Interdisciplinary Approaches to Information Security

Additional Presentation Notes

Cansecwest/core05

by Christopher Abad

1 Who am I?

Christopher Abad
Lead Scientist @ Cloudmark, Inc.
aempirei @ the-mathclub.net

2 What this presentation is about

Profiling and the Analysis of Behavior
Classification and Analysis Methods
Logically Grouping things into Categories
Effectively Modeling Information
Fighting Anonymity

3 Profile

Presenting a subjects noteworthy characteristics
A formal summary or analysis of data, often in the form of a graph or table, representing distinctive characteristics.
Recording a person's behavior and analyzing psychological characteristics in order to predict or assess their ability in a certain sphere or to identify a particular group of people.

4 Behavior

A manner of conducting oneself
Anything that an organism does involving action and response to stimulation
The response of an individual, group, or species to it's environment.
The way in which something behaves, functions or operates.

5 Appearance

An external show
An outward aspect
A sense impression or aspect of a thing
The world of sensible phenomena
The act, action, or process of appearing
The presentation of oneself in court as a party to an action often through the representation of an attorney.

6 Unfortunately

Unfortuantely, more or less, if youre the watcher or the watched, most information is too complex to automatically profile.

7 Information Modeling & Deterministic Processes

Do not incorporate random or cognitive stimuli
PRNGs, FSMs, Conrads Game of Life
Properly modeling the underlying behavioral mechanism

8 Information Modeling & Random Processes

Incorporate random input stimuli.
Sufficiently complex black-box deterministic processes appear to be random processes, eg. PRNGs.
Games of chances, quantum dynamics.
Properly sampling data and building statistical models.

9 Information Modeling & Cognitive Processes

Relatively not understood.
Incorporates extremely complex, cognitive, stimuli
Plasticity in behavior, it changes based on the state of the system.
Increasing information about the process provides increasing predictability with fractal complexity.
Current state can affect outcomes an arbitrary number of transitions later.
Cognitive processes are often naively modeled with statistical models, simply because of their inherent complexity. For example city traffic, population migration and sequences of baseball pitches, semi-living artist.

10 I Before E

10.1 I Before E

Linguistic rule first modeled deterministically.
Found to be an invalid model

10.2 Except after C and in words like neighbor and weigh

Then modeled statistically, it sometimes occurs in situations like this

10.3 The rule approaches fractal complexity

I before E except after C and in words like neighbor and weight and eidetic, and chemicals like protein and caffeine, and names like Einstein, Neiman Marcus and Rottweiler and sentences like “Neither sheik seized either form of weird leisure.” And “Go reinstall windows.”

```
grep –v ‘\([rc]ei\)|(\ei[sdlnkrtzg\)’ words-with-ei.txt | wc
```

107 additional words that don’t match any of the digrams found above

11 Gonna is not going to

```
sed ‘s/going to/gonna/g’
```

I am gonna kill you.
I am gonna the store.

```
sed ‘s/I am going to/Imma/g’
```

Imma kill you.
Imma the store.
Properly modeling many linguistic characteristics of language yields fractal complexity.\

12 Fractal Complexity & Romanesque Broccoli

Many things exhibit fractal complexity
Written Language
Distribution of wealth
Topology of Friendster
Classical and Jazz Music
Connectivity of the Internet
ACGT protein sequences in DNA codings.
The sequences of opcodes in compiled programs.
Self similarity and fractal complexity is an emergent phenomenon of a class of information that follows 1/f, Zipf’s or Inverse Power Laws from a statistical model yet are in actually cognitive processes.

13 IPL & The Midas Touch

The reason we care about power law correlations is that we're conditioned to think they're a sign of something interesting and complicated happening.

Midas Touch, everything we touch becomes inverse power law.

14 Statistical Evidence of Language

Nearly every aspect of language exhibits an inverse power-law relationship

IPL is not a true linguistic regularity.

Mandlebot proved that even monkey typing exhibits Zipf's Law.

Reglardless, it is a strong and easily observable characteristic.

15 Applications of Inverse Power Laws (IPL)

Information Fingerprinting

Language Identification

Authorship Attribution

Cryptanalysis

16 Byte Frequency Analysis

Byte frequency is the simplest form of frequency analysis.

