What's the worst that could happen?

This document contains examples of specific risk which may affect operations and which may need to be considered when writing submissions. This is not an exhaustive list: any operation could involve new risks. It is also not a pick and mix list. It is here to help you think about the sorts of risk that might need to be included in a submission to ensure that the Secretary of State has all the relevant information available when deciding whether or not to approve the submission.

Are you the most appropriate person to assess the risk of your operation? Are you capable of looking at all the areas of risk (eg, reputational, technical, legal)? If not, ask someone who is (eg, an IPT expert, PTD, OPP-LEG respectively).

Risks to Personnel

- discovery/compromise of personnel involved in installation
- risks to personnel associated with housing operation if operation is compromised
- risk to collaborators / enabling agents
- adequacy of plausible cover leading to compromise of individuals
- Risk of false attribution and dire consequences (if there were a risk of reprisals against SIS agents or Embassy staff, then an assessment from the relevant SIS Controller or Ambassador might be appropriate)

Technical Risk

Appropriate technical colleagues are best placed to provide this information, eg PTD, NDIST, GTE, JTRIG.

- Compromise of technique leading to loss of capability
- Definite technical attribution leading to loss of capability
- Compromise of equity leads to loss of capability and discovery of other operations (eg by FIS)
- Novel capabilities have unknown effects outside of lab testing conditions

Political or Reputational Risk

If you think that any of the following risks are <u>significant</u> in your operation, you should consider whether or not you have adequate operational planning and mitigation in place.

- attribution to HMG
- attribution to UK
- attribution to GCHQ
- presumed attribution to UK (the target knows it's been the subject of an attack and assumes the UK is responsible)
- mistaken attribution (the target mistakenly blames a UK ally, who in turn attributes an effect to the UK)

- Political fallout with foreign governments or intelligence partners
- Media exposure
- Compromise of commercial partners

Humint

- *Talk to Humint partners if you think you have a significant Humint risk.*
- Risk of false attribution and dire consequences (if there were a risk of reprisals against SIS agents or Embassy staff, then an assessment from the relevant SIS Controller or Ambassador might be appropriate)
- Vulnerability of collaborators and enabling agents

Risks to Relationships

- Discovery or attribution could adversely impact on working relationships and/or sharing arrangements with sister agencies and/or second parties
- Discovery or attribution could adversely impact on working relationships with commercial suppliers and ultimately restrict GCHQ's sigint cability
- Potential to compromise a partner's operation
- See also Political or Reputational Risk section

Operational Phase

- See also Discovery
- Compromise of operation during installation, the course of the operation itself or egress of traffic
- Inadequate personnel security controls
- Operation does not succeed because the installed hardware/software does not function as planned
- Operation does not success because the installed hardware/software works, but is neutralized (eg because the target network/system is upgraded or replaced)
- Operation does not succeed because the target system is not used in the expected way (eg expected commercial usage does not occur)
- Operation does not succeed because of reliance on an uncertain supply chain or other risky dependence.
- Proportionality – the operation is not specific in its targeting
- Who will have direct access to the data resulting from the operation and do we have any control over this? Could anyone take action on it without our agreement, eg could we be enabling the US to conduct a detention op which we would not consider permissible?

Discovery

Discovery is a risk itself, which can lead to almost all of the other risks featured here. What follows is a list of circumstances which can lead to discovery.

- Compromise of operation during installation

- Inadequate personnel security controls and subsequent information leak
- Discover of installed hardware (including post-operation)
- Forensic discovery of installed software
- Discovery of a suspicious audit trail/logs/registry
- Discovery of suspicious RF energy
- Suspicious profile caused by hardware/software malfunction
- Discovery of egressed traffic
- Discovery through other IT leakage
- Vulnerability to HIS or other monitoring
- Inadequate monitoring of profile generated by operation
- Inadequate review of risks during the lifetime of the operation
- Reliance on an uncertain supply chain or other risky dependencies
- Failure by operators to cover tracks, including clearing logs/changing read status of emails
- Novel capabilities and techniques having unknown effects outside of lab testing conditions
- Unforeseen changes to hardware or software leading to compromise of techniques or installation
- Hardware/software malfunctions leading change in target behaviour, potentially including forensic investigation (and potential discovery) and/or loss of target access

Legality

Any risks relating to legality of operation or of subsequent actions enabled by the operation will usually be addressed by lawyers in legal section of submission, but may include the following issues:

- liability of enabling commercial partners
- the principle of non-intervention in a sovereign country's affairs
- Could the Law of Armed Conflict apply?
- Who will have direct access to the data resulting from the operation and do we have any control over this? Could anyone take action on it without our agreement, eg could we be enabling the US to conduct a detention op which we would not consider permissible?

Reference: OPC-M/TECH.A/455 (v1.0, r206)
Date: 20 September 2011
Copy no:

HIMR Data Mining Research Problem Book

[redacted]
OPC-MCR, GCHQ

Summary
In this problem book we set out areas for long-term data mining research at the Heilbronn Institute for Mathematical Research starting in October 2011 and continuing for at least three years. The four areas are beyond supervised learning, information flow in graphs, streaming exploratory data analysis and streaming expiring graphs.

Copy	Distribution
1	NSA R1 [redacted]
2	NSA R4 [redacted]
3	NSA R6 [redacted]
4	LLNL [redacted]
[redacted]	[redacted]
6	CSEC [redacted]
7	CRI [redacted]
8	DSD [redacted]
9	GCSB [redacted]
10	ICTR [redacted]
11	ICTR-CISA [redacted]
12	ICTR-DMR [redacted]
13	ICTR-MCA [redacted]
14	NDIST [redacted]
15	IACT [redacted]
16	PTD [redacted]
17	HIMR (circ.)
18	OPC-MCR (circ.)

OPC-M/TECH.A/455 (v1.0, r206)
[96 pages]

OPC-M/TECH.A/455 (v1.0, r206)

THIS PAGE IS INTENTIONALLY LEFT BLANK

OPC-M/TECH.A/455 (v1.0, r206)

HIMR Data Mining Research Problem Book

[redacted]
OPC-MCR, GCHQ

20 September 2011

Contents

OPC-M/TECH.A/455 (v1.0, r206)

1 Introduction

The Government Office for Science reviewed GCHQ technology research in 2010 and identified that we could lengthen our technology research horizon. The Heilbronn Institute for Mathematical Research (HIMR) had shown its mettle during a one-off graph mining workshop [I60, W42] and thus the idea to more permanently expand HIMR research beyond pure maths and into data mining was born. This also fits into GCHQ's overall research and innovation strategy for the next few years [I75], where engagement with academia via HIMR is a key plank.

Like many organisations, GCHQ is having to approach the "Big Data" problem. After reviewing our current research we identified four broad areas for long-term research in mathematics and algorithms at HIMR. All of the four problem areas are about improving our understanding of large datasets:

Beyond supervised learning: Can we use semi-supervised learning and related techniques to improve the use of machine learning techniques?

Information flow in graphs: Can we identify information flowing across a communications graph, typically from timing patterns alone?

Streaming exploratory data analysis: Can we develop new techniques for understanding and visualising streaming data?

Streaming expiring graphs: Can we efficiently maintain current situational awareness of a streaming expiring graph?

HIMR researchers are free to devote their effort amongst these problems as they see fit during their classified time.

These problems have been chosen due to their SIGINT relevance and SIGINT data is provided for all these problems. However we also recognise that these problems have overlaps with current academic research areas. Thus, conditional on security considerations, HIMR researchers should be able to generalise from classified research to unclassified research and publications during their unclassified time.

Data is made available to HIMR researchers in the following forms:

Streams: GCHQ are prototyping the use of the DISTILLERY streaming architecture (see Appendix B for details). Many data analysis problems can be efficiently approached in the stream [E39] and processing in the stream brings the advantages of live situational awareness and the potential to reduce follow-on storage and processing costs.

MapReduce: GCHQ store recent communications meta-data as distributed text files in Hadoop clusters which can then be processed with MapReduce [E10] (see Appendix C for details). This environment will allow researchers to use large datasets typically spanning the last six months of collection.

Reference: We also provide some smaller datasets (e.g. reference data or data that has already been processed or truthed) as text files.

The development of techniques in Hadoop or DISTILLERY is recommended as that will enable easy technology transfer from HIMR into GCHQ.

The HIMR Deputy Director, the authors of this problem book and members of GCHQ's Information and Communications Technology Research (ICTR) business unit should be seen as the primary points-of-contact for this research. However we will also identify various other areas for classified collaboration both in GCHQ and abroad.

GCHQ imagines that the most useful outcomes of this research will come in one of the following forms:

- Classified or unclassified research papers describing new techniques (or in limited cases a literature review of existing techniques).
- Classified research papers describing new or existing techniques applied to SIGINT data.
- New analytics (typically in Hadoop or DISTILLERY) and documentation.

In this problem book we adopt two conventions:

- We distinguish between references to internal literature, external literature and websites. Citations are prefixed "I", "E" and "W" respectively. Where possible literature is made available in DISCOVER (see appendix D). We have deliberately aimed to be more comprehensive in citing internal literature than external literature; external references should be easier to find from citation paths and review papers.
- We highlight problems with a ◀ in the right-hand margin.

In the interests of brevity, this problem book does not give full definitions for all terms in use in GCHQ and the use of GCWiki [W15] is a good place to find out more.

We would like to thank the many people across the 5-eyes community who have helped us with the problem book, both in formal contributions and in informal discussions at various conferences and visits over the last year. Within GCHQ we have had plenty of support from members of ICTR (in particular [redacted] and [redacted] and PTD (in particular [redacted]).

We start the problem book with an overview of relevant SIGINT background before describing the problems in detail. In appendices we suggest some ways of working, describe GCHQ's implementations of Hadoop and DISTILLERY and describe the datasets available.

2 A brief introduction to SIGINT

This is a very brief, high-level overview for people unfamiliar with the SIGINT system, focused on what data miners need to know about the data available to them and how data mining can be applied to problems in target discovery and cyber. Researchers are encouraged to find out more by browsing GCWiki and asking questions that arise.

SIGINT is intelligence derived from intercepted signals. Although this encompasses a huge variety of emanations, we are principally concerned with COMINT: intercepted communications.

Parliament's Joint Intelligence Committee (JIC) formulates a set of priorities and requirements for intelligence on various topics, which GCHQ tries to meet by producing End Product Reports (EPR) based on intercepted communications. GCHQ has the legal authority to intercept communications for the specific purposes of safeguarding the UK's national security and economic well-being, and to prevent and detect serious crime. GCHQ always acts in accordance with UK law. All researchers who have access to SIGINT data will be given legalities training, and there is also some information in appendix E on how data should be handled.

2.1 Passive SIGINT

This section looks at some of the main stages in the 'intelligence cycle': how data gets collected, processed and analysed to produce reports for GCHQ's customers.

2.1.1 Collection

There are many ways of communicating, and consequently there are many sources of SIGINT data. Traditionally, we collect signals using a variety of masts and dishes to pick up radio or satellite signals. Increasingly, we are interested in network communications (phone calls or internet traffic), and in this case to intercept the communication we usually need an access point in the network. (Sometimes network data passes over a satellite link where we can pick it up—COMSAT collection—but more often it doesn't.) Collection of this network communication data is called *Special Source collection*, the details of which are covered by ECIs. Access to raw data collected from Special Source is protected by a COI called CHORDAL. Some information about what the underlying sensitivities are, and the processes we have in place to protect them, is provided in the CHORDAL briefing.

One final twist is that a UK service provider can be compelled by a warrant signed by the Home Secretary or the Foreign Secretary to provide us with the communications data for a specific line or account for a specified time. This goes by several names: *Lawful Intercept* (LI), *warranted collection*, and *PRESTON*.

We refer to a single internet link as a *bearer*. We collect data from a bearer using a *probe*, and our current technology can collect from a 10G bearer (i.e. a 10 gigabit-per-second link). When a bearer is connected to a probe and associated processing equipment we describe the bearer as being *on cover*. We have been building up our sustained collection of 10G bearers since about 2008, and we now have approximately 200 bearers on sustained cover, spread across Cheltenham, Bude and LECKWITH. We refer to these three sites as *processing centres*; they are abbreviated to CPC, RPC-1 and OPC-1 respectively.

We have access to many more bearers than we can have on cover at any one time, and the set we have on cover is changed to meet operational needs.[1] As well as the fact that bearers get taken on and off, it is not unusual for technical problems to interrupt processing from a bearer, both for short and prolonged periods. This means that one must be careful about making assumptions about how traffic volumes from a given end-point vary over time: see [I10] for a detailed discussion of the problem and one way to deal with it.

2.1.2 Processing

A 10G bearer produces a phenomenal amount of data: far too much to store, or even to process in any complicated way. Our way of dealing with this is a multi-component system called MVR (*massive volume reduction*). To make things manageable, the first step is to discard the vast majority of the packets we see. This is accomplished by the *Packet Processing Framework* (PPF), a software framework allowing a very limited set of matching operations to be run on specialized hardware; packets that hit on these matches are then passed back to the software layer, where more complicated processing (including sessionization, done by a platform called TERRAIN) can be performed on the selected subset of the data.

Collected data falls into two categories: metadata and content. Roughly, metadata comes from the part of the signal needed to set up the communication, and content is everything else. For telephony, this is simple: the originating and destination phone numbers are the metadata, and the voice cut is the content. Internet communications are more complicated, and we lean on legal and policy interpretations that are not always intuitive. For example, in an HTTP request, the destination server name is metadata (because it, or rather its IP address, is needed to transmit the packet), whereas the path-name part of the destination URI is considered content, as it is included inside the packet payload (usually after the string `GET` or `POST`). For an email, the to, from, cc and bcc headers are metadata (all used to address the communication), but other headers (in particular, the subject line) are content; of course, the body of the email is also content.

There are extremely stringent legal and policy constraints on what we can do with content, but we are much freer in how we can store and use metadata. Moreover, there is obviously a much higher volume of content than metadata. For these reasons, metadata feeds will usually be unselected—we pull everything we see; on the other hand, we generally only process content that we have a good reason to target.[2] GCHQ's targeting database is called BROAD OAK, and it provides *selectors* that the front-end processing systems can look for to decide when to process content. Examples of selectors might be telephone numbers, email addresses or IP ranges. A selector whose communications are currently being targeted is said to be *on cover*.

Metadata generally gives us information that we think of as *events* ('*A* communicated with *B* at time *t*'), and this terminology filters through into the name for the corporate processing and storage system for 10G bearers: Next Generation Events (NGE).

[1] In order to make decisions about which bearers to place on cover, we carry out a *cyclic survey* of all bearers. Each bearer is connected to a probe for 15 minutes and data collected about the traffic seen during that time. This is stored in the *Flexible-survey Knowledge Base* or *FKB*.

[2] We do collect some unselected content for survey and research purposes, but the requirements that our activities be necessary and proportionate strictly limit what we can do with full-take content and who can have access to it: in particular, analysts are not usually allowed to write reports based on it.

Once packets (or files or sessions created by assembling multiple packets) have been selected and they emerge from MVR, they go to several different places.

- **Content databases.** Traditional relational databases are still the ultimate point of rest for corporate content data. There are also some legacy events database stores; soon, all of GCHQ's events processing and storage will move to the systems described in the next three bullets.
- **QFDs.** *Query-focused datasets* (QFDs) pick out data and store it in a way that makes it easy to answer particular questions. For example, FIVE ALIVE is a dataset with a record for each IP event seen, consisting of the 5-tuple (timestamp, source IP, source port, destination IP, destination port) plus some information on session length and size. This lets us answer questions about the network activity of a specific IP address.[3]
- **DISTILLERY.** A stream processing platform enabling near real time processing of data. See appendix B.
- **The cloud.** A scalable distributed filesystem along with a MapReduce processing framework. See appendix C.

It is important to emphasize that even after MVR, the data volumes in the QFDs, cloud and DISTILLERY are still vast, and we don't want to ship everything back to Cheltenham. Everything is distributed across the processing centres, with limited amounts of information being sent between them: queries to the stored data are all federated to the separate processing centres, with only the results being sent back to Cheltenham and the analyst's desktop.

2.1.3 Analysis, reporting and target development

Traditionally, an analyst would be given a particular target set to look at, and his or her aim would be to use the communications of these targets to write reports answering questions of interest to policymakers. For example, the target might be the Ruritanian Ministry of Foreign Affairs (MFA), and the aim to understand their posture in forthcoming negotiations with the UK; or it might be Kawastan's air force, and the aim to understand their general intentions and specific movements in a region where UK forces are currently deployed. The point is that the target set is generally well understood, and while looking at the contacts of a known senior figure in the MFA might reveal other government ministers or officials worth targeting, the problem essentially involves analysing communications carefully selected to be likely to bear on the questions under consideration.

Counter-terrorism, and to a lesser extent increased work on serious crime, has changed this landscape dramatically. The failure of the security services to prevent the 9/11 and 7/7 attacks has been widely dissected, both in the press and in government inquiries here and in the US. Targets are no longer neatly identified by their affiliation to a foreign MFA, military, or intelligence organization: finding the targets in the first place is now one of the most important problems facing analysts, before they can even begin to assess their plans and intentions.

[3] See appendix F.1.2 for more information on this QFD.

Contact chaining is the single most common method used for target discovery. Starting from a seed selector (perhaps obtained from HUMINT), by looking at the people whom the seed communicates with, and the people they in turn communicate with (the *2-out neighbourhood* from the seed), the analyst begins a painstaking process of assembling information about a terrorist cell or network.

But what about those who simply are not on our radar at all, like the 7/7 bombers? The main driver in target discovery has been to look for known *modus operandi* (MOs): if we have seen a group of targets behave in a deliberate and unusual way, we might want to look for other people doing the same thing. For this reason, a whole tranche of problems in this book looks at ways of picking out behaviour matching a specific MO in a large dataset. Specific MOs should be treated as particularly sensitive; knowledge of MOs can give SIGINT the edge over our targets who wish to remain undiscovered.

For example, sometimes targets will buy separate mobile phones and only use them to speak to each other, believing this to be good OpSec. In fact, this is unusual behavior that makes them stand out. Analysts call these *closed loops*; to a mathematician looking at a telephony graph, they are small components disconnected from the giant component that always forms in communications graphs. Another example is the use of payphones (commonly called telephone kiosks or TKs), which are an obvious way to communicate anonymously. Looking for calling patterns between TKs, or between a TK in the UK and a number in (let us say) Pakistan, has provided valuable intelligence leads.

Many of the problems in this book invite you to find new ways to use the data we have to discover things that analysts either could never find by themselves, or would never have the time to find in practice. It is important to point out that tolerance for false positives is very low: if an analyst is presented with three leads to look at, one of which is probably of interest, then they might have the time to follow that up. If they get a list of three hundred, five of which are probably of interest, then that is not much use to them.

Once we have targets, clustering or community detection algorithms give us a way to expand them into cells without laborious work by analysts. Doing this reliably and at scale is another fundamental challenge presented in this problem book.

It is also worth saying that techniques developed for counter-terrorism analysis can also feed back into traditional diplomatic and military analysis. For example, DynamicGraph (see section 6) is a way to visualize communication events around a seed set. Many of the applications have been to counter-terrorism operations, but it was first developed to look at the communications of foreign government officials visiting London for a G20 summit in 2009 [W18].

2.2 Computer network operations and the cyber mission

2.2.1 Cyber

Traditional diplomatic and military theories imagine nation states engaging in various physical domains: land, sea, air and space. The *cyber* domain is an increasingly important new site for interactions between states, and will only become more so as time goes on. The UK government has recognized the critical importance of cyber to our strategic position: in the Comprehensive Spending Review of 2010, it allocated a significant amount of new money to cyber, at a time when almost everything else was cut. Much of this investment will be entrusted to GCHQ, and

in return it is imperative for us to use that money for the UK's advantage.

Some of the problems in this book look at ways of leveraging GCHQ's passive SIGINT capabilities to give us a cyber edge, but researchers should always be on the look-out for opportunities to advance the cyber agenda.

This section briefly discusses how sophisticated state actors (including ourselves and our five-eyes partners) currently conduct themselves in cyberspace. It is important to bear in mind that other states, in particular Russia and China, are not bound by the same legal framework and ideas of necessity and proportionality that we impose on ourselves. Moreover, there are many other malicious actors in cyberspace, including criminals and hackers (sometimes motivated by ideology, sometimes just doing it for fun, and sometimes tied more or less closely to a nation state). We certainly cannot ignore these non-state actors.

2.2.2 Attack, exploit, defend, counter

There are four basic postures an actor can take in *computer network operations* (CNO).[4]

- **Attack.** This is obviously the most directly aggressive approach. It is commonly referred to as *computer network attack* (CNA); at GCHQ, one also hears it called *effects*. The actor accesses an adversary's network and deletes his files, destroys his network connectivity, or causes other damage or inconvenience. There has been a lot of discussion, both internally and externally, about the possibility of a cyber-based attack that could cause physical damage beyond the network, for example by shutting down a power station.

- **Exploit.** GCHQ's first CNE (*computer network exploitation*) operation was carried out in the early nineties, and since then CNE has grown to the scale of a small industry in GCHQ. A typical operation involves establishing a long-term covert presence (an *implant*) on a target computer, which sends back ('exfiltrates') useful information over an extended time period. You will know from press reports and public statements by the head of Security Service that UK networks and those of our allies—both government and commercial networks—are in turn routinely targeted by other countries.

- **Defend.** CESG is responsible for protecting UK networks (primarily government networks, but the security of banks or other companies operating in the UK is also important for economic well-being) from hostile CNA or CNE activity—the acronym for this is CND, or *computer network defence*. It is important to be able to prevent attacks by rejecting malicious packets at sensors or firewalls, and to understand who is attacking us (the attribution problem), why, and what they are looking for.

- **Counter.** This is a relatively new approach for GCHQ, which might better be called active defence. As we come to understand the CNE infrastructure of a hostile actor, we can target that infrastructure and attack it, disrupt its activities, or make use of the data that someone else has exfiltrated from a network that is also of intelligence interest to us (*fourth party collection*). This is sometimes called C-CNE (*counter-CNE*), not to be confused with CCNE, which was the name of PTD for a few years.

[4]This area is rich in jargon: see [W8] for a comprehensive list, along with links to further details on the subjects mentioned here.

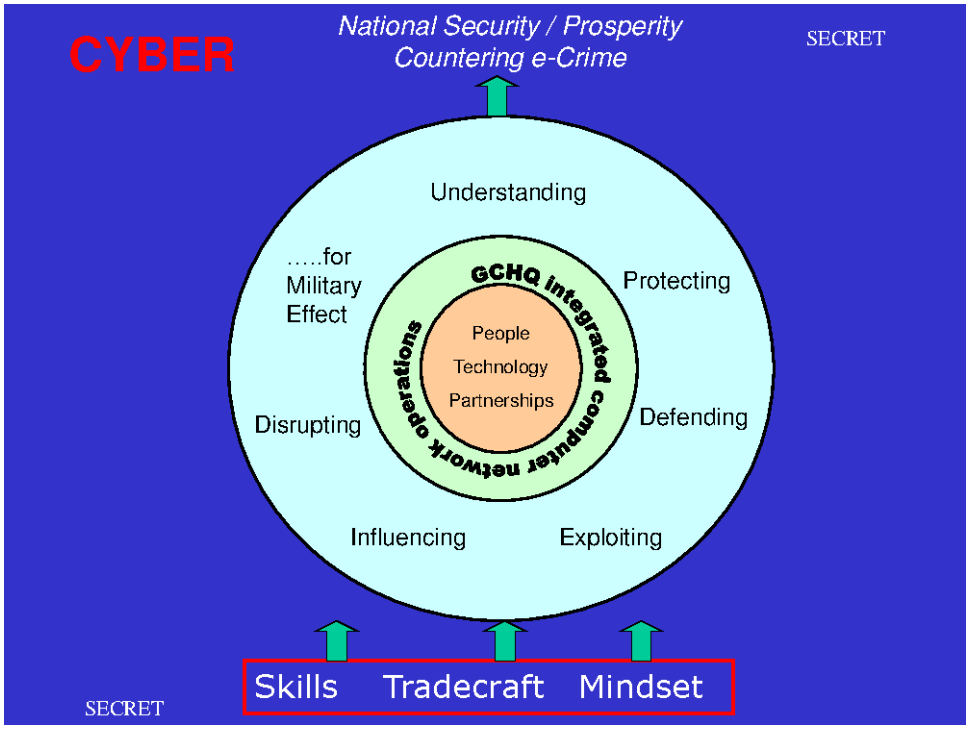


Figure 1: The Cyber Wheel.

