Diplomarbeit

Grafische Darstellung und Analyse der Gruppendependenzen von Websites mit maliziösen Inhalten

René Soller

Betreut durch
Prof. Dr. Klaus Brunnstein
und Prof. Dr. Rüdiger Valk

Arbeitsbereich
Anwendungen der Informatik in
Geistes- und Naturwissenschaften
Universität Hamburg
Vogt-Kölln Straße 30
22527 Hamburg

Zusammenfassung

Die vorliegende Arbeit befasst sich mit der grafischen Darstellung und der Analyse von Netzstrukturen. Hierbei wird insbesondere die Struktur von Websites, die maliziöse Inhalte zum Download anbieten, betrachtet.

Die Diplomarbeit ist in einen theoretischen und einen praktischen Teil gegliedert: Im theoretischen Teil werden Grundlagen der Graphentheorie, Malware, Netzstruktur und Visualisierung vermittelt und die zugrundeliegende Problematik diskutiert. Im praktischen Teil folgt dann die Erläuterung der Implementation verschiedener Programme zur Visualisierung von Netzstrukturen sowie ein Programm zum automatischen Überprüfen von Webpages auf Veränderungen. Diese Programme befinden sich in der Anlage[1] zur vorliegenden Arbeit in Form von ausführbarem Code sowie als Quelltext inklusive auswertbarer Daten. Auch die generierten Ergebnisdateien sind dort zu finden.

[1] Da die Verbreitung von Malware unerwünscht ist, ist auch die Veröffentlichung von Websites, die Malware zum Download anbieten (bzw. deren URLs), problematisch, da dies die Verbreitung fördern würde. Aus diesem Grund werden die URLs, die mit den aus dieser Diplomarbeit hervorgegangenen Programmen verarbeitet werden, nicht veröffentlicht. Aus Gründen des Copyrights und da die Programme ohne die oben genannten sensiblen Daten nutzlos wären, werden auch die Programme nur auf berechtigte Anfrage herausgegeben. Nur der „MWC-Visualizer“ wird mit fikitven Beispieldaten und eingeschränkter Funktionalität veröffentlicht.

Inhalt

Abbildungsverzeichnis

Abkürzungsverzeichnis

AGN	- Arbeitsbereich „Anwendungen der Informatik in Geistes- und Naturwissenschaften“
ASCII	- American Standard Code for Information Interchange
GIF	- Graphics Interchange Format, verlustfrei komprimiertes Bildformat
GML	- Graph Modelling Language, Dateiformat zur Speicherung von Graphen
HTML	- Hypertext Markup Language
HTTP	- Hypertext Transfer Protocol
IP	- Internetprotokoll
ISP	- Internet Service Provider
JPEG	- Joint Photografic Expert Group, komprimiertes Bildformat
MWC	- Malware Crawler
NAI	- Network Associates Inc.
TCP	- Transmission Control Protocol
URI	- Universal Resource Identifier
URL[2]	- Universal Resource Locator
VRML	- Virtual Reality Modelling Language
VTC	- Virus Test Center
W3C	- World Wide Web Consortium
WWW	- World Wide Web

Legende

Text	- Standardtext
Text	- Hervorgehobener Text
Text	- Zitate
Text	- Definitionen
Text	- Formeln
`Text`	- Programmcode oder HTML-Code
`Text`	- Links im HTML-Code
Text	- Links
Text	- Beispiel-Links[3]

[2] Eigentlich *der* URL, da umgangssprachlich in den meisten Fällen aber die weibliche Form verwendet wird, wird hier ebenfalls die weibliche Form verwendet.
[3] Diese Links sind frei erfunden, die Existenz ist aber nicht auszuschließen.

1 Einleitung

Schon lange vor der Zeit des Computers und des Internets wusste man die Visualisierung zu nutzen und zu schätzen. Bereits 1637 sagte der Philosoph und Mathematiker René Descartes:

„Imagination oder Visualisierung und besonders die Benutzung von Diagrammen haben einen entscheidenden Anteil an der wissenschaftlichen Forschung.“ [VIS]

Von René Descartes stammt auch eine sehr wichtige Visualisierung in der Mathematik: Das Kartesische Koordinatensystem. Natürlich gibt es auch weniger bedeutende Darstellungsformen, jedoch ist die Visualisierung meist die beste Möglichkeit, abstrakte Sachverhalte verständlich darzustellen.
Die Verbreitung von Viren nimmt kontinuierlich zu. Auch die Größe und Komplexität von Malware insgesamt steigt, so dass die Anti-Viren Hersteller immer mehr Zeit benötigen, Gegenprodukte zu entwickeln. Um diese kritische Zeitspanne zwischen Entdeckung und Entwicklung von Anti-Viren zu umgehen oder zumindest zu minimieren, wäre es gut, Malware schon vor ihrer großflächigen Verbreitung analysieren zu können.
Bei dem World Wide Web handelt es sich um einen abstrakten Sachverhalt, der durch eine passende Visualisierung anschaulich dargestellt werden kann. In der vorliegenden Arbeit werden Visualisierungstechniken eingesetzt, um Websites mit maliziösen Inhalten grafisch darzustellen. Dabei geht es nicht um virenverseuchte Dateien, die z.B. versehentlich auf Websites veröffentlicht wurden und bei denen nicht erkennbar ist, dass es sich um maliziöse Software handelt. Vielmehr geht es um solche, die absichtlich und mit dem Ziel der Verbreitung angeboten werden.
Hierbei werden nicht nur statistische Visualisierungen betrachtet, die einen Überblick über die angebotene Malware und über die Anzahl liefern, sondern

auch die Verteilung der Websites im World Wide Web, die solche Software zum Download anbieten.
Durch die Analyse der Strukturen dieser Websites werden Abhängigkeiten sichtbar, die Aufschluss über die Verteilung und Gruppeneigenschaften geben. Dadurch wird ein schnelleres Aufspüren der Websites ermöglicht. Bisher unbekannte Viren können schon vor ihrer Ausbreitung gefunden und passende Gegenmaßnahmen entwickelt werden.

1.1 Ziele und Grenzen

Ziel dieser Arbeit ist die grafische Darstellung von Websites, die maliziöse Inhalte bereitstellen. Es werden verschiedene Möglichkeiten der Visualisierung aufgezeigt und diskutiert. Insbesondere werden die Netzstrukturen dieser Sites analysiert.
Des Weiteren werden gefundene Netzstrukturen überwacht und Änderungen protokolliert. Durch die Visualisierung ist es möglich, sich einen besseren Überblick über die Verteilung der Websites mit maliziösen Inhalten im WWW zu verschaffen. Das Sperren derartiger Seiten kann manchmal veranlasst werden, was aber ein erneutes Erscheinen an anderer Stelle nicht ausschließen kann.
Eine bedeutende Grenze sind die auszuwertenden Rohdaten. Nur in den Rohdaten enthaltene Informationen können auch visualisiert werden.

1.2 Bestehende Visualisierungen

Es gibt bereits eine Vielzahl an Visualisierungen für WWW- und Verzeichnisstrukturen mit unterschiedlichen Ansätzen und Zielsetzungen.
Für Dokumente oder Verzeichnisse, die sich linear anordnen lassen, bietet sich die Darstellung auf sogenannten **Perspective Walls** an. Hierbei werden die einzelnen Dokumente auf einer „geknickten“ Wand angeordnet.

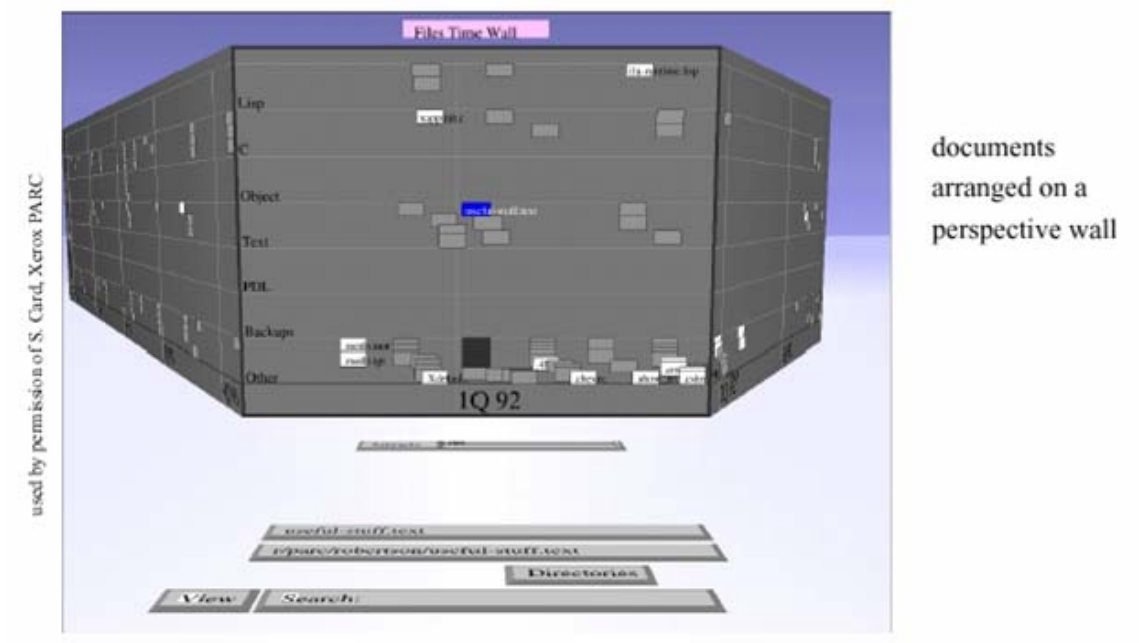


Abbildung 1 - Perspective Wall (aus [Mackinlay91])

Die Wand lässt sich drehen, so dass nicht alle Dokumente gleichzeitig angezeigt werden müssen. Man kann sie jedoch schemenhaft an den abgeschrägten Wänden erkennen und erhält so einen groben Überblick über die restliche Struktur.

Des Weiteren wird versucht, sogenannte Sitemaps, d.h. Übersichten über den Inhalt und Aufbau einer Website, grafisch zu gestalten. Diese Teilstrukturen des WWW sind meist baumartig aufgebaut und dadurch leicht darzustellen. Eine geeignete Darstellungsform sind **Hyperbolic Trees**. Hierbei wird der Verzeichnisbaum so dargestellt, als blicke man durch ein starkes Weitwinkelobjektiv. Durch Klicken auf einen Randbereich wird dieser zum Mittelpunkt verschoben. Dadurch läßt sich leicht navigieren und man verliert auch bei größeren Websites nicht den Überblick. Diese Darstellungsform eignet sich allerdings nur bei Netzen und nicht bei allgemeinen Graphen.

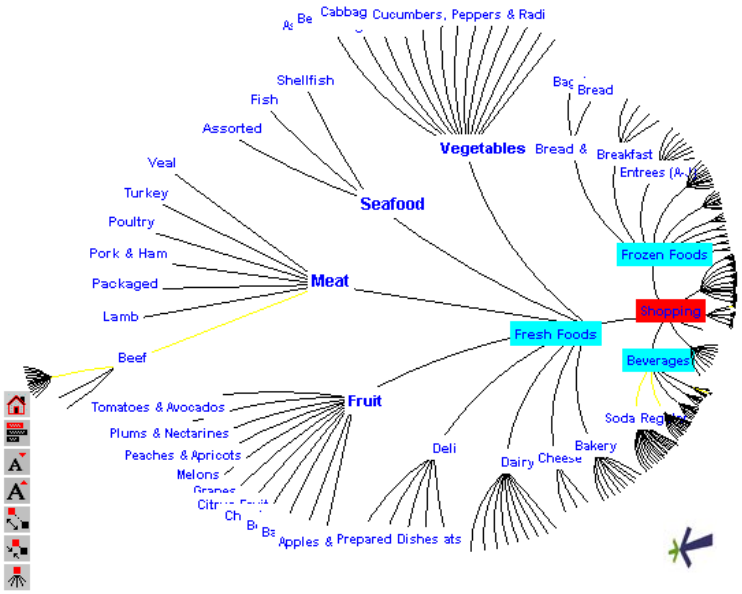


Abbildung 2 - Hyperbolic Tree (implemetiert von Inxight [Inxight])

Eine weitere gut geeignete Darstellung von Verzeichnisbäumen **bieten Cone Trees**. Hierbei wird die Baumstruktur dreidimensional dargestellt. Von einem Ausgangspunkt ausgehend verzweigen die jeweils untergeordneten Knoten kegelförmig um den Ausgangsknoten. Durch Rotation können die hinten liegenden Knoten in den Vordergrund gebracht werden. In der Studienarbeit „Generierung von Cone Trees mit VRML“ von Glen Lehnert [Lehnert] wird die Generierung von Cone Trees mit VRML beschrieben.

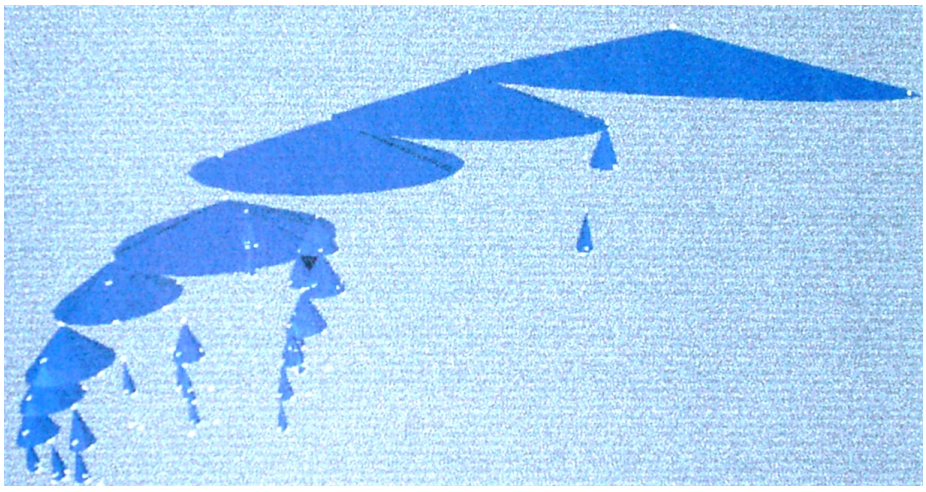


Abbildung 3 - Cone Tree (aus [Lehnert])

Sitemaps bilden nur die jeweilige Website ab, also einen sehr kleinen Teil des WWW. Visualisierungen größerer Teile des Internets findet man im Online Cyberspace-Atlas [Cyber]. In der folgenden Abbildung werden beispielsweise Internetverbindungen visualisiert (siehe [Lumeta]).

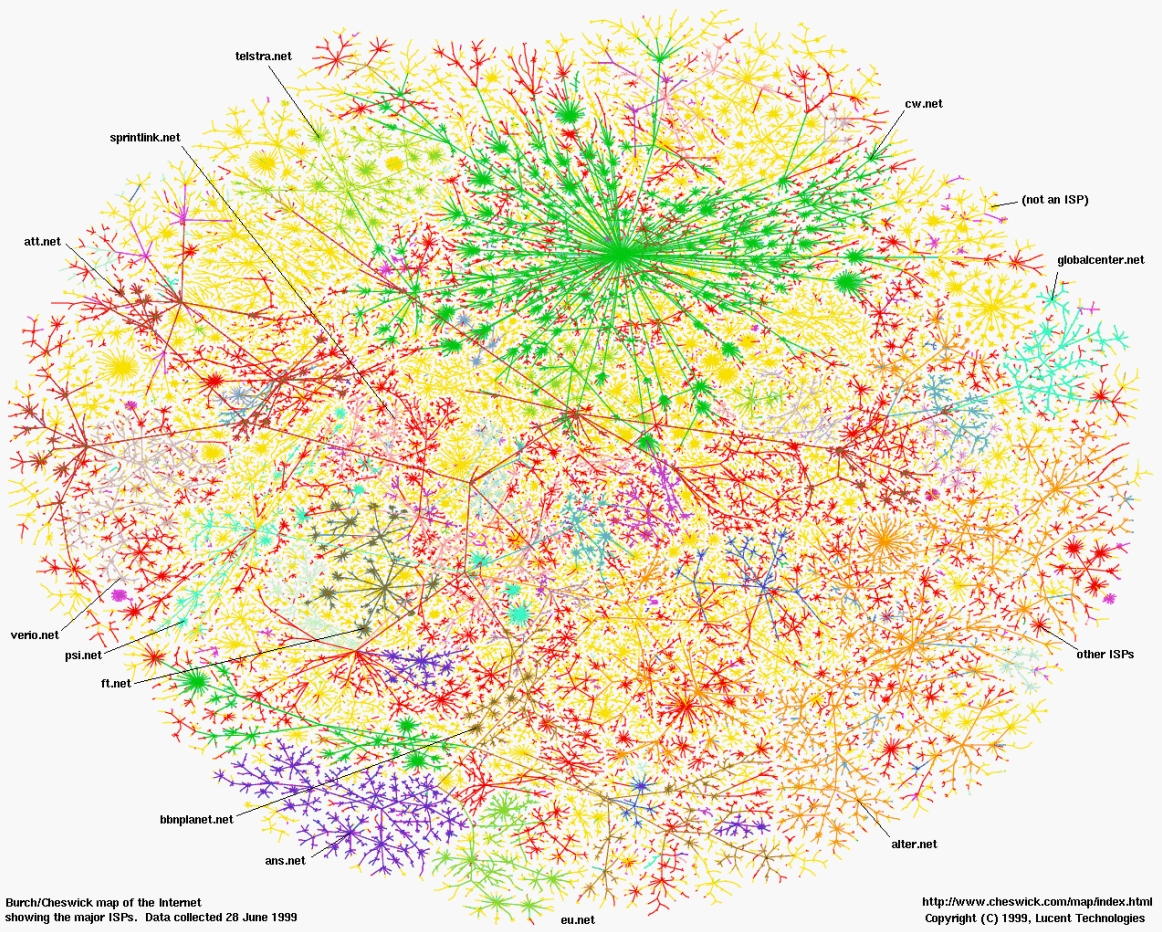


Abbildung 4 - Internetverbindungen

Ausgehend von einem Startpunkt wurden hier Routen zu einer Vielzahl von festgelegten Endpunkten (meist Webserver) verfolgt. Die gesammelten Daten (knapp 100.000 Knoten) wurden mit graphentheoretischen Algorithmen positioniert und kreisförmig dargestellt. Die verschiedenen Internet Service Provider (ISP) wurden durch unterschiedliche Farben gekennzeichnet.

2 Grundlagen

Im Folgenden werden kurz die in der vorliegenden Arbeit benutzten Grundlagen erläutert. Zum Einen werden die verschiedenen Malwarearten erläutert, die auf den dargestellten Websites zu finden sind. Dann wird der Begriff der Visualisierung und Darstellungstechniken beschrieben. Im Anschluss daran folgt die Beschreibung der WWW-Struktur und der Funktionsweise des AGN-Malware Crawlers, der die Rohdaten liefert. Zum Schluss werden einige Grundlagen und Definitionen der Graphentheorie erläutert.

2.1 Malware

Malware ist die Kurzform von malicious software[4]. Malware ist also der Oberbegriff für schädigende Software, d.h. Software, die vom Anwender unerwünschte Aktionen durchführt. Unerwünschte Aktionen sind z.B. das Löschen von Dateien oder gar der gesamten Festplatte, aber auch das Verschicken einer E-Mail kann unerwünscht sein.
In den Antivirentests des Arbeitsbereiches AGN an der Universität Hamburg [AGN] wird Malware wie folgt definiert:

„Malware" indicates all classes of software which may (intentionally) have harmful effects such as changing software (e.g. through self-replication, such as viruses and worms) or affect and destroy programs and data.

Als die sechs wichtigsten Kategorien von Malware werden dort genannt:

Virus: Maliziöse Software, die sich mit Hilfe eines „Wirts“ (d.h. einem ausführbaren Objekt oder aktivem Inhalt) innerhalb gegebener Beschränkungen und Peripherie verbreitet.

[4] malicious software = (engl.) Böswillige Software.

Wurm: Maliziöse Software, die sich ohne „Wirt“ in einem Netzwerk repliziert. Es wird also im Gegensatz zum Virus kein fremdes Programm zur Verbreitung benötigt.

Trojanisches Pferd (Trojaner): Eine potentiell schädigende Funktion, welche sich nachteilig auf ein System, Programme und Daten auswirkt oder nachteilige Auswirkungen auf Netzwerkkomponenten hat.

Intended[5] Virus (Wurm): Ein Virus (oder Wurm), der sich nicht richtig repliziert (normalerweise auf Grund von Programmierfehlern des Autors) aber trotzdem schädigende Aktionen ausführen kann.

Dropper[6]: Software, welche einen Virus, Wurm oder Trojaner in einem System, Programm oder Netzwerk installiert.

Generator (=malware creation kit): Ein Programm zum generieren von Malware, d. h. von Virenvarianten.

Eine sehr übersichtliche Zusammenfassung der Kategorien in Form einer Grafik findet man in der Diplomarbeit „Charakterisierung und Kategoriesierung von Malware zur Qualitätssicherung beim Test von Anti-Malware-Software“ von Mario Ti□ak [Ticak]:

[5] intended = (engl.) geplant, beabsichtigt.
[6] drop = (engl.) Fallen lassen.

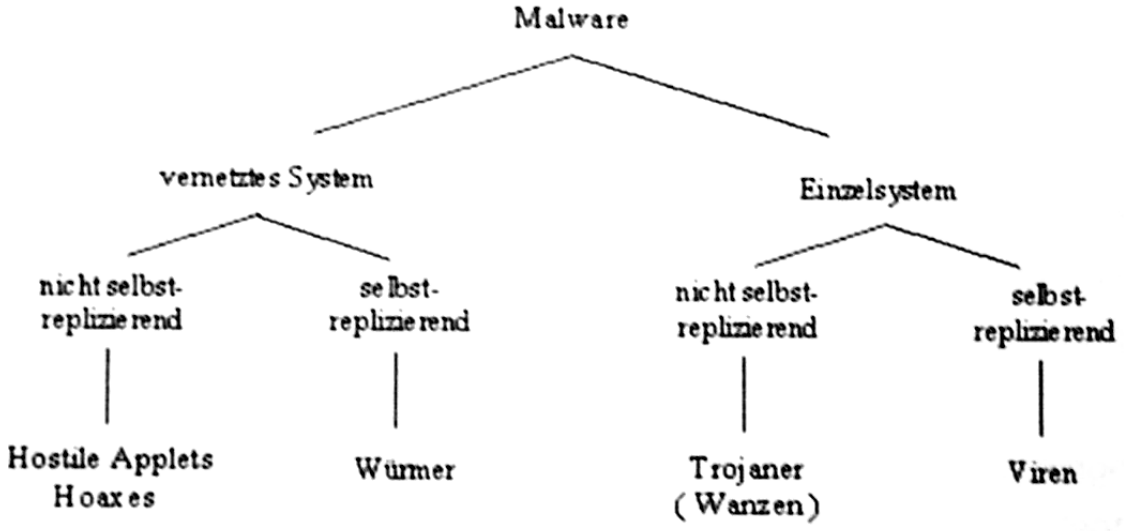


Abbildung 5 - Malwarekategorien

Die größte und bekannteste Malware-Kategorie sind Viren. Aus diesem Grunde folgt hier noch eine praxisnahe Definition von Viren aus der Doktorarbeit „Methodology of Computer Anti-Virus Research" von Vesselin Vladimirov Bontchev [Bontchev98]:

A computer virus is a sequence of symbols interpretable by a computer which, when interpreted by this computer, attaches itself to other interpretable symbol sequences in such a way that they become able to recursively spread the (possibly modified) initial sequence further.

Viren werden üblicherweise wiederum in Kategorien eingeteilt. Die Einteilung kann anhand unterschiedlicher Merkmale erfolgen, z. B. anhand des Infektionsmechanismus, der Speicherresistenz oder anhand der Schadfunktion. Eine der gebräuchlichsten Einteilungsarten ist die Kategorisierung anhand des Typs der infizierten Objekte, die im Folgenden beschrieben wird (siehe [Bontchev98]).

***Bootvirus*:** Virus, der sich in den Bootsektor eines Systems kopiert. Bei jedem Booten des Systems wird der Virus ausgeführt. Bootviren nutzen aus, dass im Bios das Booten von Diskette meist die höchste Priorität besitzt, d. h. es wird nur von der Festplatte gebootet, wenn sich keine Diskette im Laufwerk befindet. Vergisst man beim Herunterfahren des Systems versehentlich eine Diskette im Laufwerk, so wird beim nächsten Start von dieser gebootet. Falls es sich um eine infizierte Diskette handelt, wird das System nun infiziert. Da Dateien immer häufiger über das Internet anstatt durch Disketten ausgetauscht werden, hat die Anzahl neuer Bootviren sehr stark abgenommen.

***Dateivirus*:** Virus, der sich durch Anhängen[7] an ausführbare Dateien verbreitet. Der Virus verändert meistens die Einsprungadresse der Datei so, dass diese nun auf den Virus zeigt. Beim Ausführen der Datei wird dann neben dem eigentlichen Programm auch der Virus ausgeführt und verbreitet sich so.

***Dateisystemvirus*:** Virus, der nicht einzelne Dateien infiziert, sondern das Dateisystem als ganzes. Hierbei werden Verzeichniseinträge so verändert, dass sie auf den Virus zeigen. Es scheint dann so, als würden alle Dateien mit dem Virus anfangen.

***Multipartitvirus*:** Virus, der in der Lage ist, mehr als eine Objektart zu infizieren. Diese Viren infizieren also meist sowohl Dateien als auch Bootsektoren. Es bestehen dadurch für den Virus mehr Möglichkeiten einer Verbreitung, allerdings nimmt auch die Größe dieser Viren zu, was zu einer leichteren Entdeckung führt.

***Companionvirus (Doppelgängervirus)*:** Virus, der als Doppelgängerdatei eines Programmes fungiert. Versucht der Anwender dieses Programm auszuführen, so wird stattdessen der Virus gestartet. Dies kann durch

[7] oder Einfügen.

verschiedene Methoden realisiert werden. Einige Viren machen sich die Tatsache zunutze, dass unter MS-DOS Com-Dateien beim Ausführen eine höhere Priorität haben als Exe-Dateien. Der Virus sucht sich also eine Exe-Datei und kopiert sich in dasselbe Verzeichnis und mit dem gleichen Namen, jedoch mit .com als Endung. Versucht der Anwender nun die Exe-Datei zu starten, führt der Commandozeileninterpreter zuerst die Com-Datei mit dem Virus aus. Der Virus ruft anschließend die Exe-Datei auf.
Eine andere Technik besteht in der Umbenennung des Programms. Der Virus kopiert sich dann mit dem ursprünglichen Namen in das Verzeichnis.