Count the number of times each byte value occurs across a given document.

Simple way to visualize inverse power laws

Frequency analysis is frequently used in linguistics, cryptanalysis and data compression.

Frequency analysis in cryptanalysis can be used to gain additional information about plaintext in cases where statistical aspects of natural language remain unaffected by the encryption process.

In data compression, frequency analysis can help with selection of which redundant and high frequency aspects of documents should be compressed.

To uncover a basic IPL relationship in language, simply sort the bins in a histogram from most-to-least frequent.

Logit-transform graphs are associated with normal or binomial distributions which are closely related with random data.

INSERT GRAPHS DNA, PLAINTEXT, XOR, DES, RANDOM

17 Substitution Ciphers

Byte frequency analysis applied to both the plaintext and ciphertext will yield sorted histograms identical in all aspects with the exception of the bin labels.

INSERT SORTED HISTOGRAMS FOR CIPHER AND PLAIN TEXT

Taking bin labels from left-to-right of the ciphertext and plaintext sorted histograms immediately reveals the substitution cipher mapping.

Substitution cipher mapping can be created by knowing the byte frequency distribution of a given language.

Relative ordering of each successively less common byte should be reliably consistent.

INSERT SORTED HISTOGRAMS FOR CIPHERTEXT AND CORPUS AND THEN A STATISTICAL DECRYPTION

18 N-Grams and Frequency Analysis

An n-gram is a consecutive sequence of symbols from having length n.

3-grams (trigrams) for the string “ELITE” would yield “ELI”, LIT” and “ITE”
n-grams can then be tallied and histograms can be created from their associated probability distribution.

```
INSERT CODE FOR DIGRAM FREQUENCY TABULATION
```

n-grams contain additional transitional information that would not otherwise be observed by frequency analysis of non-overlapping sampling.
n-grams and markov processes are quite related
Digram and Trigram frequency analysis is often used in basic cipheranalysis but does not seem to play a major role in any major compression schemes.
A common application n-gram frequency analysis is in text classification.

19N-Grams and Text Classification

Basic n-gram text classification can be performed using a method strikingly similar to both naïve Bayesian classification and Markov chains.
The first step is to build two corpuses, typically one with characteristic a and one with characteristic ~a.
n-gram frequency tables are built from the two corpuses for a user selected n, typically 2 to 4 for most natural languages.
build a matrix of n-gram to n-gram transitional probabilities.
Each cell represents the probability of a given state transition from a given n-gram to each possible next symbol.
The percentage of the time that B follows A, etc.

```
INSERT CODE AND TABLES FOR TRANSITIONAL
MATRICES
```

20N-Gram scoring in Classification

These matrixes can be used to score a document in question in terms of probabilistic similarity to either of the two corpuses.
A document’s score in terms of a given n-gram probability matrix is its multiplicative sum of the documents path of n-gram chains through the matrix.
The matrix that results in the highest score for a given document is said to classification of the document in question.
What this similarity value actually represents is the probability of the given path, or sequence of n-grams describing the document in question being generated by the stochastic process governing the associated corpus.
The shortcomings of n-gram classification here are namely that natural language is not a stochastic or random process although its statistical properties are relatively reliable.
additional contextual insight into the next state transition may actually exist outside of the n-gram window.
These shortcomings can be readily seen in applications to text generation using Markov chains.

```
INSERT EXAMPLES OF SIMILARITY SCORING
```

20.1 User fingerprinting

Sequences of commands at a command shell also happen to be easily classifiable using n-gram analysis.

21Markov Processes

A Markov Process is a type of stochastic process that has the Markov Property.
Transitional probability distribution of future states depends only on the current state.
Additional insight into its past states provides no additional knowledge of future outcomes.
Natural language is clearly not a Markov Process.
Modeling language this way still proves useful for certain tasks.
One task it is not useful for is text generation.

22 Text Generation

Text generation is done by performing a random walk through an n-gram transition probability matrix.

INSERT A MARKOV PROCESS GRAPH AND N-GRAM MATRIX

Markov Processes built from digrams or trigrams will yield peculiar language-like gibberish, but rarely anything coherent.
For larger n’s, generated text will either have significant syntactical errors, or will not significantly deviate from the original corpus.

