Distributed Crowd Flow Analysis Using Passive Packet Sniffing

Internet of things Seminar IN4398

Jetse Brouwer 4615964
David Enthoven 4502124
Niels Hokke 4610148

***Abstract*—This paper proposes a distributed sensor network for analyzing crowd flow. This enables municipalities, urban developers, city planners and event organizers to study, analyze and optimize crowd flow. The tracking of the crowd will be done by multiple sensor-nodes strategically placed at crowd flow bottlenecks. To measure crowd flow, Wi-Fi activity of smart phones, tablets and other hand-held devices is monitored. Detecting the same devices at different locations over time provides a estimation of the movement of a single device. By using the distributed algorithm developed by the authors and the data gathered by the sensors an insight into the traffic flow on a three-way junction was given.**

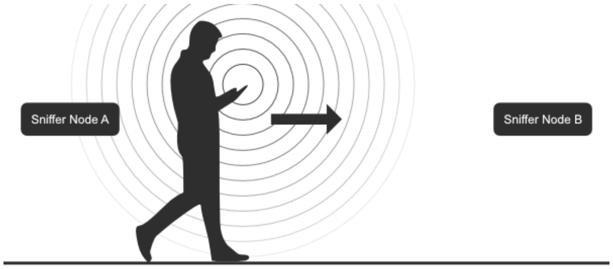

Fig. 1. The movement of a device between sniffing nodes

I. Introduction

Even when not connected, smart phones and other hand held devices still transmit Wi-Fi messages to search for nearby access points. The number of such packets transmitted may be up to 2000 per hour for an active phone [1]. Devices to which these messages are not addressed will simply ignore them under normal circumstances. It is possible however to capture and log these messages even if your device is not the desired destination device.

By changing the settings of a device, one is able to perform this so-call sniffing, and record all Wi-Fi transmitted messages surrounding the device. Since these messages includes information about the sender in the form of a MAC address it can be used to uniquely identify the device.

In this paper a distributed network of strategically placed sniffing devices is proposed which will exchange data about devices that pass by the nodes. This makes it possible to track the movements on a per device basis. This is illustrated in Figure 1.

II. High level overview

The proposed network of sniffing nodes will use a low-cost Wi-Fi chip capable of sniffing for packets and means of communicating with other nodes. It will use the MAC address to uniquely identify separate devices. When deploying several of these spread out across roads and near traffic junctions, hand-held devices will move in and out of range of these sensor. Using this data the algorithm analyzes the flows of crowds.

To ensure privacy, the nodes can communicate using a symmetric encryption scheme, as these can be implemented very efficient on lower power devices. A node then only discloses its key to it's direct neighbors, and uses the secure channel to privately transmit the detected mac addresses. For reporting the measurements it only transmits the amount of devices tracked per given direction, and no uniquely identifiable data.

III. Algorithm

The algorithm that was developed for this project supports both direct and indirect detection. This results in the ability of detecting a device's movement, even when an intermediate node fails to detect the device. The algorithm is designed in such a manner that it only needs to communicate with its one-hop neighbor, as the nodes are placed in geometrical graph the average degree does not significantly increase for a growing network.

A. Direct Detection

On detection of a packet the node has to figure out if and where the sender has been seen before. To do this every node keeps a list of sightings of it's one-hop neighbors, if the sender of the packet is found there, it knows it the device came from that position. After performing these checks it broadcasts the detected mac address and timestamp to its one-hop neighbors.

In order to ignore devices that move from one node to another through a detour (while not being detected by a in between node) a maximum travel time between nodes is defined. This means nodes can delete sightings from neighbors older than the maximum travel time, which has as a side effect reduced look-up time and memory consumption.

B. Indirect detection

As devices aren't continuously transmitting and nodes can't sense on multiple channels simultaneous, there is a possibility

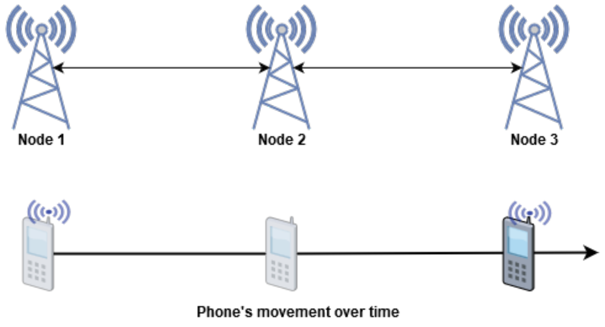

Fig. 2. Example of a hidden detection where the phone moves from node 1 to node 3 past node 2, but node 2 does not receive a packet.

that a device passes by a node undetected. To combat this we've implemented the notion of indirect detection. Take the example in figure 2 where node 1 and node 3 detect the device, but node 2 doesn't. Since node 3 isn't a one-hop neighbor of node 1 it doesn't know about the earlier sighting. however, since node 2 can see the broadcast of the two neighbors it can deduce that it must have traveled passed him without him noticing it. It should only do so if the node self didn't detect him passing and there is no direct path between node 1 and node 3 (to prevent double detection).