2.2.3 Data mining for cyber discovery

CNE and CNA activity will leave traces in passive SIGINT. One of GCHQ's key contentions in discussions on spending for cyber has been that understanding the internet through SIGINT is the best foundation on which to build any cyber capability, whether offensive or defensive. This is the very first stage in the Cyber Wheel (figure 1), which is meant to be a visual representation of all the aspects of cyber, with SIGINT at the centre. NSA produced a simpler and earlier visualization [W43] of the same idea in 2007 (figure 2).

During the initial exploitation of a target box, malicious data needs to be delivered to the target. We (as well as commercial anti-virus and security companies) try to produce *signatures* for these *infection vectors*, which packets can be matched against.

Once machines have been implanted, they will usually perform certain characteristic activities on the network. Two major functions of an implant are *beaconing*, which involves sending short periodic messages back to the implant's controller confirming that the implant is alive and available for tasking; and *exfiltration*, i.e. pulling back data from the target box.

The fact that these activities are visible in passive SIGINT presents an OpSec risk to us [W7], but also an opportunity for data mining to discover hostile CNE activity. The core of a particular actor's infrastructure might be quite small, and discovering it can open up a whole chunk of their activity to be defended against or countered.

Botnets are large collections of implanted machines under the control of a single bot-herder. They are usually associated with organized criminals rather than intelligence agencies. Again

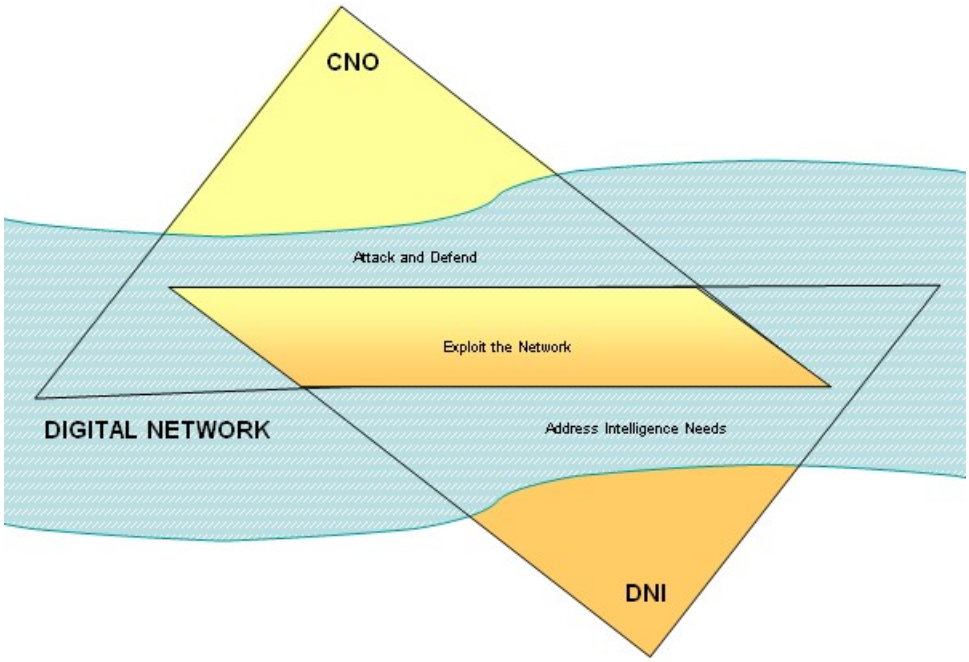


Figure 2: Another view of the relationship between CNO and Digital Network Intelligence (DNI), i.e. passive network SIGINT.

there are stereotyped behaviour patterns associated with botnets: *command and control* exchanges (also called C&C or C2), which are analogous to beaconing for implants; and *coordinated activity* in a short time window—for example many machines in the botnet simultaneously trying to access a website being targeted in a *distributed denial of service* (DDOS) attack.

Data mining offers the possibility of finding suspicious activity by detecting anomalies or outliers in bulk data. Temporal analysis and behavioural pattern-matching can be used to detect hostile network activity from CNE and botnets, but at present there is very little being done in this direction on our streaming data feeds. Several of the problem areas in the rest of this document touch on applications to these important cyber problems.

3 Beyond Supervised Learning

3.1 Introduction

Supervised learning is the machine learning task of inferring a function from training data. The training data is a set of training examples. Each example is a pair consisting of a feature vector and a desired output value (also called truthed data or labelled data). A training algorithm analyses this data and produces an inferred function, which is called a classifier (if the output is discrete) or a regression function (if the output is continuous). The inferred function should predict the correct output value for any valid input object. This requires the learning algorithm to generalise from the training data to unseen situations in a "reasonable" way.[5]

There are a vast number of supervised machine learning algorithms which can often produce functions with high accuracies on real-world data sets. However, these techniques have had surprisingly little impact in GCHQ. There are various reasons why this has been the case but the principal reason has been the difficulty in creating training sets. In particular, the difficulty comes from knowing the desired output value for many training examples, either due to the required human effort and/or uncertainty in the desired output value. This difficulty is unlikely to be a one-off issue for an operational application. The nature of communications and our data changes with time and leads to "concept drift"; any algorithm must be periodically retrained.

The aim of this research area is to improve the adoption of machine learning techniques. We suggest three ways forward on this area:

1. **Semi-supervised learning** alters the setup of supervised learning by only knowing the true value for a subset of training examples.

2. A special case of semi-supervised learning is **active learning**: in this case the training algorithm decides which examples it wants to be truthed. The aim is to make these the most informative examples rather than waste human effort on randomly chosen cases. This point-of-view also naturally works in a streaming context as a way of dealing with concept drift.

3. Allow ourselves to work with inaccurate truth data or **weak labels**. Such an approach would allow more automated labelling or reduce the human effort required.

We provide some small example datasets that have come from supervised learning problems. All examples in these datasets typically come with a label and a truth value. The scale of these datasets should not limit your imagination and larger untruthed datasets should often be obtainable either from the cloud or from a research area in GCHQ. If a very large number of unlabelled examples is found to be of value then streaming or MapReduce techniques will probably be needed.

It is important to note that the aim of this research is not necessarily to maximise the accuracy of prediction on these datasets. In the main, these datasets are fully-truthed and thus we expect that existing research on supervised learning will be competitive. Also these data sets are fixed and are thus not tracking customer interests or concept drift.

We also include the problem of **fusion of scores** that may be approachable by a natural extension of the weak labels research area.

[5] Paragraph adapted from [W41].

3.1.1 Supervised learning prior work

Supervised learning has had many applications in the research community but few applications have been deployed operationally. Some research examples over the last ten years (along with the classifier type used) are: steganography detection (Random Forest) [I74], website classification (decision tree) [I36], protocol classification (Random Forest and neural network) [W1], spam detection (Random Forest) [I44], payphone detection (Random Forest) [I3] and drug smuggler detection (logistic regression) [I77].

Random Forests

A common theme in many SIGINT applications is the use of Random Forest classifiers [E6][6]. Random Forests are an ensemble learning technique [W13]. The base learners are unpruned decision trees [W10] which then vote to reach decision. Randomness is inserted into each tree by two means. Firstly, each tree is built on a bootstrap sample of the training data. Secondly, the trees are built in a top-down manner by choosing the best feature at each node from a random subset of the features.

One reason for the use of Random Forests may be because they typically produce high accuracies with little tuning. However our feature spaces may also naturally lend themselves to Random Forests. Properties of our feature spaces include:

- Features are typically based on categorical and count data. Random Forests can handle a mixture of ordered and categorical feature types.
- Our data do not often show simple clusters. Some features (e.g. port numbers in the protocol classification example) behave a bit like ordered features and a bit like categorical features (nearby ports are sometimes associated but not always).
- Our features also show special values. A particular example could be a zero in a count could derive from missing data due to limited SIGINT visibility rather than saying anything relevant about the property of interest.

One adaptation to Random Forests considered in-house to improve accuracy and help understand the tuning of Random Forests is weighting of individual trees [I68].

Interpretability

A problem with the use of Random Forests is that their decisions can not be simply and intuitively explained to an analyst. This black box nature can lower analyst trust in a prediction. [redacted] (NSA R1) has been leading an effort to make Random Forests more interpretable [I18]. It would be good if semi-supervised models could have a broad-brush interpretability even if there are some complex exceptions that break these simple interpretations.

[6]The NSA were very early adopters of Random Forests through direct contact with [redacted] via the NSA Statistical Advisory Group (NSASAG) [W31]. The NSASAG remain a useful conduit to statisticians at US universities [W28].

Scale

The community has also considered scaling training of Random Forests to large datasets for the rare cases where one has computer-based truthing.

The most trivial scaling is naive parallelisation per tree. To do this one duplicates the data on multiple hosts, grows trees on each host independently and then combines the trees together into a final forest. The author of [I74] found this approach helpful in building classifiers for steganography detection.

NSA have looked at ways to implement Random Forests in Hadoop [I8, I4]. In GCHQ we have looked at streaming approaches with Random Decision Trees [I61] and Very Fast Decision Trees [I7].

3.1.2 Semi-supervised learning prior work

Semi-supervised learning is an area of active research in academia (see [E7] for a text-book reference and [E46, E36] for literature reviews). Given our interest with Random Forests, the recent paper on semi-supervised Random Forests may be of interest [E24].

However semi-supervised learning is less well developed in the intelligence community. LLNL have been considering active learning approaches for finding cyber attacks [I13]. Francois Theberge at CRI has looked at transductive learning (a special case of semi-supervised learning where a predictive function is not learnt at anywhere other than pre-chosen values) [I80]. The GCHQ maths summer student programme (SSP) in 2011 have been asked to look at transductive learning in the context of determining the relationship between entities [W32].

3.2 Semi-supervised learning

Semi-supervised learning algorithms "typically [use] a small amount of labeled data with a large amount of unlabeled data." [W38] This viewpoint is very desirable to GCHQ:

- Like many organisations we have large datasets of which only a tiny subset can be truthed by hand.
- We have more metadata than content. For truthing we may require content but policy or data volumes means that content is only available for a small fraction of the data covered by metadata. Therefore classifiers that run on metadata but are truthed based on limited (and not randomly selected) content are desirable.

Traditionally we have approached these problems with supervised learning and ignored all the unlabelled data.

The overarching question of this research area is can we use semi-supervised learning to our advantage? What shape must the problem have for there to be significant benefit?

3.2.1 How useful is semi-supervised learning?

There do not seem to be strong theoretical results in academia to explain the benefits of semi-supervised learning as opposed to supervised learning. Can we develop an applicable theoretical understanding of semi-supervised learning? ◄

Even if a theoretical understanding eludes us can we develop useful empirical rules of thumb on the value of semi-supervised learning? A simple starting point might be to measure gains in accuracy based on different approaches for the example SIGINT datasets. ◀

There are a range of known semi-supervised techniques; two major classes are: generative models and low-density separation [E7]. Can we tell what type of algorithm one might want to use on a particular dataset? ◀

What is the nature of a good feature space for semi-supervised learning? Are there feature transformations that could be applied to help this? ◀

If our truthing comes from an automated process then we may have untruthed examples that have failed automated classification. Alternatively if we truth a meta-data classifier based on content then our truthing will only exist where we have content. In both these examples, in contrast to the traditional viewpoint of semi-supervised learning, the truthed examples are likely not to be independently distributed of the features or classes. In the missing value imputation literature such truthing would be called "missing not at random" (MNAR). A potential approach to handle such truthing is described in [E33]. Can we build valid models when the truthing is not independent of the feature space or classes? ◀

In the above, we have assumed that each training example can be treated independently. Many SIGINT datasets have relationships between examples which can be represented as a graph. Progress is being made externally on graph-based semi-supervised learning (see [E17] and references therein) – can these external techniques be usefully applied to SIGINT problems? ◀

3.2.2 Positive-only learning

A special case of semi-supervised learning is when we only have labels for some members of one class and want to learn a binary classifier. An example is payphone classification where we have lists of some payphones and no labels for other phone numbers.

In the outside literature [redacted] [E13] presented a Bayesian approach to positive-only learning but internally [redacted] [I50] pointed out an error in their approach. However, in the world of statistical testing [redacted] has pointed out that one can still identify the most powerful test by considering the quasi-power [I49]. This approach was successfully used in a positive-only learning scenario for botnet detection [I71].

[redacted] asks, can we find or develop a theorem of the form: "a binary classifier can be trained if and only if ...". Can positive-only learning be shown to work with no other constraints? This type of theorem would also be relevant to the rest of the beyond supervised learning problem area. ◀

Can we design a new classifier for positive-only learning? ◀

3.2.3 Active learning

Many approaches to semi-supervised learning present a random subset of the data for truthing. This approach means that human effort is probably wasted classifying examples that have little impact on the learnt function. Active learning instead sets up the truthing process as a sequential process where the algorithm sequentially chooses examples for truthing based on all the information so far at its disposal. A useful review of external research in active learning is [E37].

The benefits of active learning are uncertain as an algorithm can focus on minor refinements to the current model and deliberately ignore examples that if truthed would lead to major changes to the current model. Do active learning algorithms quickly converge to a good model when allowed to choose which truthed items to use from the example datasets? ◀

A risk with active learning is that after many truthing examples one decides that the chosen algorithm is not suitable for the data set. It may not be practical to ask for more truthing with a different algorithm. What happens if you take the partially truthed dataset from one active learning run and use that dataset with a different semi-supervised learning algorithm? ◀

Active learning is a process where the algorithm and human are closely coupled and thus human factors are important. [redacted] suggests looking at active learning scenarios where the human is asked to rank two or three items rather then give a score or label. This may be easier from a human factors point of view. Can we design algorithms for active learning based on ranking pairs? How does the number of example pairs required compare to the number of truthed examples in traditional active learning? See [E35] for an example supervised approach. ◀

3.2.4 New algorithms and implementations

The asymptotic complexity of many semi-supervised learning algorithms is not good (e.g. $O(n^3)$, where n is the number of examples, or worse) [E46]. Such complexities are likely to be prohibitive on large datasets. Ideally we would like algorithms to run in $O(n \log n)$ or better.

We'd be interested in new accurate and fast semi-supervised learning techniques. The requirement to scale to large datasets will hopefully lead to streaming and/or MapReduce implementations. ◀

The SIGINT datasets provided may also inspire new techniques to enhance classification accuracies.

[redacted] (NSA R6) suggests that we may often be in the scenario that we have our truthed data as a small data set on which one can do a large amount of in-memory computation but our untruthed data as a large dataset in Hadoop. Can a learning algorithm be developed that iterates between complex in-memory analysis of the truthed data and single table scans of the untruthed data? ◀

3.3 Unreliable marking of data

An alternative approach to improve the applicability of machine learning techniques is to allow inaccurate truthing of data, so called "weak labels". We think of this case as related to semi-supervised learning; in traditional semi-supervised learning you have perfect knowledge of some cases and no knowledge of other cases – in the case of weak labels this knowledge is diffused across the entire dataset.

3.3.1 Weak labels

Scenarios where weak labels could occur are:

Semi-synthetic data: If suitable training data can not be found we may want to modify data to have the properties we want. An example of this is steganography detection [I74]. We take a large number of images from SIGINT and add steganography into some the images to make our truthed data. Errors will occur as some of these images may have steganography before we start.

Automated labelling: We might base our labels on content based-signatures that may not be accurate (e.g. for protocol classification [I70]).

Natural error: Even experts make mistakes when labelling.

Externally the field of weak labels has been rejuvenated by the use of the internet for truthing by amateurs, e.g. using Amazon's Mechanical Turk where one may have multiple labels per item [E31]. However, the field dates back many years; for example [E27] showed the impact of weak labels on nearest-neighbour classifiers and L_1-consistent estimators.

Another recent approach has been MIForests [E23] which shows an approach to adapt Random Forests to binary classifiers based on sets of inaccurately marked data.

As mentioned in section 3.2.2 by looking at quasi-power [I49] we can work directly with weakly labelled items (with some constraints on the labelling) to identify a most powerful test.

Can we understand the influence of labelling errors on different techniques? Do some traditional supervised learning techniques work out-of-the-box with weak labels? ◀

Can we develop algorithms that understand and compensate for the errors? ◀

3.3.2 Fusion of scores

A problem which might be a natural extension of this work is fusion of scores. For example, we have multiple techniques to try to infer a relationship between entities (e.g. from contacts, timing behaviour and geo behaviour). These techniques produce scores that are typically real numbers between 0 (no relationship) and 1 (a relationship exists). If these were (proportional to) independent likelihoods then these scores could simply be multiplied. However, these scores will not be independent and will not be likelihoods. How can we combine such scores in general? ◀ Can we combine such scores to posterior probabilities? How large a deviation from independent likelihoods can we cope with?

This problem is exactly the problem of weak labels if we treat one score as being a weak label and the rest of the scores as features. We have the added power that we can choose any feature as the weak label.

Internally we have considered score fusion in two main contexts:

Relationship scoring: CHART BREAKER [I31] research initially looked at handling the multiple scores derived from the email communication hypergraph but is currently being extended to handle multiple communication mediums as part of FIRST CONTACT.

Geo-reference data: We have multiple sources of data giving us information on the geolocation of an IP address. The GeoFusion project [I53] and RADONSHARPEN-B [I59] have looked at combining country labels and confidences from multiple sources to come up with a decision for an IP address's country.

Dataset	Ref	# truthed	#features	output	noisy	+ve only
Logo recognition	F.4.1	530	64	109 classes	N	N
Spam detection	F.4.2	1809	143	11 classes	N	N
Protocol classification	F.4.3	799,541	51	15 or 39 classes	N	N
Steganography detection	F.4.4	1,550,000	661	$(0, 1)$ range	Y	N
Genre classification	F.4.5	∼16,000	108	2-17 classes	N	N
Website classification	F.4.6	6,705	200	4 classes	N	N
Payphone detection	F.3.5	97,993	N/A	Binary	N	Y
Arrival process correlation	F.1.5	763,392	N/A	Binary	Y	N

Table 1: Truthed data sets. Further details about these datasets can be found in appendix F as referenced in the second column.

If we're dealing with labels rather than scores then there's a line of literature in medical statistics looking at estimating the accuracy of diagnostic tests. These are based on the Hui-Walter method of independent tests [E19, E32, E21]. Extensions have now looked at correlated tests [E11]. [E38] makes the link between these approaches and latent class models and thus this problem can be seen to be related to that being considered by [redacted] at LLNL for learning with network data [I58]. [E31] shows an extension to real valued functions.

NSA have also looked at this problem in the context of log-likelihoods that may not be independent [I45] (their approach has been reviewed by GCHQ [I26]).

3.4 Relevant data

3.4.1 Truthed datasets

We provide various SIGINT truthed datasets as summarised in table 1. Most of these data sets consist of features and truthed output for all examples. There are a few exceptions:

- The protocol classification set has some "NULL" labels for which automated signature-based classification failed. This dataset can be seen as an example of a semi-supervised set where the truthed examples are not randomly chosen.
- The payphone data set comes with no features. We do not have feature extraction in Hadoop. Implementation of the features in [I3] should not be too large a task and implementing a complete system would aid deployment.
- The arrival process correlation data set has no features extracted. Also the truthing comes as two sets where one set is richer in true cases than the other. This data is included as it is an active area of statistical research and overlaps with the information flow in graphs problem. If features are required for this set then the CLASP scores [I49] could be useful features but new approaches would also be welcome.

For the fully-truthed data sets in table 1 it is imagined that semi-supervised or weak label experiments can be conducted by hiding truth labels or perturbing truth labels.

The steganography detection dataset may also be a good dataset if we want to look at cost-sensitive feature extraction within the context of semi-supervised learning. The features are computed in classes, each with a cost. We can give some metrics on these costs if required.

It should be remembered that the aim of this problem area is not to maximise the accuracy of classification on these datasets. These datasets should be used to support improvements in understanding and algorithms.

A flaw with these datasets is that they are mostly small (having derived from experiments with supervised learning). The ability of algorithms to scale to larger sizes should be considered. The payphone data may be the most promising one to look at at scale.

3.4.2 Fusion of scores data

We provide fusion of scores data from GeoFusion. Scores in GeoFusion are typically ordered confidence labels ("low" to "very high") rather than real numbers. We provide the country and confidence from four SIGINT systems as well as the Akamai Edgescape commercial geolocation dataset. See appendix F.5 for more details on this data.

We hope that data for fusion of identifier relationship scores will be available soon. Alternatively researchers could use existing software to compute scores from telephony or C2C data on the cloud themselves – please consult the authors for more guidance on this route if required.

3.5 Collaboration points

There are several areas where one might find useful collaboration in this problem area:

ICTR-MCA: The Media Content Analysis team are looking to automatically determine the relationship between entities based on communication content and think that semi-supervised techniques are likely to be needed; [redacted] is leading on this work. [redacted] and [redacted] also think that their work on speaker identification may lead to a semi-supervised problems with weak labels. [redacted]i is also planning to revisit the problem of finding IED triggers in audio content and which may lead to a dataset with features derived from roughly continuous data (as opposed to many of the provided sets being based on discrete data), see [W44] for more details.

ICTR-DMR: [redacted] is leading a major research package on fusion of scores. [redacted]t would also be interested in any developments based on the payphone detection dataset.

US National Labs: At [redacted], [redacted] and [redacted] are working on active learning for finding anomalies in C2C data. [redacted] at LLNL and [redacted] team at Sandia National Labs have been working on large-scale machine learning algorithms. [redacted] at LLNL was interested in latent class models (potentially linked to fusion of scores) but has now been posted to Australia.

NSA R6: [redacted] is interested in large scale semi-supervised learning algorithms.

NSA R1: [redacted] is always interested in anything that can advance the arrival process correlation score. Also [redacted] is interested in techniques that may improve the interpretability of Random Forests (he particularly mentions the "Treebeard" technique [I18] as having further research possibilities).

CRI: [redacted] is working on transductive learning which is closely related to semi-supervised learning.

IBM Research: [redacted] suggests that unclassified engagement may be possible with [redacted] (IBM Research) on active learning. [redacted] also works with [redacted] from Yahoo Research who has also been working on fast online learning algorithms, exemplified by Vowpal Wabbit.

4 Information Flow in Graphs

4.1 Introduction

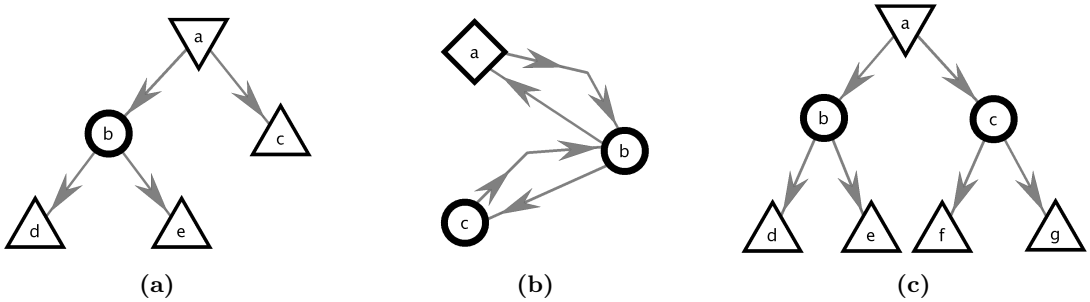
This section of the problem book concerns *information flow in graphs*. By this we mean the study and discovery of related information or messages being relayed over multiple edges in a communications graph. For this problem we will initially consider working on a static graph, although you should feel free to consider the streaming case if you desire. We get to observe a set of transactions taking place on the edges of the graph. Given these transactions we would like to be able to infer something about likely information flows across multiple edges. In most cases we will know nothing of the content of the transactions. We therefore wish to focus mainly on techniques which do not require any content knowledge. Data with content should therefore mainly be seen as truthed data for exploration and familiarisation with existing techniques.

We will now provide two motivating examples for our interest in information flow in graphs. These are chosen to reflect intelligence interests over the last decade or so.

