Summarising Wireless Network Datasets

Charlotte Knight

September 2019

Abstract

There is lots of current research which requires data about the usage of wireless networks, however lots of time and effort must be spent separating out useful information from the huge amount of data now collected by wireless network monitoring. The amount of data sent over wireless networks will only continue growing in years to come. During this project a tool has been developed which enables relevant information for network mobility research to be extracted from multiple input sources and converted to a single, more manageable output. Processing stages in the tools data flow have been separated into individual components to enable easy expansion to new input formats in the future, with the current tool handling only tcpdump and syslog input. The minimum file size reduction seen on real tcpdump traces was found to be 99.992%. The tool which has been developed has a quadratic dependency of execution time on the number of device-to-device associations in the input data. This dependency could become an issue if very large data sets need to be used with it.

Declaration

Contents

1 Introduction

The problem that this project is focused on is the efficient and useful summarisation of network trace data in order to aid research around wireless networks. This necessarily means removing information from the raw data which is not expected to be needed during the further processing of the data.

Wireless networks have rapidly gained popularity and increased in size and complexity in the past few years. This increasing demand for constant connectivity has caused research relating to network mobility, delay tolerant networking (DTN), dynamic offloading, and many similar areas to become extremely active. As researchers tackle these developing topics in wireless technology they inevitably have to look at data collected from real wireless networks. This data will include a lot of information which is not relevant to the research being completed. As transmission capacities and speeds in wireless networks have increased, the data collected over them using common methods such as tcpdump has become complex. It may take a significant amount of time to extract the useful information from such raw data. A single tool with the ability to extract commonly used information from raw network data and present it in a easily utilisable form for researchers could save time and effort.

This project document the development of such a tool; it is able to take large data sets from the collection of information over wireless networks and reduce the information content for easier onward processing. Wireless network traces are used in a wide variety of research areas and it would be impractical to consider each and every use case. The CRAWDAD archive contains some frequently used datasets of large scale which will be the focus of this project, however the tool may be used with other data as long as it is in tcpdump or syslog format. The past usage of the CRAWDAD datasets shows popularity mainly in network mobility research. Taking this into consideration the tool being developed will preserve information about encounters between mobile nodes in the network.

Encounter traces are used in several different areas of research. A tool which allows network data to be summarised into a format containing significantly less superfluous information could be beneficial to researchers since it will minimise the time needing to be spent extracting relevant information from the initial datasets. By using an output format which is already popularly used

As previously mentioned, the data which is being used as a initial focus for this project is primarily used for mobility research. Encounter information has been identified as the most beneficial information to retain from the initial input data. Research into existing intermediate formats used in mobility research proved frustrating; most research papers which were analysed gave vague (if any) descriptions of the data formats used. A common thing across several papers in multiple research areas was the use of the Opportunistic Network Environment (ONE) simulator. It would therefore be convenient for the output of the newly developed tool to be compatible with the ONE simulator. This would allow onward processing using the simulator to be undertaken by researchers.

During this project the CRAWDAD dartmouth/campus dataset [13] has been the initial focus of development. Early decisions were made based on what would be most beneficial for use with the dartmouth/campus data. This was because the dataset was identified early on as one of the most frequently used datasets regarding wireless network traces in the CRAWDAD archive. Despite this early focus on the dartmouth/campus data the data summarisation tool developed throughout this project is intended for use with any data of appropriate input format. Input formats which the tool accepts are tcpdump and syslog, although syslog input requires additional configuration information to be given. These are two extremely common formats for collection of network data and were chosen based on their availability and popularity. In addition to these two formats being compatible, the data flow has been separated into multiple processing stages to minimise the work needed in order to allow processing of new data formats. The output of the tool which has been developed is intended to facilitate easy onward processing of the summarised data. In order to meet this goal the output format is such that it can be used with the Opportunistic Network Environment simulator. The ONE simulator is commonly referenced in research papers relating to wireless networking and can take data in a strict format as input to define parameters of its simulation. To maximise the usefulness of this projects output is formatted so that it may be used as input for one of these simulations. The final tool produced during this project meets all of the high priority requirements as specified in section 4 of this document. The majority of the medium priority requirements are also met. A full evaluation of the satisfaction of these requirements, and a critical assessment of the final tool that has been developed, is given at the end of this report.

2 Context Survey

2.1 CRAWDAD Usage

2.1.1 Research

When completing this research the focus has been on one particular dataset from the CRAWDAD archive, dartmouth/campus [13]. This dataset was chosen because it is one of the most popular datasets in the archive, having been cited by 374 papers at the time of writing [2]. The most frequently cited dataset however is cambridge/haggle, the reasoning for deciding not to focus on this instead is that the Cambridge dataset is comparatively small in size and would therefore benefit much less from the summarisation which this project hopes to provide.

The dartmouth/campus dataset is collected over Dartmouth College's network. It is comprised of five years worth of data collected using various methods; tcpdump, syslog, and SNMP. As the tcpdump data is largest and most unfiltered, it would benefit most from this summarisation, and will be the initial focus. This data includes pcap files containing the headers of all transmissions sent via the Dartmouth College network from 27 on-campus buildings. More detailed information regarding the Dartmouth network can be found on the CRAWDAD

website [13].
The papers that have been selected for use in this research were chosen because they all cite the dartmouth/campus dataset. A Google Scholar [1] online search was used to retrieve the most "relevant" papers which used the chosen dataset, from these results the ones which have been most often cited in other work were selected. This selection process found papers which are relevant in the research community. As there are different versions of the dataset the search had to be repeated three times, once using the 2009 dataset, once with the 2007 dataset, and once with the 2005 dataset. For each search the five most cited results have been used. Table 1 shows a summary of the type of information each paper needed to use from the dartmouth/campus dataset. Papers in which the dataset was referenced but ultimately has not been used have been excluded.

		Properties Needed		
Paper	Topic	Device/AP Identification	Time of Transmission	Transmission Quality/Rate
Nextplace: a spatio-temporal prediction framework for pervasive systems, Scellato et al., 2011	Mobility	x	x	
Community-Aware Opportunistic Routing in Mobile Social Networks, Xiao, Wu, and Huang, 2014	Mobility	x	x	
On nodal encounter patterns in wireless LAN traces, Hsu and Helmy, 2010	Mobility	x	x	
Mobility models for systems evaluation, Musolesi and Mascolo, 2009	DTN	x	x	
Large-Scale Synthetic Social Mobile Networks with SWIM, Kosta, Mei, and Stefa, 2014	Mobility	x	x	
WAVEFORM DESIGN AND NETWORK SELECTION IN WIDEBAND SMALL CELL NETWORKS, Yang and Liu, 2014	Mobility	x		x

MAGA: A Mobility-Aware Computation Offloading Decision for Distributed Mobile Cloud Computing, Shi, Chen, and Xu, 2017	Mobility	x	x	
Flow-Based Management For Energy Efficient Campus Networks, Amokrane et al., 2015	SDN	x		x
Human behavior and challenges of anonymizing WLAN traces, Kumar and Helmy, 2009	Anonymizing WLAN Traces	x	x	
Automatic profiling of network event sequences: algorithm and applications, Meng et al., 2008	Profiling of Network Event Sequences	x	x	
Confidentiality of event data in policy-based monitoring, Montanari and Campbell, 2012	Policy-Based Monitoring	x		
Distribution of inter-contact time: An analysis-based on social relationships, Wei et al., 2013	Distribution of Inter-Contact Time	x	x	
Coverage and Rate Analysis for Facilitating Machine-to-Machine Communication in LTE-A Networks Using Device-to-Device Communication, Swain, Thakur, and Chebiyyam, 2017	Machine-to-Machine Communication	x	x	
Balancing reliability and utilization in dynamic spectrum access, Cao and Zheng, 2012	Dynamic Spectrum Access	x	x	

An Online Algorithm for Task Offloading in Heterogeneous Mobile Clouds, Zhou et al., 2018	Offloading	x	x	
State-of-the-Art Routing Protocols for Delay Tolerant Networks, Feng and Chin, 2012	DTN	x		x

Table 1: Table of the properties of CRAWDAD dartmouth/campus data used in various research projects in which it was cited. Papers are ordered by the number of other papers they have been cited by, with the most cited at the top.

2.1.2 Summary of Results

The usage of the Dartmouth College CRAWDAD dataset is primarily regarding network mobility and social interaction/encounters. As such, the most often needed information seems to be identifiers for both mobile devices and access points, and the times of connections. Network mobility is a term used to refer to the connectivity and movement of users between multiple access points. Example usage of mobility research may be a Dartmouth College student wishing to stay connected to the internet while walking around their accommodation block and moving between areas which are covered by different access points, or a system may wish to offload a task but needs to be able to expect certain location-based conditions for it to be worthwhile. DTNs aim to eliminate the issues caused by intermittent connectivity, such as is found when moving between multiple access points. The Dartmouth College data is popular for this type of research since it reflect real-life usage of a large network. Also, the buildings from which data was collected include accommodation as well as academic buildings and so can provide insight into different types of usage. I found that the majority of the papers I looked at used the movement [15] or syslog [14] tracesets as these are most tailored towards mobility research.