INSERT RANDOM WALK CODE AND GENERATED TEXT

23 Syntactical Structure

N-gram models clearly fail to capture the syntactical structure of language.
This is seen by its inability to generate simple coherent text.
This failure is partially due to the fact that structural influence exists outside of the n-gram window.
An interesting approach combines of ideas from n-gram analysis and dictionary based compression.
Dictionary based compression works on the premise that commonly occurring group of symbols, called phrases, can be replaced by a new symbol with an expansion rule.

EXAMPLE DICTIONARY THE FAT CAT ATE A RAT.

24 Dictionary Compression and Obligatory Structural Highlighting

Dictionary based compression has no obligation or real necessity to highlight structure.
A very simple extension on this idea can readily infer a hierarchical structure within documents.
The basic idea is that small phrases can be replaced by new dictionary rules and allowing for dictionary rules to recursively contain other rules.
The result is the compressed document and then the dictionary of recursively defined expansion rules, and thus grammar trees.

25 SEQUITUR

This is known as the SEQUITUR algorithm.

EXAMPLE ALGORITHM AND RECURSIVE DICTIONARY FOR THE FAT CAT…

Classification methods taking into account structural characteristics turned out to be very effective.

A phylogenetic taxonomy of Dog breeds was actually performed using this premise, as was a detailed dichotomy of the evolution of western languages in a paper entitled “Language Trees and Zipping.”

Testing PRNGs with compression is actually relatively simple.

`GZIP THE STREAMS OF PRNG LINUX VS BSD`

26 SEQUITUR and Text Classification

Text classification and authorship attribution using SEQUITUR or dictionary based compression is based on the premise that a better compression ratio in terms of a given dictionary implies that the document in question is more similar to the corpus that generated the dictionary.

Simply try to compress the document in terms of each hierarchical dictionary created from a set of corpuses.

The dictionary that results in a shorter representation or minimal description length (MLD) of the document is the most similar to the document.

`EXAMPLE OF SPANISH AND ENGLISH VS A TEXT`

27 Idiot’s Bayes

A naïve bayes classifier is a simple probabilistic classifier.

Its probability model is a result of Bayes’ Theorem and the conditional probability theorem.

The definition of conditional probability states that $P(A|B)P(B) = P(A \cap B) = P(B|A)P(A)$.

A detailed articulation of Bayes’ Theorem is not necessary to explain Naïve Bayes’ Classification due to oversimplified assumptions about independence made by this type of classification.

Naïve Bayes’ Classification assumes that each probability of an event is independent and is not affected by other events occurring.

This assumption is clearly not true for things NBC has been applied to, eg language where features are words occurring in a document.

Despite this bogus assumption, the NBC model works more often in real world applications than expected by its simple design.

SpamAssassin mostly works.

28 Application of NBC

Construct a table from words that occur in two corpuses, where each column represents one of the corpuses, and each row, a word, and each cell the number of occurrences of the given word within the given corpus.

Normalize each cell by diving it by the total number of words in the respective corpus.

Each cell in this word probability table represents the probability of a randomly sampled word from the column corpus being row word.

`INSERT WORD PROBABILITY TABLE BUILDING CODE AND IMG`

Classification is performed by calculating the multiplicative sum of the probabilities associated with all words in a given document that occur in the NBC table.

Scoring is similar to that from n-gram classification, this is called the method of maximum likelihood.
The corpus associated with the greatest score is considered to be most like the document in question.
INSERT DECISION CODE
NBC is simple extension on basic frequency analysis principles.
It takes into no account structural information of natural language.
This is a clear inherent shortcoming of this model.
NBC has a larger application domain than n-gram analysis and SEQUITUR as those methods cannot be applied to unordered sets.