The first example is a target-centric communications network. Consider the graph formed by telephone calls around a certain target set. Each call, or transaction, serves the purpose of conveying information between participants. If significant flows could be extracted then this would provide information on the structure of the target set—perhaps identifying commanders, middlemen and operatives. Now, if one of the commanders was no longer part of the network we could again examine how the flows have changed and therefore gain insight on any reorganisation that has taken place. Further, it may even be possible to identify a significant change in flows on the graph and identify a change in structure purely from transactional data.

The second example is the detection of botnet command and control infrastructure. For a botnet to be effective it needs to be able to convey commands from its controller to all infected nodes. One can imagine that with some knowledge of infected nodes it may be possible to use information flows to trace out the infrastructure, discover further infections, or even track back to find the botnet's owner. This type of capability would be of enormous interest due to the current emphasis on cyber defence.

We now define a *cascade* and discuss how we will use them to summarise the significant information flows.



(a) **(b)** **(c)**

Figure 3: Examples of cascades. A downwards pointing triangle is a source and an upwards one a sink. A diamond means the node is both the source and a sink.

Definition 1. A *cascade* is defined to be a directed connected subgraph with a single source and one or more sinks, consisting only of directed paths from the source to every sink. Being a source or sink is considered an attribute rather than necessarily having no in or out edges respectively.

Figure 3 shows some example cascades. Subfigure 3(b) demonstrates a cascade where the source is also a sink—there is nothing in the definition which prohibits this. Further, we can imagine situations with an information flow like this. For instance consider a friend to ask their friend for a favour and then having the response relayed back.

Cascades can be used to represent significant, repeated information flows on the graph. Each directed edge in the cascade, starting from the source, should occur no earlier in time than its predecessors. Algorithms developed should probably output such representative cascades.

We are interested in techniques which do not depend upon having the content of transactions as this limits their applicability. This is because much metadata is of the form "A communicated with B at time t", with few or no clues to what the content of that communication was. Because our data is in this form we place a particular emphasis on temporal correlation when surveying past work.

This section of the problem book has a relatively small number of wide problems. This is because the main problem of information flow definition and discovery is meant to be open ended with plenty of scope for exploration and experimentation[7].

4.2 Past work

We will now describe past work in related areas of research, with a particular emphasis on internal research. We will introduce key areas of work, give a sketch of their workings and provide references for further reading. External work discussed should be seen as a sample rather than a definitive list.

This subsection will first discuss methods on graphs, starting with explicitly temporal ones and then moving on to static ones. We will then discuss the extensive research that has been conducted on temporal correlation of stochastic processes. We expect that research on information flow in graphs may want to draw on all areas, perhaps applying our knowledge on temporal correlation in a graphical setting.

4.2.1 Graphical methods

There have been several approaches used to exploit timing information present in transactions on graphs. If two vertices participate in timing patterns then it is likely that they are closely related. Further if one of these vertices is a target then the other may be worth investigating more closely.

The first temporal graph algorithm in GCHQ was *Remit*, developed under contract by Detica for ICTR-DMR [I78]. A large amount of subsequent research can be seen to have been directly triggered by the *Chains* analytic within Remit. Chains is about the simplest approach possible to finding information flow in graphs. One simply defines a maximum time allowed

[7]The problem "Find and score related stochastic processes" has already had many man-years of research effort expended across dozens of approaches.

between transactions on adjacent edges and a minimum flow length. The Chains algorithm will then find all flows satisfying these conditions. Despite this simplicity trials showed that it could produce useful intelligence when applied to a target-centric telephony graph [I41]. Unsurprisingly, given its fixed time window, Chains had issues with flows spuriously going through vertices with a high activity rate. This motivated the next stage of research in temporal graph methods.

PRIME TIME [I42] was the next approach created. PRIME TIME introduced a statistical model to compensate for varying vertex activity levels. Specifically an exponential distribution is fitted to a vertex based upon its mean time between transactions. This exponential is used to calculate a p-value on waiting times for transactions on pairs of adjacent edges. If the p-value is less than some critical value then the transactions will be considered related. Furthermore the p-values are collected for future scoring of long and/or repeated flows. However the methods of combination used are ad-hoc and not statistically motivated. The original PRIME TIME paper talks of chains of related edges, although in practice only length 2 were computed. Even so this suggests the beginning of the study of information flow in graphs.

Currently a streaming version of PRIME TIME is being developed by Detica for the Streaming Analysis team in ICTR [I63].

HIDDEN OTTER is an ICTR-NE prototype that similarly tries to find temporal chains in communications data [I62]. In particular they are interested in finding things such as backhaul networks, TOR networks and botnet structures. It has the simple approach of finding temporally ordered chains of transactions on edges starting from a specified set of seed nodes. HIDDEN OTTER is essentially a reinvention of the Remit Chains algorithm, but in Hadoop.

BAKER'S DOZEN is a technique for finding batches of near-sequential phone numbers that display causal behaviour [I11]. Given population-level telephony data it generates a list of pairs of telephone numbers that are near-sequential. For each of these pairs it conducts tests to discover if they are causally related. One of these tests is temporally correlated communications with the same third party. This third party condition is important at population level as otherwise there are too many random coincidences due to identifiers merely being active at the same time. CLASP[8] was rejected for having little statistical power due to exactly this reason. The BAKER'S DOZEN test measures the proportion of events which involve a common third party and occur within t minutes of each other. A beta distributed prior and most powerful value for t were learnt from the data. The causal threshold was learnt by evaluating the statistic for 20 million random pairs and then choosing the value which led to a p-value of 10^{-6}. This statistic proved to be powerful in the sense of promoting many pairs above the causal threshold.

There has been a large amount of research on information diffusion and cascades in the external literature e.g. [E44, E18, E25, E29]. However the focus has tended to be on datasets where one can directly observe the pieces of information flowing through the network. Examples could be hashtags through Twitter or the spread of disease through a contact network. R66 at NSA have developed a MapReduce algorithm based on [E18] to track the passing of files between implanted machines [I35]. The reading rack for this problem (on [W24]) contains a number of citations of external papers considering information cascades and diffusion. Those papers should provide a good starting point in the literature, but is nowhere near exhaustive.

Internally there has been some research on block modelling [I28, I29, I30]. Block modelling

[8] Covered in subsection 4.2.2.

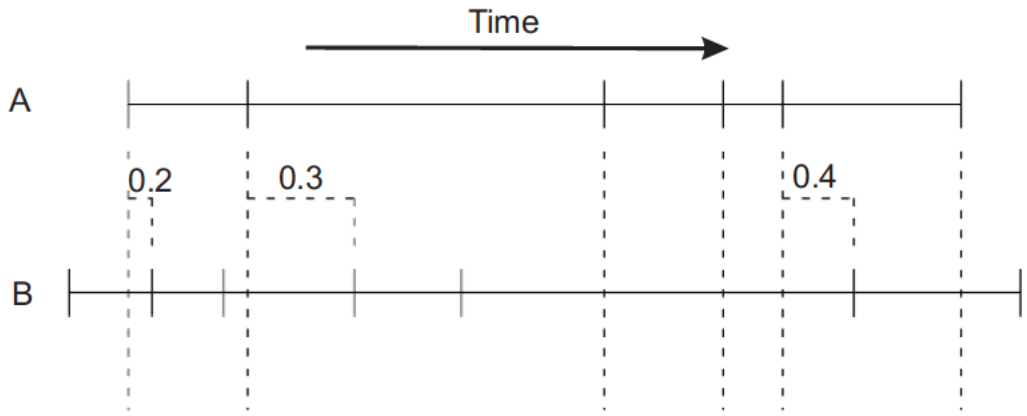


Figure 4: An example of the non-homogeneous Poisson process used in [I49, I64, I52]. Here we are testing the hypothesis that B's events are triggered by A's. We therefore "project" B's events onto A's timeline: the first falls 0.2 of the way between two A events, the second 0.3 and the third 0.4. This figure is adapted from [I64]

assumes that vertices in a graph each belong to different classes. The communication between vertices is then determined entirely by their classes. The job of a block modelling algorithm is therefore to assign vertices to classes and describe how the classes interact with each other. Although this has not so far considered information flows there is the possibility that they could be useful for block modelling. Further the outputs of certain block models may be interpretable in a similar manner to potential applications of information flows. For instance both may be able to distinguish directors, middle managers and workers in a company hierarchy. Is it possible to use block modelling to inform the discovery of information flows? ◀

4.2.2 Temporal correlation

Internally there has been much research undertaken in understanding temporal correlation between stochastic processes. This work should be a great aid in tackling the information flow in graphs problem area, especially when the information cannot be directly observed flowing over the graph. If we are interested in comparing adjacent edges then this work is directly applicable by restricting the scored processes to those with a common vertex.

This research started in 2005, motivated by a desire to find cross-media temporal correlations. An example of a cross-media correlation would be A calling B to arrange for B to initiate an instant messenger conversation with him. [I49] found that modelling the stochastic processes as non-homogeneous Poisson processes (NHPP) gave the best performance of the approaches attempted. This contrasts to PRIME TIME which models activity as a homogeneous Poisson process. Assuming that one can model the rate function of the NHPP correctly then the events of unrelated processes should fall uniformly with respect to each other. Figure 4 shows an example of this mapping. All tests seek to find deviations from this null hypothesis. The original paper proposed 14 tests for non-uniformity, some of which place particular emphasis on the start of the interval.

Ongoing research on this strand of temporal correlation can be seen to fall into two areas:

testing for non-uniformity and improving rate function estimation. The current best test for non-uniformity is the PCG statistic [I64]. This uses the fact that if there are k points uniformly distributed on (0,1) then the first of these is distributed according to a Beta(1,k) distribution. One can then calculate a left-tail p-value for each interval and combine them using Fisher's method. This simple method beats more complicated approaches [I6, I39, I23, I65] on the standard datasets. The current best technique for rate function estimation is described in [I52]. This is a two stage process. Firstly one clusters the set of stochastic processes. Secondly one counts time not in seconds but in the number of events that have occurred within a process' cluster. It is worth contemplating why this works. Consider the phones belonging to GCHQ employees – these cannot be brought into the building and so are very quiet between 9 and 5. If an employee turns their phone on at the end of the day and responds to a voicemail left earlier in the day then this activity has been triggered despite the multi-hour gap. By performing this transformation we turn this from a gap of many hours to one of a few events and we are better able to spot the causality.

The slide deck [I47] contain details of much of the research conducted before October 2010. This does not however include the cluster-based rate estimation from [I52].

Research into a streaming implementation of the PCG algorithm has been conducted in R1 at NSA [I51]. The work focuses mainly on data structures and approximations to allow the algorithm to remain within main memory. However given the large size of some of the datasets for this problem the techniques outlined may be useful should scaling prove to be a problem.

R1 have started to investigate using inference on a parametric model for how causal time series are generated [I24]. This proposes a mixture model where B's events happen either according to an underlying Poisson process or because of a causal A event. They demonstrate that this is a continous Markov process and formulate tests on whether given pairs of stochastic processes are likely to be correlated. When the model assumptions are correct their likelihood ratio statistic is tens of times more powerful than the best general methods known at small sizes for some generating parameters.

Many of these techniques are included in the CLASP software package, with new methods added once demonstrated as useful. [redacted] maintains CLASP. It is available on the LID at `/data/cryptomath_research/windata/infoproc/Software/CLASP/`

SAGA is a technique which extends a measure of item similarity to set similarity [I48]. It has provably desirable properties and has case studies that have demonstrated its utility. In particular it has been used as a method for performing temporal correlations. If one treats a stochastic process as a set of times and defines a similarity measure between times then SAGA may be applied to measure the similarity of pairs of stochastic processes. This approach is radically different to anything else attempted and can perform surprisingly well on the standard CLASP datasets.

4.3 What we care about now

This subsection will set out the problems that are of interest regarding information flow in graphs.

4.3.1 Definition and Discovery

The first and most fundamental problem is the definition and discovery of information flows in the graphs. This document deliberately declines to provide a mathematical definition of an information flow. There is not an immediately obvious definition and therefore it seems best to leave this as part of the problem. It was suggested by [redacted] of R1 that it may be a good idea to start from how you would define it given perfect information and then work backwards. However hopefully the examples given in the introduction sufficiently illustrate the type of things we hope to find. Further thinking about cascades as previously defined may help with a definition.

[redacted] of R1 suggested an approach for defining repeated information flows. One could phrase it as learning a distribution over when/which edge will have a transaction next given previous (and possibly future) activity on adjacent edges and further information, such as time of day. Repeated information flows could then be seen as high likelihood paths through this distribution. Can such a probability distribution be written down in a form where (approximately) evaluating it is tractable?

The Enron and SKB datasets are atypical of SIGINT data in that there is information on the content available. However this should be very useful for formulating definitions of information flow as it will be easier to see the flows occurring. In the SKB the flows correspond to various media being passed around the internet. The circulation of extremist media is of particular intelligence interest. It is suggested that these datasets be seen as truthed data and for gaining familiarity with techniques suggested in the literature. We are less interested in developing new techniques which depend upon having the content of transactions as this limits their applicability. We are therefore probably restricted to extracting flows which repeat rather than occur singly. What do the SKB and Enron datasets tell us about how well we can extract information flows *without* content? Can we perform exploratory data analysis on the SKB to inform the definition and discovery of flows? Can we spot typical transfer patterns?

Research on improving CLASP has been aided by the availability of two standard datasets. These each consist of two subsets—a random sample of processes for which there is no reason to believe any relationship exists, and a sample of pairs for which there is some external reason to believe a relationship may exists. This allows ROC curves [W34] to be compared between techniques and an objective comparison to take place. Can similar datasets and comparison mechanisms can be created for this problem and therefore help drive research collaboration?

There have been many different approaches to temporal correlation, both explicitly graph-based and not, as demonstrated in the previous subsection. Can we find a theory that unifies these approaches? One possible direction is to consider having placed a prior distribution on the probability of a significant temporal correlation being present. For example CLASP can be seen as putting a uniform prior over all pairs of edges, while PRIME TIME is uniform only over edges sharing a common vertex.

4.3.2 Missing data and noise

SIGINT data is almost always incomplete. In terms of this problem certain edges may not have been observed or some transactions on edges may be missing. In experiments carried out on billing records and SIGINT during the 2008 graph mining SWAMP at HIMR there was

shown be a huge disparity between our view of the world and ground truth [I73]. CSEC have perform similar analyses with similar conclusions [W4]. It is therefore important to be resilient to missing data, especially where a flow may be cut in two. An internal example of coping with missing edges is SALTY OTTER [W37]. It uses CLASP to find likely cross-media triggering patterns, for example telephone conversations typically causing instant messenger chats. The tool is essentially coping with the missing edge and allowing the information flow to carry on regardless.

There has been some external work on how sampling or missingness affects the appearance of information diffusion and cascades. [E9] evaluates how different sampling strategies affect the view of hashtag diffusion in Twitter. Clearly we do not generally get to choose how our data is sampled, but this work may help the understanding of how well/poorly we are likely to do. [E34] goes further in proposing a method to correct for missing data in information cascades. Their method assumes that cascades are k-trees, each vertex in the graph is sampled with uniform probability and the graph structure is known for sampled vertices. Given these, they claim to be successful in reconstructing properties of the original cascades. These external approaches assume that we have the content of a transaction—is there anything we can do when we do not? ◀

The obvious approach to this problem is to remove edges/transactions from a dataset to simulate poor collection. We can then evaluate different coping strategies by seeing how our performance is impacted. Here the Enron dataset is probably a good place to start, as we have ground truth and can uniquely guarantee that it is the complete dataset. However any technique developed must behave sensibly on SIGINT data.

The data that we do have has further problems beyond missingness. In particular the quality of the timing information is not as good as we might hope for. This presents at least two concrete problems. Firstly, our data tends to have second timestamps, which may be too coarse a measure for many applications. Does the granularity of the timestamps affect our chances of finding causal flows? Secondly the clocks on our probes are not synchronised. This means that there is likely to be a constant offset between events happening on different bearers. Any technique to correct for this offset will both aid this problem area and be of general interest to the internal data mining and information processing community. Can we correct for the clock offset between probes? Possible solutions may involve examining the same connection being intercepted on different bearers. ◀ ◀ ◀

4.4 Potential future interests

There are further problems in this area that may become tractable as the subject knowledge grows.

4.4.1 Performing inference on flows

Assuming that information flows can successfully be identified and extracted we should then be able to perform inference on/with them. The obvious first area to investigate would be anomaly and change detection. The interest in this was hinted at in the introduction in investigating how a target network changes after the removal of a commander. Given that this document does not even define a flow then it is not reasonable to scope this future problem any more

tightly. Should you reach a point where you can tackle this problem you should have a good idea of what anomalies and changes mean within the framework you have developed.

4.4.2 Information flow for graph generation

There are many existing models for graph generation. Examples include the Forest Fire algorithm [E26] and Bollobás-Janson-Riordan family of graphs [E5]. However, these approaches, although sequential, do not describe how graphs are truly generated. That is, they do not accurately correspond to how a graph, and the transactions on it, are generated in reality. If the definition and discovery of information flows is successful then it may be possible to use the descriptive models for graph generation. This feels far closer to how graphs are really generated. Each transaction is undertaken to convey information. Therefore adequately modelling the flows leads to the observations. [redacted] of [redacted] may well be interested in such ideas as he has stated dissatisfaction with the existing approaches. There is probably limited SIGINT interest in this problem unless a convincing argument can be made otherwise. We know of no internal work on *any* subject which has used *any* graph generation algorithm.

4.5 Relevant data

We have several relevant datasets with truthed data, of which some have already been mentioned in the main body of this section.

The Enron (appendix F.2.1) and SKB (appendix F.1.4) datasets are atypical as most SIGINT data does not have any content associated with it. They can be treated as truthed datasets for the evaluation of algorithms for extracting significant information flows.

We also have a large dump of FIVE ALIVE (appendix F.1.2) that summarises all IP connections on research bearers. There is no content associated with this data. We do have some truthing on flows that may exist in the data. Specifically, we have data on covert infrastructure (appendix F.3.3) used for exfiltrating data from CNE implants. These suspected flows can be used for both EDA and evaluation purposes. Further, we have lists of IPs that we suspect to be infected with the Conficker botnet (appendix F.3.4), either due to signatures collected or behavioural analysis. Again, we suspect that there are some information flows involving these IP addresses.

We also provide two standard datasets used for evaluating temporal correlation algorithms (appendix F.1.5). If you have any insights on how to perform temporal correlation due to your work on this problem you may wish to use these for evaluation purposes.

4.6 Collaboration points

There are several collaboration opportunities available for information flow in graphs.

NSA R1 [redacted] coordinates the research into temporal correlation and is always happy to hear of new ideas and approaches. He also indicated an interest in this new research area.

ICTR-DMR The temporal analysis tools PRIME TIME and SALTY OTTER were developed in ICTR-DMR. Although they currently are not working in this area they would certainly be interested in any results. [redacted] should be your first contact in ICTR-DMR.

ICTR-NE ICTR-NE are interested in using information flows to find Tor routes, identify backhaul routes and map botnets. They currently have a Hadoop prototype called HIDDEN OTTER which performs simple temporal chaining. They would be very interested in any work you produce and may wish to collaborate. HIDDEN OTTER was produced by [redacted] and [redacted]

ICTR-CISA The streaming analysis team have had a streaming PRIME TIME developed by Detica. They are always interested in streaming algorithms and deploying them as research prototypes. If your research takes you in a streaming direction then you should contact the streaming analysis team led by [redacted]

CCS Bowie [redacted] of Georgia Institute of Technology is a leading academic figure in large graph analysis. He is cleared and has previously worked as a consultant in NSA R1. He is now in the process of joining CCS Bowie in a similar role. He is interested in this problem area and may be a possible collaborator on both classified and unclassified work.

US National Labs At Sandia National Laboratory [redacted] and [redacted] are leading research on large graph processing for defensive analysis.

5 EDA on Streams

5.1 Introduction

5.1.1 EDA

Exploratory data analysis (EDA) is all about trying to find interesting features of data without necessarily having pre-formed hypotheses to test. One of the pioneers of EDA was J. Tukey [E41, E42], who argued for the value of EDA over the traditional statistical approach, which he called confirmatory data analysis, where one starts with an hypothesis and collects data in order to test it. In EDA, the data comes first, and what counts is understanding the data as it is.

For the data analyst, this is an open-ended problem that is not tightly defined, but for the mathematical researcher developing algorithms, things are much more concrete. The aim is to use one's intuition, guided by domain-specific knowledge from the analysts, to develop precise algorithms that provide human insight on the data.

We usually think of EDA as being concerned with

- pulling out global properties of data;
- broad-brush visualizations of data.

The second is really a variant of the first: we can reduce the data to more discrete values than a human could take in in a list or table, as long as there is a way to visualize them. (Compare summarizing pairs by a correlation coefficient, or in a scatter-plot.)

5.1.2 Streams

In a *stream* we do not have enough memory to store everything we see, and we only get to see each piece of data once. Many problems admit simple approximate solutions in the static setting by subsampling. In the stream, this option is not always available. The problems become much harder and controlling error estimates in approximate solutions is very difficult. On the other hand, streaming analysis gives us the opportunity to get situational awareness and real-time tipping from our data, as well as letting us process bigger datasets than we can afford to store. These are key benefits that we strongly want to capitalize on.

For hands-on work, we are thinking of DISTILLERY, as opposed to Hadoop (see appendices B and C).

One way to think about the problem is in terms of data structures. There are only a few structures that we typically use to keep track of data when we write programs: lists, trees, heaps, hash tables and so on. What carries through to the streaming setting? Which structures can we update in a stream? If we can tolerate some loss, can we maintain approximations to familiar data structures in the stream? If so, can we quantify and bound the errors? These streaming data structures are then the building blocks for streaming algorithms. Given a particular data stream, what is an appropriate data structure that will capture what we need to know about the data in order to answer the SIGINT questions we have?

A short survey summarizing various approaches to streaming data can be found in [E39]. The 2009 Information Processing SCAMP at La Jolla also produced relevant material [I12].

Note on terminology

When we speak of streaming graph algorithms at GCHQ, we are usually referring to what the external literature calls the *semi-streaming paradigm*. If the graph has n vertices, then we can typically store a small amount of information for each vertex, but we are not able to store all the edges or do any significant processing as each edge arrives. In other words, we assume we have $O(n \log n)$ storage, and can do $O(1)$ work per event. (Usually this can be $O(1)$ amortized work, as long as this does not cause undue back pressure: see section B.5.1.)

5.1.3 The problems

The problem areas on this topic overlap at the edges, and also tend to merge into the streaming expiring graphs problems, but to give some order to this section we loosely cluster them into four areas:

- graph problems with no sub-sampling allowed;
- visualization;
- modelling and outlier detection;
- profiling and correlation.

5.2 Graph problems with no sub-sampling

5.2.1 The framework of graphs and hypergraphs

Events data frequently has a natural representation as a graph, or more generally a hypergraph. Often, an event will be a communication between two entities, which we think of as an edge between two vertices, one vertex for each entity. There will normally be a notion of the originator and recipient of the communication, which makes the graph into a directed graph. Sometimes, a communication can involve more than two nodes, in which case we can think of it as a hyperedge, and the overall structure a hypergraph[9]. We also look at graphs other than communications graphs: for example, colocation graphs, where vertices are joined by an edge if they were geolocated to the same place at the same time; network graphs, whose edges are physical links; or even semantic graphs, where nodes are concepts and edges relations between them.

Frequently, our data will come with additional information beyond the simple fact that a communication took place. For example, each vertex will have a boolean attribute, 'Is this entity a target in BROAD OAK?' Similarly, edges might have attributes like 'duration of communication'. A common metaphor is to think of discrete attributes as *colours* and continuous attributes as *weights*. Although we often need to do algorithmic computations on the underlying graph or digraph, taking account of the available attributes can enrich the SIGINT value of any analysis we do.

[9]Some people prefer to think of simple hypergraphs as bipartite graphs, where the vertices and hyperedges are the two parts, and edges represent inclusion.

We are interested in finding global properties of graphs in a stream—exactly if possible, but we expect that approximate solutions will often be the best we can hope for. This is obviously closely related to the streaming expiring graphs topic, but in our case we are not worried about expiring edges, and we focus more on counting rather than identifying and extracting graph structures. Probabilistic counting in general (not specifically in a graph context) has been an area of active research both internally [I72, I40, I5] and externally [E16, E8] in recent years.