There are also some less frequent topics of research such as software defined networking and delay tolerant networking using the dartmouth/campus dataset. These uses seem to require a wider variety of information from the data, however these instances are much less frequent than those mentioned above. these less common cases are the only ones which mention bandwidth and quality of connection.

2.2 Formats

2.2.1 Existing Formats for Aggregation of Network Traces in Mobility Research

A lack of published information on the intermediate formats used while analysing network traces for mobility research has been found during this survey. This is likely due to the encounter data not being the final outcome of the research taking place and therefore not being considered important enough to write up.

Through varied searches of DTN, Mobility, and SDN research I have found only two examples of well documented formats for storing data on device encounters. The first of these, The ONE Simulator [10], uses several reporting options to store device encounter data. The other documented format (from [28]) that was found was an association matrix. These two sources and and analysis of the information found in them is set out in the following section of this report.

2.2.2 The ONE Simulator [10]

The ONE is a simulator which generates data intended to mimic a network of mobile nodes. It then reports this data using various reporting modules, three of these modules focus on data regarding encounters between devices.

The first and most simple of these reports contains information about the dispersion of the total number of encounters experienced by the nodes in the network. It consists of two fields, one containing the number of encounters, and the other containing the count of nodes that have experienced that number of encounters. This contains no information relating to the unique nodes between which the encounters occur, or any temporal information such as duration of the encounters.

The second format provides information on the uniqueness of the encounters that were recorded, but loses detail about the total number of encounters in the dataset. This format also contains two data fields, one containing the values from 0 to 1000; representing promilles. The second field contains the number of unique pairs encountering with frequency withing the corresponding promille. This has a benefit of being almost static in size as the number of nodes int he system increases.

The final report format from the ONE simulation which has been looked at is a combination of the two previously discussed reports. It has three fields; the first contains an identifier for each node, the second contains the total number of encounters that the node has had, and the third contains the number of unique nodes with which it has had an encounter. This still does not uniquely identify both devices in an encounter, nor does it provide and detail about the duration of the encounters.

2.2.3 Association Matrices

Thakur et al. use an association matrix to record the percentage of time each node spends in an encounter with each other node. A matrix is created for each node, each column in the matrix corresponds to the other endpoint of the encounter, and each row corresponds to a time interval. The entry in each cell represents the percentage of the time interval spent in an encounter with the columns node. This format contains the most information out of all those discussed here, however also takes more space. The space taken will increase at with the square of the number of nodes.

2.2.4 Summary of Findings

A very brief summary of the detail contained within each of the formats discussed above is given in Table 2. Despite association matrices storing the most useful information, the polynomial increase in size with the number of mobile nodes makes using them potentially ineffective in the context of this project. The aim is to summarise a large amount of data into a smaller, easier to process format. In many cases association matrices would decrease the size of the data, but by a dramatically lesser amount than the other formats discussed here. Additionally it would be possible for the

	Level of Detail (Complete, Most, Some, or None)		
Format	Endpoints	Duration	Frequency
TotalEncountersReport - The ONE [10]	Some	None	Complete
UniqueEncountersReport - The ONE [10]	Some	None	Most
EncountersVsUniqueEncounters - The ONE [10]	Most	None	Complete
Association Matrix [28]	Complete	Some	Some

Table 2: Table of existing formats for storing data about device encounters and the level of detail they contain regarding the unique endpoints and length/frequency of the encounters.

association matrix format to increase the quantity of data; for instance if N nodes had $< N$ encounters each then the association matrix for each node would include at least one redundant column. The final reporting format discussed under the ONE simulation - EncountersVsUniqueEncounters - avoids this polynomial growth, with its size increasing only linearly with the number of mobile nodes in the network. It provides less complete information regarding the unique par of nodes between which the encounter occurred, there would however be no way to preserve this information while avoiding at least N^2 growth in size.

It seems that the most complete format of those discussed in which to store encounter data while also guaranteeing a reduction in the quantity of data stored would be the EncountersVsUniqueEncounters report format. This format could also be easily modified to add additional fields such as statistics regarding encounter duration. Any additional fields would need to be carefully considered and justified in order to keep the data quantity reduction as high as possible.

3 Requirements Specification

In order of priority the requirements of this project have been listed below. These are the same requirements as set out in the DOER document submitted at the beginning of this project, and have been described in greater detail here. Each of these requirements can be implemented separately from one another, and with the completion of all high priority requirements a working product will be completed. The high priority requirements combine to describe a minimum viable product to meet the overreaching aims of this project.
Implementation of the medium and low priority requirements will complete a more versatile system which could be applied to datasets with a wider variety of sizes and formats.
All of the requirements set out here are functional requirements, in that they describe features to be implemented in order to produce a high quality summarisation tool.

3.1 High Priority

- Reduce the quantity of data from the original dataset while maintaining any information identified as useful during research into how CRAWDAD datasets are used.
  - During the context survey it was found that the most used information from these datasets was the encounters between different devices. Mobile node encounters (including the nodes involved and duration) will need to be included in the summaries for this requirement to be met.
  - It is important that the total quantity of data is reduced in the summary compared to the original dataset. For this requirement to be met it should be shown that the output is guaranteed to contain less data than the input.
- Produce summaries of the initial datasets that can be processed more efficiently than the original data.
  - In order to meet this objective it is necessary for the summary format to be as simple as possible. Superfluous information and complex file types will need to be avoided.
- Use a commonly found format to output my summaries and justify why this format is appropriate in context.
  - In section 7 (Design) a complete output format will be specified and justification given based on the research in section 3.2.
  - Specification of the output summary format should include the file type that will be used, the fields that will be included, and the variable types that will be used. All of these decisions will need to be supported by relevant and reliable research.

3.2 Medium Priority

- Allow multiple summaries to be merged (this may allow extension into distributed processing).
  - This should allow two summaries which have been output by the minimum viable product to be given as arguments at run time and combined into a single summary.
  - The output of this will use the same format as the input summaries, as if created by running the basic program on the combined network traces.
- Summarise at least two different formats of input data to create a standard output summary.
  - The system will support summarising more than one input format, but the input format should be specified at run time.
  - Whichever format the input takes the same output format should be produced.
- Allow a summary to be updated by the addition of a single data entry (this may allow extension into real-time processing).

  - This requirement will be met if a summary can have added to it a single item of data (the exact definition of which will depend on the input format of the data, for instance a single data packet transfer in a tcpdump trace). With the resulting summary being identical to the summary which would have resulted from an initial input which included the new data entry.

3.3 Low Priority

- Process datasets with an unknown input format.
  - The format of the input data should not need to be specified at run time.
  - Multiple input formats should be accepted and and the system should be able to differentiate between them in order to process each correctly.
- Identify and report if a specific summary is likely to be unrepresentative of the input dataset due to aspects such as missing data or bias.
  - Depending on the information included in the summary, how representative it is may be effected for various reasons. Due to this it is important that any information used to determine whether a summary is unrepresentative is justified.

4 Software Engineering Process

An iterative approach was taken to the development process. Using an iterative methodology leaves less room for changing priorities and requirements than other more currently fashionable methodologies such as Scrum. However, in the context of this project work did not need to be coordinated between multiple developers, and requirements were not expected to change by any significant amount. Longer development cycles were appropriate due to the relatively small amount of time expected to spent per week on this project. For these reasons an iterative approach has been considered appropriate.

During this project several separate development cycles were completed, each with a different end goal. Each cycle began with a research and planning phase which then lead to a implementation and testing phase. Testing was done continuously during the implementation of features, with additional performance testing done at the end of each implementation phase once all new features had been added. Each week throughout the project a selection of tasks was made. At time flexibility was needed as some tasks were expected to take more than one week to complete. In such cases tasks were selected multiple weeks in a row.

As a final stage in each development cycle the product is evaluated against the requirements of this project. This gives a starting point for the next cycle, requirements which are fully met do not need to be implemented again in the next cycle, but still need to be given consideration to ensure future changes do not compromise them.

Throughout this project the school's Mercurial system has been used for version control. The Mercurial repository was use not only for the source code of the the summarisation tool but also to enable me to keep track of my own progress through weekly logging of the tasks which were completed and any issues that were encountered. By keeping track of past performance future weeks workloads were able to be more efficiently and realistically planned.S

4.1 Development Cycle One: MVP

The first development cycle was intended to be completed before the start of the second semester and to produce a fully working, well tested MVP which met all the high priority objectives set out in section 4 of this report.