Algorithm 1: onPacket(SenderID, Mac address, timestamp)

```
if not seen in the last 30 seconds then
    foreach neighbor's detection list do
        remove entries older then max buffer time;
        if mac address in detection list then
            remove from detection list;
            store route (neighbor ID, own Id, list time,
              timestamp);
    broadcast (Mac address, timestamp);
```

Algorithm 2: onBroadcast(SenderID, Mac address, timestamp)

```
foreach neighbor's detection list do
    remove entries older then max buffer time;
    if mac address in detection list then
        remove from detection list;
        if no link between SenderID and neighbor and
          not directly detected then
            store indirect route (neighborID, SenderID,
              list time, timestamp);
add to SenderID's detectionlist;
```

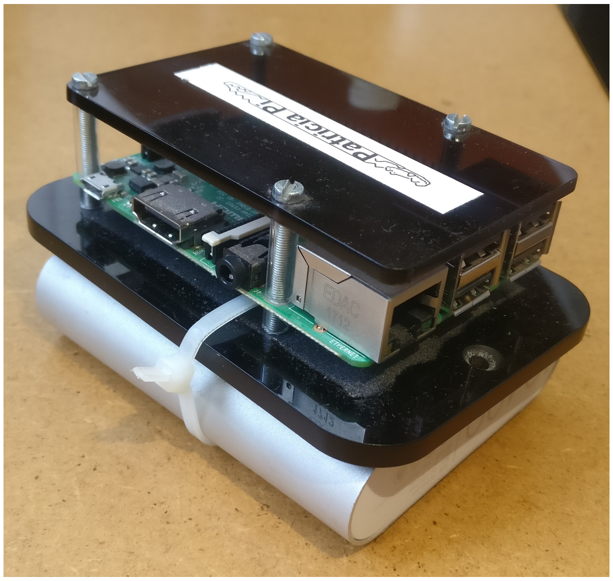

Fig. 3. One of the three measuring nodes, existing of a Raspberry Pi 3b and a powerbank.

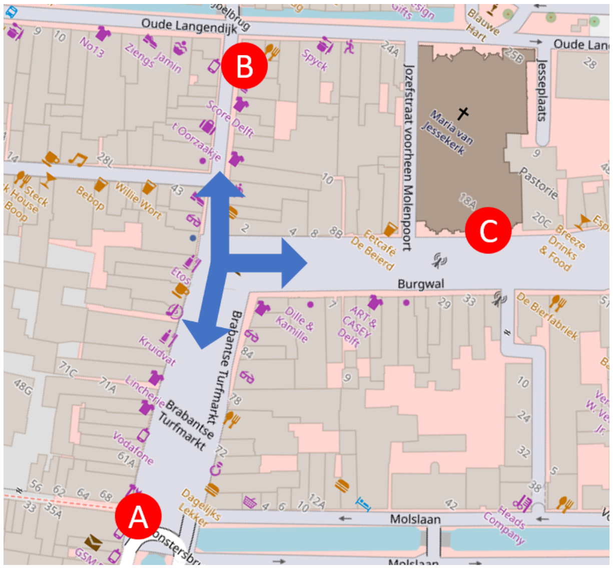

Fig. 4. The deployment of the nodes for the experiment. The blue arrows indicates the three way junction being monitored and the red dots the locations of the nodes.

IV. Experiments

To validate the idea, a proof of concept experiment was done. The algorithms described in section III were implemented on three Raspberry Pis, as seen in figure 3. Using powerbank's the three Pis were able to run in remote locations. For the location a three-way junction in the center of Delft was chosen, see figure 4. This location was chosen because it forms a three-way junction without to many allays leaving the junction. An other reason was because at this location a market is held which guarantees a reasonable flow of people.

To increase reproducibility the raw data was recorded instead of being directly processed on the Raspberry Pis. later the recorded data is played back to the Pi's to emulate the actual packet detection. This enables fast development and testing as the algorithm can be tested on hours of data in a matter of seconds.