29 Structural Analysis of Unordered Sets

The assumption of independence in NBC is a major shortcoming and is a translation of the same short comings of n-gram analysis in linear documents.
Neither NBC nor n-gram analysis take into account structural information present in data and essentially modeling data statistically as random processes.
The SEQUITUR algorithm provided a method to discern additional structural about the documents in question.
This was done through recursive tokenization of common adjacent symbols.
Unordered sets lack the concept of node adjacency.
The SEQUITUR algorithm can be readily extended if adjacency can be added to unordered sets.
Adjacency in graph theory is any two nodes that are connected by an edge.
The SEQUITUR algorithm in terms of this graph becomes an operation on adjacent nodes and their edge.
Common node pairs, and their edge, are replaced with a new node and an expansion rule. Given the additional property that an unordered set is fully connected, every node, or feature, is connected to every other node, the extension of the SEQUITUR algorithm follows nicely… Almost.
INSERT GRAPH OF LINEAR DOCUMENT VS FULLY CONNECTED

30 I,Sushi Algorithm

Given k features, k-1 edges would exist in a linear corpus as opposed to k^2-k if it were unordered.
A corpus would have to contain multiple training sets since most information modeled as unordered sets don't typically have item repetition.
A collection of open ports on a host will never contain the same port twice, nor will the order in which the ports are listed make a difference.
The digram 'TH' may occur many times in a single sentence and the order of 'T' and 'H' in the digram matters, as "HT" is distinctly different.
The exact extended algorithm, dubbed I, SUSHI, is as follows.
Given a collection of sets, recursively replace the most common two-feature pair across all sets with a new token and an expansion rule. Once no two-feature pair occurs more than once, stop. For any dictionary symbol occurring only once across all expansion rules and main corpus, expand the symbol and delete the expansion rule.

31NMAP and host structural analysis

NMAP is an open source tool for network exploration.
In a nutshell, NMAP performs host discovery, port scanning, operating system detection, and now application discovery.
In terms that we care about, NMAP provides us with a set of characteristics that describe a given host.
Characteristics are unordered, i.e. that the order in which open ports on a given host are listed provides no real information, only the fact that they are either open or closed.

```
INSERT NMAP SCAN COMMAND LINE
```

Because scan reports are in a plain textual format, all of the characteristics which we are interested in are parsed out.

```
INSERT NMAP SCAN REPORT AND INFO ON SECTIONS
```

32NMAP Corpus and intuitive structure

Our corpus is a collection of sets each representing a single host, and containing all of the characteristics about each that we are interested in interpreting a structure from.
The assumptions made are that host characteristics such as ports, services and OS are not independent features and they exhibit some kind of structure.
This is intuitively plausible because many feature pairings can be pointed out across different systems.
For example, the TCP port triplet of 135, 139, and 445 is likely to be associated with a Windows computer.
We can apply the I,SUSHI algorithm to attempt to discern structure within these sets.

33NMAP Corpus and TreeView

The output of the algorithm provides us with a structural representation of the original input sets in Newick Tree Format, now a common format used in the bioinformatics field for phylogenetic analysis.
TreeView is a useful tool in computational biology for viewing phylogenetic taxonomies.
The output is nice trees highlighting the structure we have been able to infer.

```
INSERT TREEVIEW TREES
```

Now that we have been able to infer structure from host characteristics, it is important to find out whether these structural characteristics are an outcome of random connectivity or if there is a clear IPL relation from the host characteristics.
If an IPL relationship can be successfully identified it’s not unlikely that a host classification method can be devised.

34IPL and host analysis

Host analysis would be rather difficult if there was no underlying structure and if host features exhibited strong independence, in that they appeared to have no biased associated with other features.
Given that an SMTP service is found on a given host, would it be more or less fathomable that the POP service will be found on the same host than if SMTP was not present?

This is the basis of a concept called preferential connectivity, something that is highly related to IPL relationships.

Analyzing the occurrence of independent random features and attempting to infer structure in their association will only yield incidental and random connectivity.

35 Random Connectivity vs. IPL

Random connectivity is not be what we observe in the NMAP corpus.

To test this hypothesis, additional corpuses have been randomly generated containing the same number of sets as the original corpus but where each feature is independent yet has the same likelihood of occurrence that is observed in the original corpus from our NMAP scan.

Because of this assumption of independence, structural characteristics in the original corpus should not be found in this generated corpus.