4.1.1 Research and Planning

The first few weeks of this cycle were spent researching other work in the area, and completing the context survey as detailed in section three of this report. Then, using the findings from the research, the intended output format was specified. An outline for the stages needed during the processing of the data was decided and appropriate languages and libraries for each stage were decided on.

4.1.2 Implementation and Testing

Features that were implemented during this cycle were:

- The ability to take tcpdump output as input
- Extraction of encounters from input data into a simple CSV format
- A simple command line interface allowing and input directory to be specified
- A BASH script tying together the three stages of the MVP's data flow

Once these features were implemented through tests were done using the CRAWDAD dartmouth data. The output was considered to be reasonable by inspection of the number of access points detected by the summarisation tool. In addition the length of associations and encounters were checked to make sure they were realistic in the context of the building in which the data was collected. Complete accuracy was difficult to ensure since there is not existing encounter data to compare this output to.

After completing some testing of the time efficiency of the data processing showed that this initial implementation was unreasonably slow under certain conditions. Specifically when high numbers of access points were used, the matching of associations into encounters took a long time to run. Once five access points were detected the process would take several hours to run, while the same length of input with only two distinct access points would complete processing in around twenty minutes. Speeding this up is something considered in the next cycle.

4.1.3 Evaluation Against Requirements

The product developed by the end of this cycle satisfies the first two of the high priority requirements identified in section 4. The quantity of data is significantly less in the output of the product than the input. Research was done which identified encounters between devices as the most useful information in the data, so this information was kept in the output. The third high priority requirement was only partially met at this stage; the CSV format used for output has several benefits (such as its simplicity and flexibility) however no evidence has been found to support its use in the context to encounter traces in existing mobility research.

4.2 Development Cycle Two: Extending Capabilities

In this second cycle the aim was to find a more justifiable output format, to improve the time efficiency of the existing code, and to add features which extend the functionality of the product. The medium priority requirements in section 4 should be considered while adding functionality.

4.2.1 Research and Planning

The research in this cycle was mainly technical topics. A better understanding of the MatLab language was needed in order to write more efficient code, and so further research into the strengths and weaknesses of the language was undertaken. The ONE simulator was also further researched and how the output of this project could be altered to be compatible with it.

4.2.2 Implementation and Testing

Updated and improved functionality:

- The speed dependency of the processing on the number of access points was reduced from a quadratic relationship to a linear one

These additional features were added:

- Capability to parse syslog files in addition to tcpdump

By using appropriate data structure in MatLab the languages efficient referencing and searching was able to be used. This greatly reduced processing time when the intermediate associations data needs to be matched against known access points. It was this efficiency of MatLab that made it a justifiable choice for implementing the transformation from associations to encounters, however an inefficient method of looping through associations had mistakenly been used in the first implementation.

4.2.3 Evaluation Against Requirements

During this development cycle one additional medium priority requirement has been met (ability to summarise multiple known input formats). In addition to this the performance of the summarisation tool has been improved, and progress has been made toward using a more suitable output format which will be compatible with the ONE simulator.

5 Ethics

The potential ethical risk associated with this project is low since there will be no contact with participants (such as interviews or questionnaires), and no personally identifiable data is expected to be used. Datasets of wireless network activity downloaded from the CRAWDAD archive will be used in this project. The datasets which are to be summarised in this project are sanitised, meaning that details such as IP and MAC addresses have been changed to obscure the identities of the network users. Despite this there is a minor risk that identifying information may be accidentally extracted during the processing of this data. If any personally identifiable data is extracted it will be removed from any devices which it may have been stored on and the code which caused the mistake will be reviewed to prevent it from happening again. A full ethics application has been made for this project and has been given the approval code CS14642. The ethical application form has been appended to this document in Appendix A.

6 Design

6.1 Data Flow

6.1.1 Input and Processing

Multiple formats can be processed using the command line tool that has been developed during this project. Processing multiple input formats allows for Initially only TCPdump was considered, the tool took input in the form of binary PCAP files, this was then extended to allow syslog formats to be used as input. The output produced is in an identical format regardless of the input type. Data flow for the two input types is similar and detailed in figure 1.

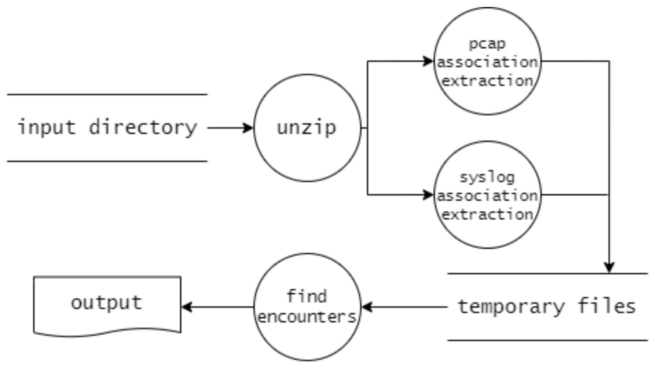
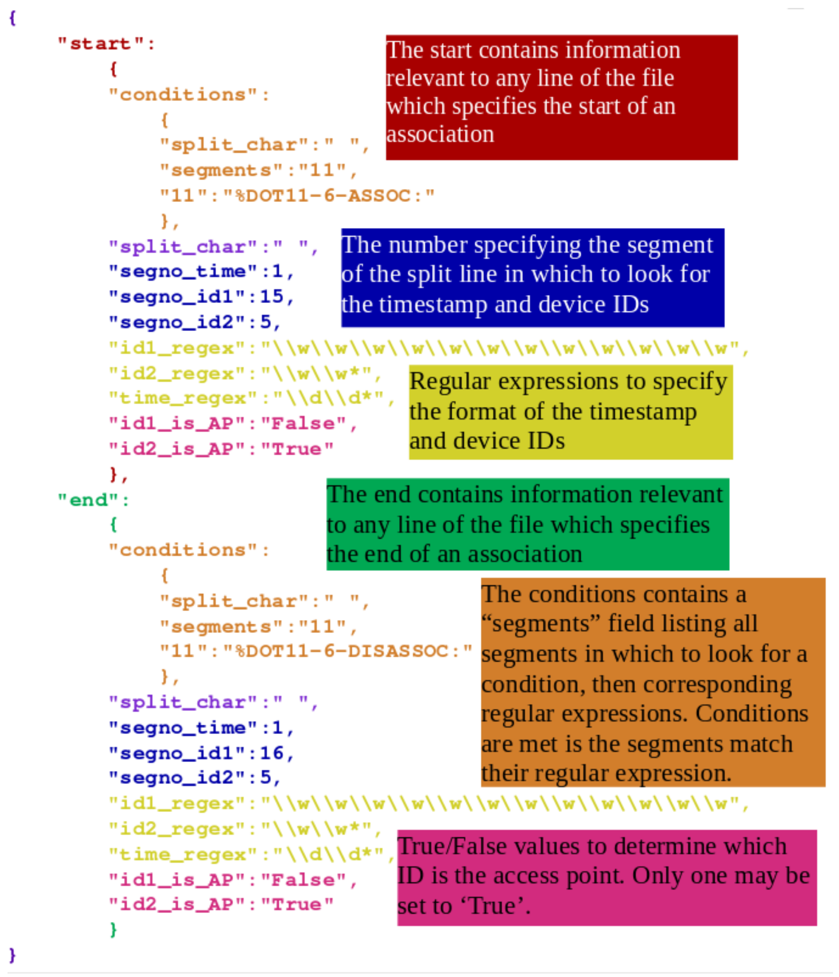


Figure 1: This figure shows the flow of data through the system, including temporary file stores and input/output to each processing stage.

The first stage of processing is to extract associations between devices and access points from the raw input. These associations will later be compared with each other to determine encounters between devices, these comparisons are expensive and so it is important that as much information as possible is removed before they are made. The intermediate format used to store association information is a comma separated value file with fields of source id, destination id, start time, end time, and AP flag. The AP flag is set to 1 only if the destination address of the association can be identified as an access point. Methods used to identify access points are discussed later in this report.