2.2 Visualisierung

Da die grafische Darstellung einen wesentlichen Teil der vorliegenden Diplomarbeit einnimmt, werden im Folgenden einige Grundbegriffe der Visualisierung erläutert.
Der Begriff Visualisierung bezeichnet den Vorgang des Umwandelns von Daten in eine Grafische Darstellung.
Um unter den vielfältigen Möglichkeiten von Darstellungsformen eine geeignete auszuwählen, benötigt man Qualitätskriterien. Anhand der Qualitätskriterien kann dann eine passende Darstellungsart ausgewählt werden. Drei allgemeine Qualitätskriterien für eine Visualisierung werden „Visualisierung – Grundlagen und allgemeine Methoden" von in Heidrun Schumann und Wolfgang Müller [SchuMü00] erläutert: Eine Darstellung soll

- expressiv
- effektiv und
- angemessen

sein. Diese drei Punkte werden im Folgenden genauer erläutert.

2.2.1 Expressivität

Die Expressivität oder Ausdrucksfähigkeit der Visualisierung ist die wichtigste Anforderung an die Darstellung. Die Daten sollen unverfälscht wiedergegeben werden, d.h. die der Visualisierung zugrunde liegenden Daten müssen durch die Visualisierung repräsentiert werden. Folgende Abbildung macht dies an einem Beispiel deutlich:

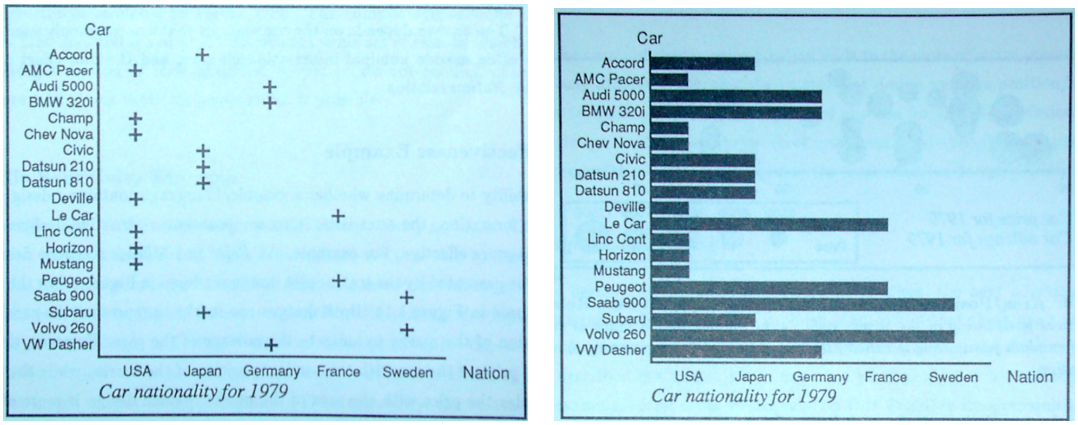


Abbildung 6 - Expressivität (aus [Mackinlay86])

In beiden Abbildungen sollen die Nationalitäten verschiedener Automarken dargestellt werden. In der ersten (expressiven) Abbildung werden den Marken in einer Matrix die jeweiligen Nationalitäten zugeordnet. In der zweiten Abbildung dagegen wird durch die Länge der Balken suggeriert, dass z.B. Autos aus Schweden größer oder besser sind, als Autos aus den USA. Diese Abbildung ist nicht expressiv.

2.2.2 Effektivität

Gibt eine Darstellung die repräsentierten Daten optimal wieder, so ist sie effektiv. Die Effektivität geht also auf die visuellen Fähigkeiten des Betrachters und auf die charakteristischen Eigenschaften des Ausgabegerätes ein. Zum Erstgenannten gehört beispielsweise die Auflösung oder die Wahrnehmungsgeschwindigkeit[8] des menschlichen Auges. Zum Zweiten gehört dagegen die Farbdarstellung sowie die Auflösung des Ausgabegerätes. Außerdem bedeutet Effektivität, dass die Visualisierung der Zielsetzung bestmöglich angepasst ist. Die Effektivität ist nicht von den Rohdaten, sondern von weiteren Faktoren, wie z.B. der Zielsetzung oder dem Betrachter und dessen Wahrnehmungseigenschaften, abhängig. Folgende Abbildung zeigt ein Beispiel einer effektiven und einer nicht effektiven Darstellung.

[8] Z.B. nimmt das Auge bei Animationen ab etwa 25 Bildern/Sekunde die Bilder nicht mehr einzeln wahr, sondern sieht sie als fließende Bewegung.

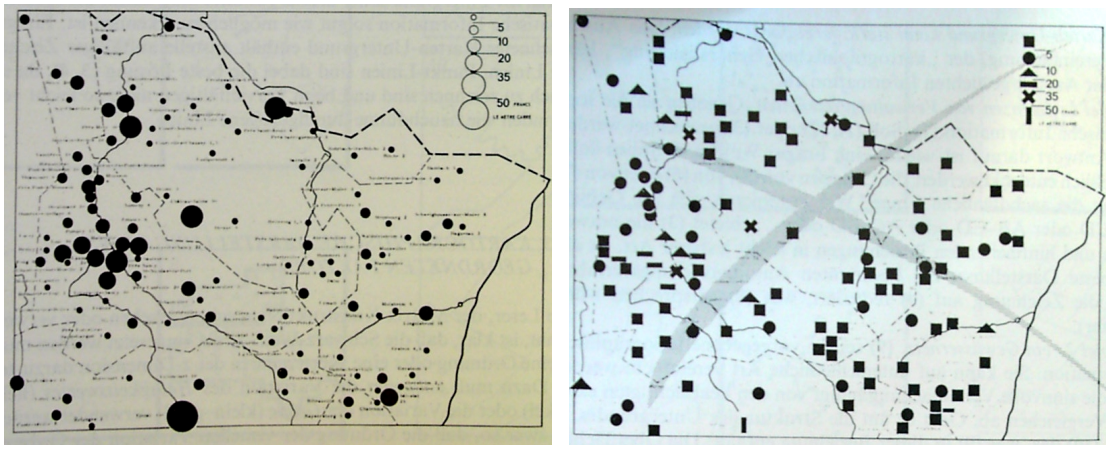


Abbildung 7 - Effektivität (aus [Bertin82])

In beiden Abbildungen sollen die Bodenpreise verschiedener Ortschaften dargestellt werden. In der ersten Abbildung wird die Höhe des Bodenpreises anhand der Größe der Kreise dargestellt. In der zweiten Abbildung werden dagegen hohe Preise durch unterschiedliche Symbole dargestellt. Dies ist nicht effektiv, da die Bodenpreise nicht wie bei der ersten Abbildung intuitiv verglichen werden können, sondern erst die Legende genauer studiert werden muss.

2.2.3 Angemessenheit

Die Angemessenheit bewertet als drittes Kriterium das Verhältnis zwischen dem Erreichen der beiden erst genannten Qualitätskriterien Expressivität und Effektivität und dem damit verbundenen Aufwand.
Die Angemessenheit beschreibt nicht die Qualität, sondern den Rechen- und Ressourcenaufwand, also den Kostenfaktor einer Visualisierung.

2.2.4 Visualisierungspipeline

Der gesamte Ablauf der Visualisierung von den Rohdaten bis zu den resultierenden Bilddaten läßt sich anhand einer Visualisierungspipeline verdeutlichen:

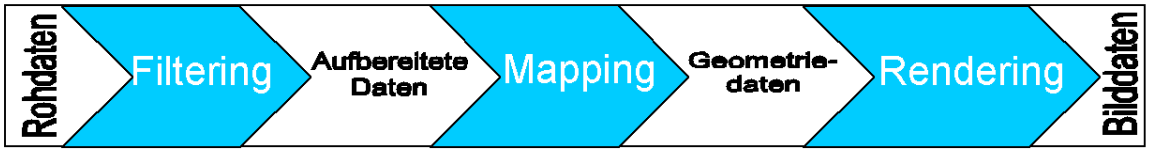


Abbildung 8 - Visualisierungspipeline

Die einzelnen blau dargestellten Verarbeitungsschritte der Visualisierungspipeline werden im Folgenden genauer erläutert. Anschließend wird noch speziell auf die verarbeiteten Daten, insbesondere die Rohdaten, eingegangen.

2.2.5 Filterung (Filtering)

Die erste Stufe ist die Datenaufbereitung (Filtering). Es handelt sich dabei um eine Daten-zu-Daten Abbildung. Hier werden die Rohdaten zum Einen vervollständigt, z.B. durch Interpolation. Zum Anderen findet aber auch eine Reduzierung statt. So werden beispielsweise Schwellwertoperationen durchgeführt oder Extremwerte bestimmt. Des Weiteren wird gerundet, was man ebenfalls als Reduzierung betrachten kann. Filtering ist häufig notwendig bei der Visualisiserung von Messdaten. Das Ergebnis des Filterings sind die aufbereiteten Daten, die im nächsten Schritt, dem Mapping, weiterverarbeitet werden.

2.2.6 Erzeugung der Geometriedaten (Mapping)

Der zweite Verarbeitungsschritt der Visualisierungspipeline ist das Mapping. Es handelt sich dabei um eine Daten-zu-Geometriedaten Abbildung. Hierbei werden den Daten, die bisher noch keine geometrischen Informationen enthalten auf geometrische Primitive (z.B. Größe, Länge, Form oder Farbe) abgebildet. In diesem Visualisierungsabschnitt muss besonders auf die

Expressivität und Effektivität geachtet werden. So entstehen die Geometriedaten, die an die nächste Stufe der Pipeline weitergegeben werden.

2.2.7 Bilddatengenerierung (Rendering)

Im letzten Verarbeitungsschritt werden dann aus den Geometrischen Daten die Bilddaten generiert (Geometriedaten-zu-Bilddaten Abbildung). Hierbei gibt es verschiedene Darstellungsarten:

- Realitätsnah: Die Daten werden möglichst realistisch dargestellt.
- Abstrahierend: Es wird von den Details abstrahiert. So können Daten z.B. als Sound wiedergegeben werden.
- Abstrakt (auch:“Mentale“ Bilder): Sowohl äußere als auch innere Strukturen werden in einem Bild wiedergegeben. Oft verborgene Eigenschaften können dadurch erkannt werden.
- Animationen: Durch das Hintereinandersetzen von Einzelbildern können oftmals beeindruckende und informative Bewegungen entstehen. Es sind auch Kombinationen mit den oben genannten Darstellungsarten möglich (z.B. realitätsnahe oder abstrahierende Animationen).

Nach der Bildgenerierung (Rendering) folgt die Bildausgabe auf dem Präsentationsmedium.
Die einzelnen Verarbeitungsabschnitte der Visualisierungspipeline werden bei der Implementation der Visualisierungen jeweils in einem eigenen Kapitel behandelt. Die in den Zwischenschritten entstandenen Daten werden dagegen nur bei Bedarf genauer erläutert.

2.3 Internet und WWW

Um die Struktur des World Wide Web zu verstehen, ist es hilfreich die Entstehungsgeschichte des Internets zu kennen. Aus diesem Grund wird im Folgenden kurz dargestellt, wie sich das Internet im Laufe der Zeit entwickelt hat.

2.3.1 Entstehungsgeschichte

Die Geschichte des Internet begann mit der Idee des US Verteidigungsministeriums zu einem dezentralen Netzwerk, um militärische Daten besser schützen zu können. 1964 gab die US Air Force bei der RAND Corporation das so genannte "dezentrale Netzwerk“ in Auftrag. Dieses Projekt scheiterte allerdings und wurde nie realisiert.

1966 griff die Advanced Research Projects Agency (ARPA), eine seit 1958 bestehende wissenschaftliche Einrichtung, deren Forschungsergebnisse in militärische Zwecke einflossen, die Idee wieder auf. Sie entschloss sich zur Vernetzung der ARPA-eigenen Großrechner. 1969 waren bereits vier Rechner der ARPA vernetzt. In den nächsten drei Jahren wuchs das Netz auf 40 Rechner, es war aber noch nicht öffentlich zugänglich und wurde deshalb auch ARPA-Net genannt. Anfang der 70er wurde das Netzwerk dann auf wissenschaftliche Einrichtungen erweitert, so dass nun auch einige Studenten Zugang zum Internet bekamen. Durch die unterschiedlichen Rechnertypen, die nun mit dem Internet verbunden waren, kam auch die Notwendigkeit eines einheitlichen Übertragungsprotokolls auf. Aus den Bemühungen um ein solches Protokoll ging schließlich das TCP/IP-Protokoll hervor. 1984 wurde das Domain Name System vorgestellt, durch das die schwer zu merkenden IP-Adressen, die nur aus Zahlen bestehen, durch leichter zu merkende Namensadressen übersetzt werden. Die einzelnen Teile solcher Namensadressen sind wie bei IP-Adressen durch Punkte voneinander getrennt. An letzter Stelle steht die Top-

Level-Domain. Dies ist entweder eine Typkennung (.org für Organisationen, .com für kommerzielle Seiten, usw.) oder eine Landeskennung (.de für Deutschland, .ch für die Schweiz, usw.). Davor steht die eigentliche Domain und davor wiederum können Sub-Domains stehen. Hinter der Top-Level-Domain stehen die Unterverzeichnisse und der Dateiname inkl. Dateiendung.

Bezeichnung der einzelnen Teile einer Internetadresse (siehe [IETF]):
Beispieladresse: www.subdomain.domain.net/verz1/verz2/index.html

Teil der Beispieladresse	Bezeichnung
www und subdomain	Sub-Domain
domain	Second-Level-Domain bzw. Haupt-Domain
net	Top-Level-Domain bzw. Domainendung
verz1	Unterverzeichnis der Tiefe 1
verz2	Unterverzeichnis der Tiefe 2
index.html	Name eines HTML-Dokuments / Dateiname
www.subdomain.domain.net	Domain
/verz1/verz2/	Verzeichnis

Die gesamte Internetadresse wird als URL bezeichnet.

So etwas wie die heutigen Browser gab es damals noch nicht. Erst Ende der 80er entwickelte Tim Berners-Lee ein kommandozeilenorientiertes Programm namens „Enquire“[9]. Es war der Vorläufer der heutigen Browser und Beginn des Projekts „World Wide Web“, das 1990 am Genfer Hochenergieforschungszentrum CERN begann. Die drei Grundsäulen des Konzepts von Tim Berners-Lee sind (siehe „World Wide Web Consortium - A Little History of the World Wide Web“ [w3c]):

1. die Spezifikation für die Kommunikation zwischen Web-Clients und Web-Servern - das sogenannte HTTP-Protokoll
2. die Spezifikation für die Adressierung beliebiger Dateien und Datenquellen im Web und im übrigen Internet - das Schema der so genannten URIs

[9] Enquire = (engl.) sich erkundigen.

3. die Spezifikation einer Auszeichnungssprache für Web-Dokumente, der Tim Berners-Lee den Namen HTML gab

1992 entstanden dann die ersten Browser „Erwise“ und „Viola“ mit grafischen Benutzeroberflächen. Verschiedene Internetdokumente konnten nun durch das einheitliche HTML-Format durch sogenannte Hyperlinks[10] verknüpft werden.

2.3.2 Struktur des WWW

Durch diese Verknüpfung der Dokumente im WWW entsteht ein Metanetz über dem Netz der physikalischen Verbindungen der Computer. So können auch HTML-Dokumente auf Computern, die nicht direkt miteinander Verbunden sind, direkt über Hyperlinks verknüpft sein.
HTML-Dokumente sind Textdateien, die durch bestimmte, in eckige Klammern eingeschlossene Steuerungssequenzen ihre Formatierung erhalten. Diese Sequenzen werden als **Tag** bezeichnet. So bewirkt beispielsweise `<BR>` einen Zeilenumbruch. Einige Tags umschließen Textteile. Das schließende Tag hat hinter der öffnenden eckigen Klammer einen Schrägstrich. Unterstrichenen Text bekommt man z.B. durch das Einschließen in `<U>` und `</U>`:

```
<U>Unterstrichener Text</U>
```

Außerdem gibt es Parameter, die hinter den Tags in der eckigen Klammer angegeben werden können. Die HTML-Dokumente, auf die durch Links verwiesen wird, sind in diesen Parametern angegeben. Ein einfacher Link könnte beispielsweise wie folgt aussehen:

```
<A HREF="www.verlinkteSeite.de">Verlinkte Seite</A>
```

Der Parameter `HREF="www.verlinkteSeite.de"` des Tags `<A>` gibt die Internetadresse an, die aufgerufen wird, wenn man im Browser auf den dargestellten Link `Verlinkte Seite` klickt.

[10] Hyperlinks werden auch oft einfach nur als Links bezeichnet.

2.4 MWC

Der AGN-Malware Crawler ist ein Resultat der Diplomarbeit „Webbasiertes Auffinden maliziöser Software mit fortschrittlichen heuristischen Verfahren“ von Sönke Freitag [MWC-00]. Im Folgenden wird die Funktionsweise des Crawlers kurz beschrieben. Bei Bedarf findet der Leser ausführlichere Informationen zum Malware Crawler in der oben genannten Diplomarbeit, die im WWW auf der Website des Fachbereichs Informatik, Arbeitsbereich AGN der Universität Hamburg [AGN] eingesehen werden kann.

Der AGN-Malware Crawler ist in drei grundlegende Komponenten unterteilt:

- Crawler
- Download-Modul mit Virenscan
- Surveiller

Die ersten beiden Komponenten werden im Weiteren genauer erläutert, die dritte Komponente ist erst später hinzugekommen und wird in Kapitel 3 genauer beschrieben.

2.4.1 Crawler

Beim Crawlen arbeitet der AGN-Malware Crawler ähnlich wie ein Browser. Begonnen wird mit einer Startseite oder Startliste, aus der alle Links, die Tags und der Text extrahiert werden. Dann werden die extrahierten Links nacheinander verfolgt und aus diesen Webpages[11] ebenfalls Tags, Text und Links nach Vorgabe der Crawl-Parameter extrahiert. Der Inhalt der Seite wird durch ein mehrstufiges heuristisches Verfahren bewertet und je nach Höhe

[11] Um eine Verwechslung mit dem Begriff „Website“ (=Zusammenschluss mehrerer HTML-Dokumente) zu vermeiden, wird hier „Webpage“ (=einzelnes HTML-Dkument) anstelle von „Webseite“ benutzt.

dieses Heuristikwertes werden die enthaltenen Links oder die zuvor extrahierten Links weiterverfolgt. Die Links werden in einer Datenbank gespeichert, so dass mehrfache Seitenaufrufe (Endlos-Schleifen) vermieden werden.

Während des Crawlens werden alle Dateiverweise von Seiten mit einem genügend hohen Heuristikwert in der „Filelist“ gespeichert. Um Angriffen von diesen Seiten zu entgehen, tarnt sich der AGN-Malware Crawler als Internet Explorer.

Es existieren folgende Tabellen:

Badlist: Tabelle mit Webpages, von denen bereits bekannt ist, dass sie maliziöse Software zum Download anbieten.

Goodlist: Tabelle mit Webpages, die vom Crawlen ausgeschlossen sind, z.B. Seiten, die fälschlicherweise einen hohen Heuristikwert erzeugen würden. So werden beispielsweise Seiten der Anti-Viren Hersteller von vornherein ausgeschlossen.

Scanlist: Tabelle der zu durchsuchenden HTML-Dokumente innerhalb einer Website (interne Links).

Extlist: Tabelle der Verweise auf externe Webpages (externe Links).

Filelist: Tabelle mit Verweisen auf Dateien, die später geprüft werden.

SurvList: Diese Tabelle wurde während der Entstehung der vorliegenden Arbeit neu hinzugefügt[12] und enthält Webpages, die periodisch auf Veränderungen untersucht werden (siehe „Surveiller“).

Der Ablauf des Crawlens im Überblick:

1.) Start-URL laden
2.) alle Links, die Tags und den Text extrahieren
3.) Die Seite heuristisch bewerten
4.) Falls der Heuristikwert einen Schwellwert erreicht: die enthaltenen Links weiterverfolgen und Dateiverweise speichern (weiter bei 2.)

2.4.2 Download

Die zweite wichtige Funktion des AGN-Malware Crawlers ist das Downloaden und Virenscannen. Hier werden die in der Tabelle „Filelist“ abgelegten Dateien geladen und mit einem Virenscanner im Heuristik-Modus[13] nach verdächtigen Inhalten untersucht. Findet der Virenscanner einen Virus oder eine verdächtige Datei, so sendet der AGN-Malware Crawler automatisch eine E-Mail.

Die Heuristik ist vierstufig, d.h. es werden nicht nur die HTML-Dokumente analysiert (Stufe I), sondern es wird auch bewertet, wenn eine Seite auf Seiten verweist, die in der „Badlist“ aufgeführt sind (Stufe II). In dieser „Badlist“ befinden sich Links auf Seiten, von denen bereits bekannt ist, dass sich auf ihnen maliziöse Inhalte befinden.

[12] ...und ist im September 2002 wieder entfernt worden. Es werden nun die Webpages der „Badlist“ vom Surveiller auf Veränderungen überprüft.
[13] Es wird nach Programmteilen gesucht, die Schaden anrichten könnten, oder die für die Replikation zuständig sind.

Die URL der analysierten Seite selbst fließt in Stufe III in den Heuristikwert mit ein. Da derzeit z.B. häufig auf russischen[14] und slovakischen Webpages maliziöse Inhalte zu finden sind, werden z.B. .ru - Domains höher bewertet als .net - Domains.
In der vierten und letzten Stufe wird schließlich noch die Entfernung der Seite zu anderen Seiten anhand der Verzeichnistiefe einbezogen, da sich viele Websites mit maliziösen Inhalten Domains teilen, beispielsweise „www.domain.net/ virenautor1/index.html“ und „www.domain.net/virenautor2/ index.html“.

In der ersten Stufe werden nicht einfach Schlüsselwörter gesucht und diesen Werte zugeordnet, die dann miteinander verknüpft werden. Vielmehr fließt auch die Nähe einzelner Schlüsselwörter zueinander mit in den Heuristikwert ein, so dass schon eine gewisse semantische Komponente vorhanden ist. Dazu erfolgt eine Aufteilung in Keywords und Qualifyer, wobei der Malware Crawler den Heuristikwert bei einigen Keywords anstatt zu erhöhen auch verringern kann, wie z.B. beim Wort „Virenscanner“.

[14] nicht zuletzt wegen der Gesetzeslage in Bezug auf Viren dort.

2.5 Graphentheorie

Dieses Kapitel gibt eine kurze Einführung in die für die vorliegende Arbeit benötigten allgemeinen mathematischen Definitionen und klärt wesentliche Grundbegriffe der Graphentheorie. Die folgenden Formeln werden später benötigt, um eine möglichst ausgewogene Darstellung des Graphen, der die Websites mit maliziösen Inhalten darstellt, zu errechnen. Nur so können evtl. vorhandene Strukturen erkannt und analysiert werden.

2.5.1 Allgemeine mathematische Definitionen

Zunächst einige allgemeine Definitionen aus der Mathematik, die nicht speziell zur Graphentheorie gehören. Mit $[M]^k$ bezeichnet man die Menge aller k-elementigen Teilmengen von M. Der ***Abstand*** zweier Punkte u und v **in der Ebene** wird definiert als

$$d(u,v) := \sqrt{(u_x - v_x)^2 + (u_y - v_y)^2} .$$

$|M|$ ist die Anzahl der Elemente der Menge M.

2.5.2 Graphentheoretische Definitionen

Ein ***Graph*** G ist ein Tupel (V,E), wobei V die Menge der Knoten[15] und E die Menge der Kanten repräsentiert.

Die Kanten sind wiederum gegeben durch Elemente aller zweielementigen Teilmengen von V ($E \subseteq [V]^2$). Abhängig von der (Un)endlichkeit von V heißt der Graph G ***endlich*** oder ***unendlich***. Unendliche Graphen sind zwar theoretisch möglich und auch sehr interessant, werden hier aber nicht weiter betrachtet.

[15] Die Knoten werden auch oft als Ecken bezeichnet.

Die Menge V ist beliebig. Allerdings ist es zur Visualisierung eines Graphen, der an sich nur ein mathematisches Modell ist, nötig, den Knoten Koordinaten zuzuweisen. Die Menge V wird also aus Punkten in der Ebene bestehen. Mit u_x und u_y werden die x- und y-Koordinaten bezeichnet.

Das führt dazu, dass zwei eigentlich „gleiche“ Graphen G_1 und G_2 als ungleich bezeichnet werden, nur weil sich die Mengen V_1 und V_2 (durch unterschiedliche Koordinaten der Knoten) unterscheiden. Aus diesem Grund werden Graphen als ***Isomorph*** definiert, wenn es eine bijektive Abbildung $\varphi: V_1 \rightarrow V_2$ gibt mit $\{u,v\} \in E_1 \Leftrightarrow \{\varphi(u),\varphi(v)\} \in E_2$ für alle $u,v \in V_1$. Isomorphe Graphen werden im Weiteren als „gleich“ bezeichnet.
Die formale Definition eines Graphen ist somit unabhängig von der grafischen ***Abbildung*** eines Graphen, wodurch es für einen Graph immer mehrere unterschiedliche Darstellungen gibt (siehe Abbildung). Umgekehrt gibt es aber für jede Abbildung genau einen zugehörigen Graphen. Die passende Abbildung ist je nach Zweckmäßigkeit zu wählen. Generell werden in der vorliegenden Arbeit Knoten als Punkte und Kanten als Verbindungslinien zwischen diesen dargestellt.