INSERT FREQUENCY DISTRIBUTION OF FEATURES

INSERT TREES GENERATED FROM TABLES BUILD FROM I,SUSHI FROM BOTH CORPUSES

INSERT FREQUENCY CHARACTERISTICS OF SYMBOLS FROM CORPUSES

36 Static Binary Analysis

Static Binary analysis is essentially the analysis of executable binary programs without having access to source code and without the need to actually run the program.

Early binary analysis methods consisted of simple pattern matching rules for know potential vulnerable conditions, such as calls to strcpy() or sprintf().

These methods are easily implemented with IDA scripts and even simple regular expression matching but as these simple vulnerable conditions became more and more apparent to now more security minded programmers this left only more obscure and complex vulnerabilities in software.

37 Binary Difference Analysis and Modularity

One way security analysts can glean insight into security holes is through security patches.

Patches are effectively the exact description of a security flaw and its addressed fix.

One assumption that patches make and is clearly true based on the fact that patches in fact do work, is that programs are modular.

Modification of a single portion of code does not necessarily impact the syntactical representation of the rest of the program.

From this assumption, it follows that we can expect that we can extrapolate the flaws and changes across two versions of a program simply by looking at their difference.

If a program were not modular, the difference would be too widespread to come to an understanding simply from their binary difference even though the programs are functionally, semantically, similar.

38 Binary Difference Analysis and Bioinformatics

A huge amount of work in the analysis of difference and functional changes of binary sequences has been performed in Bioinformatics.
DNA sequences of an ever expanding number of organisms have been stored and analyzed since the first sequencing of the Epstein-Barr Virus by the UK Medical Research Council in 1984.
DNA is a base64 code grouped in triplets called codons, each member of being one of four amino acids.
This information has since been extensively analyzed and comparisons of genes between species and sometimes across species can lead to an understanding of protein function similarities and overall species relations.
Simple computer program code is dwarfed by the sheer volume of DNA sequence code so manual genomic sequence analysis simply becomes infeasible.
The analysis of sequences is done with computers and sequence alignment algorithms that can account for many different types of mutations.
Needleman-Wunsch Algorithm for global sequence alignment was designed in 1970. It was designed to search for similarities in amino acid sequences of proteins.

39 Sequence Alignment

There are two main types of sequence alignment, local and global pair-wise alignment.
Pair-wise alignment is concerned with best finding pair-wise matches across two sequences based on some similarity scoring strategy.
The typical purpose in bioinformatics is to find homologous gene sequences and is useful in answering questions about protein function, ancestry, structural importance and molecular evolution.
Global alignment is concerned with finding the best global match for a given pair of sequences and is very useful in the alignment of similar sequences.
Local alignment is concerned with finding sections within sequences that are more strongly related.

40 Needleman-Wunsch

BEGIN EXCERPT FROM WIKIPEDIA
The Needleman-Wunsch algorithm performs a global alignment on two sequences. It is commonly used in bioinformatics to align protein or nucleotide sequences. The algorithm was proposed in 1970 by Saul Needleman and Christian Wunsch in their paper “A general method applicable to the search for similarities in the amino acid sequence of two proteins.”
The Needleman-Wunsch algorithm is an example a of dynamic programming, and is guaranteed to find the alignment with the maximum score. Needleman-Wunsch is the first instance of dynamic programming being applied to biological sequence comparison.
END EXCERPT FROM WIKIPEDIA

```
APPLT NEEDLEMAN TO PATCHED MS PROGRAM TO
EXTRACT DELTAS ALSO PROVIDE ALGORITHM CODE
```

41 Smith-Waterman and BLAST

BEGIN WIKIPEDIA EXCERPT

The Smith-Waterman algorithm is a well-known algorithm for performing local sequence alignment; that is, for determining similar regions between two nucleotide or protein sequences. The algorithm was first proposed by Temple Smith and Michael Waterman in 1981. Like the Needleman-Wunsch algorithm, on which it is a variation, Smith-Waterman is a dynamic programming algorithm. As such, it has the desirable property that it is guaranteed to find the optimal local alignment with respect to the scoring system being used (which includes the substitution matrix and the gap-scoring scheme). However, the Smith-Waterman algorithm is fairly demanding of time and memory resources: in order to align two sequences of lengths m and n, O(mn) time and space are required. As a result, it has largely been replaced in practical use by the BLAST algorithm; although not guaranteed to find optimal alignments, BLAST is much more efficient.

