Ein Multiagenten-basiertes Peer-To-Peer Netzwerk zur verteilten, effizienten Spam-Filterung

Diplomarbeit

von
Jörg Metzger

nach einem Thema von Prof. Dr. Jörg H. Siekmann
im Fachbereich 6.2, Informatik, der Universität des Saarlandes

18. März 2003

Eidesstattliche Erklärung

Hiermit erkläre ich an Eides statt, dass ich diese Arbeit selbstständig verfasst, nur die im Literaturverzeichnis zitierten Quellen benutzt und sie noch keinem anderen Prüfungsamt vorgelegt habe.

Saarbrücken, im März 2003 Jörg Metzger

Danksagung

An dieser Stelle möchte ich mich bei Herrn Prof. Siekmann für die Vergabe des interessanten Themas bedanken und der Möglichkeit, diese Arbeit an seinem Lehrstuhl durchführen zu können.
Im ganz besonderen Maße möchte ich mich bei Michael Schillo für die hervorragende Betreuung und Unterstützung während der Anfertigung dieser Arbeit bedanken.
Des weiteren danke ich meiner Freundin Andrea Lauff für die moralische Unterstützung.
Meinen Freunden Tore Knabe und Sven Jacobi danke ich für die konstruktive Kritik.

Zusammenfassung

Der Versand von Email stellt eine der schnellsten Kommunikationsformen der heutigen Zeit dar. Mit der steigenden Zahl von Internetnutzern verbreitet sich jedoch auch eine negative Form des Emailversands, die mehr und mehr Benutzer von Emailaccounts betrifft: Spamming. Beim Spamming wird der elektronische Briefkasten durch unaufgeforderte Werbemails und Emails mit dubiosem Inhalt verstopft. Das Aussortieren dieses Spam kostet den Benutzer Zeit und Geld.

In dieser Diplomarbeit stellen wir die Implementierung eines verteilten Spamfilters vor, der seine Effektivität durch die Kombination eines Textklassifizierungsalgorithmus mit der Multiagentenplattform FIPA-OS erreicht. Email wird durch Antispam-Agenten mit Hilfe des Klassifizierungsalgorithmus „*support vector machines*“ untersucht. Die Agenten tauschen Informationen über Spam in Form von Hashwerten über ein P2P-Netzwerk *(engl. peer-to-peer)* untereinander aus und verbessern so die Qualität ihrer Analyse der Emails. Hashwerte von Emails, die die Antispam-Agenten als Spam identifizieren, werden generiert und über das Netzwerk an alle anderen Agenten verschickt. Anhand der versendeten Hashwerte können die Agenten Spammails erkennen, die sie vom Mailserver abholen. Auf diese Weise wird die Qualität des verwendeten Filters kontinuierlich gesteigert.

Weiterhin optimieren wir den Informationsversand innerhalb des Netzwerkes mit Hilfe von Organisationsformen aus dem Gebiet der Sozionik und sparen somit Kommunikationskosten. Agenten, die regelmäßig gleiche Spammails erhalten, finden sich selbstständig zu Gruppen zusammen und tauschen untereinander verstärkt Informationen aus. Dies realisieren wir unter der Prämisse, dass die Agenten gar nicht wissen, welche anderen Agenten die gleichen Spammails bekommen. Wir zeigen in der experimentellen Evaluation, dass ein effizienter Informationsaustausch innerhalb der Gruppen von Agenten stattfindet, die sich durch Selbstorganisation bilden. Aufgrund des Austausches von Spaminformationen erhöhen die Antispam-Agenten ihre Genauigkeit bei der Klassifizierung von Spam.

Inhaltsverzeichnis

Abbildungsverzeichnis

Kapitel 1

Einleitung

Wir schlagen in dieser Diplomarbeit eine Brücke zwischen zwei bedeutenden Forschungsgebieten der Künstlichen Intelligenz. Wir stellen im Verlauf der Arbeit einen Algorithmus zur Textklassifizierung aus dem Bereich des Information Retrieval (IR) vor und integrieren diesen erfolgreich in ein Multiagentensystem (MAS). Das entstandene System werden wir erweitern durch Anwendung der Begriffe Vertrauen und Selbstorganisation aus einem weiteren Forschungsgebiet, der Sozionik. Das Forschungsfeld Sozionik entstand aus der Zusammenarbeit der Bereiche Soziologie und Verteilter Künstlicher Intelligenz [MALSCH 2001]. Die folgenden beiden Abschnitte geben einen Überblick über die Zielsetzung dieser Arbeit und stellen die wichtigsten Ergebnisse kurz zusammen. Die Gliederung der Diplomarbeit folgt im dritten Abschnitt.

1.1 Problemstellung und Motivation

Spam überflutet das Internet mit vielen Kopien derselben Nachricht. Die Adressaten der Nachricht haben keine Wahl, sie müssen diese empfangen. Wir befassen uns in dieser Arbeit mit der Bekämpfung dieser ungewollten Nachrichten, kurz *Spam* genannt. Spam in Form von Emailnachrichten wird direkt an den Account der individuellen Benutzer gesendet. Da der Transport der Spamnachrichten über das Internet nicht wirkungsvoll unterbunden werden kann, bleibt dem Benutzer nur die Möglichkeit, Spam von seinem Emailaccount selbst zu entfernen. Dabei kann er die Hilfe von speziellen Softwareprogrammen in Anspruch nehmen, die auf das Erkennen und Filtern von Spam spezialisiert sind. Diese sogenannten Spamfilter analysieren die empfangenen Emails mit Hilfe unterschiedlicher Methoden wie Filterregeln, Absenderlisten oder Algorithmen zur Textklassifizierung. Jedoch arbeiten diese Filter nicht zusammen, d.h. die Spamfilter der verschiedenen Benutzer können das bei der Filterung der Emails erworbene Wissen nicht austauschen. Daher wollen wir ein Netzwerk von über dem Internet verteilten Agenten entwickeln, in dem Spamfilter Daten über Spammails sicher und effizient tauschen und so ihr Wissen vergrößern, ohne Informationen über andere Emails ihres Benutzers preiszugeben. Zur Verwirklichung dieses Netzwerkes bietet

sich der Einsatz von Multiagentensystemen an. Der Spamfilter der Benutzer soll dabei durch einem Agenten repräsentiert werden. Während der Analyse der Emails kann der Agent Spaminformationen generieren, die über ein P2P-Netzwerk an andere Agenten verteilt werden. Diese erweitern ihr Wissen anhand der transferierten Informationen und verbessern ihre Genauigkeit bei der Identifizierung von Spam. Da P2P-Netzwerke schnell Größenordnungen von mehreren tausend Knoten erreichen, muss zur Reduktion von Kosten die Kommunikation innerhalb der Multiagentenplattform gesteuert und gelenkt werden. Wir kombinieren im Verlauf der Arbeit Erkenntnisse und Methoden aus den Gebieten Information Retrieval, Multiagentensysteme und Sozionik und implementieren ein verteiltes P2P-Netzwerk aus Agenten zur effizienten Erkennung von Spam. Agenten, die die gleichen Spammails erhalten, finden sich auf der Basis von Vertrauen durch Selbstorganisation zu Gruppen zusammen und tauschen verstärkt untereinander Spaminformationen aus.

1.2 Ergebnisse

Das im Verlauf dieser Arbeit entworfene und implementierte Spamfilternetzwerk zeigt, dass durch die Einbettung eines geeigneten Textklassifizierungsalgorithmus in ein Multiagentensystem hervorragende Ergebnisse bei der Erkennung von Spam erreicht werden. Wir belegen experimentell, dass die Anzahl von Spammails, die die Agenten richtig klassifizieren können, durch Zusammenschluss in einem P2P-Netzwerk gesteigert wird. Die einzelnen Antispam-Agenten können eine Teilmenge der Spammails anhand der Informationen erkennen, die sie untereinander über die Agentenplattform austauschen. Die Erkennungsrate wird durch den Einsatz von Algorithmen zur Textklassifizierung weiter gesteigert. Der Benutzer kann die einzelnen Schritte des Klassifizierungsprozesses anhand der Benutzerschnittstelle des Antispam-Agenten mitverfolgen und die Klassifizierungsentscheidung des Agenten nach seinen Wünschen abändern. Hat ein einziger Antispam-Agent eine bestimmte Spammail erkannt und im Spamfilternetzwerk verteilt, so können alle anderen Agenten des Netzwerkes die gleiche Spammail anhand der vom Agenten bereitgestellten Informationen sicher erkennen. Damit sind sie gegenüber isolierten Spamfiltern klar im Vorteil, ihre Klassifizierungsgenauigkeit verbessert sich. Böswillige Agenten können das Spamfilternetzwerk nicht angreifen, da nur Informationen über Spam ausgetauscht werden. Weiterhin zeigen wir, dass sich der Kommunikationsaufwand innerhalb des Netzwerkes durch Selbstorganisation der Agenten beträchtlich senken lässt, ohne dass Effektivität eingebüßt wird. Wir erreichen den Zusammenschluss benevolenter Antispam-Agenten, die die gleichen Spammails erhalten, ohne dass ihnen bewusst ist, dass sie die Opfer des gleichen Spammers sind. Nach Zusammenschluss tauschen diese Agenten verstärkt Informationen über das P2P-Netzwerk aus. Der Zusammenschluss und Informationsaustausch von benevolenten und böswilligen Agenten wird durch Implementierung von Vertrauen zwischen den Agenten verhindert.

Wir haben zwei Publikationen auf Konferenzen eingereicht, die auf den Ergebnissen und Schlussfolgerungen dieser Diplomarbeit fußen. Im ersten Papier [METZGER et al. In Print], welches auf der *International/Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003)* eingereicht und akzeptiert wurde, gehen wir auf den Aufbau des Spamfilternetzwerkes auf der Basis eines Multiagentensystems ein und erläutern den Austausch von Spaminformationen zwischen den Antispam-Agenten. Eine weitere Publikation [METZGER et al. submitted'03] über die Optimierung des Versands von Spaminformationen durch Selbstorganisation haben wir bei der *International Conference on Parallel and Distributed Computing (Euro-Par 2003)* eingereicht. In diesem Papier weisen wir experimentell die Erhöhung der Klassifizierungsgenauigkeit der Antispam-Agenten durch Selbstorganisation auf der Basis von Vertrauen nach.

1.3 Gliederung der Arbeit

Im folgenden Kapitel geben wir einen Überblick über die relevante Literatur sowie die theoretischen Grundlagen, die wir in dieser Diplomarbeit benötigen. Da wir Algorithmen zur Textklassifizierung in ein Multiagentensystem integrieren, schaffen wir die Grundlagen in beiden Forschungsgebieten, bevor wir uns mit dem Phänomen „Spam" näher befassen. Das dritte Kapitel wird eine detailliertere Problembeschreibung liefern. Wir werden auf die Schwierigkeiten eingehen, die durch den massiven Versand von Spam entstehen. Des weiteren präzisieren und erläutern wir die Ziele dieser Diplomarbeit. Ein Spamfilternetzwerk zur Lösung des Spamproblems werden wir im vierten Kapitel spezifizieren und dessen Implementierung vorstellen. Ebenfalls stellen wir die Funktionsweise des Antispam-Agenten vor. Der Antispam-Agent ist die Hauptkomponente des Spamfilternetzwerkes und wir werden die Abfolge seiner Arbeitsschritte bei der Erkennung von Spam beschreiben. Im fünften Kapitel realisieren wir eine Experimentalumgebung, in der der Zusammenschluss von Agenten auf der Basis von Selbstorganisation möglich ist. Wir werden die Kommunikation zwischen den Agenten optimieren. Ferner wird es möglich sein, die Effizienz des Versands von Spaminformationen innerhalb des Spamfilternetzwerkes mit Hilfe der Experimentalumgebung zu evaluieren. Die Leistungsfähigkeit des implementierten Systems belegen wir im sechsten Kapitel und weisen bestimmte Eigenschaften empirisch nach. Die Diplomarbeit wird durch das siebte Kapitel mit einer Zusammenfassung der Resultate abgeschlossen. Darin geben wir auch einen Ausblick auf zukünftige Arbeiten, die die Effektivität unseres Spamfilternetzwerkes weiter erhöhen können. Die Konfigurationen der Experimente sowie eine Anleitung zur Bedienung des Spamfilters und der Experimentalumgebung befinden sich im Anhang.

Kapitel 2

Theoretische Grundlagen

Diese Diplomarbeit hat zum Ziel, ein verteiltes Netzwerk von Agenten zur Filterung von Spammails zu implementieren. Wir wollen die Eigenschaften von *Multiagentensystemen* mit den Vorteilen der Algorithmen zur *Textklassifizierung* kombinieren. Aus diesem Anspruch heraus werden wir uns in diesem Kapitel mit den Grundlagen in zwei Gebieten der KI befassen. Zuerst beschreiben wir im ersten Kapitel die wichtigsten Konzepte aus dem Bereich der Multiagentensysteme. Ferner befassen wir uns mit der Multiagentenplattform FIPA-OS, in die die miteinander interagierenden Agenten zur Spamfilterung eingebettet werden sollen. Da diese Agenten ihre Genauigkeit bei der Erkennung von Spam mit Hilfe von Textklassifizierung weiter erhöhen sollen, stellen wir im zweiten Kapitel verschiedene Algorithmen vor, mit denen es möglich ist, den Inhalt von Emails zu analysieren und zu klassifizieren. Wir vergleichen ihre Funktionsweise bei der Klassifizierung von Spam und wählen den Algorithmus mit der höchsten Erkennungsrate von Spam aus. Schließlich werden wir im dritten Kapitel das Phänomen „Spam“ näher erläutern, sowie bereits entwickelte Möglichkeiten zur Erkennung von Spam beschreiben. Zum Abschluss stellen wir die wichtigsten Programme mit ihren Vor- und Nachteilen vor, die bereits zur Spamfilterung eingesetzt werden.

2.1 Verteilte Künstliche Intelligenz

Der Forschungsbereich der Verteilten Künstlichen Intelligenz VKI *(engl. distributed artificial intelligence)*, welcher Ende der Siebziger Jahren begründet wurde, ist ein Teilbereich der Forschungsrichtung der Künstlichen Intelligenz KI *(engl. artificial intelligence)*. Innerhalb der VKI beschäftigt man sich mit dem Studium, der Konstruktion und der Anwendung von *Multiagentensystemen* (MAS). Dabei wird davon ausgegangen, dass komplexe und verteilte Probleme nicht effizient durch einen einzigen Problemlöser gelöst werden können. Multiagentensysteme hingegen können in die verteilten, offenen und heterogenen Computerplattformen und Informationssysteme der heutigen Zeit wie dem Internet besser eingebunden werden [WEISS 1999]. Ferner

können sie eine wichtige Rolle bei der Entwicklung und Analyse von Modellen und Theorien der menschlichen Gesellschaft spielen. In Multiagentensystemen kooperieren bzw. konkurrieren mehrere Agenten miteinander. Deshalb werden wir im ersten Abschnitt zuerst den Begriff des Agenten näher beleuchten. Des weiteren werden wir die wichtigsten Arten von Agenten kurz vorstellen. Im zweiten Abschnitt befassen wir uns mit dem Aufbau der aus verteilten Agenten bestehenden Multiagentensysteme. Die FIPA-OS Plattform, die den Agenten verschiedene Funktionalitäten zur Kommunikation bereitstellt, wird anschließend im dritten Abschnitt des Kapitels behandelt.

2.1.1 Softwareagenten

Definition von Agenten

In der Literatur existiert eine Vielzahl von Definitionen für Agenten. Eine Zusammenfassung verschiedener Definitionen des Begriffes „Agent“ findet sich bei FRANKLIN und GRAESSER [1996]. Die von verschiedenen Autoren verlangten Agenteneigenschaften stimmen größtenteils überein und ergänzen einander. Eine von vielen Autoren akzeptierte Definition stammt von RUSSEL und NORVIG [1996]:

Definition 1 (Agent nach Russel und Norvig, 1996)
Ein Agent ist eine Einheit, von der man sagen kann, dass sie ihre Umwelt über Sensoren wahrnimmt und sie mit Hilfe von Effektoren beeinflusst.

Diese Definition von Agenten ist sehr weit gefasst und reflektiert den großen Anwendungsbereich des Begriffes Agent. Die Eigenschaften von Agenten werden durch weitere Autoren ergänzt. Agenten verfolgen ihr Ziel „flexibel und rational“ und führen ihre Aktionen so aus, dass ein „vorgegebenes Performanzmaß“ erreicht wird [MAES 1995]. SMITH et al. [1994] bezeichnen einen Agenten als „persistente Softwareeinheit, die einem bestimmten Zweck gewidmet ist“. Dabei besitzen Agenten die Fähigkeit zur Problemlösung [HAYES-ROTH 1995].

Die Definitionen von WOOLDRIDGE und JENNINGS [1995], WEISS [1996] ordnen dem Agenten folgende Attribute zu:

- *autonom*
  Agenten arbeiten selbstständig ohne Eingreifen von außen.

- *reaktiv*
  Intelligente Agenten können ihre Umwelt mit Hilfe von Sensoren analysieren und auf die Dynamik des Gesamtsystems in geeigneter Weise reagieren.

- *proaktiv*
  Agenten ergreifen die Initiative, sie arbeiten auf ihre eigenen Ziele hin.

- *sozial*
  Agenten können mit anderen Agenten interagieren und kooperieren, sie erkennen Konflikte und können sie lösen.

WOOLDRIDGE und JENNINGS [1995] bezeichnen diese Fähigkeiten auch als *schwache Agenteneigenschaften*. Weiß [1999] fügt diesen Eigenschaften noch *Rationalität*, *Mobilität*, *Introspektion*, *Wahrhaftigkeit* und *Benevolenz* hinzu. Agenten verfügen über ein Aktionsrepertoire, d.h. sie beeinflussen mit Aktionen ihre Umwelt. Welche Wirkung die Handlungen eines Agenten haben, hängt entscheidend von der Umgebung ab, in der er eingebettet ist. RUSSEL und NORWIG [1996] unterscheiden folgende Eigenschaften der Umgebung:

- *zugänglich vs. unzugänglich*
  In einer zugänglichen Umgebung erhält der Agent vollständige und aktuelle Informationen über den Zustand seiner Umgebung. Dies ist in der realen Welt meist nicht der Fall. Je zugänglicher eine Umgebung ist, desto leichter ist die Programmierung des darin operierenden Agenten.

- *deterministisch vs. indeterministisch*
  In einer deterministischen Welt hat jede ausgeführte Aktion garantiert einen bestimmten Effekt. Dagegen sind die Auswirkungen von Aktionen in einer indeterministischen Welt nicht immer genau voraussagbar.

- *episodisch vs. nicht-episodisch*
  In einer episodischen Umgebung finden die Handlungen des Agenten in einem abgeschlossenen Zeitraum statt. Der Agent muss sich keine Gedanken über die Auswirkungen seiner Aktionen für die Zukunft machen, d.h. es existiert keine Beziehung zwischen der Performanz des Agenten innerhalb verschiedener Szenarien.

- *statisch vs. dynamisch*
  Eine statische Umgebung bleibt, von den Aktionen des Agenten abgesehen, gleich und ändert nicht ihren Zustand. In der realen Welt finden meist viele andere Prozesse neben den Handlungen des Agenten statt, die den Zustand der Umgebung andauernd verändern.

- *diskret vs. kontinuierlich*
  In einer diskreten Umgebung gibt es eine festgelegte Anzahl von Aktionen und Wahrnehmungen (z.B. Schachspiel), wobei in einer kontinuierlichen Umgebung beliebige Komplexität erreicht werden kann.

Die Wissensbasis des Agenten lässt sich auf verschiedene Weisen aufbauen [KAZAKOV und KUDENKO 2001]. So kann ein externer Lehrer Beispiele für Aktionssequenzen vorgeben und dieses klassifizieren. Dem Agenten wird mitgeteilt, in welchen

Situationen bestimmte Aktionen optimal geeignet sind *(engl. fully supervised learning)*. Führt ein Agent eine Aktion aus, so kann er die von ihm verursachten Änderungen in seiner Umwelt mit Sensoren wahrnehmen und auf den Nutzen der Aktion hin überprüfen und einordnen. Das Trainingsmodell wird durch Lernen aus dem Feedback der Umwelt erzeugt *(engl. reinforcement learning)*. Die verschiedenen Methoden zum Aufbau der Wissensbasis des Agenten können auch kombiniert werden. So kann ein neu erzeugter Agent anhand von Trainingsdaten seine Wissensbasis aufbauen und danach selbstständig sein Wissen erweitern. Als dritter Lernansatz existiert für den Agenten auch die Möglichkeit, interessante „Muster“ seiner Umwelt aufzunehmen und daraus Konzepte über seine Umgebung zu erstellen. Diese Konzepte sollen dem Agenten helfen, seine Ziele effizient und effektiv zu erreichen *(engl. unsupervised learning)*.

Nachdem wir auf die verschiedenen Definitionen des Agenten eingegangen sind, wollen wir im Folgenden drei Architekturen von Agenten vorstellen. Dabei unterscheiden sich die Agenten durch die Art und Weise, wie sie ihre Entscheidungsfindungen durchführen:

Reaktive Agenten

Agenten, die ihre Entscheidungen nur anhand des Zustandes der aktuellen Umgebung treffen, bezeichnet man als *reaktive Agenten*. Dabei spielen Vorgänge in der Vergangenheit keine Rolle bei der Entscheidungsfindung. Die zu Grunde liegende Architektur heißt *Subsumptionsarchitektur* [BROOKS 1986]. Darin werden die Zustände direkt den Verhaltensregeln zugeordnet, wobei auch mehrere Regeln gleichzeitig „feuern“ können. Die Verhaltensregeln sind in *Schichten* angeordnet. Regeln in niedrigeren Schichten können Regeln in höheren Schichten hemmen, sie haben eine *größere Priorität*.

Weitere reaktive Architekturen finden sich bei MAES [1990] und Travers [1988]. Ein Agent wird als Menge unterschiedlicher Aufgaben dargestellt, wobei nur jeweils eine einzige gleichzeitig bearbeitet werden kann. Verschiedene Einheiten versuchen, die Kontrolle über das Verhalten des Agenten zu gewinnen und ihre entsprechende Aufgabe zu erledigen. Reaktive Agenten enthalten kaum eine explizite symbolische Repräsentation ihrer Umwelt und müssen diese nicht ständig aktualisieren. Deshalb besteht die Gefahr, dass reaktive Agenten ihr Gesamtziel aus den Augen verlieren und stattdessen nur kurzfristige Ziele verfolgen.

Deliberative Agenten

Deliberative Agenten bauen aus ihrem sensorischen Input ein Modell der Welt, in der sie situiert sind. Innerhalb dieses Modells kreieren sie einen Plan, um schnellstmöglich ihre Ziele zu erreichen *(engl. reasoning)*. Die bekannteste deliberative Agentenstruktur ist die *BDI-Architektur*. Die zu Grunde liegende Idee der Belief-Desire-Intention (BDI) Architektur von BRATMAN [1987] liegt darin, zu entscheiden, welche Ziele

man erreichen will (*Überlegung*) und auf welche Art und Weise man sie erreichen kann (*means-end Schlussfolgerungen*). Die Fakten *(engl. beliefs, Bel)* repräsentieren die aktuelle Wissensbasis des Agenten, seine Wünsche *(engl. desires, Des)* die möglichen Erwartungs- und Vorgehensweisen. Die langfristigen Ziele des Agenten werden in den Absichten *(engl. intentions, Int)* beschrieben. Hat der Agent sich auf bestimmte Absichten festgelegt, so werden diese beibehalten, bis sie erfolgreich erreicht wurden oder sie nicht mehr erreicht werden können, bzw. nicht mehr existieren. Die Absichten sind also beständig. Die weiteren Überlegungen des BDI-Agenten orientieren sich direkt an seinen Absichten. Der Zustand eines BDI-Agenten ist ein Tripel der Form (B, D, I) mit $B \subseteq Bel$, $D \subseteq Des$ und $I \subseteq Int$.

Die Entscheidungsfindung funktioniert folgendermaßen: Eine Menge von Fakten stellt die Menge der Informationen des Agenten über seine Umwelt dar. Eine Funktion nimmt die perzeptuellen Eindrücke und die aktuellen Fakten und erstellt daraus eine aktualisierte Menge von Fakten. Auf der Basis dieser Menge und der Absichten werden die Optionen, die zur Verfügung stehen, generiert. Da einige dieser Optionen selbst wieder zu Absichten werden können, findet hier eine Rückkopplung statt, die in konkreten Optionen resultieren kann. Eine Filterfunktion beschließt Absichten auf Grundlage der aktuellen Fakten, Wünsche und Absichten. Sind die Absichten festgelegt, so werden daraus die Aktionen des Agenten hergeleitet. Die BDI-Architektur findet sich beispielsweise bei JENNINGS [1993], ANASTASSAKIS et al. [2001] und in der RETSINA Architektur von DECKER et al. [1997].

Hybride Agenten

Agenten sollen eine Ausgewogenheit zwischen *zielgerichtetem* und *reaktivem* Verhalten aufweisen [WOOLDRIDGE und JENNINGS 1995]. Ein Problem bei der Entscheidungsfindung stellt die Tatsache dar, eine gute Balance zwischen reaktivem und deliberativem Verhalten zu finden. So können reaktive, verhaltensbasierte Agenten gut auf dynamische Umgebungen reagieren und sind einfach zu beschreiben. Sie schauen weder in die Vergangenheit, noch berücksichtigen sie Erwartungen über die Zukunft. Deshalb sind sie beschränkt und können komplexere Aufgaben nur schwer erfüllen. Deliberative Agenten, die über eine Wissensbasis verfügen, erfüllen zwar komplexere Aufgaben, finden sich aber in ständig wechselnden Umgebungen nicht zurecht. Eine Lösung dieses Problems bietet der Einsatz von *hybriden* Architekturen mit mehreren *parallel arbeitenden Kontrollebenen*. Die am DFKI entwickelte Agentenarchitektur InteRRaP [MÜLLER und PISCHEL 1993, MÜLLER 1997] ist ein Beispiel für eine deliberative Agentenstruktur. Jede Schicht ist mit einer eigenen *Wissensbasis* verbunden. Außerdem existiert nur zwischen der untersten Ebene, der reaktiven Schicht, eine Verbindung mit den Perzeptoren und Aktoren. Kann die reaktive Schicht mit Hilfe ihres symbolischen Weltmodells einen sensorischen Input verarbeiten, dann tut sie dies auch. Andernfalls wandert der Kontrollfluss nach der *Bottom-up-Methode* zur nächsthöheren Schicht, der *lokalen Planungsschicht*. Diese ist für den normalen, ziel-

gerichteten Planungsablauf zuständig und generiert Aktionen anhand ihres *Planungswissens* ohne Berücksichtigung von Interaktionen mit anderen Agenten. Die oberste der Schichten, die *kooperative Planungssschicht*, plant das Verhalten gegenüber anderen Agenten sowie die Koordination zwischen den Agenten (Soziales Wissen). Lokale und kooperative Planungsschicht sind die deliberativen Komponenten der InteRRaP-Architektur.

2.1.2 Multiagentensysteme

Softwareprogramme bestehen aus einer großen Zahl von Komponenten, die untereinander interagieren. Nach der Einführung in die verschiedenen Arten von Agenten können wir uns jetzt mit Systemen beschäftigen, die genau auf die Anforderungen heutiger verteilter Softwaresysteme zugeschnitten sind (Multiagentensysteme). Die Definition von Multiagentensystemen werden wir im ersten Abschnitt bereitstellen. Die verschiedenen Mechanismen wie Kooperation und Wettbewerb, die Agenten benutzen, um ihre eigenen Ziele zu erreichen, behandeln wir im zweiten Abschnitt. Die dabei angewendeten Kommunikationssprachen zum Austausch von Informationen zwischen den Agenten sind Gegenstand des dritten Kapitels. Im vierten Abschnitt fassen wir die wichtigen Eigenschaften von Multiagentensystemen zusammen.

Definition von Multiagentensystemen

Multiagentensysteme bestehen aus mehreren *autonomen Agenten*, die *unabhängig* und *verteilt kooperieren*. Dazu stehen definierte Schnittstellen zur Verfügung. Das Multiagentensystem stellt grundlegende zentrale Dienste zur Verfügung, in denen Kommunikations- und Interaktionsprotokolle spezifiziert sind. Multiagentensysteme erlauben die Dekomposition von Problemen in Teilprobleme, die von verschiedenen Agenten gleichzeitig gelöst werden. Kommunikation ist nicht zwingend notwendig zur Kooperation, da bei vollständig modellierter Umwelt die Koordination eine rein lokale Angelegenheit ist. Da ein Agent nicht seine gesamten Ressourcen zur Kommunikation aufwenden kann, muss er intern ein minimales Weltmodell unterhalten. Innerhalb von Multiagentensystemen existiert keine zentrale Steuer- und Koordinationseinheit. Des weiteren gibt es auch keine zentrale Datenbank. Die ersten Multiagentensysteme traten in der Mitte der Achtziger Jahre auf. So entwickelte DURFEE [1988] ein Multiagentensystem, in dem eine Zahl von verteilten Agenten den Straßenverkehr beobachteten und die vorbeifahrenden Fahrzeuge identifizierten. BOND und GASSER [1988] konkretisieren Multiagentensysteme mit folgender Definition:

Definition 2 (Multiagentensystem nach Bond und Gasser, 1988)
Multiagentensysteme beschäftigen sich mit der Koordination intelligenten Verhaltens einer Anzahl von autonomen, intelligenten Agenten: der Aufgabe, wie sie ihr Wissen, ihre Ziele, Fähigkeiten und Pläne vereint einsetzen können, um Probleme zu lösen.

Interaktion von Agenten

Agenten müssen ihre Aktionen innerhalb von Multiagentensystemen koordinieren, um ihre jeweiligen, bzw. die gemeinsamen Ziele zu erreichen. Es gibt zwei mögliche Szenarien von Multiagentensystemen, in denen Agenten ihr Verhalten koordinieren:

- *Kooperation*
  Die Agenten arbeiten zusammen, um ein *gemeinsames Ziel* zu erreichen. Dabei teilen sie ihr Wissen und ihre Möglichkeiten (*Teamwork*). Wenn das Problem nicht gelöst wird, scheitern alle daran arbeitenden Agenten zusammen. Bei der Zusammenarbeit können die verschiedenen Agenten *Koalitionen* bilden, die verschiedenen Koalisationsgruppen kooperieren dabei nicht miteinander. Weiterhin können Vermittlungsagenten eingesetzt werden. Tritt ein Agent dem Multiagentensystem bei, so meldet er sich mit seinen speziellen Fähigkeiten beim Vermittlungsagenten an. Dieser teilt diese Fähigkeiten den anderen Agenten des Multiagentensystems mit. So kann jeder Agent Teilprobleme an kompetente Agenten abgeben. Mit Hilfe von Vermittlungsagenten ist die korrekte An- und Abmeldung verschiedener Agenten im Multiagentensystem gewährleistet.

- *Wettbewerb*
  Die einzelnen Agenten haben unterschiedliche Zielsetzungen. Sie versuchen, ihren eigenen Gewinn auf Kosten der anderen Agenten zu maximieren [MÜLLEN und WELLMAN 1996, DAVIS und SMITH 1996]. Jeder Agent erhält eine Belohnung, wenn er sein eigenes Ziel erreicht. Agenten mit verschiedenen Zielsetzungen interagieren miteinander, indem sie Verhandlungen durchführen. Die Agenten stellen ihre verschiedenen Positionen dar und versuchen sich durch Zugeständnisse oder der Suche nach Alternativen zu einigen [WEISS 1999].

Zur Bewältigung komplexer Problemstellungen besteht die Möglichkeit, Agenten zu Organisationen zusammenzufassen [BOOCH 1994], wobei die Agenten bestimmte Rollen innerhalb der Organisation ausführen. Rollen werden von WOOLDRIDGE et al. [2000] anhand der vier Attribute Aufgaben, Rechte, Aktivitäten und Protokolle definiert. Einzelne Agenten schließen sich zu einer Gruppe zusammen, die wiederum eine Softwareeinheit bildet. Diese Gruppe wird von anderen Agenten oder Agentengruppen wie eine einzige Einheit betrachtet. Es bildet sich somit eine Hierarchie von Agenten, die sich den Veränderungen der Umwelt durch Zusammenschluss neuer und Auflösung bestehender Gruppen anpassen kann [JENNINGS 1999]. Nicht nur bestimmte Agenten, sondern ganze Teilsysteme können wiederverwertet werden. Auch holonische Agenten schließen sich zu einer Gruppe zusammen, um ein bestimmtes Problem zu lösen. Ein holonischer Agent ist rekursiv aus Unter-Agenten der gleichen Struktur aufgebaut [GERBER et al. 1999]. Holonische Agenten werden beispielsweise im TeleTruck Projekt [FISCHER et al. 1996] zur Planung von Transporten angewendet.

Kommunikation zwischen Agenten

In den beiden Szenarien Wettbewerb und Kooperation aus dem letzten Abschnitt beeinflussen sich die Agenten gegenseitig, sie interagieren miteinander. Dabei besitzt jeder Agent nur unvollständige Informationen und ist daher zur *Kommunikation* mit den anderen Agenten gezwungen. Die entstehenden Kosten für den Informationsaustausch mit einem erfahrenen Agenten sind normalerweise viel geringer als diejenigen, die anfallen, um das gleiche Wissen durch Erforschung der Umwelt oder Beobachtung anderer Agenten zu erhalten [KAZAKOV und KUDENKO 2001]. Voraussetzung für eine erfolgreiche Kommunikation zwischen Agenten ist sowohl eine gemeinsame Sprache (Syntax und Semantik) als auch eine gemeinsame *Ontologie*. Alle Agenten eines Systems belegen identische Begriffe mit der gleichen Bedeutung. Erreicht wird dies durch die Verwendung von Kommunikationssprachen wie KQML [FININ et al. 1997] und alternativ dem Standard der *Foundation for Intelligent Physical Agents* (FIPA) [FIPA 2000c]. Beide Kommunikationssprachen beruhen auf der Sprechakttheorie *(engl. speech act theory)* von SEARLE [1969]. Ein *Sprechakt* ist das Ausführen einer Handlung durch eine sprachliche Äußerung.

Eigenschaften

Multiagentensysteme zeichnen sich durch folgende Eigenschaften aus:

- *Dezentrale Verteilung*
  Das Multiagentensystem besteht aus einer Zahl von spezifischen modularen Komponenten, die bestimmte Problemaspekte lösen [SYCARA 1998]. Durch die Verteilung der Agenten ergeben sich mehrere Vorteile. So besteht nicht die Gefahr, dass eine zentrale Recheneinheit zum Flaschenhals im Gesamtsystem wird oder dass bei dessen Ausfall das gesamte System instabil wird. Komplexe Probleme werden in Teilprobleme aufgespaltet und von spezialisierten Agenten gelöst. Fällt ein Agent aus, so kann das Gesamtsystem weiterarbeiten, da die Kontrolle auf alle Agenten verteilt ist (*Robustheit*).

- *Effizienz*
  Informationen werden auf verschiedene Agenten verteilt. Diese können sie parallel und asynchron verarbeiten. Dadurch kann die Verarbeitungsgeschwindigkeit des Gesamtsystems gesteigert werden.

- *Flexibilität und Wiederverwendung*
  Zum Lösen reeller Probleme, wie sie in verteilten, offenen Systemen vorkommen [HEWITT 1986], werden Multiagentensysteme benötigt, die sich dynamisch an ihre Umgebung anpassen können. Wächst die Größe des Problems während dessen Bearbeitung durch das Multiagentensystem an, so können neue Agenten eingefügt werden (*Erweiterbarkeit*). Auch ist die Rekonfiguration der Agenten und der Einsatz in anderen Systemen möglich.

- *Kostenersparnis*
  Die Kosten für ein System von „einfachen“ Agenten sind geringer als diejenigen für ein zentrales System. Auch die Wiederverwertung einzelner Agenten oder Agentengruppen bietet einen finanziellen Vorteil gegenüber zentralisierten Systemen.

2.1.3 Agentenstandards - FIPA

Die *Foundation for Intelligent Physical Agents* (kurz FIPA) ist eine weltweite Organisation mit Sitz in Genf. Ihre Mitglieder aus Wissenschaft und Industrie stehen in ständigem Kontakt über Mailinglisten und Konferenzen. Die FIPA wurde 1996 gegründet, in einer Zeit, in der sich nach BURG [2002] etwa 60 verschiedene Agentensysteme im Wettbewerb befanden. Die verschiedenen Systeme waren inkompatibel und in sich abgeschlossen. Deshalb setzt sich die FIPA zum Ziel, *Softwarestandards* und *Spezifikationen* für heterogene und interagierende Agenten und Agentensysteme festzulegen. Diese Spezifikationen sind allgemein und theoretisch und beschrieben die Schnittstellen, die Agenten implementieren müssen, um auf FIPA-konformen Agentensystemen zu operieren. Die Spezifikationen werden durch Technische Komitees [DALE und MAMDANI 2001] erarbeitet und enthalten keine Vorschriften zur Implementierung. Um das von der FIPA verfolgte Ziel einer plattformübergreifenden Agentenkommunikation und Interaktion zu ermöglichen, sind bestimmte Agentenstandards nötig. Die Spezifikationen beschrieben diese Standards in den Gebieten Agentenkommunikation, Agentenmanagement und Agenten/Softwareintegration [O'BRIEN und NICOL 1988], die in den folgenden Abschnitten behandelt werden.

Agentenkommunikation

Um die Zusammenarbeit zwischen Agenten zu ermöglichen, ist eine mächtige Kommunikationssprache nötig. FIPA entwickelte die eigene Agentenkommunikationssprache FIPA-ACL *(engl. agent communication language)*, da andere Sprachen wie KQML [FININ et al. 1997] und ARCOL [SADEK et al. 1995] ihren Anforderungen nicht gerecht wurden. Jedoch wurden einzelne Teile und Ideen dieser Sprachen übernommen. FIPA-ACL besteht aus einer Menge von standardisierten *Sprechakten* [FIPA 2000a], im englischen *communicative acts (CA)*. Die Sprechakte beruhen auf der „Sprechakttheorie “ *(engl. speech act theory)* von SEARLE [1969]. Hinter jedem Sprechakt steht eine definierte Aussage, die einen Dialog zwischen Agenten aufbaut. Jeder Sprechakt enthält Informationen über Inhalt, Typ und möglichen Folgen des Sprechaktes, d.h. jeder andere Agent ist sich des Zwecks der Nachricht bewusst. Um dies zu erreichen, wird die entsprechende Nachricht verpackt *(engl. message encoding)*. So werden die Adresse von Sender und Empfänger, als auch weitere Attribute zur eigentlichen Nachricht hinzugefügt. Der „Umschlag“ der Nachrichten enthält Informationen, die von der FIPA-Agentenplattform zum Nachrichtenversand genutzt werden. Ferner wird ein *Konversations-ID* benutzt, um Dialoge zwischen zwei Agenten zu koordinieren. Die

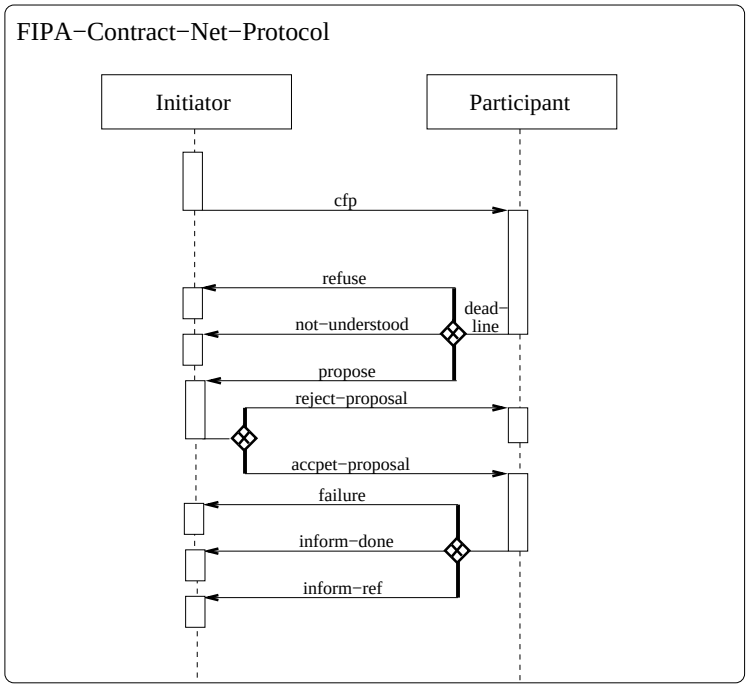

Abbildung 2.1: Das Diagramm des FIPA-Contract-Net-Protocols

agent communication language umfasst sowohl *primitive Sprechakte* wie beispielsweise „request“ und „inform“ [FIPA 2000b], als auch *zusammengesetzte CAs* . Die zusammengesetzten Sprechakte setzen sich aus Kombinationen der primitiven zusammen. Dabei wurde beachtet, dass die semantische Integrität der Sprache gewährleistet ist und dass die Agenten eine gemeinsame *Ontologie* besitzen, sich also gegenseitig verstehen.

Dialoge werden als vorgegebene Folgen von Sprechakten dargestellt und repräsentieren eine vollständige Konversation. Sie folgen den Interaktionsprotokollen *(engl. interaction protocols)* der FIPA. Der Aufbau der Dialoge basiert auf den Arbeiten von LUX und STEINER [1998] und HAUGENEDER [1994]. Jedes Interaktionsprotokoll setzt sich aus mehreren festgelegten Nachrichten zwischen einem Sender und einem Empfänger zusammen [FIPA 2000c]. Diese Nachrichtenwechsel werden anhand von Protokolldiagrammen *(engl. protocol diagrams)* dargestellt (siehe Abb. 2.1). So können *Auktionen*, *Registrierung* und *Deregistrierung* der Agenten über Interaktionsprotokolle abgewickelt werden. Jeder Agent des Systems muss sich konsistent zu den in den Protokolldiagrammen gemachten Spezifikationen verhalten. Agenten können in mehrere Konversationen mit verschiedenen Interaktionsprotokollen gleichzeitig involviert sein. Dabei fallen den Agenten bestimmte Rollen zu (z.B. die eines Verkäufers und eines Käufers eines Produktes). Mehrere Rollen können gleichzeitig an einer Interaktion teilnehmen.

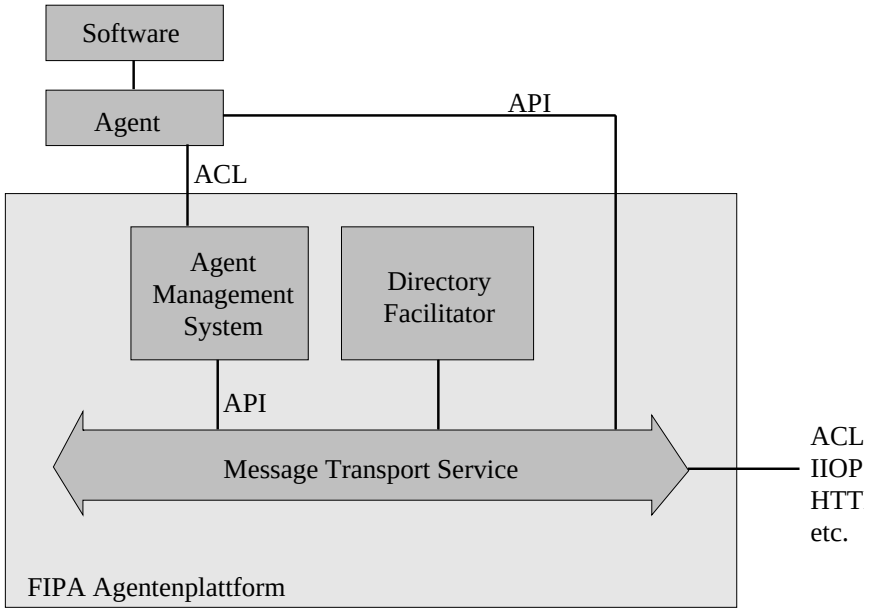

Abbildung 2.2: FIPA Agentenmanagement

Entwickler können der ACL eigene konforme Interaktionsprotokolle hinzufügen wie beispielsweise SIERRA et al. [1998] und SERRANO und OSSOWSKI [2002], d.h. diese Protokolle müssen die in [FIPA 2000a] gestellten Anforderungen erfüllen. So müssen beispielsweise übermittelte Nachrichten Inhalte wie *Sender, Empfänger*, deren *Konversations-ID* usw. enthalten.

Agentenmanagement

Im Folgenden werden die wir die Bestandteile der Agentenplattform aufzählen, die die reibungslose Interaktion zwischen den Agenten gewährleisten. Diese Agentenverwaltung soll nach O'BRIEN und NICOL [1988] folgende Charakteristika aufweisen:

- die Nachrichten sollten effizient übermittelt werden
- jeder Agent sollte einen eigenen Namen erhalten
- das Agentenmanagement sollte Fehler erkennen und berichten
- Mobilität und Sicherheit sollten bereitgestellt werden

Die Ansprüche an eine Agentenplattform sind bis auf Mobilität und Sicherheit in [FIPA 1997] verwirklicht. Die Infrastruktur dieser FIPA-OS *(engl. FIPA open source)* genannten Entwicklungsumgebung besteht aus folgenden Komponenten [O'BRIEN und NICOL 1988, POSLAD et al. 2000] und ist in Abb. 2.2 dargestellt:

- *Directory Facilitator (DF)*
  Der DF ist ein spezieller Agent, der die Agenten untereinander vermittelt (*engl. yellow pages services*). Darunter fallen Aufgaben wie Registrierung, Informationsbeschaffung über andere Agenten etc.

- *Agent Management System (AMS)*
  Das AMS teilt jedem Agenten einen eindeutigen Namen mit Hilfe des *agent name service* zu. Die Namen aller auf der Agentenplattform registrierten Agenten und deren Transportadressen, also deren aktuelle Aufenthaltsorte, werden in einer Liste gespeichert. Dies beinhaltet auch die Erzeugung, Registrierung und Löschung von Agenten (*engl. white pages services*).

- *Agent Communication Channel (ACC)*
  Die Kommunikation zwischen den Agenten innerhalb einer Agentenplattform, als auch zwischen den verschiedenen Agentenplattformen, übernimmt der *agent communication channel*. Die Agentenplattform muss zumindest das *internet inter-orb protocol (IIOP)* unterstützen.

- *Message Transport Service (MTS)*
  Über den *message transport service* werden Nachrichten innerhalb der Agentenplattform versendet. Zum effizienten Versand werden die Nachricht sowie ihre Hülle kodiert.

Da die Agenten gemäß den FIPA-Spezifikationen interagieren sollen, benötigten wir eine FIPA-konforme Agentenplattform als Basis für unser Spamfilternetzwerk. Unter den verschiedenen Implementierungen des FIPA-Standards wählen wir FIPA-OS, eine frei verfügbare Entwicklungsumgebung. FIPA-OS als Agentenplattform bietet den Vorteil, dass es von verschieden Gruppen des Deutschen Forschungszentrums für Künstliche Intelligenz (DFKI), an dem diese Diplomarbeit geschrieben ist, bereits in Projekten wie CASA und SAID eingesetzt wird. Somit können wir vom Wissen und vom Erfahrungsaustausch mit anderen Forschungsgruppen profitieren.

Agenten/Softwareintegration

Dieser Teil der FIPA-Spezifikationen beschreibt Möglichkeiten, Nicht-Agentensoftware in die FIPA-Umgebung einzubinden und darauf zuzugreifen, als wäre die Software selbst ein Agent. Damit ist ein Zugriff auf die Funktionen der Software durch FIPA-konforme Agenten möglich. Dies wird mit Hilfe von *Wrapperagenten* realisiert. Diese Agenten fungieren als Schnittstelle zwischen der Software und der FIPA-Plattform. So ermöglichen beispielsweise GEORGOUSOPOULOS und RANA [2002] mit Hilfe von *Gateway-Agenten* als Schnittstellen die Kommunikation zwischen FIPA konformen und anderen Agentensystemen. Der *Gateway-Agent* übersetzt Nachrichten des externen Multiagentensystems in eine Form, die von den Agenten des FIPA-OS verstanden wird. Der innere Aufbau der FIPA Agentenplattform ist für das externe System nicht

sichtbar. Innerhalb unseres Spamfilternetzwerkes arbeitet das Multiagentensystem mit einem Algorithmus zur Textklassifizierung zusammen. Dieser Algorithmus trägt den Namen *support vector machines* und wird im folgenden Kapitel behandelt.