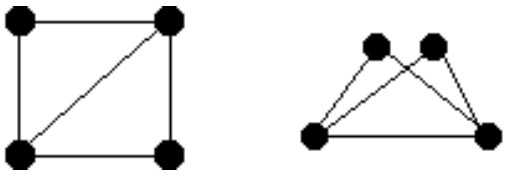


Abbildung 9 - Zwei verschiedene Abbildungen eines Graphen

Ein ***Weg*** (oft auch “einfacher Weg” genannt) der Länge k ist ein nicht leerer Graph $P=(V,E)$ mit $V=\{x_0,x_1,...,x_k\}$ und $E=\{\{x_0,x_1\},\{x_1,x_2\},...,\{x_{k-1},x_k\}\}$, wobei $x_i \neq x_j$ für alle $i \neq j$. Hierbei sind x_0 und x_k sind die ***Endknoten*** des Weges. Wege mit den gleichen Endknoten u und v bezeichnet man als u-v-Wege. Der ***Abstand*** $d_G(u,v)$ zweier Knoten u und v **in einem Graphen** wird definiert als die geringste Länge eines u-v-Weges in G und $d_G(u,v)$:=*unendlich*, falls kein eines u-v-Weg in G

existiert. Existiert für alle Knoten $u,v \in V$ mindestens ein u-v-Weg, so heißt der Graph ***zusammenhängend.*** Ist ein Graph nicht zusammenhängend, so nennt man die zusammenhängenden Teilgraphen ***Zusammenhangskomponenten***.

Graphen, die über sämtliche mögliche Kanten verfügen, d.h. $E = [V]^2$ heißen ***vollständig***. Unvollständige Graphen mit relativ wenigen[16] Kanten bezeichnet man als ***licht***, solche mit vielen Kanten als ***dicht***.
Graphen mit keinem oder nur einem Knoten nennt man ***trivial***.
Der ***Grad*** $d(v)$ eines Knoten ist die Anzahl der mit ihm verbundenen Kanten, also

$$d(v) = \sum_{x \in V, \{x,v\} \in E} (1 - eq(x,v)) \text{ mit}$$

$$eq(x,y) = \begin{cases} 0, für\_x \neq y \\ 1, für\_x = y \end{cases}$$

Der ***Durchschnittsgrad*** $d(G)$ eines Graphen G ist definiert als

$$d(G) := \frac{\sum_{x \in V} d(x)}{|V|}$$

Den höchsten vorkommenden Grad in einem nicht leeren Graphen nennt man ***Maximalgrad*** $(d_{max} := max\ \{d(v) \mid v \in V\})$.
Analog berechnet sich der ***Minimalgrad*** eines nicht leeren Graphen $(d_{min} := min\ \{d(v) \mid v \in V\})$.

In der Graphentheorie gibt es sogenannte ***gefärbte Graphen***. Hierbei wird zwischen Knotenfärbung und Kantenfärbung unterscheiden. Es handelt sich hierbei meist um die Fragestellung, mit wie vielen unterschiedlichen Farben ein

[16] Meist wird als Grenzwert |E|< |V| log |V| genommen.

Graph mindestens gefärbt werden muss, so dass je zwei benachbarten Knoten (bzw. Kanten) unterschiedlich gefärbt sind[17].

Gerichtete Graphen besitzen zusätzlich noch zwei Funktionen $init$:$E \rightarrow V$ und ter:$E \rightarrow V$, die jeder Kante eine Richtung zuweisen: $init(e)$ liefert den Anfangsknoten und $ter(e)$ den Endknoten einer Kante e.

Beim Grad eines gerichteten Graphen wird zwischen ***Eingangsgrad*** und ***Ausgangsgrad*** unterschieden. Der Ausgangsgrad d_a eines Knoten v ist die Summe aller ausgehenden Kanten, d.h.

$$d_a(v) = \sum_{x \in E} (eq(init(x), v) .$$

Analog ist der Eingangsgrad d_e eines Knoten

$$d_e(v) = \sum_{x \in E} (eq(ter(x), v) .$$

Ein Graph $G'=(V',E')$ mit $V' \subseteq V$ und $E' \subseteq E$ heißt ***Teilgraph*** von G. Gilt außerdem $\{x,y\} \subseteq E'$ *für alle* $(x,y \subseteq V'$ *und* $\{x,y\} \subseteq E)$ so nennt man den Graphen G' einen ***Untergraph*** von G.

[17] Diese Färbung ist nicht mit der weiter unten beschriebenen Färbung der Knoten aufgrund des Heuristikwertes zu verwechseln.

3 Implementation

Im Folgenden werden die implementierten Programme bzw. Programmteile aufgeführt. Die benutzten Algorithmen und die Funktionsweisen werden erläutert und es werden exemplarisch grafische Ausgaben der Programme dargestellt und ausgewertet.

3.1 Darstellung als Statistik

Um eine Vorstellung davon zu bekommen welche Malware wann am häufigsten verbreitet wird, eignet sich die Darstellung als Balkendiagramm sehr gut. Das Programm „MWC-Analyzer“ leistet dies. Dargestellt werden die Monate eines Jahres auf der x-Achse und die Anzahl der gefundenen Malware auf der y-Achse. Die Balken sind farblich unterteilt, um die unterschiedliche Malware eines Monats unterscheiden zu können. So kann man an der Höhe der Balken sehr leicht die Gesamtanzahl aufgetretener Malware eines Monats ablesen sowie anhand der Farben genauer nach Viren differenzieren. Die Grafik liefert das Programm in Form von HTML-Dateien, so dass die Ergebnisse problemlos veröffentlicht werden können. Das Programm stellt eine Erweiterung des „VPAnalyzer“ dar, das aus der Studienarbeit „Methoden und Verfahren zur Optimierung der Analyse von Netzstrukturen am Beispiel des AGN-Malware Crawlers“ [Soller01] hervorgegangen ist. Die Weiterentwicklung kommt so auch teilweise dieser alten Funktion zugute. Das erweiterte Programm kann zusätzlich zu der Jahresstatistik wie auch bei der Darstellung der in Newsgroups gefundenen Malware eine Monatsstatistik generieren. Dies ist jedoch vorerst nicht sinnvoll, da der Malware Crawler die Rohdaten in größeren Abständen liefert und der Zeitpunkt der Einbringung der Malware nicht bekannt ist.

3.1.1 Einlesen der Daten (Filtering)

Wie auch die vom NAI-Crawler, einem von Network Associates Inc. entwickelten Crawler zum Auffinden von Malware in Newsgroups (siehe [Soller01]), gelieferten Daten haben die Rohdaten vom Malware Crawler bzw. des vom Malware Crawler aufgerufen Virenscanners kein spezifiziertes Format. Die Dateien sind für das Lesen von einem Menschen optimiert und nicht dafür vorgesehen, maschinell verarbeitet zu werden. Das macht eine Anpassung der Einlesefunktion nötig, die aber aufgrund des unspezifizierten Formats sehr flexibel sein muss. Die vom AVP-Virenscanner generierte ASCII-Datei sieht wie folgt aus:

```
þ   AVPDOS32 Start  10-07-101 03:44:07

         Version 3.0  build 132
  Last update: 08.07.2001, 46873 records.

Command line: e:\temp\ /m /p /b /t=d:\ /w=e:\temp\report.txt /o /y /s /* /v
Profile defdos32.prf (from 1999-05-25 12:28:56)

e:\temp\MEMBER~1.ZIP      archive: ZIP
e:\temp\MEMBER~1.ZIP/FreeStyler.a.doc  infected: Macro.Word97.Free
e:\temp\MEMBER~1.ZIP/FreeStyler.a.doc  archive: Embedded
e:\temp\MEMBER~1.ZIP/FreeStyler.a.doc/FreeStyler    ok.
e:\temp\MEMBER~1.ZIP/FreeStyler.a.txt  ok.
e:\temp\REPORT.TXT  ok.

Current object: e:\temp

         Sector Objects :      0              Known viruses :      1
                  Files :      4               Virus bodies :      1
                Folders :      1                 Disinfected :      0
               Archives :      2                     Deleted :      0
                 Packed :      0                    Warnings :      0
                                                 Suspicious :      0
    Scan speed (Kb/sec) :     17                  Corrupted :      0
              Scan time :  00:00:01               I/O Errors :      0

Scan process completed.

Result for all objects:

         Sector Objects :      0              Known viruses :      1
                  Files :      4               Virus bodies :      1
                Folders :      1                 Disinfected :      0
               Archives :      2                     Deleted :      0
                 Packed :      0                    Warnings :      0
                                                 Suspicious :      0
    Scan speed (Kb/sec) :     17                  Corrupted :      0
              Scan time :  00:00:01               I/O Errors :      0
```

Abbildung 10 - Vom AVP-Virenscanner generierte Log-Datei

Die hervorgehobenen Informationen sind relevant: Das Datum[18] und die Namen der gefundenen Viren.

3.1.2 Auswertung der Daten (Mapping)

Das Einlesen der Rohdaten dauert aufgrund der Verteilung auf mehrere Dateien und aufgrund des Formats sehr lange. Deshalb werden die extrahierten Daten eines Monats zusammen in einer gemeinsamen Datei gespeichert. Sie müssen dann später, z.B. bei neuer Farbzuweisung, nicht erneut analysiert werden. Es werden nur die relevanten Daten in kompakter, zur maschinellen Auswertung optimierter Form gespeichert. Aus diesem Grund ist im Programm zwischen dem Einlesen der Daten und dem Erstellen der Statistik getrennt worden.

Diese Datei besteht aus *n* Zeilen mit folgendem Inhalt:

Anzahl der Bytes	Inhalt
2 Bytes	Tag des Monats
10 Bytes	Anzahl der aufgetretenen Malware
restliche Bytes bis Zeilenende[19]	Name der Malware

Folgend ein Ausschnitt einer solchen Datei:

```
...
130000000001Backdoor.InCommander.10.b
130000000001IRC-Worm.generic.bat
130000000001TrojanDropper.Win32.Joiner.j
130000000001Backdoor.DeepThroat.c
...
```

[18] Durch einen Jahr-2000-Fehler steht hier 101 anstatt 2001.
[19] ASCII-Zeichen Nr. 13 und Nr. 10.

Der Dateiname enthält jeweils das Jahr, den Monat und einen zur Unterteilung in Viren/Würmer (v) bzw. Malware[20] (m). Bisher nicht in der Statistik aufgetretene Malware wird zunächst als „m“ eingeteilt, bei bekannter Malware erfolgt die Zuordnung dann automatisch. Vorangestellt ist ein y oder x, zur Kennzeichnung der Quelle[21]. Die Dateiendung ist „.dat“.

3.1.3 Farbauswahl und Generieren der HTML-Datei (Rendering)

In den resultierenden HTML-Dokumenten wird die Malware in je zwei Balkendiagrammen gezeigt. In der oberen Grafik werden die Trojaner und in der zweiten die Viren / Würmer dargestellt. Die Balken der Malware sind farblich unterscheidbar. Es ist sinnvoll die Farben nicht einfach zufällig zu verteilen. Vielmehr sollten zusammengehörigen Gruppen - wie etwa Script-Viren - ähnliche Farben zugeordnet werden. Da das Programm diese Farben aber in der aktuellen Implementierung nicht selbständig zuordnen kann, muss dies vom Benutzer aus geschehen. Um diesem die Arbeit zu erleichtern, werden alle Farbzuordnungen als Kästchen auf einem Farbkreis dargestellt. Daneben erscheint eine Liste der Malware der entsprechenden Kategorie. Das Farbfeld ist außerdem in Bereiche eingeteilt, so dass der Benutzer die Farben leicht einordnen kann.

[20] Malware außer Viren/Würmer.
[21] y: Quelle=MWC, x: Quelle=NAI-Crawler (siehe [Soller01]).

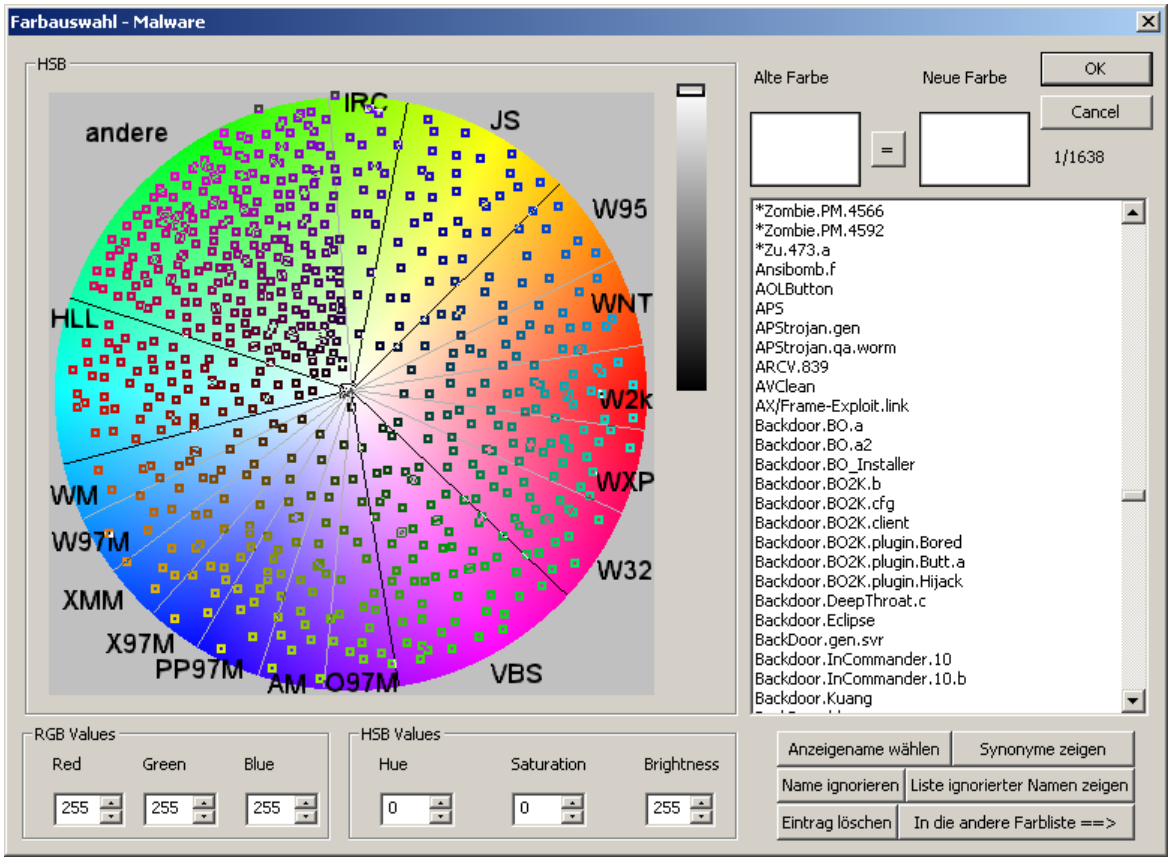


Abbildung 11 - MWC-Visualizer Farbauswahl

Durch Klicken mit der linken Maustaste kann dem markierten Eintrag in der Liste eine Farbe zugeordnet werden. Mit der rechten Maustaste kann man den zu einem Kästchen auf dem Farbkreis gehörenden Eintrag aus der Liste auswählen.

Um eine einheitliche Farbzuordnung zu ermöglichen und um ein wiederholtes Zuordnen der Farben zu vermeiden, werden die Zuordnungen gespeichert. Beim nächsten Auftreten eines Malwarenamens wird dann automatisch die entsprechende Farbe zugeordnet. Es muss also nur noch derjenigen Malware eine Farbe zugeordnet werden, die bisher nicht aufgetreten ist. Diese Malwarenamen sind rechts in der Liste ganz am Anfang zu finden und mit einem Stern gekennzeichnet.

Eine generierte HTML-Datei für einen Monat sieht dann wie folgt aus[22]:

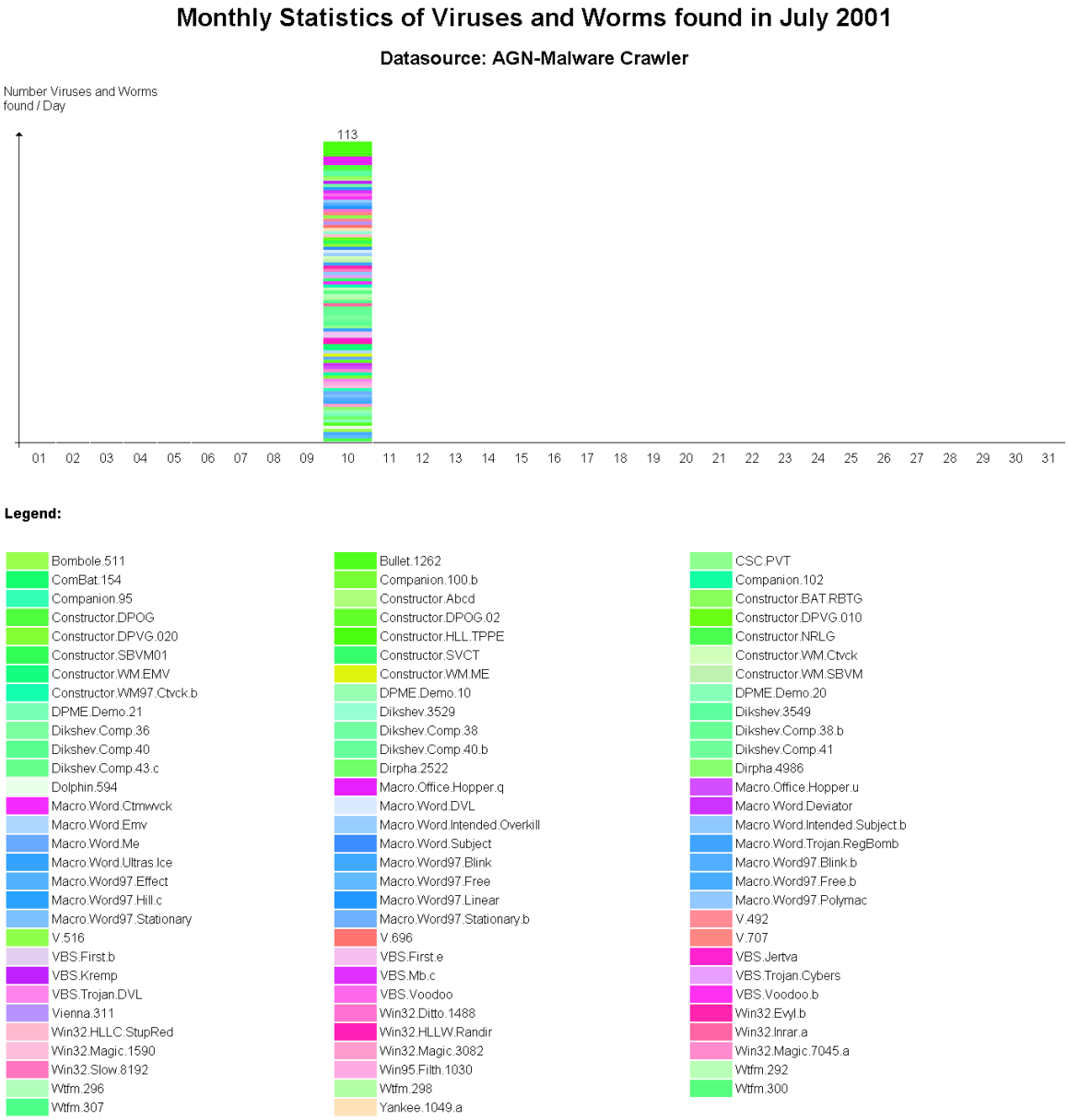


Abbildung 12 - Generierte HTML-Datei (Viren / Würmer)

[22] Der AGN-Malware Crawler lief im Juli nur an einem Tag.

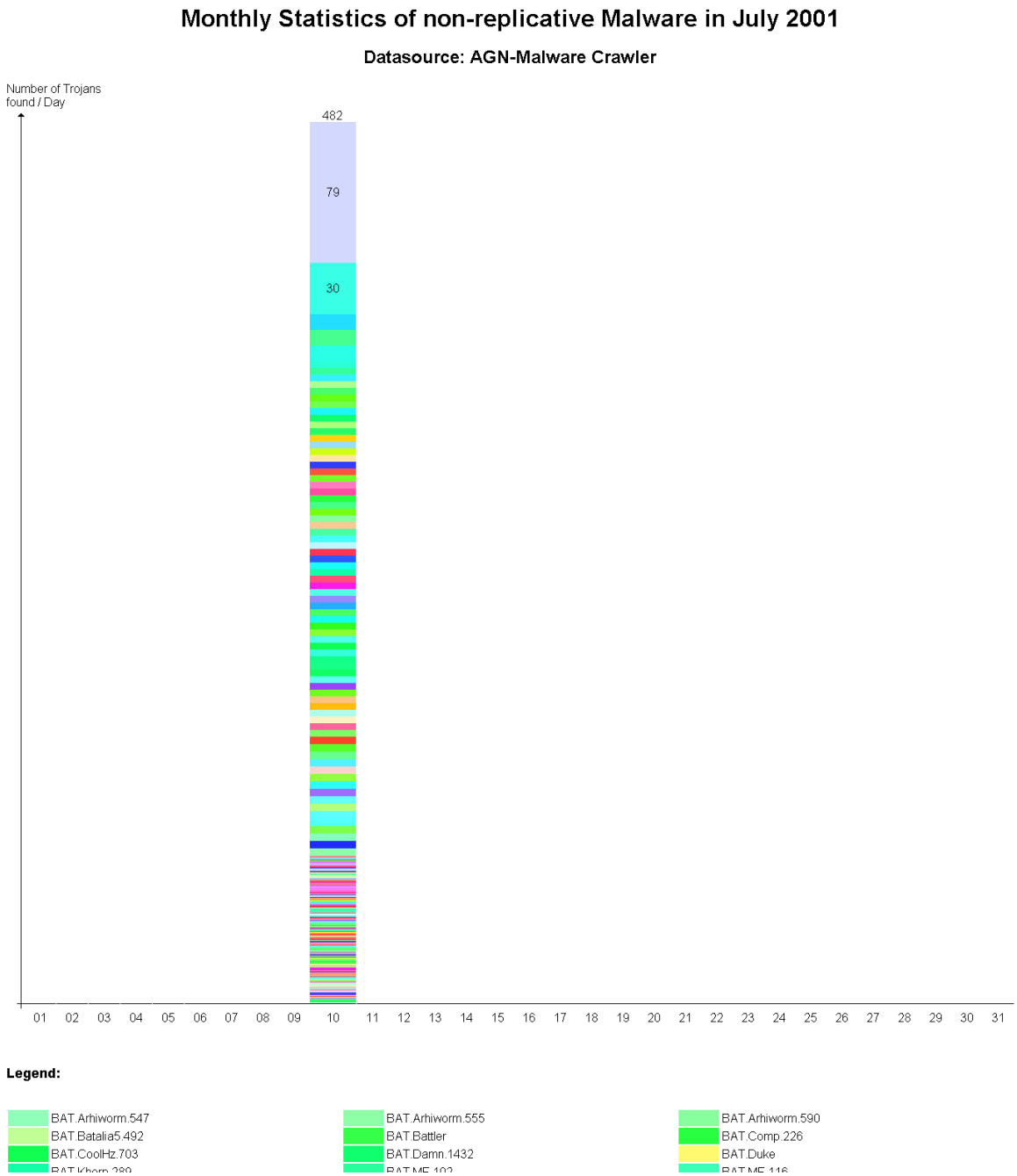


Abbildung 13 - Generierte HTML-Datei (Trojaner, gekürzte Legende)

3.1.4 Ergebnisse

Der Malware Crawler benötigt für den Download und das Scannen der zu überprüfenden Dateien mehrere Tage. Aus diesem Grund ist die Monatsstatistik, die die an den einzelnen Tagen gefundenen Viren visualisiert,

nicht sinnvoll. Diese kann zwar ebenfalls erzeugt werden, um einer eventuellen Optimierung[23] standzuhalten, wird hier aber nicht betrachtet. Außerdem lief der Malware Crawler im Entstehungszeitraum der vorliegenden Diplomarbeit nicht kontinuierlich, so dass schon allein die Auswertung anhand der Jahresstatistik nicht sehr Umfangreich ausfallen kann. Da die umfangreichsten Daten von 2001 sind, werden diese hier dargestellt.

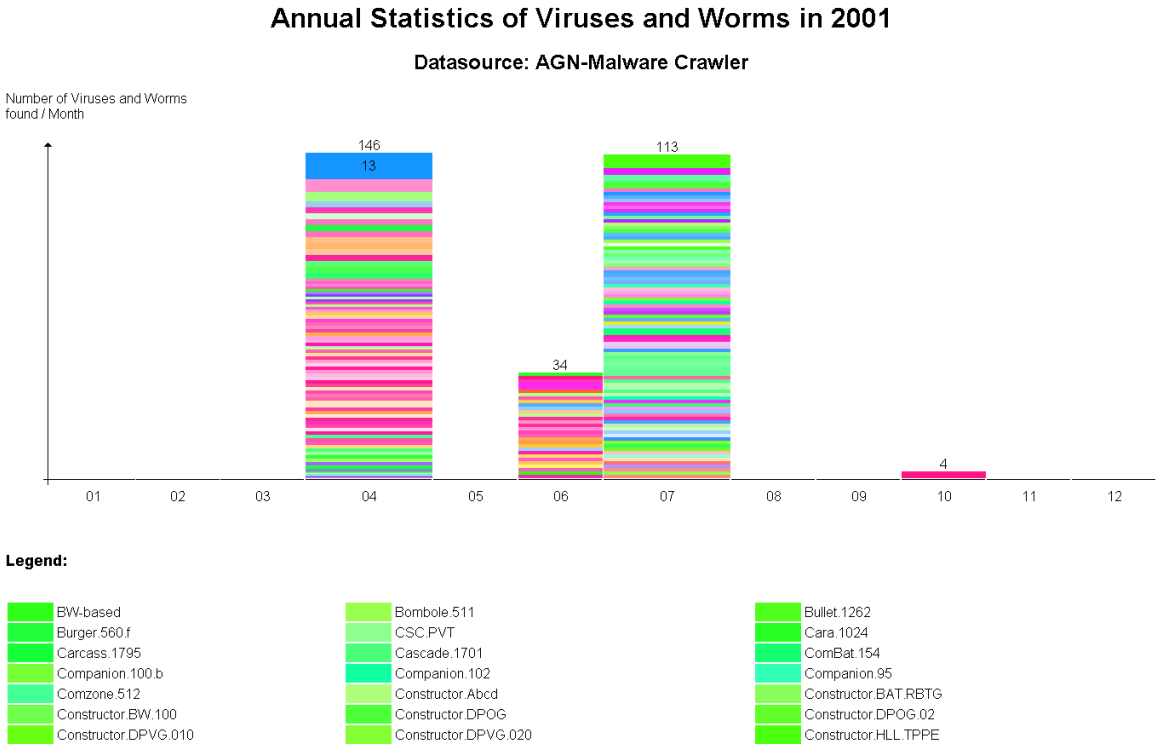


Abbildung 14 - Jahresstatistik 2001 (Viren / Würmer, gekürzte Legende)

[23] Z. B. durch den Einsatz mehrerer Rechner zum Download.