Examples of other questions that researchers use BLAST to answer are

Which bacterial species have a protein that is related in lineage to a certain protein whose amino-acid sequence I know?

Where does the DNA that I've just sequenced come from?

What other genes encode proteins that exhibit structures or motifs such as the one I've just determined?

END WIKIPEDIA EXCERPT

```
APPLY SMITH-WATERMAN ALGORITHM TO MS PROGRAMS
TO EXTRACT DELTAS AND PROVIDE ALGORITHM CODE
```

42 Sequence Alignment and Domain Shuffling

One complication of molecular evolution that causes problems for sequence alignment is domain shuffling.

Domain shuffling is the rearrangement of segments of one or more genes, each of which codes for a different structural domain in the gene product.

In binary executables, the similar effect can be seen in that certain compiler optimizations may move program code and rearrange function order in memory, sometimes even breaking single functional blocks up into non-contiguous sections of program code.

This breaks the original assumption of modularity in binary difference analysis.

Performing binary difference analysis on semantically isomorphic code different only by compiler optimizations will yield numerous changes in its sequence.

This is a form of syntactical symmetry breaking.

Even domain shuffling was a problem tackled by biologists 20 years ago.

30 years later… reverse engineering is barely moving past IDA scripts.

43 Program Flow Graphs

In their most basic form, graphs are a collections of nodes connected together with edges.

Graphs have many applications, such as in production, delivery and shipment logistics, network flow analysis, internet routing, traffic analysis and population migration.

In computer science they have been historically used to model program flow and finite state machines.

Flow diagrams, flowcharts, are used to visually describe the execution logic of a program.
Flowcharts are typically created to assist in the development and design of an application, but it has become increasingly apparent that they can provide much use in static binary analysis a la Halvar Flake.
The main idea is that if a visual flowchart of program execution can automatically be created from a compiled binary then the program code can be more easily understood.
Flowcharts address the inherent non-linearity of a complete program
They can account for domain shuffling.

44Program Roadmaps

A program can be thought of as a city roadmap, where intersections are conditional decisions where execution path branching occurs.
The actual path from point A to point B is a linear sequence of events where at each intersection only a single choice was made, so in that sense program execution is linear, but the actual space in which the program execution took place is not.
Because storage is linear, modeling this roadmap must be done in some linear manner.
A single program can be represented in many different ways.
The flow graphs of programs may be structurally isomorphic with difference analysis would imply they were very different.
Programs can be analyzed and compared in terms of their graph structure.
Vulnerability research and discovery may be performed by searching a program flow graph for sub-graphs highly correlated with a vulnerable conditions.

45Comments on Graph Isomorphism

Graph isomorphism falls just short of being an NP problem yet cannot be classified as P.
Program flow is a restricted class of graphs.
Program flows can often be represented as FSMs, are directed, can often be planar, have entry points thus can be pinned down for at least one node, and can artificially be restricted to a constant node degree ceiling by insertion of NOPs at collided jump target addresses.
These characteristics severely reduce the class of graphs into very manageable and nicely behaved class of graphs that most can be proven to be P time testable for isomorphism.

46Sequence Motifs and Binary Analysis

In bioinformatics, a sequence motif is a widespread nucleotide or amino-acid sequence that is believed to have some sort of functional significance.
In program code, standard functional calls, such as printf(), are quite common.
In a program flow graph with inlined CALLs, the sequence generated by these common function calls can considered to be sequence motifs.
Rules that define sequence motifs that can vary in structure are often created.
Rules are actually expressed as regular expressions, not unlike POSIX regex.
Probabilistic rules in the form of Markov Models are even created.

These regular expression rules are very similar to the way that binary analysis is performed to find simple vulnerable conditions.
For example, searching for strcpy() as the sign of a buffer overflow is done by searching for the sequence motif it generates in binary program code.
Many tools and IDA scripts automate this process and have even been quite effective in finding numerous security holes.