2.2 Support Vector Machines

Wir wollen den Inhalt von Emails innerhalb unseres Spamfilternetzwerkes mit Hilfe eines Algorithmus zur Textklassifizierung analysieren. Deshalb werden wir im ersten Abschnitt grundlegende Begriffe definieren und erklären. Wir haben die Eigenschaften verschiedener Algorithmen zur Textklassifizierung (Textklassifizierer) miteinander verglichen und den Klassifizierungsalgorithmus *support vector machines* ausgewählt. *Support vector machines (SVM)* sind ein relativ neuer Lernansatz. Dieser wurde von VAPNIK [1995] vorgestellt. Im zweiten Abschnitt dieses Kapitels werden wir den Algorithmus der SVM näher beleuchten. Nachdem die theoretischen Grundlagen abgearbeitet sind, erläutern wir im dritten Abschnitt die Funktionsweise von SVM anhand eines Beispiels. Im vierten Abschnitt folgt die Vorstellung verschiedener anderer Klassifizierungsalgorithmen zur Textklassifizierung sowie der Vergleich der Klassifizierungsgenauigkeit dieser Algorithmen mit der von *support vector machines*. Im selben Abschnitt erklären wir auch die Gründe für die Wahl von SVM als Textklassifizierungsalgorithmus für unseren Spamfilter.

2.2.1 Begriffsklärungen

Bevor wir die verschiedenen Algorithmen zur Klassifizierung von Text vorstellen, wollen wir die Bedeutung wichtiger Begriffe präzisieren. So definieren wir zuerst den Begriff *Klassifizierung*:

Definition 3 *Als Klassifizierung bezeichnen wir den Prozess, Objekte zu einer Menge von vordefinierten Kategorien zuzuordnen, wobei ein Objekt zu keiner oder zu mehreren Kategorien zugeordnet werden kann.*

Textklassifizierung ist somit der Prozess, eine Menge von Texten (oder Dokumenten) in Kategorien einzuteilen. Als Gebiet des Maschinellen Lernens ist Textklassifizierung ein klassisches Beispiel für ein Verfahren des überwachten Lernens *(engl. supervised learning)*. Der Algorithmus, der den Text klassifizieren soll, „lernt“ eine Klassifizierungsfunktion anhand von Trainingsdaten. Diese Trainingsdaten sind bereits in Kategorien eingeteilt und werden bereitgestellt. Nach Abschluss der Lernphase kann die Erkennungsrate der Klassifizierungsfunktion mit Hilfe von Testdaten überprüft werden. Dabei sind die Kategorien der Testdaten im voraus bekannt, werden dem System jedoch nicht mitgeteilt.

Der *Klassifizierungsfehler* eines Spamfilters gibt den Prozentsatz von Emails an, den der Filter falsch klassifiziert, d.h. den Anteil von Spammails, die er als Nicht-Spam kategorisiert und vice versa im Verhältnis zu der Gesamtzahl zu klassifizierender Emails.

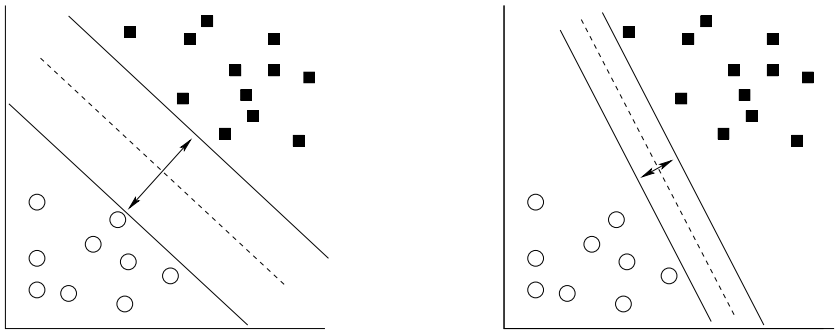

Abbildung 2.3: Konstruktion der Hyperebene mit SVM, die die Grenze zwischen den Datenpunkten zweier Kategorien maximiert.

Man ist bei der Entwicklung von Spamfiltern bestrebt, einen möglichst niedrigen Klassifizierungsfehler zu erreichen.

2.2.2 Herleitung und Definition

Der Klassifizierungsalgorithmus *support vector machines* erzeugt die Hyperebene, die Datenpunkte bestmöglich in verschiedene Kategorien unterteilt. Die Datenpunkte liegen dabei in Form von Vektoren im mehrdimensionalen Raum vor. Die durch *support vector machines* konstruierte Hyperebene unterteilt die Kategorien mit maximalem Abstand und minimiert dadurch die Anzahl der Klassifizierungsfehler. Der Vorgang der Errechnung der maximalen Hyperebene und damit das Lernen des optimalen Klassifizierers maximaler Trenngüte wird auch als Trainingsphase bezeichnet. Die Daten, die bei der Berechnung benutzt werden, heißen Trainingsdaten. Ist die Hyperebene konstruiert, so werden positive und negative Testdaten in der Testphase in die entsprechenden Kategorien eingeteilt. Ein Beispiel über Hyperebenen zwischen Datenpunkten zweier Kategorien im zweidimensionalen Raum findet sich in Abb. 2.3. Die Datenpunkte der ersten Kategorie sind durch ausgefüllte Rechtecke dargestellt, die Datenpunkte der zweiten Kategorie durch unausgefüllte Kreise. Die Hyperebene der linken Abbildung maximiert im Vergleich zur Hyperebene der rechten Abbildung die Grenze zwischen den Mengen von separierbaren Datenpunkten der beiden Kategorien. Wir werden im Folgenden die binäre Klassifizierung mit *support vector machines* für den Fall linear separierbarer Daten herleiten:

Definition 4 (Vapnik und Wu, 1998)
Eine Menge von klassifizierten Trainingsdaten

$$(x_1, y_1), \ldots, (x_l, y_l), \quad x \in R^n, y_i \in \{-1, 1\}, \quad i = 1, \ldots, l$$

heißt linear separierbar, wenn ein Vektor $w \in R^n$ und ein Skalar b existieren, so dass für $i = 1, \ldots, l$ gilt:

$$\begin{array}{rrl} w^T x_i - b \geq & 1 & if\ y_i = \quad 1 \\ w^T x_i - b \leq & -1 & if\ y_i = -1 \end{array}$$

Wir betrachten die Klassifizierung durch SVM zur Vereinfachung nur für linear separierbare Datenpunkte. Die optimale Hyperebene mit dem Vektor w^*, die die Trainingsdaten durch die größtmögliche Grenze teilt, wird bestimmt durch:

$$w^{*T} x - b = 0 \tag{2.1}$$

Sie ermittelt die Richtung $\frac{w}{|w|}$, bei der die Distanz der Projektionen der beiden Trainingsvektoren zweier verschiedener Klassen maximal ist.

Definition 5 *Die Grenze $\rho(w, b)$ ist als Distanz der beiden nächstgelegenen Punkte beider Seiten der Hyperebene definiert*

$$\rho(w, b) = \min_{\{x:y=1\}} \frac{w^T x}{|w|} - \max_{\{x:y=-1\}} \frac{w^T x}{|w|} \tag{2.2}$$

Die optimale Hyperebene (w^*, b^*) maximiert die Distanz in Formel (2.2).

$$\rho(w, b) = \frac{2}{w^*} = \max \frac{2}{|w|} \tag{2.3}$$

Geometrisch ist dies vergleichbar mit der Maximierung der Grenze oder Distanz zwischen den beiden parallelen Ebenen $w^T x = b + 1$ und $w^T x = b - 1$. Mit $\|w\|_2 = \sqrt{\sum_{i=1}^{n} w_i^2}$ kann (2.3) als quadratisches Optimierungsproblem unter linearer Ungleichheitsbedingung dargestellt werden:

$$\begin{array}{ll} & \min_{w,b} \frac{1}{2}\|w\|^2 \\ \text{mit} & y_i[(w^T x_i) - b] \geq 1, \qquad y_i \in \{1, -1\}, \quad i = 1, \ldots, l \end{array} \tag{2.4}$$

Das duale Problem der Formel (2.4) ist:

$$\begin{array}{l} \max_\alpha W(\alpha) = \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} \alpha_i \alpha_j y_i y_j (x_i^T x_j) \\ \text{mit } \sum_{i=1}^{l} \alpha_i y_i = 0, \quad \alpha_i \geq 0, \quad i = 1 \ldots l \end{array} \tag{2.5}$$

Falls α^* die Formel (2.5) löst, ist der Gewichtsvektor w^* gegeben durch:

$$w^* = \sum_{i=1}^{l} \alpha_i^* y_i x_i, \quad \text{mit} \quad \alpha_i^* \geq 0, \quad i = 1 \ldots l \tag{2.6}$$

Die x_i mit $\alpha_i \geq 0$ heißen *support vectors*, sie lösen $y_i[(w^T x_i) - b] = 1$. Bemerkenswert ist die Tatsache, dass die Lage der Hyperebene nur durch die *support vectors* festgelegt wird. Die α_i^* aller anderen Vektoren haben den Wert null und sind zur Bestimmung der Lage der Hyperebene irrelevant. Der Wert von b^* taucht nicht im dualen Problem auf und muss deshalb mit Hilfe der primalen Bedingungen erstellt werden:

$$b^* = \frac{1}{2}[w^{*T} x^*(1) + w^{*T} x^*(-1)] \tag{2.7}$$

$x^*(1)$ ist jeder *support vector* der Klasse 1, $x^*(-1)$ jeder der Klasse -1. Dies führt schließlich zum *support vector* Klassifizierer für einen beliebigen zu klassifizierenden Vektor x:

$$f(x) = sign\{w^{*T} x - b^*\} \quad \text{mit} \quad w^* = \sum_{i=1}^{N} v_i x_i^* \tag{2.8}$$

Der hergeleitete Klassifizierer arbeitet nur auf linear separierbaren Daten. $x_1^*, \ldots, x_N^*$ sind die *support vectors* und die v_i die vom Algorithmus zu bestimmenden Gewichtungen für die einzelnen *support vectors* x_i^*. Im Falle nicht linear separierbarer Datenpunkte können SVM in einem hochdimensionalen Kern-induzierten Merkmalsraum arbeiten, in dem viele reale Probleme linear trennbar sind. Da wir im Rahmen dieser Arbeit nur die grundlegende Arbeitsweise von *support vector machines* beleuchten wollen, sei für die Verwendung von SVM zur Klassifizierung nicht linear separierbarer Daten auf JOACHIMS [2001] verwiesen. Die Effizienz von SVM beruht auf der Tatsache, dass nur wenige Trainingsdaten, die *support vectors*, bei der Klassifizierung von Testdaten berücksichtigt werden müssen. Sie allein bestimmen die Lage der Hyperebene.

Implementierungen der *support vector machines* sind in SVM^{LIGHT} von JOACHIMS [1998] und dem *sequential minimal optimization* (SMO) Algorithmus von PLATT [1998] vorhanden. Ferner existiert *LIBSVM* von CHANG und LIN [2002], eine Bibliothek für *support vector machines* und Regressionen. SVM arbeiten in linearer Zeit. Es existieren keine Parameter, die zum Erreichen optimaler Klassifizierungsergebnisse angepasst werden müssen [VAPNIK et al. 1999].

2.2.3 Funktionsweise von SVM anhand eines Beispiels

Nachdem wir die Theorie der *support vector machines* im vorherigen Abschnitt behandelt haben, werden wir die Vorgehensweise der *support vector machines* bei der Klassifizierung eines Vektors anhand eines Beispiels betrachten. Sei M die Menge der *support vectors* $v_{support}$ mit $M = \{(1;1;0;0), (1;0;0;1), (1;0;1;0), (1;1;1;1)\}$. Die Gewichtungen der vier *support vectors* seien durch den SVM-Algorithmus errechnet mit $v_i = \{-4, -2, 3, 4\}$, $i \in \{1, \ldots, 4\}$ und das Skalar mit $b = 0$. Dann wird ein

Vektor v_{klass} mit $v_{klass} = (1; 1; 0; 1)$ folgendermaßen mit Hilfe des Skalarprodukts aus v_{klass} und den $v_{support}$ klassifiziert:

v_i	$v_{support}$	$Skalarprodukt$			
-4	(1;1;0;0)	-4	(1 x 1 + 1 x 1 + 0 x 0 + 1 x 0)	=	-8
-2	(1;0;0;1)	-2	(1 x 1 + 1 x 0 + 0 x 0 + 1 x 1)	=	-4
3	(1;0;1;0)	3	(1 x 1 + 1 x 0 + 0 x 1 + 1 x 0)	=	3
4	(1;1;1;1)	4	(1 x 1 + 1 x 1 + 0 x 1 + 1 x 1)	=	12
			$\sum$	=	3

Tabelle 2.1: Klassifizierung eines Vektors durch Bildung des Skalarprodukts mit den gewichteten *support vectors* und Aufsummierung der Ergebnisse

Zur Klassifizierung von v_{klass} wird das Skalarprodukt von v_{klass} mit den vier *support vectors* gebildet und mit der Gewichtung des *support vectors* multipliziert. Die Aufsummierung der Ergebnisse liefert nach Formel 2.8 die Klassifizierung K des Vektors. Da in unserem Beispiel $K = 3$ gilt, und damit $K > 0$ ist, wird der Vektor positiv klassifiziert, d.h er wird der entsprechenden Kategorie zugeordnet.

2.2.4 Vergleich verschiedener Textklassifizierer

Im Gebiet der automatisierten Textklassifizierung TC *(engl. text categorization)* existiert eine Vielzahl effizienter Algorithmen zur Filterung von Texten. Ein rein *kontentbasierender Filter* klassifiziert Informationen anhand eines Profils, welches er aus dem Inhalt der Information und der Bewertung durch einen einzelnen Benutzer erzeugt hat. Die Wurzeln dieser Algorithmen liegen im *Information Retrieval (IR)* . Nicht klassifizierte Daten werden anhand bestehender Profile in verschiedene Kategorien eingeteilt. Diese Profile werden anhand von bereits klassifizierten Trainingsdaten gelernt, welche bereitgestellt werden.

Der Klassifizierungsalgorithmus, der die Aufgabe der Einteilung der Emails in die Kategorien Spam und Nicht-Spam übernimmt und in unserem Antispam-Agenten eingesetzt wird, muss mehrere Kriterien erfüllen. So muss sich die Zeit zum Trainieren des Klassifizierers in einem vernünftigen Rahmen bewegen. Dem Benutzer des Antispam-Agenten kann keinesfalls eine Zeitspanne von mehreren Stunden für die Aktualisierung des Klassifizierers zugemutet werden. Weiterhin muss der Algorithmus die Klassifizierung einer Email in schnellstmöglicher Zeit absolvieren. Die wichtigste Eigenschaft muss jedoch eine möglichst hohe Klassifizierungsgenauigkeit sein. Wir werden nun die wichtigsten Klassifizierungsalgorithmen in der Literatur neben SVM kurz vorstellen. Danach werden wir die verschiedenen Algorithmen hinsichtlich ihrer Klassifizierungsgenauigkeit vergleichen und begründen, warum wir uns für die Verwendung des Klassifizierers *support vector machines* innerhalb des Antispam-Agenten entschieden haben.

Wir zählen im Folgenden verschiedene Algorithmen auf, die von kontentbasierenden Filtern zur autonomen Klassifizierung eingesetzt werden:

Rocchio-Algorithmus mit TF-IDF

Der Rocchio-Relevanz-Feedback-Algorithmus [ROCCHIO 1971] ist eine weit verbreitete und angewendete Lernmethode zur Textkategorisierung. Die heuristische Hauptkomponente des Rocchio-Algorithmus ist die Wortgewichtung mit Termfrequenz und inverser Dokumentfrequenz TF-IDF *(engl. term frequency / inverse document frequency)* [SALTON und BUCKLEY 1988]. Jedes Dokument wird durch einen Vektor dargestellt, dessen Attribute die zur Klassifizierung zu berücksichtigenden Wörter sind. Diese Darstellung von Dokumenten findet sich beispielsweise auch bei AGNE et al. [2002]. Dokumente mit gleichem Inhalt haben einen gleichen Vektor. Die Gewichtung eines Wortes wird mit Hilfe von Termfrequenz und Dokumentfrequenz berechnet. Die Termfrequenz *(engl. term frequency)* gibt an, wie oft ein Wort in einem Dokument vorkommt, die Dokumentfrequenz *(engl. document frequency)*, in wie vielen Dokumenten das Wort mindestens einmal vorkommt. Ein Wort ist relevant zur Klassifizierung eines Dokumentes, wenn es darin oft vorkommt. Wörter, die in allen Dokumenten häufig vorkommen, werden dagegen schwächer gewichtet. Aus der Summe von Dokumentenvektoren wird ein Prototypvektor für jede Klasse gebildet. Um ein neues Dokument zu klassifizieren, werden der Kosinus sowohl des Prototypvektors als auch des zu klassifizierenden Dokumentes bestimmt, es wird derjenigen Klasse zugeordnet, mit der sein Dokumentenvektor den größten Kosinus hat. Die komplette Herleitung kann bei JOACHIMS [1997] nachgelesen werden.

BALABANOVIC und SHOHAM [1995] verwenden die TF-IDF-Gewichtung im System LIRA, um anhand eines Benutzerfeedbacks für den Anwender relevante Webseiten aus dem Internet zu filtern. Der Rocchio-Algorithmus arbeitet in linearer Zeit und schnell auf Trainings- und Testdaten. Nachteilig wirkt sich die Ermittlung der zur Textkategorisierung optimalen Parameter aus.

Naive Bayes Klassifizierer

Der Naive Bayes Klassifizierer (NB), wie bei JOACHIMS [1997] beschrieben, berechnet die Wahrscheinlichkeit, mit der ein Text einer bestimmten Kategorie angehört. Für jede Kategorie existiert ein Wahrscheinlichkeitsmodell C. Der Naive Bayes Klassifizierer teilt einen Text mit Hilfe des Bayes Theorem in eine bestimmte Kategorie ein anhand der gemischten Wahrscheinlichkeiten von Wörtern und Kategorien. Ordnen wir jeder Kategorie eine Klasse C_j zu und bezeichnen mit d ein neu zu klassifizierendes Dokument, so versuchen wir die Wahrscheinlichkeit zu bestimmen, mit der d in C_j enthalten ist (= $Pr(C_j|d)$). Der optimale Bayessche Klassifizierer ist derjenige, für den $Pr(C_j|d)$ am höchsten ist.

Zur Vereinfachung wird bei der Verwendung des NB Klassifizierers angenommen, dass Wörter unabhängig von der Dokumentlänge und dem Erscheinen anderer Wörter im Dokument vorkommen. Der Naive Bayes Klassifizierer findet seine Verwendung zum Spamfiltern beispielsweise bei SAHAMI et al. [1998].

K-nearest Neighbor

Der k-nächste Nachbar (kNN) Klassifizierer *(engl. k-nearest neighbor)* ist einer der ältesten und am längsten erprobten Klassifizierer zur Textkategorisierung [IWAYAMA und TOKUNAGA 1995, YANG 1994]. Alle Trainingsdaten werden in einem mehrdimensionalen Raum, der durch die Anzahl der Attribute bestimmt wird, gespeichert. Sie dienen direkt zur Klassifizierung der Testdaten. Beim kNN werden zu einem beliebigen Testdokument die k nächsten Nachbarn unter den Trainingsdaten gesucht. Deren zugehörige Kategorien bestimmen die Kategorie des zu klassifizierenden Dokuments. Die Ähnlichkeitspunktzahl jedes Nachbardokuments zum Testdokument dient zur Gewichtung der zugehörigen Kategorie des Nachbarn. Liegen mehrere der k Nachbarn in derselben Kategorie, so wird zwischen ihnen die gewichtete Summe gebildet. Die Ähnlichkeit zwischen Trainingsdokument und Testdokument wird durch den Kosinus ihrer Vektoren bestimmt [YANG und LIU 1999].

ID3, C4.5 und PART

ID3 und C4.5 [QUINLAN 1993] klassifizieren Daten anhand von Entscheidungsbäumen. Im allgemeinen verfolgen Entscheidungsbäume den „Teile und Herrsche“-Ansatz. Man sucht auf jeder Baumebene ein Attribut für die Zerlegung und verarbeitet rekursiv die aus der Zerlegung entstandenen Teilprobleme. Jeder Knoten innerhalb der Entscheidungsbäume entspricht einem Test auf den Wert eines Attributs und jede Kante einem möglichen Wert dieses Attributs. Bei der Entscheidungsfindung beginnt man an der Wurzel des Baumes und geht in der Top-Down-Methode anhand bestimmter Attributwerte die entsprechenden Kanten entlang bis man zu einem Blatt gelangt. Die zu diesem Blatt des Baumes korrespondierende Kategorie ist der Entscheidungswert für die auf diesem Pfad durchlaufenden Attributwerte. Die Struktur des Algorithmus zur Erzeugung des Baumes in ID3 kann bei [ID3C4.5 1993] nachgelesen werden. Der Entscheidungsbaum von ID3 wird so aufgebaut, dass Attribute mit dem höchsten Informationsgehalt die Knoten der obersten Ebenen des Entscheidungsbaumes bilden, d.h. sie klassifizieren die Daten am stärksten. Aus dem Baum können Entscheidungsregeln generiert werden.

C4.5 ist eine Weiterentwicklung von ID3. So kann C4.5 auch Trainingsdaten mit unbekannten Attributwerten verarbeiten. Ferner kann durch *pruning* die Größe des Baumes verringert werden, indem bestimmte Äste durch einen einzigen Knoten ersetzt werden [QUINLAN 1987]. Dies verhindert die Überanpassung des Baumes *(engl. overfit-*

ting). Alternativ zur Erzeugung von Regeln aus dem kompletten Entscheidungsbaum werden bei PART [WITTEN und FRANK 1999] Teilbäume aus dem mit C4.5 erzeugten Baum herausgegriffen und separat dafür Regeln erzeugt.

Weitere Algorithmen

Neben den schon genannten existieren noch weitere Algorithmen zur Textklassifizierung, die jedoch im Vergleich mit den bisher beschriebenen Algorithmen schlechtere Klassifizierungsergebnisse liefern. Wir wollen drei dieser Algorithmen kurz vorstellen:

- *Regelbasierte Algorithmen*
  Regelbasierte Lernalgorithmen wie RIPPER [COHEN 1995b] generieren eine Menge von Regeln, die alle positiven und negativen Daten abdecken. Diese werden meist noch vereinfacht und zusammengefasst.

- *Neurale Netze*
  WIENER et al. [1995] verwenden für jede zu klassifizierende Kategorie ein separates neurales Netz. Der Aufbau eines neuralen Netzes zur Textklassifizierung gestaltet sich langwierig [YANG und LIU 1999]. Die simpelste Form eines Neuralen Netzwerkes ist das Perceptron.

- *Information Extraction (IE)*
  IE-Systeme identifizieren und extrahieren spezifische Informationen aus Texten. Informative Muster werden in *Essence* [CATALÀ et al. 2000] aus dem Text herausgesucht und mit Synonymen verglichen. Die Ausgabe der Daten erfolgt benutzerspezifisch, d.h. durch Informationsextraktion *(engl. information extraction)* können bestimmte Informationen (z.B. Uhrzeit von Nachrichten) aus den Mustern herausgefiltert werden.

Vergleich der verschiedenen Algorithmen zur Textklassifizierung

Die leistungsfähigsten Algorithmen zur Klassifizierung von Spam nach VAPNIK et al. [1999] sind *support vector machines* (mit binärer Repräsentation der Wörter), knapp vor einer modifizierten Variante von C4.5. Die binäre Repräsentation gibt an, ob ein Wort in einem bestimmten Text enthalten ist oder nicht. SVM und C4.5 werden in dieser Versuchsreihe außerdem mit RIPPER und Rocchio verglichen. Ripper klassifiziert bei einigen Trainingsdatensätzen mehr als fünf Prozent der spamfreien Emails als Spam, die anderen Algorithmen blieben alle unter diesem Schwellenwert. SVM setzen sich gegen C4.5 bei fast gleichen Erkennungsraten durch, da ihre Trainingszeit viel kürzer als die von C4.5 ist. Bei VAPNIK und WU [1998] erreichen SVM auf allen Datensätzen Genauigkeiten von 95.8% bis 98.5%. Auch unter SHANKAR und KARYPIS [2000], NEUMANN und SCHMEIER [2002] erreichen SVM die beste Performanz vor Rocchio und gewichteten Algorithmen. HIDALGO et al. [2000] vergleichen die Lernalgorithmen Naive Bayes, C4.5, K-nearest Neighbor und PART im Hinblick auf das

kostenbasierte Filtern von Spam. Nach Prüfung der Algorithmen kommen sie zu dem Schluss, dass der Algorithmus PART und der verwandte Algorithmus C4.5 [QUINLAN 1993] am besten zum Filtern geeignet ist. SVM werden hier nicht berücksichtigt. RIPPER schneidet auch bei PROVOST [1999] schlechter ab als Naive Bayes. Erreicht Naive Bayes schon nach 50 Trainingsdaten 95 Prozent Genauigkeit, sind es bei RIPPER nach 400 Trainingsdaten gerade 90 Prozent Genauigkeit. In der Wertung der besten Lernalgorithmen zur Textklassifizierung von YANG und LIU [1999] platzieren sich SVM und kNN auf den ersten Plätzen. Dagegen liegen die Performanzwerte von Naive Bayes und neuronalen Netzen in allen Trainingsdurchläufen weit hinter den zuvor genannten Algorithmen zur Klassifizierung. Beim alleinigen Vergleich von kNN und neuronalen Netzen setzt sich kNN als bester Klassifizierer durch [CÖSTER und ASKER 2000]. Überraschenderweise zeigt sich bei YANG [1999], dass der Rocchio-Algorithmus mit TF-IDF bei der Textklassifizierung signifikant schlechter abschneidet als Algorithmen wie kNN, ja sogar schlechter als RIPPER. ANDROUTSOPOULOS et al. [2000] vergleichen Naive Bayes und kNN zur Spamfilterung mit Schlüsselwörtern und zeigen, dass diese komplexeren Techniken einfachen Filtern weit überlegen sind.

Ein sehr gutes Werkzeug zum Vergleich verschiedener Klassifizierer ist *Weka* von WITTEN und FRANK [1999]. In der Weka-Entwicklungsumgebung sind Lernalgorithmen wie beispielsweise Naive Bayes, SVM, C4.5, kNN und PART enthalten. Jeder Algorithmus ist in einer eigenen Klasse implementiert. Mit dieser Sammlung können verschiedene Lernalgorithmen direkt miteinander verglichen werden [HIDALGO et al. 2000].

Zusammenfassend lässt sich sagen, dass *support vector machines* in fast allen Auswertungen unter den Algorithmen mit der besten Performanz liegt. Auch die Algorithmen C4.5 und kNN erreichen sehr gute Klassifizierungsergebnisse. Dahinter folgt Naive Bayes. Weit abgeschlagen finden sich des weiteren regelbasierte Algorithmen wie RIPPER. Jedoch liegen selbst deren Erkennungsraten bei der Klassifizierung von Spam über denen der einfachen Filter der meisten Emailprogramme. Die hervorragende Erkennungsrate von *support vector machine* kombiniert mit den kurzen Zeiten für die Konstruktion des Klassifizierers anhand der Trainingsdaten haben den Ausschlag gegeben für die Wahl von SVM als Klassifizierer innerhalb des Antispam-Agenten. Die hohe Erkennungsrate ist ein wichtiges Argument für die Akzeptanz des Antispam-Agenten beim Benutzer. Ein Eingreifen des Benutzers zur Korrektur der Klassifizierungsergebnisse, die vom Klassifizierungsalgorithmus nach einer ausreichend großen Trainingsphase selbstständig durchgeführt werden, sollte möglichst selten erforderlich sein.

2.3 Unerwünschte kommerzielle Emails - Spam

Spam hat sich in den letzten Jahren zu einer Plage entwickelt, die Zeit und Kommunikationsressourcen verschwendet. Um die Flut von Spammails wirkungsvoll zu filtern, müssen wir erst einmal klären, woher Spam kommt und was Spam genau ist. Deshalb werden wir im ersten Abschnitt dieses Kapitels den Begriff *Spam* genauer definieren. Danach zeigen wir im zweiten Abschnitt, welche negativen Folgen für den Benutzer eines Emailaccounts durch die Zusendung von Spam entstehen. Im dritten Abschnitt erklären wir, mit welchen Methoden die Absender der Spammails an die Emailadressen gelangen, und woher sie diese Adressen im Internet bekommen. Da wir einen Spamfilter entwickeln wollen, der einen möglichst großen Anteil der Spammails erkennt, verschaffen wir uns im vierten Abschnitt einen Überblick über Filtermethoden, die bereits in der Praxis angewendet werden. Schließlich folgt im fünften Abschnitt eine Auflistung der wichtigsten experimentellen und kommerziellen Email-Klassifizierungsprogramme und Spamfilter, die bereits entwickelt worden sind.

2.3.1 Definitionen und Begriffsklärungen

Da der Diplomarbeit das Ziel der Implementierung eines Spamfilters zu Grunde liegt, wollen wir im Vorfeld zuerst den Begriff *Spam* präzisieren:

Definition 6 *Unter Emailspam wird eine Email verstanden, die Werbung oder andere unerwünschte Inhalte enthält und ohne Verlangen oder Zustimmung des Empfängers an diesen versandt wurde. Werden solche Emails mit identischem Inhalt an eine Vielzahl von Empfängern gesendet, so spricht man von Spamming. Der Versender der Werbebotschaften wird als Spammer bezeichnet.*

Spam wird im Englischen auch als *unsolicited bulk email (UBE)* bezeichnet. Die Spezialform der kommerziellen Werbung nennt man *unsolicited commercial email (UCE)*. Viele Spammails enthalten Inhalte nach dem *make money fast (MMF)*-Schema. Dem Empfänger werden schnelle, unglaubwürdige Profite versprochen. Neben Spam in Form von Emails existiert auch die ursprüngliche Form des Usenet-Spam. Hierbei werden einzelne Nachrichten an mehrere Usenetgruppen versendet. Das Usenet entstand 1979 als *Peer-To-Peer* Netzwerk für den Austausch von Nachrichten zwischen Rechnern [ORAM 2001]. Auch die Benutzer des Usenets wurden Opfer der Spammer. Der Fokus dieser Arbeit liegt jedoch auf der Filterung von Emailspam.

Der weit verbreitete Begriff Spam geht auf ein in England und den USA verbreitetes Dosenfleisch mit dem Markennamen „Spam“ zurück. In einem Sketch der Komikergruppe Monty Python wird jegliches Gespräch im Raum durch Singen von „Spam, Spam, Spam ...“ unterbunden. Ein ähnlicher Effekt wird durch das massenhafte Versenden von Spammail erreicht. Das Spamming gestaltet sich als attraktiv, da (nahezu) kostenlos eine große Anzahl von Empfängern erreicht werden kann. Die negativen Auswirkungen von Spam werden im nächsten Abschnitt beschrieben.

2.3.2 Negative Auswirkungen des Spamming

Das Versenden von Spam hat für den Empfänger folgende negativen Folgen:

- *Reale Kosten*
  Viele Benutzer bezahlen ihre Onlinekosten zeitabhängig (z.B. Kosten pro Minute). Der Empfang von Spam kostet sie damit nicht nur Zeit, sondern auch Geld. Dazu kommt der Zeitverlust durch das Sortieren empfangener Nachrichten in Spam und erwünschte Emails. Das Sortieren der Nachrichten führt bei Firmen zu Verlusten in der Produktivität. Ferner wird die Bandbreite des Internets durch Versenden von Millionen von Spammails verschwendet. Da die Mehrzahl der Spammer Einträge aus veralteten Mailinglisten benutzt, werden viele Nachrichten mit einer Antwort zurückgesendet. Dies verbraucht erneut Bandbreite. Auch der Speicherplatz der Internetprovider, der benötigt wird, um Spam bis zum Abruf durch den Benutzer des jeweiligen Accounts zu speichern, wird verschwendet. Beim Spamming lässt der Werbende seine Werbung durch den Beworbenen und die gesamte Internetgemeinde bezahlen. Die Suche nach den Absendern der Spamnachrichten gestaltet sich schwierig, da die Spammer ständig wechselnde Mailserver fremder Betreiber missbrauchen [KASSEL 2002].

- *Soziale Kosten*
  Viele Benutzer sind sich der Tatsache bewusst, dass sie nach Bekanntgabe ihrer Emailadresse in Usenet, Foren oder Mailinglisten mit hoher Sicherheit Spammails empfangen werden. Deshalb nehmen sie an diesen Kommunikationsformen erst gar nicht teil.

2.3.3 Auf welche Arten sammeln Spammer Emailadressen

Sobald man aktiv an den Kommunikationsformen des Internets teilnimmt, besteht die Gefahr, das Opfer von Spammails zu werden. Spammer sammeln Emailadressen auf unterschiedliche Weisen. Automatisierte Werkzeuge tragen sich in alle möglichen Mailinglisten ein. Die Mailinglisten werden von einem *mailing list agent (MLA)* verwaltet. Dies ist ein Softwareprogramm, welches ankommende Nachrichten an alle Mitglieder der Mailingliste weitersendet. Hat sich der Spammer erst einmal in die Liste eingetragen, so kann er die Adressen aller Teilnehmer dieser Liste herausfinden, oder er benutzt die Liste als direktes Ziel für Spamattacken. Auch die Webseiten und die Foren im Internet werden automatisiert durch Spambots nach Emailadressen durchsucht. Das Sammeln der Emailadressen von Internetseiten kann dadurch verhindert werden, dass die Emailadresse zu einem komplexen Javacode konvertiert wird, der vom Browser richtig dargestellt wird, aber von den simplen automatisierten Spambots nicht mehr interpretiert werden kann. Zur Konvertierung der Adressen existieren Programme wie *Spam Vaccine* [SPAMVACCINE 2002] u.a. Die automatisch gesammelten Emailadressen werden meist an andere Spammer weiterverkauft. Besonders wertvoll

sind Adressen, bei denen sichergestellt wurde, dass empfangene Nachrichten auch von einem Menschen gelesen werden.

2.3.4 Implementierung von Filtern

Da das Aussortieren der Spammails Zeit und Geld kostet, wird versucht, die Flut der Spammails durch verschiedene Arten von Filtern einzudämmen und zu minimieren. Die Möglichkeit der Installation von Filtern existiert bei vielen konventionellen Emailprogrammen wie Outlook [OUTLOOK 2002], Pine [PINE 2002] usw., wobei diese Filter in der Regel keine hohe Trefferquote in der Klassifizierung von Spam erreichen, bzw. modifiziert werden müssen, um gute Filterergebnisse zu erlangen. Auch die Betreiber der Mailserver, d.h. der Server, die für das Versenden der Nachrichten zuständig sind, können Filter einsetzen. Die gängigen Methoden zur Filterung von Spam werden in der folgenden Aufstellung zusammengefasst:

Blacklists

Die *blacklist* ist eine Liste von IP-Adressen bekannter Spammer. Die Adressen der Versender von Spam können mit Hilfe von *traceroutes* bestimmt werden. Der Weg der Nachricht bis zu ihrem Versender wird zurückverfolgt und der Spammer zur *blacklist* hinzugefügt. Stimmt die Absenderadresse einer Email mit einem Eintrag in der *blacklist* überein, so wird die betreffende Email als Spam klassifiziert. Dieser Filter bietet nur mäßigen Schutz. Er muss ständig gewartet werden, da Spammer oft ihre Adresse ändern und ständig neue Spammer auftauchen. Emails einer versehentlich in die *blacklist* übernommenen IP-Adresse werden zu Unrecht geblockt. Dieser Filter wird in Spamblockern wie MailMarshal [MAILMARSHAL 2002], SpamEater [SPAMEATER 2002], SpamKiller [SPAMKILLER 2002] und SpamBuster [SPAMBUSTER 2002] verwendet.

Whitelists

Whitelists basieren auf demselben Prinzip wie die *blacklists*. Auch hier wird eine Liste von Adressen bereitgestellt, die mit den Absenderadressen der empfangenen Emails verglichen wird. Der Benutzer des Emailprogramms kann diese Adressen zuvor selbst spezifizieren. Stimmt die Absenderadresse der Email mit einem Listeneintrag der *whitelist* überein, so wird diese Email unter allen Umständen an den Benutzer weitergeleitet. In die *whitelist* werden also benutzerspezifisch diejenigen Absenderadressen eingetragen, von denen der Benutzer Emails ohne Rücksicht auf deren Inhalt empfangen will. *Whitelists* werden beispielsweise von SpamEater und SpamKiller verwendet.

Blocken bei unauflösbaren Domainnamen

Die Absenderadresse der Email wird in eine IP-Adresse umgewandelt. Existiert diese IP-Adresse nicht im Internet, so ist die Email Spam und wird abgelehnt. Dies blockt Spammer, die illegale Domainnamen verwenden und garantiert zumindest eine gültige Absenderadresse. Jedoch existiert keine Garantie dafür, dass diese Adresse auch diejenige ist, von der die Email erzeugt wurde.

Verwendung von Schlüsselwörtern

Der Filter durch *Schlüsselwörter (engl. keywords)* verwendet eine Datenbank mit Wörtern zur Klassifizierung. Die Datenbank enthält Wörter und Phrasen wie „*FREE, FREE, FREE*“, die in Spammails häufiger auftreten als in Nicht-Spammails. Eintreffende Nachrichten werden nach den Schlüsselwörtern durchforstet. Je größer die Anzahl von Schlüsselwörtern in einer Email ist, desto höher ist die Wahrscheinlichkeit, dass es sich bei der untersuchten Email um Spam handelt. Bei Verwendung des Filterns nach Schlüsselwörtern ist Vorsicht geboten, da diese Wörter auch in Nicht-Spammail vorkommen können und somit bei strikter Filterung fälschlicherweise als Spam klassifiziert werden. Die Klassifizierung durch Schlüsselwörter wird beispielsweise von den Spamtools MailMarshal und SpamBuster verwendet.

Weitere Filter

Neben den schon genannten einfachen Filtern existieren noch eine Reihe weiterer Filter, deren Wirkungsweise jedoch sehr beschränkt ist. Beispielsweise kann die Betreffzeile der Email nach nichtalphanumerischen Zeichen wie „$“ durchsucht werden. Auch die Absendezeit der Email ist relevant (die meisten Spammails werden nachts versendet). Ferner kann geprüft werden, ob die Email einen Anhang enthält (Spammails besitzen in der Regel keinen Anhang). Der Inhalt der Email kann nach HTML-Tags und Bildern untersucht werden (viele Spammails sind in Form einer HTML-Tabelle strukturiert). Das Spamtool SpamKiller verfolgt den Weg der Email zum Absender zurück und bietet somit die Möglichkeit zum Filtern aller Emails, die durch fremde Länder geroutet wurden. Die Wirkungsweise dieser Filter ist jedoch beschränkt.

2.3.5 Klassifizierungsprogramme und Spamfilter

Viele der zur Klassifizierung von Emails eingesetzten Programme beruhen auf modifizierten Algorithmen zum Filtern von Informationen. Dabei wird der Benutzer beim Filtern des Datenstroms unterstützt und der Informationsfilter liefert dem Benutzer die relevanten Daten [AAS 1997]. Hier eine Aufstellung verschiedener Klassifizierungsprogramme und Spamtools:

CoTraining

Der *CoTraining*-Algorithmus von BLUM und MITCHELL [2000], modifiziert durch KIRITCHENKO und MATWIN [2001], erzeugt mit nur wenigen Trainingsdaten einen schwachen Klassifizierer und verbessert diesen bei der Bearbeitung der Testdaten kontinuierlich. Dabei wird davon ausgegangen, dass sowohl die Informationen im Betreff *(engl. subject)*, als auch im Inhalt *(engl. content)* zur korrekten Klassifizierung einer Email genügen. Die Wörter im Betreff und im Inhalt werden in zwei verschiedene Mengen gesplittet und für beide Mengen wird jeweils ein Klassifizierer erzeugt. Findet einer der Klassifizierer beim Bearbeiten der Testdaten ein Beispiel, welches er mit hoher Wahrscheinlichkeit einordnen kann, so wird dieses als Trainingsbeispiel für den anderen Klassifizierer verwendet. Hiermit trainiert jeder Klassifizierer den jeweils anderen unter Verwendung von Testdaten. Die verwendeten Lernalgorithmen sind Naive Bayes und SVM.

ifile

Der Algorithmus von *ifile* [RENNIE 2000] klassifiziert Emails anhand ihres Inhalts durch Naive Bayes und leitet sie in einen von mehreren vom Benutzer erstellten Ordnern weiter. Wurde eine Email einer bestimmten Kategorie zugeordnet, so wird das zugrunde liegende Klassifizierungsmodell anhand dieser Zuordnung aktualisiert. Verschiebt der Benutzer falsch klassifizierte Emails in den richtigen Ordner, so ändert der Filter sein Modell entsprechend ab.

Klassifizierung mit Bayes

Der Ansatz von SAHAMI et al. [1998] beruht auf dem Lernen in *Bayesschen Netzen*. Jedes Dokument wird als Vektor von Wörtern aufgefasst. Dieser Vektor wird in einen binären Vektor umgewandelt. Der binäre Vektor einer Email zeigt an, welche Wörter darin enthalten sind und welche nicht. Der Naive Bayes Klassifizierer wird auf den Vektor des Textes der Email angewendet und entscheidet, ob eine Email als Spam klassifiziert wird oder nicht.

Magi

Durch Beobachtung des Verhaltens des Benutzers beim Klassifizieren von Email erzeugt *Magi (engl. mail agent interface)* [PAYNE 1994] ein entsprechendes Benutzerprofil. Für jede vom Benutzer ausgeführte Klassifizierung kreiert Magi Regeln in seiner Wissensbasis. Die Qualität der Regeln wird durch Beobachtung des Benutzerverhaltens weiter verbessert. Der Agent kann sich veränderten Benutzerprofilen dynamisch anpassen, da er von Zeit zu Zeit Regeln verwirft und neue Regeln erstellt. Magi macht dem Benutzer bestimmte Vorschläge zur Klassifizierung der Emails oder teilt völlig autonom die Nachrichten in Kategorien ein.

NewsWeeder und WebWatcher

NewsWeeder [LANG 1995] filtert Neuigkeiten aus dem Internet und ordnet sie einer bestimmten Kategorie zu. Der Benutzer kann die gefilterten Nachrichten bewerten. Basierend auf den Bewertungen passt NewsWeeder sein Benutzerprofil ständig an. Nachrichten werden von NewsWeeder mit Hilfe der TF-IDF-Gewichtung [ROCCHIO 1971] klassifiziert. Hierbei sind Wörter für die Klassifizierung relevant, deren Termfrequenz *tf* in der Nachricht besonders groß ist. Auch wird das Vorkommen der Wörter in allen Nachrichten *df* gezählt. Für ein gegebenes Dokument werden *tf* und *df* in Gewichtungen kombiniert, indem *tf* mit dem Inversen von *df* multipliziert wird.

Personal *WebWatcher* [MLADENIC 1996] ist eine Weiterentwicklung des von ARMSTRONG et al. [1995] entwickelten Programms WebWatcher. Personal WebWatcher unterstützt Benutzer bei der Suche von Informationen im Internet. Dabei analysiert das Programm das Surfverhalten des Benutzers, d.h. die vom Benutzer aufgerufenen Webseiten werden vom Programm klassifiziert und daraus ein Benutzerprofil aufgebaut. Alle vom Benutzer besuchten Verweise werden als positive Beispiele betrachtet, alle nicht besuchten Verweise als negative. Beim Besuch von neuen Internetseiten hebt das Programm diejenigen Verweise hervor, die anhand des Benutzerprofils als interessant für den Anwender angenommen werden. Verschiedene WebWatcher-Agenten können miteinander kommunizieren und Informationen über Gemeinsamkeiten ihrer Benutzerprofile austauschen. Personal WebWatcher interpretiert wie auch NewsWeeder Dokumente als TF-IDF-Vektoren.

Procmail

Procmail ist ein Open-Source-Packet von Werkzeugen [PROCMAIL 2002], die Nachrichten automatisch weiterverarbeiten können. Die Funktionsweise des in der UNIX-Umgebung eingebetteten Programms wird individuell in der Datei *.procmailrc* spezifiziert. Dort kann der Benutzer aus regulären Ausdrücken eigene Regeln zur Filterung von Emails zusammensetzen. Es kann nach dem Absender der Email genauso gefiltert werden wie nach Betreff, Empfänger und nach Begriffen im Inhalt der Email. Somit lassen sich diese Regeln auch zum Filtern von Spam einsetzen. Nach der Klassifizierung der Emails können diese in entsprechende Ordner verschoben werden. Viele andere Emailfilter wie beispielsweise SpamBouncer [SPAMBOUNCER 2002] arbeiten basierend auf dem beliebig erweiterbaren Regelwerk von Procmail.

Regelbasierte Filter

Regelbasierte Filter benutzten zur Spamfilterung ein komplexes Regelwerk von manchmal über 10.000 verschiedenen Regeln. Diese Regeln sind mit denen von Procmail vergleichbar. Auch hier werden Betreff, Inhalt, Absender und Empfänger der Email nach in den Regeln enthaltenen Kriterien untersucht und eingeordnet. Da der Benutzer komplexe Filter mit Tausenden von Regeln schwer überschauen kann, werden solche

Filter wie SpamKiller von McAfee oder BrightMail™Anti-Spam [ANTI-SPAM 2002] ständig über das Internet aktualisiert. Andere Vertreter dieser Art von regelbasierten Filtern sind Lyris MailShield [MAILSHIELD 2002] und Elm [WEINSTEIN 1992]. Die regelbasierten Filter haben den Nachteil, dass sie sich nicht dynamisch an Veränderungen anpassen und deshalb ständig aktualisiert werden müssen.

RIPPER

Der Klassifizierungsalgorithmus von *RIPPER* [COHEN 1996] fügt zu einer anfangs leeren Menge von Regeln kontinuierlich neue hinzu, bis alle positiven Beispiele der Trainingsdaten abgedeckt sind. Danach werden solange die vorhandenen Regeln angepasst, bis die Regelmenge kein negatives Beispiel mehr klassifiziert. Nach der Konstruktion der Regelmenge wird diese optimiert und in ihrer Größe minimiert. Die Filter von RIPPER bieten den Vorteil, dass der Benutzer leicht bestehende Regeln modifizieren und neue Regeln hinzufügen kann. Ist die Regelmenge erstellt, so wird sie im Verlauf des Filterns nicht mehr weiter modifiziert, sie kann sich also neuen Gegebenheiten nicht anpassen. Darin liegt die Schwäche von regelbasierten Filtern [PAZZANI 1999]. Zur Anpassung an das spezielle Problem der Klassifizierung von Emails wurde RIPPER von LEWIS und CATLETT [1994] um einen *Verlustfaktor* erweitert. Dieser Faktor wird eingeführt, um die Klassifizierungsgenauigkeit bei der Analyse von Emails weiter zu erhöhen. Des weiteren wird von COHEN [1996] das Vokabular eingeschränkt, d.h. der Algorithmus berücksichtigt nur häufig vorkommende und informative Wörter. Ripper arbeitet schnell auf Trainings- und Testdaten [VAPNIK et al. 1999].

SpamCop

SpamCop [PANTEL und LIN 1998] behandelt Emails als eine Multimenge von Wörtern und entscheidet mit Hilfe von Naive Bayes, ob eine Nachricht Spam ist oder nicht. In einer Tabelle wird für jedes Wort dessen Auftreten in Spammails und Nicht-Spammails gezählt. Das Vorkommen aller Wörter der zu klassifizierenden Email wird mit den Einträgen der Tabelle verglichen. Daraus wird auf die Wahrscheinlichkeiten geschlossen, mit der eine Email aus dieser Kombination von Wörtern eine Spammail bzw. keine Spammail ist. Nach der größeren der beiden Wahrscheinlichkeiten wird die Email klassifiziert.

Vipul's Razor

Vipul's Razor [Vipul's Razor 2002] ist ein Netzwerk zur Erkennung und Filterung von Spam. Razor beinhaltet einen verteilten und kontinuierlich aktualisierten Katalog von Spamnachrichten auf verschiedenen Servern. Der Benutzer startet einen *Razor Reporting Agent* als Erweiterung eines Emailprogramms wie beispielsweise Outlook [OUTLOOK 2002]. Vom Mailserver geladene Emails werden vom *Razor Reporting Agent*

anhand der im Inhalt der Email enthaltenen Wörter mit einem sie eindeutig bezeichnenden 20-stelligen Identifizierungscode versehen. Danach vergleicht der Agent diesen Identifizierungscode mit den Codes von Spammails, die auf den Servern liegen. Bei Übereinstimmung der Codes ist die empfangene Email Spam und wird geblockt. Existiert der Identifizierungscode auf keinem der Server, so wird die Email mit Hilfe eines regelbasierten Filters analysiert und klassifiziert. Identifiziert der Agent mit Hilfe des regelbasierten Filters eine Email als Spam, so überträgt er den selbst generierten 20-stelligen Identifizierungscode, der diese Email eindeutig kennzeichnet, an den nächsten Server. Der Server seinerseits sendet den Code an andere mit ihm verbundene Server. Der nächste *Razor Reporting Agent*, der die gleiche Spammail mit demselben Inhalt empfängt, kann diese gegen die auf den Servern enthaltenen Identifizierungscodes abgleichen und sicher als Spam erkennen. Spammails, die sich in ihrem Inhalt auch nur geringfügig unterscheiden (z.B. durch persönliche Anrede), werden von Vipul's Razor nicht als solche erkannt.