Abbildung 15 - Jahresstatistik 2001 (Trojaner, gekürzte Legende)

Die grafische Darstellung ist abhängig von den Daten, aus denen sie erzeugt wird. Durch eine Veränderung dieser Daten, z. B. durch regelmäßigeres

Scannen des Malware Crawlers, könnten bessere Ergebnisse erzielt werden, die dann auch auswertbar sind.

3.2 Darstellung als Graph

Für die Analyse der Gruppendependenzen ist eine weitere Form der Visualisierung nötig. Hierzu werden die Websites als gerichteter Graph dargestellt, wodurch zusammenhängende Gruppen hervorgehoben werden sollen. Die Knoten des Graphen repräsentieren die einzelnen Websites (dargestellt als Punkte) und die Kanten repräsentieren die Links zwischen den Websites (dargestellt als Pfeile). Die Farbe der Punkte stellt den Heuristikwert der jeweiligen Website dar. In den folgenden Abschnitten werden die Probleme, die hierbei auftreten, beschrieben sowie die implementierten Lösungen dargestellt. Weiterhin werden Methoden zur Speicherung der aufbereiteten Geometriedaten erläutert und anhand von Speicherplatzabschätzungen die beste ausgewählt. Am Ende werden die generierten Ergebnisse aufgezeigt und analysiert.

3.2.1 Probleme

Das erste Problem bei der Darstellung der URLs auf dem Bildschirm ist die Position, an der diese angezeigt werden. Diese könnte man z.B. aus dem URL-String generieren, so dass man auch die Domain-Endung (.de, .com usw.) in die Grafik einbezieht. Dies würde aber nur eine grobe Rasterung ergeben und hätte nicht den gewünschten Effekt der Visualisierung.

Die URLs sollen nicht nur einfach möglichst realistisch aufgrund des geografischen Standortes[24] oder des Inhaltes dargestellt werden, wie es z.B. im Atlas of Cyberspace [Atlas], Online Cyberspace-Atlas [Cyber] und bei Map.net™ [MAP] der Fall ist. Vielmehr sollen die Websites nach einem

[24] Z. B. durch Übersetzung der IP-Adresse in Längen und Breitengrad (siehe [IP2LatLong]).

bestimmten Algorithmus geordnet werden, der die Beziehungen der URLs untereinander berücksichtigt. So sollen URLs, die stärker miteinander verlinkt sind, näher beieinander stehen als solche, die nicht oder nur kaum miteinander verlinkt sind. Es soll also von der geografischen Position der Seite abstrahiert werden. Die Positionierung der Knoten wird in der Visualisierungspipieline im Abschnitt „Mapping“ genauer beschrieben.

Des Weiteren sollten die Seiten einer Website nicht einzeln dargestellt werden, sondern als Ganzes. Das wirft das Problem auf, zu entscheiden, ob zwei Seiten zu derselben Website gehören oder nicht. Diese Schwierigkeit ist relativ leicht zu lösen und wird in der Visualisierungspipieline im Abschnitt „Filtering“ erläutert.

3.2.2 Rohdaten

Die Rohdaten bestehen aus den vom Malware Crawler gelieferten Daten. Es werden alle Verweise übergeben, bestehend aus URL und Ursprungs-URL. Hinzu kommen außerdem noch einige Eigenschaften der URL z.B. der Heuristikwert, das Datum und die Checksumme.

Die vom Malware Crawler erzeugte Datei besteht aus mehreren Blöcken. Jeder Block beinhaltet die Informationen für einen Verweis und hat eine Größe von 808 Bytes. Die Blöcke haben folgende Struktur:

Größe in Bytes	Inhalt
254	URL
254	Verweisende-URL
4	Heuristikwert (0-9999)
10	Datum (TT-MM-JJJJ)
10	Zeit
10	Checksumme
10	Suchtiefe
254	Reserviert für Erweiterungen
2	CR und RET, d.h. chr(13) und chr(10)

3.2.3 Zusammenfassen von HTML-Seiten zu Websites (Filtering)

Es reicht nicht aus, jede Domain als eigenständige Website zu betrachten, da es einige Ausnahmen gibt, bei denen in den Unterverzeichnissen unabhängige Websites existieren. So findet man z.B. bei Freehostern wie Geocities (www.geocities.com) verschiedene Benutzer, dessen Websites die gleiche Domain besitzen (z.B. www.geocities.com/user1/index.html und www.geocities.com/user2/index.html).

Es ist also nötig, eine Liste mit Ausnahmen anzulegen, in der zum Einen die Domain enthalten ist und außerdem die Verzeichnistiefe, ab der es sich um verschiedene Websites handelt.

Es folgt eine Liste mit diesen Domains[25]. Diese Liste ist natürlich jederzeit erweiter- oder änderbar.

Domain	Verzeichnistiefe, ab der es sich um verschiedene Websites handelt
www.geocities.com	1
members.fortunecity.de	1
members.fortunecity.es	1
members.fortunecity.com	1
members.telecom.at	1
members.nextra.at	1
www.angelcities.com/members	2
www.webspawner.com/users	2
users.visi.net	1
www.angelfire.com	2
members.m4d.com	1
member.aol.com	1
members.aol.com	1
mitglied.lycos.de	1
www.members.tripod.de	1

[25] Quelle: [webspace],[google] und MWC-Daten

members.xoom.com	1
www.tripod.com	1
www.crosswinds.net	1
members.nbci.com	1
welcome.to[26]	1
surf.to	1
kickme.to	1
thx.to	1
jump.to	1
fly.to	1
go.to	1

Die Darstellung von zusammengefassten Webpages gestaltet sich wie folgt: Zum Einen werden die Punkte in unterschiedlichen Größen dargestellt, abhängig von der Anzahl der zusammengefassten Webpages. Außerdem ist nun die Farbe abhängig von mehreren Heuristikwerten. Hierbei gibt es drei Modi: Durchschnitt, Minimum und Maximum. Die naheliegende Durchschnittsbildung wird durch das Maximum bzw. Minimum ergänzt, um einzelne hohe oder niedrige Heuristikwerte nicht zu übersehen.

3.2.4 Interne Datenstruktur (Aufbereitete Daten und Geometriedaten)

Um einen Graphen im Programm verarbeiten zu können, muss zuerst eine geeignete Datenstruktur gefunden werden. Hierbei ist eine Implementation mit möglichst geringer Laufzeit zu wählen, da sehr große Datenmengen verarbeitet werden müssen. Die Laufzeit eines Graphenalgorithmus hängt wesentlich von zwei Parametern ab: Von der Anzahl der Knoten und der Anzahl der Kanten. Das macht deren vergleichende Analyse komplizierter. Benötigt ein Algorithmus beispielsweise $|V|^2$ und ein anderer Algorithmus $(|E|+|V|) \cdot Log(|E|)$ Schritte, so ist der erste Algorithmus für dichte und der zweite für lichte Graphen besser geeignet.

[26] Bei dieser und den folgenden Domains handelt es sich um Redirektoren, also um Weiterleitungen.

Bei der Darstellung von WWW-Strukturen handelt es sich um sehr viele Knoten, relativ dazu aber nur um wenige Kanten, die von einem Knoten ausgehen. Es ist also ein Algorithmus zu wählen, der für lichte Graphen optimal ist.

Eine einfache Methode zur Verwaltung von Graphen ist die Speicherung in einer sogenannten Adjazenz[27]-Matrix. Hierbei handelt es sich um ein zweidimensionales binäres Feld der Größe $|V|^2$. Ein Eintrag a_{uv} wird auf *true* gesetzt, wenn es eine Kante zwischen u und v gibt[28], andernfalls auf *false*. Da es sich aber hierbei in Bezug auf Laufzeit und Speicherplatz um eine Methode handelt, die besser für dichte Graphen geeignet ist, ist die Speicherung der Daten in Form von Adjazenz-Listen vorzuziehen. Bei Adjazenz-Listen handelt es sich um Listen, die für jeden Knoten alle mit ihm verbundenen Knoten enthalten. Bei der Berechnung der Knotenpositionen wird zwar von gerichteten Graphen auf ungerichtete abstrahiert, bei der Verwaltung der Daten muss aber dennoch die Richtung mit gespeichert werden, da diese auf dem Bildschirm dargestellt wird. Es werden also bei jedem Knoten nicht einfach die mit ihm verbundenen Knoten in die Listen aufgenommen, sondern es wird zwischen eingehenden und ausgehenden Kanten unterschieden. Theoretisch würde es zwar reichen, nur die Knoten ausgehender Links zu speichern, aus Performance-Gründen ist es aber vorteilhaft beide Arten zu speichern. Das hat den Vorteil, dass es leichter ist, alle Nachbarknoten zu finden.

Der interne Aufbau der Daten sieht folgendermaßen aus:

Eigenschaften der Klasse CKnoten:

Variablenname	Typ	Beschreibung
ptPosition	CPoint	Position auf dem Bildschirm
lngHeuristik	CArray <long,long> *	Heuristikwerte
nach	CArray <long,long> *	Ausgehende Links (IDs)
von	CArray <long,long> *	Eingehende Links (IDs)
alias	CArray <CString,CString> *	Zusammengefasste Seiten
lngMarkierung	Long	Markierung (zum Filtern)
blnFest	Bool	Fixierter Knoten
strZusammen	CString *	Einheitlicher Name der Website

* = Pointer

[27] Adjazent = (lat.) Anlieger, Anwohner, Nachbar.
[28] Bzw. von u nach v bei gerichteten Graphen.

Eigenschaften der Klasse CGraph:

Variablenname	Typ	Beschreibung
Knoten	CKnoten*	Enthaltene Knoten
lngKnotenAnzahl	Long	Anzahl der enthaltenen Knoten
bytStandTag	Byte	Stand der Daten (Tag)
bytStandMonat	Byte	Stand der Daten (Monat)
lngStandJahr	Long	Stand der Daten (Jahr)
blnElektroquick	Bool	Verwendung des schnellen Elektrofeld-Algorithmus[29]
blnElektrofeld	Bool	Verwendung des Elektrofeld-Algorithmus
blnFederkraft	Bool	Verwendung des Federkraft-Algorithmus
blnMagnetfeld	Bool	Verwendung des Magnetfeld-Algorithmus
lngFederlaenge	Long	Natürliche Federlänge
dblElektrofeld	Double	Stärke des Elektrofeldes
dblSteifheit	Double	„Steifheit“ der Federn
dblMagnetstaerke	Double	Stärke des Magnetfeldes
blnDurchschnitt	Bool	Durchschnittsberechnung verwenden
blnEinheitsFeder	Bool	Einheitliche Federlänge
dcount	Long	Interner Zähler

* = Pointer

Beim Filtering wird die Eigenschaft „ptPosition“ nicht benutzt. Diese wird erst beim Mapping mit Positionsdaten gefüllt. Diese Eigenschaft ist also erst in den Geometriedaten enthalten.

Da es sich bei der hier vorgestellten Implementation um eine interaktive Anwendung handelt, kann das Mapping der Visualisierungspipeline mehrmals durchlaufen werden. Da es sich einmal um eine Daten-zu-Geometriedaten Abbildung und einmal um eine Geometriedaten-zu-Geometriedaten Abbildung handelt, müssen notwendigerweise zwei Mappingabschnitte vorhanden sein.

[29] Dieser und die folgenden drei Algorithmen werden später erläutert.

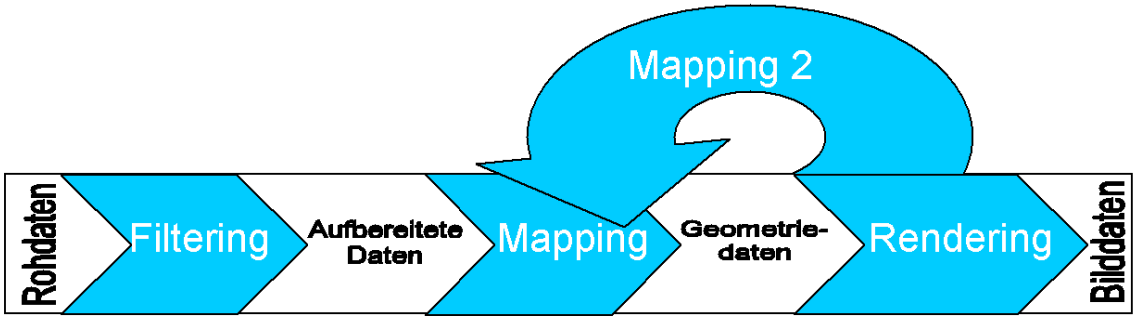


Abbildung 16 - Mehrmaliges Mapping

Das erste Mapping ist trivial: Den Knoten werden zur Initialisierung zufällige Positionen zugeteilt. Das eigentliche Sortieren kann nun beim zweiten Mapping erfolgen. Im nächsten Kapitel wird dies genauer erläutert.

Es ist möglich, die Geometriedaten zu speichern. Dadurch entfällt das oft langwierige Filtering und auch die bisherigen Durchläufe des zweiten Mappings. Die Datei ist wie folgt aufgebaut:

Header
Knoten 1
Knoten 2
…
Knoten n-1
Knoten n

Abbildung 17 - Dateistruktur

Der Aufbau des Headers:

Größe in Byte (Typ)	Inhalt
4 + 3 (Stringlänge + String)	Kennung (“VIS“)
2	Versionsnummer
6	Stand der Daten (1 Byte Tag, 1 Byte Monat, 4 Bytes Jahr)

1 (Boolean)	Einbeziehung des Elektrofeldes
1 (Boolean)	Schnelle Elektrofeldberechnung verwenden
1 (Boolean)	Einbeziehung des Magnetfeldes
1 (Boolean)	Einbeziehung der Federkraft
4 (Long)	Natürliche Länge der Federn
8 (Double)	Stärke des Elektrofeldes
8 (Double)	Stärke des Magnetfeldes
8 (Double)	„Steifheit“ der Federn
1 (Boolean)	Durchschnittsberechnung verwenden
1 (Boolean)	Alle Federn gleich lang an/aus
4 (Long)	Anzahl der Knoten
4 (Long)	Anzahl der (bisherigen) Iterationen
4 (Long)	Aktueller Zoomfaktor
4 (Long)	Aktuelle Bildschirmposition (X-Achse)
4 (Long)	Aktuelle Bildschirmposition (Y-Achse)

Der Aufbau eines Blocks:

Größe in Byte (Typ)	Inhalt
4 (Long)	x-Koordinate
4 (Long)	y-Koordinate
1 (Boolean)	Festgepinnter Knoten
4 (Long)	Anzahl der Heuristikwerte (=n)
4 (Long)	1. Heuristikwert
...	...
4 (Long)	n. Heuristikwert
4 (Long)	Anzahl der Seiten der Website (=n)
4 + Stringlänge	1. Seite der Website
...	...
4 + Stringlänge	n. Seite der Website
4 (Long)	Anzahl der ausgehenden Links (=m)
4 (Long)	Index des 1. Links
...	...
4 (Long)	Index des m. Links

3.2.5 Positionieren der Knoten (Mapping)

Ziel ist es, die Knoten des Graphen so anzuordnen, dass der Graph möglichst „übersichtlich“ wird. Diese Übersichtlichkeit lässt sich erreichen, indem durch Kanten verbundene Knoten nah zueinander und nicht verbundene Knoten weiter entfernt positioniert werden.

Eine Möglichkeit, diese Umschreibung in Formeln fassen zu können sind die Force-Directed Methoden, die z.B. in Graph Drawing – Algorithms for the visualization of graphs [GraphDraw] beschrieben werden. Hierbei werden physikalische Analogien benutzt, um die Knoten zu positionieren. Die Knoten des Graphen werden als ein System aus Körpern angesehen, zwischen denen physikalische Kräfte wirken. Ein Force-Directed Algorithmus sucht also einen Zustand, in dem die Summe der wirkenden Kräfte auf jeden Knoten gleich Null ist.

Bild a der folgenden Abbildung zeigt z.B. einen Graphen, in dem die Kanten durch Federn ersetzt wurden. Bild b zeigt einen Zustand, in dem die Summe der wirkenden Kräfte auf jeden Knoten gleich null ist. Dieser Graph kann als der Graph, den Bild c in der herkömmlichen Darstellung zeigt, interpretiert werden.

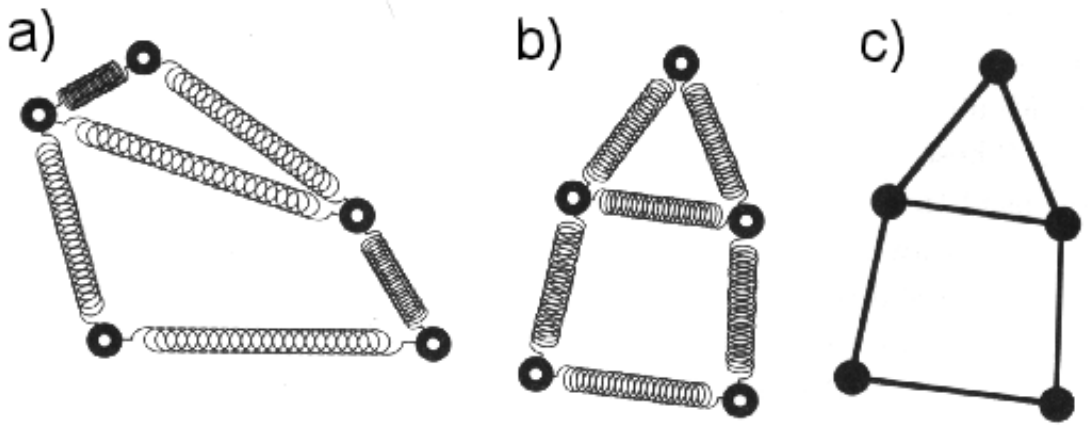


Abbildung 18 - Federkraft-Algorithmus (aus [GraphDraw])

Kräfte, die in einem Force-Directed Algorithmus simuliert werden sind z.B.:

- Federkraft
- Elektrisches Feld
- Magnetfeld
- Allgemeine Energiefunktionen

Hierbei ist es nicht nur theoretisch möglich, mehrere Kräfte zu kombinieren, sondern auch sinnvoll, um bessere Ergebnisse zu erzielen. So würde ein Federkraft-Algorithmus aus dem in der nächsten Abbildung in Bild 1 gezeigten Graphen das in Bild 2 gezeigte Ergebnis liefern. Mit einer Kombination von Federkraft und elektrischem Feld bekäme man hingegen das in Bild 3 gezeigte Resultat.

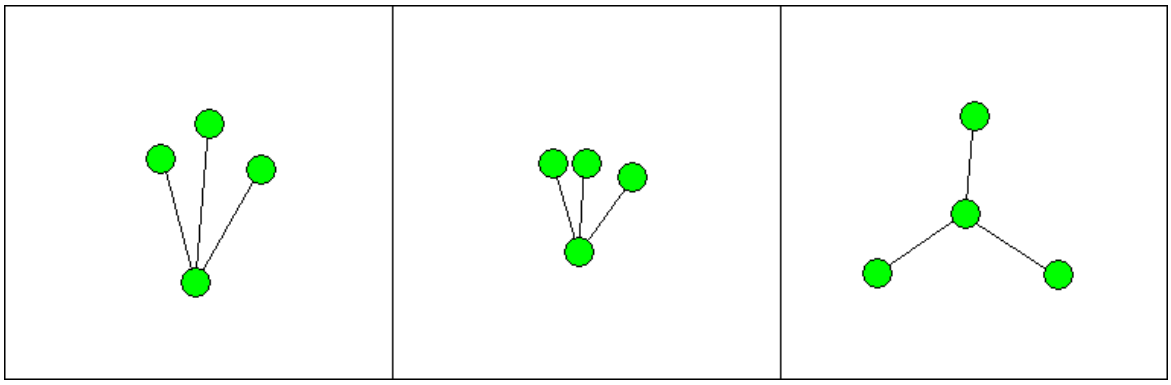


Abbildung 19 - Kombination von Federkraft und elektrischem Feld

Das elektrische Feld bewirkt also, dass die Knoten sich nicht zu nahe kommen und damit eine gleichmäßige Verteilung von Knoten mit nur einer Verbindungskante zu einem zentralen Mittelpunkt.

Es gibt verschiedene Möglichkeiten einen Force-Directed Algorithmus zu realisieren. Eine einfache, intuitive Technik arbeitet folgendermaßen: Zunächst werden die Knoten an zufälligen Positionen platziert. In jedem Iterationsschritt wird von jedem Knoten v die Kraft $f(v)$ berechnet dann in die entsprechende Richtung bewegt.
Diese einfache Methode ist sicherlich nicht die schnellste Art, ein Gleichgewicht zu finden. Es liefert jedoch eine fließende Animation der Transformation von zufälligen Positionen in einen gleichmäßigen Endzustand.
Ein Beispiel einer solchen Transformation ist in der nächsten Abbildung zu sehen.

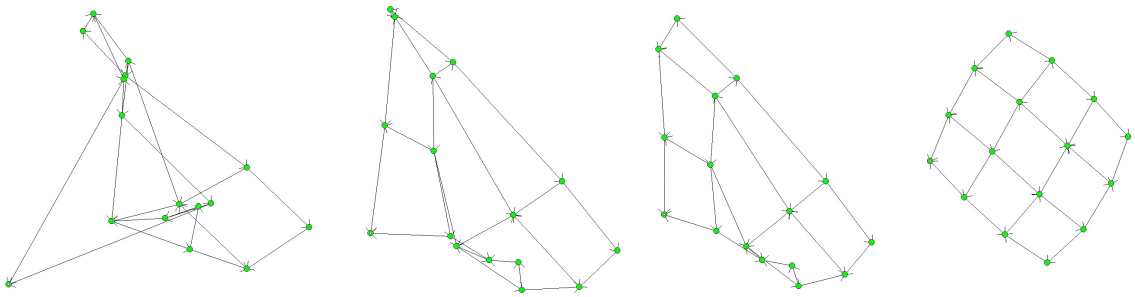


Abbildung 20 - Animationsschritte eines Force-Directed Algorithmus

Der Force-Directed Algorithmus im Überblick:

1. Knoten zufällig verteilen
2. Die jeweiligen physikalischen Kräfte berechnen
3. Knoten in Richtung der errechneten Kraft bewegen
4. Falls keine Signifikante Änderung stattgefunden hat → Ende
5. Weiter bei 2.

3.2.5.1 Herleitung der Formel zur Berechnung der Kraft

Die Simulation der Kombination aus Federkraft und elektrischem Feld reicht für die Visualsierung vollkommen aus. Im Folgenden wird die für die Berechnung der Punktpositionen notwendige Formel hergeleitet. Die Simulation eines Magnetfeldes wird besonders für Graphen, die eine zeitliche Abfolge repräsentieren, verwendet. Der Vollständigkeit halber ist der Magnetfeldalgorithmus zwar implementiert worden, auf die Herleitung der Formel wird hier aber verzichtet[30].

Zur Berechnung wird der eigentlich gerichtete Graph als ungerichtet aufgefasst, da die Einbeziehung der Richtungen der Kanten die Rechnung nur unnötig

[30] Diese ist in [Draw] beschrieben und wird weiter unten angegeben.

komplizieren würde. Außerdem würde es die resultierende Visualisierung nicht beeinflussen.

Ausgangsformel ist die Grundformel für die Kraft, die auf v wirkt:

$$F(v) = \sum_{(u,v)\in E} f_{uv} + \sum_{(u,v)\in V\times V, u\neq v} g_{uv}$$

Hierbei ist f_{uv} die Kraft, die durch die Feder zwischen u und v auf v wirkt und g_{uv} die elektrische Abstoßung die von u auf v wirkt, wobei die elektrische Kraft zwischen allen Knoten wirkt, die Federkraft hingegen nur zwischen Knoten, die miteinander verbunden sind, d.h. zwischen Knoten für die gilt: $(u, v) \in E$.

Die Kraft f_{uv} folgt dem Hooke'schem Gesetz, d.h. f_{uv} ist proportional zur Differenz zwischen dem Abstand von u zu v und der natürlichen Länge l der Feder. Diese Kraft wird zusätzlich noch mit k_1 multipliziert, um eine „Steifheit" der Federn zu simulieren. Die elektrische Kraft g_{uv} ist invers quadratisch zum Abstand von u zu v. Durch zusätzliches Multiplizieren mit k_2 kann die Stärke des elektrischen Feldes variiert werden.

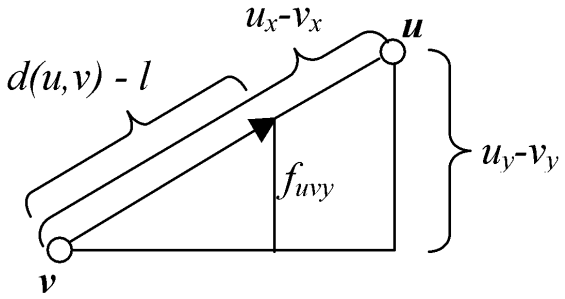


Abbildung 21 - Berechnung der y-Komponente f_{uvy} der Kraft f_{uv}

Nach dem Strahlensatz ergibt sich für die y-Komponente f_{uvy} der Kraft f_{uv}:

$$\frac{f_{uvy}}{u_y - v_y} = \frac{k_1(d(u,v)-l)}{d(u,v)} \Rightarrow f_{uvy} = k_1(d(u,v)-l)\frac{u_y - v_y}{d(u,v)}$$

Analog für die x-Komponente:

$$f_{uvx} = k_1(d(u,v) - l)\frac{u_x - v_x}{d(u,v)}$$

Bei der Berechnung der x- und y-Komponenten g_{uvx} und g_{uvy} der Kraft g_{uv} kommt außerdem ein negatives Vorzeichen hinzu, da sich hier die Knoten nicht anziehen, sondern abstoßen:

$$g_{uvx} = \frac{-k_2}{(d(u,v))^2}\frac{u_x - v_x}{d(u,v)} = \frac{k_2(v_x - u_x)}{(d(u,v))^3}$$

Analog für die y-Komponente:

$$g_{uvy} = \frac{k_2(v_y - u_y)}{(d(u,v))^3}$$

Dadurch ergibt sich die gesamte Formel für die wirkende Kraft $F(v)$:

x-Koordinate:

$$\sum_{(u,v)\in E} k_1(d(u,v) - l)\frac{u_x - v_x}{d(u,v)} + \sum_{(u,v)\in V\times V, u\neq v} \frac{k_2(v_x - u_x)}{(d(u,v))^3}$$

y-Koordinate:

$$\sum_{(u,v)\in E} k_1(d(u,v) - l)\frac{u_y - v_y}{d(u,v)} + \sum_{(u,v)\in V\times V, u\neq v} \frac{k_2(v_y - u_y)}{(d(u,v))^3}$$

Erläuterung der Variablen:

u: Knoten des Graphen

v: Neu zu berechnender Knoten

l: Natürliche Länge der Federn

k_1: *„Steifheit“* der Federn

k_2: Stärke des elektrischen Feldes

Diese Formel für die Federkraft findet sich soweit auch in der Literatur wieder (siehe [GraphDraw] und [Draw]). Sie hat leider einen kleinen Schwachpunkt. Sie funktioniert nur bei sehr lichten Graphen oder Graphen mit wenigen Knoten. Bei anderen Graphen tritt folgendes Problem auf:
Hat ein Knoten mehr als zwei Nachbarn, so ist die wirkende Federkraft so groß, dass der Knoten über das Ziel „hinaus katapultiert“ wird. Dies wird normalerweise durch k_1 und k_2 kompensiert.