5.2.2 Cliques and other motifs

An *n-clique* is a subgraph isomorphic to a complete graph K_n. In a communication graph, this corresponds to an intuitive idea of a strong, close community, where everyone communicates with everyone else.

For EDA purposes, we would like to understand the clique structure of a streaming graph. What are the cliques? If a target node belongs to a k-clique, how surprising is that?

One way to answer the second question is to get a good random graph model for the communication graph, and do Monte Carlo simulations to find out how likely k-cliques are to occur in the model graphs. There has been a lot of work on this, for example [I1, I57], but it has proved very difficult to find models that capture all the relevant properties of SIGINT graphs, or even to understand exactly what 'relevant properties' we want to capture. An alternative approach is to just work empirically with the graph we see, and try to estimate how many k-cliques it has: this gives us some measure of how surprised we should be if target nodes belong to such a clique.

This leads us to consider *probabilistic counting*. We might want to count not just cliques, but other subgraphs too: perhaps a clique with one edge missing. A *motif* in a graph is a subgraph isomorphic to a particular pattern graph: for example, when the pattern graph is a K_n, the motifs matching it are the n-cliques. There are probabilistic algorithms for counting the cardinality of a set: for example, Flajolet et al.'s hyperloglog sketches [E16], proposed on the outside and extended internally by [redacted] [I72]. There are also a variety of algorithms for counting triangles, i.e. 3-cliques. One example is [E40]; [redacted] [E8] has produced a good survey.

Is there a probabilistic counting algorithm for cliques or other motifs in a streaming graph? What can we say about error bounds? ◀

Besides counting, we might also be interested in motif *collection*. If we have two fixed target nodes then motifs containing both nodes will give information about their common neighbours. For example, how many distinct V-shapes or squares contain them both? Some CSEC work [I21] from a few years ago may be relevant.

Can we collect specified motifs containing a target node or nodes? ◀

Removing *pizza nodes* (i.e. very high-degree nodes) is likely to be an essential prior component to get useful results. Intuitively, a pizza node is likely to be a large impersonal entity like a pizza parlour or an electricity supplier: the fact that two people both communicate with the pizza node gives us no reason to think that they are linked socially.

5.2.3 Trusses

Let $k \geq 3$. A *k-truss* is a connected graph with ≥ 2 vertices such that every edge of the graph is reinforced by at least $k-2$ pairs of edges that make a triangle with that edge. This concept was developed by [redacted] at NSA: the idea is to weaken the definition of a clique (any k-clique is certainly a k-truss) to allow for a few missing edges, but still capture the notion of a cohesive community, and at the same time produce something that is computationally tractable.

People in the SIGINT community have looked quite a bit at trusses, following on from the foundational theoretical work by [redacted] [I15, I14] on properties of trusses, their relationship to cores and cliques, and streaming algorithms to find them. In particular, there has been some experimental work [I76] looking at trusses in communication graphs. The findings were surprising: there turned out to be huge k-trusses for quite large values of k, like $k = 17$. This was true even after splitting trusses at cut-points. A number of variants and generalizations have also been proposed (for example [I16, I17]).

We would like to understand why these form. Is there a better definition of truss that captures something like a closed-loop intuition (see section 2.1.3) without pulling in huge monstrosities? ◀

In particular, as we have mentioned, a truss can have cut-points, i.e. single vertices whose removal disconnects the graph. On the other hand, trusses have high edge connectivity: one has to remove at least $k-1$ edges from a k-truss to make it disconnected. Can we define truss-like structures with a different balance of vertex and edge connectivity? Do giant structures still form? ◀

Can we use a partial order derived from truss or core structures to perform hierarchical clustering? If so, can we avoid forming giant clusters? ◀

Can we understand when community detection or clustering algorithms produce giant clusters? Are there ways to prove (given some probabilistic model for the graph) that with high probability an algorithm will not produce large clusters? One specific suggestion by [redacted] is to look at clique percolation [E4] where there are multiple labels per node. ◀

What is the background distribution of sizes of k-trusses? Is there a probabilistic solution (cf. the previous section)? ◀

5.2.4 Other approaches

There are also more open-ended questions about streaming algorithms for graphs.

What graph invariants are both useful and can be found or approximated in a stream? ◀

In the academic world, there is a whole cottage industry devoted to coming up with new clustering algorithms. Many will not have much use beyond allowing someone to publish a paper. Is there a hidden gem in the open literature that the SIGINT community has missed? ◀

This problem obviously has the potential to lead one off down rabbit holes. As a concrete thing to look at, the first author has identified BIRCH [W3] as an algorithm that may deserve a hearing.

Can we compute any measurements of centrality or betweenness in a stream? (We are more interested in centrality measures in subgraphs around targets: CHART BREAKER [W6] vertex scores do something like this.) How stable are they as the graph evolves? Is there concept drift? ◀

Can we approximate the graph distance distribution, and see how it varies with the pizza threshold? ◀

This has a bearing on what hop distance we should choose for contact chaining. Conventionally, analysts focus on a 2-hop neighbourhood of their targets, but some work comparing billing records with SIGINT [I73] found that one needed to chain much, much further through SIGINT to reach a 2-hop neighbourhood from billing data. Can we use the SIGINT to billing mapping (SOLID INK to FLUID INK—see appendix F.1.6) to help decide what the right thing to measure on a telephony graph is? ◀

CSEC have also done some work [W4] on comparing SIGINT and billing records. Billing data is unlikely to be shareable, but for comparing results on different datasets, H4A would be a natural point of collaboration.

5.3 Visualization

5.3.1 Visualization in general

For most people, visualization is a crucial ingredient in the sense-making loop when given a large amount of data to analyse. GCHQ is actively developing tools for visual analytics. A large team in ICTR, split between MCA and DMR, works on semantic graphs and visualization research [W14], and a visual analytics tool called MAMBA [W27] is currently being developed in partnership with Detica. For graph visualization, NSA's Renoir application [W33] is also under active development.

As HIMR researchers explore data for themselves, they will naturally develop their own visualizations to help them understand it. We encourage them to record what they come up with: perhaps some of these ad hoc visualizations could be useful to analysts too.

HIMR's expertise is obviously in algorithms, not developing sophisticated visual analytics platforms. Nonetheless, what dynamic or interactive visual tools would be helpful to explore SIGINT data sets, if someone else could be enlisted to create them? ◀

GRINNING ROACH [W17] and PIRATE CAREBEAR [W30] are existing tools for visualizing SIGINT events, developed by DMR: they both produce plots for pattern-of-life analysis.

Dashboarding is well-established for electronic attack events, both internally and by anti-virus and security companies. Can similar methods be applied to provide useful visualizations for traditional SIGINT analysis? ◀

There is some work in progress at GCHQ [W5] on dashboarding for the 2012 Olympics, but it is fair to say that the approaches so far are not mathematically sophisticated.

5.3.2 Streaming plots

There has been some work in R1 on binning streaming data for histograms [I37].

What interesting plots can be produced in a stream? ◀

[redacted] suggests starting with QQ-plots; this is closely related to the problem of computing approximate quantiles of streaming data.

CISA have also done some work [I55] on time series modelling in a stream, including bundling up R for use in DISTILLERY: this may be a good foundation to build on.

If any algorithms of the sort discussed in section 5.2.4 that calculate summary statistics are ◀

produced, could they be refined to produce plots (from which those statistics could be read off)?

5.4 Modelling and outlier detection

5.4.1 Identifying outlier activity

Outliers (e.g. low-volume telephone numbers, small connected components) are often exactly what SIGINT is interested in. They are also exactly what gets lost in subsampling.

How can we find rare events with limited memory? ◀

Can we take 'beyond supervised learning' to the limit, and find a way to classify normal and abnormal behaviour from the data itself, without needing to train a classifier? A simple idea would be to choose an N in advance, classify the first N items in the stream as 'normal', then use a positive-only learning algorithm to build a classifier to apply to the remainder of the stream. Can we do anything more sophisticated to bootstrap a classifier out of the data itself? ◀

Work from [redacted] at LLNL is relevant to this. He is learning a Gaussian mixture model on cyber data with particle filters and asking about newness by looking at probability density. Can we ask about tail area instead? This question has also been posed to the 2011 NSASAG [W28].

Can we track new small connected components? This might be a group of targets who have dumped their old SIM cards and replaced them. ◀

5.4.2 Background distributions for significance tests

We have already touched on the idea that we want a measure of surprise when we find outliers (section 5.2.2), and for this we want to know the background distribution: what does 'normal' look like, and how can we quantify that? This section gives some specific examples of outlying behaviour that we look for, and for which we therefore want to find an empirical background distribution. Any information along these lines could also feed into tests in Dynamic Graph.

What is the distribution of the number of common neighbours of two nodes in a graph as a function of their degree? This is one way to try to measure the strength of association of two entities. ◀

What is the distribution of component sizes? Terrorist cells and other target groups have been found because they form small components (or 'closed loops', to use the analysts' term) isolated from the giant component. How surprising is it to see a node in a component of a given size? Likewise for other measures of connectivity. ◀

What significance do various CHART BREAKER [W6] relationship scores have? This involves looking at an email hypergraph, rather than just a simple graph. ◀

5.4.3 Window sizing

We often want to pull off a finite chunk from a stream, either for offline analysis or for change detection metrics.

How should we choose the window size? Is there a happy medium between a narrow window ◀

(little data, so large variance) and a large window (concept drift, so large variance) that leads to small sample variance?

Can we do something akin to ANOVA analysis to look at the effect on sample variance of sample size versus concept drift across different window sizes? ◀

5.5 Profiling and correlation

We are often interested in finding nodes that behave like a target node: they might be following the same modus operandi, or be another selector for a known target.

5.5.1 Correlations

With millions of entities, there is no hope of storing useful information on all pairs.

Is there a sparse approximation to a correlation matrix? ◀

AUTO ASSOC [W2] scores may provide a relevant example of a large correlation matrix. These are similarity scores for pairs of *target detection identifiers* or TDIs, which are unique, persistent identifiers associated to particular users or machines that indicate their presence on the network: the aim of AUTO ASSOC is to find out when multiple TDIs belong to the same user or machine. See section 3.3.2 for further discussion of association scores.

Can we keep an approximate list of the top N nodes most closely correlated with a given target node? ◀

There is also the underlying question of how to score association. This is not strictly about EDA on streams, but looking at how existing scores perform on streaming data might suggest ways of improving them.

How can we score the association between two nodes? CHART BREAKER [W6] gives a significance score between pairs of nodes based on emails exchanged. There is an ad hoc balancing between the value of an email where one side is sole recipient, cc'd or bcc'd (cf. assigning weights to golds, silvers and bronzes in medal tables). Is there a method with a better theoretical justification behind it? ◀

Can we correlate the 'busyness profiles' of nodes, for example to provide situational awareness of a DDOS attack? ◀

5.5.2 Finding behaviour that matches a model

Frequently we have a modus operandi known to be used by particular targets, and we want to search for events matching that model in streaming data. Recent work by [redacted] on low-rank approximations [E1] may be useful: she has a general framework called CPD analysis that uses tensor decompositions to model multi-variable data and extract meaningful factors as rank 1 tensors. Reducing dimension should make it easier to match up features. (This also has applications to link prediction, which is pertinent in SIGINT applications where we expect to have a lot of missing data.)

If a target disposes of his phone and buys a new one, can we rediscover it in data? ◀

Can we find IP addresses fitting the profile of, for example, a box engaged in a denial-of-service attack, or an implanted box beaconing to a C2 server? ◀

Mathematically, this may come down to something like solving an approximate subgraph isomorphism problem. CSEC's 2010 SAWUNEH did some exploratory work on data mining for cyber defence [I82], which gives some concrete examples of malicious behaviour to look for in events data. There is also existing work along these lines for botnet detection in CROUCHING SQUIRREL [I27, I71], and it may be interesting to compare with external work on streaming botnet detection by adaptive sampling [E45].

5.6 Easy entry problems

This section has some ideas for problems that do not have high entry requirements in terms of reading up on existing literature or doing lots of preliminary data manipulation: they might be a good place to start for people who like to get into things quickly.

Maths route:

- Motif finding
- Properties of trusses and their generalizations
- Finding outliers

Data route:

- Visualization
- Streaming QQ plots

5.7 Relevant data

This problem set has the advantage that EDA is needed for any and all of the streaming communication datasets we have available: the telephony, email, HRMap and cyber datasets all readily map to graphs (or hypergraphs), and present challenges for all four areas: streaming graph analytics, visualization, outlier detection and correlation. There is also a graph of the links between Wikipedia articles (appendix F.2.3) in case researchers want a static graph of links to compare with the dynamic graph of clicks provided by HRMap. Appendices F.1.3, F.1.1, F.1.2 and F.1.6 describe some particularly appropriate datasets, but most of the datasets in appendix F could usefully be explored.

Since EDA is such a general requirement, it is equally possible to work with unclassified data sets. Appendix F.2.2 describes a dataset being analysed by the UKVAC (see section 5.8.2): besides being another source of events data that is somewhat different in nature to communication data, it would also be useful to work with this data should any collaboration develop with UKVAC participants.

5.8 Collaboration points

5.8.1 Internal

ICTR-DMR. [redacted] has the best knowledge of how analysts work and what will be useful to them. He also champions research on payphone activity. [redacted] developed some of the fundamental algorithms currently used at GCHQ for contact chaining and scoring strength of association. [redacted] leads on EDA, though not specifically focused on streams. For visualization, [redacted] and [redacted] are involved with the MAMBA project.

ICTR-CISA. [redacted] is a DISTILLERY guru, and [redacted] tracks research on streaming algorithms across the community. [redacted] is also a good source of information on DISTILLERY and streaming implementations in general.

NSA/R1: information processing group. There are already good contacts with R1 from HIMR's crypt work, and it would be good to build on that: for example, [redacted] is a frequent visitor to HIMR and is always interested in questions about probability, random hypergraphs and stochastic processes. [redacted] (currently sitting in LTS) has published on EDA on streams, and is planning to write a book on the subject.

KACHINA. Sandia National Lab in the USA has a multi-year project called KACHINA to look at large graph processing for defence analysis. A good point of contact is [redacted] (NSA/R4).

Pod 58: cyber exploration. In particular [redacted] (NSA/R1), who has visited HIMR in the past.

5.8.2 External

UKVAC (UK Visual Analytics Consortium). One of the two challenge problems for Phase 2 (approximately 18 months from May 2011, subject to funding) asks for visual analysis of 120M events (several years' worth of flight arrivals and departures in the US—see section F.2.2). The brief they have been given is very closely aligned with the streaming events model described in this section, and there is as much of an overlap with the problems here as is possible at UNCLASSIFIED. If anyone in the SIGINT community is going to collaborate directly with UKVAC, it will probably be HIMR.

There are five fairly independent groups working as part of the UKVAC. Imperial [redacted]) and Oxford [redacted] do substantive mathematics. Middlesex [redacted] and UCL [redacted] do substantive non-mathematics. Bangor [redacted] seem most engaged with this dataset so far. The best thing is probably to spot promising activity that emerges, get in touch with the people doing it, contribute suggestions and hope that this leads to collaboration. In the fairly likely event that it is difficult to track what is happening and who is doing what, [redacted] (Middlesex) has high betweenness-centrality in the graph of UKVAC participants, and would be a good first point of contact.

INSTINCT. The UKVAC is sponsored by INSTINCT, a UK government project to use data mining for counter-terrorism, led out of the Home Office. They organize other activities too, most recently a public competition on ways of fusing data streams [E20]. Some of their projects will be more relevant than others, but it may be worth keeping an eye on what they are doing. Upcoming projects usually get mentioned on blogs on GCWeb; [redacted] will also be able to suggest contacts if required.

[redacted] **(AT&T).** An expert in probabilistic counting; has been keen to engage with GCHQ at the UNCLASSIFIED level.

IBM Safer Planet. This is a big corporate project covering some of the same ground as this problem book. [redacted] is in touch with the organizers, and is keen to look for opportunities to get GCHQ and HIMR involved.

6 Streaming Expiring Graphs

6.1 Introduction

Streams of transactional events often arise in SIGINT problems. A classic example would be telephony events where we have a directed event from one telephone number to another when we detect a phone call. In this case we would typically have a relatively small number of target numbers we are interested in and would collect the events around these – we call this a seeded graph. An event or transaction is a (normally timestamped) observation of an edge, and so for each edge in the graph we may see multiple events. In some recent problems, for example electronic attack events, we have been interested in looking for structure in the entire graph. A denial of service attack might be visible for example as a vertex which suddenly has many incoming edges.

We have techniques for handling the seeded case in a streaming way, expiring old edges and maintaining a current view of a graph, for example GCHQ Dynamic Graph [W12] and associated simulations completed by NSA [I2]. This research area is about investigating the second case where we are interested in tracking the full graph as it varies over time. We imagine that we want to expire old events or edges somehow. This might be by maintaining a buffer of the most recent n events, maintaining the n most recently seen edges, or by decaying edge weights over time and expiring those with the lowest weights. Other decay strategies might also be appropriate. For some problems we may not need to store the full window, and can instead find an analytic that produces equivalent results. Any algorithm should ideally parallelise so that we aren't restricted to the memory or network bandwidth available on a single computer.

In a *dynamic graph* problem the typical aim is to maintain a data structure for answering queries whilst also receiving updates to the graph. The aim is to maintain information that can be updated efficiently given the stream of changes to the graph, and to avoid total re-computation for each query. We say a graph problem is *fully dynamic* if the updates include both insertions and deletions of edges. A problem permitting only one type of update (insertion or deletion) is sometimes described as *partially dynamic*. Some literature uses the term *evolving graph* instead. An old but good overview of some dynamic graph algorithms is given in [E14].

An *expiring graph* can be thought of as being a dynamic graph where we allow arbitrary edge insertion, but edge deletion is restricted to one of a small subset of the edges, for example the oldest or lowest weight edges.

As in the EDA on Streams problem (section 5), we expect solutions to run in a streaming fashion on the DISTILLERY platform. We are also interested in how we might bootstrap such an algorithm using a map-reduce job on a Hadoop cluster where that makes sense, however this is not the main focus.

6.1.1 The Problems

In section 6.2 we list graph properties which we would like to be able to find and track as the graph evolves. We allow some freedom in how the graph evolves. Edges may decay over time with low weight edges being expired. We might maintain the most recently observed edges, or we might retain a window of the most recent events, either chosen to be a fixed size or over a fixed time period. We expect different problems to be possible with different expiry

mechanisms so the choice should be considered separately for each problem.

For each property we list, we are interested in answers to the questions posed in section 6.3. We list some further extensions in section 6.4.

6.2 Properties to find and track

In this section we assume a graph $G(V, E)$ with n vertices in vertex set V and with edge set E. We use d_v to mean the degree of vertex v.

6.2.1 Component Structure

In most SIGINT graphs we empirically expect to find a giant component containing most of the vertices (see for example [I34] and [I33]). It has been shown in the past that examining the remaining components can yield valuable intelligence [I32]. For example a HUMINT agent and their handler might use specific phones to speak to each other and never use these phones otherwise. Terrorist cells might have separate phones for calling each other; again these would never contact numbers in the giant component of the graph. We are therefore interested in finding these small components (note that this interest is very sensitive).

Component tracking has been studied for dynamic graphs for example [E2].

Can we identify small components in an expiring graph? The query could include a time since which edges should be considered, or such a time might be implicit in the expiry strategy. ◀

Can we track the component structure of an expiring graph to be able to answer a query such as "is there a path between A and B with all edges having been observed since time t?" ◀

Given an approximate solution to these problems, can we provide an error estimate, for example upper and lower bounds? These might for example take the form of the maximum proportion of queries for which we provide the wrong answer. ◀

6.2.2 Graph Distance

The distance between two nodes in a graph can be an indicator of how related they are, for example in contact-chaining analysts will often look at the two-hop contact network of a target. For some graphs we might like to be able to answer queries of the form "what is the distance from A to B with all edges having been observed since time t?". We can think of the graph as being either directed or undirected, and weighted or unweighted. We would typically remove high degree vertices before asking such a question. External work in the area includes [E15].

Give an approximate answer for the distance between any two vertices for edges observed since time t, including error bounds. ◀

A related problem is to provide alerts when the graph distance between two sets of vertices goes below some threshold, for example if two groups of targets are seen to communicate. Can we track the distance between two (possibly dynamic) sets of vertices. Can we efficiently identify when two sets of vertices have a length d path between them? ◀ ◀

6.2.3 Cliques and other motifs

In section 5.2.2 we describe the problem of counting cliques and other motifs. Network analysis suggests that some structure may be important for example in target identification or malware

detection. Where we expect a structure to be rare, the appearance of such structures may be an anomaly we wish to investigate.

In telephony data, cliques or near cliques with few connections to the remainder of the graph is a known MO of certain target groups. The members are in frequent contact with each other, but rarely call others. In the case where they never call other numbers we describe it as a *closed loop* (see section 6.2.1 on the preceding page). Can we find and track cliques or near-cliques which are persistent in the graph over time? If a new number enters a clique (or near clique) at the same time as another member ceases communication we might infer that a user has changed phone number. Can we identify such occurrences? ◀

Bounds here are likely to be based around the size of the clique found, for example given there exists a k-clique in the graph what size sub-clique does the algorithm guarantee to find?

Our graph also has a time element – edges are observed repeatedly. Given a timestamp, can we extract all cliques or near-cliques of some size where all edges have been observed since that timestamp? Can we do this for other motifs? ◀

6.2.4 Centrality Measures

The centrality of a vertex in a graph measures the relative importance of that vertex. For example it might show how important a person is within a social network, or how important a website is in terms of reachability of other sites. Common centrality measures include the degree, betweenness, and eigenvector centrality.

The simplest is the degree centrality, defined for each vertex as the number of incident links, scaled by the possible number, that is

$$C_D(v) = \frac{d_v}{n-1}.$$

Tracking the (approximate) degree centrality in $O(n)$ space is relatively easy without expiry of edges, but can we track it for each vertex in the case of an expiring graph, for both the weighted and unweighted cases. ◀

The vertex betweenness centrality of vertex v is (informally) the proportion of all shortest paths in the graph which pass through vertex v. If σ_{ab} is the number of shortest paths between a and b, and $\sigma_{ab}(v)$ is the number of shortest paths between a and b passing through v then

$$C_B(v) = \sum_{a \neq v \neq b \in V} \frac{\sigma_{ab}(v)}{\sigma_{ab}}.$$

We are not particularly interested in the global betweenness centrality, but would be interested in ways to track it for specific subgraphs, for example the 2-hop graph around some set of seed vertices.

Is it possible to maintain an approximation to the betweenness centrality for a set of vertices as the graph (and the vertex set) evolves? Some internal work in this field is described in [I46]. ◀

The eigenvector centrality scores nodes in such a way that high scoring nodes contribute more score to their neighbours than low scoring nodes. A variant is the Google PageRank algorithm [E30]. In the basic case, the eigenvector centrality of vertex v is the corresponding entry in the eigenvector of the adjacency matrix of G corresponding to the largest eigenvalue.

Again we aren't directly interested in the global eigenvector centrality measures, but would be interested to track local variants, for example Personalized PageRank or the internally developed KL-Relative PageRank [W45].

Can we track any personalized variants of the eigenvector centrality for some (possibly changing) set of vertices in an expiring graph? There has been internal work on updating eigenvectors and eigenvalues as more edges are observed at the Information Processing SCAMP in 2009, see [I81], along with a report on its possible implementation [I56]. ◀

6.3 Questions relevant to all properties

6.3.1 Approximation

Typically it is not necessary to know the exact values of the properties listed above, and we can make do with an approximation. For an approximation to be useful it should include some form of error bounds, although the form these take will depend on the specific problem. They could include ϵ-δ bounds, strict upper and lower bounds, errors with a known statistical distribution, etc.

Are there approximate solutions to any of the problems listed? Where an exact solution exists, how does the computational cost (time and memory) compare? ◀

6.3.2 Computational Cost

For each of the problems listed in section 6.2 we would like to know the cost of evaluating the properties in this way. For our purposes cost is CPU time and memory usage as a function of the data size (asymptotics are important but we also care about the constants as derived from experiments).

We typically work under the semi-streaming graph model where we allow ourselves $O(n \log(n))$ space. For example, we might imagine storing a component ID for each vertex.