TCPdump output PCAP files are very strictly structured. This allows for them to be processed without any additional information from the user. However the initial parsing is expensive with respect to time since so much information is contained within them. In comparison to this, syslog files are relatively quick to parse for associations, but the user must include a configuration file to specify the format used. Details such as defining features of association end points and the format of device identification values need to be given to the tool before it can extract associations. The expected format of a configuration file for use with syslog is detailed in figure 2.



```
{
    "start":
        {
        "conditions":
            {
            "split_char":" ",
            "segments":"11",
            "11":"%DOT11-6-ASSOC:"
            },
        "split_char":" ",
        "segno_time":1,
        "segno_id1":15,
        "segno_id2":5,
        "id1_regex":"\\w\\w\\w\\w\\w\\w\\w\\w\\w\\w\\w\\w",
        "id2_regex":"\\w\\w*",
        "time_regex":"\\d\\d*",
        "id1_is_AP":"False",
        "id2_is_AP":"True"
        },
    "end":
        {
        "conditions":
            {
            "split_char":" ",
            "segments":"11",
            "11":"%DOT11-6-DISASSOC:"
            },
        "split_char":" ",
        "segno_time":1,
        "segno_id1":16,
        "segno_id2":5,
        "id1_regex":"\\w\\w\\w\\w\\w\\w\\w\\w\\w\\w\\w\\w",
        "id2_regex":"\\w\\w*",
        "time_regex":"\\d\\d*",
        "id1_is_AP":"False",
        "id2_is_AP":"True"
        }
}
```

The start contains information relevant to any line of the file which specifies the start of an association

The number specifying the segment of the split line in which to look for the timestamp and device IDs

Regular expressions to specify the format of the timestamp and device IDs

The end contains information relevant to any line of the file which specifies the end of an association

The conditions contains a "segments" field listing all segments in which to look for a condition, then corresponding regular expressions. Conditions are met is the segments match their regular expression.

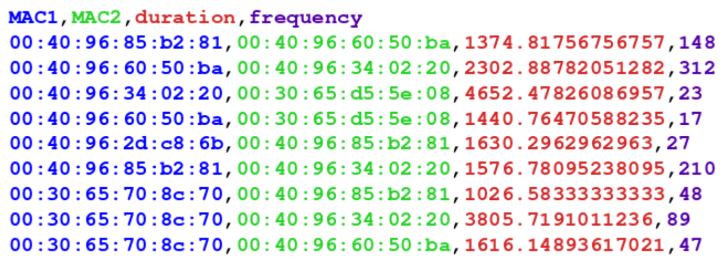
True/False values to determine which ID is the access point. Only one may be set to 'True'.

Figure 2: This figure shows a configuration file used for the processing of the dartmouth/campus Aruba syslog files. The fields have been colour coded for readability and descriptions of the fields have been given. The colour does not reflect any formatting of the actual configuration file.

Tying together the components of this data flow is a BASH script. Temporary files are used for storing associations in the intermediate stages, these files are then read by the MatLab script which compares associations to find encounters. A final output file is then given as a comma separated values file.

6.1.2 Summary Output

The final output specifies the endpoints of encounters, the average time of encounters between the given end points, and the number of encounters found between the given end points. Only one entry in the CSV file is present for each unordered pair of end points. A small section of an output summary is shown in figure 3, it has been colour coded for ease of reading.



```
MAC1,MAC2,duration,frequency
00:40:96:85:b2:81,00:40:96:60:50:ba,1374.81756756757,148
00:40:96:60:50:ba,00:40:96:34:02:20,2302.88782051282,312
00:40:96:34:02:20,00:30:65:d5:5e:08,4652.47826086957,23
00:40:96:60:50:ba,00:30:65:d5:5e:08,1440.76470588235,17
00:40:96:2d:c8:6b,00:40:96:85:b2:81,1630.2962962963,27
00:40:96:85:b2:81,00:40:96:34:02:20,1576.78095238095,210
00:30:65:70:8c:70,00:40:96:85:b2:81,1026.58333333333,48
00:30:65:70:8c:70,00:40:96:34:02:20,3805.7191011236,89
00:30:65:70:8c:70,00:40:96:60:50:ba,1616.14893617021,47
```

Figure 3: This figure shows a small section of an output file produced by the summarisation tool developed during this project. The columns have been colour coded for readability, but this does not reflect any formatting of the output CSV.

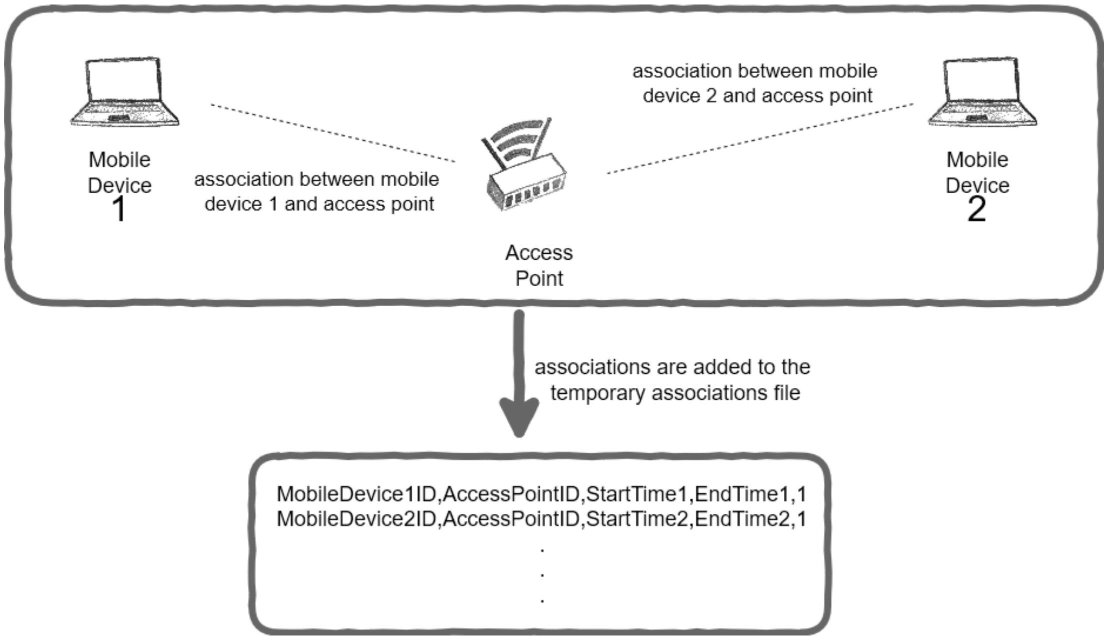
The data sets which I am using in this project have in the past frequently been used for research into mobility and encounter patterns between users [24] [30] [9] [21] [11] [16] [29]. To maximise the usefulness of this project it is intended that the output from this summarisation tool will be able to help in identifying whether a dataset is appropriate for use in mobility research, and to provide a standard format between multiple input formats which can be used for onward processing.

6.2 Associations

In this project two identifiable devices are considered to be associated with each other for any period of time during which they are exchanging data, a graphical representation of this is shown in figure 4. Although in this context only associations between one mobile node and one access point are relevant, a device might only be identified as an access point after several packets have been exchanged with it. It is therefore important that all associations are detected and recorded until a full list of access points is available.

6.2.1 Identifying Access Points

Access points need to be determined as only the associations with access points are needed in the later stage of this processing. Access points may be identified from tcpdump traces by reading the flags in the TCP header of the packets sent to them. If the SYN control flag of TCP packets is checked, when it is set the destination of the packet can be determined to be an access point. Once a packet is identified as an access point a flag is set by it in the intermediate temporary associations CSV file. This flag is used in later processing to remove all irrelevant associations from the data



Mobile Device 1
association between mobile device 1 and access point
Access Point
association between mobile device 2 and access point
Mobile Device 2
associations are added to the temporary associations file

```
MobileDevice1ID,AccessPointID,StartTime1,EndTime1,1
MobileDevice2ID,AccessPointID,StartTime2,EndTime2,1
                         .
                         .
                         .
```

Figure 4: This diagram shows a high level representation of two associations and the corresponding network connections which would cause them. If these two associations were active during the same period of time, an encounter would exist between the two mobile devices.

and to create a record of all detected access points in the network.

This method of detecting access points means that access points which do not exchange TCP data with any mobile nodes will not be detected, and therefore encounters which occur through them will not be recorded. It is unlikely for a network trace running over a significant period of time that an access point will associate with multiple devices and yet not transfer any TCP packets with them.

When parsing syslog files the location and format of the access points identifications should be specified in the configuration file. As all association and disassociation messages in a syslog file should contain an access point, all lines in the intermediate temporary associations CSV file created from a syslog input will be set as true. Despite this potentially seeming to be a waste of time to set all flags to true, it is necessary to maintain the intermediate format so that the next stage of processing can be completed using the same method regardless of initial input format. Some input files showed hundreds of access points being detected. This is likely due to each physical access point having many separate MAC addresses with which it will communicate. There is no way to identify which addresses correspond to the same physical access point without a record to match them against. Since such a record is not available for the CRAWDAD dartmouth/campus data (and likely wouldn't be immediately available for most network traces someone might want to use with this summarisation tool) the decision has been made to only consider an encounter to have taken place is both mobile devices are connected to the same access point MAC address. This is similar to saying that each MAC address being used by a physical access point has been considered as a separate device.