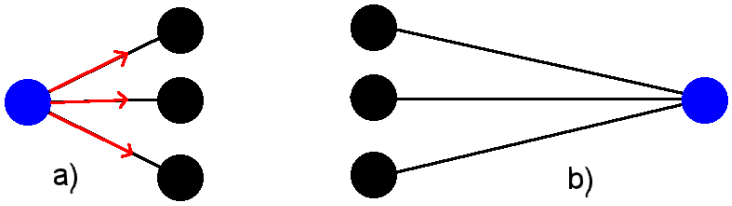


Abbildung 22 - Knotenkatapult

Bei einer großen Knotenanzahl reicht dies aber nicht mehr aus, bzw. müssten diese Werte stark angepasst werden. Es ist aber nicht möglich, die Knotenanzahl im Voraus zu wissen. Außerdem sollte der Algorithmus allgemein sein und auch bei Graphen mit wenigen Knoten funktionieren.
Aus diesem Grunde wird bei der Berechnung der Federkraft die berechnete Kraft für jeden Punkt durch die Anzahl der Nachbarn geteilt. Dies ähnelt einer Mittelpunktsberechnung und bewirkt, dass sich die Kräfte nicht „aufschaukeln“ können.
Bei der Berechnung der elektrischen Kraft ist der Effekt nicht ganz so gravierend. Zwar wirkt sie zwischen allen Knoten, was für eine große Gesamtknotenanzahl sehr große Werte ergeben kann, jedoch nimmt die elektrische Kraft auch stark mit zunehmender Entfernung ab.

Eine weitere Force-Directed Methode, speziell für gerichtete Graphen, besteht in der Simulation eines Magnetfeldes. Dabei wirken die Kanten des Graphen wie magnetische Eisenstangen. Dies bewirkt eine Ausrichtung der Kanten in

eine vorgegebene Richtung. Eine Formel zur Berechnung der Magnetfeldkraft f_{mag} findet sich in Drawing Graphs [Draw]:

$$f_{mag}(u,v) = b \bullet d(u,v)^{c1} \bullet \theta^{c2} \bullet e^{\perp}$$

Erläuterung der Variablen:

u: Knoten des Graphen

v: Neu zu berechnender Knoten

b: Stärke des Magnetfeldes

c_1: Parameter zur Einbeziehung des Abstandes

c_2: Parameter zur Einbeziehung des Winkels

θ: Winkel zwischen dem Magnetfeld und der Kante e

$e^{\perp}$: Einheitsvektor, rechtwinklig zur Kante e zwischen u und v, in die θ verringernde Richtung

Da dies eher bei Graphen von Vorteil ist, die z.B. einen Datenfluss repräsentieren, wurde hier nur eine einfache Variante mit homogenem (senkrecht) Magnetfeld und mit $c_1 = 0$, $c_2 = 1$ und $\theta = |u_x - v_y| \,/\, d(u,v)$ implementiert.

3.2.5.2 Optimierungen

Der oben beschriebene Federkraft-Algorithmus lässt sich noch weiter verbessern, indem man die natürliche Länge l der Federn abhängig von den mit ihr verbundenen Knoten macht. Man setzt z.B. $l = l_{uv}$ mit gerichteter Kante von u nach v und

$$l_{uv} = Grundlänge \bullet max(d_a(v), 1).$$

Weiterhin sind einige Optimierungen bei der Berechnung der elektrischen Kraft nötig. Da diese zwischen allen Knoten wirkt, ist der Rechenaufwand quadratisch: Es muss für jeden Knoten der Abstand zu jedem anderen ermittelt

und daraus die elektrische Kraft berechnet werden. Dies stellt bei kleineren Graphen kein Problem dar, der größte hier untersuchte Graph hat aber ca. 86.000 Knoten. Dies bedeutet 7,396 x 10^9 Berechnungen pro Iterationsschritt.
Eine erhebliche Beschleunigung des Algorithmus bringt die folgende Einschränkung in der Berechnung: Es wird jeweils nur die elektrische Kraft zwischen zwei Knoten berechnet, die über einen Weg der Länge 2 verbunden sind. So werden weiterhin sternförmig angeordnete Knoten gleichmäßig um den zentralen Mittelpunkt verteilt und der Rechenaufwand wird linear.

3.2.6 Rendering

Um genauer analysieren zu können, bedarf es der Möglichkeit, bestimmte Bereiche genauer zu betrachten. Hierzu gibt es zwei Funktionen:

- Zoomfunktion
- Ausschnittsverschiebung

Mit der Ausschnittsverschiebung wird der zu untersuchende Bereich ausgewählt. Der angezeigte Bereich auf dem Bildschirm ist hierbei nur ein Teil der gesamten Ebene (virtueller Schirm). Mit der Zoomfunktion kann der Bereich dann vergrößert werden. Zur besseren Übersicht und Orientierung gibt es außerdem noch die Möglichkeit, Gitternetzlinien einzublenden.
Des Weiteren besteht die Möglichkeit, bestimmte Knoten auszublenden. Dies erlaubt beispielsweise die Hervorhebung einzelner Komponenten (bei unzusammenhängenden Graphen).

3.2.7 Bilddaten

Die Bilddaten werden als HTML-Datei erzeugt. Außerdem werden Grafiken erzeugt, die in das HTML-Dokument eingebunden werden. Dies ist zum Einen der Teil des Graphen, der auf dem Bildschirm sichtbar ist. Zum Anderen wird eine Grafik der Farbskala erzeugt. Die Skala zeigt die Zuordnung der

Knotenfarben zu den Heuristikwerten. Beide Grafiken werden als JPG-Datei generiert. Weiterhin ist im HTML-Dokument eine Legende und allgemeine Daten des Graphen, wie z.B. Knotenanzahl, Linkanzahl oder Durchschnittsgrad. Es besteht auch die Möglichkeit, nur die Bildschirmausgabe als JPG-Datei zu speichern, bzw. die verschiedenen Einzelbilder einer Sortieriteration automatisch zu speichern, um später daraus eine Animation zu generieren[31].

3.2.8 Ergebnis

Kleinere Graphen werden durch den Algorithmus sehr schnell umsortiert. Bei den vom Malware Crawler gelieferten umfangreichen Daten dauert dies schon wesentlich länger. Dabei kann man ab mehreren hundert Iterationen typische Strukturen erkennen. Die folgenden Strukturen wurden in der Darstellung der vom Malware Crawler gelieferten Daten vom Juli 2001 nach 200 Iterationsschritten gefunden.
Trotz Umsortierung der Knoten sind bei den aktuellen Daten mit über 86.000 Knoten Strukturen nur schwer zu erkennen. Daher wurden zunächst ältere Daten mit ca. 3000 Knoten untersucht. Für die dort gefundenen Strukturen wurden dann Funktionen implementiert, die diese Strukturen auch in größeren Graphen sichtbar machen und extrahieren können.
Es kristallisierte sich eine grundlegende Klasse von Strukturen heraus, die in allen Datensätzen vorkamen:

Definition:
Ein ***Bündel mit n Basisknoten*** besteht aus einer Gruppe von Knoten (***Bündelknoten***), die genau n eingehende und keine ausgehenden Kanten besitzen, wobei die eingehenden Kanten jeweils von denselben n Knoten (***Basisknoten***) ausgehen.

Es wurden Bündel mit $n < 4$ gefunden:

[31] Dies kann zum Beispiel mit dem Programm „GIF Movie Gear“ [MovieGear] erfolgen.

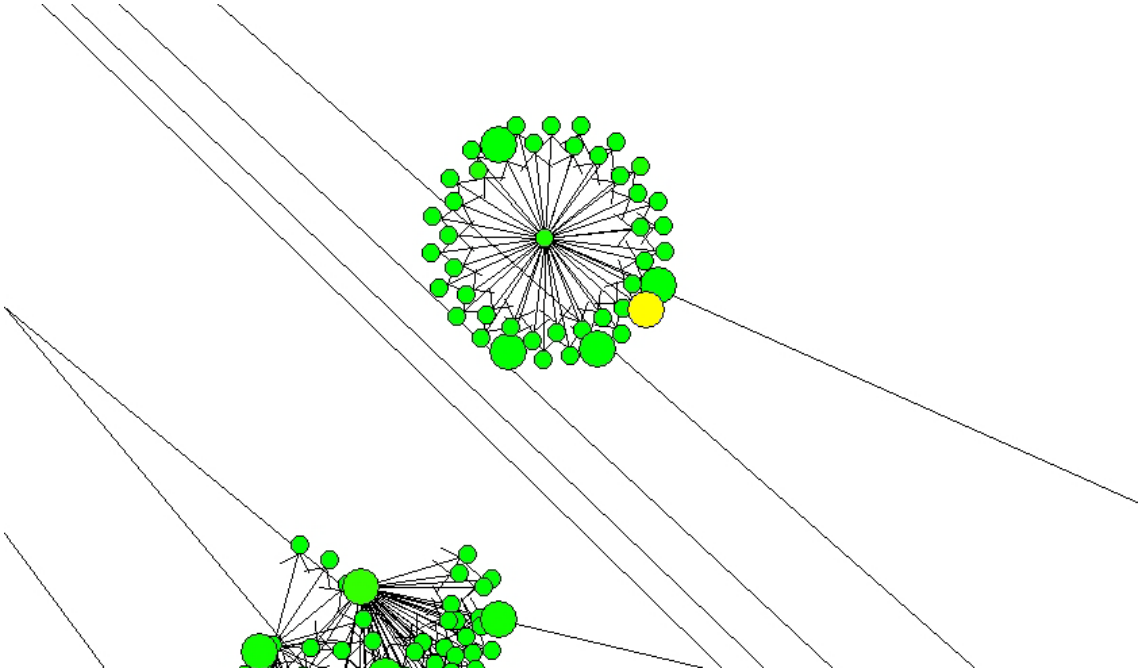


Abbildung 23 - Bündel mit einem Basisknoten

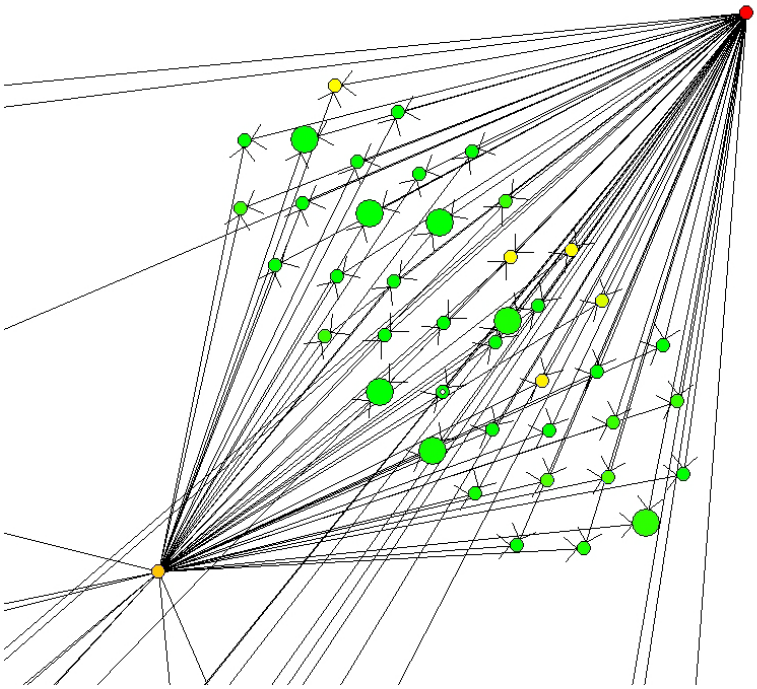


Abbildung 24 - Bündel mit 2 Basisknoten

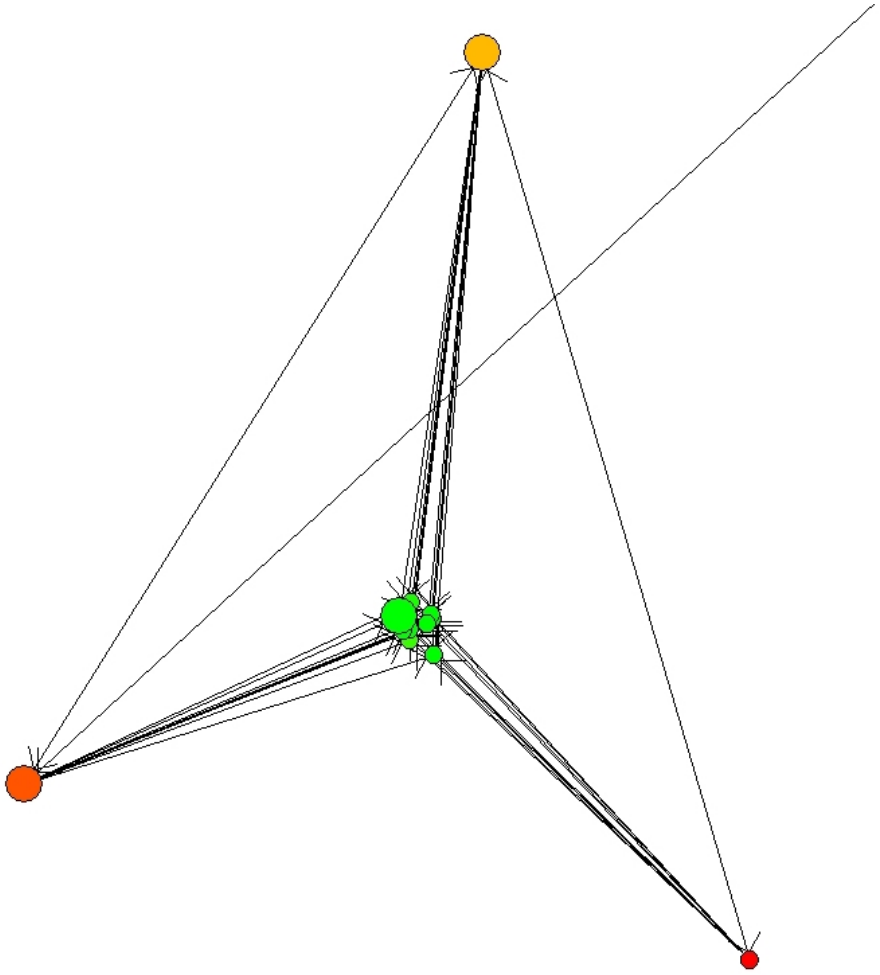


Abbildung 25 - Bündel mit 3 Basisknoten

Weitere Abbildungen der gefundenen Strukturen befinden sich im Anhang der vorliegenden Arbeit sowie auf der beiliegenden Daten-CD.

Bei dem Aufspüren von gelöschten Websites, die an anderer Stelle wieder auftreten, sind die Basisknoten von großer Bedeutung. Da die Basisknoten bei einem Umzug eines Bündelknotens meist schnell aktualisiert werden, müssen nur die Links der Basisknoten bei Veränderung der enthaltenen Links verfolgt werden. Dabei ist es um so effektiver, je mehr Basisknoten vorhanden sind. Selbst wenn dann bei einem Basisknoten die Aktualisierung ausbleibt oder sich verzögert, kann die gesuchte Seite aufgrund eines anderen Basisknoten aufgespürt werden.

Die Basisknoten enthalten meistens nur eine Linksammlung mit Verweisen zu Websites, die maliziöse Software zum Download anbieten, stellen selbst aber keine Malware bereit. Die Bündelknoten dagegen stellen nur Downloads bereit, verlinken aber nicht auf andere Websites. Da diese außerdem möglichst unentdeckt bleiben wollen, ist in vielen Fällen auch der Heuristikwert sehr klein. Der Heuristikwert der Basisknoten ist dagegen eher höher, so dass eventuell auch bisher unentdeckte Websites durch manuelles Überprüfen der Links auf Basisknoten gefunden werden können.
In jedem Fall bringen die gefundenen Strukturen einen klaren Vorteil beim Finden weiterer Websites mit maliziösen Inhalten. Es wäre allerdings sinnvoll, davon abzusehen, eine Löschung der weniger gefährlichen Basisknoten zu veranlassen, um den „Pfad“ zu den gefährlichen Websites nicht zu verlieren.

3.3 Relation zum Heuristikwert des MWC

Die Heuristik ist eines der Hauptbestandteile des Malware Crawlers. Es hat sich gezeigt, dass die Heuristik teilweise bei irrelevanten Seiten einen hohen und umgekehrt bei relevanten Seiten einen niedrigen Heuristikwert liefert. Um abschätzen zu können, ob es sich bei diesen Abweichungen um vernachlässigbare Ausnahmen handelt und um evtl. die Heuristik zu verbessern ist es nötig, eine Relation zwischen dem vom MWC gelieferten Heuristikwert und den gefundenen Viren zu visualisieren. Dazu dient das Programm „HeurRelation“.

3.3.1 Das Programm „HeurRelation“

Das Programm stellt in einem Koordinatenkreuz die Abhängigkeit dieser beiden Größen dar. Auf der x-Achse ist der Heuristikwert und auf der y-Achse die Anzahl der gefundenen Viren abgetragen. Im Idealfall sollten die Punkte auf der Winkelhalbierenden der x- und y-Achse oder zumindest in der Nähe liegen:

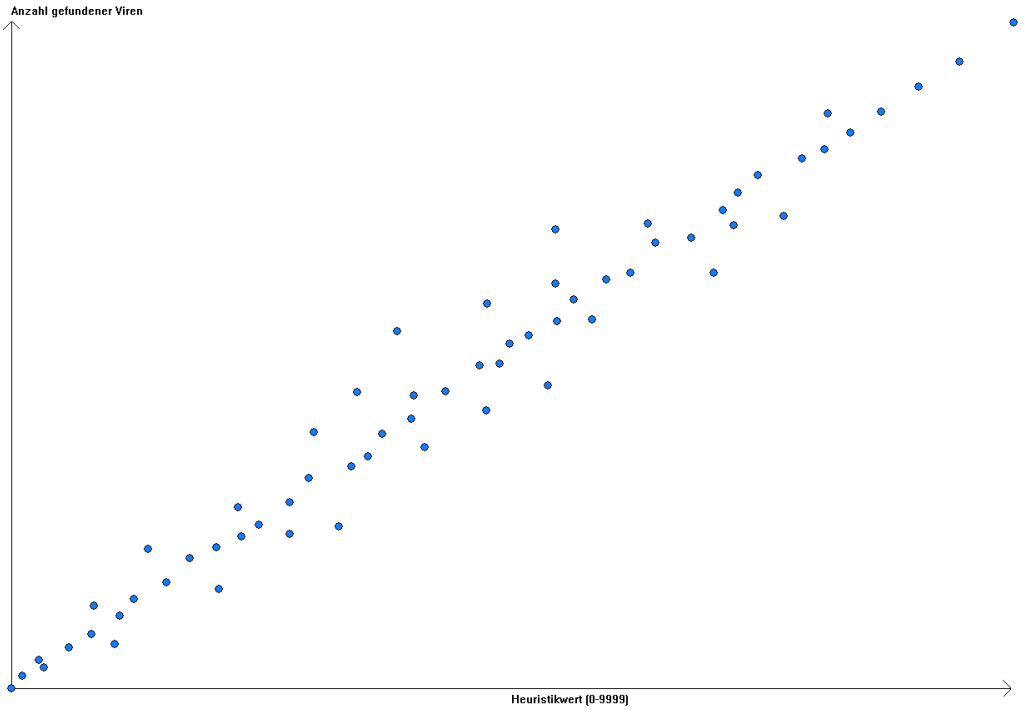


Abbildung 26 - Idealfall der Relation

Das Programm liest die Logdateien des Malware Crawlers ein und extrahiert die Internetadresse, den Heuristikwert und die Anzahl dort gefundener Viren. Die Werte der x-Achse reichen von 0 bis 9999, die Werte der y-Achse von 0 bis zum Maximum der auf einer Webpages gefundener Malware. Da dieses im Einzelfall recht viele sein können, kann auch ein maximaler y-Achsen Wert eingestellt werden. Webpages, die mehr Malware enthalten, werden dann am oberen Rand rot dargestellt.

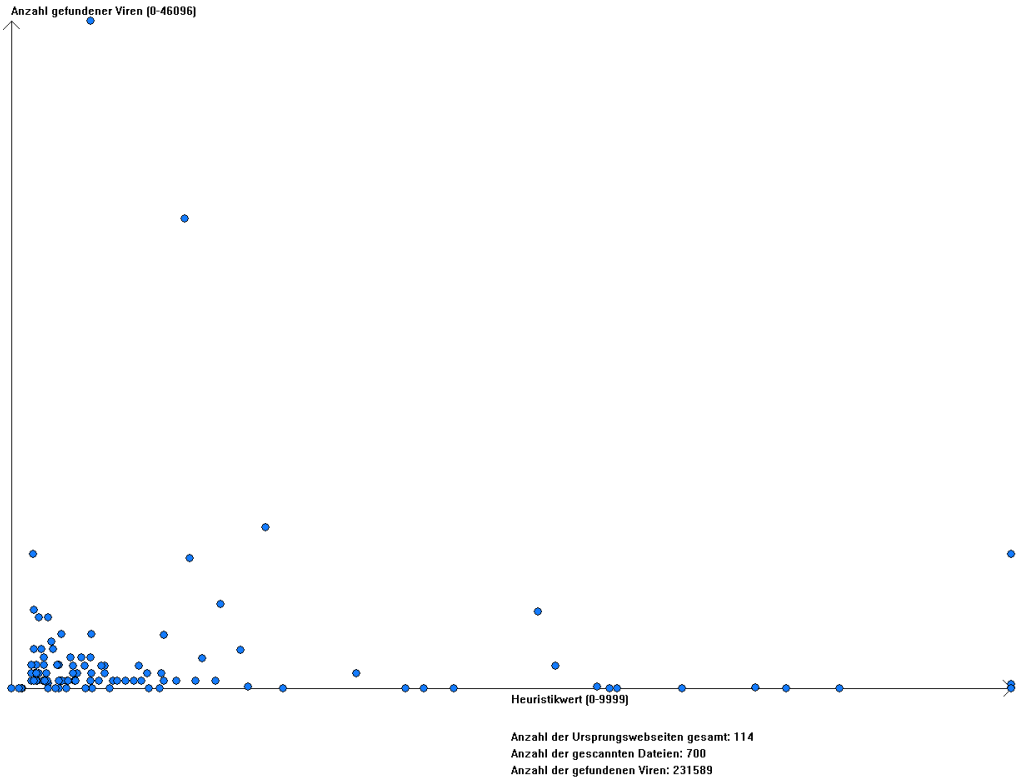


Abbildung 27 - Relation gefundenener Viren zum Heuristikwert des MWC (Daten von März bis Dezember 2001)

3.3.2 Ergebnis

In der oben gezeigten Abbildung kann man erkennen, dass das Ergebnis stark vom Idealfall abweicht. Das hat mehrere Gründe. Zum Einen versuchen einige Anbieter von Malware im WWW die Website möglichst unauffällig zu gestalten[32], um nicht zu leicht von Crawlern gefunden zu werden. Dies führt zu einem niedrigen Heuristikwert trotz einer Vielzahl angebotener Malware. Andererseits sind auch Seiten relevant, die zwar keine oder nur Malware anbieten aber zu solchen Seiten verlinken. Diese Websites haben dann berechtigterweise einen hohen Heuristikwert, obwohl dort nur wenig Malware zum Download angeboten wird.

Da einige Seiten mehrere Tausend[33] Viren zum Download anbieten, häufen sich die Websites im unteren Teil der Grafik. Durch Einstellung eines

[32] Z. B. durch Darstellung von Text in Bildern und anderen Methoden (siehe [MWC-00]).
[33] Die Website mit dem größten Angebot stellte 46096 (!) maliziöse Dateien bereit.