For most problems it would be possible to collect a window of data from the stream and re-compute the required statistics at the desired query interval. Whether this is practical depends on the window size, the frequency of updates to the graph, and the frequency (and latency) with which an answer to the query must be returned. The trade-offs should be considered – incremental updates might take more compute overall, but in situations where we can take some automated action based on the results then we might be willing to accept the cost to gain the low latency.

In most settings it is unnecessary to know the answer to a question for every edge addition or deletion, and it is instead sufficient to be able to compute the answer after each batch update, so long as those updates are sufficiently fast.

Furthermore, the online process may track and store data to allow efficient updates, but to get the desired answer may then require us to further process the data we have stored. We might then choose to run this further processing less often, for instance at the request of an analyst. This is a perfectly valid approach, and could be particularly valuable if the data structure lends itself to answering multiple types of query.

Concrete questions include:

- What is the (mean) cost of an update? What is the worst-case cost? ◀

- At what query frequency does the total computational cost of incremental updates become lower than the cost of total re-computation on each query? How does this depend on batch size? ◀
- How does the computational cost vary depending on features of the graph, for example diameter or average distance. ◀
- How does the cost vary depending on the window size or the decay rate? ◀

6.3.3 Expiry Policy

As well as affecting the speed and computational cost of an algorithm, the choice of expiry mechanism will affect the accuracy of the results.

For windowed data, how should we choose the window size to ensure we get realistic results at a reasonable speed? Is it possible to dynamically change the window size? ◀ ◀

For decaying data, how should we chose our decay rate to maintain realistic results? Can we change the decay rate without restarting the algorithm? ◀

6.4 Further Questions

6.4.1 Parallel and Distributed processing

For high rate data feeds it may be necessary to process the data on multiple nodes of a cluster. The data feed would be split between nodes and these streams cannot be combined until their rate is sufficiently reduced. Which of the graph properties can be computed in a parallel way? ◀

Some SIGINT data sources are split between multiple geographical sites, with limited bandwidth between them. Is it possible to solve any of these problems for the (virtual) stream of joined data? In this case we would expect to process each feed at the collection site and send a much smaller set of data between sites, either periodically or in order to answer a query. ◀

6.4.2 Bootstrapping

For some properties it may be possible to get an initial approximation to the correct values by running a map-reduce query on an events Hadoop cluster. Can we make use of bootstrapping to improve the efficiency of our processing? This might be particularly relevant when we process the stream in parallel and wish to split the vertices over multiple nodes of a cluster with each node being responsible for some proportion of the vertices. ◀

6.4.3 Anomaly Detection

For many of the properties we wish to compute, we would also like to be able to produce an alert for anomalies in the data. For example, in the web graph, if a vertex is suddenly connected to a large number of other vertices this may indicate a denial of service attack. Alerts may be used to trigger additional processing, for example capture and storage of relevant data or additional processing to categorise the event. Some internal research in this area can be found in [I20].

Can we detect significant changes in the properties we are tracking? How soon are we able ◀

to detect the change after it initially occurs?

6.4.4 Resilience

Over time our collection posture changes. Bearers are tasked and de-tasked, and processing systems can fail. This can have a significant effect, especially when monitoring for anomalies. How are the algorithms affected by variations in data volumes? ◀

If we identify that a change is due to a processing failure, can we account for that when generating future alerts once the processing resumes? See for example [I10]. ◀

6.4.5 Queries on graphs with attributes

Many SIGINT graphs have some form of attributes associated with vertices and edges, for example the location of a phone. It can be useful to answer queries where we restrict ourselves to vertices with a particular value for some attribute. Is it possible to modify your algorithm to enable queries on vertices with particular attributes? ◀

6.5 Relevant Data

Any streaming graphical data is suitable for these problems, giving a variety of options. Examples include HRMap, telephony, email, SQUEAL alerts and IP flow metadata. All provide a stream of events with some notion of a source and destination vertex, along with the timestamp of the event.

In addition we have various reference datasets. For example section F.3.1 describes a database of websites of interest to counter terrorism and BROAD OAK lists known target phone numbers and email addresses (see section F.3.2). These could be used to identify if an extracted graph structure has a higher density of targets than would be expected.

The idea of wanting to process a stream of edges is not specific to the intelligence community, and so external collaboration should be possible given a suitable dataset.

6.6 Collaboration Points

There are the following potential collaboration opportunities both within and outside the intelligence community.

KACHINA: Sandia National Lab in the USA has a multi-year effort called KACHINA which includes the Questa project to look at large graph processing for defence analysis. They hold security clearances, and would be an obvious group to collaborate with. Points of contact are [redacted] (NSA employee deployed to Sandia) and [redacted]

[redacted] is engaged both in external research at Georgia Tech and as a researcher in R1 at NSA. His external research in the field includes [E3, E12, E28]. Collaboration should be possible on both classified and unclassified problems.

Pod58 – Cyber Exploration: The Pod runs until the end of January 2012, and aims to use analysis frameworks including DISTILLERY to support analysis of Cyber data. [redacted] s the R1 research lead in the Pod.

NSA/R: Various people around the research division at NSA would be good people with whom to collaborate. Specific names include [redacted] and [redacted] who normally attend the various five-eyes conferences. [redacted] is a GCHQ integree at NSA working on data mining problems including the integration of streaming analysis and MapReduce based analysis.

ICTR-CISA: This team is responsible for streaming analysis research at GCHQ. Much of their work revolves around the platform (DISTILLERY) however they are also active in developing algorithms. [redacted] is the team lead.

IBM Research: As part of the InfoSphere Streams (DISTILLERY) platform, IBM are developing a "Graph Analytics Toolkit". This is in its early phases and there is potential to collaborate on this (at an UNCLASSIFIED level), and potentially have any algorithms developed incorporated into the toolkit. Initial contact can be made through [redacted] in ICTR.

A Ways of working

This section gives a few thoughts on ways of working. The aim is to build on the positive culture already established in the Institute's crypt work. HIMR researchers are given considerable freedom to work in whatever way suits them best, but we hope these ideas will provide a good starting-point.

A.1 Five-eyes collaboration

As on the crypt side, we hope that UKUSA collaboration will be a foundation-stone of the data mining effort at HIMR. This problem book is full of links to related research being carried out by our five-eyes partners, and researchers are very strongly urged to pursue collaborative angles wherever possible—above all, to get to know the people working on the same problems and build direct relationships. Researchers are encouraged to attend and present at community-wide conferences (principally SANAR and ACE), as funding and opportunity allows.

We hope that informal short visits to and from HIMR will also be a normal part of data mining life. HIMR has a tradition of holding short workshops to focus intensively on particular topics, where possible with participation from experts across the five eyes community. Frequently these are held during university vacations, to allow our cleared academic consultants to take part. Each summer, HIMR hosts a **SWAMP**: a two-month long extended workshop on (traditionally) two topics of high importance, similar to the SCAMPs organized by IDA. We hope that HIMR researchers will feel inspired to suggest possible data mining sub-topics for future SWAMPs.

A.2 Knowledge sharing

Inevitably, there is a formal side to reporting results: technical papers, conference talks, code handed over to corporate processing, and so on. But informal dissemination of ideas, results, progress, set-backs and mistakes is also extremely valuable. This is especially true at HIMR, for several reasons.

- There is a high turnover of people, and it is important that a researcher's ideas (even the half-baked ones) don't leave with him or her.
- Academic consultants form an important part of the research effort: they may only have access to classified spaces a few times a year for a few days at a time, so being able to catch up quickly with what's happened since their last visit is crucial to help them make the most of their time working with us.
- HIMR is physically detached from the rest of GCHQ, and it's important to have as many channels of communication as possible—preferably bidirectional!—so that this detachment doesn't become isolation. The same goes even more so for second party partners as well.

In HIMR's METEOR SHOWER work, knowledge sharing is now primarily accomplished through two compartmented wikis hosted by CCR Princeton. For data mining, there should be more flexibility, since almost none of the methods and results produced will be ECI, and

in fact they will usually be STRAP1 or lower. Paradoxically, however, the fact that work can be more widely shared can mean that there is less of a feeling of a community of interest with whom one particularly aims to share it: witness the fact that there is no shining model of data mining knowledge sharing elsewhere in the community for HIMR to copy!

We suggest that as far as possible, data miners at HIMR build up a set of pages on GCWiki (which can then be read and edited by all five-eyes partners) in a similar way to how crypt research is recorded on the CCR wikis. They can then encourage contacts at GCHQ and elsewhere to watch, edit and comment on relevant pages. In particular, the practice of holding regular *bull sessions*[10] and taking live wiki notes during them is highly recommended.

If any researchers feel so inclined, GCBlog and the other collaborative tools on GCWeb are available, and quite suitable for all STRAP1 work. For informal communications with people from MCR and ICTR, there is a chat-room called `himr_dm`: anyone involved in the HIMR data mining effort can keep this open in the background day by day. There is also a `distillery` room that is sadly under-used: in principle, it discusses SPL and the corporate DISTILLERY installations.

For any STRAP2 work that comes along, there are currently no good collaborative options: creating an email distribution list would be one possibility.

A.3 Academic engagement

The first test for HIMR's classified work must be its applicability and usefulness for SIGINT, but given that constraint, GCHQ is keen to encourage HIMR researchers to build relationships and collaborate with academic data miners, and publish their results in the open literature. Of course, security and policy will impose some red lines on what exactly is possible, but the basic principle is that when it comes to data mining, SIGINT data is sensitive, but generally-applicable techniques used to analyse that data often are not. Just about everyone nowadays, whether they are in academia, industry or government, has to deal with big data, and by and large they all want to do the same things to it: count it, classify it and cluster it. If researchers develop a new technique that can be published in an open journal once references to SIGINT are excised, and after doing a small amount of extra work to collect results from applying it to an open source dataset too, then this should be a win-win situation: the researcher adds to his or her publication tally, and HIMR builds a reputation for data mining excellence.

Of course, there may be occasions when publication is not appropriate, for example where a problem comes from a very specific SIGINT situation with no plausible unclassified analogy. Day-to-day contact with the Deputy Director at HIMR should flag up cases like this early on. There are also cases where we feel we have an algorithmic advantage over the outside that is worth trying to maintain, and this can be further complicated if equity from other partners is involved, or if a technique brings in ideas from areas like crypt where strict secrecy is the norm. The Deputy Director should be consulted before discussing anything that might be classified in a non-secure setting: he or she can further refer the question to Ops Policy if necessary. Over

[10] Informal meetings at blackboards where people briefly describe work they have been doing and problems they have encountered, with accompanying discussion from others in the room. The rules: people who wish to speak bid the number of minutes they need (including time for questions). Talks are ordered from low to high bid, with ties broken arbitrarily. You can ask questions at any time. You can leave at any time. If you manage to take the chalk from the speaker, you can give the talk.

time, researchers will build up a good idea of what is sensitive and what is not, but in the first instance, erring on the side of caution is a sound starting point where classified information is involved.

Similarly, if there are grey areas about when work should count as part of a researcher's classified or unclassified effort, this can be settled by an informal conversation with the HIMR Director or Deputy Director.

B DISTILLERY

DISTILLERY is a project to deliver a platform for near-real-time streaming analytics. It is a research partnership between NSA and IBM Research, with GCHQ also having been involved for a number of years. DISTILLERY was released by IBM as a commercial product in 2010 as *IBM InfoSphere Streams*, often shortened to just *Streams*. We use the three terms synonymously. For more on the DISTILLERY platform, and links to plenty of other useful pages see [W11].

Central to the DISTILLERY platform is the stream processing paradigm. We use the terminology of the Streams documentation. A Streams application is made up of one or more *composite operators*. A composite operator contains one or more *operators*, each of which has zero or more *input ports* and zero or more *output ports*. Data takes the form of *tuples* conforming to a *schema*, where the schema defines the names and types of the entries which make up a tuple. Streams of tuples flow along the edges of the flow graph between the operator ports and the operators carry out some kind of transformation on these streams. When built and launched into the Streams platform, operators are placed in a series of *processing elements* or PEs connected according to the application flow graph. Each PE contains one or more operators (by default exactly one, but we can combine multiple operators into a single PE for efficiency).

Crucially, we can process data as it arrives. If we know in advance what questions we would like to ask of the data then we may never need to store the data. Instead, we build a processing flow to answer the question on the stream of data. Our output is a stream of answers. As well as saving storage, processing data provides other advantages. An obvious one is near-real-time tipping. Given some event of interest, we can alert an analyst as soon as we observe that event. We can typically provide a tip-off within a second of the event occurring, although the latency of the analyst is somewhat higher. However we do not restrict ourselves to tipping a human. Observing an event might cause us to take some other action, for example collecting more detailed data for identifiers that appear in the initial event.

Streams applications are written in SPL, the *Streams Processing Language* [W40], and are run on InfoSphere Streams version 2. Older applications were written in SPADE and run on InfoSphere Streams version 1. We are currently converting our applications from SPADE to SPL, and we plan for most new applications to be written in SPL.

B.1 When would I use InfoSphere Streams?

Stream based processing is useful any time where you want to produce results as soon as possible after the relevant events occur or when we cannot reasonably store all the data required for a problem. In these situations we can use Streams to handle the plumbing between our operators. It provides parallelism over multiple hosts in a cluster whilst also providing some resiliency against system failures. Through the use of Import and Export operators an application can be developed and deployed in stages, and data streams can be shared with other users.

Import operators will also allow you to take advantage of the data streams already available on the cluster. In this case, someone else will already have arranged for the feed to be delivered from our front-end collection systems, and your application need not be concerned with format changes.

B.2 Documentation and Training

Documentation is linked from [redacted] and [redacted] Further documentation can be found on a Streams cluster at [redacted]

The SPL documentation consists of:

- SPL Introductory Tutorial: start here after following the instructions on Getting Started below.
- SPL Language Specification: describes the language itself.
- SPL Standard Toolkit Reference: describes the operators provided in Streams.
- SPL Standard Toolkit Types and Function: describes the built in functions.
- SPL Config Reference: covers additional configuration options which allow you to alter the behaviour of operators or the runtime platform.
- SPL Compiler Usage Reference: describes the many compiler options in detail.
- SPL Operator Model Reference: the information you need to write a new operator.
- SPL Streams Debugger Reference: describes how to use the debugger.
- Studio Installation and User's Guide: describes the Eclipse development tools available to help you write SPL.
- Installation and Administration Guide: covers how to install Streams and to configure a Streams instance.

Training may be available from IBM UK organised through QA – contact [redacted] or details of upcoming courses. IBM, NSA and GCHQ have published a paper on design principles which may be useful [E43].

B.3 Logging on and Getting Started

Access to a DISTILLERY cluster is via ssh, and you will initially use the BHDIST cluster (see below). Use one of b[redacted] Once you log on for the first time you should go through the steps listed on the getting started GCWiki page [W16], although the following steps should be sufficient to get you started.

Configure key based ssh access (accept the defaults presented by ssh-keygen):

```
ssh-keygen -t dsa
cd ~/.ssh
cat id_dsa.pub > authorized_keys
chmod 600 *
```

You should then be able to type 'ssh localhost' and it won't ask for a password. This is essential as DISTILLERY uses ssh to launch commands and processes on all nodes (even when running only on localhost).

Add the following to your .bashrc file:

```
# Source global definitions
if [ -f /etc/bashrc ]; then
    . /etc/bashrc
fi

umask 0027
export JAVA_HOME=/opt/ibm/java-x86_64-60/
export ECLIPSE_HOME=/opt/eclipse-spl
export PATH=$JAVA_HOME/bin:opt/eclipse-spl:$PATH

export STREAMS_SPLPATH=/opt/distillery/toolkits
source /[redacted]
```

Configure a streams public/private keypair (to avoid needing a password to stop/start jobs) with:

```
streamtool genkey
```

Set up a hostfile to tell Streams which hosts to use. The hosts file is in [redacted]e and for now should contain a single line with the host you're using but in the blackhole.net domain as this uses a faster network switch, e.g.

```
[redacted]
```

You should now be able to follow the SPL Introductory Tutorial linked from [redacted].

For using the Eclipse tools, including the ability to view your jobs in a flow graph, then use eclipse with /[redacted] This should be in your path if you followed the instructions above. Figure 5 shows an example of multiple jobs connected together in a shared DISTILLERY instance, as seen through the Streams Live Graph view in Eclipse.

Many people choose to run a VNC session on the cluster to provide a desktop environment. For instructions see the DISTILLERY pages on GCWiki.

The two main clusters used for research work are listed in table 2. To get an account on either contact [redacted]

B.4 Data

Data typically arrives into a DISTILLERY cluster via either a UDP or TCP socket from our front-end processing systems. UDP is used where we need to avoid delays in our processing causing delays earlier in the processing chain – instead we just drop the extra data. We are moving to using TCP and then using a threaded port on the next operator so we can measure our data losses (see section B.5.1 on page 58).

Your home directory is shared over the cluster, as is /[redacted]. Applications need to be run from a shared location so all nodes can access them. Results should be saved to [redacted] rather than your home directory as the filesystem is local to the cluster.

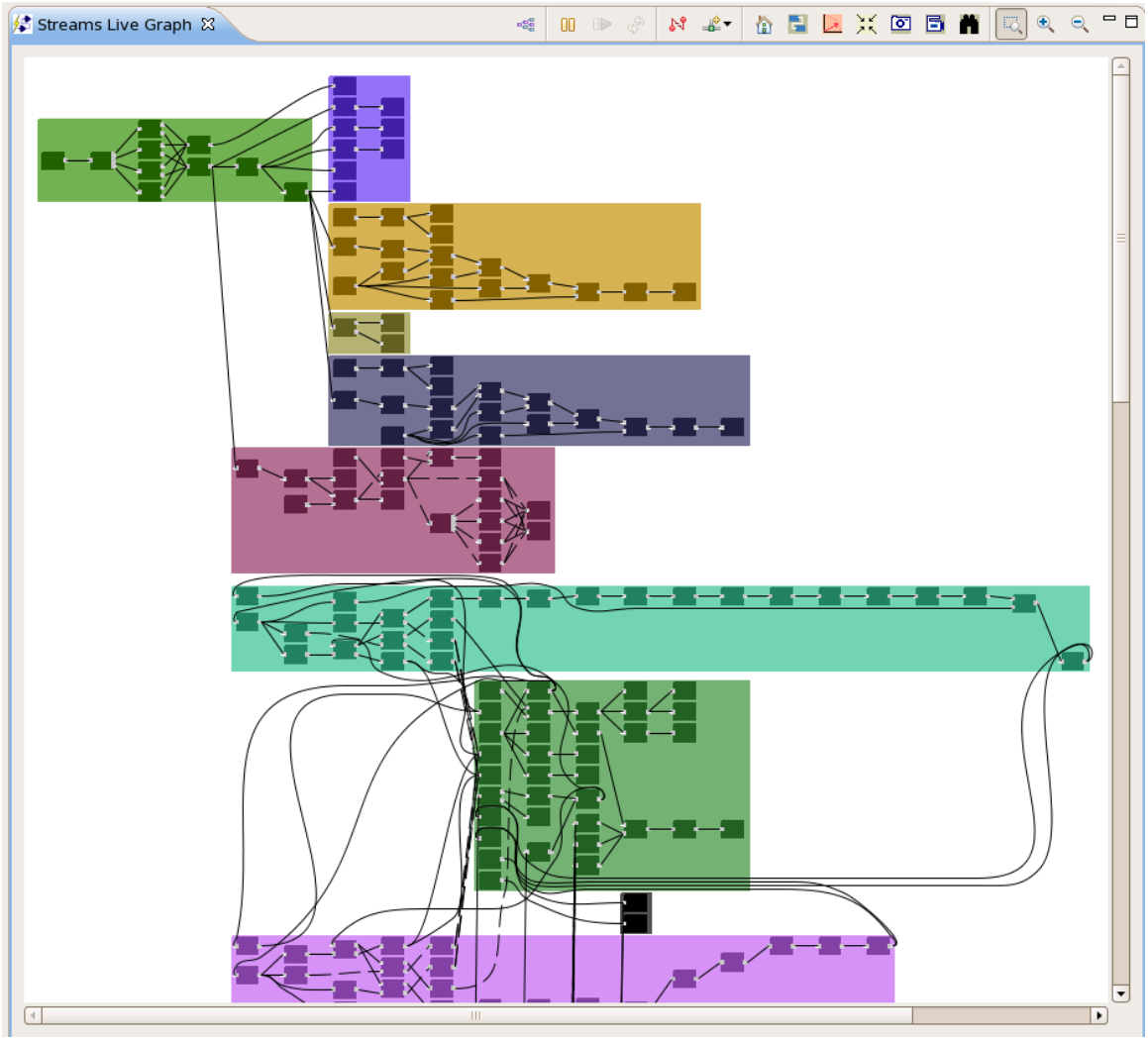


Figure 5: Streams Live Graph view: interlinked jobs in the AHS EXPLORE DISTILLERY cluster.

Node names	Number of nodes	Account management	Purpose
[redacted]	10	ICTR	Development and operational prototypes. Data from ICTR research probes.
[redacted]	3	AHS	Development and "Explore" prototypes. Data from MVR and mailorder.

Table 2: DISTILLERY clusters available for use.

B.5 Conventions

The DISTILLERY clusters contain operational prototypes as well as development code, so it is important to consider the impact on others when running jobs.

When running jobs on the live data feeds, try to do initial processing and data reduction on the same host as the data import to avoid unnecessary network use. Avoid causing back pressure to the live feeds – see section B.5.1. Ideally you should test your code on a small sample before connecting to the live data feeds, although this may not always be possible.

B.5.1 Use threaded ports on shared data

If the incoming data rate is faster than you can process then by default you will cause the incoming data to slow down, causing *back pressure*. If you are reading from a shared data stream then this affects everyone reading from that stream – all processing will be slowed down. This may cause data to be lost further up the chain, for example at the point where it is received from the front-end probes.

To avoid causing this problem, you should normally configure the first operator of a job to drop tuples if it has too many already waiting to be processed. Typically this would be an `Import()` operator, which is configured as follows:

```
stream<SomeTupleType> I1 = Import() {
  param subscription : DataFeed == "SomeData";
}
stream<myschema> I2 = Functor(I1) {
  config threadedPort : queue(I1, Sys.DropLast);
}
```

The queue function has an optional third parameter which specifies the buffer size (in tuples), and the second option can be replaced by `Sys.DropFirst`.

When reading from a file then you should not set such a buffering configuration, since you want to read the data as fast as you can process it but without discarding any tuples.

B.5.2 Operator Toolkits and Namespaces

SPL (DISTILLERY) code is stored in toolkits. These split into two broad types – tookits of operators and toolkits containing applications. The Five Eyes repositories of toolkits are held in MadForge and are described at [redacted]. When we wish to share our operators (typically once they are tried and tested) then we will add them to MadForge. Before that we store the code in a Git repository on `http://github.ar.gchq`, with the repository name matching the toolkit name but prefixed with "spl-". Instructions for creating a new repository can be found at [redacted]

Most of the repositories get built at least once a week and deployed to the cluster. To add a repository to the build list contact [redacted] or [redacted] To have new versions of your toolkit be automatically deployed you must ensure that you increment the toolkit version number. Toolkits are installed to `/opt/distillery/toolkits` and can then be

used in your applications. Rather than hard-code this path we put it into the `STREAMS_SPLPATH` environment variable.

Toolkits containing operators are placed in the `gchq.*` namespace, for example `gchq.ingest` contains the TcpLineReader operator for multi-threaded reading of SIGINT data. Application toolkits are in the `gchq.app.*` namespace to differentiate them. These are not installed into `/opt/distillery/toolkits` but are instead checkout out into `/streams/apps` if you want to run them.

One particularly important toolkit is [redacted]. Despite the name, this is in fact installed into [redacted] and contains the schemas for the data available in the cluster. This is needed when importing data into your application. As an example, HRMap data matches the `HRMapRecord` schema. Details for all the datasets described in section F can be found in [redacted]

B.6 Further help and resources

The DISTILLERY team in ICTR-CISA are the best points of contact for questions. [redacted] is the team lead and can cover most types of issue. [redacted] is the best contact for infrastructure issues. In OPC-MCR the best contact for DISTILLERY questions is [redacted]

There is a `distillery` room on the instant messaging server (accessed using Pidgin, see appendix D). This can be used to ask questions on SPADE, SPL, and the infrastructure. Although the ICTR team do not make much use of it at present, there is normally someone there who can help.