SpamAssassin™

SpamAssassin™ [SPAMASSASSIN 2002] ist ein Vertreter der regelbasierten Filter, welcher eine breite Palette von heuristischen Tests benutzt, um den Betreff und den Inhalt von Emails zu analysieren. Weiterhin sucht der Algorithmus nach Mustern, welche in Spammails häufig auftreten. Der Filter greift auf mehrere *blacklists* sowie die Datenbasen von Vipul's Razor zu. Jede Regel von SpamAssassin™ erhält eine Gewichtung. Treffen mehrere Regeln auf die zu klassifizierende Email zu, so werden die Gewichtungen der Regeln aufsummiert und aus dem Ergebnis abgeleitet, ob es sich um eine Spammail handelt.

Brightmail™Anti-Spam

Eine weitere Möglichkeit zum Filtern von Spam wird im Werkzeug Brightmail™Anti-Spam verwendet. Das hinter dem Programm stehende Unternehmen sammelt Spam aus strategisch über dem Internet verteilten Emailaccounts. Die Spammails werden durch Experten analysiert, die daraufhin selbst Regeln zum Blocken von Spam schreiben bzw. das bestehende Regelwerk ergänzen. Alle fünf Minuten wird das aktualisierte Regelwerk über Server an den beim Benutzer installierten Spamfilter gesendet. Die Möglichkeit der Aktualisierung der Filter über das Internet wird auch vom Spamtool SpamKiller verwendet.

Andere Filtersysteme

Im Rahmen des Verbundprojektes *Adaptive READ* [Adaptive READ 2003] werden selbstlernende Systeme zur Informationssuche entwickelt, die sich an die Bedürfnisse der Benutzer anpassen. Dabei kommen Verfahren aus dem Bereich des Maschinellen Lernens zum Einsatz. Im Rahmen des Projektes sollen unter anderem die Emails des

Posteingangs von Unternehmen automatisch sortiert werden. Weitere Filtersysteme sind Lira [BALABANOVIC und SHOHAM 1995], PSUN [SORENSEN und ELLIGOTT 1995], Newt [SHETH 1994] und SIFT [GARIA-MOLINA und YAN 1995]. Lira lernt wie NewsWeeder Benutzerprofile anhand von bewerteten Internetseiten über TF-IDF-Gewichtung. Das Benutzerprofil in PSUN wird automatisch anhand der vom Benutzer gelesenen und damit relevanten Usenetartikel generiert. Ähnlich wie PSUN arbeitet Newt. Der Benutzer bewertet die vom System gefundenen Artikel und Newt generiert daraus entsprechende Benutzerprofile.

2.3.6 Zusammenfassung

Die meisten kommerziellen Emailprogramme bieten heutzutage die Möglichkeit zum Filtern von Spam. Jedoch müssen die Regeln zur Klassifizierung meist mühsam selbst vom Benutzer eingegeben und aktualisiert werden. Diese Art des Filterns ist vorwiegend erfahrenen Benutzern vorbehalten. Die hier aufgelisteten Tools sind speziell auf die Klassifizierung von Text spezialisiert und liefern dementsprechend auch ohne Modifikation durch den Anwender gute Ergebnisse. Viele der Spamtools wie Procmail, Magi oder Ripper klassifizieren Nachrichten mit Hilfe von Regeln, die vom Programm selbst erzeugt werden. Dagegen benötigen Filter wie Brightmail™Anti-Spam ständige Wartung ihres Regelwerkes über das Internet. Ein weiterer zum Filtern verwendeter Algorithmus ist Naive Bayes in ifile und SpamCop. *Support vector machines* werden beispielsweise in CoTraining eingesetzt.

Die Klassifizierungsprogramme sind aufgrund der unterschiedlichen Textdatensätze, auf denen sie getestet wurden, schwer zu vergleichen. Jedoch erreichen alle Klassifizierer Erkennungsraten von mindestens 85 Prozent. Manche Programme wie Ripper erreichen hohe Erkennungsraten erst nach der Bearbeitung vieler Trainingsdatensätze, wobei andere Klassifizierer, die mit Naive Bayes oder SVM arbeiten, weniger Training benötigen [PROVOST 1999]. Die in diesen Programmen eingesetzten Algorithmen zur Textklassifizierung haben wir in Kapitel 2.2.4 verglichen.

Wir wollen die Klassifizierungsgenauigkeit unseres verteilten Spamfilters durch den Austausch von Spaminformationen erhöhen und somit den Anteil der Emails verringern, die falsch klassifiziert werden. Die verschiedenen Aufgaben, die wir bei der Implementierung des verteilten Spamfilters zu lösen haben, stellen wir im nächsten Kapitel vor. Ferner gehen wir genauer auf das Problem „Spam“ ein und erläutern, warum wir eine Kombination aus Multiagentensystem und Textklassifizierungsalgorithmus zur Filterung von Spam einsetzen wollen.

Kapitel 3

Problembeschreibung

In diesem Kapitel wird die in dieser Arbeit behandelte Problemstellung diskutiert. Wir werden im ersten Abschnitt intensiv auf das Spamproblem eingehen und erläutern, warum wir ein verteiltes Multiagentensystem für ein geeignetes Mittel zur Bekämpfung von Spam halten. Danach zeigen wir im zweiten Abschnitt die wesentlichen Problemstellungen auf, die wir bei der Umsetzung des verteilten Spamfilters zu lösen haben. Im dritten Abschnitt geben wir einen Ausblick auf Anwendungsgebiete für das von uns entwickelte verteilte Spamfilternetzwerk.

3.1 Präzisierung des Spamproblems

Eines der größten Probleme der heutigen Zeit im Emailverkehr stellt die fehlende Möglichkeit zur sicheren Authentifikation von Emailservern und Absendern dar. Dadurch kann der wahre Absender vieler Emails gar nicht zurückverfolgt werden. Durch das Fehlen der Möglichkeit, den Absender von Emails eindeutig zu identifizieren, ist es bestimmten Personen möglich, Vorteile aus dieser Sicherheitslücke zu ziehen. Sie senden Emails an eine große Anzahl von Personen, die diese gar nicht empfangen wollen. Dies ist auch die Definition dieser Art von ungewollten Emails, die als Spam bezeichnet wird. Unter die Kategorie Spammails fallen auch ungewollte Werbemails und Emails dubioser Anbieter von Erotikseiten oder Kreditvermittler. Die Kosten für das Versenden der Spammails an Tausende von Emailadressen sind gering. Auf Grund der vernachlässigbaren Kosten rentiert sich Spammen schon bei geringen Antwortzahlen. Die Kosten für Personen, die pro Tag eine Vielzahl von Spammails aussortieren müssen, sind extrem hoch. Sobald man seine Emailadresse in irgendeiner Form im Internet preisgibt, läuft man Gefahr, auf die Liste der Spammer gesetzt zu werden. Spammer scannen Internetseiten oder Foren systematisch nach Emailadressen. Besonders wertvoll für Spammer sind dabei Mailinglisten, denn durch eine einzige Email können alle Teilnehmer dieser Liste erreicht werden. Die Listen mit überprüften Emailadressen werden an andere Spammer weiterverkauft, die diese dann für ihre eigenen Zwecke nutzen.

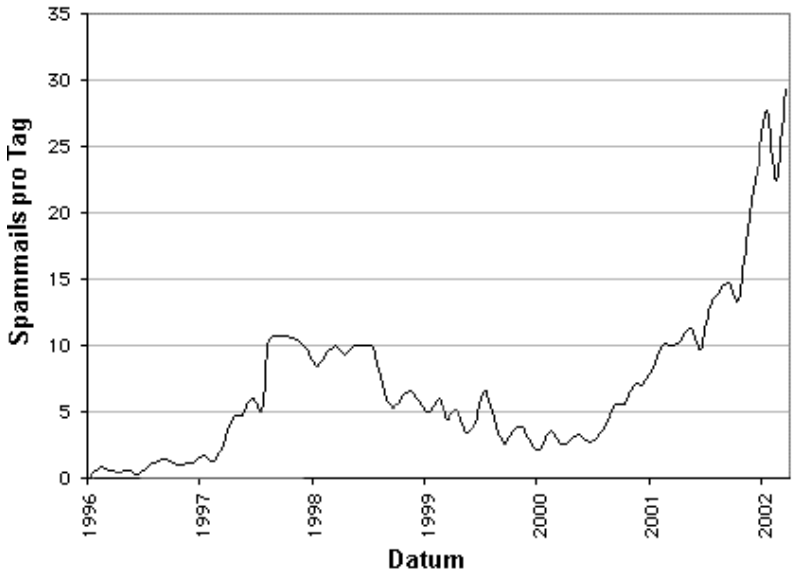

Abbildung 3.1: Anzahl von Spammails, die seit 1996 pro Tag auf einem Emailaccount eingegangen sind

Die einzigartige Tatsache, dass keine andere Art von Werbung den Versender so wenig und die Empfänger so viel kostet, lässt sich an einem Beispiel verdeutlichen. AOL erhielt bis zur richterlichen Verfügung pro Tag 1.8 Millionen Spammails von einem einzigen Spammer. Nimmt man an, dass der typische AOL-User nur 10 Sekunden zum Identifizieren und Aussortieren der Nachrichten benötigt, so werden nur bei AOL 5.000 Stunden pro Tag zum Entfernen von Spam gebraucht. Der zeitliche Aufwand zum Aussortieren von Spam gehört zu den Kosten, die der Empfänger von Spammails zu tragen hat. Experten schätzen, dass Spam bis zum Ende 2002 die Menge von ca. 25 Prozent aller Emails in der Mailbox des Benutzers ausmacht. Eine im November und Dezember 2002 durchgeführte repräsentative Umfrage der marketagent.com Marktforschungs GmbH [MARKETAGENT 2003] hat ergeben, dass 500 Millionen Spammails pro Woche die elektronischen Postkästen der deutschsprachigen Internetnutzer überfluten. Nur knapp sieben Prozent der Internetnutzer bleiben vor der Spamplage verschont. Weitere Statistiken bezeugen, dass die Anzahl von Spam in den letzten beiden Jahren exponentiell angestiegen ist. In Abb. 3.1 ist der Anstieg von Spam eines Benutzers mehrerer Emailaccounts seit 1996 [STATISTIC 2002] festgehalten. Die Anzahl empfangener Spammails dieses Benutzers steigt in den letzten Jahren stark an. Diese Statistik ist repräsentativ für die meisten Benutzer von Emailaccounts. Die Spammer werden durch fehlende globale Gesetze zur Eindämmung von Spam noch in ihren Bestrebungen bestärkt. So ist man auf eigene Mittel zur Filterung von Spam angewiesen. Der Betreiber des Mailservers kann diesen so konfigurieren, dass er Email

nach bestimmten Kriterien filtert und Email, die von bestimmten Domänen stammen, blockt. Zum Erreichen eines effektiven Schutzes vor Spam muss der Mailserver in regelmäßigen Intervallen rekonfiguriert werden. Ferner besteht die Gefahr, dass Emails gelöscht werden, die keinen Spam enthalten. Somit stellt die Konfiguration des Mailservers nur eine unbefriedigende Lösung des Spamproblems dar. Deshalb wurden zur Eindämmung der Flut von Spammails eine Vielzahl von Softwareprogrammen entwickelt (siehe Kapitel 2.3.5), die Spam erkennen und filtern. Bei der Filterung wird oft versucht, Spam durch die Anwendung von Regeln zu erkennen. Vertreter dieser Kategorie wie [ANTI-SPAM 2002] besitzen eine Datenbasis mit einem Regelwerk. Jedoch ist es schwierig, diese große Anzahl von Regeln an Wechsel im Inhalt der Spammails anzupassen und aktuell zu halten. Andere Spamfilter [RENNIE 2000, PANTEL und LIN 1998] arbeiten mit Algorithmen, die zur Textklassifizierung entwickelt wurden. Obwohl bestimmte Spamfilter eine hohe Zahl von Spammails erkennen und löschen können, arbeiten sie dennoch isoliert von anderen Spamfiltern. Deshalb können sie keine Informationen über Spam an andere Filter übermitteln, um ihr Wissen über Spam zu erweitern und neue Formen von Spam zu erkennen. Es existiert nur eine einzige Version eines kommerziellen verteilten Spamfilters [Vipul's Razor 2002], der Spaminformationen über ein Netzwerk verbreitet. Dort werden Spaminformationen jedoch zentral auf Servern gespeichert. Wir wollen einen Spamfilter entwickeln, der ohne zentralen Server auskommt und die Spaminformationen verteilt auf den Knoten eines P2P-Netzwerkes speichert.

3.2 Problemstellung

Wir wollen ein Netzwerk von Agenten zur Spamfilterung implementieren, die mit Hilfe von *peer-to-peer* Technologien intelligent Informationen untereinander austauschen können, um so ihre Klassifizierungsgenauigkeit bei der Erkennung von Spam zu verbessern. Die Agenten schließen sich zu einem Multiagentensystem zusammen und tauschen bei der Kommunikation miteinander Informationen über Spam aus. Dieses Agentensystem soll durch einen Algorithmus zur Textklassifizierung ergänzt werden. Damit das zu implementierende Spamfilternetzwerk auch effektiv Spam erkennen kann, befassen wir uns mit drei wesentlichen Fragestellungen. Zuerst muss der Agent Spaminformationen generieren und in eine komprimierte Form bringen, so dass die Informationen schnell zwischen den Agenten ausgetauscht werden können. Danach müssen wir uns überlegen, wie ein sinnvolles Zusammenspiel des Multiagentensystems mit dem Filter durch Textklassifizierung erreicht werden kann. Als dritten Punkt wollen wir die Kommunikation zwischen den einzelnen Agenten optimieren, d.h. Agenten sollen zur Einsparung von Kommunikationskosten nur die für sie relevanten Spaminformationen erhalten. Die Effektivität unseres Spamfilters werden wir anschließend experimentell evaluieren und seine Performanz bei der Identifizierung von Spam messen.

3.2.1 Spezifizierung der auszutauschenden Spaminformationen

Hat ein Agent eine Email als Spam identifiziert, so sendet er diese Information an andere Agenten des Netzwerkes. Während der Implementierung muss deshalb zuerst geklärt werden, in welcher Form Informationen über Spam am geschicktesten und schnellsten über das Netzwerk verbreitet werden können. Welche Daten soll ein Agent aus dem Inhalt der Email extrahieren, damit andere Agenten bei Erhalt der gleichen Spammail erkennen können, dass diese bereits von einem Agenten des Netzwerkes als Spam klassifiziert wurde? Wir sprechen von Gleichheit zweier Emails, wenn ihr Inhalt *(engl. content)* denselben Text aufweist. Obwohl gleiche Spammails aufgrund der ausgetauschten Informationen erkannt werden sollen, muss es unmöglich sein, vom Aufbau der Spaminformationen auf den ursprünglichen Inhalt der Spammail zu schließen. Wir müssen also eine Datenstruktur definieren, die *hohen Informationsgehalt* unter *Wahrung der Datensicherheit* bietet. Weiterhin sollen anhand der ausgetauschten Informationen auch Spammails mit ähnlichem Inhalt erkannt werden. Diese Form des Spam wird zum Überlisten der Mechanismen herkömmlicher Spamfilter in steigendem Maße von Spammern versendet.

3.2.2 Zusammenspiel von Multiagentensystem und Textklassifizierung

Die sinnvolle Kombination verschiedener Filter zur Identifizierung von Spammails soll die Erkennungsrate des Gesamtsystems gegenüber isolierten Spamfiltern erhöhen. Die Agenten des Spamfilternetzwerkes sollen auf bewährte Filtertechnologien wie der Erkennung von Spam durch Textklassifizierung zurückgreifen können. Wir werden daher verschiedenen Filtermethoden zur Erkennung von Spam vergleichen und diejenigen auswählen, die mit dem Multiagentensystem auch sinnvoll kombiniert werden können. Dabei soll ein Algorithmus zur Textklassifizierung in unserer Agentensystem eingebettet werden. Danach präzisieren wir die Funktionsweise der Agenten beim Analysieren der Email. Wir legen fest, in welcher Reihenfolge und Kombination die Erkennung von Spam mit Hilfe der ausgetauschten Informationen und mit der Unterstützung durch die Textklassifizierung durchgeführt wird.

3.2.3 Optimierung der Kommunikation durch Selbstorganisation

Ist das Spamfilternetzwerk aus Antispam-Agenten implementiert, wollen wir Mechanismen finden, um die Kommunikation zwischen den Agenten zu optimieren und damit Kommunikationskosten zu sparen. Informationen über Spam sollen nicht einfach an alle beliebigen Agenten des Spamfilternetzwerkes verschickt werden. Stattdessen sollen die Agenten nur Informationen erhalten, die für sie auch von Nutzen sind. Dabei verfolgen wir das Ziel, dass nur diejenigen Agenten Spaminformationen untereinander austauschen, die beispielsweise im selben Emailverteiler sind, d.h. die auch die gleichen Spammails erhalten. Die Agenten besitzen keine Informationen über den Aufbau

der Emaillisten von Spammern. Deshalb müssen Lösungen und Wege gefunden werden, wie sich Agenten, die demselben Verteiler angehören und somit dieselben Spammails erhalten, zusammenfinden können. Beim Gruppieren der Agenten durch Selbstorganisation wollen wir uns der Erkenntnisse aus dem Gebiet der Sozionik bedienen und die dort vorgestellten holonischen Organisationsformen verwenden. Agenten innerhalb einer holonischen Organisation sollen verstärkt Spaminformationen austauschen.

3.2.4 Experimentelle Evaluation

Nach Abschluss der Implementierung des Spamfilters müssen wir klären, in wieweit wir die in den letzten Abschnitten aufgelisteten Ziele erfüllt haben, die wir mit dem Spamfilternetzwerk erreichen wollen. Dazu werden wir die Effektivität des Spamfilternetzwerkes zur Erkennung von Spam anhand verschiedener Performanzmaße messen. Ferner zeigen wir, dass unser Spamfilternetzwerk Eigenschaften eines *small world network* besitzt. Diese Netzwerke zeichnen sich durch geringe durchschnittliche Pfadlänge mit wenigen Kanten aus, d.h. Informationen zwischen den Knoten werden effektiv und schnell weitergegeben. In Hinblick auf die Selbstorganisation der Agenten stellen wir Hypothesen auf, die sich mit dem Zusammenschluss der einzelnen Agenten zu einem Netzwerk von Organisationen befassen. Diese Hypothesen werden aufzeigen, dass die Verwendung einer holonischen Organisation aus dem Gebiet der Sozionik zur Einsparung von Kommunikationskosten innerhalb des Netzwerkes bei konstanter Effizienz der Spamfilter führt.

3.3 Praktische Anwendung

Die Konzepte zur verteilten Spamfilterung, die wir im Rahmen dieser Diplomarbeit vorstellen, lassen sich direkt auf die Praxis anwenden. So haben wir bereits ein Spamfilternetzwerk auf der Basis der FIPA-OS Agentenplattform implementiert, in dem die Klassifizierung von Text mit *support vector machines* und der Austausch von Spamdaten über das Netzwerk erfolgreich kombiniert sind. Je mehr Antispam-Agenten in dieses Netzwerk integriert sind, desto höher ist die Rate von Spammails, die durch den Austausch der Spaminformationen erkannt und gefiltert werden kann.

Die Methoden zum Zusammenschluss der Agenten durch Selbstorganisation und zur Bildung von Organisationsformen haben wir schon im Zusammenhang mit der Filterung von Spam getestet. Vorstellbar wäre ein verteiltes Spamfilternetz mit einer großen Zahl von Teilnehmern, in dem sich die Agenten der Mitarbeiter des DFKI zu einer eigenen Organisation zusammenschließen könnten. Die Agenten innerhalb der Organisation könnten verstärkt Spaminformationen austauschen und sich gegenseitig beim Filtervorgang unterstützen.

Kapitel 4

Funktionsweise des Antispam-Agenten und Architektur des Spamfilternetzwerkes

In diesem Kapitel stellen wir das von uns implementierte Spamfilternetzwerk vor, welches auf dem Zusammenspiel von *support vector machines* und einem Multiagentensystem beruht. Jedem Benutzer ist ein Antispam-Agent zugeordnet, welcher dessen Emails vom Mailserver lädt und mit Hilfe von mehreren ihm zu Verfügung stehenden Filtern analysiert. Der genaue Arbeitsablauf des Antispam-Agenten und die einzelnen Schritte beim Filtern von Spam finden sich ab dem ersten Abschnitt. Die Agenten erhöhen die Anzahl erkannter Spammails durch Kommunikation über ein P2P-Netzwerk *(engl. peer-to-peer)*. Darum werden wir in Abschnitt 4.8 auf den Verwendungszweck von P2P-Netzwerken eingehen. Danach zeigen wir, wie das P2P-Kommunikationsmodell sinnvoll auf das Netzwerk von Agenten zur Filterung von Spam übertragen werden kann. In Abschnitt 4.9 stellen wir die Agentenplattform vor, die die Kommunikation des Netzwerkes steuert und den Versand der Nachrichten zwischen den Agenten übernimmt. Die Knoten des P2P-Netzwerk sind die Antispam-Agenten, die für ihren Benutzer Emails vom Mailserver laden und analysieren. Den Arbeitsablauf des Antispam-Agenten stellen wir in den folgenden Abschnitten vor:

4.1 Funktionsweise des Antispam-Agenten

Die Aufgabe des Antispam-Agenten besteht in der sicheren Erkennung von Spammails und der Weiterleitung von Informationen über Spam an andere Agenten. Um diese Aufgabe zu erfüllen, kann der Agent auf drei Arten von Filtern zur Erkennung von Spammails zurückgreifen. Dabei nutzen die Antispam-Agenten den Vorteil, dass sie in ein Multiagentensystem integriert sind und über ein P2P-Netzwerk miteinander kommunizieren können. Durch Austausch von Informationen über Spam in Form von Hashwerten und der Klassifizierung der vom Mailserver geladenen Emails anhand der ausgetauschten Informationen können die Antispam-Agenten einen Teil der

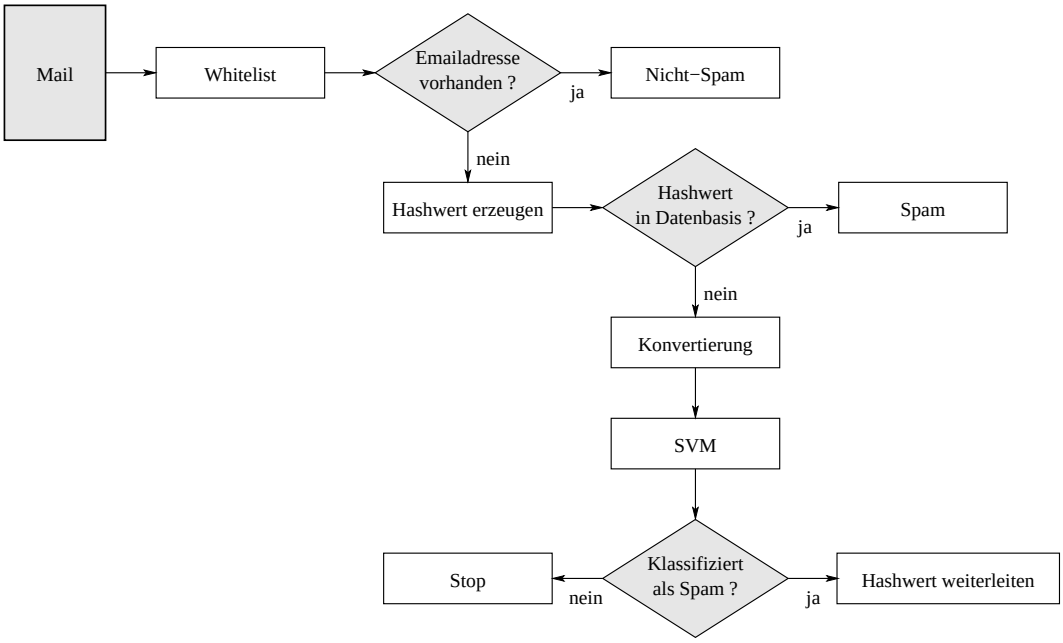

Abbildung 4.1: Reihenfolge der Arbeitsschritte des Antispam-Agenten

Spammails erkennen. Den dabei eingesetzten Filter bezeichnen wir als *Filter durch Hashwert-Vergleich*. Unterstützt werden die Agenten weiterhin durch den Textklassifizierungsalgorithmus *support vector machines* sowie durch einen Filter, der auf der Verwendung von *whitelists* basiert. Durch das Zusammenspiel der drei Filter und den Transfer von Informationen können die Antispam-Agenten eine höhere Klassifizierungsgenauigkeit bei der Erkennung von Spam erreichen, als Spamfilter, die isoliert und ohne Kommunikationsmöglichkeit allein mit SVM Spam erkennen sollen.

Wir wollen nun die verschiedenen Arbeitsschritte des Antispam-Agenten kurz vorstellen und in den nachfolgenden Abschnitten die einzelnen Arbeitsschritte näher erläutern. Die Reihenfolge der Arbeitsschritte ist in Abb. 4.1 dargestellt. Zuerst holt der Antispam-Agent die Emails des Benutzers von dessen Mailserver ab und speichert diese in seiner Datenbank. Dieser Vorgang wird im folgenden Abschnitt näher beschrieben. Danach wird jede Email durch die Verwendung der Filter klassifiziert. Zuerst durchlaufen die Emails den ersten Filter, die *whitelist*. Hierbei kann der Benutzer über die grafische Schnittstelle des Antispam-Agenten eine Liste von Emailadressen spezifizieren und speichern, die er für vertrauenswürdig hält. Alle Emails, die als Absender eine dieser vertrauenswürdigen Emailadressen enthalten *(engl. whitelist)*, werden als Nicht-Spam klassifiziert wie in Abschnitt 4.3 näher beschrieben. Emails, die nicht durch den *whitelist*-Filter klassifiziert wurden, werden danach auf die Klassifizierung durch den Filter durch Hashwert-Vergleich vorbereitet. Für jede Email wird ein Hashwert erzeugt. Wir definieren den erzeugten Hashwert folgendermaßen:

Definition 7 *Der Hashwert bildet den Inhalt einer Email in komprimierter Art und Weise auf eine 22-stellige Prüfsumme ab. Die Inhalte zweier Emails können anhand der für beide Emails generierten Hashwerte verglichen werden. Der Grad der Gleichheit der Emailinhalte wird durch Differenz der in den Hashwerten gespeicherten Buchstabenhäufigkeiten errechnet.*

Der Algorithmus zur Transformation des Textinhaltes der Email in einen Hashwert ist in Abschnitt 4.4 erklärt. Hat der Antispam-Agent für eine Email den korrespondierenden Hashwert erzeugt, so wird dieser Wert mit einer Liste von Hashwerten verglichen, die aus Spammails erzeugt und lokal in einer Datenbank gespeichert wurden. Diese Liste wird ständig durch Hashwerte aktualisiert, die von Agenten über das Netzwerk versendet werden. Der zweite Filter mit Klassifizierung durch Vergleich der Hashwerte erreicht seine Effektivität nur durch die Kommunikation des Antispam-Agenten mit den anderen Agenten des Netzwerkes. Der Filter durch Hashwert-Vergleich ist in der Lage, einen Teil der Spammails zu identifizieren. Stimmt der Hashwert einer zu klassifizierenden Email mit einem der Werte in der Datenbank überein, so wird die korrespondierende Email als Spam klassifiziert und gekennzeichnet. Emails, die der Filter durch Vergleich der Hashwerte nicht aussortiert, werden zur Verarbeitung durch den dritten und letzten Filter vorbereitet (siehe Abschnitt 4.5). Der dritte Filter klassifiziert die Emails durch Analyse des Textinhaltes und wird mit dem Namen *support vector machines* bezeichnet. Auf die Spamerkennung durch den *support vector machines* Algorithmus wird in Abschnitt 4.6 eingegangen. Klassifiziert der dritte Filter die Email als Spam, so sendet der Antispam-Agent den korrespondierenden Hashwert an alle Agenten des Netzwerkes. Mit Hilfe dieser Hashwerte können die Filter durch Hashwert-Vergleich anderer Agenten die gleiche Email im betreffenden Filterungsschritt als Spam identifizieren. Der Benutzer kann mit Hilfe der Schnittstelle des Antispam-Agenten den Klassifizierungsprozess jederzeit mitverfolgen. Außerdem bietet sich ihm die Möglichkeit, die vom Agenten getroffene Klassifizierungsentscheidung über die Benutzerschnittstelle nach seinen Wünschen zu korrigieren. Somit kann er auch bestimmen, welche Hashwerte von Spammails an andere Agenten des Spamfilternetzwerkes versendet werden.

Der Antispam-Agent erreicht eine hohe Spamerkennungsrate durch die geschickte Kombination der drei Filter *whitelist*, Filtern durch Hashwert-Vergleich und *support vector machines*. Die ohnehin hohe Klassifizierungsgenauigkeit des Filterns durch *support vector machines* wird durch die Kommunikation der Agenten des Multiagentensystems und den Austausch von Hashwerten weiter gesteigert, was wir im sechsten Kapitel experimentell nachweisen.

4.2 Herunterladen der Emails vom Mailserver

Um die Funktionalität des Antispam-Agenten nutzen zu können, muss dessen Benutzer über einen Emailaccount auf einem Mailserver verfügen. Für den Zugriff des

Antispam-Agenten auf den Mailserver benutzen wir das JavaMail^{TM} Interface [Java Mail 2000]. Dieses Interface kann mit Hilfe des POP- oder IMAP-Protokolls auf den Mailserver zugreifen und die Emails herunterladen. In der aktuellen Version des Antispam-Agenten ist nur der Zugriff per POP-Protokoll *(engl. post office protocol)* möglich. Das IMAP-Protokoll *(engl. internet message access protocol)* unterstützt darüber hinaus den Zugriff mehrerer Benutzer auf mehrere Mailordner des Mailservers, sofern dieser diese Optionen unterstützt. Der Zugriff per POP-Protokoll ist für das Herunterladen der Emails und die Demonstration unserer Implementierung ausreichend. Dazu kann der Benutzer über die Schnittstelle des Antispam-Agenten die für den Zugang zum Mailserver relevanten Daten in einer Maske eingeben und speichern. Zu diesen Zugangsdaten gehören die Adresse des Mailservers, sowie der Benutzername und das Passwort zur Authentifizierung. Über das JavaMail^{TM} Interface wird durch Drücken der Schalters „Mail abholen“[1] die Verbindung zum Server geöffnet und die dort liegenden Nachrichten temporär in einer Liste gespeichert. Der Benutzer kann sich über das Register „Emailübersicht“ eine Zusammenstellung aller Nachrichten mit Angabe von Absender, Empfänger und Betreff der jeweiligen Emails ansehen. Ist der Klassifizierungsalgorithmus beim Herunterladen aktiviert, so existieren in der Übersicht für jede Email des weiteren Angaben darüber, ob sie als Spam klassifiziert wurde. Die über die Schnittstelle des Antispam-Agenten angezeigte Liste der klassifizierten Emails ist in Abb. 4.2 zu sehen. Ist eine Email klassifiziert, so wird der Benutzer in der letzten Spalte der Tabelle darüber informiert, welcher Filter die Email klassifiziert hat. In den folgenden Abschnitten beschreiben wir die verschiedenen Arbeitsschritte des Antispam-Agenten bei der Klassifizierung einer Email im Detail.

4.3 Filterung durch eine Whitelist

Der erste Filter, den der Antispam-Agent zur Analyse von Emails einsetzt, ist der Filter durch eine *whitelist*. In dieser Liste kann der Benutzer über die grafische Schnittstelle des Antispam-Agenten die Emailadressen von vertrauenswürdigen Emailabsendern eingeben und speichern. Empfängt der Benutzer eine Email, deren Absenderadresse in der *whitelist* aufgeführt ist, so wird diese als Nicht-Spammail klassifiziert. In die *whitelist* werden benutzerspezifisch diejenigen Absenderadressen eingetragen, von denen der Benutzer Emails ohne Rücksicht auf deren Inhalt erhalten will. Durch die Verwendung der *whitelist* wird sichergestellt, dass Emails von vertrauenswürdigen Absendern nicht aufgrund ihres Inhaltes als Spam klassifiziert werden. Ist eine Email aufgrund der *whitelist* als Nicht-Spam klassifiziert, so wird ihr Inhalt nicht weiter mit Hilfe der beiden anderen Filter überprüft. Daher sollten die Einträge in der *whitelist* mit Bedacht gewählt werden.

[1] oder auch optional beim Start des Antispam-Agenten

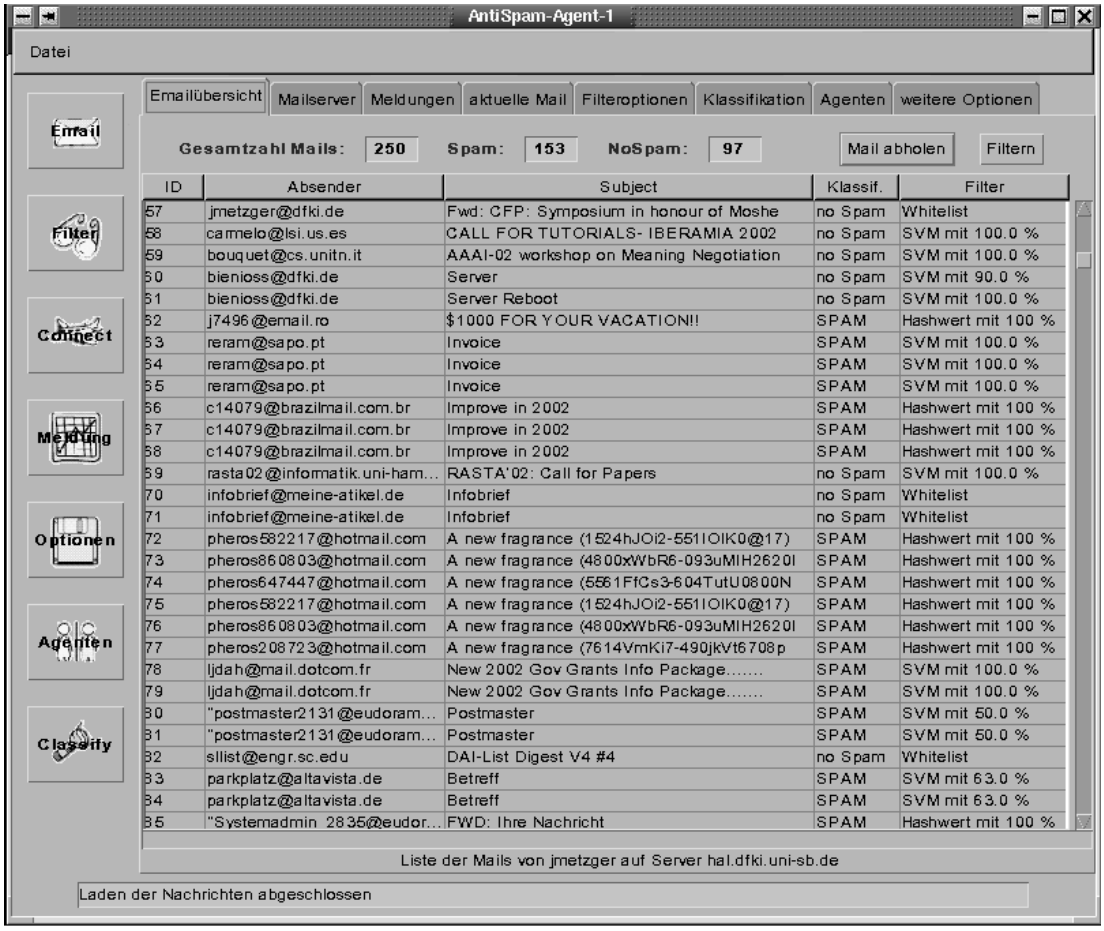

Abbildung 4.2: Schnittstelle des Antispam-Agenten mit Liste klassifizierter Emails

4.4 Erzeugung von Hashwerten

Der zweite Filter, den der Antispam-Agent bei der Analyse der Emails einsetzt, ist der Filter durch Hashwert-Vergleich. Aus dem Inhalt der Emails werden Hashwerte erzeugt und im Netzwerk von Antispam-Agenten verteilt. Der Hashwert bildet den Inhalt der Email in komprimierter Form ab. Im ersten Abschnitt werden wir zeigen, wie man mit Hilfe des *secure hash algorithm* [SH-STANDARD 1995] aus dem Inhalt von Emails einen Wert erzeugen kann, der für den Versand im Netzwerk geeignet wäre. Weiterhin werden wir auf die Nachteile der durch dieses Verfahren generierten Werte eingehen. Im zweiten Abschnitt stellen wir unseren eigenen Algorithmus zur Generierung von Hashwerten vor, mit denen es auch möglich ist, ähnliche Spammails zu erkennen. Wir demonstrieren im dritten Abschnitt den Algorithmus zur Erzeugung von Hashwerten aus dem Inhalt einer Email an einem Beispiel, bevor wir im letzten Abschnitt des Kapitels die Vor- und Nachteile der Verwendung der durch den Algorithmus erzeugten Hashwerte gegeneinander aufwiegen.

4.4.1 Generierung von Hashwerten durch den Secure Hash Algorithm

Eines der wesentlichen Ziele des Spamfilternetzwerkes besteht in der effektiven und schnellen Verteilung von Spaminformationen unter den Antispam-Agenten. Mit Hilfe dieser Informationen soll es jedem benevolenten Agenten möglich sein, einen Teil der Spammails ohne den Klassifizierungsalgorithmus SVM als solche zu erkennen. Dieser Filter durch Vergleich der Hashwerte soll die Klassifizierungsgenauigkeit des Antispam-Agenten erhöhen. Wie lassen sich nun aber Informationen über Spammails von einem Agenten X anfertigen und versenden, so dass ein Agent Y bei Erhalt der gleichen Spammail diese direkt erkennen kann? Eine triviale Möglichkeit wäre der Versand des kompletten Inhalts der Spammail von X nach Y, sobald der SVM-Algorithmus von X die Email als Spam erkannt hat. Dieser Text könnte von Agent Y gespeichert werden. Bei Erhalt der gleichen Spammail würde Y durch Vergleich des Textes mit den Texteinträgen in seiner Datenbank die Email als Spam erkennen. Die Nachteile dieser Methode liegen auf der Hand. Es wäre leicht möglich, die verschickten Inhalte abzufangen und auszulesen. Würde dieses Verfahren eine zufällige Nicht-Spammail falsch klassifizieren und versenden, so würde diese private Email anderen Benutzern des Netzwerkes zugänglich gemacht werden. Weiterhin wären die Kosten für den Versand der Inhalte der Emails enorm. Daher müssen Informationen, die den Inhalt entsprechender Spammails genau identifizieren, in komprimierter und verschlüsselter Form übertragen werden.

Mit Hilfe des *secure hash algorithm (SHA-1)* [SH-STANDARD 1995] ist es möglich, aus beliebigem Text der Länge $< 2^{64}$ Bit eine Identifizierungsnummer zu generieren. Diese hexadezimale SHA-Nummer besitzt eine Länge von 160 Zeichen und könnte effizient innerhalb des Spamfilternetzwerkes unter den Agenten ausgetauscht werden. Jeder Agent würde die im Netzwerk verschickten Nummern sammeln und in seiner Datenbank in Form einer Liste von SHA-Nummern speichern. Holt der Agent nun eine Email ab, so würde er daraus eine SHA-Nummer erzeugen und diese mit den Nummern seiner Datenbank vergleichen. Bei Übereinstimmung der erzeugten Nummer mit einem Eintrag aus der Datenbank wäre die Email als Spam identifiziert ohne Verwendung eines Textklassifizierers.

Ein gravierender Nachteil der SHA-Nummern liegt darin, dass der SHA-Algorithmus für zwei Emails, die sich nur durch ein einziges Zeichen voneinander unterscheiden, unterschiedliche SHA-Nummern erzeugt. Emails mit ähnlichem Inhalt können also anhand des SHA-Nummernvergleichs nicht identifiziert werden. Diese Eigenschaft der SHA-Nummern verhindert aber gerade die Erkennung bestimmter Arten von Spammails. So besitzen Spammer die Möglichkeit, den Empfängernamen des Benutzers an verschiedenen Stellen in den Inhalt der Email einzufügen. Durch das Einbinden benutzerspezifischer Informationen unterscheiden sich die Inhalte der Spammails nur geringfügig voneinander. Der *secure hash algorithm* erzeugt für jeden Benutzer jedoch

eine andere SHA-Nummer. Die Antispam-Agenten anderer Benutzer könnten die ähnliche Email mit demselben Spammer als Absender nicht anhand ihrer Datenbank identifizieren. Der Spamfilter mit SHA-Nummern wäre gegen diese Art von Spammails wirkungslos. Da diese Form von Spammails, die sich für die jeweiligen Empfänger nur geringfügig voneinander unterscheiden, in steigender Zahl verschickt werden, wollen wir auch für die Identifizierung ähnlicher Emails eine Lösung finden.

4.4.2 Generierung von Hashwerten durch Vergleich der Buchstabenhäufigkeiten

Damit auch ähnliche Emails miteinander verglichen werden können, erzeugen wir Emailinformationen mit einem von uns entwickelten Verfahren. Für eine bestimmte Anzahl repräsentativer Buchstaben des Alphabets messen wir die relative Buchstabenhäufigkeit BH ihres Auftretens im Inhalt der Email. In der aktuellen Konfiguration des Spamfilternetzwerkes ist die Menge der relevanten Buchstaben $M = \{a, e, h, i, l, n, o, r, s, t, u\}$. Sei Anz_{gesamt} die Gesamtzahl von Buchstaben im Inhalt der Email. Dann ergibt sich für die Buchstaben b_i, $i \in \{1, \ldots, |M|\}$ die Buchstabenhäufigkeit zu:

$$BH(b_i) = \frac{|b_i|}{Anz_{gesamt}} \cdot 100 \tag{4.1}$$

$BH(b_i)$ wird auf eine ganzzahlige natürliche Zahl $\in \{1, \ldots, 99\}$ gerundet. Einstellige Buchstabenhäufigkeiten werden durch eine führende Null ergänzt. Den Hashwert für die $|M|$ relevanten Buchstaben b_i erzeugen wird durch Aneinanderreihen der zweistelligen Buchstabenhäufigkeiten in einer festgelegten Reihenfolge. Dadurch erhalten wir eine Nummer mit $2|M|$ Stellen, die wir als Hashwert bezeichnen. Selbst wenn die Häufigkeiten aller Buchstaben des Alphabets in die Erzeugung des Hashwertes einfließen, besitzt dieser mit 52 Stellen eine Länge, die mit geringen Kommunikationskosten über das Netzwerk versendet werden kann.

Den Vergleich zweier Hashwerte führt der Antispam-Agent auf folgende Weise durch: Zuerst werden die zweistelligen Buchstabenhäufigkeiten aus den Hashwerten extrahiert. Danach wird aus den Häufigkeiten jedes Buchstabens der beiden Hashwerte die Differenzen gebildet und die erhaltenen $|M|$ Differenzwerte aufsummiert. Je größer der Wert dieser Summe SW ist, desto mehr unterscheiden sich die Häufigkeiten der Buchstaben in den Inhalten der zu den Hashwerten korrespondierenden Emails. Der Summenwert SW wird durch eine Funktion $f(SW)$ in einen Prozentwert überführt, der für den Benutzer des Antispam-Agenten eine höhere Aussagekraft besitzt als der Summenwert. Da sich selbst die Buchstabenhäufigkeiten ähnlicher Emails schon um wenige Prozentpunkte unterscheiden, muss die Funktion $f(SW)$ geringe Unterschiede in den Buchstabenhäufigkeiten der zu vergleichenden Hashwerte vernachlässigen und mit wachsendem Summenwert SW exponential ansteigen. Mit der Funktion $f(SW) = 100 - 1.25^{SW}$ errechnen wir den Unterschied zweier Hashwerte zu:

$$Differenz = f(SW) = 100 - 1.25^{SW} \tag{4.2}$$

Damit können wir nun den Vorgang des Erkennens von Spammail mit Hilfe von Hashwerten näher beleuchten. Der Antispam-Agent sammelt die Hashwerte, die von anderen Agenten im Netzwerk verteilt werden und speichert sie lokal in einer Liste. Sobald er nun eine Email vom Mailserver abgeholt hat, erzeugt er für dessen Inhalt nach dem oben beschriebenen Verfahren den korrespondierenden Hashwert. Diesen Wert vergleicht er nacheinander mit jedem Hashwert-Eintrag seiner lokalen Datenbank durch Addition der Buchstabenhäufigkeiten beider Hashwerte und Konvertierung des Summenwertes mit der Funktion $f(SW)$. Daraus errechnet er die Übereinstimmung beider Hashwerte und gibt über die Benutzerschnittstelle den Differenzwert aus, der dem Benutzer den Grad der Ähnlichkeit anhand eines Prozentwertes $\in \{0, \ldots, 100\}$ deutlich macht. Der Benutzer kann einen Schwellwert spezifizieren, bei dessen Überschreiten die beiden Hashwerte als identisch betrachtet werden und die entsprechende Email als Spam gekennzeichnet wird. Dabei wird der Benutzer über die Schnittstelle des Antispam-Agenten durch Anzeige der prozentualen Übereinstimmung informiert, in wie weit der Hashwert einer vom Server geladenen Email mit einem Spam-Hashwert aus der Datenbank übereinstimmt.

Durch Variation des Schwellwertes über einen Schieberegler der Benutzerschnittstelle ist es dem Anwender des Antispam-Agenten möglich, die Funktionsweise dieses Spamfilters durch Hashwert-Vergleich zu beeinflussen. Regelt er dem Schwellwert auf 100 Prozent hoch, so werden nur diejenigen Emails als Spam klassifiziert, deren Buchstabenhäufigkeiten absolut genau mit denen der in der Datenbank enthaltenen Hashwerte übereinstimmt, d.h. ähnliche Spammails werden auf diese Weise nicht erkannt. Wird der Schwellwert niedriger eingestellt, so werden zwar ähnliche Spammails erkannt. Der niedrigere Schwellwert bewirkt aber, dass eine zu einem Hashwert gehörende Email schon dann als Spam klassifiziert wird, wenn ihr Hashwert ungefähr mit einem Eintrag der Datenbank übereinstimmt. Auch wenn sich die Werte beider Hashwerte an wenigen Stellen unterscheiden, werden die korrespondierenden Emails als identisch betrachtet. Wählt man den Schwellwert zu gering[2], so wächst die Gefahr, dass der Filter zwei Hashwerte als übereinstimmend kennzeichnet, die nicht aus der gleichen Spammail generiert wurden, sondern zufällig ähnliche Buchstabenhäufigkeiten aufweisen. Die Anzahl der Einträge in der Datenbank des Antispam-Agenten ist auf 100 Einträge begrenzt. Da der Agent Hashwerte, die ihm zugesendet werden, mit einem Zeitstempel versieht, kann er bei Erhalt neuer Hashwerte die ältesten aus seiner Datenbank entfernen. Der Schwellwert, bei dessen Überschreiten die zu den Hashwerten gehörenden Emails als identisch betrachtet werden, ist standardmäßig auf 95 Prozent gesetzt. Dem Benutzer bleibt es vorbehalten, den Schwellwert nach seinen Wünschen abzuändern.