6.2.2 Initiation

A design decision needed to be made regarding the conditions that would specify the beginning of an association between two devices. This is an important part of the design of this system since it defines the meaning of an association, and therefore the meaning of an encounter in the final output of the system.

Identifying association request packets could be used as a method of identifying the beginning of associations. However it has been found during development that often two devices are in contact and transmit data between themselves without ever sending an association request packet. Since an association is open during the period of time which two devices are communicating for, a simplistic way of finding the start of an association may be the timestamp of the first packet transmitted between two identifiable devices.

My decision on this is that with a tcpdump input, the initiation of an association is triggered by a packet being transmitted between two devices which do not currently have an ongoing association. While an association is ongoing a transmission between the endpoints will not trigger a new association. If only one packet is transmitted between the endpoints within the timeout period it is not considered as an association. This is because it may have been a message sent to an unavailable destination, and in such a situation no contact is made between the source and destination devices.

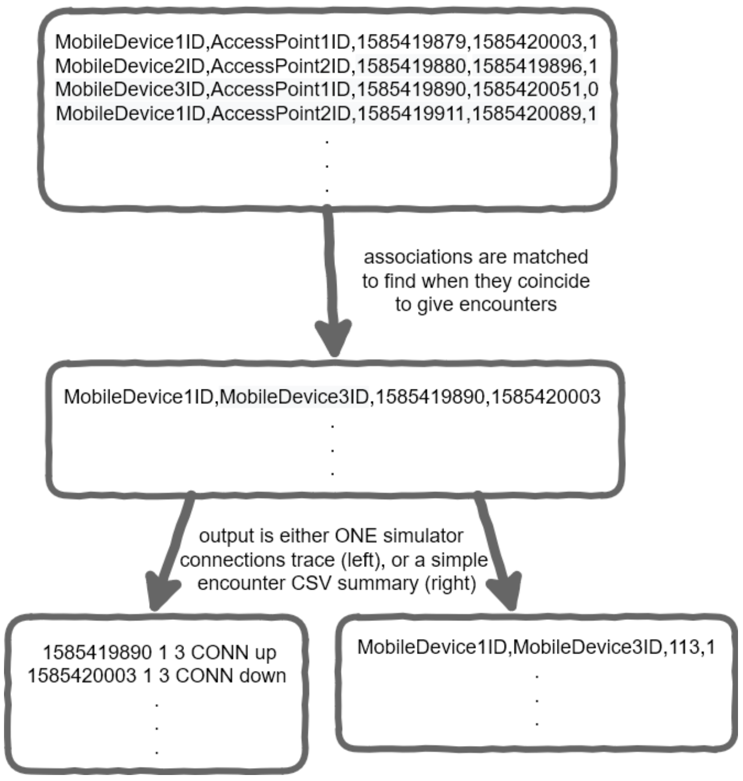
When a syslog file is used as input the configuration file should contain conditions for a message to define the start of an association. These condition will have to be checked against each line in the syslog file to find the beginning of each association. The endpoints of the association should also have their identifications location and format within the line described in the configuration file. In the case that a configuration file specifies conditions which are ambiguous (for instance causing multiple matches for a devices identification within the line) or too strict (for instance not finding any association start messages or not being able to identify any device identifications which match the conditions given int he configuration file) then the user should be warned so that they can update the configuration file if necessary.

6.2.3 End of Association

In some cases a specific message or packet type may be found which determines that a device has moved out of range of an access point. This would mean that the association between that device and access point has come to an end. However, in the pcap traces which I have been focusing on it seems common that the end of an association is not explicitly signalled in the trace. In these cases another method of detecting an appropriate time for ending an association is needed.

Research into work already done with the CRAWDAD data [12] [8] shows that a timeout of thirty minutes has previously been considered to be adequate for determining the end of a session, leading to a deauthentication of the device. This timeout length will therefore be use in this project to catch the end of associations when no explicit disassociation is found. When working with a tcpdump input, if no packets are transmitted between two associated devices for a thirty minute period the end of the association will be recorded using the timestamp of the most recent packet transmitted between them. When a syslog input is being used, conditions which identify a message showing the end of an association should be specified in the configuration file similarly to how they were specified for the beginning of an association.

6.3 Encounters



```
MobileDevice1ID,AccessPoint1ID,1585419879,1585420003,1
MobileDevice2ID,AccessPoint2ID,1585419880,1585419896,1
MobileDevice3ID,AccessPoint1ID,1585419890,1585420051,0
MobileDevice1ID,AccessPoint2ID,1585419911,1585420089,1
.
.
.
```

```
MobileDevice1ID,MobileDevice3ID,1585419890,1585420003
.
.
.
```

```
1585419890 1 3 CONN up
1585420003 1 3 CONN down
.
.
.
```

```
MobileDevice1ID,MobileDevice3ID,113,1
.
.
.
```

associations are matched to find when they coincide to give encounters
output is either ONE simulator connections trace (left), or a simple encounter CSV summary (right)

Figure 5: This diagram shows a example of how associations may be matched together to find encounters, and how the encounters may then be output in both available formats. Timestamps here are left as Unix epoch times since only a fragment of data is shown. In actuality times would be converted to an offset (in seconds) from the beginning of the first encounter found.

Encounter between devices are considered to begin at the earliest time where both devices are connected to the same access point, and to end as soon as one of the two devices disassociates from the access point. An example of how an encounter would be found from a set of associations is shown in figure 5. To find details of an association it is necessary to know which devices identification belong to access points, and the times at which other devices were associated with each access point. These associations will then need to be matched for overlap in the time periods they were active, and the access point that was involved.

It is likely that these matches will need to be found in a large set of associations, over a long time period, and many access points. It will therefore be important for the implementation of this to be efficient.

6.4 UI

6.4.1 Overview

The tool produced by this project is intended to be used by researchers in fields such as network mobility and DTNs. It is therefore appropriate to assume that a command line UI is sufficient, as users will be familiar with using terminal applications. Spending time improving the efficiency and functionality of the tool is more important than developing graphical interface which can be used without experience of using the command line. In addition to the unnecessary nature of a graphical interface, a command line tool gives users the ability to script and to pipe, this may be essential for researchers working with large quantities of data and complex data flows.

When large datasets are used with the tool it may be the case that a significant amount of time is spent processing the data. Due to this it is important that the UI keeps the user updated on the progress made. This should allow a user to know that the tool is working, and if possible to show them how far through the process the tool is. It should be expected that a large dataset may take several minutes (or even hours) to process, but to minimise user frustration this should be made transparent. By keeping the user informed of the amount of progress made in real-time the user is less likely to believe something has gone wrong and terminate the process while progress is in reality being made.

Ideally the tools UI should output a brief description of the stage of processing it is at, a progress bar which is updated in real time, and a small amount of additional information (such as directory names which are being used) so that the user knows a mistake hasn't been made. Hopefully this will allow users to avoid mistakes such as running a process for several minutes only to find the wrong input was used.

6.4.2 Runtime Options

The tool is run by using a BASH script 'summarise.sh'. When a user runs the script they have several optional arguments to use for setting various runtime parameters.

- -f <<format>> is used to specify the format of the input files. Currently only 'syslog' and 'tcpdump' are valid formats to use with this flag, and if the flag is not used then the tool assumes a default format of tcpdump pcap output.
- -i <<input_dir>> is used to specify an input directory. This is the directory which will be searched for appropriate input files for the tool, if the -i flag is not set then the current directory is used as a default.

- -o <<output_file>> is used to specify a path for the output file. If the flag is not used then the script will create a file in the current directory with name of format YYYY-MM-DD_hh-mm-ss_summary.csv (with the current date and time are used where appropriate).
- -c <<config_file>> is used to specify a configuration file for use during processing. If a configuration file is not required to process input of the specified format then the configuration file will be ignored, but a warning will be output to terminal. If no configuration file is specified when the format demands it, an error message is output to stderr and the script will exit immediatly.

7 Implementation

7.1 Overview

The data processing done by the summarisation tool is split into three stages: preparing the data for processing, extracting the associations between devices, and matching these association to find encounter between mobile devices. The data preparation stage simply consists of unzipping the input files and piping them to the next stage. The software component used for extraction of associations depends on the format of the input, however both of the implemented association extraction processes will output a temporary file of exactly the same format. This format can be read by the stage three process which finds the encounters between associations, it takes a temporary file as input and gives output as specified in the 'Design' section of this document.

Figure 6 shows how the components of the summarisation tool are connected. There are separate association extraction components for each input type. As long as the correct interfaces are used for input and output it is a simple matter to connect a new program to parse and extract associations from a different input format.