All relevant resources, are linked from [redacted]

C Hadoop

Hadoop is a software framework that supports static data-intensive distributed applications. Its design is heavily based on that of Google's infrastructure as disclosed in [E10]. It is designed to be scalable rather than fast and efficient. This means that for any given task there is likely to exist a more efficient solution. However the ease of parallelism more than counteracts this in most cases. In this model of parallelism the computation is shipped to the data, rather than data to computation, therefore saving large amounts of network traffic. Typically Hadoop is installed across a *cluster* of computers, which are often referred to as *clouds*. Indeed the only reason to install it on a single computer is for testing purposes.

Hadoop consists of two main components, the Hadoop Distributed File System (HDFS) and MapReduce. As a user it should not be necessary to worry about how the file system is implemented. Instead one can consider it to act just like a very large filesystem. However should one wish to use Hadoop to process data then knowledge of MapReduce is required. Fortunately the key concepts of MapReduce are simple and easily understood. As the name suggests there are two stages to any MapReduce job—a *Map* and a *Reduce*. In the map stage one receives successive input records. For each input one produces zero, one or many output records in the form of *key-value* pairs. These output records then go into a *shuffle* phase, in which all records are sorted so that the reduce stage receives all records with a common key together. This *reduce group* is then processed together and again zero, one or many output records may be produced. As the entire output of the Mapper is being sorted it is possible to perform a *secondary sort* to provide data to the Reducer in an advantageous order. This is done by specifying that grouping should only consider part of the key, whereas ordering should consider all of it. A common use of this is to provide time ordered data in a reducer for a particular identifier.

The Hadoop framework is written in Java. Java is therefore a popular choice for writing Hadoop MapReduce applications. Using Java one has access to the full functionality of Hadoop and is recommended for sustainable code. However it is not necessary to know any Java to get MapReduce jobs running on Hadoop using the *Streaming* package. Streaming is invoked from the command line on a Hadoop node. Any script or program that accepts data on STDIN and outputs it to STDOUT can be specified as a Mapper or Reducer. This significantly lowers the entry barrier and is ideal for quickly trying out ideas where the full Java treatment seems like overkill.

C.1 When would I use Hadoop?

The short answer is whenever you want to batch process a large amount of static data. There is not really any other option for such computations within GCHQ.

A slightly longer answer is that Hadoop clusters are where GCHQ has chosen to keep its bulk events data. This is due to the large amount of data processing power Hadoop offers. With hundreds of hard disks working simultaneously multiple gigabytes can be read per second. This allows the processing of the multi-terabyte datasets we intercept. By having the data in its raw state it is possible to ask a huge number of different questions of it. This can be contrasted with the QFDs which also store very large amounts of data, but are databases optimised for a specific type of analyst queries. The QFDs therefore do not offer a sensible data mining

platform.

Hadoop generally excels when your algorithm can be expressed in a small number of MapReduce steps. It is less efficient when implementing iterative algorithms. This is because between iterations all state must be written down to disk and then read back in again. This extra I/O cost can easily end up swamping the time taken to perform the computations. Sometimes there may be no other way of performing an algorithm given the size of the data and the only solution is patience[11].

C.2 Documentation and Training

The standard Hadoop documentation is available linked from [redacted] This consists of:

- A MapReduce Tutorial that shows you how to write MapReduce applications in Java.
- An introduction to Hadoop Streaming. Although not a full tutorial all the information you need to run Streaming jobs is there.
- An overview of the Hadoop command line arguments.
- The Java documentation of the Hadoop API. If writing Hadoop in Java this is extremely useful.

The Hadoop page on GCWiki [W20] has many resources, including:

- 6 lectures and 2 exercises from Cloudera, a Hadoop consultancy company.
- An overview of Hadoop by IBM's Jimeng Sun.

Tom White's book *Hadoop: The Definitive Guide* is probably the best book currently available on Hadoop. It is also available on NSA's Safari book library [W35].

Classroom based training should also be available. TDB have organised internal training led by GCHQ employees. Some people have also attended a multi-day training course offered by Cloudera.

C.3 Logging on and Getting Started

Access to Hadoop clusters is via ssh. You will ssh to an *edge* node. These are not part of the compute cluster but do allow you to submit jobs and interact with HDFS.

Instructions for accessing SUN STORM, the largest cluster, are available at [redacted] The other clusters are detailed in table 3.

Some useful aliases for your `.bashrc` are given below. Adding these will save you a huge amount of typing and make interacting with HDFS feel more like using a regular filesystem.

[11] ICTR-DMR are currently developing *Bagel* an implementation of Google's Pregel distributed graph mining solution. While still in its early stages Bagel keeps its state in memory and therefore avoids this extra I/O cost between steps.

Cluster name	Num nodes	Purpose
SUN STORM	897	Cheltenham events cluster
GOLD MINE	125	Cyber/content cluster
HAGER AWEL	800	Bude events cluster
Woody	133	ICTR research cluster
Buzz	42	ICTR research cluster

Table 3: GCHQ's Hadoop clusters

```
export HADOOP_HOME=/opt/hadoop/current
alias hadoop=${HADOOP_HOME}/bin/hadoop
alias hstream='hadoop jar $HADOOP_HOME/contrib/streaming/hadoop-streaming-0.20.10.jar'
alias hl='hadoop fs -ls'
alias hjobl='hadoop job -list'
alias hjobk='hadoop job -kill'
alias hjob='hadoop job'
alias hjar='hadoop jar'
alias hc='hadoop fs -count'
alias hput='hadoop fs -put'
alias hget='hadoop fs -get'
alias hf='hadoop fs'
alias hcat='hadoop fs -cat'
alias hdu='hadoop fs -du'
export TMOUT=36000000
```

C.4 Data

There are a large number of datasets available on the corporate clusters. These typically each occupy a subdirectory under `data`. The datasets on SUNSTORM are listed at [redacted] has equivalent datasets containing data processed at Bude rather than Cheltenham.

C.5 Conventions and restrictions

The three corporate clusters are all configured similarly. This subsection refers to their configurations—for the research clusters all bets are off and ICTR-DMR should advise you of any restrictions should you gain access.

C.5.1 Scheduler

The clusters all have the *Fair Scheduler* installed [W19]. This replaces the vanilla FIFO that Hadoop has installed by default. Fair scheduling is a method of assigning resources to jobs such that all jobs get, on average, an equal share of resources over time. When there is a single job running, that job uses the entire cluster. When other jobs are submitted, task slots that free up are assigned to the new jobs, so that each job gets roughly the same amount of CPU time.

Each user (and special processing users) are assigned their own *pool*. The scheduler tries to give each pool an equal amount of time on the cluster. Processing users also have a minimum number of map and reduce slots below which they will not drop if they request them. Further each user is restricted to having a single concurrent running job.

C.5.2 HDFS /user/yoursid space

On logging into a corporate cluster you will have a HDFS home directory created at `/user/yoursid`. This is where the results of your Hadoop jobs will end up by default. That is, if you don't specify an absolute path,it will be taken relative to your home directory. Your home directory has a size limit on it (believed to be 2TB). If you need more space than this then you should contact the cluster administrators to find a solution.

C.6 Running Hadoop on the LID

It is discouraged to use either SUN STORM or HAGER AWEL for developing code as they are both somewhat production systems. It is therefore a good idea to iron out bugs elsewhere if possible to ensure your code will not bring the cluster down. The easiest way to do this is probably on a pseudo-distributed Hadoop installation, following the instructions given in the standard Hadoop documentation. If you wish to do this on the LID then you need to do slightly more to get around issues with localhost not always being the same depending which box you are on. Following these instructions should give you working Hadoop instance. If the standard ports are already in use then more configuration properties need to be added. At this point it's probably best to either try another machine or ask for some advice.

1. Make sure you can execute a passwordless ssh to your machine. This must be done using the machine's hostname, *not* localhost. This is because there are multiple different LID servers, each with a different idea of what localhost is. By adding one machine's localhost to the `known_hosts` file you will cause yourself problems. If passwordless ssh does not work execute the following commands.

```
ssh-keygen -t rsa -P '' -f ~/.ssh/id_rsa
cat ~/.ssh/id_rsa.pub >> ~/.ssh/authorized_keys
```

2. Choose a directory in which to install Hadoop. This should be somewhere visible from all LID machines. Following shell scripting we will refer to this as `$HADOOP_HOME`. In fact you might want to put the following into your `.userprofile` along with the other aliases given previously.

```
export HADOOP_HOME=/path/to/Hadoop/dir/
```

3. Untar the hadoop tarball into `$HADOOP_HOME`.

4. In `$HADOOP_HOME/conf/hadoop-env.sh` add the line

```
export JAVA_HOME=/usr/lib/jvm/java-1.6.0/
```

5. Make the directories `$HADOOP_HOME/data` and `$HADOOP_HOME/name`

6. In `$HADOOP_HOME/conf/hdfs-site.xml` add the entries

```
<configuration>
  <property>
    <name>dfs.replication</name>
    <value>1</value>
  </property>

  <property>
    <name>dfs.data.dir</name>
    <value>HADOOP_HOME/data</value>
  </property>

  <property>
    <name>dfs.name.dir</name>
    <value>HADOOP_HOME/name</value>
  </property>
</configuration>
```

   Where HADOOP_HOME is replaced with the Hadoop home directory. Using shell variables won't work here as the configuration files are read verbatim.

7. In `$HADOOP_HOME/conf/mapred-site.xml` add the entries

```
<configuration>
  <property>
    <name>mapred.job.tracker</name>
    <value>HOSTNAME:9001</value>
 </property>
</configuration>
```

   Where HOSTNAME is replaced with the hostname of the machine you are on.

8. In `$HADOOP_HOME/conf/core-site.xml` add the entries

```
<configuration>
  <property>
    <name>fs.default.name</name>
    <value>HOSTNAME:9000</value>
  </property>
</configuration>
```

   Where HOSTNAME is replaced with the hostname of the machine you are on.

9. In `$HADOOP_HOME/conf/masters` and `$HADOOP_HOME/conf/slaves` replace `localhost` with the hostname of the machine you are on.

10. Run `$HADOOP_HOME/bin/hadoop namenode -format` to format the namenode.

11. Run `$HADOOP_HOME/bin/start-all.sh` to start the Hadoop daemons.

12. To check this has worked OK go to `http://HOSTNAME:50070/` to check on the status of the namenode and `http://HOSTNAME:50030/` for the jobtracker.

13. Now you can try to run a toy Hadoop job. Copy the input files into the distributed filesystem: `$HADOOP_HOME/bin/hadoop fs -put conf input`. Now run some of the examples provided: `$HADOOP_HOME/bin/hadoop jar hadoop-*-examples.jar grep input output 'dfs[a-z.]+'`.

When you log out of this LID session the Hadoop daemons will be killed by the logoff script. You will therefore need to restart them in your next LID session. However each time you log into the LID you cannot guarantee which machine you will be allocated. If you are allocated a different machine to that where you installed Hadoop you will not be able to directly restart it. Instead you will need to do so over ssh:

```
ssh HOSTNAME 'HADOOP_HOME/bin/start-all.sh'
```

Again HOSTNAME is the machine on which you originally installed Hadoop. You can then submit jobs and interact with HDFS from any LID machine, i.e. including the one you currently have a session on. Hadoop will then carry on running until the end of the next session you are assigned on the machine on which you installed it.

C.7 Further help and resources

In OPC-MCR [redacted] is the best contact for Hadoop questions, [redacted] can also offer advice, particularly on Streaming. Outside of MCR there is a large community of Hadoop users and administrators. The best way to contact this community is probably through the `rough_diamond` chatroom on the Jabber server.

There are a large number of resources available on GCWiki. Some highlights, in no particular order:

- [redacted]`_(Work_Package)`: The main page for SILVER LINING, the work package within TDB that provides Hadoop clusters. It links to many places and *may* stay more up to date than this document.
- [redacted]`_-_User_Guide`: An initial user guide for the SUN STORM cluster. However many of the tips hold in general across all Hadoop clusters.
- [redacted]`_-_Streaming_interface`: A short guide to user Hadoop Streaming with code examples to run on data on SUN STORM.
- [redacted]: SILVER LIBRARY is a library of Hadoop parsers, writables and other utility classes to simplify development of MapReduce analytics in Java. This pages describes at least some of it. Links to *Utilities* and *Search* are on the right hand side. The library is strongly recommded for Java MapReduce on the corporate clusters as it has built in parsers that save users having to understand how the events are structured.

D Other computing resources

There are various computing options available to HIMR researchers beyond the bulk data sources of Hadoop and DISTILLERY. There is expertise in these environments at HIMR so we only briefly document these options here. Further information can be found at [W22, W21].

Firstly researchers have access to the Microsoft Windows environment of **VALHALLA**. VALHALLA is the standard desktop and provides email, Microsoft Office, web browsing, instant messaging and a gateway to other systems. The /data/himr_dm/ filesystem should be accessible in Windows with VALHALLA (at the time of writing the Windows mount location is not known).

Instant messaging is accessible via the Pidgin application. Many employees of GCHQ can be found online both for direct messaging and in chat rooms. The following chat rooms are of particular note: himr_dm (HIMR data mining research), distillery (DISTILLERY users), rough_diamond (Hadoop users), hecsupport (compute clusters queries) and lid_support (LID support). Instructions for getting going on Pidgin can be found at [W29].

From VALHALLA researchers can access **DISCOVER**. This is GCHQ's document repository. Literature for this research task has been filed at DISCOVER 10499535. Other sources of information that are of particular note are the the collaborative GCWiki [W15], which contains information about many GCHQ activities, and the Safari online bookshelf [W35], which provides electronic versions of many technical books.

The primary interactive data analysis environment will be the Linux Interactive Desktop (**LID**). The LID provides a remote desktop onto a RedHat Linux box. Various mathematical tools such as R, Matlab, Mathematica, Maple, Sage and Magma are available. Scripting languages are available: Perl is the most commonly used scripting language in GCHQ but Python is starting to gain traction. Compilers are also available for C, C++ and Fortran. It is worth noting that GCHQ have imported the general repository for R packages, CRAN, at [W9] and implementations of many machine learning techniques can be found there.

There are two Linux compute clusters available. **MOUNT MCKINLEY** is probably the machine of choice and has 652 compute nodes each with 8 cores, giving a total of 5216 cores. The cores are clocked at 2.4GHz. Each node has 32GB of RAM and there is a fast interconnect between nodes. MOUNT MCKINLEY can be accessed from VALHALLA. The catch with MOUNT MCKINLEY is there are few user tools available and hence it should primarily be seen as a place to run compiled code (Perl and Python scripting is also available). MOUNT MCKINLEY is also used for operational processing so researchers will need to abide by conventions around HIMR's use. An older compute cluster called **SEPANG** is also available but is expected to be decommissioned shortly. SEPANG is firewalled from the rest of the GCHQ network and does not have easy access to any of the data sources described; however it does have a wide range of user tools installed and is reserved for HIMR's sole use.

Both the LID and MOUNT MCKINLEY user nodes mount [redacted] If you want to analyse data from Hadoop or DISTILLERY on LID or MOUNT MCKINLEY then you will need to transfer the data with scp.

E Legalities

This appendix is intended as a brief guide to the legal information of most relevance to your work at HIMR. However [W25] and your legalities training should be treated as the definitive references.

E.1 Overview

GCHQ always complies with UK law[12]. In particular we are bound by the Regulation of Investigatory Powers Act (RIPA) and Intelligence Services Act (ISA). RIPA requires GCHQ to have arrangements in place to minimise its retention and dissemination of intercepted material. RIPA also applies specific protection to the communications of people in the UK. ISA requires GCHQ to have arrangements in place to ensure that it obtains or discloses information only in the proper discharge of its functions or for the purpose of any criminal proceedings.

The complete and official compliance guide can be found in [W25]. In general we must be able to demonstrate that our actions are both necessary and proportionate. We show that our actions are necessary and proportionate by producing an accountable record for oversight and audit. This typically takes the form of an HRA (Human Rights Act) justification. The Human Rights Acts defines the basic rights everyone must have respected. In particular there is a right to privacy which can only be violated "in the interests of national security, public safety or the economic well-being of the country, for the prevention of disorder or crime, for the protection of health and morals, or for the protection of the rights and freedoms of others." For GCHQ this means we must justify our activities as being in the interests of *national security*, the *economic well-being* of the UK, or in support of the prevention or detection of *serious crime*.

You should bear in mind that you have signed and are bound by the Official Secrets Act (OSA). In particular you should take care in discussing or releasing potentially classified data and techniques. If you are unsure on an item's classification then you should seek guidance from the data owner. Our data and information is also exempt from the Freedom of Information Act (FOIA). All documents should carry the same caveat as this document.

As accessing the content of an individual's communications is regarded as more invasive than examining its metadata there are tighter restrictions imposed on such data. Content need not necessarily be an email or phone call. For example, the content of a URI beyond the first slash is considered content, as are the specifics of someone's online mapping activity.

E.2 Procedures

We now highlight some specific procedures that should be followed when working with bulk metadata.

Detailed policy guidance for corporate Hadoop clusters can be found at [W26]. We give the most relevant information here. If you are extracting a dataset or performing analyses in a way which is not expected to target an individual then there is no need to do anything. However if the criteria specify individuals, or behaviours which are sufficiently precise that they apply to only a few individuals, then you will need to complete a manual HRA log [W23]. You do not need to complete this every time that you perform the same extraction, or perform follow-on

[12]The bulk of our work is also compliant with the policies and laws of five-eyes partners.

analysis using similar techniques and for a similar purpose, so long as you write your first log in a way that is just general enough to cover your current work. Further if your analyses specify five-eyes individuals or organisations, or if the query includes data that is designated as content then you will need Sensitive Targeting Approval in addition to completing a manual HRA log.

When completing a manual HRA log the application name should be "SILVER_LINING" if working on a Hadoop cluster. The reason should be "NS" (national security), a JIC priority of "1" and MIRANDA number of "20135" ("Intelligence in support of GCHQ research work intended to maintain and develop general purpose capabilities in the field of target communications in order to be able to meet such intelligence requirements as may be specified now and in the future"). If you are developing new techniques then the query type should be "QFD_DEVELOPMENT", or if selecting data that focuses down to a few individuals then "BULK_EXTRACT".

Queries in DISTILLERY should also be logged using the manual HRA logging service and most of the guidance above applies. Until "DISTILLERY" is added to the list of applications, please use "BLACK_HOLE". The data source should be the source most closely matching the feeds you are using, otherwise use "AD_HOC_EXTERNAL_DATA". In order to complete the "Number of Results Returned" field you will need to submit the log after you have run the query – there is currently no way for you to update a manual HRA log.

F Data

In this appendix we summarise the datasets made available at the outset of this research. Researchers are encouraged to work with GCHQ staff to find other datasets if required.

F.1 SIGINT events

Firstly we describe datasets of raw SIGINT events: typically these are available as live datasets in Hadoop or DISTILLERY.

F.1.1 SALAMANCA

The contents of this dataset are classified TOP SECRET STRAP2 CHORDAL.

GCHQ collects telephone call record events from a wide variety of sources, and these are stored in a database called SALAMANCA [W36]. This data is also fed to the SUN STORM cloud and the BHDIST DISTILLERY cluster (and other DISTILLERY clusters). This data is a relatively low rate feed of user events, around 5000 events per second, and can be viewed as either a directed or undirected graph. It could be used for the streaming EDA and streaming expiring graphs topics as well as feature extraction for payphones for the beyond supervised learning topic.

In general we have better collection of calls where the two sides are in different countries, although for some countries we also have good collection of in-country calls. This means the graph can have some unusual features and it is worth bearing these in mind when examining features of the graph. Some properties of SIGINT collected telephony graphs are discussed in [I33, I34]. A comparison between SIGINT collected call records and billing records is given in [I73].

On SUN STORM the data can be found under [redacted] in folders named by date. The full format is as described in the Interface Control Document [I79] but with an additional field at the end which uniquely identifies the event within SUN STORM.

In DISTILLERY the data is forwarded into the shared SPL instance on the BHDIST cluster by running a `V1TCPSource` in client mode. The resulting stream can be subscribed to using the subscription `DataFeed=="Salamanca" && EventType=="FullCallRecord"`.

The full data contains many attributes, but the relevant ones are the timestamp and callLength along with identifiers. Records will typically have some of dialledNumber, dialledNumberNorm, callerID and callerIDNorm, where the "Norm" versions may have been normalised, for example by adding the country code. The normalised versions are in E.164 format and give the fully qualified number as opposed to the digits actually dialled which could include just the local number.

Some identifiers are specific to mobile telephony, including the IMSI (which is an ID for a SIM card), the IMEI (which is an ID for a mobile phone handset), and the MSISDN (which should match one of dialledNumberNorm and callerIDNorm). To know which side of the call these attributes refer to you must also read the CallDirection attribute which is either "MO" for mobile originated (i.e. the IMSI and IMEI relate to the callerID) or "MT" for mobile terminated (i.e. the IMSI and IMEI relate to the dialledNumber).

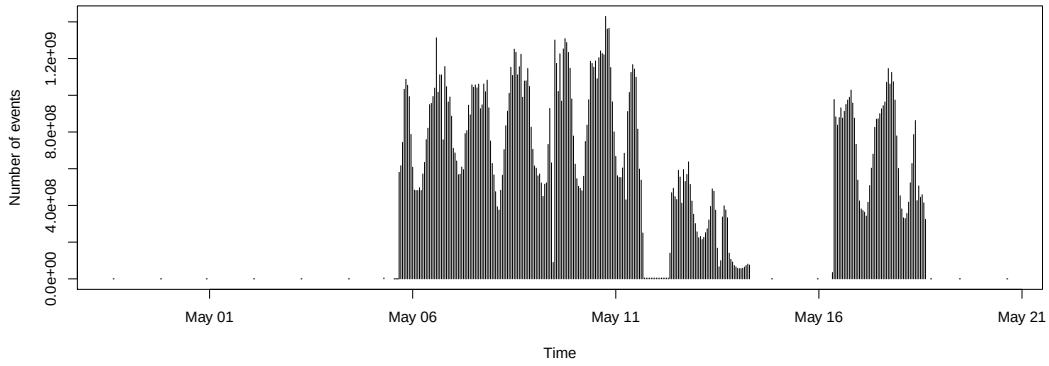


Figure 6: Plot of events counts per hour in our snapshot of FIVE ALIVE data. The period from May 6-11 is clearly the best quality.

F.1.2 FIVE ALIVE

FIVE ALIVE is an ICTR prototype Query Focused Dataset (QFD) providing access to bulk IP-IP connection events, giving a unique unselected view of all activity on SIGINT bearers. Each record in FIVE ALIVE summarises a *flow* between two IP addresses. This summary consists of:

- The start of flow time, unfortunately at second granularity in the static dataset, but microsecond granularity in DISTILLERY.
- The source and destination IPs and ports and the protocol—together these are known as the 5-tuple, hence the name FIVE ALIVE.
- Optionally extra information on flow size and direction depending upon the protocol.

The data format is fully described in [I9].

We have a snapshot of FIVE ALIVE data covering, with gaps, approximately 6-19 May 2011. Figure 6 shows the number of events per hour in this snapshot. This snapshot is available:

- On the GOLD MINE Hadoop cluster at [redacted] in hdfs. The dates on the subdirectories indicate when it was loaded into the cluster and should be ignored.
- On Mount McKinley at [redacted]

There is also a feed of streaming FIVE ALIVE data on the BHDIST DISTILLERY cluster. It is a high-rate feed (around 1 million events a second) and is published in multiple "Splits". They can be imported in the shared instance using the subscription `DataFeed == "FiveAlive" && EventType == "FlowRecord" && Split == "N"` where N ranges from 0 to 11 (but this may be increased to accommodate additional bearers). Don't leave out the Split condition or you'll get all the data in one feed. Ideally you should run your initial processing on the same host as the data is published to reduce network load. This data could also be made available at Bude to provide a multi-site high-rate feed.

F.1.3 HRMap

The contents of this dataset are classified TOP SECRET STRAP2 CHORDAL.