[2] Schwellwert unter 90 Prozent Übereinstimmung

4.4.3 Vergleich der Übereinstimmung zweier Hashwerte am Beispiel

Bevor eine Email durch Hashwert-Vergleich klassifiziert werden kann, muss zuerst der zu ihrem Inhalt korrespondierende Hashwert erzeugt werden. Sei der Inhalt der Email gegeben durch den Text „Susi singt“ und sei $M = \{s, i, n\}$ die Menge der zur Klassifizierung relevanten Buchstaben des Alphabets. Die Gesamtzahl von Buchstaben im Inhalt der Email beträgt $Anz_{gesamt} = 9$. Die Buchstabenhäufigkeiten von $M = \{s, i, n\}$ betragen bei Vernachlässigung von Groß- und Kleinschreibung:

$Buchstabe\ b_i$	$BH(b_i)$
s	$3/9 = 33\%$
i	$2/9 = 22\%$
n	$1/9 = 11\%$

Tabelle 4.1: Aus dem Text der Email „Susi singt“ errechnete Buchstabenhäufigkeiten für $M = \{s, i, n\}$

Durch Aneinanderreihen der Buchstabenhäufigkeiten ergibt sich für den Text der Hashwert$_1$ = 332211. Nun wird dieser nacheinander mit den maximal 100 Einträgen von Hashwerte der Datenbank verglichen und die Übereinstimmung errechnet. Sei Hashwert$_2$ = 312309 ein beliebiger Hashwert dieser Datenbank. Dann errechnen sich die Differenzen zwischen den Buchstabenhäufigkeiten beider Hashwerte zu:

Hashwert$_1$	33	22	11	
Hashwert$_2$	31	23	09	
Differenz	02	01	02	Gesamt: 5

Das Ergebnis der Aufsummierung der drei Differenzwerte ergibt den Summenwert $SW = 5$. Der Wert von SW wird nun in die Funktion $f(SW)$ eingesetzt und die Übereinstimmung zwischen Hashwert$_1$ und Hashwert$_2$ beträgt somit:

$$\begin{aligned} Differenz = \quad f(SW) \quad &= 100 - 1.25^{SW} \\ = \quad f(5) \quad &= 100 - 1.25^5 \approx 97\% \end{aligned}$$

Der Wert der Übereinstimmung von 97 Prozent überschreitet den standardmäßigen Schwellwert von 95 Prozent. Die Email, aus der Hashwert$_1$ erzeugt wurde, wird durch den Filter durch Hashwert-Vergleich als Spam klassifiziert.

4.4.4 Vor- und Nachteile der Klassifizierung durch Vergleich der Hashwerte

Nachdem wir den Algorithmus zur Generierung und zum Vergleich von Hashwerten vorgestellt haben, fassen wir nun die Vor- und Nachteile dieses Verfahrens zusammen:

Vorteile:

- Wurde eine Email von einem Antispam-Agenten als Spam identifiziert und der entsprechende Hashwert an alle Agenten des Spamfilternetzwerkes gesendet, so können die anderen Agenten die gleiche Spammail durch Hashwert-Vergleich sicher identifizieren, wenn sie die gleiche Spammail nach Erhalt des Hashwertes von ihrem Mailserver herunterladen.

- Durch Vergleich von Buchstabenhäufigkeiten können auch Spammails mit ähnlichem Inhalt erkannt werden. Dabei kann der Benutzer den Schwellwert individuell einstellen, bei dessen Überschreiten der Antispam-Agent zwei Hashwerte als identisch betrachtet.

- Beim Versand der Hashwerte innerhalb des Netzwerkes ist die Datensicherheit gewährleistet. Es ist nicht möglich, aus dem Aufbau des Hashwertes und damit aus den Buchstabenhäufigkeiten auf den ursprünglichen Inhalt der Email zu schließen.

Nachteile:

- Es ist nicht auszuschließen, dass sich die Buchstabenhäufigkeiten zweier aus verschiedenen Texten generierter Hashwerte soweit ähneln, dass beim Vergleich der entsprechenden Hashwerte der Schwellwert von 95 Prozent Übereinstimmung überschritten wird und die Hashwerte als identisch gekennzeichnet werden. Seien Hashwert_1 und Hashwert_2 diese identischen Hashwerte. Dann könnte eine zu klassifizierende Nicht-Spammail mit zugehörigem Hashwert_1 anhand der Übereinstimmung mit dem Spam-Hashwert_2 aus der Datenbank irrtümlicherweise als Spam klassifiziert werden. In Kapitel 7.3 verweisen wir deshalb auf Verfahren, die die Gefahr der Generierung gleicher Hashwerte aus verschiedenen Emailinhalten minimieren. Zur Untersuchung der Eigenschaften des Netzwerkes aus Antispam-Agenten genügt hierzu schon eine Verkleinerung der Hashwert-Datenbank auf maximal 100 Einträge.

- Die Antispam-Agenten schöpfen nur Nutzen aus den Spam-Hashwerten, wenn sie diese rechtzeitig erhalten, also bevor sie die entsprechende Spammail von ihrem Emailserver laden und klassifizieren. Die Hashwerte müssen also schnell und effizient über das Netzwerk verteilt werden.

Wird eine vom Mailserver geladene Email nicht durch Hashwert-Vergleich als Spam klassifiziert, so wird ihr Inhalt in einen Vektor konvertiert und an den Filter durch *support vector machines* weitergeleitet. Diesen Vorgang beschrieben wir im nächsten Abschnitt.

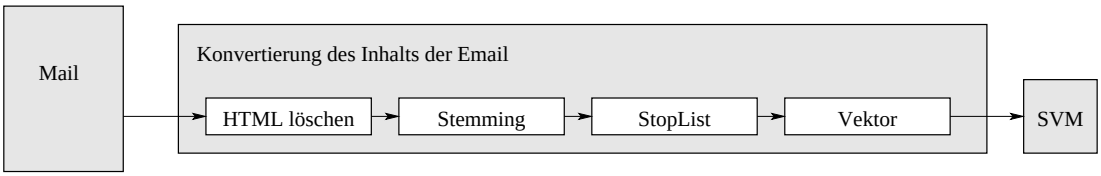

Abbildung 4.3: Schritte bei der Konvertierung des Inhaltes der Emails zu einem Vektor

4.5 Konvertierung des Inhalts der Email

Bevor der Inhalt der Email mit Hilfe des dritten Filters, des Textklassifizierungsalgorithmus SVM, analysiert werden kann, muss der darin enthaltene Text die Konvertierungsschritte durchlaufen, die in Abb. 4.3 dargestellt sind. Der Inhalt der Email muss in eine Form gebracht werden, die vom Textklassifizierungsalgorithmus verarbeitet werden kann. Dazu werden als erstes die HTML-Tags aus dem Text entfernt. Aus dem verbleibenden Text werden alle Wörter extrahiert und in eine Liste eingetragen. Danach wird die Anzahl unterschiedlicher Wörter reduziert, indem alle Suffixe abgetrennt und die Wörter auf ihren Wortstamm reduziert werden *(engl. stemming)*. Optional können Wörter aus dem Text herausfiltert werden, die im Sprachgebrauch oft vorkommen und deren Vorhandensein nicht auf den Inhalt des Textes schließen lässt (mit Hilfe einer *stoplist*). Ferner wird im letzten Abschnitt dieses Kapitels erklärt, wie der bearbeitete Text in einen Vektor umgewandelt wird, welcher durch SVM klassifiziert werden kann. Die genannten Konvertierungsschritte stellen wir nun in den folgenden Abschnitten vor:

4.5.1 Löschen der HTML-Tags

Mit vielen Emailprogrammen lassen sich Nachrichten generieren, die Textformatierungen in Form von HTML-Tags enthalten. Mit Hilfe dieser Tags ist es möglich, die Schrift in verschiedenen Farben und Formen darzustellen oder Bilder direkt in die Email einzufügen. Für die Analyse des eigentlichen Inhalts der Nachricht spielen die HTML-Formatierungen keine Rolle. Ferner können diese Formatierungen von Spammern dazu eingesetzt werden, für den Benutzer unsichtbare Informationen in die Email einzufügen. Deshalb entfernt der Antispam-Agent im ersten Konvertierungsschritt alle HTML-Tags aus dem Inhalt der Email. Die Tags werden anhand ihres Aufbaus mit öffnender und schließender Klammer erkannt und alle Informationen zwischen den Klammern sowie die Klammern selbst werden aus dem Text gelöscht. Der HTML-bereinigte Inhalt der Email wird über die Schnittstelle des Antispam-Agenten in einem Textfenster unterhalb des ursprünglichen Inhaltes der Email dargestellt und ist somit direkt mit dem Inhalt der Originalnachricht vergleichbar.

4.5.2 Erstellung einer Wörterliste

Zur Klassifizierung des Inhalts der Email werden nur die darin enthaltenen Wörter benötigt. Deshalb werden im zweiten Konvertierungsschritt alle Sonderzeichen und Zahlen anhand ihres ASCII-Codes erkannt und aus dem HTML-bereinigten Text gelöscht. Weiterhin wird der Textinhalt nach Email- und Internetadressen durchsucht und diese ebenfalls aus dem Text entfernt. Zum Schluss werden alle Wörter der Länge eins aus dem Text gefiltert. Damit werden auch durch Leerzeichen getrennte Buchstabenkombinationen wie „M O N E Y“ aus dem Inhalt der Email entfernt. Der auf diese Weise bereinigte Text wird sequentiell gescannt und die verbliebenen Wörter nacheinander in eine Wörterliste *(engl. bag of words)* eingetragen[3]. Mehrfach vorkommende Wörter werden in der Liste auch mehrfach abgespeichert. Über die Schnittstelle des Agenten kann der Benutzer die Wörterliste durch Auswahl der Register „Filteroptionen, Bag of Words“ einsehen.

4.5.3 Stemming der Wörterliste

Die Anzahl der Wörter, die bei der Klassifizierung berücksichtigt werden müssen, lässt sich durch verschiedene Verfahren verringern. Normalerweise liegen die Wörter in einem Text in vielen morphologischen Varianten vor. Wir machen uns die Tatsache zu Nutze, dass wir zur Reduzierung des Wortschatzes die unterschiedlichen morphologischen Varianten eines Wortes zu einer einzigen repräsentativen Stammform verschmelzen können. Deshalb trennen wir die Suffixe der Wörter ab, ohne dass sie ihre Bedeutung verlieren. Ein Verfahren, welches diese Aufgabe erledigt, ist der von Martin Porter entwickelte Stemmer [PORTER 1980]. Der *Porter Stemmer* lässt sich auf englische Wörter anwenden. Jedes durch diesen Algorithmus bearbeitete Wort durchläuft fünf Runden. In jeder Runde wird getestet, ob bestimmte Regeln auf das Wort angewendet werden können. Ist dies der Fall, so wird der entsprechende Suffix vom Wortstamm abgetrennt. Dabei ist es für die Textklassifizierung irrelevant, ob der entstehende Term ein sinnvolles Wort ergibt. So formt der Porter Stemmer die Wörter „introduction“ und „introduced“ in die passende Stammform „introduc“ um. Durch die Verwendung des Porter Stemmers wird das Vokabular, welches zur Darstellung des Inhalts der Emails benötigt wird, erheblich reduziert. Wir haben die Javaversion des Porter Stemmers in den Filtermechanismus des Antispam-Agenten eingebunden. Die Liste der gestemmten Wörter kann über die Benutzerschnittstelle des Antispam-Agenten durch Auswahl der Register „Filteroptionen, Bag of Words“ aufgerufen werden.

4.5.4 Filtern durch die Liste genereller Wörter

Die Anzahl der Wörter der Wörterliste kann weiter verringert werden, indem generelle Wörter ohne spezifische Bedeutung aus der Liste entfernt werden. Die Wörter der Wörterliste werden mit der Liste der generellen Wörter *(engl. stoplist)* verglichen. Bei

[3]Die Reihenfolge der Wörter in der Liste spielt für den Klassifizierungsalgorithmus keine Rolle.

Übereinstimmung wird das entsprechende Wort aus der Wörterliste entfernt. Generelle Wörter ohne Bedeutung sind im englischen beispielsweise „the“, „of“ oder „on“. Die *stoplist* unseres Antispam-Agenten enthält insgesamt 20 Wörter.

Die Verwendung der *stoplist* zur Reduzierung des Vokabulars ist in der Literatur umstritten. So kommen etwa VAPNIK und WU [1999] zum Schluss, dass die *stoplist* zur Textklassifizierung nicht eingesetzt werden sollte. Sie begründen dies damit, dass es nicht offensichtlich ist, welche Wörter in die *stoplist* eingefügt werden sollen. So ist es sinnvoll, Wörter mit trivialer Bedeutung wie „a“ und „an“ in die *stoplist* einzufügen. Aber schon die Entscheidung, ob das Wort „now“ in die *stoplist* aufgenommen werden soll, hängt von der Klassifizierungsaufgabe ab. So könnte diesem Wort bei der Klassifizierung von Texten in die Kategorien „Texte von heute“ und „Texte von gestern“ eine zentrale Rolle zufallen. Durch Einfügen von „now“ in die *stoplist* würde dieses Wort aus dem Textinhalt herausgefiltert und die Aussagekraft des Textes würde verringert werden. Nach VAPNIK und WU [1999] soll die Entscheidung, welche Wörter bei der Klassifizierung eine Rolle spielen, dem Textklassifizierungsalgorithmus überlassen werden. Deshalb kann der Konvertierungsschritt des Löschens von Wörtern aus der *stoplist* über die Benutzerschnittstelle des Antispam-Agenten optional deaktiviert werden.

4.5.5 Textrepräsentation als Attributvektor

Der Attributvektor spiegelt das Vorkommen 500 gestemmter Wörter eines von uns generierten Vokabulars im Inhalt der Email wieder. Dieses Vokabular erzeugen wir aus einem Korpus von 1000 Emails, die wir von einem Verteiler des DFKI über den Zeitraum eines halben Jahres gesammelt haben. Alle Wörter der 1000 Emails werden durch den Porter Stemmer auf ihre Stammform gebracht. Dadurch verkleinern wir die bei der Klassifizierung zu berücksichtigende Menge verschiedener Wörter. Die gestemmten Wörter der 1000 Emails werden nacheinander in eine Liste eingetragen. Wir zählen die Vorkommen der in der Liste enthaltenen gestemmten Wörter und sortieren diese nach ihrer Häufigkeit in eine Vokabularliste ein. Alle gestemmten Wörter bis auf die 500 gestemmten Wörter mit dem größten Vorkommen werden aus der Vokabularliste entfernt. Nur das Vorkommen dieser 500 ausgewählten Wörter der Vokabularliste wird bei der Klassifizierung durch SVM berücksichtigt. Damit besteht der Attributvektor jeder Email aus 500 Werten, die dem Auftreten der gestemmten Wörter im Inhalt der Emails entsprechen. Die Vokabularliste mit den 500 gestemmten Wörtern ist im Anhang dieser Diplomarbeit aufgeführt.

Nachdem wir beschrieben haben, mit welchem Verfahren die Vokabularliste der 500 gestemmten Wörter generiert wird, können wir den Vorgang der Konvertierung des Inhaltes einer Email in den korrespondierenden Attributvektor beschreiben: Nachdem die in den letzten Abschnitten erläuterten Konvertierungsverfahren die HTML-Formatierungen aus dem Textinhalt der Email entfernt, die Suffixe der Wörter ab-

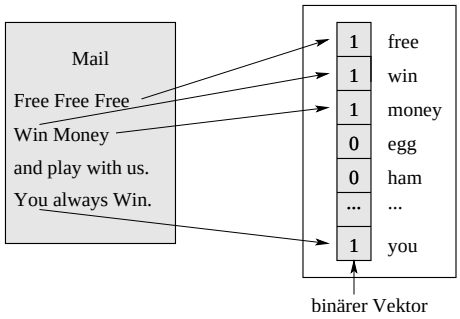

Abbildung 4.4: Abbildung des Inhalts einer Email auf einen binären Vektor

getrennt und optional generelle Wörter anhand der *stoplist* gelöscht haben, liegt das Vokabular der Email als Liste von gestemmten Wörtern in Form einer Wörterliste vor. Diese Wörter werden nun in den Attributvektor umgewandelt. Zuerst werden alle Werte des Vektors mit null initialisiert. Ein Algorithmus prüft für jedes gestemmte Wort W der Wörterliste, ob W als Attribut des Vektors vorhanden ist. Ist dies der Fall, so wird der korrespondierende Vektorwert auf eins gesetzt. Kommt ein Wort der Wörterliste mehrfach im Attributvektor vor, so wird der Vektorwert nicht weiter erhöht. Der Vektor enthält daher nur Werte der Menge $\{0, 1\}$ und wird deshalb auch als binärer Vektor bezeichnet. Findet der Algorithmus zu einem Wort der Wörterliste nicht das vergleichbare Wort des Vektors, so wird dieses verworfen. Der Vektor spiegelt also nur das Vorkommen der 500 gestemmten Wörter der Vokabularliste wieder. Abb. 4.4 zeigt die Überführung des Inhaltes einer Email in den korrespondierenden binären Vektor an einem Beispiel.

4.6 Klassifizierung mit Support Vector Machines

Nachdem der Inhalt der Email konvertiert wurde und die Häufigkeiten der 500 Wörter des Vokabulars in Form eines binären Vektors vorliegen, untersucht der Antispam-Agent den Vektor mit Hilfe eines Textklassifizierungsalgorithmus. In diesem Filter verwenden wir den Algorithmus *support vector machines (SVM)* zur Einteilung der Emails in die Kategorien Spam und Nicht-Spam. Auf die Implementierung des Textklassifizierungsalgorithmus in unserem Antispam-Agenten und die Bedienung von SVM wird im nächsten Abschnitt eingegangen. Danach fassen wir im zweiten Abschnitt die Vor- und Nachteile von *support vector machines* zusammen und testen im dritten Abschnitt die Klassifizierungsgenauigkeit des SVM-Algorithmus anhand einer von uns gesammelten Menge von Emails.

4.6.1 Implementierung des SVM-Algorithmus im Agenten

Nachdem der SVM-Algorithmus sich durch Vorteile wie seiner hohen Klassifizierungsgenauigkeit gegenüber anderer Algorithmen durchgesetzt hat (siehe Kap. 2.2.4),

wurde eine entsprechende Javaversion in den Quellcode des Antispam-Agenten eingebunden. Dabei handelt es sich um $jSVM$, ein Java Wrapper für Thorsten Joachims SVM^{LIGHT} [JOACHIMS 1998]. $jSVM$ kann in seiner Datenbasis mehrere Tausend *support vectors* speichern. Bei der Anwendung von $jSVM$ wird zwischen zwei Phasen unterschieden, der *Trainingsphase* und der *Klassifizierungsphase*.

Trainingsphase

Bevor der Klassifizierer seine Arbeit aufnehmen kann, baut er anhand von Trainingsvektoren ein Modell in Form einer Trainingsmatrix auf. Mit Hilfe dieses Modells ist eine spätere Analyse der Emails erst möglich. Installiert der Benutzer den Antispam-Agenten, so wird gleichzeitig eine Datei mit Trainingsdaten installiert. Bei dieser Datei handelt es sich um die Menge klassifizierter Vektoren von 50 Spammails und 50 Nicht-Spammails. Ein Vektor ist in folgender Form gespeichert:

$$Vektor =< C; w_1, w_2, \ldots, w_i > \quad mit \;\; w_i, C \in \{0, 1\}, \;\; i \in 1, \ldots, 500 \qquad (4.3)$$

Dabei stellen die Wörter $w_1, w_2, \ldots, w_i$ die Attributwerte des Vektors dar. Ein Attributwert hat den Wert eins, wenn das entsprechende Attribut im Inhalt der zum Vektor korrespondierenden Email vorkommt, ansonsten ist der Wert null. Der Klassifizierungswert C gibt an, ob die Email als Spam klassifiziert wurde ($C = 1$), oder als Nicht-Spam (C=0). Anhand der 100 Trainingsdatensätze kann schon beim ersten Start des Antispam-Agenten ein Spamklassifizierer erlernt werden. Über die Schnittstelle des Antispam-Agenten kann der Benutzer jederzeit aus den Vektoren der abgespeicherten Trainingsdaten einen neuen Klassifizierer erzeugen. Dies geschieht durch Drücken des Schalters „SVM-Modell Trainieren“ im Klassifizierungsmenü. Je mehr Vektoren dem Trainingsset zugefügt werden, desto höhere Erkennungsquoten können mit dem Klassifizierer erreicht werden. So wächst im Laufe der Zeit die Anzahl der Vektoren in der Datei der Trainingsdaten. Sobald ein Klassifizierer erlernt ist, wird dieser in Zukunft für die Klassifizierung der Emails eingesetzt. Das zugehörige Klassifizierungsmodell wird in einer zweiten Datei abgespeichert. Natürlich ist es dem Benutzer des Antispam-Agenten freigestellt, die Vektorwerte der Trainingsdatei zu löschen und eigene Werte einzufügen.

Klassifizierungsphase

Die Analyse des Inhalts einer Email in der Klassifizierungsphase gestaltet sich folgendermaßen: Nachdem der zu der Email korrespondierende binäre Vektor (wie in Kapitel 4.5.5 beschrieben) erzeugt ist, wird dieser an den SVM-Algorithmus übergeben. Dieser übernimmt die Entscheidung, ob die Email in die Kategorie Spam einzuordnen ist. Nach der Klassifizierung des Vektors gibt $jSVM$ einen Wahrscheinlichkeitswert $\in [0, \ldots, 1]$ zurück. Je höher der Wahrscheinlichkeitswert ist, desto sicherer hat der

Algorithmus die Email als Spam erkannt. Die von $jSVM$ durchgeführte Klassifizierung kann vom Benutzer korrigiert werden, indem er das Klassifizierungsergebnis abändert. Ist der Benutzer mit dem Ergebnis der Klassifizierung zufrieden, bzw. hat er dieses nach seinen Vorstellungen abgeändert, kann er den gerade klassifizierten Vektor zur Menge der Trainingsvektoren durch Drücken des Schalters „Vektor zur Trainingsmenge“ hinzufügen. Der Vektor wird zusammen mit dem Klassifizierungswert C gespeichert. Damit lässt sich die Menge der Trainingsvektoren beliebig vergrößern. Auf diese Weise abgespeicherte Vektoren werden beim Trainieren eines neuen Klassifizierers berücksichtigt.

Da der Benutzer die Klassifizierungsergebnisse beeinflussen kann, ist es möglich, die Arten von Emails zu spezifizieren, die $jSVM$ als Spam klassifiziert. Der Benutzer kann beispielsweise Werbemails als Nicht-Spam klassifizieren und die entsprechenden klassifizierten Vektoren als Trainingsdaten abspeichern. Der daraus erzeugte Klassifizierer wird anhand der Trainingsmatrix nun solche Werbemails ebenfalls als Nicht-Spam klassifizieren. Somit passt sich der Klassifizierungsalgorithmus des Antispam-Agenten nach entsprechendem Training beliebig an die Wünsche des Benutzers an.

4.6.2 Vorteile von Support Vector Machines

In diesem Abschnitt wollen wir die Vorteile von *support vector machines* noch einmal zusammenfassen:

- SVM schlagen in der Klassifizierungsgenauigkeit alle anderen Methoden. Bei der Klassifizierung von Spam erreichen SVM Genauigkeiten von über 95 % [VAPNIK und WU 1998].

- SVM minimieren die Anzahl der Klassifizierungsfehler *(engl. structural risk minimization)* durch Erzeugung der Hyperebene mit maximaler Trenngüte.

- SVM sind robust gegen *overfitting*. Mit *overfitting* wird ein Problem bezeichnet, welches im gesamten Bereich des Maschinellen Lernens auftritt. Beim Erzeugen des Klassifizierers aus den Trainingsdaten wird dabei ein Modell erzeugt, welches zu speziell auf die Trainingsdaten ausgerichtet ist, d.h. es besitzt keine Allgemeingültigkeit. Im Extremfall wird für jeden Datensatz der Trainingsdaten ein Satz von Klassifizierungsregeln erstellt, die nur für die in diesem Datensatz auftretenden Attributwerte eine Klassifizierung liefern und damit auf neue Datensätze gar nicht anwendbar sind. Das Problem des *overfitting* wird von SVM dadurch umgangen, dass die Lage der Hyperebene nur durch einen Bruchteil der Trainingsvektoren, den *support vectors*, festgelegt wird. Die Verteilung der Daten im Vektorraum spielt für die Erzeugung des Klassifizierers keine Rolle. Nur wenige Trainingsdaten sind für die Erstellung des Klassifizierers relevant, die Gefahr des *overfitting* ist minimiert.

4.6.3 Klassifizierungsgenauigkeit von SVM

Wir testen nun die Klassifizierungsgenauigkeit des *support vector machines* Algorithmus, den wir in den Antispam-Agenten eingebunden haben, an einem Korpus von 433 Emails. Diese Emails haben wir über den Zeitraum von einem halben Jahr von einem Verteiler des DFKI gesammelt. Wir trainieren einen SVM-Klassifizierer anhand von 240 Spammails und 98 Nicht-Spammails als Trainingsdaten. Der SVM-Algorithmus erzeugt daraus eine Klassifizierungsmatrix mit 54 *support vectors*. Danach überprüfen wir die Genauigkeit des erzeugten Klassifizierers anhand von Testdaten bestehend aus 59 Spammails und 36 Nicht-Spammails. In Tabelle 4.2 folgen die Ergebnisse der Klassifizierung:

	$richtig\ klassif.$	$falsch\ klassif.$
$Spam$	56	3
$Nicht - Spam$	34	2

Tabelle 4.2: Anteil der Spam- und Nicht-Spammails, die durch SVM richtig und falsch klassifiziert werden

Nach ANDROUTSOPOULOS et al. [2000] lässt sich der Klassifizierungsfehler f_{class} errechnen als Quotient der falsch klassifizierten Spam- und Nicht-Spammails und der Gesamtzahl klassifizierter Emails. Somit beträgt der Fehler des Antispam-Agenten beim Klassifizieren der 95 Emails:

$$f_{class} = \frac{3+2}{95} = 5.26\% \tag{4.4}$$

Ein Klassifizierungsfehler weit unter zehn Prozent stimmt mit den Ergebnissen überein, die andere Textklassifizierer mit SVM erreicht haben [VAPNIK und WU 1998, YANG und LIU 1999, NEUMANN und SCHMEIER 2002]. Die klassifizierte Menge von Emails ist zu gering, um eine fundierte Aussage über die Qualität von SVM innerhalb des Antispam-Agenten zu machen. Jedoch liegt der Klassifizierungsfehler unseres Spamfilters in einem Bereich, der es ihm ermöglicht, die meisten Emails der Testdaten des DFKI-Verteilers richtig in die Kategorien Spam und Nicht-Spam zu trennen.

4.7 Beschränkung des Antispam-Agenten auf drei Filter

Neben den von uns verwendeten Filtern existieren noch eine Vielzahl anderer Filter, die beim Erkennen von Spam eingesetzt werden könnten. Wir wollen anhand von zwei ausgewählten Filtermechanismen erklären, warum wir uns beim Filtern der Emails durch den Antispam-Agenten auf drei Filter beschränkt haben. Diese Filter sind der

Filter durch Hashwert-Vergleich, durch die Liste vertrauenswürdiger Emailadressen (*whitelist*) und der Filter durch den Textklassifizierer *support vector machines*.

- **Filtern durch eine Blacklist**
  Die *blacklist* ist eine Liste von Emailadressen und damit mit der von uns verwendeten *whitelist* vergleichbar. In die *blacklist* werden alle Emailabsender aufgenommen, die Spammails versenden. Da sich die Adressen der Spammer ständig ändern, muss diese Liste ständig gewartet und aktualisiert werden. Ferner ist es den Spammern möglich, jede beliebige Emailadresse als Absender einer Spammail anzugeben. Daher wird die Spammerliste durch Emailadressen aufgebläht, die nur einmalig zur Versendung von Spam benutzt wurden. Aufgrund dieser Nachteile wurde eine *blacklist* nicht in den Antispam-Agenten eingefügt.

- **Filtern durch Schlüsselwörter**
  Da der Textklassifizierungsalgorithmus bereits die Kombination von Wörtern im Inhalt der Email zur Identifizierung von Spam einsetzt, ist der Einsatz von Schlüsselwörtern unnötig.

4.8 Peer-To-Peer Netzwerke

Nachdem wir in den vorherigen Abschnitten auf die Funktionsweise des Antispam-Agenten eingegangen sind, befassen wir uns nun mit dem Informationsaustausch innerhalb des Spamfilternetzwerkes. Der Austausch von Spaminformationen findet zwischen den Antispam-Agenten als verteilten Knoten des P2P-Netzwerkes statt. Deshalb ist es sinnvoll, den Begriff der *peer-to-peer* Kommunikation näher zu beleuchten. Die Anwendungsgebiete von P2P-Technologien und Netzwerken werden wir im ersten Abschnitt vorstellen. Da die Benutzer der Spamfilter über dem Internet verteilt sind, bietet sich ein P2P-Netzwerk zum schnellen Versand von Informationen an. Der Aufbau des Spamfilternetzwerkes auf der Basis eines P2P-Modells wird im zweiten Abschnitt beschrieben. Im dritten Abschnitt gehen wir auf den Aufbau der Nachrichten ein, die zwischen den Antispam-Agenten über das P2P-Netzwerk versendet werden.

4.8.1 Anwendungsgebiete von P2P-Netzwerken

Peer-to-peer (P2P) Kommunikation ist eines der Schlagwörter der letzten Jahre. Bekannte Tauschbörsen [GNUTELLA 2001, NAPSTER 2002] nutzen das P2P-Kommunikationsmodell. *Peer-to-peer* beschreibt ein Kommunikationsmodell [SUKANEN 2002], in dem jede Partei dieselben Möglichkeiten besitzt und eine Kommunikationssitzung starten kann. Im Gegensatz zum P2P-Modell mit gleichberechtigten Parteien steht das Klient/Server- und das *Master/Slave*-Modell. Jeder P2P-Kommunikationsknoten besitzt sowohl Server- als auch Klientrechte. P2P-Netzwerke werden im Internet zum

Tausch von Dateien verwendet. Dazu müssen die verschiedenen Benutzer nur dasselbe Programm wie beispielsweise GNUTELLA [2002] oder NAPSTER [2002] verwenden, die sich in ihrer Architektur geringfügig unterscheiden. Während im Netzwerk von Napster ein zentraler Knoten alle Anfragen bearbeitet und weiterleitet (zentralisierter Ansatz), interagieren im Gnutellanetzwerk mehrere zentrale Knoten miteinander und leiten die Anfragen anderer Knoten weiter (semi-zentraler Ansatz). Auch Unternehmen nutzen P2P-Netzwerke und sparen somit Kosten für die Anschaffung und Wartung eines zentralen Servers. Rechen- und zeitintensive Anwendungen wie das SETI@Home-Projekt [HIPSCHMAN 2002] können aufgeteilt und parallel an verschiedenen verbundenen Computern berechnet werden. P2P-Modelle sind fehlertoleranter als Klient/Server-Modelle. Auch bieten sie Hackern keinen zentralen Angriffspunkt.

Agentenbasierte P2P-Netzwerke finden sich unter anderem bei SMITHSON und MOREAU [2002]. Agenten stellen Suchanfragen innerhalb des P2P-Netzwerkes, welche von Suchagenten verarbeitet und weitergeleitet werden. Die Eigenschaften des Gnutella-Netzwerkes werden im Multiagentensystem von LANGLEY et al. [2001] genutzt. Jeder Agent meldet sich im Netzwerk mit seinen Diensten an. Benötigt ein Agent einen bestimmten Dienst, so sendet er zuerst eine Anfrage an die Agenten im lokalen Netzwerk. Schlägt die lokale Suche fehl, so wird eine Suchanfrage an das P2P-Netzwerk gerichtet und geeignete Agenten gefunden.

4.8.2 Verwendung der Peer-To-Peer Architektur im Spamfilternetzwerk

Die P2P-Architektur eignet sich hervorragend für den Einsatz im Spamfilternetzwerk. Die über dem Internet verteilten Antispam-Agenten entsprechen den Knoten des P2P-Netzwerkes. Für jeden Benutzer eines Emailaccounts wird ein Antispam-Agent eingerichtet. Diese Agenten tauschen gleichberechtigt Informationen über Spammails untereinander aus. In das Agentennetzwerk können zu jeder Zeit Agenten ein- und austreten, ohne die Performanz des gesamten Systems zu beeinträchtigen. Obwohl es denkbar wäre, Informationen über Spam zentral auf einem Server zu speichern, bietet die P2P-Architektur große Vorteile gegenüber dem zentralistischen Ansatz. Zu diesen Vorteilen gehören die Robustheit des Gesamtsystems sowie größere Sicherheit gegen Angriffe auf das Spamfilternetzwerk. Ferner kommunizieren die Agenten über das P2P-Netzwerk und ermöglichen auf diese Weise den Nachrichtenaustausch innerhalb des Multiagentensystems. Der Informationsaustausch zwischen den Agenten bleibt selbst dann gewährleistet, wenn eine größere Zahl von Agenten *offline* und damit nicht mit dem Netzwerk verbunden ist. Agenten, die dem Netzwerk beitreten, erhalten durch Datenaustausch mit anderen Agenten schnell eine fundierte Wissensbasis und erhöhen so ihre Genauigkeit bei der Erkennung von Spam. Durch den ständigen Austausch von Informationen können die Agenten ihre Wissensbasis stets anhand der durch andere Agenten übermittelten Daten aktualisieren und vergrößern.

```
(inform
    :sender (X)
    :receiver (Y)
    :content
        <Liste der Hashwerte>
    :language 'SL0')
```

Abbildung 4.5: FIPA-ACL Nachricht zum Versand der Hashwerte von Agent X zu Agent Y

4.8.3 Versand der Hashwerte zwischen den Agenten

Der Informationsaustausch zwischen den verteilten Antispam-Agenten über das P2P-Netzwerk wird durch die FIPA-OS-Plattform übernommen, in der sie eingebettet sind. Die Übermittlung der Hashwerte von Agent X zu Agent Y geschieht durch FIPA-ACL Nachrichten, deren Form in Abb. 4.5 abgebildet ist. Die Angabe von Sender und Empfänger dient zur sicheren Übermittlung der Nachricht über die Agentenplattform. Agent X sendet den CA „inform“ mit der Liste seiner Hashwerte als Inhalt an den Agenten Y. Agent Y sendet nach Empfang der Nachricht eine Antwort an Agent X, die als Inhalt die Liste der Hashwerte von Y enthält. Somit haben Agent X und Y ihre Listen von Hashwerten ausgetauscht.

4.9 Kommunikation im Netzwerk aus Agenten

Um die P2P-Kommunikation zwischen den Antispam-Agenten zu gewährleisten, bedienen wir uns der in Kapitel 2.1.3 vorgestellten FIPA-OS-Agentenplattform. Diese Plattform stellt Dienste zur Registrierung und Deregistrierung der Agenten bereit. Ferner können die Antispam-Agenten über den *agent management service* (AMS) auf die Liste der momentan an der Plattform registrierten Agenten zugreifen. Sobald sich ein Agent X im Netzwerk angemeldet hat, greift er auf die Liste der aktiven Agenten zu und speichert diese lokal in einer Liste ab. Der Agent selbst erhält von der Agentenplattform eine eindeutige Identifizierungsnummer. Danach beginnt Agent X, in regelmäßigen Intervallen seine gesammelten Informationen über Spammails an alle anderen Agenten des Netzwerkes zu senden. Da den Nachrichten, die innerhalb der FIPA-OS-Plattform versendet werden, weitere Information wie Sender und Empfänger hinzugefügt werden, können die anderen Agenten des Netzwerkes den Absender der Spaminformationen feststellen. Befindet sich die Identifizierungsnummer von Agent X nicht in ihrer Datenbank von aktiven Agenten, so fügen sie die Identifizierungsnummer ihrer Liste von aktiven Agenten hinzu. Auf diese Weise ist sichergestellt, dass jeder Agent des Netzwerkes eine aktualisierte Liste von aktiven Agenten unterhält.

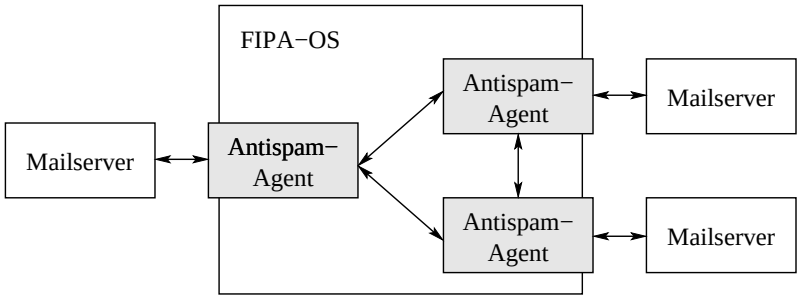

Abbildung 4.6: Architektur des Spamfilternetzwerkes

Nachdem die anderen Agenten des Netzwerkes die Identifizierungsnummer von X gespeichert haben, schicken sie ihre eigenen Spaminformationen an Agent X zurück. So kann Antispam-Agent X innerhalb kürzester Zeit auf die Spaminformationen aller anderen Agenten des Netzwerkes zugreifen. Die Einbettung der Antispam-Agenten in die FIPA-OS-Architektur wird in Abb. 4.6 verdeutlicht. Jeder Antispam-Agent kommuniziert mit dem vom Benutzer spezifizierten Mailserver sowie allen anderen Agenten des P2P-Netzwerkes.

Kapitel 5

Experimentalumgebung

Zur Evaluation des im vierten Kapitel vorgestellten Spamfilternetzwerkes mit den in FIPA-OS implementierten Antispam-Agenten haben wir eine Experimentalplattform entworfen, die den Austausch der Hashwerte zwischen den Agenten des Netzwerkes simuliert. Die Experimentalumgebung wurde entwickelt, um die Wechselwirkungen zwischen einer Vielzahl von Agenten beim Datenaustausch effektiv darzustellen und aufzuzeichnen. Die ursprüngliche FIPA-OS-Implementierung ist auf die Existenz realer Emailaccounts angewiesen und der Informationsaustausch zwischen mehr als fünf Antispam-Agenten kann nicht effektiv simuliert werden. In der Experimentalumgebung, die wir in diesem Kapitel vorstellen, ist es hingegen möglich, den Emailempfang und Datenaustausch von mehr als hundert Antispam-Agenten gleichzeitig zu simulieren. Dabei nehmen wir an, dass jeder Agent zu Beginn der Simulation bereits über eine Liste aller anderen Agenten verfügt. Des weiteren garantieren wir einen fehlerfreien Transport der Nachrichten zwischen den Agenten des Netzwerkes. Durch Implementierung von Vertrauen zwischen den Agenten und den Einsatz von Selbstorganisation reduzieren wir die Kommunikationskosten für den Informationsaustausch innerhalb der Experimentalumgebung. Der erste Abschnitt dieses Kapitels befasst sich mit dem Aufbau der Experimentalumgebung und zeigt die Aufgaben der darin enthaltenen Verteiler- und Antispam-Agenten. Im zweiten Abschnitt zeigen wir, wie der Datenversand zwischen den verschiedenen Agenten mit Hilfe von Selbstorganisation optimiert wird. Anschließend werden wichtige Eigenschaften der Experimentalumgebung im dritten Abschnitt zusammengefasst und erläutert.

5.1 Aufbau der Experimentalumgebung

Die Experimentalumgebung stellt eine rundenbasierte Simulation des Austausches der Hashwerte zwischen den Antispam-Agenten dar. Der Ablauf jeder Runde ist in einzelne Schritte eingeteilt. Zuerst generieren Verteileragenten Emails und senden sie an simulierte Emailaccounts der verschiedenen Gruppen von Antispam-Agenten. Der Vorgang des Emailversands ist im ersten Abschnitt detailliert beschrieben. Ist der Versand

abgeschlossen, so führen die Antispam-Agenten nacheinander ihre Aktionen aus. Dazu gehört als erstes das Abholen der Emails von ihrem Account. Daraufhin werden die abgeholten Emails klassifiziert und für jede empfangene Spammail ein Hashwert generiert. Diese Hashwerte werden lokal gespeichert. Ebenfalls werden die von anderen Agenten des Netzwerkes erhaltenen Hashwerte bearbeitet. Am Ende der Aktionsphase des Antispam-Agenten bestimmt dieser eine Anzahl von Agenten des Netzwerkes, denen er in der aktuellen Runde Hashwerte zuschickt. Die Hashwerte werden in Nachrichten verpackt. Der Aufbau der Nachrichten muss der in Kapitel 5.1.3 gemachten Spezifikation folgen. Nach Abschluss der Aktionsphase folgt die Wahlphase. In dieser Phase können sich Agenten durch Selbstorganisation zusammenschließen und darüber entscheiden, ob sie sich mit anderen Agenten verbinden wollen. Diese Entscheidungen werden auf der Basis von gegenseitigem Vertrauen der Antispam-Agenten ausgeführt. Die einzelnen Aktionen der Antispam-Agenten sind in Kapitel 5.1.2 erklärt. Am Schluss jeder Runde versendet die Experimentalumgebung die Nachrichten, die die Agenten erzeugt haben.

5.1.1 Verteileragenten

Den Verteileragenten der Experimentalumgebung kommt die Aufgabe zu, Emails automatisiert zu erzeugen und an die Accounts der Antispam-Agenten zu versenden. Bestimmte Verteileragenten können wir so konfigurieren, dass sie die Rolle von Spammern übernehmen, die an bestimmte Gruppen von Emailaccounts Spammails senden. Ferner können wir die Parameter anderer Verteileragenten so wählen, dass diese Nicht-Spammails generieren und an die Antispam-Agenten verteilen. Die von den Verteileragenten erzeugten Emails werden folgenderweise aufgebaut: Zuerst generiert der Verteiler eine fortlaufende eindeutige Emailnummer. Danach fügt er dieser Nummer die Information hinzu, ob die Email Spam enthält oder nicht. Der prozentuale Anteil von Spammails an den insgesamt vom Verteileragenten erzeugten Emails lässt sich für jeden Verteiler individuell festlegen. Die Antispam-Agenten sind zu Gruppen zusammengefasst. Agenten einer Gruppe empfangen die gleichen Emailnummern. Jeder Verteileragent sendet seine generierten Emails an die Emailaccounts einer bestimmten Gruppe von Antispam-Agenten. Die Agentengruppen, an die ein Verteiler sendet, werden mit Hilfe des Parameters $Gruppen$ spezifiziert. Somit können wir genau festlegen, welche Antispam-Agenten von welchen Verteilern Emails erhalten. Für jeden Verteileragenten legen wir die Häufigkeit fest, mit der er an die für ihn spezifizierten Agentengruppen sendet. Durch die Agentengruppen simulieren wir die von den Spammern erworbenen Listen mit Emailadressen.

Der Account des Antispam-Agenten wird durch eine Liste simuliert, in der die betreffenden Emailnummern mitsamt der Information, ob die betreffende Email Spam darstellt oder nicht, abgelegt werden. Abb. 5.1 zeigt eine ausgewählte Konfiguration der Experimentalumgebung mit vier Verteileragenten. Der erste Verteiler mit der Identifizierungsnummer null sendet mit einer Wahrscheinlichkeit von 80 Prozent pro Runde

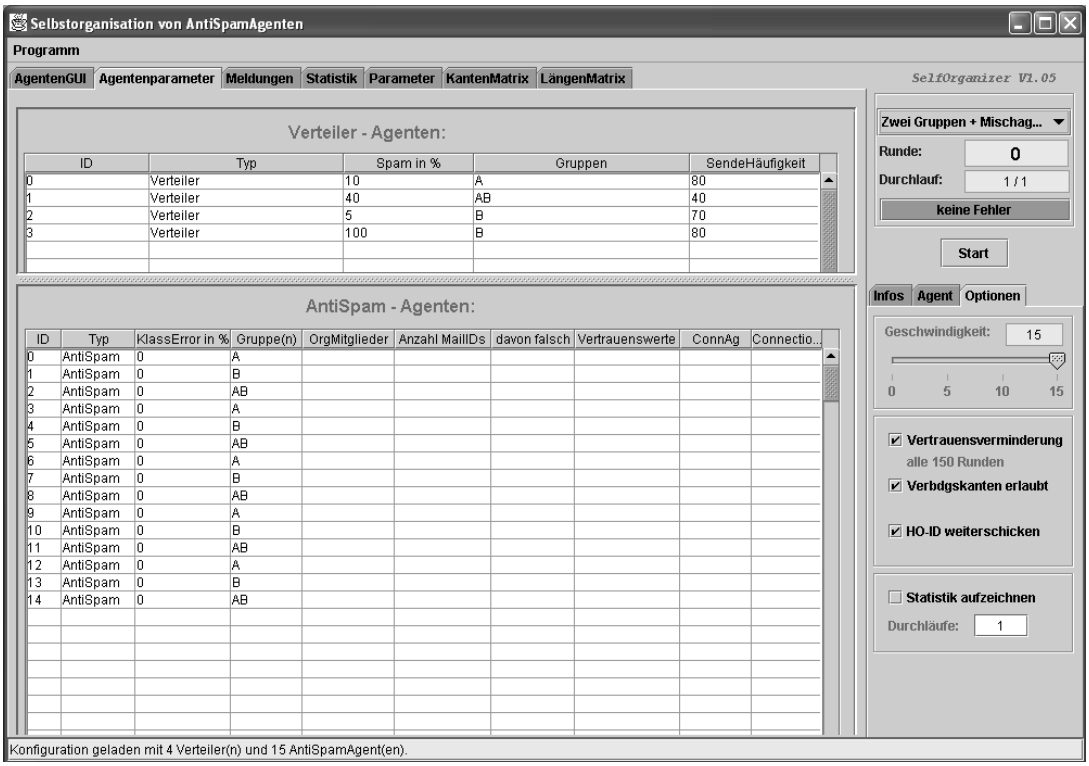

Abbildung 5.1: Benutzerschnittstelle des Antispam-Agenten mit der Übersicht der Konfigurationen von Verteiler- und Antispam-Agenten

eine generierte Email an alle Agenten der Gruppe A, dies sind die Antispam-Agenten mit den Identifizierungsnummern 0, 3, 6, 9 und 12. Die Email wird mit einer Wahrscheinlichkeit von 10 Prozent in die Kategorie Spam fallen, was der Verteiler durch Würfeln entscheidet. Der Emailversand eines Spammers wird durch Einsatz des Verteilers mit der Identifizierungsnummer drei simuliert. Die von ihm generierten Emails sind mit absoluter Sicherheit Spam und werden an alle Antispam-Agenten aus Gruppe B gesendet. Durch die Kombination mehrerer Verteileragenten lassen sich die an bestimmte Agentengruppen gesendeten Emails und der Anteil an Spammails beliebig variieren.

5.1.2 Antispam-Agenten

Die Antispam-Agenten sind die Aktoren des Spamfilternetzwerkes. Sie empfangen und bearbeiten Emails, die von den Verteileragenten versendet werden. Wie viele Emails ein Agent erhält, hängt von seiner *Gruppenzugehörigkeit* ab. Gehört ein Agent zur Gruppe A, so erhält er alle Emails, die von Verteileragenten verschickt werden, deren Parameter $Gruppen$ den Buchstaben A enthält. Ein Antispam-Agent, der gleichzeitig mehreren Gruppen angehört, empfängt auch alle an diese Agentengruppen verteilten Emails. Wir bezeichnen im Folgenden die Agenten mit X,Y und Z und die

Gruppen der Agenten mit A,B und C. Wollen wir ausdrücken, dass ein Agent zu einer bestimmten Menge von Gruppen gehört, so fügen wir die Gruppen als Index an den Buchstaben des Agenten an. So bezeichnen wir beispielsweise einen Agenten X, der den Gruppen A und C angehört, mit X_{AC}.

Jeder Anti-Spamagent verfügt über ein *Eingangs*- und ein *Ausgangsfach*. Zu Beginn jeder Runde verarbeitet er die Einträge seines Eingangsfachs, bis er dieses vollständig geleert und damit abgearbeitet hat. Das *Eingangsfach* kann drei verschiedene Arten von Einträgen enthalten.

- *Durch Verteileragenten zugestellte Emails*
  Holt der Agent in der aktuellen Runde Emails von seinem Account ab, so befinden sich danach alle Emails im Eingangsfach des Agenten (als eine Menge von Emailnummern), die seit der letzten Abholung von den Verteileragenten erstellt, an den Agenten adressiert und gesendet worden sind.

- *Von Antispam-Agenten empfangene Hashwerte*
  Alle Hashwerte, die andere Agenten des Spamfilternetzwerkes innerhalb der vorherigen Runde an den Antispam-Agenten geschickt haben, befinden sich ebenfalls im Eingangsfach.

- *Wahlanfragen*
  Ein beliebiger Anti-Spamagent X hat periodisch die Möglichkeit, bei Überschreitung gewisser Vertrauensschwellwerte eine Verbindung mit einem anderen Agenten Y einzugehen, bzw. die Aufnahme von Y in die Organisation von X vorzuschlagen. Die Nachrichten, die zur Abstimmung über die Aufnahme von Agenten, bzw. beim Zusammenschluss von Organisationen benötigt und versendet werden, liegen als dritte Art von Einträgen im Eingangsfach vor.

Da jedem Agenten eine eindeutige Nummer vergeben wird, kann er gezielt Nachrichten an bestimmte Agenten leiten. Dazu schiebt er die betreffende Nachricht unter Angabe seiner Absendernummer, der Nummer des Empfängers, des Typs sowie des Inhaltes der Nachricht in sein Ausgangsfach. Am Ende jeder Runde sorgt die Experimentalumgebung für den korrekten Versand der Nachrichten von den Sendern zu den Empfängern. Nach Abschluss jeder Runde sind somit die Ausgangsfächer aller Antispam-Agenten geleert. Die Funktionalität der in Abschnitt 4.1 vorgestellten Antispam-Agenten wurde vollständig auf die gleichnamigen Agenten der Experimentalumgebung übertragen. Die Bearbeitung der drei Arten von Einträgen des Eingangsfachs wird in den folgenden Abschnitten beschrieben. Danach wird die Prozedur zum Füllen der Ausgangsfächer erklärt.