To validate our results, a short video of 30 seconds at the location of the nodes was recorded every 5 minutes. Using this footage the actual flow was estimated by manually counting

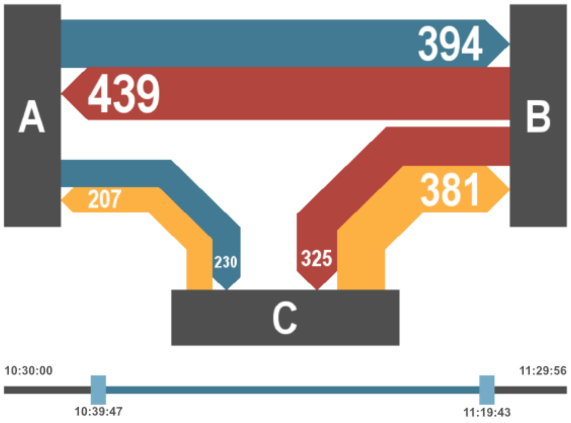

Fig. 5. Bidirectional sankey diagram to visualize the crowd flow in a set time window

people passing by. The assumption is made that the arrival-rate of the people within the five minutes can be approached by a poison distribution. Therefore, the thirty second footage is used to estimate the amount of passers-by for the whole five minute segment by extrapolating.

V. Results

The results of the experiment can be seen in table I. The direction column indicates whether the measured flow goes into the intersection or out of the intersection. In this table the in/out-flow estimated by the 30 second video is compared to the detected flow by the system. Taking the inaccuracy of the estimated flow in mind the detected flows approaches the estimated ones. The largest error is in node A, this large error might be due to node A being located in front of a phone shop with more than 40 active phones. These phones may have caused significantly more package collisions or an increase back-off time, causing node A to miss more of the actual flow.

Another proposition of the system is that it can determine the ratio between flows from A-to-B and flows from A-to-C. However, from the ground truth data it is not possible to determine these ratios. This because only the number of people going in/out the junction were counted, and not which branch of the three-way junction was taken.

If all the sniffed flows from table II are combined to get the detected flows per node direction in table I, and these flows are divide according to the measured ratios from tables III and IV, the exit rate for every node's in/outflow can be estimated. For example, the total outflow of C, is the inflow of A multiplied by the ratio A-to-C plus the total inflow of B multiplied by the ratio B-to-C.

$$C_{out} = A_{in} \times Ratio_{A \to C} + B_{in} \times Ratio_{B \to C}$$

The results of these value for all three nodes can be found in tables V and VI. When compared to the actual flows all calculated flows are within 17%. Except for node A who is 30% off, which is speculated to be influenced heavily by being in front of a phone store.

TABLE I
Estimated ground truth from videos and detected

Node	Direction	Video	Detected
A	In	973	730
	Out	1018	749
B	In	995	928
	Out	810	905
C	In	640	672
	Out	680	676

TABLE II
Detected Total Per Routes

Flow	Detected
A → C	267
B → C	409
B → A	519
C → A	230
A → B	463
C → B	442

TABLE III
Detected Inflow ratios

Flow	Ratio
A → B	63%
A → C	37%
B → A	56%
B → C	44%
C → A	66%
C → B	34%

TABLE IV
Detected Outflow ratios

Flow	Ratio
B → A	69%
C → A	31%
A → B	51%
C → B	49%
A → C	39%
B → C	61%

A. *Data visualization*

To provide an intuitive visualization of the crowd flow a tool was developed which illustrates the flow in the form of a bidirectional Sankey-like diagram. A screen shot of this tool is shown in Fig. 5. The range slider in the bottom of the interface provides a quick way to look at a specific time window.

VI. Discussion

The data shown is solely of one data-set. Although we have done multiple separate measurements of over one hour the gathered data of only the last one proved to be usable. The major contributor towards erroneous data gathering was that the nodes were placed to close to one another. This meant that a significantly large number of devices were measured to go back and forth between the nodes, whilst in reality they were just standing still in a position which was detectable by two or three nodes. The Second reoccurring problem was that for some reason the sniffing program shut of mid measurement which resulted in a lot of frustrated hair-pulling.

VII. Conclusion

A distribute network of sniffer node modules may very well be used to reliable map and analyze the crowd flow over a certain area. When using the described algorithms each node is able to keep a list of the route which the passing devices have taken. When the intermediate connection between nodes is simulated the measured passers by are remarkably close to the measured amount of devices passing by.

The experimental results show a promising correlation between the amount of detected devices and the actual number

TABLE V
Detected vs actual inflow

Node	calc.	Count.	Ratio
A	683	973	70%
B	1117	995	112%
C	708	640	111%

TABLE VI
Detected vs actual outflow

Node	calc.	Count.	Ratio
A	977	1018	96%
B	836	810	103%
C	794	680	117%

of passing devices. Although the number of observed passing devices are around 95% of the actual amount of passers by (with quite a bit of variance), it still provides quick and easy ballpark figure to analyze crowd flow

The added value of this approach is that it is able to track movement across a junction, instead of the directional throughput on a node. It does so without enabling global tracking of a single device which is a privacy preserving way to analyze crowd flow.

References

[1] Julien Freudiger, How Talkative is your Mobile Device? An Experimental Study of Wi-Fi Probe Requests, WiSec'15 June 22-26 2015, New York City, NY, USA