7.2 Stage One; Decompressing Files

The datasets downloaded from the CRAWDAD site are initially compressed (as gzip files). A directory of compressed pcap files can be given as a parameter to the summarisation script. Each file in the given directory will be individually decompressed using gzip and piped into the first stage of the processing. This unfortunately causes associations which span multiple pcap files to be split at the temporal boundaries of the files.

7.3 Stage Two; Extracting Associations

Libpcap is a library written in C/C++ which provides functions for the analysis of network traffic, including reading and extracting information from pcap files such as those produced in tcpdump traces. Tcpdump was the utility used to capture the CRAWDAD dartmouth dataset on which this project is initially focused. There are several wrappers of libpcap written for different languages such as pycap [23] for Python and jpcap [6]. These are generally not very regularly maintained and have very little documentation compared to Libpcap with C/C++. When initially reading in pcap files as input C++ will be used to extract the necessary information. In this stage of processing only information about the times of associations and MAC addresses of involved devices should be kept. Most of the processing will be done in subsequent stages, the

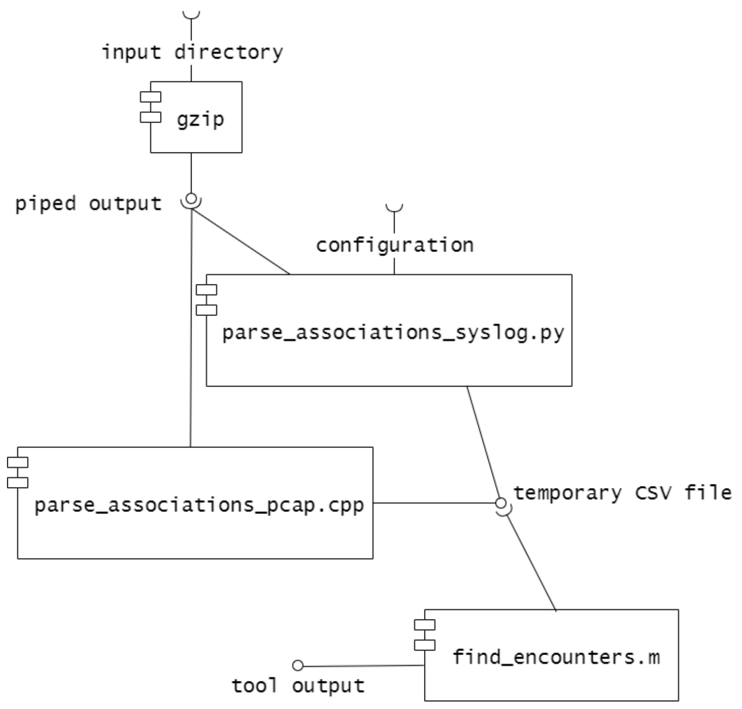


Figure 6: This figure shows the different software components involved in the summarisation tool, and how they interface with each other. This is a high level representation of the system and shows no detail regarding how each component and interface works.

main aim of this stage is to remove as much unnecessary information as possible, and to convert into an appropriate format for the next stage.

In this stage of processing a hash map of ongoing associations is built up during the reading of all packets from a pcap file. The map is updated every time a packet is read which signals the beginning of an association or the end of one. In the case of the beginning of an association, the map is updated by setting the MAC address pair to the two addresses between which the packet is sent (this is used as the key), and start and end times of the association are set to be the timestamp on the packet. Every time that a packet is read between a pair of devices the end time of the pairs current association is updated to the packets timestamp. Each packet is also checked to find whether the destination node of the pair is an access point. Since packet-specific information will be discarded after this stage it is necessary to identify access points with a flag. When the association ends, the values stored in the map are output in string delimited by commas. The values which are output are the source and destination MAC addresses, and start and end times of the association, and a flag (0 or 1) value identifying whether the destination device is an access point. This output is used as input in the subsequent processing stage.

7.4 Stage Three; Finding Encounters

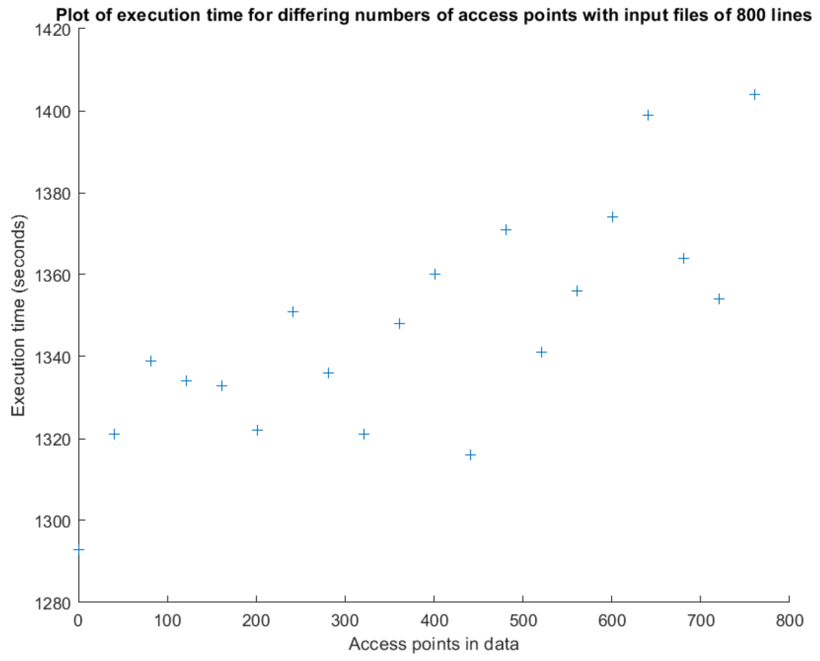


Figure 7: This figure shows the dependence of execution time on the number of access points in a file of associations for a file length of 800 lines. The files used as input to the MatLab code for this evaluation were fabricated to include specific numbers of distinct access point IDs. A roughly linear positive correlation is seen between the number of access points and the execution time.

The third stage of processing is written using MatLab. This decision was made due to the efficiency of the MatLab language when manipulating large tables of data (such as are used during this project) and my previous experience with the language in comparison to others similar such as Mathematica or Maple. Consideration was also given to continuing to develop this stage of processing using C++, similarly to the previous stage. This could potentially increase the efficiency and speed up processing time, however my familiarity with C++ is fairly limited and therefore the efficiency gained would likely not be worth the cost in development time by using C or C++.

This stage of processing has three main aims, firstly to extract a list of access points. Having a list of access points is important since encounters need to occur through access points (as described in the 'Design' section). The second aim is to match up the timings of associations between mobile nodes and access points such that encounters can be identified. The final aim is to then find the average duration and frequency of encounters between each distinct pair of mobile nodes.

Initially the implementation of this stage of processing became drastically slower

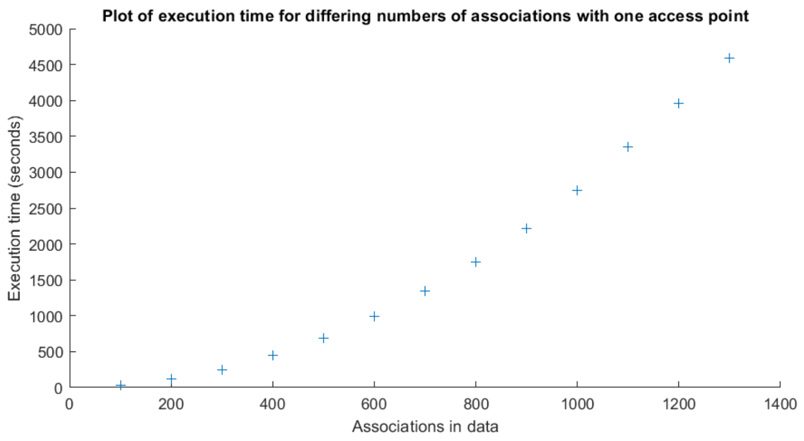


Figure 8: This figure shows the dependence of execution time on the number of associations in the intermediate file. Results are mean values averaged over ten iterations of each number of assciations. The files used as input to the MatLab code for this evaluation were fabricated to include specific numbers associations and a static number of access points. A power two positive correlation is seen between the number of associations and the execution time.

at high numbers of access points, this issue has been solved such that an almost linear behaviour in execution time if measured as the number of access points increases (shown in figure 7) whereas in figure 8 it can be seen that the execution time still increases with a quadratic curve as the line length of the intermediate associations file increases. The limiting factor is now the total length of the input file for this stage of processing.