Abholung und Klassifizierung der Emails

Der Agent holt nach einer zufälligen Anzahl von Runden seine Emails von seinem Account. Diese Zeitspanne würfelt der Agent unter Beachtung der Unter- und Obergren-

ze $Mailabholung_{min}$ und $Mailabholung_{max}$ aus. Damit wird das Verhalten simuliert, dass der Benutzer eines Emailaccounts in sporadischen Abständen seine Emails vom Mailserver herunterlädt. Die geladenen Emails werden im Eingangsfach des Antispam-Agenten abgelegt.

Analog zur Vorgehensweise im Spamfilternetzwerk aus dem vierten Kapitel wird für jede Email ein Hashwert erzeugt. Der Einfachheit halber entspricht der Hashwert der Emailnummer. Anschließend vergleicht der Antispam-Agent den generierten Hashwert mit den Einträgen in seiner lokalen Datenbank. Die Datenbank wird durch eine Liste von Hashwerten repräsentiert. Ferner wird für jeden Hashwert in einer *Zählvariablen* $ZV_{Hashwert}$ festgehalten, wie oft dieser schon an den betreffenden Agenten gesendet wurde. Ist der Hashwert der zu klassifizierenden Email in der Datenbank vorhanden und wurde der Wert dem Agenten bereits mindestens zweimal[1] zugesandt, so wird die Email als Spam markiert. Durch Überprüfung von $ZV_{Hashwert}$ kann der Antispam-Agent feststellen, ob er den Hashwert bereits mehr als einmal empfangen hat. *Hashwerte aus der Datenbank werden nur zur Klassifizierung der Emails des Antispam-Agenten eingesetzt, wenn der Agent den gleichen Hashwert mindestens zweimal über das Netzwerk empfangen hat*. Die Befolgung dieser Regel beim Einsatz der Hashwerte verhindert folgendes Szenario: Da die Klassifizierungsfehler der Agenten größer null sind, kommt es vor, dass Agenten Hashwerte zu Nicht-Spammails generieren. Diese nutzlosen Hashwerte versenden sie an die anderen Agenten im Netzwerk. Jeder Agent, der einen nutzlosen Hashwert erhält, würde mit dessen Hilfe die gleiche Nicht-Spammail ebenfalls als Spam klassifizieren. Durch Einsatz der Zählvariablen $ZV_{Hashwert}$ ist sichergestellt, dass mindestens zwei Agenten die gleiche Nicht-Spammail fälschlicherweise als Spam klassifizieren und an die gleichen Agenten versenden müssen, damit dieser Fall eintritt. Innerhalb unseres Netzwerkes aus Agenten hat sich die Wahl von $ZV_{Hashwert} = 2$ als geeignet erwiesen, um bei der Klassifizierung mit Hashwerten den Anteil falsch klassifizierter Emails gering zu halten. Damit setzt jeder Agent nur diejenigen Hashwerte zur Klassifizierung ein, die er mindestens zweimal über das Netzwerk empfangen hat. An dieser Stelle wollen wir darauf hinweisen, dass wir bei der Simulation der Textklassifizierers zur Vereinfachung annehmen, dass die Agenten beim Vorgang der Klassifizierung der Emails unabhängig voneinander agieren. Im realen Einsatz jedoch werden die Textklassifizierer unterschiedlicher Benutzer ein und dieselbe Spammail mit erhöhter Wahrscheinlichkeit aufgrund des Aufbaus ihrer Klassifizierungsmatrix als Spam erkennen. Die beiden Klassifizierungsergebnisse sind nicht voneinander unabhängig. Existiert der zur klassifizierenden Email korrespondierende Hashwert nicht in der Datenbank des Agenten, so wird keine Klassifizierung durch Hashwerte durchgeführt. Stattdessen wird die in Abschnitt 4.6 erläuterte Klassifizierung der Email mit Hilfe des Algorithmus „*support vector machines*“ simuliert. Den Anteil der Emails, die der SVM-Algorithmus der Agenten falsch klassifiziert, kann zu Beginn der Simulation für jeden Antispam-Agenten indi-

[1] Den Wert der Zählvariablen legen wir auf $ZV_{Hashwert} = 2$ fest.

viduell festgelegt werden. Hierzu gibt der Parameter *Klassifizierungsfehler* f_{class} den prozentualen Anteil der Emails an, die der simulierte SVM-Algorithmus des Agenten falsch in eine der Kategorien Spam oder Nicht-Spam einteilt. Daraufhin übernimmt der Agent die Emailnummer aller als Spam klassifizierten Emails als Hashwerte in seine lokale Datenbank. Setzt man den Klassifizierungsfehler auf 100 Prozent, so lässt sich das Verhalten böswilliger Agenten nachahmen. Böswillige Agenten senden falsche Hashwerte, zu denen keine korrespondierende Spammail existiert, durch das Netzwerk. Dadurch versuchen sie, die Effektivität des gesamten Systems herabzusetzen. Mit einem Klassifizierungsfehler von 100 Prozent erzeugt ein Anti-Spamagent für jede Nicht-Spammail einen Hashwert und schickt diesen durch das Spamfilternetzwerk. Die in Abb. 5.1 gezeigte Konfiguration der Experimentalumgebung enthält 15 Antispam-Agenten, die den Gruppen A, B und AB zugeordnet sind.

Bearbeitung empfangener Hashwerte

Ein Antispam-Agent X verfährt mit empfangenen Hashwerten nach deren Herkunft. Haben im Verlauf der Simulation der Absender des Hashwertes und Agent X auf Basis von Vertrauen eine Kante zueinander ausgebildet, so nimmt X den empfangenen Hashwert ohne Überprüfung in seine Datenbank auf und erhöht die Zählvariable $ZV_{Hashwert}$ des Hashwertes um eins. Die Hashwerte der restlichen Agenten des Spamfilternetzwerkes werden zwar mit den Einträgen der eigenen Datenbank verglichen, aber nicht gespeichert. Die in die Datenbank eingetragenen Hashwerte wurden zuvor entweder durch den Agenten selbst aus Spammails generiert, bzw. durch mit ihm in holonischen Organisationen verknüpften Agenten zugesendet. Stimmt der vom Absender versandte Hashwert mit einem Eintrag der Datenbank überein, so erhöht der Empfänger den Vertrauenswert zum Absender des Hashwertes. Ein Hashwert, der ein Agent X von einem Agenten Y erhält, und der mit einem Eintrag der Datenbank von X übereinstimmt, bezeichnen wir auch als *evaluierbaren Hashwert*. Hat Agent X die beiden Hashwerte verglichen, so verwirft er den empfangenen Hashwert, falls er ihn von Agenten erhält, mit denen er nicht über eine Kante verbunden ist. Diese Vorgehensweise verhindert, dass X zur Klassifizierung der Emails durch Hashwert-Vergleich Hashwerte von nicht vertrauenswürdigen Agenten verwendet[2].

Bearbeitung von Wahlanfragen

Unter Wahlanfragen fallen alle Nachrichten, die zur Kommunikation der Agenten beim Zusammenschluss dienen. Bevor Organisationen bestimmte Agenten einbinden, werden die aufzunehmenden Agenten auf Grund der zu ihnen aufgebauten Vertrauenswerte geprüft. Der Vertrauenswert jedes Agenten der Organisation zu allen Agenten der aufzunehmenden Agentengruppe muss den Schwellwert S_{HO} überschreiten. Anhand der Vertrauenswerte stimmen die Agenten einem Zusammenschluss zu, bzw. dieser wird abgelehnt. Lehnt ein Agent Y die Wahlanfrage eines Agenten X ab, so setzt X

[2] die wohlmöglich von böswilligen Agenten versendet wurden

den Vertrauenswert zu Y auf null. Dadurch wird sichergestellt, dass Agent X seine Wahlanfrage nicht periodisch immer an denselben Agenten Y stellt, der auf Grund seiner Verbindungen zu anderen Agenten keine weitere Verbindung mehr eingehen kann. Ausführlich wird auf den Zusammenschluss zwischen Organisationen und deren Agenten in Kapitel 5.2.2 eingegangen.

Füllen der Ausgangsfächer

Das *Ausgangsfach* des Antispam-Agenten wird mit zwei verschiedenen Arten von Nachrichten gefüllt. Dabei handelt es sich in erster Linie um Nachrichten mit einem Hashwert als Inhalt, der zur Information anderer Agenten des Netzwerkes über Spam dienen. Die Regeln, die ein Agent beim Versand der Hashwerte befolgen muss, finden sich in Kapitel 5.2.2. Ferner wird das Ausgangsfach auch mit Nachrichten gefüllt, die für die Kommunikation beim Zusammenschluss von Agenten benötigt werden. Die verschiedenen Nachrichten, die zwischen den Agenten zur Bildung von Organisationen ausgetauscht werden, fassen wir in Kapitel 5.2.3 zusammen.

Die Anzahl der Nachrichten, die jeder Anti-Spamagent pro Runde versendet, darf einen bestimmten Schwellwert nicht überschreiten. Dieser Parameter *Nachrichtenlimit* NL kann zu Beginn der Simulation festgelegt werden und gilt für alle Antispam-Agenten. Jede versendete Nachricht verursacht den Agenten Kommunikationskosten von eins. Ein Agent darf pro Runde maximal NL Nachrichten versenden. Damit wird sichergestellt, dass der Nachrichtenversand zwischen den Agenten eine gewisse Bandbreite nicht überschreitet und somit die Kommunikationskosten des Systems reduziert werden.

5.1.3 Nachrichtenversand

Der Versand der Nachrichten zwischen den Agenten des Spamfilternetzwerkes übernimmt die Experimentalumgebung. Hierzu werden nacheinander die Ausgangsfächer aller Antispam-Agenten abgearbeitet und die darin enthaltenen Nachrichten an die gewünschten Empfänger überstellt. Die Nachrichten enthalten folgende Attribute:

- *Priorität*
  Das $Nachrichtenlimit\ NL$ des Spamfilternetzwerkes spezifiziert die Anzahl von Nachrichten, die jeder Agent maximal pro Runde versenden kann. Ist der Wert von NL innerhalb des Netzwerkes groß genug gewählt, werden alle Nachrichten des betreffenden Agenten, die sich in dessen Ausgangsfach befinden, vom System versendet. Jedoch soll auch gewährleistet werden, dass die Funktionalität des Netzwerkes auch bei starker Einschränkung der Kommunikation zwischen den Agenten gewahrt bleibt. Hierzu ist es unabdingbar, dass Nachrichten, die beim Prozess des Zusammenschlusses der Agenten auftreten, die höchste Priorität erhalten. Damit ist sichergestellt, dass sich der Zusammenschluss von

Agenten bzw. ganzer Organisationen nicht wegen eines Kommunikationsengpasses bei einem einzelnen Agenten verzögert oder der Zusammenschluss gar nicht zustande kommt. Nachrichten, die einen Hashwert enthalten, besitzen die zweithöchste Priorität. Die dritthöchste Priorität wird Nachrichten zugeordnet, die als Inhalt eine Anfrage eines einzelnen Agenten bezüglich der Aufnahme in eine Organisation betreffen. Damit wird in großen Spamfilternetzwerken sichergestellt, dass Agenten nicht ihre gesamten Kommunikationsressourcen zur Beantwortung der Anfragen einzelner Agenten verschwenden. Die niedrigste Priorität wird den Nachrichten zugeordnet, die zum Aufbau von Vertrauen zwischen den Agenten dienen.

- *Sender und Empfänger*
  Mit Hilfe dieser Attribute kann das System die korrekte Zustellung der Nachrichten gewährleisten. Ferner können die Agenten jeder empfangenen Nachricht eindeutig einem bestimmten Agenten des Netzwerkes als Absender zuordnen.

- *Nachrichtentyp*
  Analog zur Kommunikationssprache in FIPA-OS [FIPA 2000a] bestehen die Nachrichten zwischen den Agenten der Experimentalumgebung aus standardisierten *Sprechakten*. Anhand des Nachrichtentyps kann der Empfänger einer Nachricht klar ersehen, ob es sich um eine Anfrage zum Zusammenschluss von Agenten oder um die Mitteilung eines Hashwertes o.ä. handelt.

- *Nachrichteninhalt*
  Im Inhalt der Nachricht sind weitergehende Informationen enthalten. So werden beispielsweise beim Zusammenschluss zweier Organisationen HO_1 und HO_2 jedem Agenten aus HO_1 die Identifizierungsnummern der Antispam-Agenten aus HO_2 mitgeteilt und umgekehrt.

In der Aktionsphase wird das Ausgangsfach des Agenten mit maximal NL Hashwerten gefüllt. In der Wahlphase werden weitere Nachrichten in das Ausgangsfach des Agenten eingefügt. Dies sind Nachrichten, die beim Zusammenschluss von Agenten versendet werden. Da die Wahlphase nach der Aktionsphase folgt, kann durch das Hinzufügen der Nachrichten in der Wahlphase die Anzahl der Nachrichten im Ausgangsfach das $Nachrichtenlimit$ überschreiten. Beim Versand aller dieser Nachrichten würde der betreffende Agent seine ihm zustehende Nachrichtenbandbreite überschreiten. Da aber die Nachrichten im Ausgangsfach nach steigender Priorität sortiert sind, werden nur die NL Nachrichten mit der höchsten Priorität an die Eingangsfächer anderer Agenten versendet. Die Nachrichten mit niedrigster Priorität werden verworfen. Somit ist sichergestellt, dass kein Agent sein $Nachrichtenlimit$ überschreitet. Die Funktionalität des Spamfilternetzwerkes bleibt gewährleistet, da nur Nachrichten niedriger Priorität nicht versendet werden. Nach Abschluss des Nachrichtenversands befinden sich die Nachrichten in den Eingangsfächern der entsprechenden Empfänger. Somit können sie in der folgenden Runde abgearbeitet werden.

5.2 Optimierter Datenversand durch Selbstorganisation

Durch intelligente Gruppierung der Antispam-Agenten zu holonischen Organisationen erreichen wir einen optimierten Datenversand zwischen den Agenten des Netzwerkes. Die Gruppierung der Agenten bezeichnen wir als *Selbstorganisation*. Eine auf unser Spamfilternetzwerk sinnvoll anwendbare Definition von Selbstorganisation findet sich bei KLIR [1991]:

Definition 8 (Selbstorganisation nach Klir, 1991)
Ein selbstorganisierendes System ist ein System, welches dazu tendiert, seine Performanz durch Verbesserung der Organisation seiner Elemente im Laufe der Zeit zu steigern, um ein Ziel zu erreichen.

Das Ziel des Spamfilternetzwerkes besteht in einer Maximierung der Anzahl identifizierter Spammails. Agenten finden sich zusammen und bilden Verbindungskanten aus, über die sie verstärkt Informationen austauschen. Sie senden ihre Spaminformationen nur an eine ausgewählte Teilmenge der Agenten. Dadurch wird ein wesentlicher Nachteil des ursprünglichen Spamfilternetzwerkes beseitigt. Dort wurden die von einem Agenten generierten Hashwerte an alle anderen Agenten geschickt. Damit waren die Kommunikationskosten sehr hoch. Nach Tel „besteht die Möglichkeit, dass der Durchsatz eines Kommunikationsnetzwerkes dramatisch sinkt, wenn mehrere Nachrichten gleichzeitig unterwegs sind“ [TEL 1994]. Um dies zu verhindern, wollen wir die Menge der versendeten Nachrichten minimieren bei Bewahrung der Effizienz des Netzwerkes aus Antispam-Agenten.

In der Experimentalumgebung finden sich die benevolenten Agenten mit gleicher Gruppenzugehörigkeit auf der Basis von Vertrauen zusammen. Damit wird auch ein weiterer wichtiger Sicherheitsaspekt von verteilten Netzwerken erfüllt. Böswillige Agenten können kein Vertrauen zu benevolenten Agenten des Systems aufbauen und werden aus dem sich bildenden Netz von Agentengruppen ausgeschlossen. Diese Eigenschaft werden wir im sechsten Kapitel experimentell nachweisen.

Informationsaustausch zur Ausbildung von Vertrauen zwischen Agenten wird seit längerem in Multiagentensystemen wie beispielsweise von LASHKARI et al. [1994] verwendet. Auf eine ausführliche Betrachtung des Begriffs „Vertrauen“ wollen wir im Rahmen dieser Diplomarbeit auf Grund der Vielzahl unterschiedlicher Definitionen und der Komplexität des Begriffs verzichten. Dazu sei auf weiterführende Informationen bei CASTELFRANCHI und FALCONE [1998] verwiesen. Vertrauen im Kontext der Spamfilterung wird zwischen Agenten aufgebaut, wenn ein Agent X von einem Agenten Y Informationen über Spam erhält, und X deren Gültigkeit anhand der Informationen seiner Datenbank überprüfen kann. Der Vorgang des Vertrauensaufbaus zwischen den Agenten der Experimentalumgebung wird detailliert im folgenden ersten

Abschnitt beschrieben. Haben bestimmte Agenten im Laufe der Simulation Vertrauen zueinander aufgebaut, können sie sich nach Überschreiten bestimmter Schwellwerte zusammenschließen und in einer *holonischen Organisation* (HO) vereinigen, welche an das von SCHILLO [2002] beschriebene *virtuelle Unternehmen* angelehnt ist. Die Eigenschaften dieser Organisation sowie die Aufgaben ihrer Mitglieder werden im zweiten Abschnitt vorgestellt. Schließlich folgen im dritten Abschnitt die zur Selbstorganisation der Agenten zu holonischen Organisation benötigten Wahlprotokolle, die den korrekten Zusammenschluss der Agenten zu Netzen von Organisationen gewährleisten.

5.2.1 Vertrauensaufbau zwischen den Antispam-Agenten

Die einzige Information, die den Agenten über andere Agenten zum Aufbau von Vertrauen zur Verfügung steht, sind die Hashwerte, die diese über das Spamfilternetzwerk an sie weiterleiten. Wir müssen daher einen Mechanismus entwickeln, welcher es den Antispam-Agenten ermöglicht, Vertrauen anhand der empfangenen Hashwerte aufzubauen und sich mit Agenten, die gleiche Spammails erhalten, zusammenzuschließen. Um dies zu erreichen, verfolgen die Antispam-Agenten das Ziel, die Anzahl evaluierbarer Hashwerte zu maximieren. Damit sind diejenigen Hashwerte gemeint, die von einem Agenten X an einen Agenten Y gesendet wurden, und die mit den Einträgen der Datenbank von Y übereinstimmen. Agent Y vergleicht den empfangenen Hashwert mit den Einträgen seiner Datenbank und kann dessen Gültigkeit nur dann evaluieren, wenn er den gleichen Wert anhand der gleichen Spammail entweder selbst generiert oder wenn er den gleichen Hashwert von einem Agenten seiner Organisation empfangen hat. Befindet sich der gleiche Hashwert in der Datenbank, so erhöht Agent Y den Vertrauenswert von X um eins. Der Vertrauenswert zwischen X und Y wird hierbei durch eine natürliche Zahl abgebildet und ist zu Anfang der Simulation mit null initialisiert. Je größer der Vertrauenswert eines Agenten X zu einem Agenten Y ist, desto mehr Vertrauen besitzt X zu Y. Zwei zentrale Forderungen für den Aufbau von Vertrauen zwischen Agent X und Y müssen erfüllt werden:

- *Agenten dürfen nur Vertrauen zueinander aufbauen, wenn sie sich mindestens in einer gemeinsamen Gruppe befinden und somit eine gemeinsame Menge von Spammails erhalten.*

  Seien zwei Agenten X und Y aus verschiedenen Gruppen gegeben. Die Zuordnung zu verschiedenen Gruppen bewirkt, dass X und Y keine identischen Emails erhalten. Daraus folgt, dass die von ihnen aus Spammails generierten Hashwerte mit höchster Wahrscheinlichkeit nicht übereinstimmen. Deshalb besitzen sie auch disjunkte Mengen von Hashwerten in ihren Datenbanken. Das Vertrauen zwischen den Agenten wird aber nur erhöht, wenn Agent X zu einem von Agent Y zugesandten Hashwert bereits einen identischen Eintrag in seiner Datenbank besitzt und daher dessen Korrektheit überprüfen kann. Aufgrund ih-

rer disjunkten Datenbanken von Hashwerten kann zwischen X und Y kein Vertrauen gebildet werden. Gehören umgekehrt zwei benevolente Agenten X und Y mindestens einer gemeinsamen Gruppe an, so erhalten sie die gleichen Spammails, generieren die gleichen Hashwerte und können die zugesandten Hashwerte mit den Einträgen ihrer Datenbank vergleichen. Bei erfolgreichem Vergleich erhöhen X und Y das Vertrauen zueinander, der Vertrauensaufbau zwischen X und Y findet statt.

- *Benevolente Agenten dürfen nur Vertrauen zu anderen benevolenten Agenten aufbauen.*

  Agenten im Spamfilternetzwerk haben ausschließlich die Möglichkeit, Hashwerte für Spammails zu versenden. Um die Funktionalität des Netzwerkes zu beeinflussen, müsste ein böswilliger Agent Hashwerte verteilen, die zu keinen Spammails korrespondieren. Diese wertlosen Werte können von benevolenten Agenten nicht evaluiert werden. Sie verfügen über keinen passenden Hashwert in ihrer eigenen Datenbank, da benevolente Agenten nur Werte aus Spammails generieren. Ein benevolenter Agent X kann jedoch Hashwerte eines benevolenten Agenten Y auswerten, solange sich beide mindestens in einer gemeinsamen Gruppe befinden. Dann bekommen sowohl X und Y die gleichen Spammails und generieren daraus die gleichen Hashwerte. Sendet X einen Hashwert an Y, und hat Y ebenfalls die korrespondierende Email als Spam klassifiziert, so findet Y den gleichen Hashwert in seiner Datenbank. Damit hat er den Hashwert evaluiert und erhöht das Vertrauen zu Agent X.

Konkret speichert jeder Agent der Experimentalumgebung die Vertrauenswerte zu anderen Agenten des Spamfilternetzwerkes in einer Liste ab. Dabei kann er jeden Agenten anhand der Identifizierungsnummer erkennen, die beim Versand von Hashwerten innerhalb des Netzwerkes mitgesendet wird.

Im Laufe der Simulation kann sich die Situation ergeben, dass ein benevolenter Agent X eine Nicht-Spammail fälschlicherweise als Spammail klassifiziert[3] und den generierten Hashwert in seiner Datenbank speichert. Sendet ein böswilliger Agent Y den gleichen ungültigen Hashwert an X, so evaluiert dieser den Wert erfolgreich und erhöht das Vertrauen zu Y. Ist die Rundenzahl der Simulation groß genug gewählt, so kann die geschilderte Situation mehrfach auftreten und der Schwellwert S_{HO} zur Bildung einer Verbindung zwischen X und Y kann überschritten werden. Damit würde sich eine Verbindung von X zu dem böswilligen Agenten Y ergeben. Um dies zu verhindern, bietet die Experimentalplattform die Möglichkeit, die Vertrauenswerte aller Agenten untereinander periodisch nach einer bestimmten Rundenzahl um eins zu vermindern. Diese Prozedur führt dazu, dass die sich sporadisch bildenden Vertrauenswerte zwischen benevolenten und böswilligen Agenten vermindert werden. Auch die

[3] falls sein Klassifizierungsfehler größer null ist

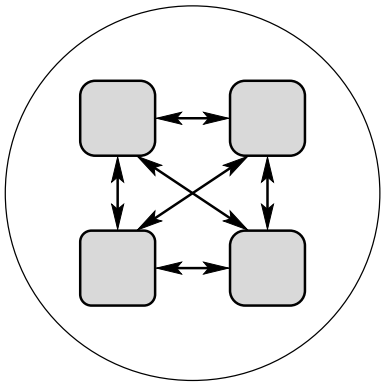

Abbildung 5.2: Schematische Darstellung einer holonischen Organisation mit Clique aus vier über Kanten verbundenen Agenten

Vertrauenswerte zwischen benevolenten Agenten werden um eins vermindert. Jedoch steigen die Vertrauenswerte zwischen benevolenten Agenten einer Gruppe so stark an, dass sie durch die Prozedur nur unwesentlich vermindert werden.

5.2.2 Verbindungen in der holonischen Organisation

Zu Beginn der Simulation existieren keine Verbindungen zwischen den Agenten. Haben die Vertrauenswerte zwischen Agenten durch Evaluation der empfangenen Hashwerte den von uns festgelegten Schwellwert S_{HO} überschritten, so schließen sie sich zusammen. Den Vorgang bezeichnen wir als *Selbstorganisation*. Die Organisation, in der sich die Agenten vereinigen, nennen wir *holonische Organisation HO*. Die maximale Anzahl von Mitgliedern in der HO wird durch den Parameter O_{max} bestimmt. Die Antispam-Agenten innerhalb der holonischen Organisation sind gleichberechtigt. Jeder Agent ist mit allen anderen Agenten der HO über eine direkte ungerichtete Kante verbunden (siehe Abb. 5.2). Ein Agent kann gleichzeitig nur Mitglied in einer Organisation sein. Die verwendete Netzwerktopologie, in der eine Kanten zwischen zwei beliebigen Knoten existiert, wird in der Literatur als *Clique* bezeichnet. In diesem Zusammenhang definieren wir den Begriff *Organisationsnetz*:

Definition 9 *Unter einem Organisationsnetz verstehen wir ein über Kanten verbundenes Netzwerk aus mindestens zwei holonischen Organisation. Zwei holonische Organisationen HO_1 und HO_2 dürfen innerhalb des Organisationsnetzes nur durch höchstens eine Verbindungskante miteinander verbunden sein.*

Ein optimales Organisationsnetz ist gebildet, wenn jede beliebige Organisation HO mit jeder anderen Organisation des Netzes über genau eine Kante verbunden ist. Die Kante zwischen zwei holonischen Organisation HO_1 und HO_2 bezeichnen wir als *Verbindungskante*, die durch die Verbindungskante miteinander verbundenen Agenten X aus HO_1 und Y aus HO_2 heißen *Verbindungsagenten*. Die Verbindungskanten

stellen die einzigen Verbindungen zwischen den holonischen Organisation dar. Jeder Anti-Spamagent darf höchstens eine Verbindungskante zu einer anderen Organisation ausbilden.

Ein Beispiel für ein optimales Organisationsnetz findet sich in Abb. 5.4. Innerhalb optimaler Organisationsnetze können die Hashwerte effizient und schnell verteilt werden, was wir in der experimentellen Evaluation nachweisen. Wir werden nun im folgenden Abschnitt auf die Aufgaben der Mitglieder der holonischen Organisation eingehen. Weiterhin errechnen wir eine Obergrenze für die Anzahl von Hashwerten, die ein Agent aufgrund seiner Mitgliedschaft höchstens innerhalb seiner Organisation versenden muss. Die optimale Organisationsgröße zum Versand der Hashwerte an eine möglichst große Anzahl von Antispam-Agenten leiten wir im zweiten Abschnitt her. Im dritten Abschnitt stellen wir zwei Arten von unzulässigen Verbindungen zwischen Organisationen vor, die sich nach deren Zusammenschluss bilden können. Wir beschreiben einen Algorithmus, welcher die unnötigen Verbindungen nach Zusammenschluss der Agenten automatisch löscht.

Verpflichtungen und Nutzen der Mitgliedschaft in der HO

Sei N die Anzahl der Mitglieder der holonischen Organisation mit $N \leq O_{max}$. Dann ergibt sich die Gesamtzahl von Kanten in dieser HO zu $\frac{N(N-1)}{2}$. Zwei holonische Organisationen HO_1 und HO_2 können miteinander verschmelzen, solange ihre Gesamtgröße O_{max} nicht überschreitet. Ist die Gesamtgröße von HO_1 und HO_2 größer als die maximale Organisationsgröße O_{max}, so bietet sich den holonischen Organisationen die Möglichkeit, eine Verbindungskante zu errichten. Zwischen zwei Organisationen HO_1 und HO_2 darf hierbei nur jeweils eine Verbindungskante existieren. Diese Einschränkung garantiert, dass der gleiche Hashwert innerhalb einer Runde nicht über mehrere Verbindungen zwischen zwei Organisationen mehrfach ausgetauscht wird. Die Verbindungskante wird von einem Agenten aus HO_1 zu einem Verbindungsagenten aus HO_2 gebildet. Die Gesamtzahl ges_{Org} von Organisationen, mit denen eine holonische Organisation HO gleichzeitig verbunden sein kann, beträgt somit $ges_{Org}(HO) \leq N$.

Aus der Mitgliedschaft in der holonischen Organisation ergeben sich für die einzelnen Mitglieder Rechte und Pflichten. In jeder Runde müssen die Agenten einer Organisation der Größe O_{max} drei Regeln befolgen, die sie verpflichten, Hashwerte an andere Agenten des Organisationsnetzes zu senden:

1. Generiert ein Agent in einer Runde eine Menge von Hashwerten, so ist er verpflichtet, einen zufällig aus dieser Menge gewählten Hashwert an alle anderen $O_{max} - 1$ Agenten innerhalb seiner holonischen Organisation, sowie an seinen Verbindungsagenten zu versenden. Der Agent versendet nur einen selbst generierten Hashwert pro Runde, um seine Kommunikationskosten für diesen Vorgang kleiner gleich O_{max} zu halten.

2. Empfängt ein Agent über seine Verbindungskante zu einer anderen holonischen Organisation einen Hashwert, so muss er diesen an alle Mitglieder seiner Organisation weitersenden. Ein Agent erhält über seine Verbindung zu einer anderen Organisation pro Runde höchstens zwei Hashwerte, die er an die $O_{max} - 1$ Mitglieder seiner Organisation weiterleitet. Der Versand der beiden Hashwerte kostet höchstens $2(O_{max} - 1)$ pro Runde.

3. Empfängt ein Agent eine Menge M von Hashwerten, die andere Agenten seiner Organisation generiert haben, so verpflichtet er sich, einen dieser Werte an seinen Verbindungsagenten zu senden. Der zu sendende Hashwert wird zufällig aus der Menge M ausgewählt. Damit kommen auf den Agenten Kommunikationskosten von eins zu. Der betreffende Verbindungsagent muss nach der zweiten Regel den betreffenden Hashwert innerhalb seiner Organisation verteilen. Damit betragen die zusätzlichen Kosten für den Verbindungsagenten höchstens $O_{HO_2} - 1$, mit O_{HO_2} = Anzahl der Organisationsmitglieder des Verbindungsagenten.

Diese drei Regeln für den Versand von Hashwerten führen dazu, dass ein Hashwert folgenden Weg innerhalb des Netzes aus Organisationen nehmen kann, sofern die Agenten den Wert weiterleiten. Ein beliebiger Agent X aus HO_1 sendet genau einen Hashwert pro Runde nach dessen Generierung an alle Mitglieder der eigenen Organisation. In der folgenden Runde senden die Mitglieder der Organisation HO_1 den Wert an ihre jeweiligen Verbindungsagenten benachbarter Organisationen. Schließlich verteilen die Verbindungsagenten den Hashwert in der dritten Runde innerhalb ihrer eigenen Organisation. Danach wird dieser Hashwert nicht mehr weiter versendet. Durch Beendigung des Versands der Hashwerte nach drei Runden wird verhindert, dass der gleiche Hashwert mehrfach an die gleichen Agenten gesendet wird. Somit legt jeder versandte Hashwert maximal einen Pfad der Länge drei zurück. Damit alle am Versand der Hashwerte innerhalb dieser drei Runden beteiligten Agenten unterscheiden können, ob sie den Wert von einem Agenten innerhalb ihrer Organisation oder einem Verbindungsagenten erhalten haben, ist die Herkunft des Hashwertes im Nachrichtentyp festgehalten. Der gerade beschriebene Versand eines Hashwertes von einem Agenten X an die Mitglieder seines Organisationsnetzes bestehend aus den Organisationen HO_1 bis HO_3 innerhalb von drei Runden wird in Abb. 5.3 dargestellt.

Da ein Agent X die drei Regeln zur Begrenzung der Kommunikationskosten befolgt, leitet er nur einen Teil der generierten und empfangenen Hashwerte an die Mitglieder seines HO und seinen Verbindungsagenten weiter. Die Befolgung der drei Regeln führt somit dazu, dass Agent X nicht jeden Hashwert über Pfade mit maximaler Länge $l_{path} = 3$ zu allen anderen Agenten des Organisationsnetzes sendet. Jedoch generieren auch andere Agenten des Organisationsnetzes, die den gleichen Hashwert noch nicht erhalten haben, nach Abholung der entsprechenden Email den gleichen Hashwert und schicken diesen ebenfalls durch das Netz aus Organisationen. Dadurch werden identische Hashwerte von verschiedenen Punkten gleichzeitig effizient im Organisationsnetz

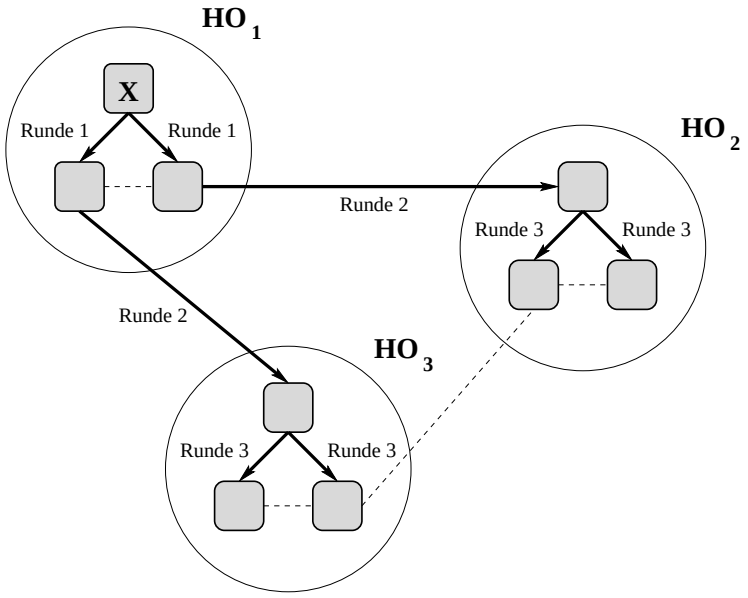

Abbildung 5.3: Versand eines Hashwertes von Agent X an die Agenten des Organisationsnetzes innerhalb von drei Runden

verteilt, obwohl wir die Kommunikationsmöglichkeiten der einzelnen Agenten durch den Parameter $Nachrichtenlimit$ nach oben beschränken.

Die Gesamtkosten $Cost_{Hashwert}$, die zum Versand der Hashwerte im Netz aus holonischen Organisationen pro Agent innerhalb einer Runde anfallen, können nach Berücksichtigung der drei Regeln zum Hashwert-Versand nach oben durch dem Term

$$\begin{aligned} Cost_{Hashwert} &\leq 2(O_{max}-1)+(O_{max}-1)+2 \\ &\leq 3(O_{max}-1)+2 \\ &\leq 3\,O_{max}-1 \end{aligned} \tag{5.1}$$

abgeschätzt werden. Ein Agent sendet pro Runde maximal *drei Hashwerte an jedes Mitglied seiner HO* und maximal *zwei Hashwerte an seinen Verbindungsagenten*. Zum reibungslosen Ablauf der Simulation muss also garantiert werden, dass der Parameter $Nachrichtenlimit\ NL$ größer als $3O_{max}-1$ gewählt wird. Hat ein Agent die höchstens $3O_{max}-1$ Nachrichten mit Hashwerten als Inhalt an die mit ihm direkt verbundenen Agenten geschickt, so verbleiben ihm die restlichen Nachrichten bis zum Erreichen von NL zum Aufbau von Vertrauen zu anderen Agenten des Spamfilternetzwerkes. Ein Agent zieht seinen Nutzen aus der Mitgliedschaft in der HO daraus, dass er durch Organisationsmitglieder evaluierbare Hashwerte erhält und speichert. Holt er nach Erhalt eines Hashwertes dessen korrespondierende Spammail von seinem Emailaccount ab, so klassifiziert er diese als Spam anhand seiner Hashwert-Datenbank.

Ein Agent X im Spamfilternetzwerk erhält zusätzlich von beliebigen Agenten außerhalb seiner Organisation zum Aufbau von Vertrauen eine zufällige Anzahl von Hashwerten, die er jedoch nicht abspeichert. Durch Mitgliedschaft im Netz aus holonischen Organisation kann ein Agent pro Runde maximal $3O_{max} - 1$ weitere Hashwerte erhalten. Da diese Hashwerte nur von benevolenten, vertrauenswürdigen Agenten des Organisationsnetzes bereitgestellt wurden, übernimmt Agent X diese in seine lokale Hashwert-Datenbank. Anhand dieser Hashwerte ist Agent X in der Lage, Spammails durch Hashwert-Vergleich mit Hilfe seiner Datenbank zu erkennen. Hashwerte von Agenten seines Organisationsnetzes sind für X wertvoll, weil sie von Agenten stammen, die die gleichen Spammails wie X erhalten. Somit kann X sie einsetzen, wenn er die korrespondierenden Spammails vom Mailserver abholt. Die Hashwerte von Agenten außerhalb seines Organisationsnetzes verwendet Agent X nur zum Aufbau von Vertrauen und verwirft sie anschließend wieder.

Optimale Organisationsgröße zum effizienten Versand der Hashwerte

Jeder Agent des Netzwerkes ist bestrebt, Verbindungen mit möglichst vielen Agenten einzugehen. Je mehr Verbindungen der Agent X eingeht, desto größer wird die Anzahl evaluierbarer Hashwerte, die er empfängt. Die Anzahl lässt sich durch Herstellung von Verbindungskanten zwischen seiner Organisation und anderen Organisationen weiter erhöhen. Im optimalen Fall beträgt die Anzahl $Anz_{max}(X)$ der Knoten (und damit Agenten), die Agent X über alle Pfade der Längen $l_{Pfad} \in \{1, \ldots, 3\}$ erreichen kann:

$$Anz_{max}(X) = O_{max}^2 + (O_{max} - 1) \tag{5.2}$$

Über direkte Kanten mit $l_{Pfad} = 1$ erreicht Agent X die $O_{max} - 1$ Mitglieder seiner HO. Über die Verbindungsagenten seiner Organisation lassen sich die O_{max}^2 Mitglieder der O_{max} verbundenen holonischen Organisationen über Pfade der Längen $l_{Pfad} \in \{2, 3\}$ erreichen. Wählt man die maximale Organisationsgröße $O_{max} \geq \sqrt{N}$, N = Gesamtzahl Agenten des Netzwerkes, so erreicht Agent X im optimalen Fall über Pfade mit maximaler Länge $l_{path} = 3$ folgende Anzahl von Agenten:

$$\begin{aligned} Anz_{max}(X) &\geq \sqrt{N}^2 + (\sqrt{N} - 1) \\ &\geq N + (\sqrt{N} - 1) \\ &> N \end{aligned} \tag{5.3}$$

Aus Formel (5.3) folgt, dass bei optimalem Zusammenschluss[4] der Agenten ein beliebiger Agent Y über maximal drei Kanten jeden anderen Agenten erreicht. Über diese Struktur von Kanten lassen sich die Hashwerte sehr schnell im Netzwerk verbreiten.

[4] Jede Organisation ist über Verbindungskanten mit allen anderen Organisationen des Netzwerkes verbunden

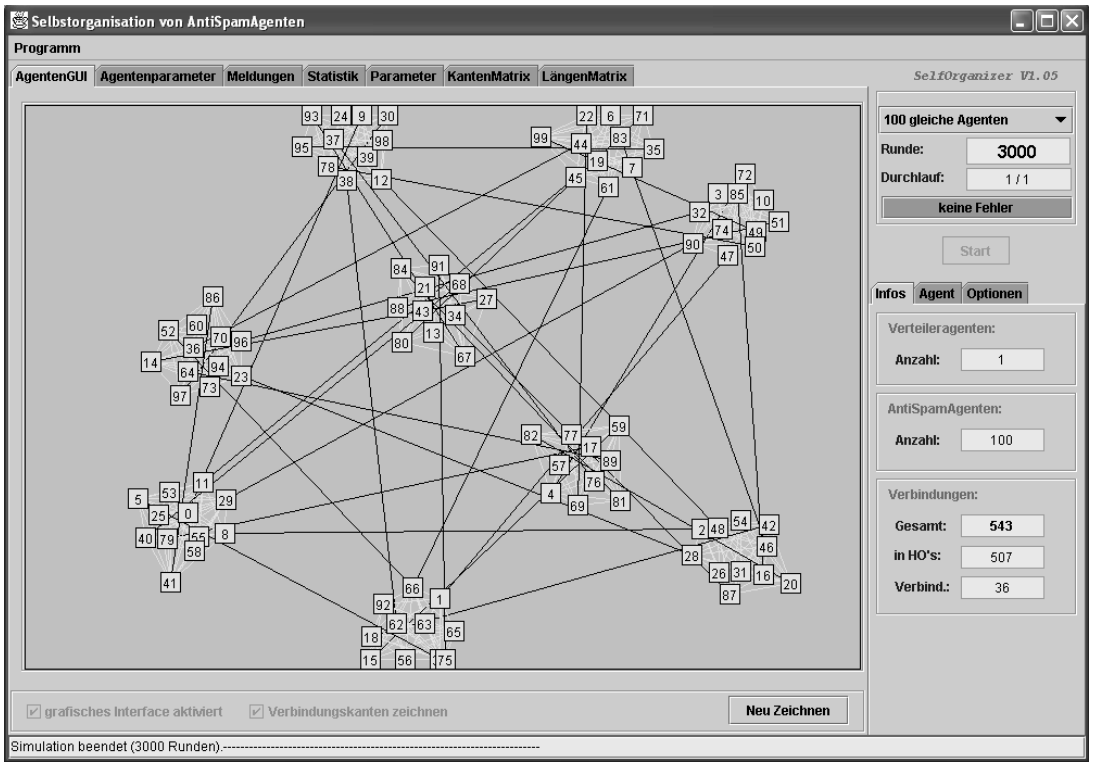

Abbildung 5.4: Netz aus holonischen Organisationen nach Beendigung der Simulation

Der optimale Zusammenschluss der Agenten lässt sich nicht immer erreichen, denn beim Vorgang der Selbstorganisation kann ein Organisation HO_1 nur dann mit einer Organisation HO_2 verschmelzen, wenn gilt:

$$O_{HO_1} + O_{HO_2} \leq O_{max} \tag{5.4}$$

Bilden sich im Lauf der Simulation zwei Organisationen, deren Gesamtgröße O_{max} überschreitet, so können sie sich nicht zusammenschließen. Somit erreichen nicht alle Organisationen die maximale Organisationsgröße, die Bedingung aus Formel (5.3) kann nicht erfüllt werden. Obwohl sich innerhalb der Experimentalumgebung nicht alle Agenten zu Organisationen maximaler Organisationsgröße vereinigen, schließen sich die Agenten dennoch zu Netzen zusammen, in denen die Hashwerte über kurze Pfade versendet werden können. Diese Eigenschaft werden wir in Kapitel 6.2.2 nachweisen.

In Abb. 5.4 ist die Gruppierung von 100 Agenten aus der gleichen Gruppe nach Beendigung der Simulation dargestellt. Es haben sich neun holonische Organisationen gebildet, die über 36 Verbindungskanten verbunden sind. Jeder Knoten X ist von jedem beliebigen Knoten Y über einen Pfad der Länge $l_{Pfad} \leq 3$ erreichbar. Die maximale Organisationsgröße in dieser Simulation beträgt $O_{max} = 12$. Die durchschnittliche Pfadlänge ist $l = 2.6$.

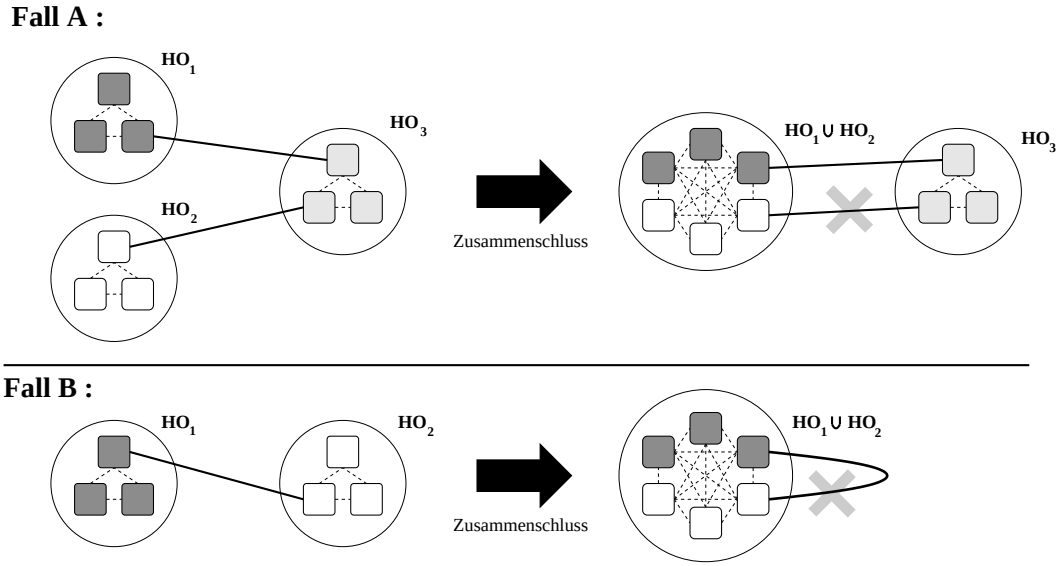

Abbildung 5.5: Szenarien, in denen Verbindungskanten nach dem Zusammenschluss von Organisationen gelöscht werden müssen

Algorithmus zur Auflösung unzulässiger Verbindungen zwischen Organisationen

Schließen sich zwei Organisationen HO_1 und HO_2 zusammen, so kann es zu unzulässigen Kombinationen von Verbindungskanten kommen. Ein Algorithmus am Ende jeder Runde garantiert, dass folgende Regeln beim Zusammenschluss die Konsistenz des Netzwerkes erhalten:

- Fall A: Existiert sowohl eine Verbindungskante $\overline{HO_1HO_3}$ von Organisation HO_1 zu Organisation HO_3 als auch eine Kante $\overline{HO_2HO_3}$ von Organisation HO_2 nach HO_3, so ist eine der beiden Kanten nach Zusammenschluss von HO_1 und HO_2 zu löschen.
- Fall B: Sei $\overline{HO_1HO_2}$ eine Kante zwischen HO_1 und HO_2. Nach Zusammenschluss der beiden Organisationen werden automatisch Kanten zwischen allen Mitgliedern von HO_1 und HO_2 aufgebaut. Die Verbindung $\overline{HO_1HO_2}$ wird nicht mehr benötigt und muss gelöscht werden.

Die beiden Szenarien sind als Fälle A und B in Abb. 5.5 dargestellt. Auf der linken Seite befinden sich die Organisationen vor dem Zusammenschluss. Die zu löschende überflüssige Kante ist auf der rechten Seite der Abbildung jeweils mit einem Kreuz markiert.

5.2.3 Wahlprotokolle für den Zusammenschluss von Agenten

Die Agenten der Experimentalplattform sind im Round-Robin-Verfahren berechtigt, Anfragen an andere Agenten zwecks eines Zusammenschlusses zu stellen. Den Agen-

ten wird mit steigender Identifizierungsnummer nacheinander im Abstand von sechs Runden erlaubt, ein Wahlverfahren zu starten. Dadurch wird vermieden, dass sich verschiedene gleichzeitig stattfindende Wahlverfahren gegenseitig beeinflussen. Das Wahlprotokoll legt die Reihenfolge der zu versendenden Nachrichten zwischen zwei Mengen von Agenten fest, die sich zusammenschließen wollen. Die maximale Länge eines Wahlverfahrens beträgt mit den von uns festgelegten Wahlprotokollen höchstens fünf Runden. Dabei wählt ein Agent X denjenigen Agenten Y für einen Zusammenschluss aus, der den höchsten Wert in seiner Liste von Vertrauenswerten besitzt. Anfragen kann Agent X nur an Agenten schicken, die sich nicht in seiner eventuell vorhanden Organisation sowie in allen direkt damit verbundenen Organisationen befindet, welche er in einer Liste speichert. Wahlvorgänge zwischen Agenten werden abgebrochen, falls der Vertrauenswert eines Agenten zu einem der Agenten der anderen Agentenmenge kleiner als der Schwellwert S_{HO} ist. In diesem Fall bricht der Agent die Wahl ab, indem er nicht die nächstfolgende Nachricht im Wahlprotokoll an die anderen Agenten schickt *(engl. deadline)*. Bleibt diese Nachricht in der folgenden Runde aus, so bricht die Wahl erfolglos ab und die Agenten schließen sich nicht zusammen. Zur Vereinfachung nehmen wir an, dass der Datenaustausch zwischen Sender und Empfänger immer reibungslos funktioniert, d.h. es treten keine Kommunikationsfehler[5] beim Austausch von Informationen auf.