47Lexical Cohesion

Lexical cohesion is the characteristic continuity of semantic mean of text.
Lexical cohesion is a direct result of a body of writing "being about the same thing."
For example, a group of sentences in a paragraph are not random and disconnected.   Rather, they coherently "stick together."
The study of cohesion, lexical or otherwise, is quite possibly the most important component of linguistics.
There are two main types of cohesive relationships, reiteration and collocation, and Halliday and Hasan go further to classify cohesion into five categories based on specific word dependency relationship.
Reiteration and collocation can be described in effectively simpler terms.
Reiteration involves the repetition of lexical items across sentences.

48Three examples of reiteration would be the following sentence pairs.

```
The fly isn't moving.  The fly is dead.
The fly isn't moving.  The bug is dead.
The fly isn't moving.  It is dead.
```

Notice that the reiteration of the word 'fly' can be through repetition or replaced by superordinates or pronouns.

49Collocation is a more broad association between lexical items.

Collocation occurs any time that a pair of words occurring near each other is related in some way. Two examples of collocation are as follows.

```
I love cats. I hate dogs.
I'm using the computer.  I'm typing a paper.
```

Love and hate are antonyms but are related to each other by being members of some classifiable semantic set.
Cats and dogs also have a clear semantic relation.
Computer and typing are also semantically related, although are as clearly classifiable, as the action of typing is not unique to the use of computers.
Lexical cohesion usually occurs over a sequence of sentences all spanning some central topic of discussion and these sequences are known as lexical chains.
Lexical cohesion is important because it provides a clear contextual aid in the interpretation of otherwise potentially ambiguous terms, and most importantly it provides coherence and structure in discourse and thus helps construct a larger meaning in text.
Lexical cohesion has a number of applications in text classification.

50 Collocation and WordNET

One valuble preexisting lexicon for lexical cohesion and collocation is the thesaurus.
A thesaurus is a network of nodes (word) and edges (relations).
Semantic relation can be measured in terms of distance in a thesaurus.
Subject and topic transition and detection can be performed using this information by finding dense word cluster representation across any given input.

51 Collocation and functional binary analysis

An interesting application of the collocation concept would be in functional binary analysis.
Building a thesaurus-like lexicon based on commonly collocated sequence motifs can potentially help automate code subject detection.
Associating higher level ideas to code blocks would increase efficiency of analysis.
Because tokenizing code into sequence motifs is not nearly as cut-and-dry as tokenizing words from language, building a relational lexicon is a rather dubious task.

52 In Closing

Unfortunately, there was simply not enough time to fully articulate each idea and fully describe each algorithm presented.
Additional detailed information can be found throughout www.the-mathclub.net
A book is on the way expanding on these concepts and many others.
Feel free to come up and talk to me anytime during the conference.
Questions now?

53 References and Tools

Carlos M Nash. "Cohesion and Reference in English Chatroom Discourse." Proceedings of the 38th Hawaii International Conference on System Sciences. 2005

D. Michie, D.J. Spiegelhalter, C.C. Taylor. "Machine Learning, Neural and Statistical Classification." 1994

Jane Morris. "Lexical Cohesion Computed by Thesaural Relations as an Indicator of the Structure of Text." Computational Linguistics Volume 17, Number 1. 1991

Hang Li. "Generalizing Case Frames Using a Thesaurus and the MDL Principle." Computational Linguistics Volume 24, Number 2. 1998.

A.A. Tsonis, C. Schultz, and P.A. Tsonis. “Zipf’s Law and the Structure and Evolution of Languages,” Complexity, 2(5): pp. 12-13, 1997.

B. Mandelbrot: Adaptation d’un message a la linge de transmission. I & II. Comptes rendus (Paris). 232: pp. 1638–1640 and 2003–2005. 1951.

Viktor Pekar. "Modeling Semantic Coherence from Corpus Data: The Fact and the Frequency of a Co-occurrence." Coyote Papers 12, 1-8 Language in Cognitive Science. 1999.

Z. K. Silagadze. "Citations and the Zipf-Mandelbrot's law." arXiv:physics/9901035 v2 26 Jan 1999.

Mary Cook. "Experimenting to Produce a Software Tool for Authorship Attribution." 2003.