8 Evaluation and Critical Appraisal

The high priority requirements of this project have been met, with the possible exception of using a commonly found file format to output the summaries. There are several pieces of evidence and justification for this. One of the requirements was to reduces the quantity of data through summarising it, while maintaining useful information. The information in the final summaries produced has been chosen based on research into past studies completed using the CRAWDAD datasets. Key data which was used in the studies such as the encounters between devices, device identifications for the endpoints, and encounter duration are all present in the summary output. It can also be shown quantitatively that the quantity of data has been significantly reduced from the input data to the summarisation, figure 9 shows how the quantity of data is reduced for each building in the dartmouth/campus data. On average this is a reduction in the number of bytes of over 99%.

This reduction in size also contributes somewhat to meeting the second requirement specified; the summary should be able to be processed more efficiently than the original data. Such a reduction in size denotes a reduction in the complexity and information content of the data. Assuming that the information required for further

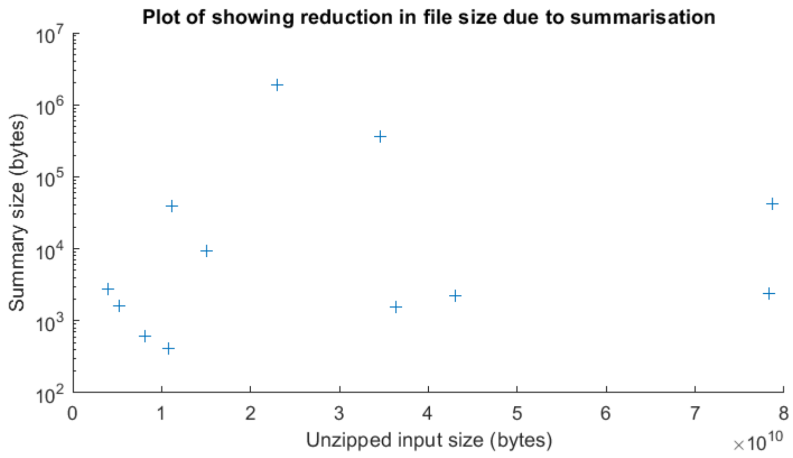


Figure 9: This figure shows the number of bytes in the unzipped input files compared to the bytes in the output file. This data was collected over the dartmouth/campus dataset where each input is the network trace of a specific campus building over a single measurement period. It can be seen that the number of bytes per files is reduced by between 3 and 8 orders of magnitude. This is a minimum reduction of 99.992%.

processing is still present in the summary, it will be much quicker to find in future processing tasks. In addition to this the output of the tool is a simple CSV format, meaning that many tools and software packages will be compatible with it. The final high priority requirement was that the output should use a common format which is appropriate to the context of this project. The use of specific data fields in the output are justified fully in section 3.2, the fields were chosen with consideration to past use of the dartmouth/campus datasets with the aim to include only information which was seen to be frequently relevant during previous research using the datasets. Although a CSV file could be argued to be a common format, it has little to no specific relevancy to the field of mobility research, requiring that specifically developed tools would need to be developed to further process the summaries produced by this project.

Summaries are able to be produced based on both syslog or tcpdump/pcap input files. The output produced using each input type is of the same format, however the format of the input needs to be specified at run time using an argument. This meets one of the medium priority requirements given earlier in this report and specified at the beginning of this project.

Unfortunately due to time restrictions no low priority requirements were able to be implemented. However it was kept in mind throughout development that adding further data formats to the input possibilities may become necessary (due to the low priority requirement of processing files of unknown input format), and this has influenced the structure of the system. The second stage of processing (extracting associations) is done within a separate software component for each different format, and adding a new component for a different input format would be trivial assuming it had been developed with the appropriate output format.

9 Conclusions

The summarisation tool discussed in this document provides a way to convert multiple formats of collected network data into a single output format. This includes tcpdump network traces and syslog files (as long as some details of the syslog format are given). If the input data is comprised of a large number of zipped files (such as pcap files segmented by time like those in the dartmouth Dartmouth dataset) the unzipping process can take a significant amount of time. However, this is unavoidable as the files cannot be processes while compressed. A large amount of input data can also lead to the third stage of processing taking a significant time to run, however this is determined by the number of associations and access points in the data and not by the total input size. These time dependencies have been taken into account during the development and have been minimised where possible.

It is also a concern that the output of the tool is not in a commonly used format and would require newly developed systems for onward processing. Future work on this tool would focus on formatting output so that it could be easily easily input to other systems such as the ONE simulator. The summarisation tool has also been specifically designed to be easy to build on in terms of new input formats; only a small part of the system would need to be re-written for a new input type to be added to the system.

References

[1] URL: https://scholar.google.com.

[2] *About CRAWDAD*. accessed: 20.09.2019. 2014. URL: http://crawdad.org/about.html.

[3] Ahmed Amokrane et al. “Flow-Based Management For Energy Efficient Campus Networks”. In: *IEEE Transactions on Network and Service Management* 12 (Dec. 2015), pp. 1–1. DOI: 10.1109/TNSM.2015.2501398.

[4] G. Baudic, Tanguy Pérennou, and Emmanuel Lochin. “Following the Right Path: Using Traces for the Study of DTNs”. In: *Computer Communications* 88 (May 2016). DOI: 10.1016/j.comcom.2016.05.006.

[5] Lili Cao and Haitao Zheng. “Balancing reliability and utilization in dynamic spectrum access”. In: *IEEE/ACM Transactions on Networking (TON)* 20.3 (2012), pp. 651–661.

[6] Patrick Charles. *Network Packet Capture Facility for Java*. accessed: 12.30.2019. June 2013. URL: https://sourceforge.net/projects/jpcap/.

[7] Zhenxin Feng and Kwan-Wu Chin. *State-of-the-Art Routing Protocols for Delay Tolerant Networks*. Oct. 2012. URL: http://arxiv.org/pdf/1210.0965.p.

[8] Tristan Henderson, David Kotz, and Ilya Abyzov. “The Changing Usage of a Mature Campus-Wide Wireless Network”. In: *Proceedings of the 10th Annual International Conference on Mobile Computing and Networking*. MobiCom ’04. Philadelphia, PA, USA: Association for Computing Machinery, 2004, pp. 187–201. ISBN: 1581138687. DOI: 10.1145/1023720.1023739. URL: https://doi.org/10.1145/1023720.1023739.

[9] Wei-jen Hsu and Ahmed Helmy. “On nodal encounter patterns in wireless LAN traces”. In: *IEEE Transactions on Mobile Computing* 9.11 (2010), pp. 1563–1577.

[10] Ari Keränen, Jörg Ott, and Teemu Kärkkäinen. “The ONE Simulator for DTN Protocol Evaluation”. In: *SIMUTools ’09: Proceedings of the 2nd International Conference on Simulation Tools and Techniques*. Rome, Italy: ICST, 2009. ISBN: 978-963-9799-45-5.

[11] S. Kosta, A. Mei, and J. Stefa. “Large-Scale Synthetic Social Mobile Networks with SWIM”. In: *IEEE Transactions on Mobile Computing* 13.1 (Jan. 2014), pp. 116–129. DOI: 10.1109/TMC.2012.229.

[12] David Kotz and Kobby Essien. “Analysis of a Campus-Wide Wireless Network”. In: *Proceedings of the 8th Annual International Conference on Mobile Computing and Networking*. MobiCom ’02. Atlanta, Georgia, USA: Association for Computing Machinery, 2002, pp. 107–118. ISBN: 158113486X. DOI: 10.1145/570645.570659. URL: https://doi.org/10.1145/570645.570659.

[13] David Kotz et al. *CRAWDAD dataset dartmouth/campus (v. 2009-09-09)*. Downloaded from `https://crawdad.org/dartmouth/campus/20090909`. Sept. 2009. DOI: `10.15783/C7F59T`.

[14] David Kotz et al. *CRAWDAD dataset dartmouth/campus (v. 2009-09-09)*. Downloaded from `https://crawdad.org/dartmouth/campus/20090909/syslog`. traceset: syslog. Sept. 2009. DOI: `10.15783/C7F59T`.

[15] David Kotz et al. *CRAWDAD dataset dartmouth/campus (v. 2009-09-09)*. Downloaded from `https://crawdad.org/dartmouth/campus/20090909/movement`. traceset: movement. Sept. 2009. DOI: `10.15783/C7F59T`.

[16] Udayan Kumar and Ahmed Helmy. "Human behavior and challenges of anonymizing WLAN traces". In: *GLOBECOM 2009-2009 IEEE Global Telecommunications Conference*. IEEE. 2009, pp. 1–6.

[17] Lewis M Mackenzie. *networkpages*. accessed: 12.30.2019. URL: `http://www.dcs.gla.ac.uk/~lewis/networkpages`.

[18] Xiaoqiao Meng et al. "Automatic profiling of network event sequences: algorithm and applications". In: *IEEE INFOCOM 2008-The 27th Conference on Computer Communications*. IEEE. 2008, pp. 266–270.