Bevor sich zwei Mengen von Agenten aus HO_1 und HO_2 zusammenschließen können, müssen eine Reihe von Bedingungen erfüllt sein. Wie schon erläutert, darf die Gesamtgröße von HO_1 vereinigt mit HO_2 die maximale Organisationsgröße O_{max} nicht überschreiten. Ferner müssen alle Agenten aus Organisation HO_1 zu allen Agenten aus HO_2 einen Vertrauenswert größer S_{HO} besitzen und umgekehrt, d.h. ihre Vertrauenswerte untereinander dürfen einen bestimmten Schwellwert nicht unterschreiten. Um alle diese Bedingungen mit möglichst geringen Kommunikationskosten in kurzer Rundenanzahl zu überprüfen, haben wir Protokolle entwickelt, welche dieser Aufgabe gerecht werden. Dabei ist anzumerken, dass das Wahlverfahren während der laufenden Simulation mit Versand der Hashwerte etc. stattfindet. Die Wahl erstreckt sich über mehrere Runden, je nachdem wie komplex die sich verbindenden Organisationen aufgebaut sind. Haben die Agenten das Protokoll zur Bildung einer Organisation erfolgreich abgeschlossen, so erhalten sie alle den Befehl zum Zusammenschluss. Dabei achten die Agenten darauf, dass die Liste der Mitglieder der neu gebildeten Organisation für alle Agenten aus HO_1 und HO_2 aktualisiert wird. Nach der Komplexität der Organisationen HO_1 und HO_2 unterscheiden wir drei Arten von Wahlprotokollen für verschiedene Wahlverfahren:

[5] z.B. auf Grund eines fehlerhaften physikalischen Übertragungsmediums

Wahlprotokoll für den Zusammenschluss zweier Agenten

Das Wahlverfahren für den erfolgreichen Zusammenschluss zweier Agenten X und Y dauert drei Runden. In der ersten Runde stellt X die Anfrage für den Zusammenschluss an denjenigen Agenten Y, zu dem X den höchsten Vertrauenswert besitzt. Übersteigt das Vertrauen von Y zu X den von uns festgelegten Schwellwert S_{HO}, so nimmt Y die Wahl an und nimmt X in die Liste seiner Organisationsmitglieder auf. In der zweiten Runde sendet er den Befehl zum Zusammenschluss an X. Diesen Befehl erhält Agent X in der dritten Runde und übernimmt Y in die Liste seiner Organisationsmitglieder. Eine holonische Organisation aus X und Y ist gebildet. Für die Kommunikationskosten $Cost_{Join}$ des erfolgreichen Wahlverfahrens gilt: $Cost_{Join} = 2$.

Wahlprotokoll für den Zusammenschluss von Organisation und einzelnem Agenten

Der Zusammenschluss von Agent X und holonischer Organisation HO läuft innerhalb von vier Runden ab. In der ersten Runde stellt X die Anfrage an einen Agenten Y aus der Organisation HO. Falls der Vertrauenswert von Y zu X größer S_{HO} ist, sendet Y in der zweiten Runde eine Anfrage an alle Mitglieder der HO bezüglich der Aufnahme von X. Alle positiven Antworten erhält Y im Verlauf der dritten Runde. Haben alle Agenten aufgrund ihrer Vertrauenswerte einer Aufnahme von X zugestimmt, so sendet Agent Y in Runde vier den Befehl zum Zusammenschluss an alle Agenten aus HO sowie an X. Als Anhang ist diesem Befehl die aktualisierte Mitgliederliste mit den Agenten aus HO und Agent X beigefügt. Die Kommunikationskosten $Cost_{Join}$ betragen $Cost_{Join} = 2+2(O_{HO}-1)$, wobei O_{HO} die aktuelle Mitgliederzahl von HO ist.

Wahlprotokoll für den Zusammenschluss von zwei Organisationen

Das Wahlprotokoll für den Zusammenschluss zweier Organisationen stellt das komplexeste der drei Protokolle dar. Seine einzelnen Schritte sind in Abb. 5.6 dargestellt. Die Pfeile zwischen den Aktoren stellen die untereinander versendeten Nachrichten dar. Zu Beginn stellt der Initiator X aus Organisation HO_1 die Anfrage *(engl. request)* für einen Zusammenschluss an den Empfänger Y aus HO_2. Innerhalb der Anfrage übermittelt er die Liste aller Mitglieder aus HO_1. Ist der Vertrauenswert von Y zu allen Agenten aus HO_1 größer als S_{HO}, so sendet er die Anfrage mit der Liste der Mitglieder von HO_1 als Anhang weiter an die Mitglieder seiner Organisation HO_2. Gleichzeitig sendet ein eine Bestätigung an X zurück, die die Fortführung des Wahlverfahrens gestattet (in Abb. 5.6 sind die Nachrichten, die die Fortführung der Wahl gestatten, mit dem Label „*ok*“ gekennzeichnet). Im Anhang dieser Nachricht wird die Liste der Mitglieder aus HO_2 mitgesendet. In der dritten Runde leitet der Initiator X die Liste zur Überprüfung der Vertrauenswerte weiter an die Mitglieder seiner Organisation HO_1. Zur gleichen Zeit überprüfen die Agenten aus HO_2 ihre Vertrauenswerte zu allen Agenten aus HO_1. Ist der Vertrauenswert eines Agenten aus HO_2 zu

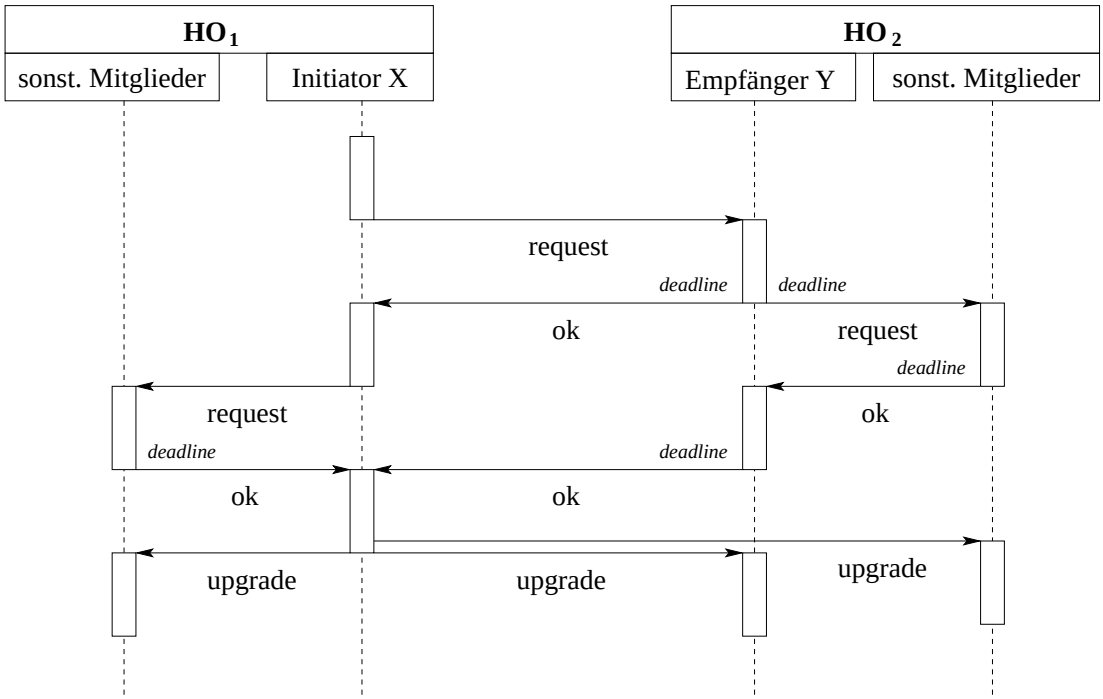

Abbildung 5.6: Wahlprotokoll für den Zusammenschluss zweier Organisationen

jedem Agenten aus HO_1 größer S_{HO}, so sendet er eine Bestätigung zur Fortführung des Wahlverfahrens an Y. Agent Y zählt die Menge der eintreffenden Bestätigungen. Hat HO_2 die Organisationsgröße O_{HO_2}, so fährt er mit dem Protokoll nur fort, wenn er von allen anderen $O_{HO_2} - 1$ Agenten aus HO_2 eine Bestätigung erhalten hat. Ist diese Bedingung erfüllt, so sendet er in der vierten Runde eine Bestätigung zurück an den Initiator X. In dieser Runde haben die anderen Agenten von HO_1 die Überprüfung der Vertrauenswerte der Agenten aus HO_2 abgeschlossen. Jeder schickt (sofern sein Vertrauenswert zu allen Agenten aus HO_2 größer S_{HO} ist) eine Bestätigung an X. Hat X die Bestätigungen aller Agenten seiner Organisation sowie die Bestätigung vom Empfänger Y erhalten, so weiß er, dass alle Agenten aus HO_1 und HO_2 zum Zusammenschluss bereit sind. Daher sendet er in der fünften Runde den Befehl zum Zusammenschluss (in Abb. 5.6 als *„upgrade“* bezeichnet) an alle Agenten aus HO_1 und HO_2. In dieser Nachricht ist die zusammengesetzte Mitgliederliste aller Agenten der neu gegründeten Organisation mit $O_{neu} = O_{HO_1} + O_{HO_2}$ enthalten. Somit sind alle beteiligten Agenten über den Zusammenschluss und die nun aktuell gültige Mitgliederliste ihrer neu gegründeten Organisation informiert. Die am Wahlverfahren beteiligten Agenten können jederzeit das Wahlverfahren abbrechen, indem sie auf die Anfragen und Mitteilungen der anderen Agenten nicht antworten. Diese Situation wird in Abb. 5.6 als *deadline* bezeichnet. Damit wird der Vorgang des Zusammenschlusses der Organisationen beendet.

Die Kommunikationskosten des Wahlverfahrens sind anhand der Abb. 5.6 leicht herzuleiten. Nachrichten zwischen Initiator X und Empfänger Y haben Kosten eins. Die Kommunikation zwischen X und seinen Mitgliedern aus HO_1 hat Kosten $O_{HO_1} - 1$, der Versand der Nachrichten zwischen Y und den Agenten aus HO_2 kostet $O_{HO_2} - 1$. Damit ergeben sich die Gesamtkosten $Cost_{gesamt} = 4 + 3(O_{HO_1} - 1) + 3(O_{HO_2} - 1)$.

Ausbildung einer Verbindungskante zwischen zwei Organisationen

Eine Verbindungskante zwischen zwei Organisationen HO_1 und HO_2 wird ausgebildet, wenn sich beide Organisationen zusammenschließen wollen, der Zusammenschluss aber aufgrund der Größen der beiden Organisationen nicht möglich ist, da folgende Bedingung nicht erfüllt wird: $O_{HO_1} + O_{HO_2} \leq O_{max}$. Zusätzlich darf keine Verbindungskante zwischen den beiden Organisationen existieren. Die Ausbildung einer Verbindungskante zwischen zwei Organisation geschieht nach folgendem Prinzip: Zuerst sendet der Initiator X aus HO_1 an Empfänger Y aus HO_2 eine Anfrage zum Zusammenschluss beider Organisationen. Zu diesem Zeitpunkt ist sich X noch nicht bewusst, dass die Gesamtgröße der beiden Organisationen O_{max} überschreitet. Da in dieser Anfrage die Mitgliederliste von HO_1 mitgeschickt wird, kann Agent Y die Summe $O_{HO_1} + O_{HO_2}$ bilden. Übersteigt diese die maximale Organisationsgröße, so kann kein Zusammenschluss stattfinden. Stattdessen sendet Y den Befehl zur Ausbildung einer Verbindungskante zwischen X und Y an den Initiator, falls sein Vertrauen zu X größer $S_{Netzwerk}$ ist. Ebenfalls sendet er die Liste der Mitglieder von HO_1 an alle Mitglieder seiner Organisation. Auf diese Weise sind alle Agenten von HO_2 über alle mit ihrer Organisation durch eine Kante verbundenen Organisationen informiert. Diese Informationen speichern sie lokal ab. In der folgenden Runde erhält Agent X den Befehl zur Ausbildung der Verbindungskante mit der Liste der Mitglieder von HO_2 im Anhang. Er sendet seinerseits diese Liste an alle Mitglieder aus HO_1, die diese speichern. Somit sind alle Agenten aus HO_1 und HO_2 darüber informiert, dass eine direkte Verbindung zwischen ihren Organisationen existiert.

Da jeder Agent einer Organisation HO eine Liste der durch Kanten mit HO verbundenen Agenten besitzt, wird der Versuch eines beliebigen Agenten X aus HO unterbunden, einen Zusammenschluss mit Agenten aus Organisationen anzustreben, mit denen seine Organisation bereits über ein Kante verbunden ist.

5.3 Zusammenfassung der Eigenschaften der Experimentalumgebung

In diesem Abschnitt fassen wir die Eigenschaften zusammen, die aus der Struktur des Netzes aus holonischen Organisationen sowie den von uns gesetzten Restriktionen beim Zusammenschluss von Agenten direkt folgen. Weitere Eigenschaften des

Spamfilternetzwerkes bezüglich der Effizienz des Austausches von Hashwerten sowie des Ausschlusses böswilliger Agenten werden wir im sechsten Kapitel empirisch nachweisen. Das Netzwerk aus Agenten zeigt folgende Eigenschaften:

- *Geringe durchschnittliche Pfadlänge*
  Wie in Kapitel 5.2.2 gezeigt, lässt sich durch geeignete Wahl der maximalen Organisationsgröße mit $O_{max} \geq \sqrt{N}$, wobei N die Gesamtzahl von Antispam-Agenten ist, von einem beliebigen Agenten X jeder andere Agent im Netzwerk über Pfade mit maximaler Länge drei erreichen. Dies ermöglicht eine schnelle und effiziente Verbreitung der Hashwerte.

- *Einfluss der maximalen Organisationsgröße*
  Die maximale Organisationsgröße O_{max} hat einen entscheidenden Einfluss auf die effizienten Verknüpfung der holonischen Organisationen. Wird O_{max} zu klein gewählt, so bildet sich eine große Anzahl kleiner Organisationen, die nur eine geringe Zahl von Verbindungen zu anderen Organisationen ausbilden können. Eine geringe durchschnittliche Pfadlänge kann nicht mehr garantiert werden.

- *Wahlprotokolle*
  Der Zusammenschluss einzelner Agenten bzw. von holonischen Organisationen benötigt einen hohen Kommunikationsaufwand. Um diesen Aufwand so gering wie möglich zu halten, verwenden die Agenten bei der Entscheidung, ob sie sich zusammenschließen, sowie bei der Verschmelzung zu einer Organisation, die in Kapitel 5.2.3 vorgestellten Wahlprotokolle. Darin wird der Nachrichtenverkehr zwischen den Agenten genau geregelt. Die Befolgung der Protokolle garantiert, dass das Wahlverfahren innerhalb kurzer Rundenzahl mit einer geringen Anzahl von Nachrichten auskommt. Ferner wird gewährleistet, dass nach dem Zusammenschluss eventuell vorhandene unzulässige Verbindungskanten entfernt werden und die Konsistenz des Netzwerkes erhalten bleibt.

Weitere Eigenschaften des Netzwerkes aus Agenten werden wir im folgenden Kapitel experimentell nachweisen. Wir beantworten die Frage, wie viele Spammails tatsächlich durch das Netzwerk aus Antispam-Agenten erkannt werden. Ebenso testen wir, ob die Struktur der Organisationsnetze für einen schnellen und effizienten Versand der Hashwerte sorgt.

Kapitel 6

Experimentelle Evaluation und Analyse

In diesem Kapitel wird das in dieser Arbeit vorgestellte Spamfilternetzwerk anhand der im fünften Kapitel beschriebenen Experimentalumgebung evaluiert. Wir werden bestimmte Variablen der Experimentalumgebung in verschiedenen Versuchen variieren und ihren Einfluss auf die Bildung der Netze aus holonischen Organisationen und den Hashwert-Versand untersuchen. Die Eigenschaften des Netzwerkes aus holonischen Organisationen überprüfen wir empirisch, denn „wir können das Verhalten eines Programms nicht nur dadurch erklären, dass wir es öffnen und seine Struktur sorgfältig durchlesen“ [COHEN 1995a]. Deshalb werden wir das Verhalten des Spamfilternetzwerkes in bestimmten Hypothesen beschreiben und anhand der gemachten Untersuchungsergebnisse validieren. Einen Überblick über alle Variablen und ihren Einfluss auf die Struktur der Agenten innerhalb der Experimentalumgebung geben wir im ersten Kapitel. Danach stellen wir im zweiten Abschnitt dieses Kapitels Hypothesen auf, deren Korrektheit wir in unseren Experimenten testen werden. Es folgt eine Beschreibung der Versuchsanordnung, innerhalb der die Hypothese getestet wird, sowie eine Analyse der beobachteten Resultate. Die wichtigsten Erkenntnisse, die wir aus der Evaluation der Hypothesen gewonnen haben, fassen wir im dritten Abschnitt zusammen. Abschließend werden wir im vierten Abschnitt die Effektivität der Struktur unseres Spamfilternetzwerkes untersuchen. In diesem Zusammenhang weisen wir nach, dass die Antispam-Agenten sich zu einem *small world network* zusammenschließen. Diese Netzwerke zeichnen sich durch geringe durchschnittliche Pfadlängen zwischen beliebigen Knoten mit möglichst geringer Kantenzahl aus. Kurze Pfade sind wichtig für den schnellen Versand der Hashwerte im Netzwerk und deshalb ist die *small world network* Eigenschaft ein wichtiges Indiz für die effiziente Struktur unseres Spamfilternetzwerkes.

6.1 Experimentelles Design und Parameter der Experimentalumgebung

Um die Abhängigkeiten bestimmter Parameter der Experimentalumgebung untereinander herauszufinden, bedienen wir uns der Analysemethoden aus den Sozialwissenschaften [COHEN 1995a]. Wir überprüfen den Einfluss von Faktoren, die wir verändern können auf die Effizienz des Netzwerkes aus Antispam-Agenten beim Vorgang der Klassifizierung von Emails. Wir werden innerhalb der verschiedenen Versuchsanordnungen die unabhängigen Variablen variieren und den Einfluss auf die Funktionsweise des Spamfilternetzwerkes untersuchen. Mit Hilfe der gesammelten Messdaten prüfen wir unsere Hypothesen auf ihre Korrektheit. Die unabhängigen Variablen sind im Verlauf der Experimente direkt beeinflussbar. Wir werden testen, welche Veränderungen sich innerhalb der Experimentalumgebung bei Manipulation der unabhängigen Variablen ergeben. Wir listen nun die unabhängigen Variablen auf und erklären ihren Einfluss auf die Experimentalumgebung.

6.1.1 Maximale Organisationsgröße

Die maximale Organisationsgröße O_{max} gibt die höchstmögliche Anzahl von Agenten an, die sich gleichzeitig innerhalb einer holonischen Organisation HO bewegen dürfen. Innerhalb des HO ist jeder Agent durch eine Kante mit den anderen Agenten der Organisation verbunden. Die Größe von O_{max} legt auch die maximal mögliche Anzahl von Verbindungskanten fest, die die Organisation zu anderen holonischen Organisationen ausbilden darf. Da jeder Agent nur eine Verbindung zu einer anderen Organisation unterhalten darf, beträgt die maximal mögliche Anzahl disjunkter Verbindungen der Organisation HO zu anderen Organisationen O_{max}.

6.1.2 Nachrichtenlimit

Das Nachrichtenlimit NL bestimmt die Anzahl von Hashwerten, die jeder Agent höchstens pro Runde versenden darf. Dabei muss nach Formel 5.1 auf Seite 77 garantiert sein, dass $NL \geq 3\,O_{max} - 1$ ist. Hat ein Agent Hashwerte nach den Regeln aus Kapitel 5.2.2 versendet, so kann er die restlichen Nachrichten zur Kommunikation und dem Vertrauensaufbau mit Agenten außerhalb des Netzes aus holonischen Organisationen verwenden.

6.1.3 Mailabholung$_{min}$ und Mailabholung$_{max}$

Der Zeitrahmen, in dem ein Agent seine Emails vom Mailserver abholt, wird durch eine Zufallszahl bestimmt. Deren Wert bewegt sich zwischen Mailabholung$_{min}$ und Mailabholung$_{max}$ und wird, sobald der Agent die Emails vom Server geladen hat, wieder neu innerhalb des Intervalls der beiden Grenzwerte ausgewürfelt. Damit simulie-

ren wir den Sachverhalt, dass Benutzer ihre Emails in unregelmäßigen Abständen vom Mailserver laden.

6.1.4 Schwellwerte für den Zusammenschluss von Agenten

Die Vertrauenswerte der Agenten untereinander werden zu Beginn der Simulation mit null initialisiert. Im Verlauf der Simulation bauen benevolente Agenten gegenseitiges Vertrauen auf. Übersteigt dieses Vertrauen den Schwellwert S_{HO}, so können die beteiligten Agenten das Wahlverfahren zum Zusammenschluss zu einer holonischen Organisation starten. Dieser Schwellwert kann von uns variiert werden. Ferner ist es auch möglich, den Grenzwert $S_{Netzwerk}$ festzulegen, bei dessen Überschreiten eine Verbindungskante zwischen verschiedenen holonischen Organisationen möglich wird und somit ein Netz aus Organisationen erzeugt werden kann. Je höher die beiden Schwellwerte S_{HO} und $S_{Netzwerk}$ gewählt werden, desto länger dauert es, bis sich die Agenten zu Netzen aus Organisationen zusammenschließen.

6.1.5 Schwellwert $ZV_{Hashwert}$ für die Verwendung von Hashwerten zur Klassifizierung

Mit Hilfe des Schwellwertes $ZV_{Hashwert}$ können wir bestimmen, wie oft ein Agent den gleichen Hashwert empfangen muss, bis er ihn zur Klassifizierung seiner vom Mailserver abgeholten Emails einsetzen darf. Je höher der Wert von $ZV_{Hashwert}$ gewählt ist, desto mehr Agenten müssen den gleichen Hashwert im Netzwerk versenden, bevor dieser zur Erkennung von Spam eingesetzt wird. Im Organisationsnetz aus benevolenten Agenten wird damit verhindert, dass ein ungültiger Hashwert direkt von allen Agenten des Netzwerkes eingesetzt wird. Damit wird die Anzahl falsch klassifizierter Emails auf Grund des Hashwert-Versands stark reduziert. Jedoch verringert sich mit Erhöhung von $ZV_{Hashwert}$ auch die Menge der Spammails, die die Agenten durch Hashwert-Vergleich als Spam erkennen. Ein Antispam-Agent setzt Hashwerte, die er weniger als $ZV_{Hashwert}$ Mal erhalten hat, nicht zur Klassifizierung ein. In unserem Netzwerk aus 60 Agenten hat sich ein Schwellwert von $ZV_{Hashwert} = 2$ bewährt. $ZV_{Hashwert}$ kann in Spamfilternetzwerken mit einer größeren Agentenzahl beliebig erhöht werden.

6.1.6 Parameter der Verteileragenten

Für die Verteileragenten, die die Emails auf die simulierten Mailserver der einzelnen Agenten verteilen, können folgende Parameter festgelegt werden:

- *Sendehäufigkeit*
  Die Sendehäufigkeit legt die Wahrscheinlichkeit fest, mit der ein bestimmter Verteiler pro Runde Emails verschickt.

- *Spam in Prozent*
  Mit diesem Faktor lässt sich der Anteil von Spammails an der Gesamtzahl von Emails festlegen, die der Verteileragent an die simulierten Mailserver der Agenten sendet.

- *Gruppen*
  Mit dieser Einstellung können wir festlegen, an welche Gruppen von Agenten der jeweilige Verteiler seine Emails sendet.

6.1.7 Parameter der Antispam-Agenten

- *Gruppenzugehörigkeit*
  Der Parameter Gruppenzugehörigkeit legt fest, zu welchen Gruppen der Agent gehört. Damit ist auch spezifiziert, von welchen Verteilern der betreffende Agent seine Emails erhält. Der Agent selbst ist sich seiner Zugehörigkeit zu bestimmten Gruppen nicht bewusst.

- *Klassifizierungsfehler*
  Um den Sachverhalt zu simulieren, dass der Klassifizierungsalgorithmus *support vector machines* aus Kapitel 4.6 einen Teil der empfangenen Emails mit einer gewissen Fehlerquote in die Kategorien Spam und Nicht-Spam einteilt, kann der prozentuale Anteil der Emails festgelegt werden, die jeder Agent mit dem SVM-Algorithmus falsch klassifiziert. Je höher der Klassifizierungsfehler f_{class} eines Agenten gewählt ist, desto größer ist die Zahl von Hashwerten, die er fälschlicherweise für Nicht-Spammails generiert.

6.2 Hypothesen

Die im Folgenden aufgestellten Hypothesen haben zum Ziel, die Effizienz des Spamfilternetzwerkes und dessen Aufbaus experimentell zu belegen, sowie bestimmte Eigenschaften nachzuweisen. Die beiden ersten Hypothesen beleuchten die Effizienz des Spamfilternetzwerkes, d.h. wie viele Spammails können tatsächlich anhand des Vergleichs der Hashwerte erkannt werden. In den letzten drei Hypothese weisen wir empirisch bestimmte Eigenschaften des Spamfilternetzwerkes nach. Diese Eigenschaften des Netzwerkes aus Antispam-Agenten werden wir experimentell überprüfen.

Wir behandeln jede Hypothese in einem eigenen Kapitel. Dabei formulieren wir im ersten Abschnitt der Kapitel zuerst die Hypothese. Danach erläutern wir die Hypothese im zweiten Abschnitt genauer. Die Beschreibung der Versuchsanordnung zur Überprüfung der Hypothese findet sich im dritten Abschnitt des jeweiligen Kapitels. Schließlich präsentieren wir im vierten Abschnitt die Resultate, analysieren die Daten

und beurteilen die Hypothese in Hinblick auf ihre Richtigkeit. Die durchgeführten Serien von Simulationen werden zur zuverlässigen statistischen Analyse vielfach durchgeführt, die erhaltenen Werte werden über alle Simulationen einer Serie gemittelt.

6.2.1 Hypothese 1: Effektivität der Organisationsnetze

Hypothese 1 *Hat sich im Laufe der Simulation ein Netz aus holonischen Organisationen gebildet, werden diesem Netzwerk keine weiteren Kanten mehr hinzugefügt und ist die Bildung des Netzes von Organisationen somit abgeschlossen, so können darin enthaltene Agenten einen Teil der Spammails anhand der Hashwert-Einträge ihrer Datenbank identifizieren.*

Erläuterung

Die Funktionalität des Spamfilternetzwerkes soll darin bestehen, möglichst viele Spammails anhand zuvor empfangener korrespondierender Hashwerte zu erkennen. Weiterhin sollen möglichst wenige Nicht-Spammails fälschlicherweise als Spam klassifiziert werden. Zu Beginn der Simulation nimmt jeder Agent nur die selbst generierten Hashwerte in seine lokale Datenbank auf. Haben sich jedoch die Vertrauenswerte der benevolenten Agenten untereinander soweit erhöht, dass Netze aus holonischen Organisationen gebildet werden können, so nimmt ein darin enthaltener Agent die Hashwerte anderer Agenten dieses Netzes ohne Evaluation auf der Basis von Vertrauen auf. Ab diesem Zeitpunkt kann ein Agent einen Hashwert von Mitgliedern seines Organisationsnetzes empfangen und speichern, welcher eine Spammail identifiziert, die er selbst erst in den nächsten Runden von seinem Mailserver abholen wird. Anhand des Hashwertes ist er nach Abholung der Spammail in der Lage, diese eindeutig als Spam zu klassifizieren. Je größer das Organisationsnetz aus Agenten gleicher Gruppenzugehörigkeit ist, desto mehr Hashwerte werden innerhalb dieses Verbundes generiert und ausgetauscht. Damit wächst auch der Anteil von Spammails, die mit Hilfe der ausgetauschten Hashwerte klassifiziert werden können.

Versuchsszenario

In diesem Szenario tauschen 60 Agenten aus einer Gruppe untereinander Hashwerte aus. Wir messen das Volumen von Emails, welches von allen Agenten anhand der Datenbank aus Hashwerten als Spam identifiziert wird. Dabei unterscheiden wir zwischen der Menge von Spammails, die als Spam erkannt werden und der Menge an Nicht-Spammails, die fälschlicherweise als Spam klassifiziert werden. Diese beiden Mengenwerte messen wir für jeweils 100 Runden und tragen den Anstieg in zwei Graphen gegeneinander ab. Innerhalb von 100 Runden werden an die Agenten N_{Spam} = 6000 Spammails und N_{NoSpam} = 6000 Nicht-Spammails gesendet. Das Szenario wird in zwei Versuchsläufen durchgespielt, wobei wir den Wert des Klassifizierungsfehlers f_{class} aller Agenten variieren.

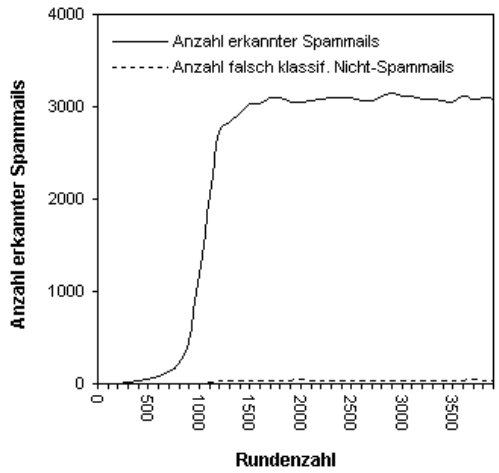

Abb. 6.1: Anzahl der klassifizierten Emails aller Agenten mit $f_{class} = 5\%$

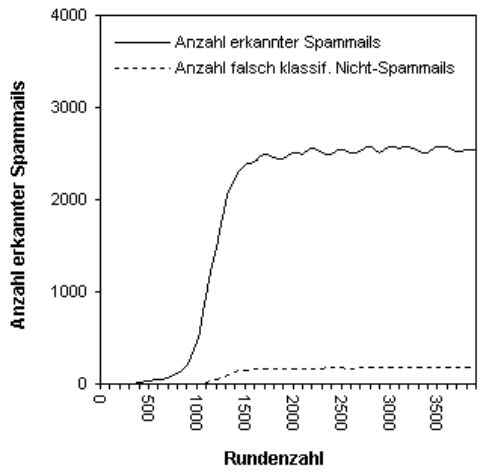

Abb. 6.2: Anzahl der klassifizierten Emails aller Agenten mit $f_{class} = 15\%$

Auswertung und Analyse

Wir verfolgen die Anzahl erkannter Spammails pro 100 Runden sowie die Anzahl der als Spam klassifizierten Nicht-Spammails. Wie man an den Graphen 6.1 und 6.2 erkennen kann, werden zu Beginn der Simulation keine Emails anhand des Hashwert-Vergleichs klassifiziert. Die im Netzwerk versendeten Hashwerte werden ausschließlich zum Vertrauensaufbau verwendet, jedoch nicht zur Klassifizierung. Nach 1000 Runden sind die Schwellwerte für die Bildung von Organisationen und Organisationsnetzen überschritten und bis zur Runde 2000 ist hat sich Netz aus Organisationen gebildet, in dem die Agenten zusammengeschlossen sind. Die im Organisationsnetz versendeten Hashwerte werden nun auch in den Datenbanken der Agenten gespeichert und zur Klassifizierung von Emails verwendet. Je mehr Agenten dem Organisationsnetz beitreten, desto größer ist die Anzahl der darin versendeten Hashwerte. Die Menge erkannter Spammails anhand des Hashwert-Vergleichs steigt, wie in den Graphen abgebildet, von Runde 1000 bis 2000 während der Bildung des Organisationsnetzes stetig an und manifestiert sich auf hohem Niveau. Die Anzahl von Spammails, die durch den Hashwert-Vergleich erkannt werden, ist im Netzwerk aus Agenten mit geringerem Klassifizierungsfehler $f_{class} = 5\%$ höher als im Netzwerk der Agenten mit $f_{class} = 15\%$.

In Anlehnung an ANDROUTSOPOULOS et al. [2000] bezeichnen wir mit $n_{E \to F}$ die Anzahl von Emails, die zur Kategorie E gehören und die der Filter der Kategorie F zugeordnet hat mit $E, F \in \{NoSpam, Spam\}$. Wir fassen in Tabelle 6.1 den Anteil der Spammails zusammen, der allein durch die Klassifizierung anhand der Hashwerte erkannt wird. Die Zahlen beziehen sich dabei auf die Mengen von Spammails und Nicht-Spammails, die nach Abschluss der Bildung des Organisationsnetzes durchschnittlich in der Zeitspanne von 100 Runden an alle Agenten des Netzes versendet werden, und dem Anteil davon, der als Spam klassifiziert wird.

f_{class}	$Spammails$			$Nicht - Spammails$		
	$n_{Spam \to Spam}$	N_{Spam}	$Prozent$	$n_{NoSpam \to Spam}$	N_{NoSpam}	$Prozent$
5	3092	6000	51.53 %	35	6000	0.58 %
15	2539	6000	42.31 %	180	6000	3.01 %

Tabelle 6.1: Anteil der Spam- und Nicht-Spammails, die durch Hashwert-Vergleich als Spam klassifiziert werden

Das Spamfilternetzwerk erkennt je nach Klassifizierungsfehler der Agenten etwa die Hälfte der Spammails anhand des Hashwert-Vergleichs. Der Anteil erkannter Spammails an den gesamt versendeten Spammails wird in der Literatur auch *recall* genannt. Die restlichen Spammails werden an den Algorithmus zur Textklassifizierung *support vector machines* weitergeleitet. In der folgenden Tabelle 6.2 notieren wir für beide Filter die Anzahl $n_{Spam \to NoSpam}$ von Spammails pro 100 Runden, die fälschlicherweise als Nicht-Spam klassifiziert werden. Ebenso messen wir die Anzahl $n_{NoSpam \to Spam}$ von Nicht-Spammails, die SVM und Hashwert-Vergleich als Spam klassifizieren.

f_{class}	$Hashwert - Vergleich$		SVM	
	$n_{Spam \to NoSpam}$	$n_{NoSpam \to Spam}$	$n_{Spam \to NoSpam}$	$n_{NoSpam \to Spam}$
5	0	35	146	298
15	0	180	519	873

Tabelle 6.2: Anzahl der Emails, die durch Hashwert-Vergleich und SVM falsch klassifiziert werden

Die Variable $n_{Spam \to NoSpam}$ des Filters durch Hashwert-Vergleich besitzt den Wert null, da dieser Filter nur Emails in die Kategorie $Spam$ einteilt. Den Klassifizierungsfehler des Spamfilternetzwerkes errechnen wir mit Hilfe der folgenden Formel aus [ANDROUTSOPOULOS et al. 2000]:

$$f_{class}(Spamfilternetzwerk) = \frac{n_{NoSpam \to Spam} + n_{Spam \to NoSpam}}{N_{Spam} + N_{NoSpam}} \qquad (6.1)$$

Wir addieren den Wert von $n_{Spam \to NoSpam}$ des Filters durch Hashwert-Vergleich und den Wert $n_{Spam \to NoSpam}$ des *support vector machines* Filters. Ebenso addieren wir die beiden Werte der Filter für $n_{NoSpam \to Spam}$. Die beiden Summen sind in Tabelle 6.3 aufgeführt. Durch Einsetzen der Zahlen aus Tabelle 6.3 in Formel 6.1 können wir den Klassifizierungsfehler des gesamten Spamfilternetzwerkes mit den zwei Filtern Hashwert-Vergleich und *support vector machines* errechnen. Das wichtigste Ergebnis unserer experimentellen Evaluation erhalten wir durch Vergleich des Klassifizierungsfehlers des *support vector machines* Algorithmus der einzelnen Agenten f_{class} mit dem Klassifizierungsfehler $f_{class}(Spamfilternetzwerk)$ des gesamten Spamfilternetzwerkes.

f_{class}	$Spamfilternetzwerk$		
	$n_{Spam \rightarrow NoSpam}$	$n_{NoSpam \rightarrow Spam}$	$f_{class}(Spamfilternetzwerk)$
5	146	333	**3.99** %
15	519	1053	**13.10** %

Tabelle 6.3: Anzahl falsch klassifizierter Emails beider Filter und Klassifizierungsfehler des Spamfilternetzwerkes

Der Klassifizierungsfehler des gesamten Spamfilternetzwerkes ist geringer als der des SVM-Algorithmus der einzelnen Agenten. Durch die Kombination von *support vector machines* und Multiagentensystem wird die Klassifizierungsgenauigkeit der Antispam-Agenten also erhöht. Der Anteil von Emails, die die Antispam-Agenten falsch klassifizieren, sinkt durch die Einbettung der Spamfilter der Benutzer in ein Multiagentensystem. In diesem Versuch haben wir gezeigt, dass die Antispam-Agenten durch Kommunikation miteinander und Austausch der Hashwerte ihren Klassifizierungsfehler verringern.

6.2.2 Hypothese 2: Effizienz des Austausches der Hashwerte

Hypothese 2 *Innerhalb der zusammenhängenden Netze aus holonischen Organisationen findet trotz Einschränkung der Kommunikation zwischen den Agenten mit Hilfe des Parameters $Nachrichtenlimit$ ein schneller Austausch von Hashwerten zwischen einer großen Zahl von Agenten statt. Für jede versendete Spammail an Agenten des Organisationsnetzes messen wir den prozentualen Anteil von Agenten, die die Spammail anhand des korrespondierenden Hashwertes in ihrer Datenbank erkennen. Dies bedeutet, dass ihnen der Hashwert von anderen Agenten bereits in einer vorherigen Runde zugeschickt wurde. Je höher dieser Anteil ist, desto effektiver wurde dieser Hashwert im Netzwerk verteilt.*

Erläuterung

Durch den Nachweis dieser Hypothese soll gezeigt werden, das sich innerhalb der Struktur des Spamfilternetzwerkes, dass die Agenten durch den Aufbau gegenseitigen Vertrauens bilden, Informationen über Spam in Form von Hashwerten schnell verbreiten lassen. Der Zweck des Netzwerkes besteht darin, möglichst viele Hashwerte in kurzer Rundenzahl an möglichst viele Agenten zu schicken. Ein Agent kann eine Email anhand seiner Datenbank nur als Spam klassifizieren, wenn er den korrespondierenden Hashwert bereits durch andere Agenten erhalten hat. Deshalb ist es die Aufgabe des Netzwerkes aus Antispam-Agenten, für eine vom Verteileragenten verschickte Spammail schnellstmöglich den entsprechenden Hashwert zu erzeugen und zu versenden. Die Agenten, die die Spammail nach deren Generierung zuerst vom Verteiler laden, müssen sie mit Hilfe des Textklassifizierungsalgorithmus SVM erkennen. Eine Erken-

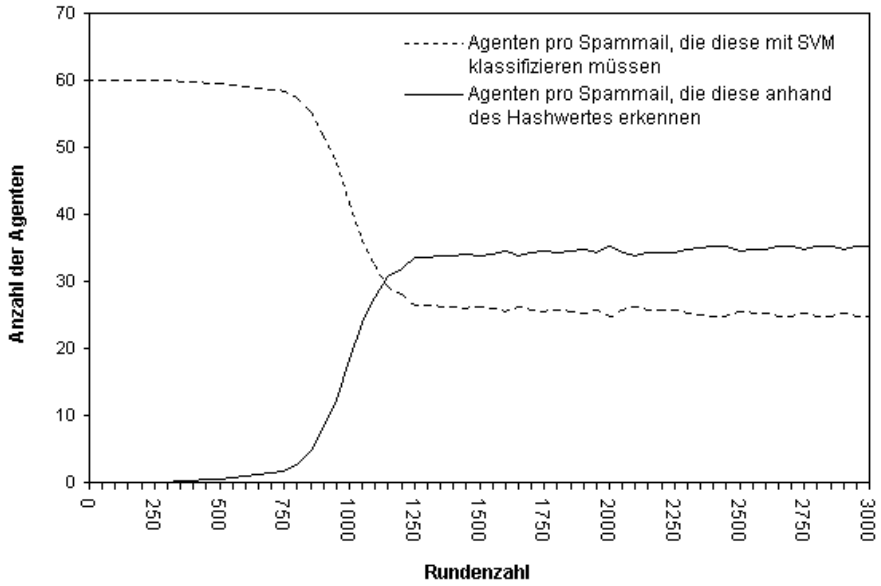

Abbildung 6.3: Anzahl der klassifizierten Spammails durch Hashwert-Vergleich und SVM

nung anhand der Datenbank ist nicht möglich, da der Hashwert noch nicht von anderen Agenten des Netzes versendet wurde. Erkennt der Textklassifizierungsalgorithmus die Email als Spam, so wird der Hashwert durch Ausführung der Regeln aus Kapitel 5.2.2 an andere Agenten des Netzes gesendet. Erst in den folgenden Runden können andere Agenten von dem sich im Netzwerk verbreitenden Hashwert profitieren, indem sie diesen zur Klassifizierung der gleichen Spammail nach deren Abholung vom Verteiler einsetzen. Je weniger Agenten gezwungen sind, die Spammail mit Hilfe des Textklassifizierungsalgorithmus anstatt durch die Hashwerte aus ihrer Datenbank zu erkennen, desto effektiver ist der Versand der entsprechenden Hashwerte.

Versuchsszenario

Diesem Szenario gehören 60 Agenten mit $f_{class} = 2\%$ an. Wir messen während der Simulation für jeden Spam-Hashwert, wie groß die Anzahl von Agenten ist, die die zu diesem Hashwert gehörende Spammail anhand ihrer Datenbank mit Hilfe des Hashwert-Vergleichs als Spam klassifizieren konnten. Dagegen messen wir zusätzlich für jeden Hashwert die Zahl von Agenten, die zur Klassifizierung der korrespondierenden Spammail den *support vector machines* Algorithmus einsetzen mussten.

Auswertung und Analyse

Wie in der ersten Hypothese gezeigt, liegt der Klassifizierungsfehler beim Erkennen von Spam anhand des Hashwert-Vergleichs unter dem Fehler von *support vector machines*. Je mehr Spammails also durch Hashwert-Vergleich erkannt werden, desto

weniger müssen an den Algorithmus *support vector machines* mit höherem Klassifizierungsfehler weitergeleitet werden. In Diagramm 6.3 ist zum einen der Verlauf der Kurve gezeichnet, welche den Anteil der Agenten darstellt, die die in der jeweiligen Runde erzeugte Spammail anhand des Hashwert-Vergleichs erkennen. Zum anderen zeigt die zweite Kurve die Menge von Agenten, die zur Klassifizierung der Spammail statt des Hashwert-Vergleichs den *support vector machines* Algorithmus einsetzen mussten, weil der zu der Spammail korrespondierende Hashwert nicht in ihrer Datenbank vorhanden war. Die in den beiden Kurven dargestellte Anzahl von Agenten addiert sich in jedem Punkt der X-Achse zu 60.

Zu Beginn der Simulation werden die versendeten Hashwerte nur zur Vertrauensbildung herangezogen und daraufhin wieder verworfen. Damit ist jeder der 60 Agenten gezwungen, die empfangene Spammail mit Hilfe des *support vector machines* Algorithmus zu identifizieren. Je größer die Anzahl von Agenten ist, die sich ab Runde 500 im Netz aus Organisationen zusammenschließen, desto mehr Hashwerte können in den lokalen Datenbanken der Agenten gespeichert und zur Klassifizierung von Spammails eingesetzt werden. Nach Abschluss der Bildung des Organisationsnetzes können mehr als die Hälfte der Spammails allein durch den Vergleich ihrer Hashwerte mit den Einträgen der lokalen Datenbanken der Antispam-Agenten erkannt werden. Eine Rate von über 50 Prozent erkannter Spammails allein durch den Vergleich der Hashwerte untermauert die aufgestellte Hypothese, dass Hashwerte schnell und effizient im Spamfilternetzwerk trotz Einschränkung der Kommunikation durch Einhaltung des Nachrichtenlimits verteilt werden. Die Struktur des Spamfilternetzwerkes ermöglicht kurze Pfade zwischen beliebigen Knoten über möglichst wenige Kanten. Netzwerke mit diesen Eigenschaften werden auch als *small world* Netzwerke bezeichnet. Wir werden die *small world* Eigenschaft des Spamfilternetzwerkes in Kapitel 6.3 nachweisen und damit einen weiteren Beleg für die effiziente und schnelle Datenübertragung innerhalb des Netzwerkes liefern.

6.2.3 Hypothese 3: Anzahl der Verbindungen eines Agenten in Abhängigkeit von der Gruppenzugehörigkeit

Hypothese 3 *Ein Agent X mit Zugehörigkeit zu mehreren Gruppen (Mischagent) geht schneller Verbindungen mit anderen Agenten ein als ein Agent Y, der nur einer Gruppe angehört. Die Zugehörigkeit zu mehreren Gruppen verschafft X auch eine größere Zahl von Verbindungen zu anderen Agenten als X.*

Erläuterung

Wir bezeichnen mit X_{AB} den Mischagenten, der den Gruppen A und B angehört. Ferner seien ein Agent Y_A gegeben, der nur zur Gruppe A gehört, sowie ein Agent Z_B aus Gruppe B. X_{AB} erhält alle Emails, die von den Verteileragenten an die Gruppen

A und B geschickt werden. Dadurch erhöht sich die Menge von Spammails, die dieser Agent empfängt. Deshalb vermuten wir, dass X_{AB} häufiger Hashwerte generieren und verschicken kann als die Agenten, die nur einer Gruppe zugeordnet sind. Durch den erhöhten Versand von Hashwerten können andere Agenten verstärkt Vertrauen zu X_{AB} aufbauen. Auch kann Agent X_{AB} einen größeren Teil der Hashwerte, die er erhält, evaluieren, da er die Emails aller Gruppen empfängt. Wir vermuten, dass diese beiden Faktoren dazu führen, dass sich die Vertrauenswerte anderer Agenten zu Agent X_{AB} stärker erhöhen als beispielsweise zu Agent Y_A, der nur zu einer Gruppe gehört. X_{AB} bildet auf Grund der höheren Vertrauenswerte frühzeitig Verbindungen zu anderen Agenten aus, da die Schwellwerte für die Bildung von Verbindungen schneller überschritten werden als bei Y_A. Da Mischagent X_{AB} zu allen Gruppen gehört, kann er zu allen Agenten Vertrauen aufbauen. Agent Y_A hingegen erhöht jedoch nur die Vertrauenswerte zu den Agenten seiner eigenen Gruppe. Somit bieten sich für X_{AB} ein breiteres Spektrum von Agenten für den Aufbau von Verbindungen an, als für die Agenten Y_A oder Z_B. Deshalb vermuten wir, dass X_{AB} eine größere Zahl von Verbindungen ausbildet als die anderen beiden Agenten.

Versuchsszenario

Wir teilen 60 Agenten gleichmäßig in die Gruppen A, B und AB ein. Weiterhin erhöhen wir in den verschiedenen Durchläufen schrittweise die Schwellwerte für den Zusammenschluss der Agenten zu Organisationen und Netzen aus Organisationen. Während im ersten Versuchslauf ein Schwellwert von $S_{HO} = 6$ zur Bildung einer Organisation genügt, muss im vierten Lauf ein Vertrauenswert von $S_{HO} = 30$ überschritten werden. Auch der Schwellwert für die Ausbildung von Verbindungskanten zwischen den Organisationen wird in den Versuchsläufen von $S_{Netzwerk} = 10$ auf $S_{Netzwerk} = 35$ erhöht. Wir messen die Rundenzahl, nach der die Agenten aus den Gruppen A, B und AB im Schnitt Verbindungen zu anderen Agenten eingehen. Ebenso messen wir den prozentualen Anteil von Agenten, mit denen die einzelnen Agenten der drei Gruppen nach Abschluss der Simulation über ein Organisationsnetz und damit über Pfade maximaler Länge $l_{path} = 3$ verbunden sind.

Auswertung und Analyse

Die dritte Hypothese kann durch die Auswertung der Testdaten nicht belegt werden. Stattdessen ist genau das Gegenteil der erwarteten Reaktionen der Agenten eingetreten. Die Mischagenten gehen später Verbindungen zu anderen Agenten ein als diejenigen Agenten, die nur der Gruppe A oder der Gruppe B angehören (Abb. 6.4). Weiterhin bilden sie weniger Verbindungskanten zu anderen Agenten aus als die Agenten, die nur einer Gruppe angehören (Abb. 6.5).