Maximalwertes von 4000 gewinnt man einen besseren Überblick über die Websites. Dabei kristallisiert sich eine zunächst überraschende Struktur:

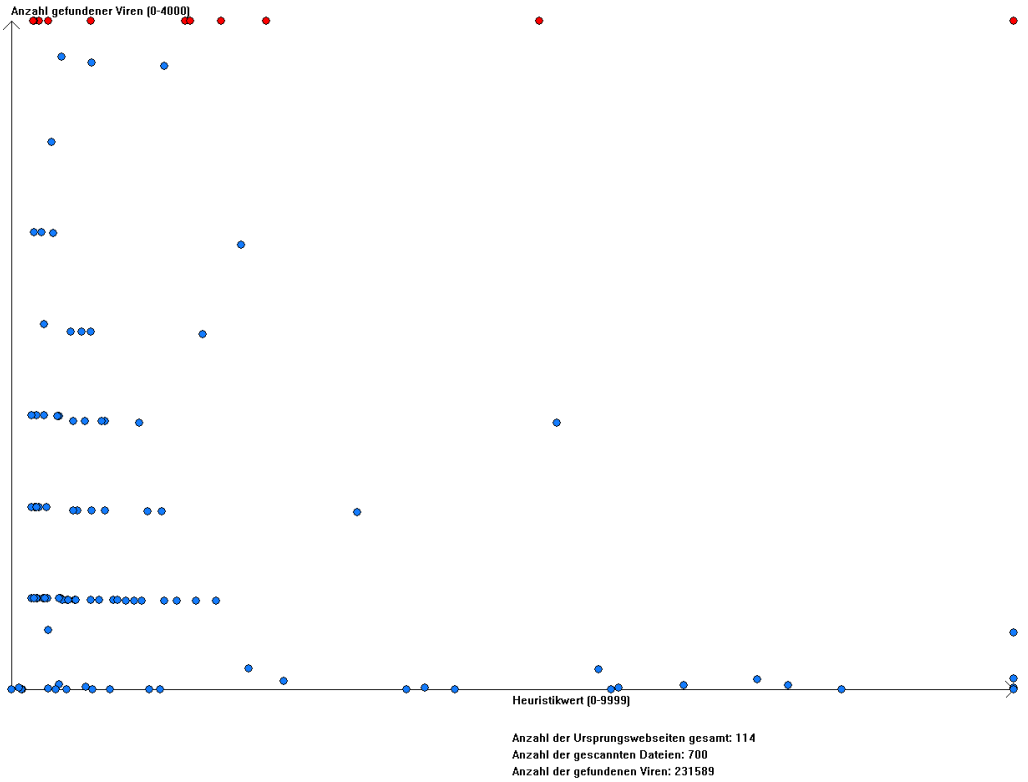


Abbildung 28 - Relation mit Maximalwert 4000

Man sieht mehrere Ebenen, auf denen Gruppen von Websites, die alle ungefähr die gleiche Anzahl von maliziösen Dateien anbieten, zu sehen sind. Die Ursache dafür liegt wahrscheinlich darin, dass sich diese Anbieter untereinander austauschen, so dass Websites der selben Ebene immer die gleiche Anzahl oder zumindest ähnlich viele Dateien zum Download anbieten. Weiterhin haben alle diese Gruppen einen relativ niedrigen Heuristikwert und je mehr Malware angeboten wird, desto niedriger ist der maximale Heuristikwert der Gruppe. Diese Websites scheinen also eher eine unauffällige Erscheinungsweise zu bevorzugen, um nicht aufzufallen und entfernt zu werden. Im oberen Bereich sind außerdem kleinere Gruppengrößen zu beobachten. Der Grund hierfür ist aber möglicherweise, dass die Websites mit

einer höheren Anzahl an Malware sich besser tarnen und viele Sites dadurch einen Heuristikwert von 0 ergeben und somit nicht erfasst wurden.

Zur weiteren Analyse der Ebenen werden diese wie folgt benannt:

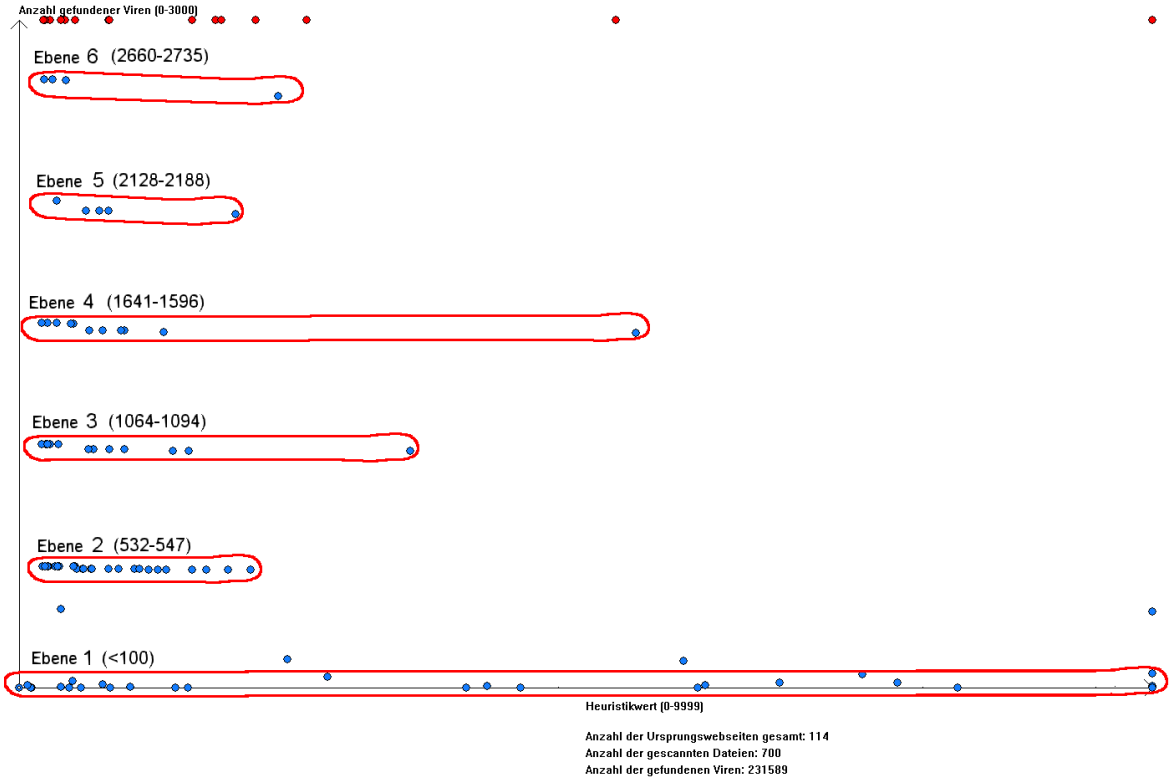


Abbildung 29 - Benennung der Ebenen

Es wird nun untersucht, inwieweit die Ebenen untereinander verlinkt sind und wie stark die Websites in einer Ebene verlinkt sind. Hierzu wurden die URLs der Ebenen mit dem Programm „HeurRelation“ in einzelne Dateien extrahiert, die dann mit dem „MWCVisualizer“ eingelesen und den vom AGN-Malware Crawler bereits gelieferten Link-Strukturen zugeordnet wurden. Die Gruppen wurden anhand der zugehörigen Ebene manuell angeordnet.

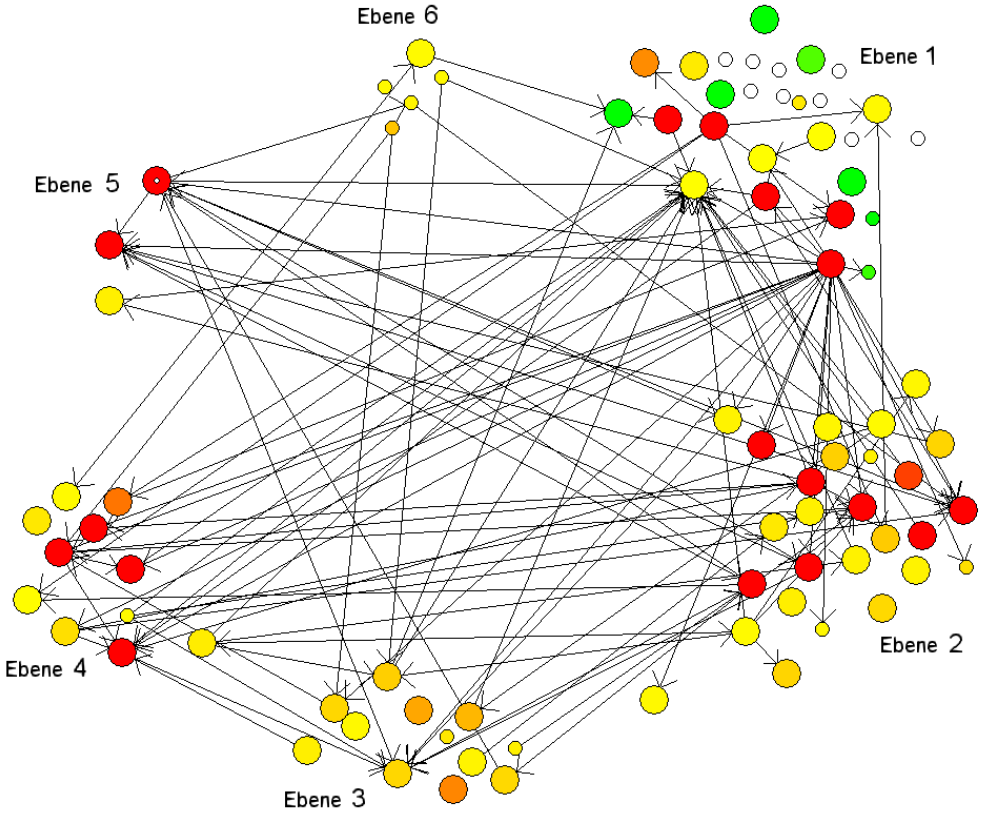


Abbildung 30 - Die Struktur der Ebenen (Daten von März bis Dez. 2001)

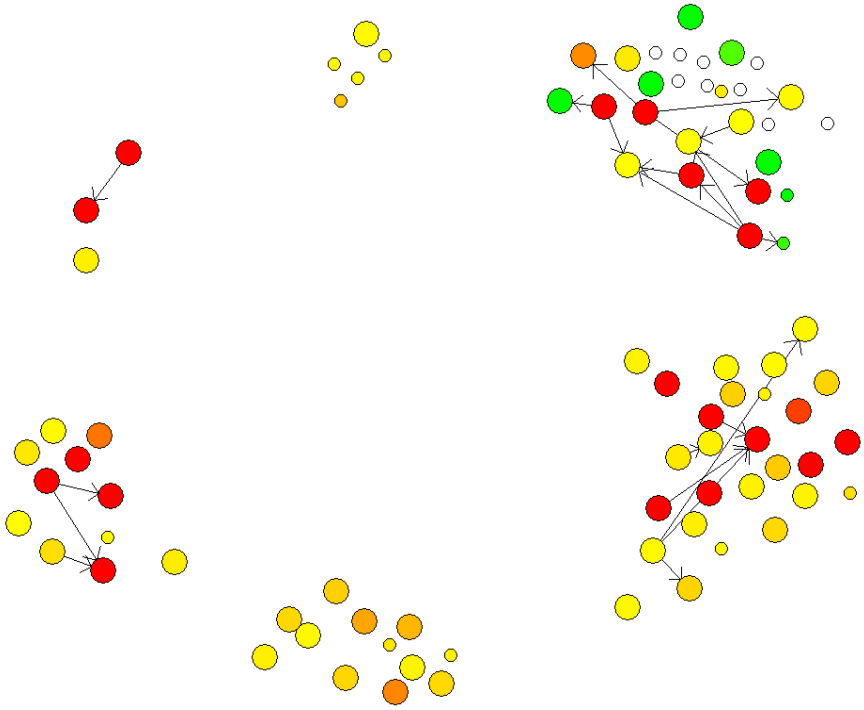


Abbildung 31 - Verlinkung der einzelnen Ebenen untereinander

In Abbildung 31 ist zu erkennen, dass die Verlinkung innerhalb der Ebenen nicht sehr hoch ist.
Um die Verlinkung jeweils zweier Ebenen besser untersuchen zu können, befinden sich, ergänzend zu Abbildung 30, Abbildungen mit den Links jeweils zweier Ebenen im Anhang. Für die weitere Analyse sind die Links zwischen den Ebenen zusätzlich in einer Tabelle zusammengefasst:

		nach Ebene					
		1	2	3	4	5	6
von Ebene	1	12	12	2	8	2	0
	2	7	10	6	5	6	0
	3	3	4	2	2	1	0
	4	5	5	3	3	0	1
	5	2	5	1	0	1	1
	6	2	1	2	2	0	0

Abbildung 32 - Anzahl der Links zwischen den Ebenen[34]

Es fällt auf, dass Verlinkungen der höheren Ebenen geringer ist[35], als bei den unteren Ebenen. Das hängt zum Einen mit der höheren Anzahl der URLs der oberen Ebenen zusammen. Zum Anderen kann es aber auch ein weiterer Hinweis auf das Bestreben der Websitebetreiber, nicht entdeckt zu werden, sein.

Als Fazit lässt sich festhalten, dass auch wenn die Qualität der Heuristik nicht eindeutig bewertbar ist, interessante Erkenntnisse aus der Relation gewonnen werden konnten.

[34] Die Werte der Diagonalen bezeichnen die Links innerhalb einer Ebene.
[35] Sowohl untereinander als auch zu den anderen Ebenen.

3.4 Test auf Veränderung von Webpages

Es ist zwar positiv, wenn Websites mit maliziösen Inhalten von den Freehostern aus dem Internet entfernt werden, jedoch tauchen die gleichen Seiten in den meisten Fällen schon kurze Zeit später an anderer Stelle wieder auf. Um die Webpages auch an der neuen Position ausfindig zu machen, macht man sich die starke Vernetzung dieser Seiten untereinander zunutze. Webpages, die zuvor Links zu den verschobenen Seiten enthielten, werden diese voraussichtlich nach dem Umzug aktualisieren. Man braucht diese Webpages also nur auf Veränderungen zu überwachen. Diese Aufgabe erledigt der MWC-Surveiller.

3.4.1 MWC-Surveiller

Der MWV-Surveiller ist in den AGN-Malware Crawler integriert und testet periodisch[36] alle Seiten in der „SurvList“ auf Veränderungen. Alle veränderten Seiten werden per E-Mail gemeldet, so dass evtl. neue Links in die Datenbank aufgenommen bzw. veraltete gelöscht werden können.

Bei der ersten intuitiven Implementation stellte sich heraus, dass es nicht ausreicht, die Veränderungen anhand einer Checksumme zu ermitteln. Bei einem ersten Testdurchlauf meldete der MWC-Surveiller schon nach einer Periode von 5 Minuten bei ca. 50% aller in der „SurvList“ enthaltenen Links eine Veränderung. Dies ist auf die Verwendung von Countern, Uhren-Elementen u.ä. zurückzuführen. Vielmehr müssen die Seiten gefiltert werden, um unwichtige Veränderungen zu ignorieren.

Der MWC-Surveiller sowie der eingebaute Filter werden im Folgenden vorgestellt.

[36] Zur Zeit einmal am Tag.

Durch einen Timer wird die Hauptfunktion des MWC-Surveillers, wie oben beschrieben, periodisch aufgerufen. Nun werden die in der „SurvList“ enthaltenen URLs der Reihe nach geladen und gefiltert. Anschließend wird eine Checksumme generiert und mit einer bereits vorhandenen verglichen. Alle veränderten Einträge werden zwischengespeichert und nach Durchlauf der „SurvList“ gesammelt per E-Mail gemeldet. Folgendes Flussdiagramm verdeutlicht dies.

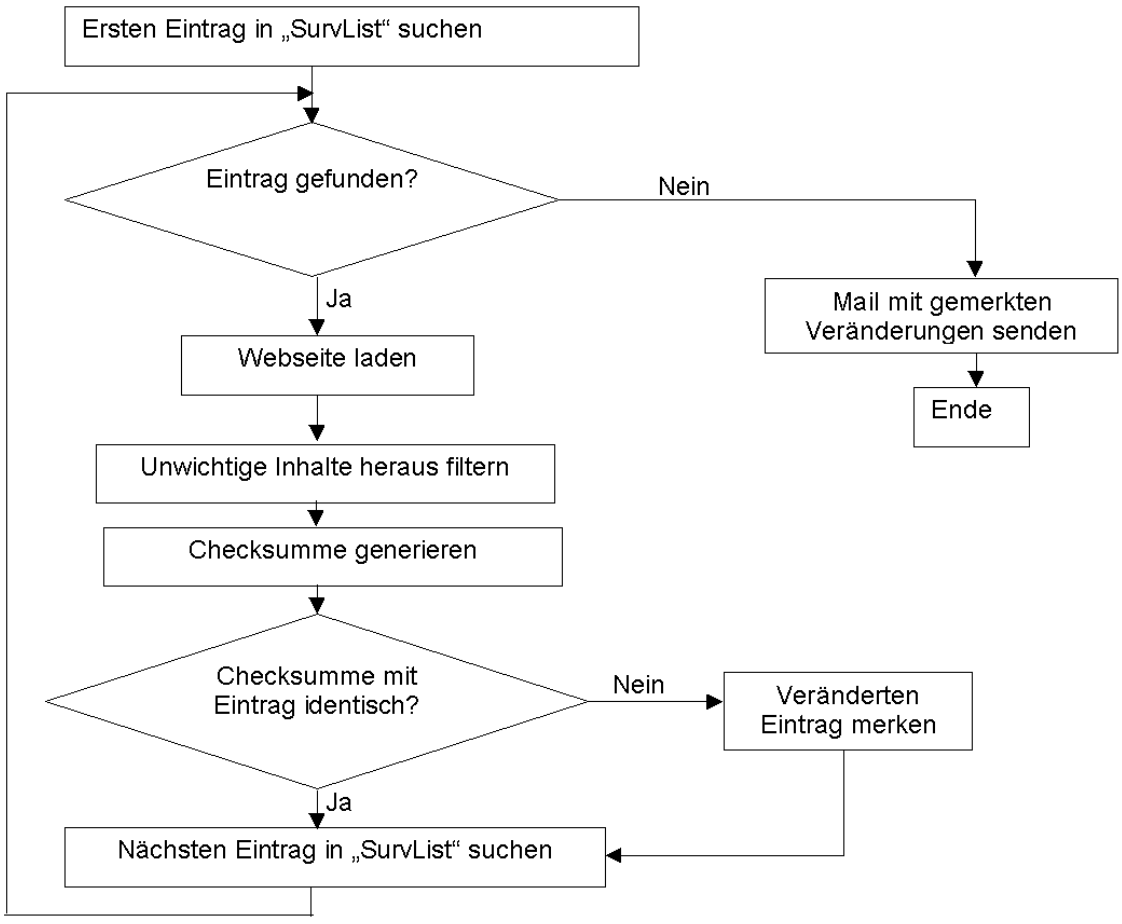


Abbildung 33 - Surveiller-Flussdiagram

Der Filter ist, wie auch die Link-, Tag- und Textextraktion beim AGN-Malware Crawler[37] in eine DLL ausgelagert worden. Es werden drei Parameter übergeben:

Parameter	Typ	Beschreibung	Rückgabe[38]
RetString	String	HTML-Code	Gefilterter Text
RetString2	String	HTML-Code	Extrahierte Links
Filter	String	String aus Nullen und Einsen (ASCII)	-

Die Parameter müssen mit einem Nullbyte (chr(0)) abgeschlossen werden. Die HTML-Datei wird in Blöcke zerlegt und der extrahierte Text zwischen den Blöcken wird durch senkrechte Striche (|) getrennt zurückgeliefert. Die Nullen und Einsen im Parameter „Filter“ stellen die Blöcke dar. Falls an der betreffenden Stelle im Filter eine Null steht, so enthält der zurückgelieferte Block anstatt des Textes einen Stern (*). Ist der Filter länger als die Anzahl der vorhandenen Blöcke, so werden die restlichen Nullen und Einsen ignoriert. Ist er dagegen kürzer, werden die restlichen Blöcke herausgefiltert.

Die speziell für den Surveiller hinzugekommene Tabelle „SurvList“ hat folgenden Aufbau:

Feld	Typ	Inhalt
BL_Infos	String[30]	Kurzbeschreibung
BL_URL	String[254]	URL der zu überwachenden Seite
BL_Visit	Datum	Datum
BL_Chksum	String[20]	Checksumme
BL_Changed	Boolean	Flag für Veränderung der Seite
BL_Mask	Memo	Filter
BL_Content	Memo	Maskierter Seiteninhalt
BL_Change	Memo	Veränderungen seit letztem Aufruf
BL_Cache	Memo	Cache (kompletter Seiteninhalt)
BL_AltCach	Memo	Alter Cache (vor der Veränderung)

[37] siehe [Soller01].
[38] Die Parameter werden „by reference“ übergeben, so dass sie gleichzeitig als Rückgabewerte fungieren können.

Zum Initialisieren wird der Filter zunächst vollständig mit Einsen gefüllt, um nichts auszufiltern. Nach zwei Durchläufen kurz hintereinander werden dann die Blöcke herausgefiltert, die sich verändert haben.

Die in der „SurvList“ enthaltenen URLs wurden manuell eingefügt, um möglichst relevante Seiten zu überwachen. Aus diesem Grund ist es sehr aufschlussreich, die Gruppendependenzen dieser Seiten zu betrachten. Hierzu wurden die Daten der „SurvList“ exportiert und in den MWC-Visualizer eingelesen. Da in der SurvList keine Informationen über den Heuristikwert und die Links von und zu anderen Seiten enthalten sind, wurden zuerst die Daten der „export.mwc“ eingelesen und dann die URLs, die gleichzeitig auch in der „SurvList“ enthalten sind (inklusive der Verlinkungsdaten und den Heuristikwerten) extrahiert. Um die Position innerhalb der Gesamtstruktur betrachten zu können, werden die restlichen Knoten nicht ganz entfernt, sondern nur ausgeblendet oder hellgrau dargestellt.
Die Datei „survlist.sur“ mit den URLs der SurvList wird durch folgende Visual FoxPro-Befehle im Direktfenster generiert:

```
set default to d:\src\vfp6\mwc
use surveil shar
copy to survlist.sur delimited with "#" all
```

Die Einträge der Datenbanktabelle werden so zeilenweise, durch Kommata getrennt und gegebenenfalls[39] in „#“ eingeschlossen als Textdatei exportiert.

3.4.2 Ergebnis

Die aus der „export.mwc“ extrahierten Websites der SurvList sind trotz der geringen Anzahl stark untereinander verlinkt. Es gibt zwar auch viele unabhängige Knoten, aber ein Großteil der Websites ist zusammenhängend.

[39] z. B. bei Texteinträgen wie etwa der URL oder der Beschreibung.

Die folgende Abbildung zeigt den zusammenhängenden Kern der extrahierten Websites:

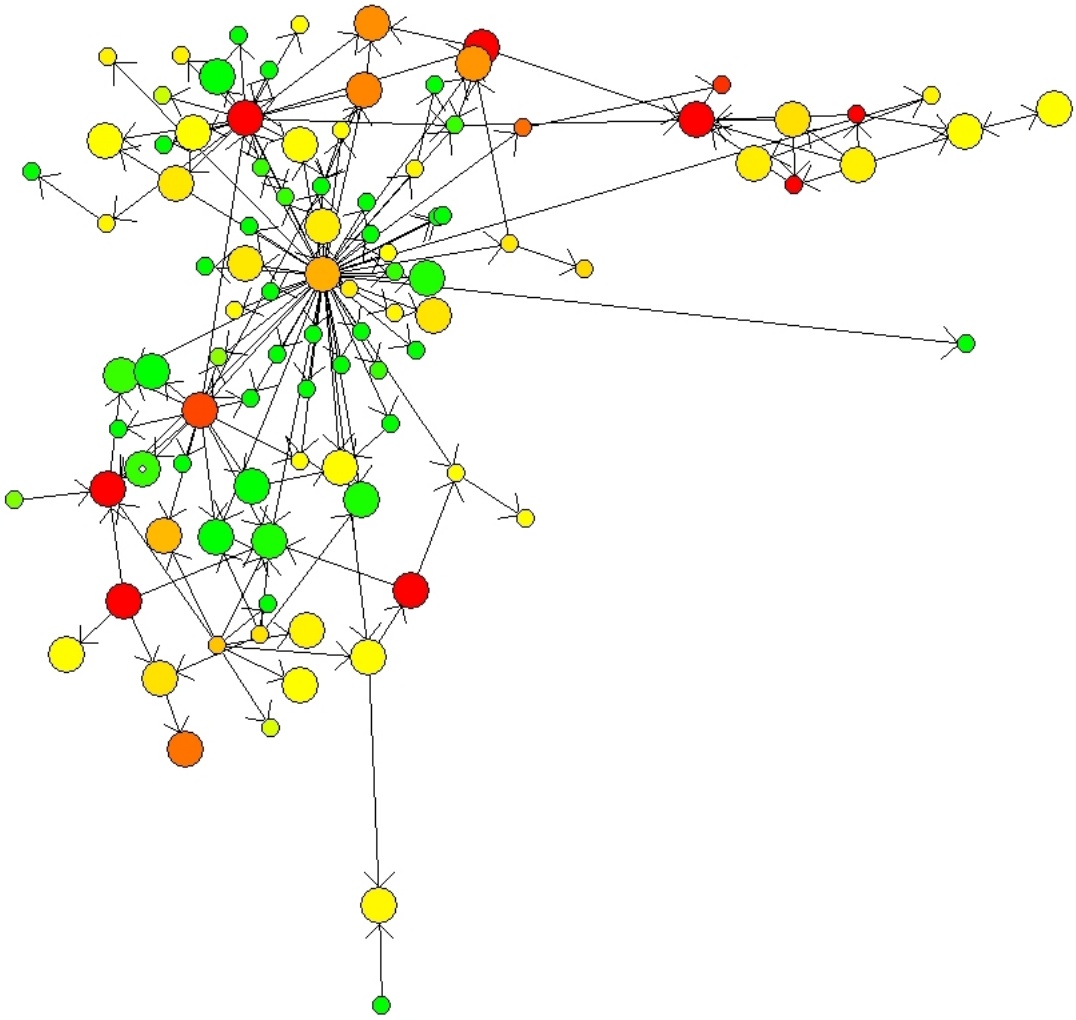


Abbildung 34 - Websitestruktur der „SurvList“ (vom Mai 2002)

4 Ausblicke

Es gibt eine große Anzahl von Möglichkeiten der Visualisierung von Netzstrukturen. Im Laufe dieser Diplomarbeit sind bereits einige Programme zur Visualisierung entstanden, die gute Ergebnisse liefern. Andere Visualisierungsformen sind aber durchaus denkbar und sinnvoll. Außerdem gibt es noch Verbesserungsmöglichkeiten der MWC-Heuristik. Hierdurch bekäme man mehr und bessere Daten. Es gibt bereits Überlegungen, die Heuristik durch Integration von HTML-Attributen in den Heuristikwert zu verbessern. Das Programm zur Darstellung der Relation zwischen gefundenen Viren und Heuristikwert ist eine gute Hilfe beim Verbessern der Heuristik, da hierdurch eine direkte Kontrolle der Qualität erfolgt. Im Folgenden werden einige andere Ansätze der Visualisierung sowie mögliche Erweiterungen des Malware Crawlers aufgezeigt.

