*, 2014, http://crawdad.org/kth/walkers/.