Dieses unerwartete Resultat können wir uns erklären, indem wir den Anstieg der Vertrauenswerte zwischen den Mischagenten und den Agenten aus den Gruppen A und B

analysieren. Wir tragen dazu die möglichen Kombinationen aus Sendern und Empfängern beim Hashwert-Versand in Tabelle 6.4 ein und errechnen die Wahrscheinlichkeiten, mit denen die verschiedenen Agentengruppen Vertrauenswerte zueinander bilden. Seien X_{AB} ein Mischagent und Y_A und Z_B Vertreter der beiden restlichen disjunkten Gruppen. Dann ist der Anteil von Hashwerten, die ein Agent von anderen Agentengruppen evaluieren kann, abhängig von der Gruppenzugehörigkeit von Sender und Empfänger des versendeten Hashwertes.

Versand	Evaluierbare Hashwerte	Vertrauenserhöhung
$Y_A \rightarrow Y'_A$	100 %	++
$Z_B \rightarrow Y'_A$	0 %	-
$X_{AB} \rightarrow Y'_A$	50 %	+
$Y_A \rightarrow Z'_B$	0 %	-
$Z_B \rightarrow Z'_B$	100 %	++
$X_{AB} \rightarrow Z'_B$	50 %	+
$Y_A \rightarrow X'_{AB}$	100 %	++
$Z_B \rightarrow X'_{AB}$	100 %	++
$X_{AB} \rightarrow X'_{AB}$	100 %	++

Tabelle 6.4: Anteil evaluierbarer Hashwerte in Abhängigkeit von der Gruppenzugehörigkeit des Senders und Empfängers

Wir gehen zur Veranschaulichung davon aus, dass die Datenbanken der Agenten mit gleicher Gruppenzugehörigkeit dieselben Hashwerte enthalten. Ferner verfüge der Mischagent in seiner Datenbank über Hashwerte beider Gruppen A und B. Da die Agenten gleichmäßig in den drei Gruppen verteilt sind und sich zufällig bis zur Bildung von holonischen Organisationen Hashwerte zusenden, ist die Wahrscheinlichkeit aller neun Versandmöglichkeiten zum Vertrauensaufbau gleich groß. Jeder Agent verteilt die von ihm versendeten Hashwerte gleichmäßig auf die Agenten der drei Gruppen A, B und AB. Jedoch unterscheiden sich die Wahrscheinlichkeiten, mit denen die Agenten der drei Gruppen empfangene Hashwerte evaluieren können. So kann der Mischagent X_{AB} die Hashwerte aller Agenten auswerten und erhöht deshalb seine Vertrauenswerte zu den Agenten der Gruppen A, B und AB gleichmäßig. Agent Y_A erhöht das Vertrauen zu Agenten seiner eigenen Gruppe am stärksten. Das Vertrauen zu den Mischagenten erhöht Y_A nur im moderaten Maße, da er nur die Hälfte der Hashwerte von X_{AB} evaluieren kann, die aus Spammails an die Gruppe A stammen. Die Hashwerte, die X_{AB} aus Spam an die Gruppe B generiert hat, kann Y_A nicht evaluieren. Ebenso kann er keine Hashwerte von Z_B auswerten, da sie sich in disjunkten Gruppen befinden. Entsprechendes gilt umgekehrt auch für Agent Z_B. Die Agenten Y_A und Z_B erhöhen somit das Vertrauen zu Agenten ihrer eigenen Gruppe am stärksten. Deshalb verbinden sich auch zuerst und bevorzugt mit Agenten aus ihrer eigenen Gruppe.

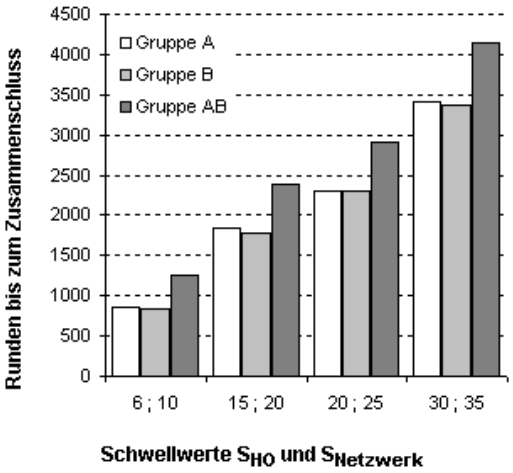

Abb. 6.4: Runden bis Zusammenschluss

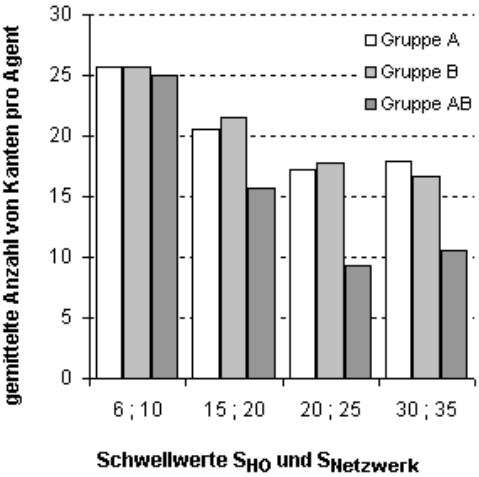

Abb. 6.5: Anzahl der Verbindungen

Da die Mischagenten das Vertrauen zu den Agenten aus allen drei Gruppen A, B und AB gleichmäßig erhöhen, können sie nicht gezielt verstärkt Vertrauen zu anderen Agenten aus ihrer Gruppe AB aufbauen, wie dies die Agenten mit Gruppenzugehörigkeit zu einer Gruppe tun. Dadurch verlangsamt sich der Zusammenschluss der Mischagenten mit anderen Agenten. Am Ende der Simulation sind die Mischagenten prozentual mit weniger Agenten im Netzwerk verbunden als die Agenten mit Gruppenzugehörigkeit zu einer Gruppe (siehe Abb. 6.5). Dieser Effekt verstärkt sich durch Erhöhung der Schwellwerte S_{HO} und $S_{Netzwerk}$.

Der Mischagent X_{AB} hat keine Präferenzen bei der Wahl des Agenten, mit dem er sich zusammenschließen will. Da aber die Agenten Y_A und Z_B innerhalb ihrer eigenen Gruppe bevorzugt Verbindungen eingehen, bleibt X_{AB} nichts anderes übrig, als sich in vielen Fällen mit anderen Mischagenten zu vereinigen. Die gemittelte Rundenzahl, bei der die Agenten aus den Gruppen A, B und AB Verbindungen eingehen, wird in Abb. 6.4 abgebildet. Man erkennt, dass die Agenten mit Gruppenzugehörigkeit zu einer Gruppe nach einer geringeren Rundenzahl Verbindungen eingehen als die Mischagenten. Je größer die Schwellwerte S_{HO} und $S_{Netzwerk}$ gewählt werden, desto später verbinden sich die Agenten.

Da sich die Agenten mit Zugehörigkeit zu einer Gruppe bevorzugt verbinden, sind die Mischagenten gezwungen, sich mit den übrigen Agenten, den Mischagenten gleicher Gruppenzugehörigkeit, zu verbinden. Somit bilden sich nur wenige Organisationen, in denen Agenten der Gruppen A und AB, bzw. B und AB gemischt vorkommen. Die Mischagenten verteilen sich also nicht gleichmäßig auf alle Organisationen, sondern sammeln sich meist in separaten Organisationen. So finden sich durch Selbstorganisation auch Agenten zusammen, die aus mehreren gleichen Quellen Spammails beziehen.

6.2.4 Hypothese 4: Abhängigkeit zwischen Klassifizierungsfehler und Vertrauen

Hypothese 4 *Je größer der Klassifizierungsfehler eines Agenten X ist, desto weniger Vertrauen bauen andere benevolente Agenten zu ihm auf. Deshalb sinkt mit steigendem Klassifizierungsfehler von X die Anzahl von Verbindungen, die andere benevolente Agenten mit X eingehen.*

Erläuterung

Ein Agent X mit hohem Fehlerwert f_{class} bei der Klassifizierung von Emails mit Hilfe von *support vector machines* erkennt einen hohen Prozentsatz der Spammails nicht, bzw. klassifiziert Nicht-Spammails als Spam. Das führt dazu, dass ein großer Anteil der generierten Hashwerte nicht von den anderen Agenten mit niedrigem Klassifizierungsfehler aus der gleichen Gruppe erkannt und evaluiert werden und vice versa. So können benevolente Agenten des Spamfilternetzwerkes nur langsam Vertrauen zu Agent X aufbauen. Deshalb wird auch nur schwerlich der Schwellwert S_{HO} erreicht, bei dessen Überschreiten eine Verbindung zwischen benevolenten Agenten und Agent X initiiert wird. Dies ist der Grund dafür, dass Agenten mit hohem Klassifizierungsfehler aus den Netzen der holonischen Organisationen benevolenter Agenten ausgeschlossen werden. Damit bietet die Plattform gleichzeitig den Anreiz für Agenten, einen möglichst geringen Klassifizierungsfehler anzustreben, sofern der größte Teil der Agenten des Netzwerkes benevolent ist.

Versuchsszenario

Dieses Szenario enthält 60 Agenten, wobei 40 dieser Agenten (Gruppe $GOOD$) den Klassifizierungsfehler $f_{class} = 5\%$ aufweisen. Der Klassifizierungsfehler der restlichen 20 Agenten (Gruppe BAD) ist ungleich höher, nämlich $f_{class} = 90\%$. Die Agenten der Gruppe BAD simulieren durch ihren hohen Klassifizierungsfehler das Verhalten böswilliger Agenten, d.h. sie speisen überwiegend Hashwerte ins Netzwerk ein, zu denen keine Spammails existieren. Dieses Szenario wird in vier Versuchsläufen durchgespielt, wobei wir den Schwellwert zur Bildung von Organisationen S_{HO} und den Wert zur Bildung von Organisationsnetzen $S_{Netzwerk}$ variieren.

Auswertung und Analyse

Die vier Versuchsdurchläufe liefern für verschiedene Schwellwerte S_{HO} und $S_{Netzwerk}$ die Ergebnisse, die in Tabelle 6.5 zusammengefasst sind. $GOOD$ ist die Gruppe der benevolenten Agenten mit $f_{class} = 5\%$ und BAD die Gruppe mit $f_{class} = 90\%$. In der Tabelle tragen wir den prozentualen Anteil der benevolenten und böswilligen Agenten ab, mit der ein beliebiger Agent aus $GOOD$ im Schnitt verbunden ist. Dieselben Werte finden sich ebenso für die Agenten der Gruppe BAD.

$Lauf$	S_{HO}	$S_{Netzwerk}$	$Verbind.GOODzu$		$Verbind.BADzu$	
			$GOOD$	BAD	BAD	$GOOD$
1	3	5	99.65 %	88.96 %	100 %	88.96 %
2	5	10	99.48 %	61.96 %	100 %	61.96 %
3	10	15	99.65 %	9.50 %	100 %	9.50 %
4	20	25	100 %	0 %	100 %	0 %

Tabelle 6.5: Anteil der Agenten aus $GOOD$ und BAD, die sich bei Variierung der Schwellwerte S_{HO} und $S_{Netzwerk}$ zusammenschließen

Die Aussage der Hypothese, dass Agenten mit hohem Klassifizierungsfehler nicht in das Organisationsnetz der Agenten mit niedrigem f_{class} aus der Gruppe $GOOD$ aufgenommen werden, können wir anhand der Versuchsergebnisse bestätigen. Die Agenten der Gruppe $GOOD$ verbinden sich fast vollständig miteinander, d.h. jeder benevolente Agenten kann jeden anderen Agenten aus $GOOD$ über Pfade mit maximaler Länge $l_{path} = 3$ erreichen[1]. Auch die 20 Agenten der Gruppe BAD finden sich zu einem eigenen Organisationsnetz zusammen. Jeder Agent aus BAD klassifiziert 90 Prozent der Nicht-Spammails fälschlicherweise als Spam, generiert dafür Hashwerte und versendet diese an andere Agenten. Diese ungültigen Hashwerte können größtenteils nur von Agenten der Gruppe BAD evaluiert werden.

Somit sorgt der Hashwert-Versand und damit verbundene Vertrauensaufbau dafür, dass sich die Agenten der beiden Gruppen zu zwei Netzen von Organisationen zusammenschließen. Wählt man die Schwellwerte S_{HO} und $S_{Netzwerk}$ wie im vierten Versuchslauf groß genug, so kann man erreichen, dass die beiden Organisationsnetze vollständig voneinander isoliert sind, d.h. es existieren keine Kanten zwischen den Netzen aus benevolenten und böswilligen Agenten. Je geringer die Schwellwerte gewählt sind, desto höher ist die Wahrscheinlichkeit, dass sich Agenten der beiden Gruppen miteinander verbinden. Dies geschieht, weil die Agenten der Gruppe $GOOD$ fünf Prozent der Emails falsch klassifizieren, d.h. sie klassifizieren im geringen Maße auch einige Nicht-Spammails als Spam. Die daraus erzeugten Hashwerte können von den Agenten aus BAD evaluiert werden und es baut sich Vertrauen zwischen Agenten der beiden Gruppen auf. Entsprechendes gilt für die Agenten aus BAD, da sie 10 Prozent der Emails richtig klassifizieren. Sind die Schwellwerte wie im ersten Durchlauf praktisch auf null gesetzt, so verbinden sich die Agenten aus $GOOD$ und BAD komplett miteinander. Geringe Schwellwerte sind daher ungeeignet zur effizienten Trennung der Agentengruppen mit unterschiedlichem Klassifizierungsfehler. Deshalb müssen zur Trennung der benevolenten von den böswilligen Agenten die Schwellwerte S_{HO} und $S_{Netzwerk}$ größer 20 gewählt werden, wie im vierten Durchlauf geschehen.

[1] Abzulesen in der Tabelle 6.5 in Spalte vier

6.2.5 Hypothese 5: Anzahl evaluierbarer Hashwerte

Hypothese 5 *Der Anteil evaluierbarer Hashwerte am Gesamtvolumen der versendeten Hashwerte steigt im Verlauf der Simulation an, bis sich die benevolenten Agenten zu Netzen aus holonischen Organisationen zusammengefunden haben. Ist die Ausbildung von Kanten beendet, verbleibt die Anzahl evaluierbarer Hashwerte auf konstant hohem Niveau.*

Erläuterung

Ein wichtiger Indikator für den effizienten Versand der Hashwerte innerhalb des Spamfilternetzwerkes stellt die Anzahl der evaluierbaren Hashwerte des Gesamtsystems dar. In Kapitel 5.2.1 wurde gezeigt, dass Agenten nur Hashwerte anderer Agenten evaluieren können, wenn sie benevolent sind und mindestens einer gemeinsamen Gruppe angehören.

Zu Beginn der Simulation senden die Agenten ihre generierten Hashwerte an zufällig gewählte Agenten des Netzwerkes. Agenten aus disjunkten Gruppen können die Werte, die sie sich zusenden, nicht evaluieren, da sie die korrespondierenden Emails nicht erhalten. Damit ist auch die Bandbreite zum Versenden der Hashwerte verschenkt. Agenten gleicher Gruppenzugehörigkeit schließen sich im Verlauf der Simulation zu Netzen aus holonischen Organisationen zusammen. Allen Agenten dieses Netzes aus Organisationen ist gemeinsam, dass sie sich mindestens in einer gemeinsamen Gruppe befinden. Das bedeutet wiederum, dass sie mindestens einen gemeinsamen Verteileragenten besitzen, welcher ihnen Emails zusendet. Sie erzeugen aus der gleichen Email identische Hashwerte. Seien die benevolenten Agenten X und Y aus der gleichen Gruppe. Sei weiterhin X derjenige Agent, der zuerst seine Emails vom Verteileragenten abholt. Dann generiert X aus den empfangenen Spammails Hashwerte und speichert sie in seiner Datenbank ab. Agent Y holt seine Emails in einer der folgenden Runden ab und erzeugt daraus ebenfalls Hashwerte. Sendet er diese an Agent X, so kann X die Hashwerte evaluieren, d.h. er findet in seiner lokalen Datenbank identische Hashwerte und das Vertrauen von X zu Y wird erhöht. Je höher die Anzahl gemeinsamer Verteileragenten von X und Y ist, desto mehr Hashwerte können sie evaluieren. Mit wachsender Anzahl gemeinsamer Verteileragenten bauen X und Y schneller Vertrauen zueinander auf.

Innerhalb des Netzes aus Organisationen sind die Agenten nach den Regeln aus Kapitel 5.1.2 verpflichtet, sich gegenseitig zusätzliche Hashwerte zu schicken. So steigt mit der Bildung der holonischen Organisationen und der Verbindungen zwischen ihnen die Anzahl evaluierbarer Hashwerte stetig an. Weiterhin vergleichen wir den Anstieg der Anzahl evaluierbarer Hashwerte in einem Szenario mit Agenten aus derselben Gruppe mit Szenarien, in denen zwei und drei Gruppen verschiedener Agenten existieren. Die Agenten der verschiedenen Gruppen empfangen verschiedene Emails. Deshalb kann

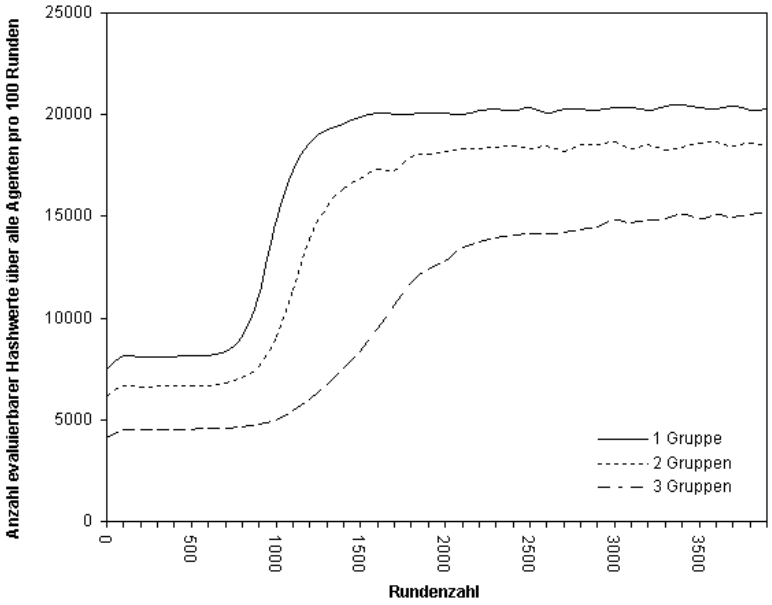

Abbildung 6.6: Anzahl evaluierbarer Hashwerte verschiedener Gruppen von Agenten

ein Agent X aus Gruppe A niemals einen Hashwert eines Agenten Z aus Gruppe B auswerten und vice versa. Somit ist zu erwarten, dass die Anzahl evaluierbarer Hashwerte im Szenario mit Agenten gleicher Gruppenzugehörigkeit höher liegt als in den Szenarien mit Agenten aus verschiedenen Gruppen.

Versuchsszenario

Die Summe der evaluierbaren Hashwerte aller Antispam-Agenten wird im Verlauf der Simulation alle 100 Runden gemessen. Die Simulation wird in drei verschiedenen Versuchsanordnungen mit verschiedenen Agentengruppen wiederholt und der Anstieg des Volumens evaluierbarer Hashwerte in einem gemeinsamen Schaubild zusammengefasst. In der ersten Versuchsanordnung existiert nur eine einzige Gruppe von Agenten, in der zweiten Versuchsanordnung existieren zwei Gruppen von Agenten und in der letzten drei verschiedene Gruppen von Agenten. Die genaue Aufsplittung der Agenten in die verschiedenen Gruppen ist im Anhang detailliert beschrieben. Zu beachten hierbei ist die Tatsache, dass in den Szenarien mit mehreren Agentengruppen auch Mischagenten existieren, d.h. Agenten die gleichzeitig in mehreren Gruppen vorhanden sind. Das Szenario enthält innerhalb jeder Versuchsanordnung eine Gesamtzahl von 60 Agenten.

Auswertung und Analyse

Zu Beginn der Simulation senden die Agenten Hashwerte zufällig an andere Agenten. Im Szenario mit Agenten aus nur einer Gruppe kann jeder Agent Hashwerte ande-

rer Agenten evaluieren, sofern er sie selbst in einer vorhergehenden Runde generiert hat. In den anderen beiden Szenarien können Agenten aus Gruppe A beispielsweise Hashwerte von Agenten der Gruppe B nicht evaluieren. Deshalb ist die Anzahl evaluierbarer Hashwerte im Szenario mit einer Agentengruppe, wie in Abb. 6.6 ersichtlich, am höchsten. Bis zur Runde 700 haben sich die Vertrauenswerte der Agenten untereinander soweit erhöht, dass die Schwellwerte S_{HO} und $S_{Netzwerk}$ zur Bildung von Netzen aus Organisationen überschritten werden. Innerhalb der sich ab diesem Zeitpunkt bildenden Organisationsnetze werden nun verstärkt Hashwerte getauscht. Diese Werte können aufgrund der gleichen Gruppenzugehörigkeit innerhalb von Organisationen auch evaluiert werden. So steigt die Zahl evaluierbarer Hashwerte in allen drei Szenarien an. Die Bildung der Organisationsnetze ist größtenteils bis zur Runde 2000 abgeschlossen, was man in Abb. 6.6 anhand des konstanten Verlaufs der Kurven ab Runde 2000 erkennen kann. Der Zwang der Agenten, in den Organisationsnetzen verstärkt Hashwerte zu versenden und die gleiche Gruppenzugehörigkeit der Agenten in den Organisationen führt dazu, dass die Zahl evaluierbarer Hashwerte bis zum Ende der Simulation auf hohem Niveau verbleibt.

Die Anzahl evaluierbarer Hashwerte hat sich vom Beginn bis zum Ende der Simulation in allen drei Szenarien mehr als verdoppelt. Damit ist gezeigt, dass durch die Selbstorganisation der Agenten zu Organisationsnetzen der Anteil von Hashwerten signifikant erhöht wird, der auch für die Agenten von Nutzen ist. Ein benevolenter Agent kann (wie schon gezeigt) einen Hashwert nur dann verwerten, wenn er diesen von einem benevolenten Agenten mit gleicher Gruppenzugehörigkeit empfangen hat. Und das wird durch die Struktur der Organisationsnetze garantiert.

6.3 Small World Networks

Wir stellen an die Agenten der Experimentalumgebung die wesentliche Forderung, dass sie Hashwerte in kurzer Rundenzeit mit möglichst geringen Kommunikationskosten innerhalb des Netzes aus holonischen Organisationen verbreiten. Diese Forderung nach effizientem Datenversand lässt sich durch möglichst kurze Pfade zwischen beliebig gewählten Agenten X und Y erreichen. *Small world* Netzwerke [STROGATZ und WATTS 1998] erfüllen die Ansprüche nach geringen durchschnittlichen Pfadlängen zwischen den Knoten mit einer möglichst geringen Anzahl von Verbindungskanten. Wir stellen das Phänomen der *small world* Netzwerke im ersten Abschnitt dieses Kapitels vor und gehen näher auf ihre Eigenschaften ein. Im zweiten Kapitel werden wir zeigen, dass bei optimaler Ausbildung des Netzes aus Organisationen mit passend gewählter maximaler Organisationsgröße O_{max} unsere Experimentalumgebung die Eigenschaften eines *small world* Netzwerkes erfüllt und dadurch ein effizienter Datenaustausch gewährleistet ist.

6.3.1 Der Aufbau von Small World Networks

Netzwerke von dynamischen Systemen sind weder streng regulär angeordnet, noch bilden sie eine komplett zufällige Struktur. Ihre Struktur liegt vielmehr zwischen diesen beiden Extremen. Diese Eigenschaft gilt beispielsweise für soziale Strukturen. So fand MILGRAM [1967] heraus, dass zwei zufällig aus der Weltbevölkerung gewählte Personen über kurze Ketten von Bekanntschaften verbunden sind. Dieses soziale Netzwerk zeigt erstaunliche kurze Pfadlängen und eine starke lokale Verdichtung der Knoten (Knoten besitzen eine Vielzahl direkter Nachbarn). Der Verdichtungskoeffizient C *(engl. clustering coefficient)* innerhalb dieser Netzwerke ist folgendermaßen definiert: wenn ein Knoten k Verbindungen zu Anz_k Nachbarknoten besitzt, so können zwischen diesen Nachbarknoten maximal $\frac{Anz_k(Anz_k-1)}{2}$ ungerichtete Kanten existieren. Sei C_k der Anteil dieser Kanten, die tatsächlich ausgebildet werden. Dann wird C als Mittelwert über die C_k aller Knoten k errechnet. C gibt den Anteil der Nachbarknoten von k an, die untereinander selbst wieder Nachbarn sind. Diese Eigenschaft wird auch als Verdichtung *(engl. clustering)* bezeichnet. Der Verdichtungskoeffizient eines vollständig verbundenen Graphen ist $C = 1$, der eines Zufallsgraphen ist $C_{random} = \frac{Anz_k}{N}$, wobei N die Gesamtzahl der Knoten und Anz_k die durchschnittlich von jedem Knoten ausgehende Anzahl von Kanten darstellt. C_{random} wird für Netzwerke mit großer Knotenzahl beliebig klein. Die durchschnittliche Pfadlänge l_{random} eines Zufallsgraphen gleicher Knotenzahl N approximieren wir mit $\frac{logN}{logAnz_k}$ [STROGATZ und WATTS 1998].

Small world Netzwerke wurden intensiv von WATTS UND STROGATZ [1998] untersucht. Dabei definieren sie folgende Eigenschaften von *small world* Graphen:

1. Der Verdichtungskoeffizient C ist viel größer als der eines zufälligen Graphen mit gleicher Anzahl von Kanten und durchschnittlicher Anzahl von Kanten pro Knoten.

2. Die durchschnittliche Pfadlänge $l_{smallworld}$ ist fast so gering wie die durchschnittliche Pfadlänge l_{random} des korrespondierenden Zufallsgraphen.

Small world Graphen erreichen eine mit der von Zufallsgraphen vergleichbare Pfadlänge durch die Ausbildung einer geringen Zahl von Kanten längerer Reichweite (*engl. short cuts*) zwischen verschiedenen Knoten. Diese Kanten mit längerer Reichweite stellen Verbindungen zwischen beliebigen Knoten des Netzwerkes dar. Seien X und Y zwei Knoten, die eine *short cut* Kante bilden. Durch die Ausbildung dieser Kante verkürzt sich nicht nur die Pfadlänge zwischen X und Y, sondern auch die Pfade zwischen den Nachbarschaftsknoten von X und den Nachbarschaftsknoten von Y werden dramatisch verkürzt. Bringt man in einen regulären Graphen ein geringe Anzahl der Kanten längerer Reichweite ein, so verkürzt sich auch dort die durchschnittliche Pfadlänge dramatisch. Nach *Comellas* et al. [2000] sind *small world* Netzwerke „vielversprechende Kandidaten für Kommunikationsnetzwerke, da typische Datenflussmuster in Kommunikationsnetzwerken hohe Verdichtung mit einer geringen

Anzahl von Verbindungen längerer Reichweite zeigen“. So zeigt auch das Internet in seinem Aufbau die charakteristischen Eigenschaften eines *small world* Netzwerkes [ADAMIC 1999].

6.3.2 Small World Eigenschaften des Spamfilternetzwerkes

Das Netzwerk aus Organisationen, welches die Antispam-Agenten im Laufe der Simulation bilden, zeigt die Eigenschaften eines *small world* Netzwerkes. Zum einen sind die Knoten innerhalb jeder holonischen Organisation stark verdichtet, da jeder Agent mit jedem Mitglied seiner Organisation direkt über eine Kante in einer *Clique* verbunden ist. Zum anderen entsprechen die Verbindungskanten zwischen den Organisationen den im vorherigen Abschnitt beschriebenen *short cuts*. Die *small world* Eigenschaften unseres Netzwerkes lassen sich auch experimentell in einem Szenario mit 60 Agenten aus derselben Gruppe nachweisen. Wir variieren in den verschiedenen Versuchsläufen die maximale Organisationsgröße O_{max}. Die durchschnittlichen Pfadlänge l wird nach Beendigung der Simulation mit Hilfe eines All-Pairs-Shortest-Path-Algorithmus errechnet, der in einer Adjazenzmatrix die Pfadlängen aller Knotenkombinationen ausgibt. Die durchschnittliche Pfadlänge $l_{Netzwerk}$ des Netzes aus holonischen Organisationen errechnet sich als Mittelwert der in der Matrix aufgeführten Pfadlängen.

O_{max}	$l_{Netzwerk}$	l_{random}	$C_{Netzwerk}$	C_{random}	*erreichbare Knoten über Pfade der Länge* $\in \{1, \ldots, 3\}$
12	2.52	1.78	0.90	0.14	99.2 %
10	2.57	1.93	0.82	0.17	96.2 %
8	2.64	2.07	0.75	0.12	91.5 %
6	3.28	2.46	0.63	0.09	53.6 %

Tabelle 6.6: Pfadlängen und Verdichtungskoeffizienten des Spamfilternetzwerkes im Vergleich zum korrespondierenden Zufallsgraphen

In Tabelle 6.6 werden die charakteristische Pfadlänge $l_{Netzwerk}$ und der Verdichtungskoeffizient $C_{Netzwerk}$ von vier Simulationsdurchläufen mit unterschiedlicher maximaler Organisationsgröße mit denen eines Zufallsgraphen gleicher Knotenzahl N und durchschnittlicher Anzahl von Kanten pro Knoten verglichen. In allen Simulationen zeigen die gebildeten Netze von holonischen Organisationen den *small world* Effekt:

$$l_{Netzwerk} \geq l_{random} \quad \text{und} \quad C_{Netzwerk} \gg C_{random} \tag{6.2}$$

Die durchschnittlichen Pfadlängen der Netzwerke aus Antispam-Agenten stimmen in etwa mit den Längen des korrespondierenden Zufallsgraphen überein. Der Verdich-

tungskoeffizient $C_{Netzwerk}$ übersteigt jedoch den Wert von C_{random} um ein Vielfaches. Dies sind gerade die beiden Bedingungen eines *small world* Netzwerkes. Somit haben wir experimentell die *small world* Eigenschaft unseres Spamfilternetzwerkes nachgewiesen.

In der letzten Spalte von Tabelle 6.6 wird die prozentuale Knotenzahl angegeben, die durchschnittlich von einem beliebigen Knoten des Netzwerkes über Pfade mit maximaler Länge $l_{path} = 3$ erreicht wird. Je höher diese Knotenzahl ist, desto effizienter werden die Hashwerte im Netzwerk verbreitet. Die Anzahl Anz_{max} der Knoten, die maximal von einem beliebigen Agenten des Netzwerkes über Pfade mit maximaler Länge $l_{path} = 3$ erreicht wird, beträgt nach der Herleitung in Kap. 5.2.2 maximal $Anz_{max} = O_{max}^2 + (O_{max} - 1)$. Je kleiner also die maximale Organisationsgröße gewählt wird, desto weniger Knoten können bei fester Anzahl von Agenten über diese Pfade erreicht werden. Wählt man $O_{max} < \sqrt{N}$ mit $N = Anzahl\ der\ Knoten$, so sinkt nach der obigen Formel die Zahl von Knoten erheblich, die im Durchschnitt von einem beliebigen Punkt des Graphen über Pfade mit maximaler Länge $l_{path} = 3$ erreicht werden können. Dies lässt sich auch anhand der Versuchsergebnisse bestätigen. So können die 60 Agenten innerhalb des Szenarios mit $O_{max} = 6$ im Schnitt jeweils nur noch 53.6 Prozent der Agenten des Organisationsnetzes über Kanten maximaler Länge drei ansprechen, während sie in den anderen drei Szenarien mit $O_{max} \geq 8$ über 90 Prozent der Knoten erreichen. Dies lässt sich folgendermaßen erklären: Mit kleiner werdendem O_{max} muss bei gleich bleibender Agentenzahl eine größere Zahl kleiner Organisationen gebildet werden. Jeder Knoten darf maximal nur eine Kante zu einer benachbarten Organisation ausbilden. Ein Knoten kann alle Knoten der Organisationen, die mit seiner eigenen Organisation über Verbindungskanten direkt verbunden sind, über Pfade maximaler Länge drei erreichen. Die Anzahl von benachbarten Organisationen und die Zahl darin enthaltener Agentenknoten ist bei kleinem O_{max} geringer, als in Szenarien mit größerem O_{max}. Die durchschnittliche Pfadlänge $l_{Netzwerk}$ wächst also bei Verkleinerung von O_{max}, die Anzahl erreichbarer Knoten sinkt.

6.4 Analyse der Hypothesen und der Small World Eigenschaft

Viele der von uns getesteten Hypothesen wurden anhand der durchgeführten Simulationen bestätigt, jedoch gab es auch unerwartete und überraschende Ergebnisse. Die erwarteten Ergebnisse sind:

- Die Agenten können mehr als die Hälfte der Spammails über Hashwerte identifizieren, trotz der kurzen Intervalle, in der die entsprechenden Hashwerte über das Netzwerk gesendet werden müssen. Die Identifizierung von mehr als der Hälfte der Emails allein durch den Hashwert-Vergleich zeigt die Effizienz des Spamfilternetzwerkes. Der Klassifizierungsfehler des gesamten Spamfilternetzwerkes

ist geringer als der Klassifizierungsfehler des SVM-Algorithmus der Antispam-Agenten. Wir haben in den Experimenten gezeigt, dass der Klassifizierungsfehler der Antispam-Agenten durch den Austausch der Hashwerte innerhalb des Multiagentensystems verringert wird.

- Benevolente Agenten, die die gleichen Spammails erhalten, schließen sich zu Netzen aus holonischen Organisationen zusammen. Durch die Bildung der Organisationen und ihrer Verbindungen zueinander wird der Versand der Hashwerte kanalisiert, d.h. Hashwerte werden nicht wahllos zu Agenten verschickt, die in einer anderen Gruppe sind und damit diese Werte niemals benötigen. Stattdessen tauschen Agenten der einzelnen Gruppen verstärkt Hashwerte untereinander aus, die sie auch evaluieren und zur Erkennung von Spammails einsetzen können.

- Die hohe Zahl erkannter Spammails zeigt den effizienten und schnellen Versand der Hashwerte über das Netzwerk. Aufgrund der Struktur der Organisationsnetze können Hashwerte von jedem Agenten des Netzwerkes über kurze Pfade zu jedem beliebigen anderen Agenten des Netzes gesendet werden.

- Im Kapitel 6.3.2 wurden die *small world* Eigenschaften der Netzwerke von Antispam-Agenten experimentell nachgewiesen. Die wesentlichen Merkmale dieser Netzwerke sind in *Cliquen* organisierte verdichtete Knoten, zwischen denen eine geringe Anzahl von Verbindungskanten längerer Reichweite *(engl. short cuts)* existieren. *Small world* Netzwerke sind vielversprechende Kandidaten für Kommunikationsnetzwerke [COMELLAS et al. 2000]. Die *small world* Struktur unseres Netzwerkes aus Antispam-Agenten ermöglicht einen effizienten und schnellen Austausch von Hashwerten über kurze Pfade, da *small world* Netzwerke „schnelle Reaktionszeiten ermöglichen“ [LAGO-FERNÁNDEZ et al. 2000].

- Bei geeigneter Wahl der maximalen Organisationsgröße O_{max} verbinden sich alle benevolenten Agenten einer Gruppe mit geringem Klassifizierungsfehler zu einem Netz aus Organisationen. Auch Agenten mit hohem Klassifizierungsfehler finden sich zu eigenen Organisationsnetzen zusammen.

Die unerwarteten Ergebnisse umfassen:

- Mischagenten können den Vorteil, dass sie mehr Spammails erhalten, als Agenten mit Zugehörigkeit zu nur einer Gruppe, nicht ausspielen. Stattdessen gehen sie weniger Verbindungen zu anderen Agenten ein. Weiterhin benötigen die Mischagenten eine größere Zahl von Runden zum Zusammenschluss mit anderen Agenten, als die Agenten mit Zugehörigkeit zu nur einer Gruppe.

- Mischagenten werden nicht wahllos in alle Agentengruppen integriert, sondern schließen sich mit Mischagenten gleicher Gruppenzugehörigkeit zusammen. Dazu sind sie gezwungen, weil Agenten mit Zugehörigkeit zu einer Gruppe sich untereinander verstärkt verbinden.

Kapitel 7

Ergebnisse und Schlussfolgerungen

In diesem Kapitel fassen wir die wichtigsten Schlussfolgerungen aus der Diplomarbeit zusammen. Wir haben eine Spamfiltersoftware entwickelt, die Spam durch Kombination mehrerer Filtermechanismen sicher und schnell erkennt. Die Eigenschaften unseres Spamfilternetzwerkes sind im folgenden Abschnitt zusammengefasst. Abschließend zeigen wir im zweiten Abschnitt, wie die Kommunikation im Netzwerk aus Antispam-Agenten durch Einsatz von Vertrauen und Selbstorganisation verbessert wird. Außerdem werden wir im dritten Abschnitt Ansatzpunkte aufzeigen, an die im Rahmen zukünftiger Arbeiten angeknüpft werden kann und aus denen weitere Ergebnisse gewonnen werden können.

7.1 Spamfilterung durch Kombination von Multiagentensystemen und Support Vector Machines

Um die große Zahl von Spammails zu bekämpfen, die Tag für Tag auf den Emailaccounts eingehen, bedarf es eines mächtigen Emailfilters mit hoher Zuverlässigkeit. Die Kombination von Multiagentensystemen mit Algorithmen zur Textklassifizierung erhöht die Erkennungsrate der Filtersoftware, in unserem Fall der Antispam-Agenten. Informationen über Spam breiten sich über das Netzwerk zu allen anderen Agenten aus. Wir haben den Antispam-Agenten der Benutzer von Emailaccounts die Möglichkeit gegeben, Spaminformationen über die Multiagentenplattform FIPA-OS miteinander auszutauschen und so ihr Wissen über Spam zu vergrößern. Wird eine Spammail durch den Klassifizierungsalgorithmus *support vector machines* als Spam identifiziert, so generiert der Agent mit dem von uns entwickelten Verfahren aus den Buchstabenhäufigkeiten der Email einen Hashwert. Der Wert wird über das P2P-Netzwerk an die über dem Internet verteilten Agenten des Multiagentensystems verschickt. Diese Agenten sind nun in der Lage, die zu dem Hashwert korrespondierende Spammail sicher zu erkennen, ohne erneut den Inhalt der Email analysieren zu müssen. Ebenfalls werden durch Vergleich der Hashwerte Spammails mit ähnlichem Inhalt identifiziert. Agenten, die dem Netzwerk beitreten, empfangen Hashwerte von anderen Agenten des Spam-

filternetzwerkes und bauen schnell eine Datenbank mit aktuellen Spaminformationen auf. Der zweite Filter, *support vector machines*, unterstützt die Antispam-Agenten bei der Aufgabe der Klassifizierung unbekannter Emails zur Erzeugung neuer Hashwerte. Jeder Benutzer kann die Erzeugung seiner individuellen Trainingsmatrix beeinflussen, indem er die Klassifizierungsergebnisse von SVM abändert. Auf diese Weise kann er beeinflussen, welche Arten von Emails durch SVM als Spam klassifiziert werden. Der Funktionsumfang des Antispam-Agenten wird durch einen dritten Filtermechanismus erweitert, der eine *whitelist* zur Klassifizierung einsetzt. Die empfangenen Emails werden mit vom Benutzer spezifizierten Emailadressen vertrauenswürdiger Absender verglichen und entsprechende Emails als Nicht-Spam klassifiziert.

7.2 Reduzierung von Kommunikationskosten durch Selbstorganisation

Der Versand der Hashwerte an alle Agenten des Spamfilternetzwerkes verursacht hohe Kommunikationskosten. Diese Kosten können enorm gesenkt werden, wenn nur die Agenten untereinander Spaminformationen austauschen, die die gleichen Spammails erhalten, also in derselben Gruppe sind. Daher haben wir ein Verfahren entwickelt, welches diesen Agenten ermöglicht, sich zusammenzuschließen ohne zu wissen, welcher Gruppe sie angehören. Zur Lösung dieses Problems haben wir Methoden aus dem Bereich der Sozionik angewendet. Die Agenten vergleichen empfangene Hashwerte mit den Einträgen ihrer Datenbank und erhöhen bei Übereinstimmung ihr Vertrauen zum Absender des Hashwertes. Somit bauen nur diejenigen Antispam-Agenten zueinander Vertrauen auf, die dieselben Spammails erhalten und daraus identische Hashwerte erzeugen. Auf der Basis von Vertrauen schließen sich benevolente Agenten derselben Gruppe durch Selbstorganisation zu Netzen aus holonischen Organisationen zusammen und tauschen dort verstärkt Informationen. Der Austausch der Hashwerte über das P2P-Netzwerk führt zu einer Verbesserung der Klassifizierungsgenauigkeit der einzelnen Antispam-Agenten.

Wir haben weiterhin eine Experimentalumgebung entwickelt, in der wir den Informationsaustausch zwischen den Antispam-Agenten durch Interaktion simuliert haben. Verschiedene Eigenschaften des Spamfilternetzwerkes haben wir anhand von Hypothesen mit Hilfe der Experimentalumgebung überprüft. Wir haben nachgewiesen, dass sich Agenten, die gleiche Spammails erhalten, in Organisationsnetzen organisieren und verstärkt Informationen tauschen. Die Kommunikationskosten des gesamten Netzwerkes werden durch den intelligenten Zusammenschluss der Agenten gesenkt bei steigender Klassifizierungsgenauigkeit der einzelnen Antispam-Agenten. Durch die Bildung von Organisationsnetzen bilden die Agenten eine Netzwerkstruktur, die *small world* Eigenschaften aufweist. Ebenfalls werden böswillige Agenten aus dem Organisationsnetz der benevolenten Agenten ausgeschlossen. Allein der von uns entwickelte Filter durch

Vergleich der Hashwerte hat das Potential, die Hälfte der Spammails mit geringem Klassifizierungsfehler zu identifizieren. Die restlichen Emails werden durch *support vector machines* gefiltert. Die Klassifizierungsgenauigkeit der Antispam-Agenten und damit die Anzahl erkannter Spammails erhöht sich nach deren Einbettung in die Multiagentenplattform. Wir haben somit unser Ziel erreicht, die Genauigkeit der Agenten bei der Analyse von Emails zu erhöhen, da diese neben *support vector machines* auch die über das P2P-Netzwerk versendeten Hashwerte zur Klassifizierung einsetzen.

7.3 Ausblick

Die Kombination von Textklassifizierung und Multiagentensystemen innerhalb des Spamfilternetzwerkes hat sich als sinnvolle Alternative zu herkömmlichen, isolierten Spamfiltern erwiesen. Jedoch sind weitere Maßnahmen zur besseren Akzeptanz beim Anwender denkbar, des weiteren lässt sich der Anwendungsbereich des Emailfilters weiter vergrößern:

- Um eine einfache Installation des Spamfilters auf verschiedenen Betriebssystemen zu ermöglichen, wäre eine Loslösung des Spamfilternetzwerkes von der FIPA-OS Plattform wünschenswert. Stattdessen würden die Antispam-Agenten über ein P2P-Netzwerk vergleichbar dem Gnutellanetzwerk [GNUTELLA 2001] kommunizieren. Die Dienste der Agentenplattform wie Registrierung und Deregistrierung müssten vom P2P-Netzwerk übernommen werden. Auch kann in zukünftige Versionen des Spamfilternetzwerkes ein Stemmer für deutschsprachige Wörter integriert werden, so dass auch Emails mit deutschsprachigem Inhalt analysiert und klassifiziert werden können.

- Bei steigender Zahl von Agenten im Netzwerk steigt die Gefahr, dass die Buchstabenhäufigkeiten zweier verschiedener Emails zufällig übereinstimmen. Die daraus erzeugten Hashwerte würden trotz unterschiedlichem Emailinhalt übereinstimmen. Um dieses Risiko zu minimieren, könnte die Zahl der relevanten Buchstaben, die der Agent zur Erzeugung der Hashwerte heranzieht, erhöht werden. Ebenfalls könnte die Gesamtzahl von Buchstaben im Inhalt der Spammail im korrespondierenden Hashwert gespeichert werden. So würden die Agenten aus zwei Emails mit gleicher Buchstabenhäufigkeit, aber unterschiedlicher Länge zwei verschiedene Hashwerte generieren.

- Der Filter durch Vergleich der Hashwerte kann in der momentanen Konfiguration nur Spam filtern, da Hashwerte ausschließlich aus dem Inhalt von Spamnachrichten erzeugt werden. Jedoch wäre es auch denkbar, dass nach dem gleichen Verfahren auch Hashwerte für Nicht-Spammails erzeugt und über das Netzwerk verschickt würden. Diese Hashwerte könnten ebenso zum Hashwert-Vergleich eingesetzt werden. Diese Erweiterung macht jedoch nur für Agenten Sinn, die die gleichen Nicht-Spammails erhalten (z.B. Mitglieder derselben Mailinglisten).

Im Bezug auf die Selbstorganisation der Agenten wären folgende Aspekte zu beleuchten:

- Wir sind bei der Spezifikation von Protokollen für den Zusammenschluss von Agenten in der Experimentalumgebung davon ausgegangen, dass Nachrichten ohne Kommunikationsfehler über das Netzwerk versendet werden. Daher haben wir auf eine Fehlerbehandlung verzichtet, damit der Zusammenschluss von Agenten in möglichst geringer Rundenzahl abgeschlossen ist. In P2P-Netzwerken muss jedoch mit der Duplizierung, dem Verschwinden und der Fehlleitung von Nachrichten gerechnet werden. Die Protokolle für den Zusammenschluss der Agenten könnten in diesem Hinblick um eine Fehlerkorrektur ergänzt werden.

- Im Spamfilternetzwerk schließen sich diejenigen Agenten zusammen, die die gleichen Spammails erhalten. Den Benutzern dieser Agenten könnte ermöglicht werden, über das P2P-Netzwerk Kontakt zueinander aufzunehmen. Gemeinsam könnten sie herausfinden, woher Spammer die Emailadressen ihrer Gruppe gesammelt haben. Sie könnten die Einträge der Emailadressen im entsprechenden Forum, Usenet oder der Homepage besser absichern oder auch löschen, um die Aufnahme in weitere Spammerlisten zu vermeiden.

- Böswillige Agenten, die sich zu Anfang der Simulation das Vertrauen anderer Agenten durch den Versand evaluierbarer und zu Spammails korrespondierender Hashwerte erschlichen haben, könnten auf diese Weise in das Netz aus Organisationen aufgenommen werden. Wäre die Aufnahme erreicht, könnten sie mit dem Versand ungültiger Hashwerte beginnen und die Funktionsweise des Netzwerkes beeinträchtigen. Um dies zu verhindern, sollten Agenten die Anzahl evaluierbarer Hashwerte ihrer Organisationsmitglieder ständig überprüfen. Unterschreitet die Anzahl evaluierbarer Hashwerte eines bestimmten Agenten X in einem gewissen Zeitintervall einen Schwellwert, so sollte es den anderen Agenten aus der Organisation von X möglich gemacht werden, den betreffenden böswilligen Agenten X aus der Organisation wieder auszuschließen.

Anhang A

Konfiguration der Experimente

In Tabelle A.1 sind die Parameter der Experimentalumgebung während der Experimente zusammengestellt. Dabei bezeichnen wir mit O_{max} die maximal mögliche Anzahl von Mitgliedern innerhalb der holonischen Organisation, mit S_{HO} und $S_{Netzwerk}$ werden die Schwellwerte für den Zusammenschluss der Agenten und Organisationen spezifiziert. Das Nachrichtenlimit gibt an, wie viele Nachrichten jeder Agent maximal pro Runde versenden darf. Haben wir zur Überprüfung einer Hypothese mehrere Versuchsdurchläufe durchgeführt, so sind die dabei veränderten Parameter hervorgehoben.