4.1 3D-Darstellung

Eine Ausdehnung der Visualisierung in die dritte Dimension könnte einen besseren Überblick über das nicht leicht zu überschauende Dickicht von Websites verschaffen. Die Sortierung der Knoten wäre sicherlich effektiver, da eine weitere Dimension wesentlich mehr Bewegungsraum schaffen würde. Die Knoten würden sich nicht mehr so leicht am Erreichen ihrer optimalen Position hindern, da es nun die dritte Dimension zum Ausweichen gibt. Es ist ohne Weiteres möglich, die bestehenden Algorithmen an den 3D-Raum anzupassen. Die Schwierigkeit besteht allerdings in der enormen Datenmenge, die bewältigt werden muss. Zu den ohnehin schon bestehenden Problemen des hohen Datenaufkommens käme nun das Problem, die Repräsentation in angemessener Geschwindigkeit abzubilden. Standardmäßig liefert Visual C++ nur Grafikroutinen zum Darstellen von zweidimensionalen Objekten wie z.B. Kreise und Linien. Bei einer Berechnung im 3D-Raum müssen die Daten also durch eigene Algorithmen wieder in den 2D-Raum umgerechnet werden, um sie

am Bildschirm darstellen zu können. Dies bedeutet wiederum mehr Rechenaufwand. Außerdem müssten Algorithmen zum Drehen und Zoomen des Graphen implementiert werden, um diesen auch von allen Seiten betrachten zu können. Der Schwerpunkt bei der Realisierung einer 3D-Darstellung liegt also in der Implementierung und Optimierung von schnellen Algorithmen zur 3D-Darstellung.
Zur Präsentation im Internet wäre eine Darstellung im VRML-Format denkbar, wobei nur kleine Ausschnitte dargestellt werden könnten. G. Lehnert [Lehnert] kommt in einer Abschätzung zur Größe von Cone Trees in VRML-Dateien zu dem Ergebnis, dass eine Datei mit 10.000 Knoten ca. 4400 KB belegen würde. Der größte in dieser Arbeit betrachtete Graph besteht aus ca. 86.000 Knoten und würde demnach ungefähr 37,84 MB belegen. Der Download einer so großen Datei ist für den normalen Anwender (noch) nicht zumutbar. Für Spezialisten wäre dies zwar keine große Hürde, allerdings kommen noch weitere erschwerende Faktoren hinzu. So hätte ein Programm zur Anzeige von VRML Schwierigkeiten, die Datei in angemessener Zeit einzulesen, auszuwerten und darzustellen. Hinzu kommt, dass es sich hierbei nicht nur um Baumstrukturen handelt, sondern um komplexe Graphen. Zur Geschwindigkeit schreibt G. Lehnert:

[...] Um die 1652 Knoten darzustellen, brauchte der Pentium 200 MMX Rechner immerhin schon einige Minuten[...]. Die Navigation innerhalb dieser erschaffenen Welt erwies sich als sehr schwergängig, da die Anforderungen an die Hardware enorm hoch sind.

Auch wenn die Leistungsfähigkeit der Computer gestiegen ist und weiter steigt, wird es noch einige Zeit dauern, um alle 86.000 Knoten in akzeptabler Geschwindigkeit darstellen zu können.

4.2 Grafische Darstellung der Zeitlichen Änderungen

Als weitere Erweiterung ist auch das Hinzunehmen der Zeit-Dimension denkbar. So ließe sich eventuell ermitteln, in welchen Regionen gelöschte Websites besonders häufig wieder auftauchen und wo besonders oft Änderungen vorgenommen werden. Dabei kann es sogar sein, dass gelöschte Websites im Graphen wieder an derselben Stelle auftreten, da sie wieder von denselben Websites verlinkt wurden. Außerdem könnte man beim Crawlen und Download Prioritäten setzen und Bereiche, in denen kaum Veränderungen stattfinden, weniger oft untersuchen als solche, bei denen sehr häufig Änderungen festgestellt werden.

4.3 Anbindung eines Navigationssystems an den MWC

Eine benutzerfreundliche Erweiterung des Malware Crawlers ist die Anbindung eines Navigationssystems. Hierbei wäre es möglich, die bereits besuchten Websites als Graph darzustellen und dem Anwender durch Anklicken von Links in unbesuchte Regionen zu ermöglichen, die Suchrichtung zu beeinflussen. Die Heuristik würde somit durch die menschliche Intelligenz ergänzt. Allerdings besteht dadurch natürlich auch die Gefahr, sich in die Irre führen zu lassen, sich „zu verlaufen“.

Um einen genauen Überblick über die Relevanz der Seiten zu bekommen, könnte man eine manuelle Bewertung der Seiten hinzufügen. Seiten können automatisch an den Internet Explorer gesandt und dort beurteilt werden. Eine Beurteilungsskala könnte beispielsweise wie folgt aussehen:

Beurteilungswert	Bedeutung
0	Unbekannt
1	Harmlos
2	Links auf Virensites (ohne das Anbieten von Malware)
3	Virensites (Anbieten von Malware)

Natürlich ist auch eine feinere Abstufung denkbar. Der ermittelte Wert könnte dann anstatt des Heuristikwertes farblich in die Grafik eingebunden werden. Bei unbekanntem Wert oder noch nicht beurteilten Seiten würde weiterhin die Farbe des Heuristikwertes erscheinen. Es müsste hierbei natürlich kenntlich gemacht werden, dass es sich nicht um eine manuelle Beurteilung handelt. Dies wäre möglich, indem verschieden Darstellungsformen der Knoten, z. B. Quadrate für manuelle Beurteilungswerte und Kreise für Heuristikwerte, verwendet werden. Natürlich können aufgrund der großen Datenmenge nicht alle Knoten manuell bewertet werden, aber für sehr relevante oder sehr irrelevante Bereiche kann dies sehr nützlich sein. Es gibt auch Aufschluss über die Qualität der Heuristik.

Anhang A: Detailierte Bilder des Graphen der MWC-Daten

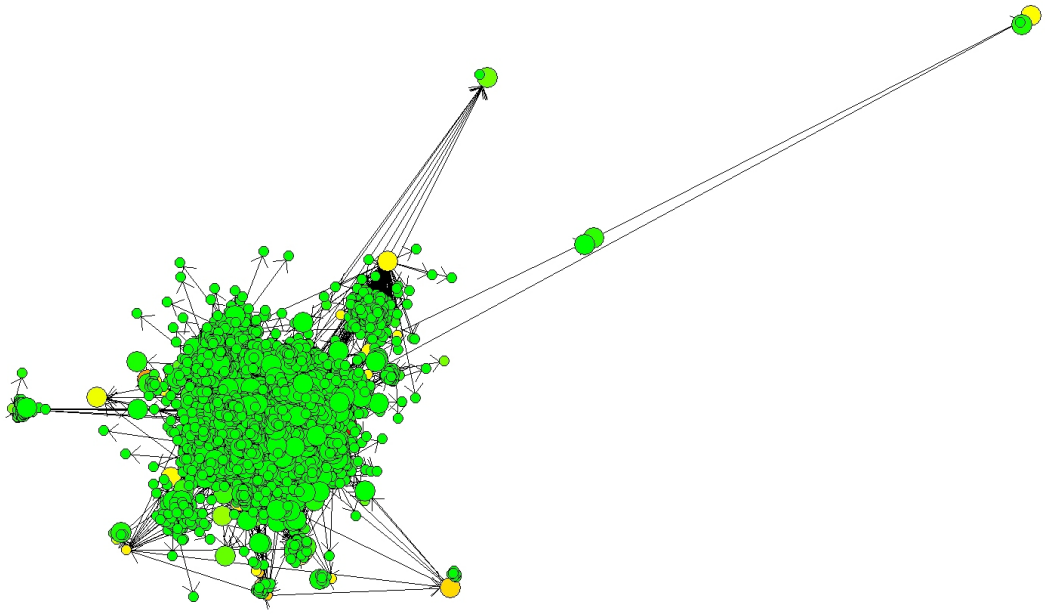


Abbildung 35 - Graph der Daten vom September 2002 nach 21 Iterationen

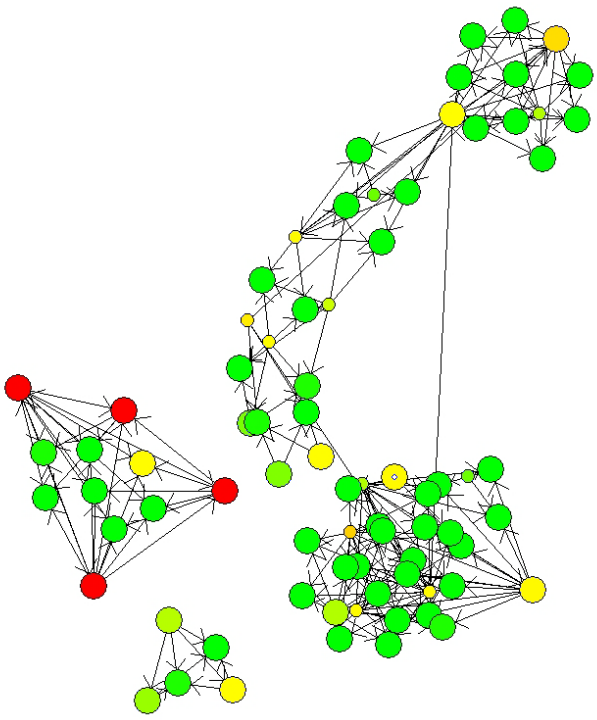


Abbildung 36 - Nur 3er-Bündel

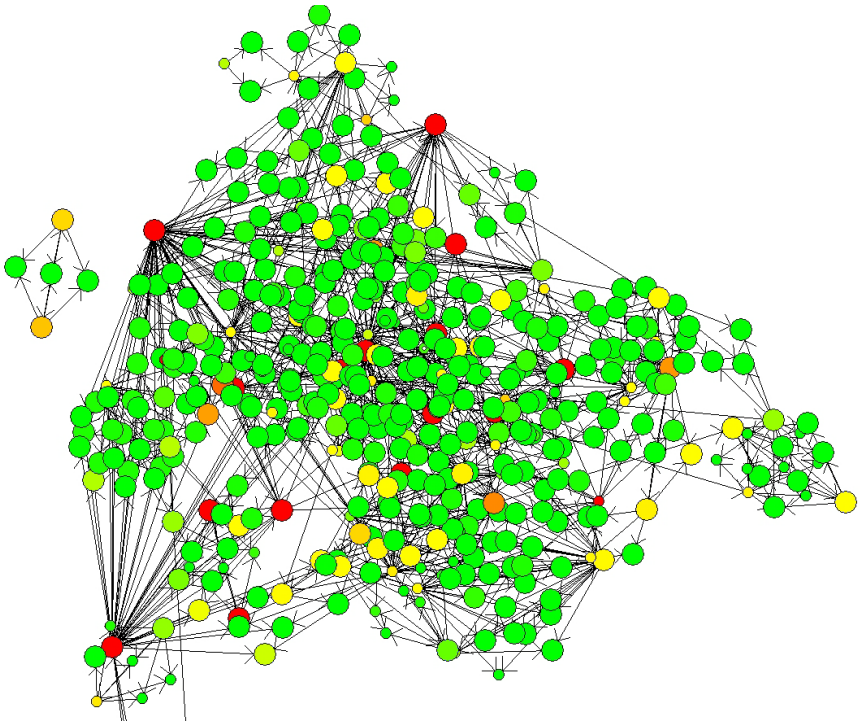


Abbildung 37 - Nur 2er-Bündel

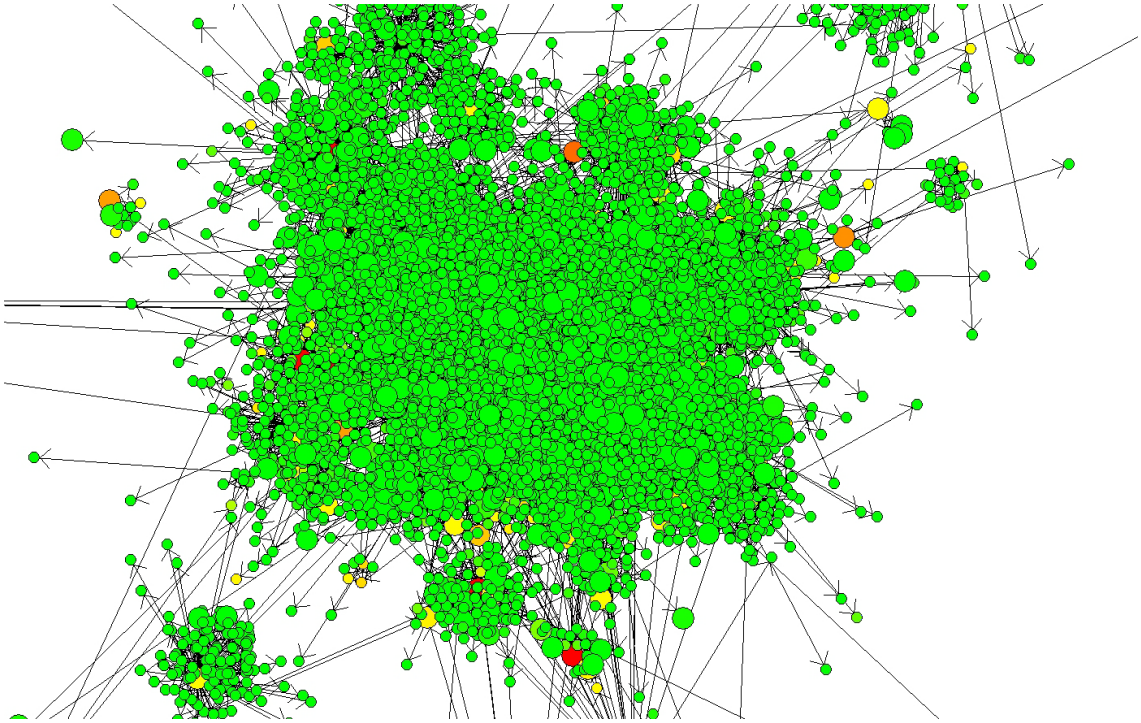


Abbildung 38 - Nur 1er-Bündel

Verlinkungen jeweils zweier Ebenen (Heuristikrelation)

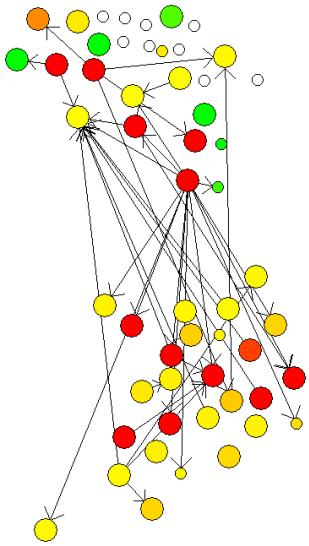


Abbildung 39 – Links zwischen Ebene 1 und Ebene 2

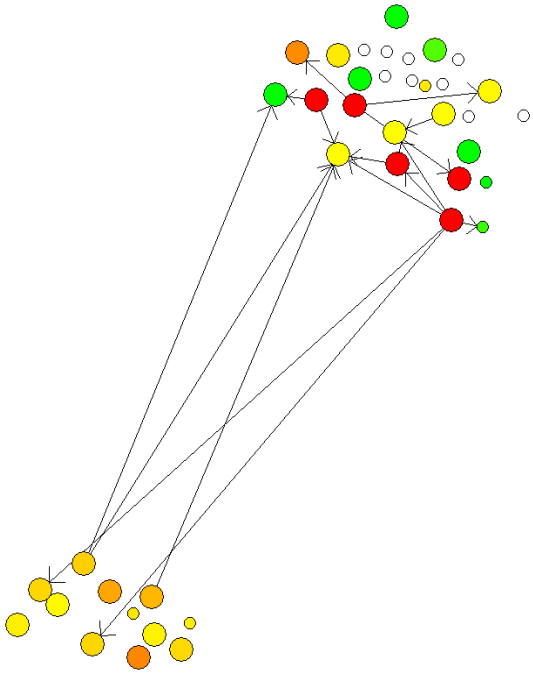


Abbildung 40 - Links zwischen Ebene 1 und Ebene 3

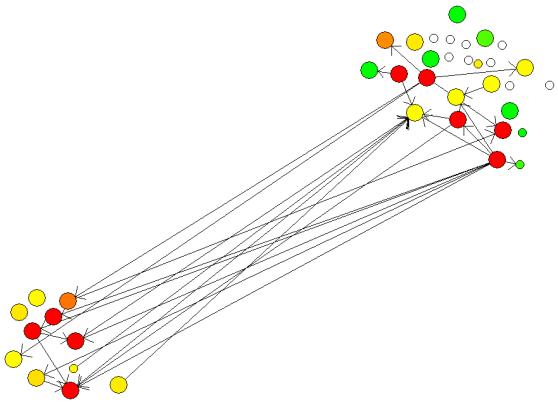


Abbildung 41 - Links zwischen Ebene 1 und Ebene 4

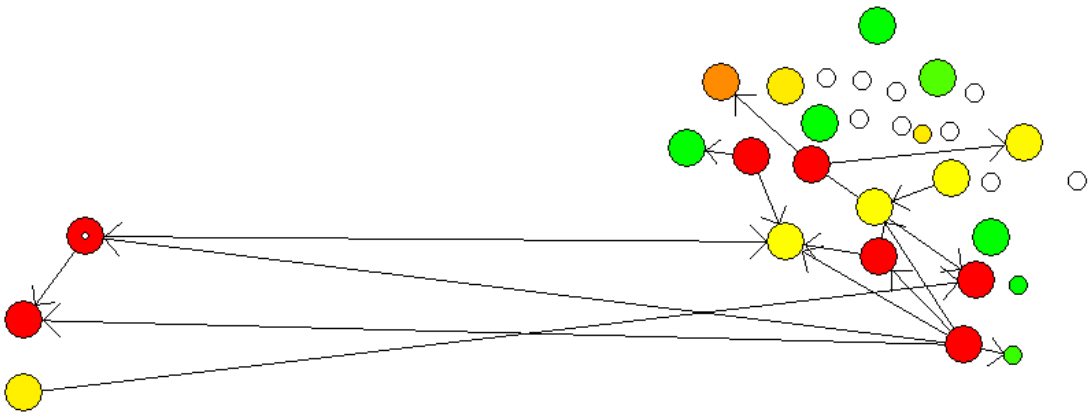


Abbildung 42 - Links zwischen Ebene 1 und Ebene 5

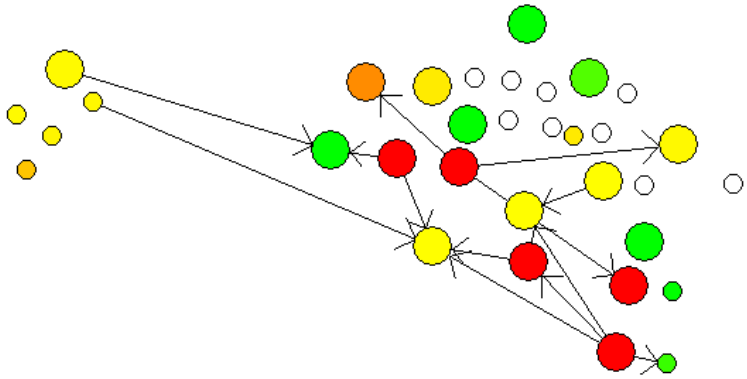


Abbildung 43 - Links zwischen Ebene 1 und Ebene 6

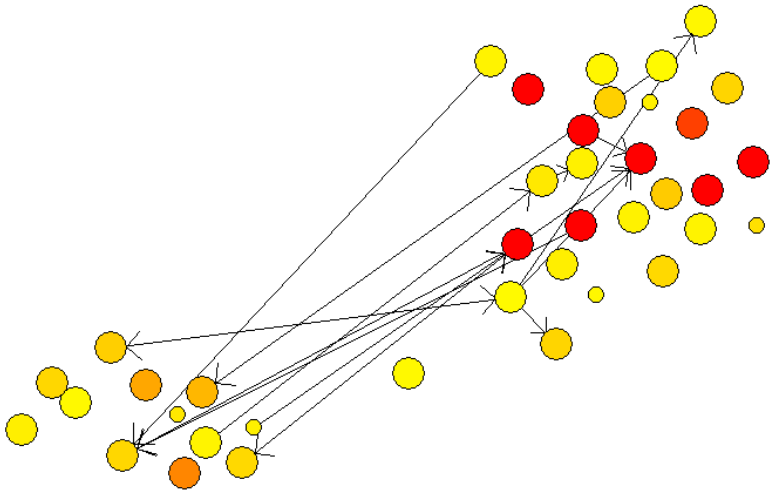


Abbildung 44 - Links zwischen Ebene 2 und Ebene 3

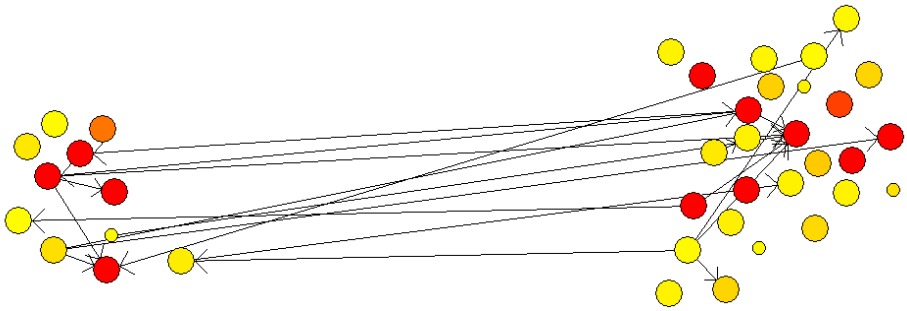


Abbildung 45 - Links zwischen Ebene 2 und Ebene 4

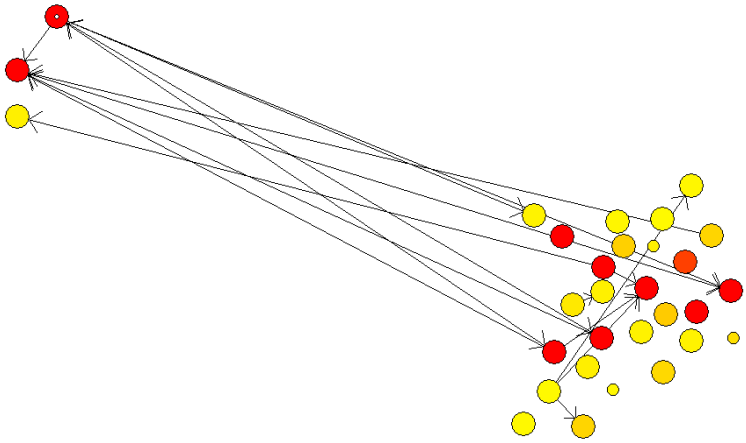


Abbildung 46 - Links zwischen Ebene 2 und Ebene 5

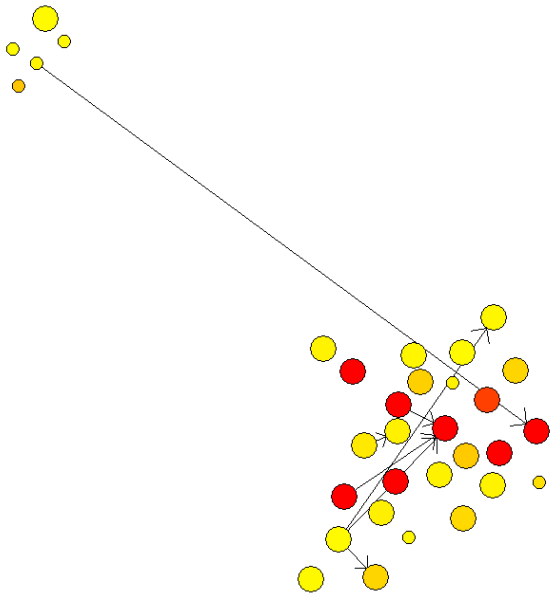


Abbildung 47 - Links zwischen Ebene 2 und Ebene 6

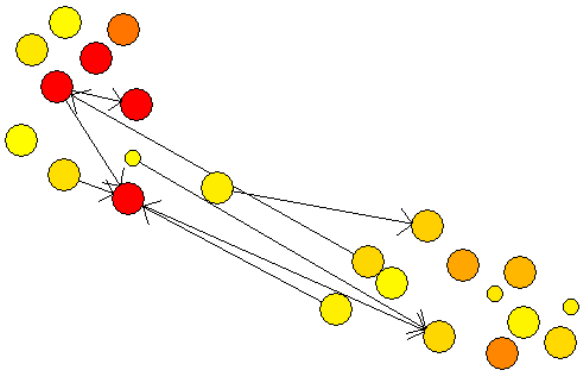


Abbildung 48 - Links zwischen Ebene 3 und Ebene 4

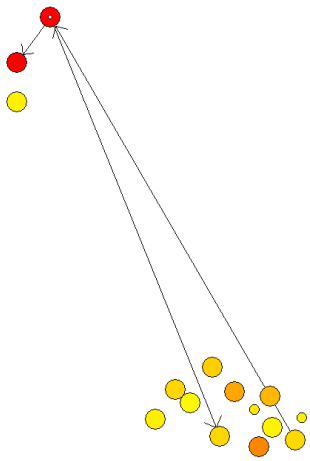


Abbildung 49 - Links zwischen Ebene 3 und Ebene 5

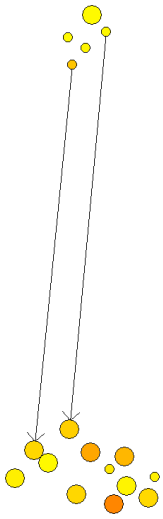
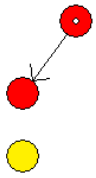


Abbildung 50 - Links zwischen Ebene 3 und Ebene 6



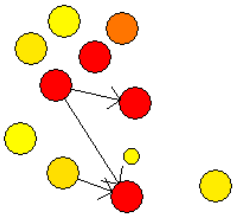


Abbildung 51 - Links zwischen Ebene 4 und Ebene 5

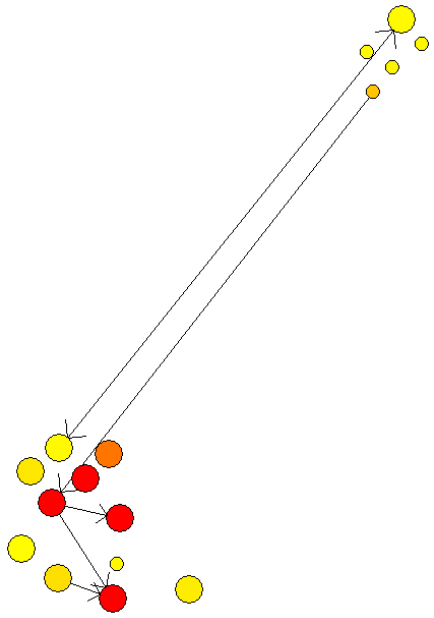


Abbildung 52 - Links zwischen Ebene 4 und Ebene 6

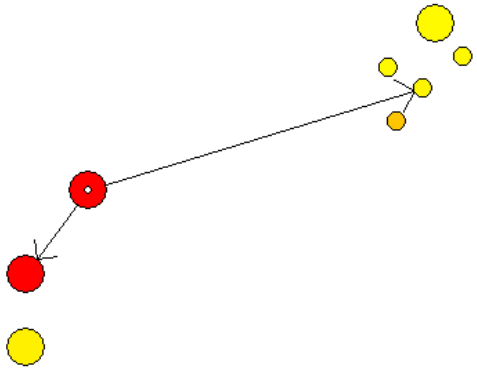


Abbildung 53 - Links zwischen Ebene 5 und Ebene 6

Anhang B: Programmbeschreibung zum „MWC-Visualizer“

Der MWC-Visualizer ist das Kernprogramm und somit das umfangreichste Programm der Diplomarbeit. Es liest die vom AGN Malware-Crawler gelieferten Daten ein und visualisiert sie in Form eines Graphen. Dabei werden zusammengehörende Webpages zu Websites zusammengefasst und der vom Malware-Crawler errechnete Heuristikwert farblich dargestellt. Der Graph lässt sich durch das automatische Umsortieren der Knoten in eine übersichtliche Struktur bringen. Das Programm bietet zahlreiche Funktionen zur Analyse des Graphen.