Dario Benedetto, Emanuele Caglioti, Vittorio Loreto. "Language Trees and Zipping." arXiv:cond-mat/0108530 v2 19 Dec 2001.

Reka Albert and Albert-Laszlo Barabasi. "Statistical mechanics of complex networks." REVIEWS OF MODERN PHYSICS, VOLUME 74, JANUARY 2002.

Craig G. Nevill-Manning, Ian H. Witten: Identifying Hierarchical Strcture in Sequences: A Linear-Time Algorithm. Journal of Artificial Intelligence Research (JAIR), Volume 7. 1997.

Halvar Flake. "Graph-Based Binary Analysis." Blackhat USA. 2003.

Halvar Flake. “More fun with Graphs.” Blackhat Federal. 2003.

"MEART: The Semi-Living Artist." http://www.fishandchips.uwa.edu.au/

Marshall Beddoe. "Protocol Informatics." Toorcon. 2004.

Richard F. Voss. "Evolution of long-range fractal correlations and 1/f noise in DNA base sequences." Phys Rev Lett. 1992 Jun 22;68(25):3805-3808.

G. Caldarelli, R. Marchetti and L. Pietronero. "The fractal properties of Internet." Europhys. Lett., 52 (4), pp. 386–391 (2000)

Wentian Li. "Large-Scale Patterns in DNA Texts." 1999.

Damian H. Zanette. "Zipf's law and the creation of musical context." arXiv:cs.CL/0406015 v1 7 Jun 2004.

Zipf, G. K. "The Psycho-Biology of Language." Boston: Houghton Mifflin. (1935)

Zipf, G. K. "Human Behaviour and the Principle of Least Effort." Cambridge, MA: Addison-Wesley. (1949)

Papoulis, A. "Bayes' Theorem in Statistics" and "Bayes' Theorem in Statistics (Reexamined)." §3-5 and 4-4 in Probability, Random Variables, and Stochastic Processes, 2nd ed. New York: McGraw-Hill, pp. 38-39, 78-81, and 112-114, 1984.

Filotti, Mayer. "A polynomial time algorithm for determining isomorphism of graphs of fixed genus" In Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing, pages 236-243, 1980.

Miller: "Isomorphism testing for graphs of bounded genus", In Proceedings of the Twelfth Annual ACM Symposium on Theory of Computing, pages 218-224, 1980.

Hopcroft, Wong: "A linear time algorithm for isomorphism of planar graphs" In Proceedings of the Sixth Annual ACM Symposium on Theory of Computing, pages 172-184, 1974.

Luks: "Isomorphism of graphs of bounded valence can be tested in polynomial time" Proc. 21st IEEE FOCS Symp., 1980, 42,49

Hoffmann: "Group-Theoretic Algorithms and Graph Isomorphism" Lecture Notes in Computer Science 136, Springer 1982 (Chapter V).

Babai,Grigoryev,Mount: "Isomorphism of Graphs with Bounded Eigenvalue Multiplicity" In Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing, pages 310-324, San Francisco, California, 5-7 May 1982.

Nelson ,et. al. "Automatically generating a topic description for text and searching and sorting text by topic using the same." United States Patent 5,937,422. Assignee: USA as represented by the NSA. August 10, 1999

TreeView. Tree drawing software for Apple Macintosh and Windows. http://taxonomy.zoology.gla.ac.uk/rod/treeview.html
GraphViz. Graph Visualization Software. http://www.graphviz.org/
BLAST. http://www.ncbi.nlm.nih.gov/blast/
ClustalW. A general purpose multiple sequence alignment program for DNA or proteins. http://www.ebi.ac.uk/clustalw/
SEQUITUR. inferring hierarchies from sequences. http://sequitur.info/
I,Sushi. Identifying Hierarchical Structure in Unordered Sets. http://the-mathclub.net/index.php/Identifying_Hierarchical_Structure_in_Unordered_Sets.
SpamAssassin. http://spamassassin.apache.org/
NMAP. http://www.insecure.org/
GNUPLOT. http://www.gnuplot.info/
ALION. Pairwise Alignment. http://motif.stanford.edu/alion/
WordNET. http://wordnet.princeton.edu/