When a user requests a webpage from the internet, this is observed in SIGINT as an HTTP GET request. As well as the page requested it often contains the URL of the previously viewed page. The hostname of the requested page is the "HOST" and the hostname of the previous page is the "REFERRER". When we consider just the hostnames rather than the full URI then this is considered events data. This can be viewed as a directed graph of hostnames, and is given the name HRMap at GCHQ. It is a moderately high rate stream (around 20000 events per second) which should be suitable for the streaming EDA and streaming expiring graphs topics.

Since many web pages point to other web pages on the same server, a large proportion of HRMap events have the hostname matching the referrer. Many records will have no referrer. This happens if the user typed the URL, uses a bookmark, or has configured their browser not to send the referrer attribute.

As well as the host and referrer, an HRMap record also contains a timestamp (in seconds), the client IP address, the client port, and the client HTTP header fingerprint (HHFP) which is a hash of various headers sent by the client and can be used to approximately distinguish clients behind a gateway [I38].

HRMap data is available in DISTILLERY on the bhdist cluster. It can be imported in the shared instance using the subscription `DataFeed == "HRMap" && EventType == "HostReferrer"`. HRMap could also be made available at Bude to give a multi-site streaming graph

Static HRMap is available on the SUN STORM Hadoop cluster at [redacted] and [redacted] Now that the HAGEL AWEL Hadoop cluster at Bude is operational, data collected there will no longer be loaded onto SUN STORM. Data collected at Bude now instead is loaded onto HAGER AWEL at [redacted] The Bude data at Cheltenham will gradually age off and be deleted 6 months after its load date.

F.1.4 SKB

The contents of this dataset are classified TOP SECRET STRAP2 CHORDAL UKEO.

The Signature Knowledge Base is a system for tracking file transfers made on the internet. A record is made each time we see certain file types being transferred. Each file is identified by its format and a hash of some of its content. Whilst this does mean we can store the data, hash collisions are inevitable. Therefore one cannot guarantee that all records referring to the same hash are in fact the same file. Further we only process a small number of different file formats. The dictionary of which file types are logged is given in [I86].

Each single line record in the SKB dataset has the format:

```
date time src_IP dst_IP frag_# IP_ID len protocol_# src_port dst_port seq_#
ack_# file_offset file_type file_signature src_geo dst_geo
```

e.g.

```
22:59:58 03-08-2011 192.168.2.1 10.0.0.1 16384 45872 1398 6 80 53302 4032239316
4106241239 256 SWF-Compressed-V9 [redacted] Geo-IP-Src 32
55.0436;37.3378;MOSCOW;RU;5MMM Geo-IP-Dst 25 40.4;-3.68;MADRID;ES;7LMH
```

For more details on how the logging is performed and the hashes calculated see [I67, I22].

The SKB data is stored both in a QFD for analysts to query and in BLACKHOLE. We have an extract of 1 week on SKB data at [redacted] If you require more data then it is possible to extract some using the blacktools interface.

F.1.5 Arrival Processes

The contents of this dataset are classified SECRET STRAP 2.

There are two standard datasets that have been used to evaluate all approaches to temporal correlation. These are known as the *telephony* and *C2C* datasets. They both consist of records of stochastic processes in the format

```
<name>\t<Number of events>\t<Space separated event times>
```

and files containing pairs of identifiers to be scored in the format

```
<name1> <name2>
```

The data is stored in `/data/himr_dm/data/arrival_processes`.

Telephony data

The original event times were taken from 18 weeks worth of telephony data. These event times were then transformed to give the times in the `.sps` files.

Random pairs of event times are generated this way: choose a pair of distinct originating numbers, A and B; choose from the set $\{1, \ldots, 17\}$; circularly shift the event times of B by δ weeks (i.e. modify the event times of B by adding $\delta \times 604800$ to them modulo $T = 18 \times 604800$, 604800 being the number of seconds in a week). The purpose of the cyclic shift is to reduce the effect of any "random" pairs in which some of A's calls truly cause B to make calls. Shifting by one-week multiples is done to retain the time-of-day and day-of-week structure of the data. If time interval $[0, T)$ can be partitioned into two subintervals, one containing all of the events of A and the other containing all of the events of B, then (A, B) is rejected as a random pair for experimental purposes since presumably no one would consider that they might be correlated. The stochastic processes generated in this way are in the file [redacted]`s`. This file contains 151,811 processes. B's name has δ appended to show the size of the shift. For example if 441242221491 had been shifted 3 weeks it would be called `441242221491.03`.

Causal pairs of event times are generated this way: generate a random pair (A, B); randomly select proportion ρ of A's call initiation times; to each selected initiation time, add the duration of the corresponding call plus a delay drawn from an exponential distribution with mean ϕ seconds; merge the resulting times into B's call initiation times. A proportion of A's calls cause B to make a call, and B makes these calls after the causative call of A ends and after a

random delay. The causal stochastic processes are in files [redacted]s. The standard one used in experiments is [redacted]s. This file contains 155,746 processes. B's name has δ and A's name appended. For example if 441242221491 had been shifted 3 weeks and had a causal dependency on 441242226816 it would be called `441242221491.03.441242226816`.

Both the random and causal pairs for CLASP to score are listed in the file [redacted]-[redacted]

C2C data

The event times were taken from 93 days of C2C presence activity. Records are logged each time the identifier is seen performing an activity. The timestamps in the C2C data are unaltered to provide a realistic test dataset. The stochastic processes are in the file `ip.all.sps.new.sun2`. There are 457,305 processes in this file.

There are also two pairs files. The file [redacted] contains 431,689 random pairs. The file [redacted] contains 45,932 pairs in which the proportion of causal pairs was thought by the data experts to be relatively high, compared to the proportion of causal pairs in the set of all identifier pairs. The exact criteria for making it onto these lists can be found in section 2.4 of [I49]. The names of identifiers consist of the username, a series of dots and dashes and then the identifier type. There are no transformations done to the names in the C2C data.

F.1.6 SOLID INK and FLUID INK

The contents of this dataset are classified SECRET STRAP1.

These are quite old telephony datasets, but we feel it is worth highlighting them to HIMR researchers because the view they offer is so unusual.

SOLID INK is three weeks of telephony events from 2007, as seen from billing records. FLUID INK is an approximate subset of SOLID INK, but as seen via GCHQ's SIGINT collection. Our points of access mean that we mostly collect calls between the target country and the rest of the world; therefore in-country calls are likely to be missing from FLUID INK. Indeed, SOLID INK has 2.7 billion events involving 74 million numbers, while FLUID INK has only 136 million events involving 15 million numbers.

There are also various sources of SIGINT noise which are poorly understood, such as missing calls, duplicate calls, node mislabelling and timing errors. We only have anonymized versions of the datasets available: for legal reasons, we could not retain the unminimized versions this long.

Each INK data set has four fields: timestamp, user-1, user-2 and a number. Unfortunately, the timestamp fields seem to have become corrupted somewhere along the line, and in different ways in each of the datasets. However, timestamp deltas within each set are probably still correct (in seconds). In FLUID INK the call direction is user-1 to user-2, and the fourth field

is call duration in seconds. In SOLID INK the fourth field is a code in $\{1, 2, 3, 4\}$, where:

$$1 = \text{Voice user-1 to user-2}$$
$$2 = \text{SMS user-1 to user-2}$$
$$3 = \text{Voice user-2 to user-1}$$
$$4 = \text{SMS user-2 to user-1}$$

The datasets are available at [redacted] and [redacted]. There are also [redacted] versions, where events involving pizza nodes have been removed.

A very interesting analysis of these datasets came out of the 2008 graph mining SWAMP at HIMR [I73], which revealed just how great the disparity between SIGINT and 'ground truth' can be, for example when it comes to contact chaining. CSEC have also done some work that largely confirms and replicates those results [W4].

F.1.7 Squeal hits

The contents of this dataset are classified TOP SECRET STRAP2 CHORDAL UKEO.

Squeal is a signature-based system for detecting electronic attacks, see [W39]. When a potential attack is detected a hit is forwarded to DISTILLERY. Each hit contains the source and destination IPs and ports, the timestamp, the hit details and geolocation for the IP addresses. By examining multiple hits we may be able to learn about the attacks. For example, we might look for multiple IP addresses that launch attacks in a similar way.

A stream of Squeal hits is initially created on the AHS Explore DISTILLERY cluster, however this is also forwarded to the shared SPL instance on the BHDIST cluster. It can be imported using the subscription `DataFeed=="Squeal" && EventType=="SquealHit"`. This is a low rate stream, around 75 events per second, and contains the hits from all sites.

Squeal hits are available on the SUN STORM Hadoop cluster at `/data/ead`. This covers events collected from all sites.

F.2 Open-source graphs and events

We also provide some open-source graphical and events based data which may be of specific relevance to this research.

F.2.1 Enron

The contents of this dataset are classified UNCLASSIFIED.

Enron was an American energy company that collapsed in 2001 due to massive financial fraud and eventual bankruptcy. After criminal proceedings were completed the complete emails of around 150 Enron employees, mostly senior management, were publicly released. There are approximately half a million emails covering November 1998 to July 2002. There is a brief introduction to this dataset in [E22]. This gives a few summary statistics, such as number of emails per user and conversation thread length. We have a copy of `enron.sql`, the SQL

database file which contains this data. The format of this is not particularly nice so we have also extracted a simpler data set, which hopefully contains the data needed for this work. If not it is not too hard to return to the original file to gather more data. This formatted file is called `enron_transactions.txt`. Each email has one record for each recipient, however each line contains all relevant information. Each record is tab separated with the following fields

```
<time> <timezone> <message_id> <sender> <recipient> <recipient_type> <subject>
```

These data files can be found at [redacted]

F.2.2 US flights data

The contents of this dataset are classified UNCLASSIFIED.

The American Statistical Association's Data Expo '09 asked for analysis of a large dataset of US flight arrivals and departures. The data was made available to the public by the US Department of Transport's research arm, RITA (Research and Innovative Technology Administration), and covers the years 1987 to 2008. The Expo '09 website is mirrored at [redacted] It contains a fuller description of the problem, as well as the winning posters produced by participants in the competition.

When the Home Office decided to fund UKVAC research (see section 5.8.2), it was decided to provide the researchers with two unclassified challenge problems in order to focus their efforts. One was chosen by the HUMINT agencies: to predict the next winners of Nobel prizes. The second came from GCHQ, and was to do further analysis on the RITA flights data. In fact, the second author of this problem book was largely responsible for selecting the problem and framing its statement, so it mirrors very closely the point of view of this problem book, particularly section 5. The flights are meant to be an unclassified proxy for SIGINT events data, and although the dataset can just about be handled in core on modern hardware, UKVAC participants were strongly encouraged to process the data in a stream.

We hope that researchers will be able to compare their approaches, especially on visualization questions, with what the UKVAC comes up with: having a common dataset should help with that. In case any direct collaboration emerges with UKVAC participants, having the dataset they are working on to hand will obviously also be a significant help.

The data consists of 22 bzipped CSV files, one for each year. Each record has 29 fields, described in table 4. Supplemental CSV files describe the codes used for airports, carriers and some individual planes: see the [redacted] page on the Expo '09 mirror.

F.2.3 Wikipedia graph

The contents of this dataset are classified UNCLASSIFIED.

This is not directly relevant for SIGINT, but there are several reasons why it might be handy to have around. Many outside algorithms get tested on this graph, so it might be useful for benchmarking, or as test data to apply algorithms intended for external publication to. It is also a foil for HR map (appendix F.1.3): although that data set does not contain internal clicks between Wikipedia pages (so there is no direct comparison), nonetheless it might be interesting

Index	Name	Description
1	Year	1987–2008
2	Month	1–12
3	DayofMonth	1–31
4	DayOfWeek	1 (Monday) – 7 (Sunday)
5	DepTime	actual departure time (local, hhmm)
6	CRSDepTime	scheduled departure time (local, hhmm)
7	ArrTime	actual arrival time (local, hhmm)
8	CRSArrTime	scheduled arrival time (local, hhmm)
9	UniqueCarrier	unique carrier code
10	FlightNum	flight number
11	TailNum	plane tail number
12	ActualElapsedTime	actual time in minutes
13	CRSElapsedTime	scheduled time in minutes
14	AirTime	air time in minutes
15	ArrDelay	arrival delay, in minutes
16	DepDelay	departure delay, in minutes
17	Origin	origin IATA airport code
18	Dest	destination IATA airport code
19	Distance	in miles
20	TaxiIn	taxi in time, in minutes
21	TaxiOut	taxi out time in minutes
22	Cancelled	was the flight cancelled?
23	CancellationCode	reason for cancellation (A = carrier, B = weather, C = NAS—air traffic control system failure, D = security)
24	Diverted	1 = yes, 0 = no
25	CarrierDelay	in minutes
26	WeatherDelay	in minutes
27	NASDelay	in minutes
28	SecurityDelay	in minutes
29	LateAircraftDelay	in minutes

Table 4: Flights data fields.

to compare in broad terms how algorithms perform on the dynamic HR map of clicks versus a static graph of links.

The data files are in [redacted]. There is a list of vertices, [redacted], which are all the articles on the English Wikipedia at a certain point in 2008. Whenever an article links to another article, there is a corresponding line in [redacted], giving the source and target of the link (as indices into the `.title` file).

F.3 SIGINT reference data

To help researchers enrich their research findings we provide lists of websites of interest and target selectors. We also provide lists of covert infrastructure and known payphones to support research on information flow in graphs and positive-only learning.

F.3.1 Websites of interest

The contents of this dataset are classified TOP SECRET STRAP2 UKEO.

A list of websites of interest is available in a database on [redacted]. These have been manually classified through open source research and contain radical and extremist sites along with many others. These may be useful when examining HRMap data to determine target density.

To get a list of radical and extremist sites, first get a username and password from [redacted]. Then connect to the database and run the query as follows:

```
~db2user/sqllib/bin/db2 connect to DIST1
~db2user/sqllib/bin/db2 "select SITENAME, RADICALISM, Type, URL
       from [redacted]
       where RADICALISM = 'Radical' or RADICALISM = 'Extremist'"
~db2user/sqllib/bin/db2 connect reset
```

It is also possible to use this data directly in DISTILLERY using the Database toolkit.

F.3.2 Target selectors

The contents of this dataset are classified TOP SECRET STRAP2 UKEO.

Our target knowledge database is BROAD OAK which includes the ability to task various selector types including phone numbers and email addresses. The resulting list of selectors is sometimes called the target dictionary and is delivered to our DISTILLERY clusters at least once a day, and is also available on our Hadoop clusters. This data could be used to see if some result set contains an increased density of targets.

For DISTILLERY, the telephony and C2C dictionaries are delivered in separate streams and can be imported with `DataFeed=="BroadOak" && EventType=="TargetSelector"` and `DataFeed=="BroadOakC2C" && EventType=="TargetSelector"` respectively. A re-send of the latest dictionary can be requested by sending a UDP packet to [redacted] (port 10450 for telephony, 10460 for C2C) containing the line `resend`. This could be sent with a UDPSink operator.

On SUN STORM, the BROAD OAK reference data (all target types) is in HDFS at [redacted].

When using selectors to examine parts of a graph then this is considered targetting and an HRA log must be completed. See appendix E on page 67 for details.

F.3.3 Covert Infrastructure

The contents of this dataset are classified TOP SECRET STRAP1.

GCHQ has knowledge of, and collection from, CNE acceses owned by foreign intelligence agencies. This is done without their permission and is known as fourth party collection. As data is exfiltrated from target networks we should be able to see information flows over their infrastructure. Data on foreign covert infrastructure can be found at [redacted].

We also have knowledge of our own covert infrastructure. However this data is understandably more sensitive. Work is still ongoing to explore the possibility of making this dataset accessible to HIMR researchers.

F.3.4 Conficker botnet

The contents of this dataset are classified SECRET STRAP1.

GCHQ has an interest in being able to detect botnets operating in the wild. This is currently done using packet content fingerprinting and specific behaviours of certain bot software. However we would like to be able to detect botnets only by their generalisable activity. For the Conficker botnet we have a list of IP addresses that hit against either the packet fingerprinting or a Conficker specific activity profile. The fingerprinted IPs can be found at [redacted] This set should be largely reliable as the signature is believed to be highly discriminative. The behaviourally identified IPs are at [redacted] These are slightly more tentative and are based on the Conficker software contacting remote IPs on specific ports. Of course this can happen randomly so only those IPs which perform a significantly high number, after Bonferoni correction, make the list. As Conficker contains a peer-to-peer (P2P) component we believe that there may be information flows involving these potentially infected IPs.

F.3.5 Payphones

The contents of this dataset are classified TOP SECRET STRAP1.

Analysts are interested in understanding telephone numbers in their analysis. A particular feature of a number they would like to know is whether the number is a payphone. The fact that a number is a payphone would suggest that contact chaining through the number is not recommended. On the other hand some target discovery work starts with a known modus operandi of payphone usage (which targets follow to make it hard to target their communication) and so looking for communication between payphones is the starting point.

However GCHQ have lists of payphones for very few countries. The aim is start from partial lists of payphones in some countries and extend to full lists of payphones in those countries

Country	# known payphones	Filename	Notes
Spain	93,634	[redacted]	Believed near complete
Pakistan	3,117	[redacted]	Partial
Pakistan	118	[redacted]	Partial (FATA region only)
Barbados	761	[redacted]	Partial
Surinam	363	[redacted]	Partial

Table 5: Known payphones

and in other countries based on call meta-data. This problem is an example of positive-only learning.

[redacted] has provided lists of payphones in four countries as described in table 5.

GCHQ have recently moved their telephony event data to the cloud and we do not have feature extraction algorithms for this data. However the basic features in [I3] should be easy to implement from scratch as an Hadoop analytic using the data as described in appendix F.1.1. We also provide the source code for the original SPIKY ROCK feature extraction as C code.

The use of payphones is an active interest so a complete Hadoop feature extraction and classification analytic would be likely to be directly taken on by GCHQ and results fed into the the LUCKY STRIKE database. ◀

The data and the old SPIKY ROCK source code is available in [redacted]

F.4 SIGINT truthed data

To support the beyond supervised learning research we provide several SIGINT truthed datasets from recent research.

F.4.1 Logo recognition

The contents of this dataset are classified SECRET STRAP1.

We are interested in automatically detecting the source of videos on the internet through the recognition of logos in the video. We have previously researched logo detection and have recently looked at supervised machine learning for logo recognition [I19]. [redacted] has provided the data from this research.

The feature space is derived as follows:

1. Logo detection algorithms give us the logo and mask (i.e. the logo shape) as 8-bit images. The mask is binary in that values are either 0 or 255 (i.e. black or white).

2. Both the logo and mask are independently downscaled to 8×8. During these down-sampling processes the results are rescaled so that they retain their original pixel depth (i.e. 8-bit).

3. These 2 resulting 8-bit images are pointwise multiplied to give a 16-bit image (with a range of 0-65535).

4. This 8×8 16-bit image is our feature vector.

Sam is happy to look at extracting other features if HIMR actively research this dataset.

We provide 530 truthed samples. The data has been truthed to 109 classes. 7 classes have more than 20 examples, 67 classes have only 1 example.

[redacted] from ICTR-MCA is happy to work with JTRIG to try and provide larger untruthed datasets if the researchers decide to work on this problem.

The data is available from [redacted]. The first field "Class" is a numeric representation of the class and the remaining fields are the features.

F.4.2 Spam detection

The contents of this dataset are classified SECRET STRAP1.

Spam emails are a large proportion of emails seen in SIGINT. GCHQ would like to reduce the impact of spam emails on data storage, processing and analysis. Most external spam detectors work by analysing the content of an email however policy and processing mean this option is not always open to us. We must work on features derived from events alone. We therefore lower our target and instead aim to classify email addresses by the type of emails they send.

[redacted] has provided datasets from his team's research into this problem. They built a classifier called MYOFIBRIL [I44]. The dataset consists of an 1809 example email addresses with 143 features each truthed into 11 classes. Note that one class is "multiple_classes" and one class is "uncertain".

This data set is provided in [redacted] This directory also contains a PDF documenting the dataset in more detail [I43].

It would be possible to use the data at [redacted] on SUN STORM (reading these files with SILVER LIBRARY is recommended) to generate untruthed feature vectors. However it should be noted that the collection posture of GCHQ has changed considerably since the truth data was collected.

F.4.3 Protocol classification

The contents of this dataset are classified SECRET STRAP1.

GCHQ is interested in understanding C2C traffic on bearers. One approach is to use signatures of known applications but signatures can not cover all traffic. We therefore look at the alternative approach of classifying traffic based on its behaviour. Such approaches may also provide a way to understand traffic in encrypted tunnels.

We provide datasets from 7 different bearers provided by [redacted]) [I54] and used in [I70] (see [W1] for related research). Each bearer's data consists of a little over 100,000 example TCP flows with 51 features truthed to 15 broad classes (and 39 detailed classes). These classifications have been obtained by binary content signatures. Note that one class is "NULL" which indicates that no signature hit on that flow.

These datasets allow one to check the robustness of a classifier against concept drift both in time (the data spans a little over a year [I70]) and across bearers.

The data is provided at [redacted] Note each bearer's data is arbitrarily split into training and test sets but this split need not be preserved.

F.4.4 Steganography detection

The contents of this dataset are classified TOP SECRET STRAP1.

Some targets try to hide their communications through the use of steganography. One approach is to slightly alter the coefficients in a JPEG image to encode the hidden data whilst trying to minimise visual changes in the JPEG. We would like a classifier that can identify such altered images (commonly called stegged images).

[redacted] has provided data used to build Random Forest classifiers [I74]. There are a few interesting features of this dataset:

- This problem can be viewed either as a classification or a regression problem. One can either regress on the density of embedded data or classify whether there is any embedded data or not.
- The truthing process simulated embedding data into a large sample of images. Some of these images might have started with steganography in them and so the truthing may not be accurate.
- This dataset has the most truthed examples. The original research had 50,000 clean and 50,000 stegged images (for each of 4 steg types) for training and 500,000 clean and 200,000 stegged images (per steg type) for validation and testing. The reason for such a large test and validation set is that we want to ensure a very low false positive rate.
- There are the most features of the truthed datasets. There are 661 features (introduced in [I74]). Features are computed in classes and so could be used to experiment with cost-sensitive feature selection.

Data is available at [redacted] as compressed CSV files. The first two fields should be ignored and the final field of the stegged files contains the simulated stegging rate. The rest of the fields are the features (summarised in [I74]).

F.4.5 Genre classification

The contents of this dataset are classified SECRET STRAP1.

A way to make analysts more efficient and also allow further analytics is to add labels to textual content to describe the genre of the content. [redacted]) has provided text data sets classified into genres along with the AURA feature extractor [I83].

AURA extracts 108 features from text documents as described in [I85, I84]. Some features are from basic counts, some are based on email headers and some are based on more advanced textual analysis. AURA can be run on a text document with:

```
java -jar aura.jar <text_file> | head -108
```

The first 108 lines of AURA's output are the feature values. Other information about the file is returned in the remaining lines.

Three corpuses are provided. PEEC and genre-id are UNCLASSIFIED. News-Personal is classified. Each item is a file. The classification of items is encoded by the subdirectory an item is stored in. There are some duplicate items in these directories.

The corpuses and AURA are a [redacted]

If you are new to text classification then [I25] may be good background reading.

F.4.6 Website classification

The contents of this dataset are classified TOP SECRET STRAP1.

We would like to label webpages by the type of information on the page. In this case we want to identify pages that contain information on chemical, biological, radiological or nuclear (CBRN) weapons. [redacted] has previously researched this as a supervised learning problem [I36] and has provided his data from this research.

Webpages have been labelled by an analyst into four classes: CW (chemical weapons), BW (biological weapons), RN (radiological/nuclear) and NI (not interesting).

Data is provided at [redacted] in the "arff" format used by Weka. This format can be treated as a CSV file after removing the header lines.

The most important file is all[redacted]. This file is the full dataset used to produce the original classifier. It contains vectors for all documents in the CBRN dataset, where each feature corresponds to a single word, and the value of each feature is the number of instances of the word in the document divided by total number of words in the document.

[redacted] also produced lists of the most prevalent words across each topic (in the folder as Bvector, Cvector, Rvector, Nvector and RNvector), and then developed a dataset where each feature was a count of words from each list found in the document (the two [redacted] files). [redacted] reports that these features did not produce very good results compared to individual words, so the dataset wasn't refined much further, but you may find it interesting.