[19] Mirko Montanari and Roy H Campbell. "Confidentiality of event data in policy-based monitoring". In: *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012)*. IEEE. 2012, pp. 1–12.

[20] Mirco Musolesi and Cecilia Mascolo. "Car: context-aware adaptive routing for delay-tolerant mobile networks". In: *IEEE Transactions on Mobile Computing* 8.2 (2008), pp. 246–260.

[21] Mirco Musolesi and Cecilia Mascolo. "Mobility models for systems evaluation". In: *Middleware for network eccentric and mobile applications*. Springer, 2009, pp. 43–62.

[22] Anthony J Nicholson and Brian D Noble. "Breadcrumbs: forecasting mobile connectivity". In: *Proceedings of the 14th ACM international conference on Mobile computing and networking*. ACM. 2008, pp. 46–57.

[23] Pynetwork. *pynetwork/pypcap*. accessed: 12.30.2019. July 2019. URL: `https://github.com/pynetwork/pypcap`.

[24] Salvatore Scellato et al. "Nextplace: a spatio-temporal prediction framework for pervasive systems". In: *International Conference on Pervasive Computing*. Springer. 2011, pp. 152–169.

[25] Yan Shi, Shanzhi Chen, and Xiang Xu. "MAGA: A Mobility-Aware Computation Offloading Decision for Distributed Mobile Cloud Computing". In: *IEEE Internet of Things Journal* PP (Nov. 2017), pp. 1–1. DOI: `10.1109/JIOT.2017.2776252`.

[26] M. Sun et al. “Efficient Articulation Point Collaborative Exploration for Reliable Communications in Wireless Sensor Networks”. In: *IEEE Sensors Journal* 16.23 (Dec. 2016), pp. 8578–8588. DOI: `10.1109/JSEN.2016.2611594`.

[27] S. N. Swain, R. Thakur, and S. R. M. Chebiyyam. “Coverage and Rate Analysis for Facilitating Machine-to-Machine Communication in LTE-A Networks Using Device-to-Device Communication”. In: *IEEE Transactions on Mobile Computing* 16.11 (Nov. 2017), pp. 3014–3027. DOI: `10.1109/TMC.2017.2684162`.

[28] Gautam S Thakur et al. “Gauging human mobility characteristics and its impact on mobile routing performance”. In: *International Journal of Sensor Networks* 11.3 (2012), pp. 179–191.

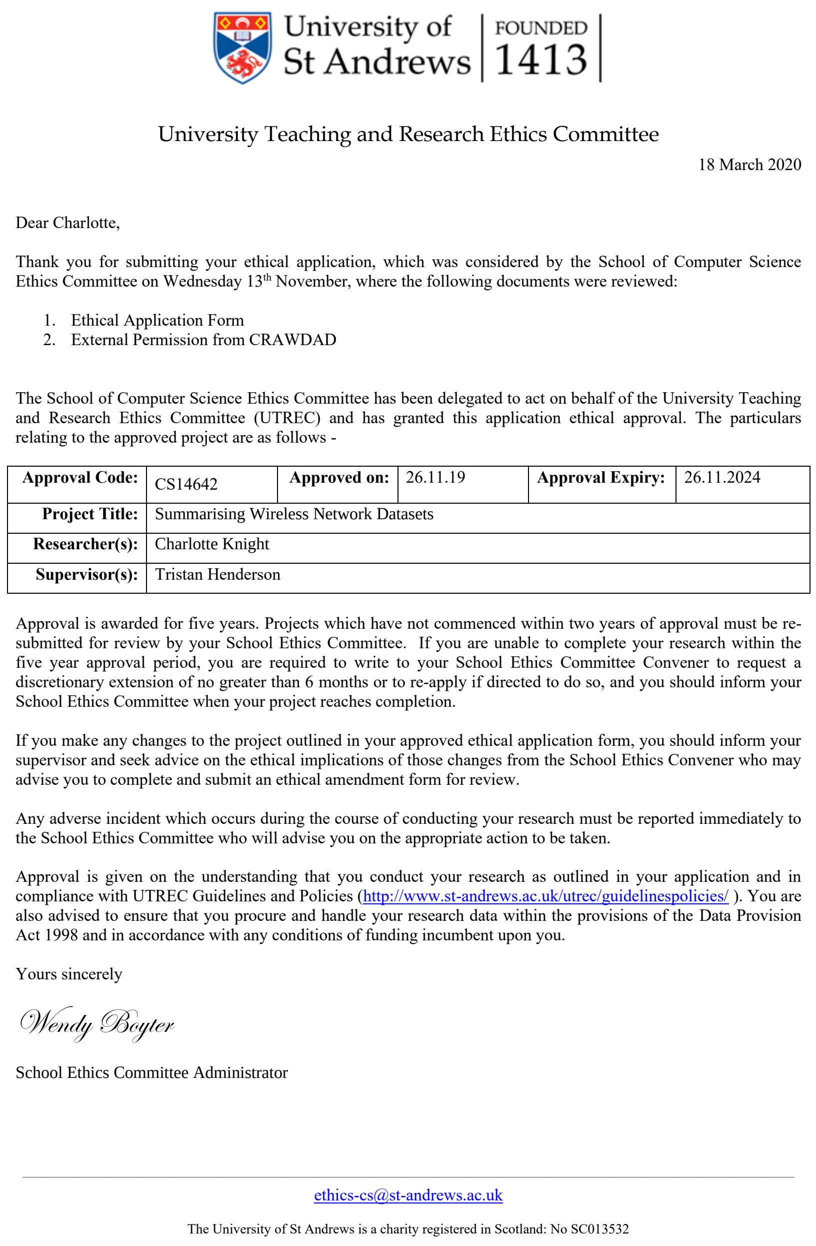
[29] K. Wei et al. “Distribution of inter-contact time: An analysis-based on social relationships”. In: *Journal of Communications and Networks* 15.5 (Oct. 2013), pp. 504–513. DOI: `10.1109/JCN.2013.000090`.

[30] M. Xiao, J. Wu, and L. Huang. “Community-Aware Opportunistic Routing in Mobile Social Networks”. In: *IEEE Transactions on Computers* 63.7 (July 2014), pp. 1682–1695. DOI: `10.1109/TC.2013.55`.

[31] Yu-Han Yang and K. J. Ray Liu. *WAVEFORM DESIGN AND NETWORK SELECTION IN WIDEBAND SMALL CELL NETWORKS*. June 2014. URL: `http://hdl.handle.net/1903/14834`.

[32] Bowen Zhou et al. “An Online Algorithm for Task Offloading in Heterogeneous Mobile Clouds”. In: *ACM Transactions on Internet Technology* 18 (Jan. 2018), pp. 1–25. DOI: `10.1145/3122981`.

A Appendix A: Ethics Approval



University Teaching and Research Ethics Committee

18 March 2020

Dear Charlotte,

Thank you for submitting your ethical application, which was considered by the School of Computer Science Ethics Committee on Wednesday 13th November, where the following documents were reviewed:

1. Ethical Application Form
2. External Permission from CRAWDAD

The School of Computer Science Ethics Committee has been delegated to act on behalf of the University Teaching and Research Ethics Committee (UTREC) and has granted this application ethical approval. The particulars relating to the approved project are as follows -

Approval Code:	CS14642	**Approved on:**	26.11.19	**Approval Expiry:**	26.11.2024
Project Title:	Summarising Wireless Network Datasets				
Researcher(s):	Charlotte Knight				
Supervisor(s):	Tristan Henderson				

Approval is awarded for five years. Projects which have not commenced within two years of approval must be re-submitted for review by your School Ethics Committee. If you are unable to complete your research within the five year approval period, you are required to write to your School Ethics Committee Convener to request a discretionary extension of no greater than 6 months or to re-apply if directed to do so, and you should inform your School Ethics Committee when your project reaches completion.

If you make any changes to the project outlined in your approved ethical application form, you should inform your supervisor and seek advice on the ethical implications of those changes from the School Ethics Convener who may advise you to complete and submit an ethical amendment form for review.

Any adverse incident which occurs during the course of conducting your research must be reported immediately to the School Ethics Committee who will advise you on the appropriate action to be taken.

Approval is given on the understanding that you conduct your research as outlined in your application and in compliance with UTREC Guidelines and Policies (http://www.st-andrews.ac.uk/utrec/guidelinespolicies/). You are also advised to ensure that you procure and handle your research data within the provisions of the Data Provision Act 1998 and in accordance with any conditions of funding incumbent upon you.

Yours sincerely

Wendy Boyter

School Ethics Committee Administrator

ethics-cs@st-andrews.ac.uk

The University of St Andrews is a charity registered in Scotland: No SC013532