B.1 Installation und Beispieldaten

Zur Installation wird das ZIP-Archiv mit den Dateien in ein beliebiges Verzeichnis entpackt. In diesem Verzeichnis befinden sich anschließend auch die Verzeichnisse „WebsiteExtern“, „WebsiteIntern“ und „Bilder“. Dort befinden sich nach der Webpagegenerierung alle Dateien, die für die Darstellung im Internet notwendig sind und die abfotografierten Bilder.

B.2 Bedienung

Nach dem Start (MWCVisualizer.exe) sieht man einen leeren Bildschirm und folgende Menü- und Symbolleisten:

Abbildung 54 - Menü- und Symbolleisten

Einige Funktionen sind bis zum Öffnen einer Datei oder bis zum Erzeugen eines Knoten außer Betrieb (grau). In der obigen Abbildung wurden allerdings zur besseren Übersicht die vollständig aktiven Symbolleisten dargestellt.

B.2.1 Die Symbolleisten

Bei niedriger Bildschirmauflösung kann es sein, dass die Symbole auf der rechten Seite der Symbolleiste nur halb oder gar nicht mehr sichtbar sind. Die Symbole sind daher so angeordnet, dass sich die weniger oft benötigten Funktionen auf der rechten Seite befinden. Des Weiteren sind die Funktionen auf zwei Symbolleisten aufgeteilt. Die untere Symbolleiste enthält weniger oft benötigte Funktionen und läßt sich verschieben oder ausblenden. Alle Funktionen sind außerdem auch über das Menü aufrufbar. Die folgende Tabelle gibt einen kurzen Überblick über die Funktionen der Symbolleisten und die entsprechende Position im Menü. Da anschließend alle Funktionen des Menüs genauer erläutert werden, wird hier nur stichwortartig die Funktion beschrieben.

Symbol	Funktion	Position im Menü
	Datei neu	Datei → Neu
	Speichern	Datei → Speichern
	Datei öffnen	Datei → Öffnen
	Knoten löschen	Bearbeiten → aktuellen Knoten löschen
	HTML erzeugen	Bearbeiten → HTML-Datei erzeugen (intern)
	HTML erzeugen	Bearbeiten → HTML-Datei erzeugen (extern)
	JPG erzeugen	Bearbeiten → Bildschirm abfotografieren
	Einstellungen	Bearbeiten → Einstellungen
	Dateieigenschaften	Bearbeiten → Dateieigenschaften

	Info	Ansicht → Info über den Knoten / Graphen
	Sortierung anhalten	Bearbeiten → Sortierung anhalten
	Sortierung starten	Bearbeiten → Sortierung starten
	Sortieren	Bearbeiten → Sortieren (eine Iterartion)
	Knoten neu verteilen	Position → Knoten zufällig verteilen

Symbolleiste 2

Symbol	Funktion	Position im Menü
	Hineinzoomen	Ansicht → Hineinzoomen
	Hinauszoomen	Ansicht → Hinauszoomen
	Standardzoom	Ansicht → Standardzoom
	Zentrieren	Position → Zentrieren (Ursprung)
	Zentrieren	Position → Zentrieren (Graphmittelpunkt)
	Zentrieren	Position → Zentrieren (Markierter Knoten)
	1er Bündel zeigen	Ansicht → 1er Bündel zeigen
	2er Bündel zeigen	Ansicht → 2er Bündel zeigen
	3er Bündel zeigen	Ansicht → 3er Bündel zeigen
	Alle Filter aus	Ansicht → Alle Filter ausschalten
	Knoten befestigen / lösen	Bearbeiten → aktuellen Knoten befestigen / lösen
	Alle Knoten lösen	Bearbeiten → alle Knoten lösen
	Zusammenhangstest	Ansicht → auf Zusammenhang testen

B.2.2 Das Menü

Im Folgenden werden alle Funktionen des MWC-Visualizers genau beschrieben. Hierbei werden die Funktionen in der Reihenfolge erläutert, in der sie sich im Menü befinden.

Datei

Neu (alle Knoten löschen): Alle vorhandenen Knoten und Links werden entfernt und der Iterationszähler auf Null gesetzt.

Öffnen: Es können zwei Dateiarten geladen werden: VIS- und MWC-Dateien. Die MWC-Dateien sind die vom Malware Crawler gelieferten Dateien. Sie enthalten nur die Graphstruktur ohne die Knotenpositionen. Aus diesem Grund werden die Knoten zunächst zufällig verteilt. Die VIS-Dateien enthalten dagegen auch die Positionsdaten, die Zoomstufe, Bildschirmposition und Iterationszähler.

Speichern: Der Graph wird unter dem aktuellen Namen gespeichert. Die aktuelle Zoomstufe und die Bildschirmposition werden dabei ebenfalls gespeichert. Ebenso der Iterationszähler und die Dateieigenschaften (Natürliche Länge der Federn usw.)

Speichern unter: Hier kann der Dateiname und –typ, unter dem gespeichert werden soll, ausgewählt werden. Es werden zwei Dateiformate unterstützt: VIS und GML.
Der Graph kann zum Einen im internen VIS-Format gespeichert werden. Hierbei werden auch darstellungsspezifische Daten gespeichert, wie z.B. die aktuelle Zoomstufe oder die Bildschirmposition. Das VIS-Format ist ein sehr spezielles Format, dass (bisher) von keiner anderen Anwendung unterstützt

wird. Um dennoch eine weiterführende Bearbeitung in verschiedenen anderen Anwendungen[40] zu ermöglichen, kann der Graph aber auch als GML-Datei gespeichert werden.

Untermenge laden: Von anderen Programmen (z. B. Malware Crawler oder HeurRelation) erzeugte Ist-Dateien können mit dieser Funktion eingelesen werden. Diese Knoten werden dann im aktuellen Graphen markiert.

Untermenge hinzufügen: Falls bereits eine Untermenge geladen wurde, können mit dieser Funktion weitere Untermengen hinzugefügt werde. Die bestehenden Markierungswerte werden hierbei um Eins erhöht.

Bearbeiten

Aktuellen Knoten löschen: Der aktuelle Knoten und alle Verbindungen von und zu diesem Knoten werden gelöscht.

Unzusammenhängende Komponenten löschen: Der Graph wird auf Zusammenhang getestet. Falls er nicht zusammenhängend ist und genau eine größte Komponente existiert, werden alle Komponenten (ausgenommen der größten) gelöscht.

Ausgeblendete Knoten löschen: Alle Knoten, die aufgrund des Filters ausgeblendet wurden, werden gelöscht.

Unmarkierte Knoten löschen: Alle Knoten, denen kein Markierungswert zugewiesen wurde, werden gelöscht.

HTML-Datei erzeugen (intern / extern): Der aktuelle Bildschirminhalt wird abfotografiert, als JPG-Datei gespeichert und mit Legende und allgemeinen Informationen zum Graphen in ein HTML-Dokument eingebunden. Dieses

[40] z.B. Graphlet [Graphlet99], einem Programm zum Bearbeiten von Graphen.

befindet sich anschließend im Verzeichnis „WebsiteIntern“ bzw. „WebsiteExtern“. Die externe Website enthält nur die Grafik, ohne eingebundene Links.

Bildschirm abfotografieren: Der aktuelle Bildschirminhalt wird abfotografiert und mit Hilfe einer Bibliothek [Jpg] als JPG-Datei gespeichert. Das Bild wird im Verzeichnis „Bilder“ gespeichert, wobei im Dateinamen noch Informationen über die Anzahl der bisherigen Iterationen, die Zoomstufe und die Bildschirmposition (Position des virtuellen Schirms) stehen.
Es ist zu beachten, dass der aktuelle Bildschirminhalt im Bereich des Programmfensters abfotografiert wird, was nicht notwendigerweise dem Fensterinhalt entspricht. Ist beispielsweise ein anderes Fenster geöffnet und liegt dieses vor dem Programmfenster, so wird es anstatt des Graphen abfotografiert.

Knoten zufällig verteilen: Den Knoten werden wie beim Laden einer MWC-Datei wieder zufällige Positionen zugeteilt. Der Iterationszähler wird auf Null gesetzt.

Sortierung starten: Startet die Sortierung. Die Knotenpositionen werden laufend neu berechnet. Die Anzahl der Iterationen sowie der Wert zur Veränderung des Graphen wird laufend in der Statuszeile angegeben.

Sortierung anhalten: Hält eine laufende Sortierung an. Die laufende Iteration wird allerdings noch sortiert.

Knoten ordnen (eine Iteration): Berechnet einmal alle Knotenpositionen neu.

Knoten befestigen / lösen: Der aktuelle Knoten wird fixiert bzw. die Fixierung aufgehoben. Die Positionen befestigter Knoten wird bei Neuberechnung der Knotenpositionen nicht geändert. Befestigte Knoten wirken aber trotzdem auf

die anderen Knoten. Durch die Befestigung einiger Knoten wird eine Bewegung (z.B. Drehung) des gesamten Graphen trotz statischer Gestalt verhindert. Fixierte Knoten sind durch einen kleinen Kreis am Rand markiert.

Alle Knoten lösen: Alle befestigten Knoten werden wieder gelöst. Das erspart das Suchen und das einzelne Lösen befestigter Knoten.

Einstellungen: Folgende allgemeine Einstellungen können verändert werden

- **Filterwert:** Knoten mit einem niedrigerem Heuristikwert als der hier eingestellten werden herausgefiltert.
- **Mindestwert / Maximalwert:** Es werden nur Knoten mit Heuristikwerten angezeigt, die innerhalb dieser beiden Grenzen liegen.
- **Nur diese beiden Randwerte anzeigen:** Ist diese Funktion aktiv, so werden nur Knoten mit einem der beiden oben eingestellten Markierungswerte angezeigt.
- **Grenzwert für den Farbverlauf**: Der Farbverlauf ist aufgeteilt in 2 lineare Verläufe: von Grün (Heuristikwert 0) bis Gelb (der hier frei einstellbare Grenzwert) und von Gelb bis Rot (Heuristikwert 9999). Dadurch kann die nichtlineare Verteilung der Heuristikwerte besser visualisiert werden.
- **Gefilterte Knoten/Kanten ganz ausblenden**: Ist diese Funktion aktiv, so werden ausgeblendete Knoten/Kanten nicht grau dargestellt, sondern vollständig ausgeblendet.
- **Markierung anzeigen von ... bis:** Es werden nur Knoten angezeigt, die innerhalb dieser Markierungswerte liegen.
- **Alle Filter zurücksetzen:** Alle Filter werden ausgeschaltet.
- **Sicherheitsabfragen an/aus:** Wenn die Sicherheitsabfragen ausgeschaltet sind wird nicht nochmal nachgefragt, wenn z. B. ein oder alle Knoten gelöscht wird/werden.
- **Neue Knoten manuell hinzufügen:** Erlaubt das manuelle hinzufügen von Knoten mit der rechten Maustaste.

- **Wartedialog beim Sortieren anzeigen:** Mit dieser Funktion kann ein Wartedialog mit Abbruch-Button beim Umsortieren der Knoten eingeblendet werden.
- **Gitterlinien an/aus und Gitterabstand:** Wenn die Gitterlinien eingeschaltet sind werden sie in dem hier eingestellten Abstand angezeigt.
- **Qualität der JPG-Bilder:** Hier wird die Qualität der JPG-Bilder auf einer Skala von 1-100 eingestellt. Je niedriger der Wert, desto kleiner ist die erzeugte Datei und desto geringer ist die Qualität.
- **JPG erzeugen an/aus und Periode der Erzeugung:** Hier wird eingestellt, ob der Bildschirm nach jeder x-ten Iteration abfotografiert werden soll. So kann später eine Animation des Sortiervorgangs erstellt werden. Falls der Bildschirm abfotografiert wird, muss das Fenster mit dem Graph sichtbar sein, ansonsten wird das sichtbare Fenster abfotografiert.
- **Sortierung anhalten bei einem Änderungswert von ... oder weniger:** Ist diese Funktion aktiviert, so stoppt die Sortierung automatisch, wenn die Änderung der Federlängen den eingestellten Wert erreicht oder unterschreitet.
- **Heuristiktyp**: Gibt an, wie der Heuristikwert bei zusammengefassten Seiten berechnet werden soll (Durchschnitt, Maximum, Minimum)
- **Sortierergebnis jede ... Iteration anzeigen und dann anhalten:** Aktualisiert den Bildschirm nach jeder x-ten Iteration und hält dann ggf. an. Diese Funktion ist nützlich, da bei Graphen mit sehr vielen Knoten schon die Darstellung auf dem Bildschirm lange dauert. Da meistens die Zwischenschritte nicht von Bedeutung sind, kann

Die Einstellungen werden beim Betätigen der OK-Schaltfläche gespeichert und beim nächsten Programmstart wieder geladen.

Dateieigenschaften: Dateispezifische Einstellungen zur Berechnung der neuen Knotenpositionen:

- **Natürliche Länge der Federn:** Dieser Wert ist die Länge der Federn, die diese im ungedehnten Zustand haben.

- **Steifheit der Federn:** Faktor, der angibt, wie schnell die Federn zu ihrer natürlichen Länge streben.
- **Stärke der elektrischen Felder:** Dieser Wert gibt an, wie stark sich die Knoten gegenseitig abstoßen.
- **Durchschnittsberechnung an/aus:** Gibt an, ob die berechnete Kraft anschließend durch die Anzahl der verbundenen Nachbarn geteilt wird. Diese Option empfiehlt sich bei Graphen mit sehr vielen Knoten.

Ansicht

Hinein und hinaus zoomen: Hiermit kann in den Graphen hinein bzw. aus dem Graphen heraus gezoomt werden.

Standardzoom: Der Zoom wird wieder auf 1/1 gestellt.

Info: Zeigt folgende allgemeine Informationen an:
- Zoomfaktor (1/1 = Normalgröße 2/1= zweifache Vergrößerung usw.)
- Bildschirmposition auf dem virtuellen Schirm (Bildschirmmittelpunkt)
- Anzahl der Webpages (URLs)
- Anzahl der Websites (Knoten)
- Anzahl der Links (gesamt)
- Durchschnittsgrad des Graphen

Außerdem folgende Informationen zum aktuellen Punkt:
- Zusammengefasste URLs
- Heuristik
- Punktposition
- Anzahl eingehender Links
- Anzahl ausgehender Links

Auf Zusammenhang testen: Testet, ob der Graph zusammenhängend ist.

Statusleiste: Statusleiste ein- oder ausblenden.

Symbolleiste 2: Die zweite Symbolleiste kann mit dieser Funktion ein- oder ausgeblendet werden.

Gitterlinien: Die Gitterlinien können über diesen Menüpunkt ein- und ausgeschaltet werden. Der Gitterabstand entspricht dem im Einstellungsdialog gesetzten Wert.

Alle Filter ausschalten: Alle Knoten werden wieder angezeigt

Alle Markierungen entfernen: Alle Markierungen werden entfernt, so dass keinem Knoten mehr eine Markierung zugeordnet ist.

1er Bündel zeigen /
2er Bündel zeigen /
3er Bündel zeigen: Alle Knoten, außer den zu den Bündeln gehörenden, werden ausgeblendet.

Suchen...: Hier kann nach einer URL (oder einem Teil der URL) gesucht werden. Der gefundene Knoten wird zu aktuellen Knoten.

Weitersuchen: Sucht den nächsten Knoten, der zum unter „Suchen..." eingestellten Suchbegriff passt. Der gefundene Knoten wird zum aktuellen Knoten.

Position

Zentrieren (Ursprung): Der Ausschnitt des virtuellen Schirms wird auf den Nullpunkt (Ursprung) gesetzt.

Zentrieren (Graphmittelpunkt): Der Mittelpunkt des Graphen wird ermittelt und der Ausschnitt des virtuellen Schirms wird auf diese Position gesetzt.

Zentrieren (aktueller Knoten): Der aktuelle Knoten wird zentriert, d. h. der Ausschnitt des virtuellen Schirms wird auf diese Position gesetzt.

Graph zum Ursprung schieben: Bei der Positionsberechnung kann es durch Rundungsfehler zu einer minimalen Bewegung des Graphen kommen. Damit der Graph bei sehr vielen Iterationsschritten nicht aus dem Definitionsbereich „rutscht“, kann der Graph zum Ursprung verschoben werden.

B.2.3 Virtueller Schirm

Der Bildschirm zeigt immer einen Ausschnitt eines virtuellen Schirms. Der Wertebereich des virtuellen Schirms reicht auf der x- und y-Achse jeweils von rund -2 Mrd. bis 2 Mrd. Die Größe des dargestellten Ausschnitts hängt von der Zoomstufe, der aktuellen Bildschirmauflösung und der Fenstergröße ab.
Klickt man mit der linken Maustaste auf eine leere Stelle auf dem Bildschirm, so wird diese Position zentriert.
Durch Klicken mit der linken Maustaste auf einen Knoten wird dieser ausgewählt und zentriert, d. h. der Ausschnitt des virtuellen Schirms wird auf diese Position gesetzt. Den ausgewählten Knoten erkennt man an dem kleinen weißen Kreis in der Mitte des Knotens.

B.2.4 Knoten manuell einfügen

Falls diese Funktion im Einstellungsdialog eingeschaltet ist, lassen sich Knoten und Links manuell hinzufügen.
Durch Klicken mit der rechten Maustaste auf den leeren Bildschirm kann ein Knoten erzeugt werden. Durch Klicken mit der rechten Maustaste auf einen Knoten und gleichzeitiges Ziehen der Maus kann ein Link erzeugt werden.

B.2.5 Kontextmenü

Wird ein Knoten mit rechts angeklickt, ohne die Maus zu bewegen, so erscheint ein Kontextmenü mit einigen Informationen (Heuristikwert, Gesamknotenanzahl usw.) und den URLs, die diesem Knoten zugeordnet sind. Wird ein Eintrag des Kontextmenüs ausgewählt, so wird der Browser mit dieser URL aufgerufen.

B.2.6 Filter (Markierungswert)

Jeder Knoten hat als Eigenschaft einen Markierungswert. Anhand des Wertes können bestimmte Knoten ausgeblendet werden. Die Werte der Knoten, die angezeigt, also NICHT gefiltert werden, lassen sich im Einstellungsdialog festlegen.
Folgende Funktionen verändern die Markierungswerte der Knoten:

- **Auf Zusammenhang testen** und
  Unzusammenhängenden Komponenten löschen: Jede Komponente bekommt einen unterschiedlichen Wert. Knoten einer Komponente haben also auch gleiche Werte. Somit kann man sich einzelne Komponenten anzeigen lassen,
- **Bündel zeigen:** Jedes Bündel bekommt einen unterschiedlichen Wert. Knoten eines Bündels haben also auch gleiche Werte. Somit kann man sich einzelne Bündel anzeigen lassen.
  Diese Funktion verändert gleichzeitig auch die Einstellung „Markierung anzeigen von... bis...“,
- **Untermenge laden... und Untermenge hinzufügen (siehe oben)**
- und natürlich **alle Markierungen entfernen.**

Anhang C: Quellenverzeichnis

Quellen zum Thema Graphentheorie und Mathematik

[AlgGraph] Algorithmische Graphentheorie, Volker Turau, Addison-Wesley 1996

[Draw] Drawing Graphs - Methods and Models, Michael Kaufmann, Dorothea Wagner, Springer 2001

[Graph] Graphentheorie, Reinhard Diestel, Springer 1996

[GraphDraw] Graph Drawing - Algorithms for the visualization of graphs, Giuseppe Di Battista, Peter Eades, Roberto Tamassia, Ioannis G. Tollis, Prentice-Hall, Inc. 1999

[Math] Taschenbuch der Mathematik, I. N. Bronstein, K. A. Semendjajew, G. Musiol, H. Mühlig, Verlag Harri Deutsch 1997

[Repetit] Repetitorium der höheren Mathematik, Gerhard Merziger, Thomas Wirth, Binomi Verlag, 3. Auflage, o. J.

[Thulas] Graphs - Theory and algorithms, K. Thulasiraman, M. N. S. Swamy, John Wiley and son, Inc. 1992

Quellen zum Thema Malware

[AGN] http://agn-www.informatik.uni-hamburg.de/
Website des Fachbereichs AGN

[Bontchev98] Methodology of Computer Anti-Virus Research, Vesselin Vladimirov Bontchev, Doktorarbeit, 1998

[MWC-00] Webbasiertes Auffinden maliziöser Software mit fortschrittlichen heuristischen Verfahren, Sönke Freitag, Diplomarbeit, 2000

[Ticak] Charakterisierung und Kategoriesierung von Malware zur Qualitätssicherung beim Test von Anti-Malware-Software, Mario Ti□ak, Diplomarbeit, 1999

Quellen zum Thema Visual C++ Programmierung

[AlgC92] Algorithmen in C, Robert Sedgewick, Addison-Wesley 1992

[AlgDat02] Algorithmen und Datenstrukturen, Thomas Ottman, Peter Widmayer, Spektrum, Akademischer Verlag 2002

[CodeGuru] http://www.codeguru.com/
Website mit Informationen und Quellcode zur Visual C++ Programmierung.

[Cpp] Die C++ Programmiersprache, Bjarne Stroustrup, Addision-Wesley 1992

[Einf96] Einführung in Visual C++ 4.x, Frank Heimann, Guido Krüger, Nino Turianskyj, Addison-Wesley 1996

[Essential] Mike and Phani's Essential C++ Techniques, Michael Hyman, Phani Vaddadi, Apress 1999

[Inside] Inside Visual C++ 6.0, David Kruglinski, George Sheperd, Scot Wingo, Microsoft Press 1998, 5th edition

[Jpg] http://www.smalleranimals.com
Bibliothek zum Lesen und Schreiben von JPGs und BMPs

[Pract93] Practical C, Steve Oualline, O'Reilly 1993

Quellen zum Thema Visualisierung

[Atlas] Atlas of Cyberspace, Martin Dodge und Rob Kitchin, Addison- Wesley 2001

[Bertin82] Graphische Darstellungen, Jacques Bertin, de Gruyter, 1982

[Cyber] http://www.cybergeographiy.org
Online Cyberspace-Atlas

[IP2LatLong] http://cello.cs.uiuc.edu/cgi-bin/slamm/ip2ll/
Übersetzung von IP-Adressen in Geografische Koordinaten

[Lehnert] Generierung von Cone Trees mit VRML, Glen Lehnert, Studienarbeit, 1998

[Lumeta] http://www.lumeta.com
Projekt zur Visualisierung von Internetverbindungen

[Mackinlay86] Automatic Design of Graphical Presentations, Jock D. Mackinlay, Dissertation, 1986

[Mackinlay91] http://www.ifs.tuwien.ac.at/ifs/lehre/lva_ss2002/se188095/wall.html
Perspective Walls, Mackinlay et al. 1991

[MAP] http://maps.map.net/
Grafische Darstellung von Webseiten anhand ihrer inhaltlichen und lokalen Relevanz

[SchuMü00] Heidrun Schumann und Wolfgang Müller, Visualisierung – Grundlagen und allgemeine Methoden, Springer Verlag 2000

Übergreifende Quellen

[Alg] Algorithmen, Robert Sedgewick, Addison-Wesley 1992

[Google] www.google.net
WWW-Suchmaschine

[Graphlet99] http://www.infosun.fmi.uni-passau.de/Graphlet/
Programm zum Bearbeiten von Graphen

[IETF] http://www.ietf.org/
The Internet Engineering Task Force

[MovieGear] http://www.gamani.com/
Grafikprogramm zum erstellen von GIF-Animationen

[SelfHTML] http://www.netzwelt.com/selfhtml/
Einführung in die Erstellung und den Aufbau von HTML-Dateien

[Soller01] Methoden und Verfahren zur Optimierung der Analyse von Netzstrukturen am Beispiel des AGN-Malware Crawlers, René Soller, Studienarbeit, 2001

[Spertus98] http://www.mills.edu/ACAD_INFO/MCS/SPERTUS/Thesis/thesis.html
ParaSite: Mining the Structural Information on the World-Wide Web

[VIS] Visualisierung – Prolog und Einführung
www.agc.fhg.de/uniGoethe/lehre/ws0102/unterlagen/visualis00einfuehrung.pdf

[Webspace] http://www.webspace-free.com/
Liste mit Anbietern von Webspace

[w3c] http://www.w3.org/History.html
World Wide Web Consortium - A Little History of the World Wide Web

Anhang D - Eidesstattliche Erklärung

Ich versichere, die vorliegende Diplomarbeit selbständig und nur unter Benutzung der angegebenen Hilfsmittel angefertigt zu haben.

Hamburg, den

_____

René Soller