If required the original HTML pages and classifications may be available but could not be easily found at the time of writing the problem book.

F.5 Fusion of scores data

The contents of this dataset are classified SECRET STRAP2 CHORDAL UKEO.

The fusion of scores problem can occur in several contexts. In the following we describe the creation of IP geolocation reference data.

We want to know the geolocation of IP addresses for many analytic processes. In the main, there are two types of data we use:

Commercial Various commercial providers provide estimates for IP address locations. ICTR-NE have provided Akamai's Edgescape dataset.

SIGINT We find that commercial providers sometimes give poor locations in areas of the world of SIGINT interest. We therefore need to augment the commercial data and choose to do this through analysis of locations referenced in IP collection (e.g. in user profiles or web forms). We hope that these locations give evidence towards the location of the

users of that IP address. See [I69] for the scoring approach currently used. ICTR-NE have provided data from five different types of IP data (called INJUNCTION, PSYCHIC SALMON, RAGING BULLFROG, ROBOTIC FISH and TIMID TOAD).

The aim is to use these different sources to come up with the best estimate of an IP address's location. For simplicity we recommend considering geolocation to country-level only.

The Edgescape data comes as five gzipped text files; each file covers a different range of IP address space. Each line describes a subnet (an IP address or IP address range). The first field gives the IP address range as a subnet and a subnet mask. The second field contains information about the subnet as key-value pairs. In particular the "country_code" field is their guess of the country and the first letter of the "confidence" field gives the confidence in their estimate. Confidences are either high "H", medium "M" or low "L".

Each SIGINT system dataset also comes as a gzipped text file with each line describing a subnet. The full format is described in [I66] but we describe the important features here. The first field is the subnet and the second field the subnet mask (typically 24). The last field is a semi-colon separated field where the penultimate field is the country and the last character of the last field is the confidence. Confidences are either high "H", medium "M" or low "L". These confidences should not be treated as being on the same scale as Edgescape but should be comparable between the five SIGINT systems.

The files can be found at [redacted]

THIS PAGE IS INTENTIONALLY LEFT BLANK

References

— Internal Literature —

[I1] [redacted] A maximum-entropy algorithm for generating random graphs, 2008. Report to appear: see slides DISCOVER 12747135.

[I2] [redacted] Modeling and simulation of streaming dynamic graphs using the Cray XMT. Technical Report MR/TECH/010/09, R1, NSA, 2009. DISCOVER 11806816.

[I3] [redacted] SPIKY ROCK: Automatic classification of telephone type via usage statistics. Technical Report MR/TECH/003/04, NSA R1, March 2004. DISCOVER 12211328.

[I4] [redacted] COMET: A recipe for learning and using large ensembles on massive data. *arXiv*, cs.LG/1103.2068, March 2011. DISCOVER 12192977.

[I5] [redacted] Lossy counting and hierarchical heavy-hitter algorithms. In *ACE*, May 2011. DISCOVER 12768692.

[I6] [redacted] Comments on NSASAG 07-04: Correlation of temporal sequences. Technical report, University of California, Berkeley, October 2007. DISCOVER 12689034.

[I7] [redacted] DISCOVER 12134962, June 2009.

[I8] [redacted] Random Forests in MapReduce. Technical Report MR/TECH/033/10, NSA R1, August 2010. DISCOVER 10833587.

[I9] [redacted] Technical report, GCHQ, August 2011. DISCOVER 7996430 Please contact [redacted] (ICTR-FSP) for access.

[I10] [redacted] Adapting SIGINT timeseries data to account for variation in collection posture. Technical Report OPC-M/Tech.B/58, GCHQ, February 2011. DISCOVER 7895676.

[I11] [redacted] BAKER'S DOZEN – a method for batch phone discovery. Technical Report B/6749BA/5001/4/102, GCHQ, March 2008. DISCOVER 12585962.

[I12] [redacted] The La Jolla SCAMP 2009 problem book - information processing. Technical Report SCAMP Working Paper L3/09, IDA-CCR, 2009. DISCOVER 12907519.

[I13] [redacted] Density estimation techniques for detecting DNS tunnels. In *SANAR*, October 2010. DISCOVER 10833937.

[I14] [redacted] Trusses and fast graph clustering. Slides from SANAR, 2005. DISCOVER 12750235.

[I15] [redacted] Trusses: cohesive subgraphs for social network analysis. Technical Report TR-R63-001-05, NSA/R63, 2005. DISCOVER 12892477.

[I16] [redacted] Trapezes: more swinging than trusses. Technical Report TR-R63-004-09, NSA/R63, 2009. DISCOVER 12892478.

[I17] [redacted] Semitrusses, semitrapezes, efficient computation, and streaming. Technical Report TR-R63-002-10, NSA/R63, 2010. DISCOVER 12892476.

[I18] [redacted] Interpreting random forest classification. Technical report, NSA R1, September 2009. DISCOVER 12193642.

[I19] [redacted] Technical Report TRMCA/Inf/656, ICTR-MCA, June 2010. DISCOVER 12667333.

[I20] [redacted] A simple birth-and-death process for scoring anomalous behavior in a random steady-state graph. Technical Report SCAMP Working Paper L38/02, IDA-CCR, 2003. DISCOVER 11775537.

[I21] [redacted] Building 2-out+ graphs from streaming data. Technical Report H-TR-017-06, CSEC, November 2006. DISCOVER 12750234.

[I22] [redacted] Upgrade to tcp-file-signature-log – engineering specification. Technical Report B/6805BA/5001/1, GCHQ, February 2008. DISCOVER 12740356 Please contact [redacted] (ICTR-FSP) for access.

[I23] [redacted] Clustering of multiple point process streams. Technical report, University of Florida, July 2007. DISCOVER 12680283.

[I24] [redacted] A model-based approach to detecting correlated stochastic processes. Technical report, NSA, September 2011. DISCOVER 13432581.

[I25] [redacted] Text categorisation – a beginners guide. Technical Report B14/INF/569, GCHQ, February 2008. DISCOVER 7878359.

[I26] [redacted] Creating probabilities and combining evidence in NSA's evidence store. Technical Report TRMCA-INF-690, GCHQ, July 2010. DISCOVER 10835062.

[I27] [redacted] CROUCHING SQUIRREL v4: Filtering and classifying using behavioural vector analysis. Technical Report B/7934/5001/3/104, GCHQ, December 2010. STRAP2/ORCON: contact author for access or see slides DISCOVER 11925562.

[I28] [redacted] Role assignment in private messaging social networks. Technical Report B/7167BA/5001/4/102, GCHQ, December 2008. DISCOVER 13042933 Please ask [redacted] for access.

[I29] [redacted] Bayesian block modelling. Technical Report B/7635BA/5001/4/101, GCHQ, November 2009. DISCOVER 13036032.

[I30] [redacted] Weighted bayesian block modelling sanitised. Technical Report B/7713BA/5001/4/102, GCHQ, February 2010. DISCOVER 13036033.

[I31] [redacted] Techniques for measuring the strength of communications in email event graphs. Technical Report B/6845BA/5001/4/102, ICTR-DMR, February 2008. DISCOVER 12656488.

[I32] [redacted] The case for target discovery using closed loops against the Islamist terrorist threat in the UK - a technical perspective. Technical Report B/7618BA/5001/4/102, ICTR, GCHQ, 2009. DISCOVER 12861894.

[I33] [redacted] Properties of SIGINT-collected communication graphs, 2003. Technical Report SCAMP Working Paper L39/03, IDA-CCR, 2004. DISCOVER 11770806.

[I34] [redacted] Properties of SIGINT-collected communication graphs. Technical Report SCAMP Working Paper L32/02, IDA-CCR, 2003. DISCOVER 12502484.

[I35] [redacted] The NetInf algorithm as a MapReduce job. Technical report, NSA, May 2011. DISCOVER 13202916.

[I36] [redacted] Automated categorisation of CBRN related webpages. Technical Report B/7470BA/5001/5/105, ICTR-CISA, July 2009. DISCOVER 12669793.

[I37] [redacted] Histogramming in the streaming environment. In *ACE*, 2007. DISCOVER 12632758.

[I38] ICTR-FSP. GCHQ TR-FSP HTTP header fingerprint format. B/7535BA/5001/1, 2009. DISCOVER 2450313.

[I39] [redacted]. Detecting dependence among multiple point processes. Technical report, University of Pittsburgh, August 2007. DISCOVER 12681522.

[I40] [redacted] Estimating set cardinality under streaming conditions. In *ACE*, 2011. DISCOVER 12754015.

[I41] [redacted] Timing patterns in call records data with i2 Pattern Tracer and Remit. Technical Report B/4351BA/1700/16, GCHQ, July 2003. DISCOVER 12207962.

[I42] [redacted] Timing analysis 2004 - using significant temporal chains for call records target development. Technical Report B/5372BA/1700/16, GCHQ, October 2004. DISCOVER 12200466.

[I43] [redacted]. Pizza node classification – sharable data set for Random Forests classification. Technical Report B/6316BA/5001/4/102, GCHQ, December 2006. DISCOVER 12189035.

[I44] [redacted] Classifying email addresses by their behaviour in bulk events data. Technical Report OPH-M/TECH.A/458, GCHQ, December 2006. DISCOVER 12183192.

[I45] [redacted] Creating, retaining and combining confidence estimates in a cloud-like world, July 2010. DISCOVER 10833599.

[I46] [redacted] Topological measures of evolving graphs: Dynamic betweenness centrality. Technical Report CCR Working Paper 1690, IDA-CCR, 2008. DISCOVER 11770815.

[I47] [redacted] Detecting correlated sequences of events. Slides from SANAR 2010, October 2010. DISCOVER 12595305.

[I48] [redacted] Extending pairwise element similarity to set similarity efficiently (sanitized version). Technical Report MR/TECH/032/10, NSA, December 2010. DISCOVER 12497649.

[I49] [redacted] Detecting correlated sequences of events: sanitized version. Technical Report OPH-M/Tech.A/456, GCHQ, August 2006. DISCOVER 3730313.

[I50] [redacted] and [redacted] Elkan and Noto's "Learning classifiers from only positive and unlabeled data" is fatally flawed. Technical Report MR/TECH/009/10, NSA R1, February 2010. DISCOVER 10833916.

[I51] [redacted] Streaming temporal relation additive probability. Technical report, NSA, April 2009. DISCOVER 12594200.

[I52] [redacted] CLASPing at straws: Bootstrapping and clustering to improve product performance. Technical report, NSA, In Preparation 2011. DISCOVER 12588233.

[I53] [redacted] Improvements to GeoFusion scoring. Technical Report B/7854/5001/3/104, GCHQ, August 2010. DISCOVER 12839607.

[I54] [redacted] Application characterisation: Data set specification. Technical Report B/6728BA/5001/1, GCHQ, December 2007. DISCOVER 12189832.

[I55] [redacted] Embedding R within InfoSphere Streams for online time series analysis and predictions. Technical report, GCHQ ICTR-CISA, June 2011. DISCOVER 12267642.

[I56] [redacted] Towards implementation of an algorithm for updating eigenvalues and eigenvectors of streaming graphs. Technical report, ICTR, GCHQ, 2011. DISCOVER 12663078.

[I57] [redacted]. Generating realistic random graphs. Technical report, HIMR, September 2008. DISCOVER 12747134.

[I58] [redacted] Unsupervised learning on network data. DISCOVER 12589745.

[I59] [redacted] RADONSHARPEN-B. DISCOVER 12838456, October 2010.

[I60] [redacted] 2008 Bristol SWAMP: Problems in graph mining. Technical Report OPC-M/TECH.A/6, GCHQ, 2008. DISCOVER 12768417.

[I61] [redacted] Streaming decision trees. `http://wiki.gchq/images/3/37/RDTrees.tar.gz`, June 2007. DISCOVER 12134963.

[I62] [redacted] HIDDEN OTTER: Detection of multi-hop temporal chains in IP traffic. Technical Report B/7937BA/5001/3/104, GCHQ, December 2010. DISCOVER 7810599 Ask [redacted] (ICTR-NE) for access.

[I63] [redacted] Streaming PRIME TIME application design report. Technical Report INCA1323D003-1.1, Detica, December 2010. DISCOVER 12211763.

[I64] [redacted]A new technique for correlating stochastic processes. Technical Report B/7523BA/5001/4/102, GCHQ, September 2009. DISCOVER 12214368.

[I65] [redacted] NSASAG problem 07-04: Correlation of temporal sequences. Technical report, University of Washington, 2007. DISCOVER 12687220.

[I66] [redacted] GeoFusion VOLSUNGA interface specification. Technical Report B/6745BA/5001/1, ICTR-FSP, January 2008. DISCOVER 13550106.

[I67] [redacted] File signature bulk-logging technique – engineering specification. Technical Report B/6173BA/B13/106, GCHQ, July 2006. DISCOVER 12742157.

[I68] [redacted] Internet flow classification: Random forests and importance-sampled learning ensembles. In *ACE*, October 2007. DISCOVER 12187796.

[I69] [redacted] A probabilistic score for IP geolocation from INJUNCTION-style data. Technical Report OPC-MCR/TECH.B/4, OPC-MCR, November 2007. DISCOVER 13548375.

[I70] [redacted] Application characterisation: Generalisation to the unknown. Technical Report OPC-M/TECH.B/7, GCHQ, April 2008. DISCOVER 12187131.

[I71] [redacted] Enhanced behavioural detection of botnet command-and-control servers. Technical Report OPC-M/TECH.B/53, GCHQ, July 2010. DISCOVER 12209112.

[I72] [redacted] Traffic sketches for measuring bearer similarity and pairing. Technical Report OPC-M/TECH.B/55, GCHQ, October 2011. DISCOVER 12750231.

[I73] [redacted] A comparison between billing records and the view in SIGINT: SOLID INK vs FLUID INK. Technical Report OPC-M/TECH.B/17, GCHQ, 2009. DISCOVER 12676097.

[I74] [redacted] Improved generic detection of steganography in JPEG coefficients. Technical Report OPC-M/TECH.A/452, GCHQ, April 2011. DISCOVER 12191941.

[I75] [redacted] GCHQ research & innovation strategy 2011 - 2015. DISCOVER 12013908.

[I76] [redacted] Blocks, bridges and cutvertices in large communications graphs. Technical Report OPC-M/TECH.B/19, HIMR, October 2008. DISCOVER 12750236.

[I77] [redacted] Using predictive modelling to identify cocaine drug smugglers. Technical Report B/4029BA/B1700/13, GCHQ, November 2002. DISCOVER 12815822.

[I78] [redacted] Results document for call record timing analysis. Technical Report CAA146D005-1.0, Detica, November 2001. DISCOVER 12204886.

[I79] TDB. Interface control document (ICD) for the SALAMANCA input handler – external generic feed interface. Technical Report PC/00117CPO/4542/PC0093/000/50, GCHQ, 2010. DISCOVER 12680902.

[I80] [redacted] Knowledge discovery at the new CRI. In *SANAR*, October 2010. DISCOVER 12208587.

[I81] [redacted] Updating eigenvalues and eigenvectors of streaming graphs. Technical Report SCAMP Working Paper L22/09, IDA-CCR, 2010. DISCOVER 11806837.

[I82] [redacted] SAWUNEH 2010 — cyber defence event mining. In *ACE*, May 2011. DISCOVER 12754021.

[I83] [redacted]DISCOVER 12369711.

[I84] Aura features: Algorithmic description. DISCOVER 12380899.

[I85] Aura features: Brief description. DISCOVER 12380897.

[I86] SKB definitions. [redacted] Please ask [redacted] (ICTR-FSP) for access.

— External Literature —

[E1] [redacted] The canonical tensor decomposition and its application to data analysis, June 2009. DISCOVER 12740014.

[E2] [redacted] An empirical study of dynamic graph algorithms. *ACM Journal on Experimental Algorithmics*, pages 192–201, 1996. DISCOVER 11402183.

[E3] [redacted] STINGER: Spatio-temporal interaction networks and graphs (STING) extensible representation, May 2009. DISCOVER 11821050.

[E4] [redacted] Network science applications to global communications. In *NetSci*, 2008. DISCOVER 12804967.

[E5] [redacted] and [redacted] The phase transition in inhomogeneous random graphs. arXiv:math/0504589v3, June 2006. DISCOVER 12763412.

[E6] [redacted] Forests. *Machine Learning*, 45, 2001. DISCOVER 13286408.

[E7] [redacted] editors. *Semi-supervised learning*. MIT Press, 2010.

[E8] [redacted] Data stream algorithms intro, sampling, entropy. Slides from Bristol Maths workshop, 2008. DISCOVER 12805861.

[E9] [redacted] How does the data sampling strategy impact the discovery of information diffusion in social media? *Association for the Advancement of Artificial Intelligence*, 2010. DISCOVER 10763381.

[E10 [redacted] MapReduce: Simplied data processing on large clusters. In *OSDI*, 2004. DISCOVER 12192986.

[E11] [redacted] Bayesian approaches to modeling the conditional dependence between multiple diagnostic tests. *Biometrics*, 57:158–167, 2001. DISCOVER 12192608.

[E12] [redacted]. Massive streaming data analytics: A case study with clustering coefficients, 2010. DISCOVER 11821048.

[E13] [redacted]. Learning classifiers from only positive and unlabeled data. In *KDD*, Las Vegas, August 2008. ACM. DISCOVER 12195326.

[E14] [redacted] Dynamic graph algorithms, 1999. DISCOVER 11402184.

[E15] [redacted] Graph distances in the data-stream model. *SIAM Journal on Computing*, 38(5):1709–1727, 2008.

[E16] [redacted] HyperLogLog: the analysis of a near-optimal cardinality estimation algorithm. *AofA*, 2007. DISCOVER 12747198.

[E17] [redacted]. Semi-supervised ranking on very large graphs with rich metadata. In *KDD*, August 2011. Available from Microsoft Research's website.

[E18] [redacted] Inferring networks of diffusion and influence. In *KDD'10*, Washington D.C., 2010. ACM. DISCOVER 12762008.

[E19] [redacted] Estimating the error rates of diagnostic tests. *Biometrics*, 36:167–171, 1980.

[E20] INSTINCT. Have I got "views" for you?: gathering and analysing publicly available data to gain an understanding of current events. Technical report, October 2011. DISCOVER 12626622.

[E21] [redacted] Bayesian estimation of disease prevalence and the parameters of diagnostic tests in the absence of a gold standard. *American Journal of Epidemiology*, 141(3):263–272, 1995. DISCOVER 12195321.

[E22] [redacted] Introducing the Enron corpus. Technical report, Carnegie Mellon University, 2004. DISCOVER 12763413.

[E23] [redacted] MIForests: Multiple-instance learning with randomized trees, 2010. DISCOVER 12192687.

[E24] [redacted] Semi-supervised random forests, 2011. DISCOVER 12192698.

[E25] [redacted] Social media analytics: Part 1: Information flow. Slides, Stanford University, August 2011. Presented at KDD 2011 DISCOVER 13561614.

[E26] [redacted], [redacted] and [redacted] Graph evolution: Densification and shrinking diameters. *ACM Transactions on Knowledge Discovery from Data*, 1(1), 2007. DISCOVER 12761498.

[E27] [redacted] Learning with an unreliable teacher. *Pattern Recognition*, 25(1):79–87, 1992.

[E28] [redacted] Compact graph representations and parallel connectivity algorithms for massive dynamic network analysis. In *23rd IEEE International Parallel and Distributed Processing Symposium (IPDPS)*, May 2009. DISCOVER 11816428.

[E29] [redacted] Sparsification of influence networks. In *KDD'11*, pages 529–537, San Diego, CA, August 2011. ACM. /discover13560696.

[E30] [redacted] The PageRank citation ranking: Bringing order to the web, 1998.

[E31] [redacted] Supervised learning from multiple experts: Whom to trust when everyone lies a bit. In *Proceedings of the 26th International Conference on Machine Learning*, 2009. DISCOVER 12197907.

[E32] [redacted] "TAGS", a program for the evaluation of a test accuracy in the absence of a gold standard. *Preventative Vetinary Medicine*, 53:67–81, 2002.

[E33] [redacted] A method for inferring label sampling mechanism in semi-supervised learning. In *Advances in Neural Information Processing Systems*, volume 17, 2005. DISCOVER 13287597.

[E34] [redacted] Correcting for missing data in information cascades. Technical report, Stanford University, December 2010. DISCOVER 10763155.

[E35] [redacted] Combined regression and ranking. In *KDD*. ACM, July 2010. DISCOVER 12815522.

[E36] [redacted] Learning with labeled and unlabeled data. Technical report, University of Edinburgh, December 2002. DISCOVER 13287596.

[E37] [redacted] Active learning literature survey. Technical report, University of Wisconsin-Madison, 2010. DISCOVER 12195329.

[E38] [redacted] When do latent class models overstate accuracy for binary classifiers?: With applications to jury accuracy, survey response error, and diagnostic error. Technical Report WP-08-10, Northwestern University, May 2009. DISCOVER 12192699.

[E39] [redacted] Streaming data. *WIREs Computational Statistics*, January 2011. DISCOVER 12197914.

[E40] [redacted] Fast counting of triangles in large real networks: algorithms and laws. In *ICDM*, 2008. DISCOVER 12805858.

[E41] [redacted] The future of data analysis. *Ann. Math. Stat.*, 1962. DISCOVER 12804965.

[E42] [redacted] *Exploratory data analysis*. Addison-Wesley, 1977.

[E43] [redacted] Design principles for developing stream processing applications. *Software – Practice and Experience*, 2010. [redacted]

[E44] [redacted] Modeling information diffusion in implicit networks. Technical report, Stanford University, 2010. DISCOVER 12763414.

[E45] [redacted] Boosting the scalability of botnet detection using adaptive traffic sampling. In *ASIACCS*, March 2011. DISCOVER 12804966.

[E46] [redacted] Semi-supervised learning literature survey. Technical Report TR 1530, University of Wisconsin-Madison, July 2008. DISCOVER 13288447.

— Websites —

[W1] Application characterisation. [redacted]

[W2] AUTO ASSOC. [redacted]

[W3] BIRCH (data clustering). [redacted]

[W4] CARBON COPY. [redacted]

[W5] CASK: situational awareness for the 2012 Olympics [redacted]

[W6] CHART BREAKER. [redacted]

[W7] CNE OpSec pages. [redacted]

[W8] CNO glossary. [redacted]

[W9] CRAN. [redacted]

[W10] Decision tree learning on Wikipedia. [redacted]

[W11] DISTILLERY. [redacted]

[W12] Dynamic Graph. [redacted]

[W13] Ensemble learning on Wikipedia. [redacted]

[W14] Fused analysis and visualisation research. [redacted].

[W15] GCWiki. [redacted]

[W16] Getting started on BHDIST. [redacted]

[W17] GRINNING ROACH. [redacted]

OPC-M/TECH.A/455 (v1.0, r206)

[W18] Ground breaking intelligence capabilities used during recent G20 summit. [redacted]

[W19] Hadoop Fair Scheduler guide. [redacted]

[W20] Hadoop on GCWiki. [redacted]

[W21] HIMR IT upgrade. [redacted].

[W22] HIMR self help. [redacted]

[W23] HRA logging. [redacted]

[W24] Information flow in graphs GCWiki page. [redacted]

[W25] Legal compliance. [redacted]

[W26] Legalities SUN STORM/BLACK HOLE. [redacted]

[W27] MAMBA. [redacted]

[W28] NSASAG. [redacted]

[W29] Pidgin setup. [redacted]

[W30] PIRATE CAREBEAR. [redacted].

[W31] Random Forests. [redacted]

[W32] Relationship analysis. [redacted]

[W33] Renoir. [redacted]

[W34] ROC curves. [redacted]

[W35] Safari books online. [redacted].

[W36] SALAMANCA. [redacted]

[W37] SALTY OTTER. [redacted]

[W38] Semi-supervised learning on Wikipedia. `http://wikipedia.gchq/index.php` | [redacted]

[W39] Squeal eAD and cipher detection PPF app. [redacted]

[W40] Streams Processing Language. [redacted]

[W41] Supervised learning on Wikipedia. [redacted]

[W42] SWAMP 2008. [redacted].

[W43] What's the relationship between CNO and DNI? [redacted].

[W44] WHITERAVEN. [redacted]

[W45] [redacted] KL-Relative PageRank. [redacted]