Parameter der Agentenplattform zur Evaluation der Hypothesen 1-5 und zum Nachweis der small world network Eigenschaft (SWN):

$Hypoth.$	$Lauf$	O_{max}	$Nachrichtenlimit$	$Runden$	S_{HO}	$S_{Netzwerk}$
1	1-2	10	35	4000	6	10
2	1	10	35	3000	6	10
3	1	4	17	5000	**6**	**10**
	2	4	17	5000	**15**	**20**
	3	4	17	5000	**20**	**25**
	4	4	17	5000	**30**	**35**
4	1	10	35	4000	**5**	**10**
	2	10	35	4000	**10**	**15**
	3	10	35	4000	**20**	**25**
5	1-3	10	35	4000	6	10
SWN	1	**6**	**23**	4000	6	10
	2	**8**	**29**	4000	6	10
	3	**10**	**35**	4000	6	10
	4	**12**	**41**	4000	6	10

Tabelle A.1: Parameter der Experimentalumgebung in den verschiedenen Experimenten

In den folgenden Tabellen finden sich die Parameter der Agenten während der Ausführung der Experimente. In der linken Tabelle haben wir die Parameter der Verteileragenten zusammengestellt, in der rechten Tabelle die eingestellten Parameter der Antispam-Agenten. Für die Verteileragenten legt der Parameter $Spam$ fest, wie hoch der prozentuale Anteil der Spammails an der Gesamtzahl generierter Emails ist. Des weiteren bestimmen wir über den Parameter *Häufigkeit* die Wahrscheinlichkeit, mit der ein Verteiler pro Runde sendet. Eine *Häufigkeit* von 100 Prozent bewirkt, dass der Verteiler jede Runde eine Email generiert, niedrigere Werte verringern die Anzahl versendeter Emails. Neben dem Klassifizierungsfehler f_{class} finden sich für die Antispam-Agenten die Aufschlüsselung der Agenten nach Gruppen. Agenten derselben Gruppe erhalten vom selben Verteiler ihre Emails. Nicht in den Tabellen aufgeführt sind die Unter- und Obergrenze für das Intervall, in dem die Antispam-Agenten ihre Emails von den simulierten Mailservern abholen. Diese beiden Werte sind während allen Simulationen konstant auf Mailabholung$_{min}$ = 10 und Mailabholung$_{max}$ = 15 gesetzt, d.h. die Antispam-Agenten holen ihre Emails alle 10 bis 15 Runden vom Mailserver ab.

Agentenkonfiguration für Hypothese 1:

$Verteiler - Agenten$			
$Agent$	$Spam$	$Gruppen$	Häufigkeit
0	100%	A	100%
1	0%	A	100%

$Antispam - Agenten$			
$Lauf$	$Gruppe$	$Anzahl$	f_{class}
1	A	60	5%
2	A	60	15%

Agentenkonfiguration für Hypothese 2:

$Verteiler - Agenten$			
$Agent$	$Spam$	$Gruppen$	Häufigkeit
0	100%	A	100%
1	0%	A	100%

$Antispam - Agenten$			
$Lauf$	$Gruppe$	$Anzahl$	f_{class}
1	A	60	2%

Agentenkonfiguration für Hypothese 3:

$Verteiler - Agenten$			
$Agent$	$Spam$	$Gruppen$	Häufigkeit
0	100%	A	100%
1	100%	B	100%
2	100%	AB	100%

$Antispam - Agenten$			
$Lauf$	$Gruppe$	$Anzahl$	f_{class}
1-4	A	20	5%
	B	20	5%
	AB	20	5%

Agentenkonfiguration für Hypothese 4:

$Verteiler - Agenten$			
$Agent$	$Spam$	$Gruppen$	Häufigkeit
0	100%	A	100%
1	0%	A	100%

$Antispam - Agenten$			
$Lauf$	$Gruppe$	$Anzahl$	f_{class}
1-3	A	40	5%
1-3	A	20	90%

Agentenkonfiguration für Hypothese 5:

$Verteiler - Agenten$			
$Agent$	$Spam$	$Gruppen$	Häufigkeit
0	100%	A	100%
1	100%	B	100%
2	100%	C	100%

$Antispam - Agenten$			
$Lauf$	$Gruppe$	$Anzahl$	f_{class}
1	A	60	5%
2	A	20	5%
	B	20	5%
	AB	20	5%
3	A	10	5%
	B	10	5%
	AB	10	5%
	AC	10	5%
	BC	10	5%
	ABC	10	5%

Agentenkonfiguration für SWN:

$Verteiler - Agenten$			
$Agent$	$Spam$	$Gruppen$	Häufigkeit
0	100%	A	100%

$Antispam - Agenten$			
$Lauf$	$Gruppe$	$Anzahl$	f_{class}
1-3	A	60	5%

Anhang B

Bedienung des Antispam-Agenten

Wir werden nun auf die wichtigsten Steuerungselemente der grafischen Benutzerschnittstelle des Antispam-Agenten eingehen. Die Dateien mit dem Quellcode des Agenten sind als Zip-Archiv [METZGER 2003] erhältlich. Weitere Hinweise zur Installation des Antispam-Agenten sind in der Datei „Readme.txt“ enthalten, die dem Zip-Archiv beiliegt. Die verschiedenen Auswahlfenster des Antispam-Agenten lassen sich über die Registerleiste (Abb. B.2) oder optional durch Betätigen der Schalter auf der linken Schaltfläche des Agenten auswählen. Folgende Menüpunkte sind verfügbar:

Liste der Emails

Nach Auswahl des Registers „Emailübersicht“ wird dem Benutzer eine Tabelle präsentiert, in der eine Übersicht aller vom Mailserver geladenen Emails mit Absender, Empfänger und Betreff angezeigt wird. Durch Betätigen des Schalters „Mail abholen“ kann der Benutzer seine Emails vom Mailserver abholen. Ist die Option „Email bei Abholung vom Server filtern“ in den weiteren Optionen aktiviert, so werden die Emails beim Herunterladen vom POP3-Server gleichzeitig klassifiziert und in die Kategorien Spam und Nicht-Spam eingeteilt. Neben den Spalten *Absender*, *Empfänger* und *Betreff* finden sich in der Tabelle ebenso das Klassifizierungsergebnis des Antispam-Agenten nach der Analyse des Inhaltes der Email, sowie die Angabe des Filters, der die entsprechende Email klassifiziert hat.

Mailserver

Durch Auswahl des Registers „Mailserver“ gelangt der Benutzer zu einer Eingabemaske, in der er die Zugangsdaten seines POP3-Mailservers eingeben kann. Dazu gehören der Server- und Benutzername, sowie das zur Authentifikation benötigte Passwort. Die Einträge werden durch Betätigen des Schalters „speichern“ gesichert.

Meldungen

Im Auswahlfenster „Meldungen “ werden die Statusmeldungen des Antispam-Agenten in einer Liste ausgegeben. So kann der Benutzer den Ablauf der Initialisierung und

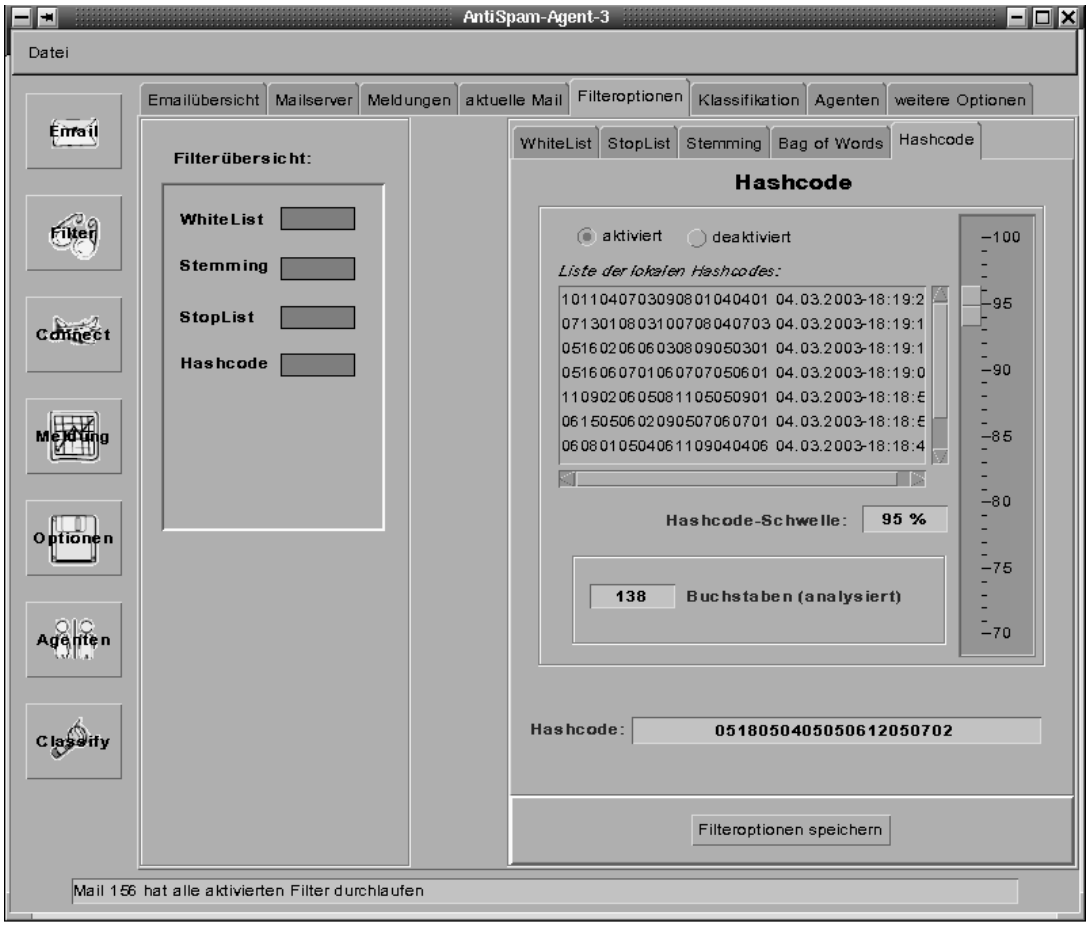

Abbildung B.1: Benutzerschnittstelle mit Tabelle der Hashwerte sowie dem Hashwert der ausgewählten Email

Anmeldung des Agenten an der FIPA-OS Plattform sowie die Schritte bei der Analyse der Emails mitverfolgen. Ferner werden in diesem Fenster auch Fehlermeldungen ausgegeben, die der Agent generiert.

Aktuelle Mail

Nach Auswahl des Registers „Aktuelle Mail“ kann sich der Benutzer den Inhalt der Email betrachten, die er durch Anklicken in der Übersicht aller Emails ausgewählt hat. Neben dem Inhalt der Email finden sich in diesem Fenster auch Angaben wie Sender, Empfänger und Betreff der ausgewählten Email. In einem zweiten Textfenster kann der Benutzer den Inhalt der Email ohne HTML-Tags betrachten.

Filteroptionen

Über die Auswahl des Registers „Fiteroptionen“ kann der Benutzer nähere Details der einzelnen Filter einsehen. So kann er in einer Eingabemaske Emailadressen eingeben

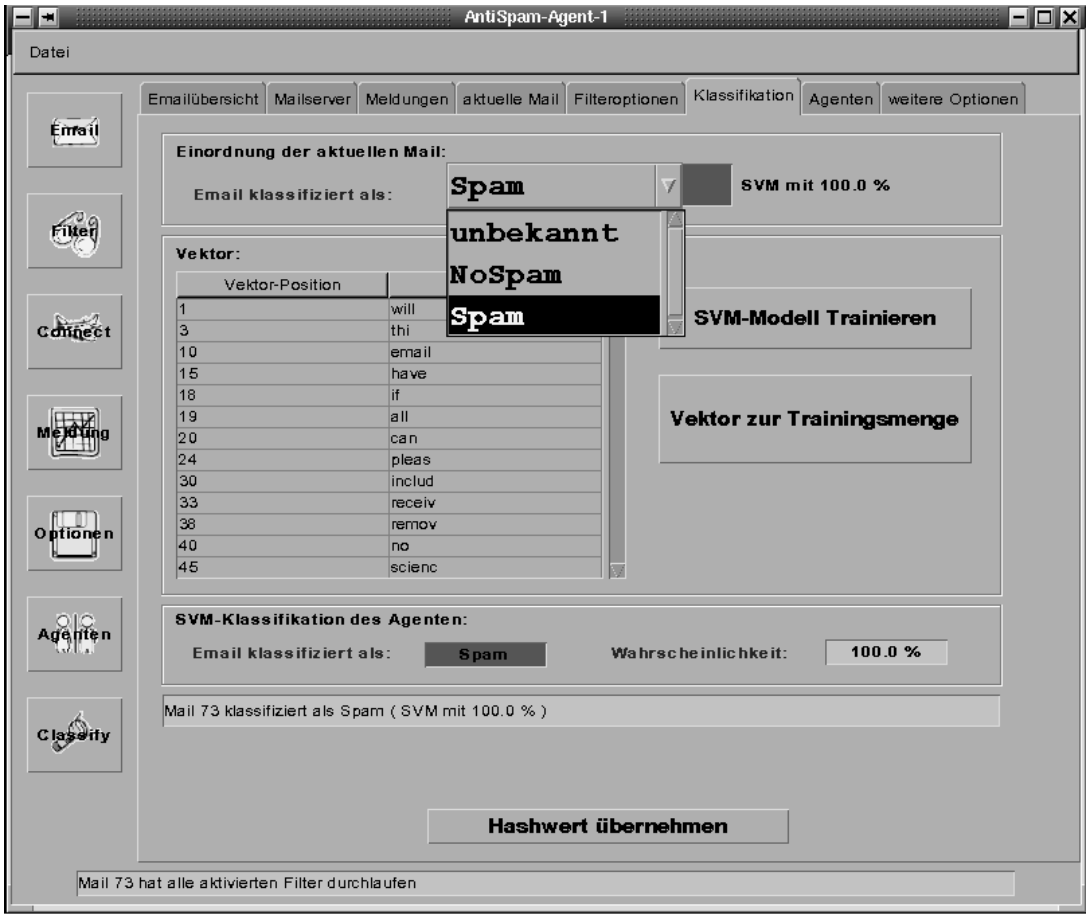

Abbildung B.2: Benutzerschnittstelle mit Tabelle der Vektorattribute sowie dem Klassifizierungsergebnis von SVM

und speichern, die zur Filterung der Emails mit Hilfe der *whitelist* eingesetzt werden. Weiterhin kann die aktuelle Liste der Hashwerte angezeigt werden, sowie der Hashwert der momentan ausgewählten Email (Abb. B.1). Über einen Schieberegler lässt sich der Grad der Ähnlichkeit festlegen, bei dessen Überschreiten der Filter durch Hashwert-Vergleich zwei Hashwerte als identisch betrachtet. Hat der Benutzer eine Email in der Übersicht der Emails ausgewählt, so werden die Wörter des *bag of words* in einer Liste dargestellt.

Klassifizierung

Die Klassifizierungsentscheidungen des Antispam-Agenten können durch Auswahl des Registers „Klassifikation“ (Abb. B.2) verändert werden. In einer Statuszeile wird das Klassifizierungsergebnis der Analyse der aktuell gewählten Email durch *support vector machines* angezeigt. Weiterhin werden alle Attribute des Attributvektors der Emails in einer Liste wiedergegeben, die im Inhalt der aktuellen Email vorkommen.

Dies sind genau die gestemmten Wörter des Inhaltes der Email, die einen korrespondierenden Eintrag in der Vokabularliste besitzen. Der Benutzer kann das Klassifizierungsergebnis mit Hilfe der Checkbox abändern (die geöffnete Checkbox mit Möglichkeit zur Abänderung des Klassifizierungsergebnisses für die aktuell gewählte Email ist in Abb. B.2 zu erkennen). Durch Betätigen des Schalters „Vektor zur Trainingsmenge“ kann der Benutzer den Vektor der aktuellen Email mit dem (eventuell von ihm modifizierten) Klassifizierungsergebnis als Trainingsvektor zur Menge der Trainingsdaten des Antispam-Agenten hinzufügen. Mit Hilfe dieser Trainingsdaten in Form von abgespeicherten, klassifizierten Attributvektoren kann durch Betätigen des Schalters „SVM-Modell Trainineren“ eine neue Klassifizierungsmatrix erzeugt und gespeichert werden. Der *support vector machines* Algorithmus lädt diese Matrix beim erneuten Start des Antispam-Agenten. Emails, die der Agent vom Mailserver lädt, werden durch die gewichteten *support vectors* klassifiziert, die in der Klassifizierungsmatrix enthalten sind. Als weitere Option besteht für den Benutzer die Möglichkeit, den Hashwert der aktuellen Email in die lokale Datenbank des Antispam-Agenten zu übernehmen (sofern die aktuelle Email als Spam klassifiziert wurde). Wurde der Hashwert einer Spammail durch Betätigen des Schalters „Hashwert übernehmen“ in die Datenbank des Agenten übernommen, so wird sie vom Antispam-Agent zusammen mit den anderen lokal gespeicherten Hashwerten an die anderen aktiven Agenten der FIPA-OS-Plattform in regelmäßigen Intervallen gesendet.

Agenten

Durch Auswahl dieses Registers gelangt man zur aktuellen Übersicht der Agenten des Netzwerkes, die mit dem Antispam-Agenten über die FIPA-OS-Plattform verbundenen sind. Diese werden anhand ihrer Identifizierungsnummer angezeigt, welche von der Agentenplattform vergeben wurde.

Weitere Optionen

Über das Register „weitere Optionen“ kann der Benutzer weitergehende Optionen des Antispam-Agenten einstellen und speichern. So kann spezifiziert werden, ob beim Start des Antispam-Agenten direkt eine Verbindung zum Mailserver des Benutzers hergestellt werden soll und die Emails heruntergeladen werden. Weiterhin lässt sich bestimmen, ob Emails beim Herunterladen vom Server automatisch gefiltert werden. Der Benutzer kann ebenfalls die Anzahl von Emails spezifizieren, die der Antispam-Agent maximal vom Mailserver lädt.

Anhang C

Bedienung der Experimentalumgebung

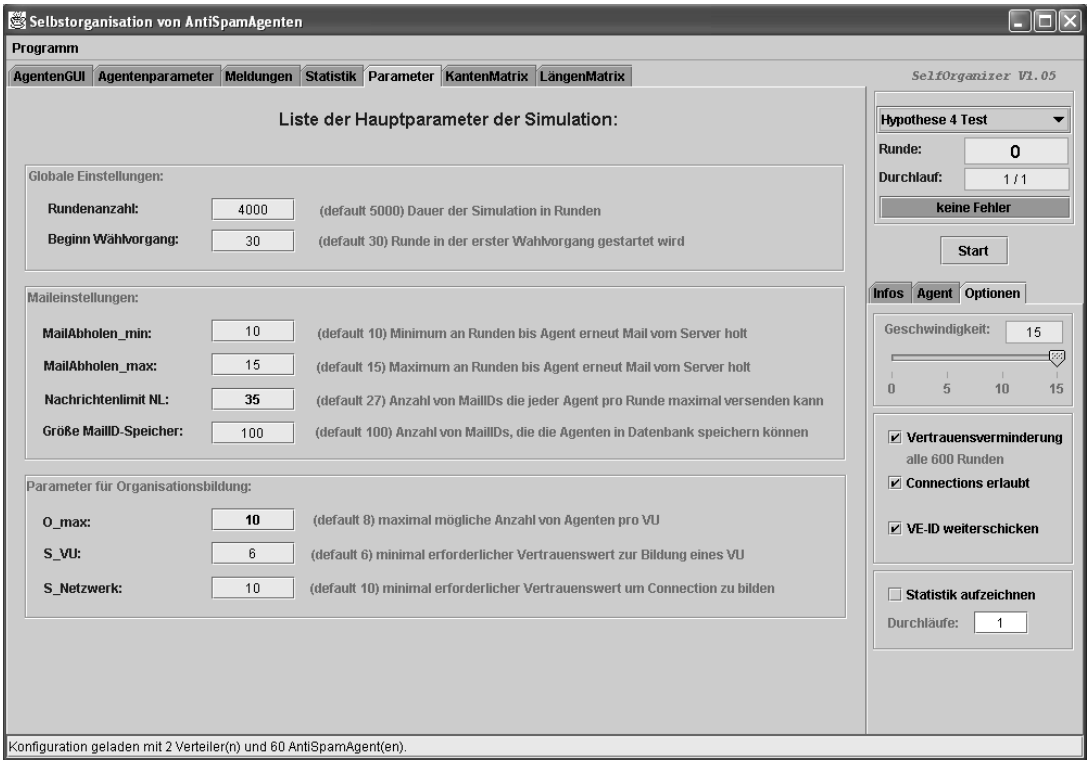

Abbildung C.1: Einstellung der Parameter der Experimentalumgebung über die grafische Benutzerschnittstelle

Wir werden in diesem Abschnitt die wichtigsten Funktionalitäten der grafischen Benutzerschnittstelle der Experimentalumgebung vorstellen. Ein Zip-Archiv mit dem Quellcode der Experimentalumgebung, mit der wir die Eigenschaften des Spamfilternetzwerkes evaluiert haben, kann unter [METZGER 2003] heruntergeladen werden.

Darin befinden sich die Java-Dateien zur Ausführung des Programmcodes sowie die Konfigurationsdateien. In den Konfigurationsdateien sind die Parameter der Verteiler- und Antispam-Agenten für die verschiedenen Simulationsdurchläufe gespeichert. Die Dateien lassen sich mit jedem beliebigen Texteditor verändern. Im Umfang des Zip-Archivs ist die Datei „Readme.txt“ enthalten, in der sich Informationen über Installation und Ausführung der Experimentalumgebung befinden. Nach Start der Experimentalumgebung öffnet sich die grafische Benutzerschnittstelle. Die gewünschte Konfigurationsdatei kann durch Auswahl in der Combobox geladen werden (siehe Abb. 5.1 rechts oben). Über die Registerkarte „Parameter“ gelangt man zu den Einstellungen der allgemeinen Parameter der Experimentalumgebung. Die Parameter sind mit Standardwerten belegt, können über die grafische Schnittstelle jedoch individuell abgeändert werden. Folgende Werte können über die Benutzerschnittstelle verändert werden (Abb. C.1): Gesamtrundenzahl, Beginn des Wahlverfahrens zum Netz aus Organisationen, Mailabholung$_{min}$ und Mailabholung$_{max}$, Nachrichtenlimit NL, Größe des Hashwert-Speichers, maximale Organisationsgröße O$_{max}$ sowie S$_{HO}$ und S$_{Netzwerk}$. Die wichtigsten Parameter wurden bereits in Kapitel 6.1 erläutert. Weiterhin lässt sich im Register „Optionen“ auf der rechten Seite der Benutzerschnittstelle die Geschwindigkeit der Simulation steuern. Weitere Screenshots der Benutzerschnittstelle der Experimentalumgebung wurden bereits in Abb. 5.1 und Abb. 5.4 gezeigt und erläutert. Der Zusammenschluss der Agenten während der Simulation kann unter Register „AgentenGUI“ verfolgt werden. Unter Register „LängenMatrix“ kann auf die Adjazenzmatrix der Pfadlängen aller Knotenkombinationen zugegriffen werden, die mit Hilfe des *All-Pairs-Shortest-Path*-Algorithmus zu Ende der Simulation berechnet werden. Meldungen über den Verlauf der Simulation und der Bildung von holonischen Organisationen finden sich im gleichnamigen Register „Meldungen“.

Anhang D

Vokabularliste

Der Attributvektor, der zur Verarbeitung des Inhaltes einer Email durch SVM generiert wird, spiegelt das Vorkommen von 500 gestemmten Wörtern eines ausgewählten Vokabulars im Inhalt der Email wieder. Dieses Vokabular haben wir nach dem Verfahren aus Kap. 4.5.5 erzeugt und in der Vokabularliste gespeichert. Die Emails werden durch den SVM-Filter nur anhand der Vorkommen der Wörter der Vokabularliste klassifiziert. Die gestemmten Wörter der Vokabularliste sind nach ihrer Häufigkeit des Vorkommens in in den von uns gesammelten Emails eines Mailverteilers des DFKI sortiert. Die folgenden Aufstellung listet die 500 gestemmten Wörter der Vokabularliste in alphabetischer Reihenfolge auf:

abstract	allow	avail	can	commun	currenc
accept	also	back	canada	compani	current
access	american	base	card	complet	dai
achiev	analysi	becaus	case	complex	data
act	ani	been	cash	compon	databas
activ	apolog	befor	cd	comput	date
addit	appli	below	center	confer	de
address	applic	best	chair	consid	deadlin
advanc	approach	between	challeng	contact	dear
affili	april	bologna	chang	contain	debt
after	architectur	both	check	contribut	decis
ag	area	bring	cia	control	demonstr
agent	artifici	bui	click	cooper	depart
agentbas	aspect	build	cochair	coordin	descript
agentori	associ	busi	cognit	copi	design
ai	australia	but	collabor	cost	detail
aim	author	call	color	could	develop
algorithm	autom	cam	commerc	countri	dfki
all	autonom	camerareadi	committe	credit	di

diet
differ
digit
discoveri
discuss
distribut
do
dont
down
draft
due
dvd
dynam
each
econom
effect
eg
either
electron
email
encourag
engin
enter
environ
etc
euro
european
evalu
even
event
everi
exchang
exercis
exist
expect
experi
extend
fat
fax
few
field
file
final
find
first
follow
forev
form
formal
format
found
framework
franc
free
full
further
futur
gener
germani
get
girl
give
go
goal
green
group
grow
guarante
ha
have
held
help
here
home
hotel
how
human
if
imag
implement
import
improv
includ
increas
industri
inform
institut
instruct
integr
intellig
intend
interact
interest
intern
internet
interoper
into
invest
investor
invit
involv
issu
japan
journal
juli
june
just
know
knowledg
la
languag
last
learn
least
level
life
like
limit
link
list
live
loan
logic
look
lose
loss
lotteri
low
made
madrid
mai
mail
main
major
make
manag
mani
march
market
me
mean
mechan
medicin
messag
method
methodolog
michael
million
mine
mobil
model
mondai
monei
month
more
mortgag
most
much
multiag
multimedia
multipl
must
name
nation
natur
need
netherland
network
never
new
no
none
north
note
notif
now
number
object
obtain
off
offer
onli
onlin
ontolog
open
opportun
order
organ
organis
origin
other
out
over
own
page
paper
parallel
part
particip
payment
pdf
peni
peopl
per
perform
person
phone
place
plan
pleas
posit
possibl
postscript
potenti
power
practic
present
price
privaci
problem
proceed
process
product
prof
profil
profit
program
promot
propos
protocol
provid
public
publish
purchas
qualiti
question
quot
rate
read
reason
receiv
recent
recommend
reconfigur
refin
regard
regist
registr
relat
relev
remov
report

request
requir
research
result
review
right
risk
robot
save
scienc
scientif
scope
section
secur
see
select
sell
semant
send
sent
septemb
seri
servic
session
set
sever
sex
share
should
show
signific
simul
site
size
so
social
societi
softwar
some
spain
special
specif
standard
start
state
statement
still
structur
studi
subject
submiss
submit
success
such
support
switzerland
symposium
system
take
talk
technic
techniqu
technolog
teen
term
texa
textdecor
than
thank
thei
their
them
theoret
theori
there
these
think
third
those
through
time
todai
togeth
tool
topic
track
trade
trust
tutori
two
uk
under
unit
univ
univers
universidad
unsubscrib
up
url
usa
user
vega
veri
version
visit
wa
wai
want
we
web
websit
week
weight
welcom
well
were
what
when
where
which
while
who
will
wireless
wish
within
without
word
work
workshop
world
would
write
year

Literaturverzeichnis

[AAS 1997] AAS, K. (1997). *A Survey on Personalised Information Filtering Systems for the World Wide Web*. Technischer Bericht 922, Norwegian Computing Center.

[ADAMIC 1999] ADAMIC, L.A. (1999). *The Small World Web*. In: ABITEBOUL, S. und A.-M. VERCOUSTRE, Hrsg.: *Proceedings of the Third European Conference on Research and Advanced Technology for Digital Libraries*, Nr. 1696, S. 443–452. Springer-Verlag.

[Adaptive READ 2003] ADAPTIVE READ (2003). *Adaptive READ Homepage*. http://www.adaptive-read.de/start.html.

[AGNE et al. 2002] AGNE, S., A. HUST, S. KLINK, M. JUNKER, A. DENGEL, C. ALTENHOFEN, J. FRANKE, I. RENZ und B. KLEIN (2002). *Text-Mining in Adaptive Read*. Künstliche Intelligenz (2/02), S. 30–33.

[ANASTASSAKIS et al. 2001] ANASTASSAKIS, G., T. RITCHINGS und T. PANAYIOTOPOULOS (2001). *Multi-agent Systems as Intelligent Virtual Environments*. KI 2001, LNAI 2174, S. 381–395.

[ANDROUTSOPOULOS et al. 2000] ANDROUTSOPOULOS, I., G. PALIOURAS, V. KARKALETSIS, G. SAKKIS, C. SPYROPOULOS und P. STAMATOPOULOS (2000). *Learning to filter Spam E-Mail: A Comparison of a Naive Bayesian and a Memory-Based Approach*. In: *Workshop on Machine Learning and Textual Information Access, Fourth European Conference on Principles and Practice of Knowledge Discovery in Databases*, S. 1–13.

[ANTI-SPAM 2002] ANTI-SPAM, BRIGHTMAIL (2002). *Brightmail: Anti-Spam Data Sheet*. http://www.brightmail.com/products-as.html.

[ARMSTRONG et al. 1995] ARMSTRONG, R., D. FREITAG, T. JOACHIMS und T. MITCHELL (1995). *WebWatcher: A Learning Apprentice for the World Wide Web*. In: *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogenous, Distributed Environments*, S. 6–12. AAAI Press.

[BALABANOVIC und SHOHAM 1995] BALABANOVIC, M. und Y. SHOHAM (1995). *Learning Information Retrieval Agents: Experiments with Automated Web Browsing*. In: *Proceedings of the AAAI Spring Symposium on Information Gathering from Heterogenous, Distributed Resources*, S. 13–18. AAAI Press.

[BLUM und MITCHELL 1998] BLUM, A. und T. MITCHELL (1998). *Combining Labeled and Unlabeled Data with Co-Training*. In: *Proceedings of the Eleventh Annual Conference on Computational Learning Theory*, S. 92–100. Morgan Kaufmann, Los Angeles, US.

[BOND und GASSER 1988] BOND, A. und L. GASSER (1988). *Readings in Distributed Artificial Intelligence*. Morgan Kaufmann, Los Angeles, US.

[BOOCH 1994] BOOCH, G. (1994). *Oject-orientated analysis and design with applications*. Addison Wesley.

[BRATMAN 1987] BRATMAN, M.E. (1987). *Intentions, Plans and Practical Reason*. Harvard University Press: Cambridge.

[BROOKS 1986] BROOKS, R.A. (1986). *A Robust Layered Control System For A Mobile Robot*. IEEE Journal of Robotics and Automation, 2(1):14–23.

[BURG 2002] BURG, B. (2002). *Official FIPA presentation*. Technischer Bericht f-out-00111, Foundation for Intelligent Physical Agents.

[CATALÀ et al. 2000] CATALÀ, N., N. CASTELL und M. MARTIN (2000). *ESSENCE: a Portable Methodology for Acquiring Information Extraction Patterns*. In: *Proceedings of the Fourteenth European Conference on Artificial Intelligence (ECAI-2000)*, S. 411–415. Berlin.

[CHANG und LIN 2002] CHANG, C. und C. LIN (2002). *LIBSVM: a Library for Support Vector Machines (Version 2.33)*. http://www.csie.ntu.edu.tw/ cjlin/papers/libsvm.ps.gz.

[COHEN 1995a] COHEN, P.R. (1995a). *Empirical Methods for Artificial Intelligence*. MIT Press, Cambridge, MA.

[COHEN 1995b] COHEN, W.W. (1995b). *Fast inductive rule induction*. In: *Proceedings of the Twelfth International Conference on Machine Learning*, S. 115–123. Morgan Kaufmann, Los Angeles, US.

[COHEN 1996] COHEN, W.W. (1996). *Learning Rules that classify e-mail*. In: *Proceedings of the AAAI Spring Symposium on Machine Learning in Information Access*, S. 18–25. AAAI Press.

[COMELLAS et al. 2000] COMELLAS, F., J. OZON und J. PETERS (2000). *Deterministic Small-World Communication Networks*. Information Processing Letter, 76(1–2):83–90.

[CÖSTER und ASKER 2000] CÖSTER, R. und L. ASKER (2000). *A Similarity-based Approach to Relevance Learning*. In: *Proceedings of the Fourteenth European Conference on Artificial Intelligence*, S. 276–280. IOS Press, Amsterdam.

[DALE und MAMDANI 2001] DALE, J. und E. MAMDANI (2001). *Open Standards for Interoperating Agent-Based Systems*. Software Focus, 1(2).

[DAVIS und SMITH 1996] DAVIS, R. und R. SMITH (1996). *Negotiation as a Metapor for Distributed Problem Solving*. Artificial Intelligence, 20(1):63–100.

[DECKER et al. 1997] DECKER, K., A. PANNU, K. SYCARA und M. WILLIAMS (1997). *Designing Behaviors for Information Agents*. In: *Proceedings of the First International Conference on Autonomous Agents*, S. 404–412. ACM Press.

[DURFEE 1988] DURFEE, E.H. (1988). *Coordination of Distributed Problem Solvers*. Kluwer Academic, Boston.

[FALCONE und CASTELFRANCHI 1998] FALCONE, R. und C. CASTELFRANCHI (1998). *Principles of Trust for MAS: Cognitive Anatomy, Social Importance and Quantification*. In: *Proceedings of the Third International Conference on Multi-Agent Systems*, S. 72–79. IEEE Computer Society.

[FININ et al. 1997] FININ, T., Y. LABROU und J. MAYFIELD (1997). *KQML as an Agent Communication Language*. Software Agents, MIT Press, Cambridge, MA.

[FIPA 1997] FIPA (1997). *FIPA97 Specification, Part I Agent Management, Issue I*. http://www.fipa.org.

[FIPA 2000a] FIPA (2000a). *FIPA ACL Message Structure Specification*. http://www.fipa.org/specs/fipa00061/XC00061E.html.

[FIPA 2000b] FIPA (2000b). *FIPA Communicative Act Library Specification*. http://www.fipa.org/specs/fipa00037/XC00037H.html.

[FIPA 2000c] FIPA (2000c). *FIPA Interaction Protocol Library Specification*. http://www.fipa.org/specs/fipa00025/XC00025E.html.

[FISCHER et al. 1996] FISCHER, K., J. MÜLLER und M. PISCHEL (1996). *Cooperative Transportation Scheduling: An Application Domain for DAI*. Journal of Applied Artificial Intelligence. Special Issue on Intelligent Agents, 10(1):1–34.

[FRANKLIN und GRAESSER 1996] FRANKLIN, S. und A. GRAESSER (1996). *Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents*. In: *Proceedings of the Third International Workshop on Agent Theories, Architectures and Languages*, S. 21–35. Springer-Verlag, Heidelberg.

[GARIA-MOLINA und YAN 1995] GARIA-MOLINA, H. und T. YAN (1995). *SIFT - A Tool for Wide-Area Information Dissemination*. In: *Proceedings of the 1995 USENIX Technical Conference*, S. 177–186. SENIX Associations, Berkeley, CA.

[GEORGOUSOPOULOS und RANA 2002] GEORGOUSOPOULOS, C. und O. RANA (2002). *An Approach to Conforming a MAS into a FIPA-compliant System*. In: *Proceedings ot the First International Joint Conference on Autonomous Agents and Multiagent Systems*, S. 968–975. ACM Press.

[GERBER et al. 1999] GERBER, C., J. SIEKMANN und G. VIERKE (1999). *Holonic Multi-Agent Systems*. Technischer Bericht RR-99-01, Deutsches Forschungszentrum für Künstliche Intelligenz.

[GNUTELLA 2001] GNUTELLA (2001). *The Gnutella Protocol Specification v0.4*. Technischer Bericht Document Revision 1.2, Clip2 Distributed Search Solutions.

[HAUGENEDER 1994] HAUGENEDER, H. (1994). *IMAGINE Final Project Report*. IMAGINE Technical Report Series.

[HAYES-ROTH 1995] HAYES-ROTH, B. (1995). *An Architecture for Adaptive Intelligent Systems*. Artificial Intelligence: Special Issue on Agents and Interactivity, 72(1–2):329–365.

[HEWITT 1986] HEWITT, C. (1986). *Offices are open Systems*. ACM Transactions on Office Information Systems, 4(3):271–287.

[HIDALGO et al. 2000] HIDALGO, J.M. GÓMEZ, M. M. LÓPEZ und E. P. SANZ (2000). *Combining Text and Heuristics for Cost-Sensitive Spam Filtering*. In: *Proceedings of the Fourth Workshop on Computational Natural Language Learning*, S. 99–102. Lisbon, Portugal.

[HIPSCHMAN 2002] HIPSCHMAN, R. (2002). *How SETI@home works*. http://setiathome.ssl.berkeley.edu/about_seti/about_seti_at_home_1.html.

[ID3C4.5 1993] ID3C4.5 (1993). *Building Classification Models: ID3 and C4.5*. http://www.cis.temple.edu/ ingargio/cis587/readings/id3-c45.html.

[IWAYAMA und TOKUNAGA 1995] IWAYAMA, M. und T. TOKUNAGA (1995). *Cluster-based text categorization: a comparison of category search strategies*. In: *Proceedings of the Eighteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 273–281. ACM Press, New York, US.

[Java Mail 2000] JAVA MAIL (2000). *Java Mail (TM) API Design Specification*. http://java.sun.com/products/javamail/JavaMail-1.2.pdf.

[JENNINGS 1993] JENNINGS, N.R. (1993). *Specification and implementation of a belief desire joint-intention architecture for collaborative problem solving*. Journal of Intelligent and Cooperative Information Systems, 2(3):289–318.

[JENNINGS 1999] JENNINGS, N.R. (1999). *Agent-Based Computing: Promise and Perils*. In: *Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, S. 1429–1436. Morgan Kaufmann, Los Angeles, US.

[JOACHIMS 1997] JOACHIMS, T. (1997). *A Probabilistic Analysis of the Rocchio Algorithm with TFIDF for Text Categorization*. In: *Proceedings of the Fourteenth International Conference on Machine Learning*, S. 143–151. Morgan Kaufmann Publishers, San Francisco, US.

[JOACHIMS 1998] JOACHIMS, T. (1998). *Text Categorization with Support Vector Machines: Learning with Many Relevant Features*. In: *European Conference on Machine Learning*, S. 137–142. Springer Verlag, Heidelberg.

[JOACHIMS 2001] JOACHIMS, T. (2001). *Tutorial: Support Vector Machines*. Technischer Bericht, Cornell University, Computer Science Department.

[KASSEL 2002] KASSEL, A. (2002). *Im 0190-Sumpf: Die mühsame Suche nach Schuldigen*. c't 2002, Heft 13, S. 92–93.

[KAZAKOV und KUDENKO 2001] KAZAKOV, D. und D. KUDENKO (2001). *Machine Learning and Inductive Logic Programming for Multi-Agent Systems*. Lecture Notes in Computer Science, 2086:246–270.

[KIRITCHENKO und MATWIN 2001] KIRITCHENKO, S. und S. MATWIN (2001). *Email Classification with Co-Training*. In: *Proceedings of the IBM Centre for Advanced Studies Conference (CASCON 2001)*. http://www.site.uottawa.ca/ stan/papers/2001/cascon2002.pdf.

[KLIR 1991] KLIR, G.J. (1991). *Facets of Systems Science*. Plenum Press.

[LAGO-FERNÁNDEZ et al. 2000] LAGO-FERNÁNDEZ, L.F., R. HUERTA, F. CORBACHO und J. SIGÜENZA (2000). *Fast Response and Temporal Coherence Oscillations in Small-World Networks*. Physical Review Letters, 84:2758–2761.

[LANG 1995] LANG, K. (1995). *NewsWeeder: Learning to filter netnews*. In: *Proceedings of the Twelfth International Conference on Machine Learning*, S. 331–339. Morgan Kaufmann, San Mateo, CA, US.

[LANGLEY et al. 2001] LANGLEY, B.K., M. PAOLUCCI und K. SYCARA (2001). *Discovery of Infrastructure in Multi-Agent Systems*. In: *Proceedings of the Workshop on Infrastructure for Agents, MAS, and Scalable MAS*.

[LASHKARI et al. 1994] LASHKARI, Y., M. METRAL und P. MAES (1994). *Collaborative Interface Agents*. In: *Proceedings of the Twelfth National Conference on Artificial Intelligence*, Bd. 1, S. 444–449. AAAI Press, Seattle, WA.

[LEWIS und CATLETT 1994] LEWIS, D. und J. CATLETT (1994). *Heterogeneous uncertainity sampling for supervised learning*. In: *Proceedings of the Eleventh Annual Conference on Machine Learning*, S. 148–156. Morgan Kaufmann, San Francisco, US.

[LUX und STEINER 1998] LUX, A. und D. STEINER (1998). *Understanding Cooperation: An Agent's Perspective*. Readings In Agents, Morgan Kaufmann.

[MAES 1990] MAES, P. (1990). *Designing Autonomous Agents*. MIT Press, Cambridge, MA.

[MAES 1995] MAES, P. (1995). *Artificial Life meets Entertainment: Life like Autonomous Agents*. Communications of the ACM, 38(11):108–114.

[MAILMARSHAL 2002] MAILMARSHAL (2002). *Marshal Software Solutions: Spam and Relay Blocking with MailMarshal*. http://www.marshalsoftware.com /issues/spam.asp/MSASP/RefID.MARSHAL/MSASP.HTML.

[MAILSHIELD 2002] MAILSHIELD (2002). *Lyris Mailshield: Data Sheet*. http://www.lyris.com/products/mailshield/essentials/ms_flyer.pdf.

[MALSCH 2001] MALSCH, T. (2001). *Naming the Unnamable: Socionics or the Sociological Turn of/to Distributed Artificial Intelligence*. Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2001), 4:155–186.

[MARKETAGENT 2003] MARKETAGENT (2003). *Marktforschungs GmbH: Studie eMail-Spamming*. http://www.marketagent.com.

[METZGER 2003] METZGER, J. (2003). *Download der Experimentalumgebung und des Antispam-Agenten*. http://www.dfki.de/ schillo/schillo/teaching/Diplomarbeit /JoergMetzger/index.html.

[METZGER et al. In Print] METZGER, J., M. SCHILLO und K. FISCHER (In Print). *A Multiagent based Peer-To-Peer Network in Java for Distributed, Efficient Spam Filtering*. In: *Proceedings of the International/Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003)*.

[METZGER et al. submitted'03] METZGER, J., M. SCHILLO und K. FISCHER (submitted'03). *Self-Organization in a Peer-To-Peer Network for Distributed, Efficient Spam Filtering*. In: *Proceedings of the International Conference on Parallel and Distributed Computing (Euro-Par 2003)*.

[MILGRAM 1967] MILGRAM, S. (1967). *The small world problem*. Psychology today, 2:60–67.

[MLADENIC 1996] MLADENIC, D. (1996). *Personal WebWatcher: design and implementation*. Technischer Bericht IJS-DP-7472, School of Computer Science, Carnegie Mellon University Pittsburgh, PA, USA.

[MÜLLEN und WELLMAN 1996] MÜLLEN, T. und M. WELLMAN (1996). *Some Issues in the Design of Market-Orientated Agents*. Intelligent Agents II, Springer-Verlag, S. 283–298.

[MÜLLER 1997] MÜLLER, J. (1997). *A Cooperation Model for Autonomous Agents*. In: MÜLLER, J.P., M. WOOLDRIDGE und N. JENNINGS, Hrsg.: *Intelligent Agents III (LNAI Volume 1193)*, S. 245–260. Springer-Verlag, Berlin.

[MÜLLER und PISCHEL 1993] MÜLLER, J. und M. PISCHEL (1993). *The Agent Architecture InteRRaP: Concept and Application*. Technischer Bericht RR-93-26, Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI).

[NAPSTER 2002] NAPSTER (2002). http://www.napster.com.

[NEUMANN und SCHMEIER 2002] NEUMANN, G. und S. SCHMEIER (2002). *Shallow Natural Language Technology and Text Mining*. Künstliche Intelligenz (2/02), S. 23–26.

[O'BRIEN und NICOL 1988] O'BRIEN, P. D. und R. C. NICOL (1988). *FIPA - Towards a Standard for Software Agents*. BT Technology Journal, 16(3):51–59.

[ORAM 2001] ORAM, A. (2001). *Peer-to-Peer: Harnessing the Power of Disruptive Technologies*. O'Reilly&Associates, Inc., CA.

[OUTLOOK 2002] OUTLOOK (2002). *Outlook 2002 Product Guide*. http://www.microsoft.com/office/outlook/evaluation/guide.asp.

[PANTEL und LIN 1998] PANTEL, P. und D. LIN (1998). *SpamCop - A Spam Classification and Organization Program*. In: *Proceedings of AAAI-98 Workshop on Learning for Text Categorization*, S. 95–98. AAAI Press.

[PAYNE 1994] PAYNE, T. (1994). *Learning Email Filtering Rules with Magi: A Mail Agent Interface*. Doktorarbeit, University of Aberdeen, Scotland.

[PAZZANI 1999] PAZZANI, M. (1999). *A Framework for Collaborative, Content-Based and Demographic Filtering*. Artificial Intelligence Review, 13(5–6):393–408.

[PINE 2002] PINE (2002). *University of Washington: Pine Information Center*. http://www.washington.edu/pine/.

[PLATT 1998] PLATT, J. (1998). *Sequential Minimal Optimization: A fast algorithm for training support vector machines*. Technischer Bericht MST-TR-98-14, Microsoft Research.

[PORTER 1980] PORTER, M.F. (1980). *An algorithm for suffix stripping*. Program, 14(3):130–137.

[POSLAD et al. 2000] POSLAD, S., P. BUCKLE und R. HADINGHAM (2000). *The FIPA-OS Agent Platform: Open Source for Open Standards*. In: *Proceedings of the Fifth International Conference and Exhibition on the Practical Application of Intelligent Agents and Multi-Agents*, S. 355–368.

[PROCMAIL 2002] PROCMAIL (2002). http://www.procmail.org.

[PROVOST 1999] PROVOST, J. (1999). *Naive-Bayes vs. Rule-Learning in Classification of Email*. Technischer Bericht TR-99-284, University of Texas at Austin, Artificial Intelligence Lab.

[QUINLAN 1987] QUINLAN, J.R. (1987). *Simplifying Decision Trees*. International Journal of Man-Machine Studies, 27:221–234.

[QUINLAN 1993] QUINLAN, J.R. (1993). *C4.5: Programs for machine learning*. Morgan Kaufmann.

[RENNIE 2000] RENNIE, J.D.M. (2000). *ifile: An Application of Machine Learning to E-Mail Filtering*. In: *Proceedings of the KDD-2000 Workshop on Text Mining, Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

[ROCCHIO 1971] ROCCHIO, J. (1971). *Relevance feedback information retrieval*. Gerard Salton, Editor, The SMART Retrieval System - Experiments in Automatic Document Processing, S. 313–323.

[RUSSEL und NORWIG 1996] RUSSEL, S. und P. NORWIG (1996). *Artificial Intelligence, A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ.

[SADEK et al. 1995] SADEK, M.D., P. PRETIER, V. CADORET, A. COZANNET, P. DUPONT, A. FERRIEUX und F. PANAGET (1995). *A Co-operative Spoken Dialogue System based Upon a Rational Agent Model: A First Implementation of the AGS Application*. In: *Proceedings of the ESCA/ETR Workshop on Spoken Dialogue Systems: Theories and Applications*.

[SAHAMI et al. 1998] SAHAMI, M., S. DUMAIS, D. HECKERMAN und E. HORVITZ (1998). *A Bayesian Approach to Filtering Junk E-mail*. In: *Proceedings of the AAAI'98 Workshop Learning for Text Categorization*, S. 55–62. AAAI Technical Report WS-98-05.

[SALTON und BUCKLEY 1988] SALTON, G. und C. BUCKLEY (1988). *Term Weighting Approaches in Automatic Text Retrieval*. Information Processing and Management, 24:513–523.

[SCHILLO In Print] SCHILLO, M. (In Print). *Self-Organization and Adjustable Autonomy: Two Sides of the Same Medal?*. Connection Science.

[SEARLE 1969] SEARLE, J.R. (1969). *Speech Acts: An Essay in the Philosophy of Language*. Cambridge University Press.

[SERRANO und OSSOWSKI 2002] SERRANO, J.M. und S. OSSOWSKI (2002). *An Approach to Agent Communication Based on Organisational Roles*. In: *Proceedings of the Sixth International Workshop on Cooperative Information Agents*, Bd. 2446 d. Reihe *Lecture Notes in Computer Science*, S. 241–248. Springer Verlag, Heidelberg.

[SH-STANDARD 1995] SH-STANDARD (1995). *Federal Information Processing Standards Publication 180-1*. National Institute of Standards and Technology.

[SHANKAR und KARYPIS 2000] SHANKAR, S. und G. KARYPIS (2000). *A Feature Weight Adjustment Algorithm for Document Categorization*. Proceedings of the Sixth International Conference on Knowledge Discovery and Data Mining (ACM SIGKDD 2000).

[SHETH 1994] SHETH, B.D. (1994). *A Learning Approach to Personalized Information Filtering*. Doktorarbeit, Electrical Engineering and Computer Science Dept., MIT, Cambridge, Mass.

[SIERRA et al. 1998] SIERRA, J.M., N. JENNINGS, P. NORIEGA und S. PARSONS (1998). *A Framework for Argumentation-Based Negotiation*. In: *Proceedings of the Fourth International Workshop on Agent Theories Architectures and Languages*, Bd. 1365 d. Reihe *Lecture Notes in Computer Science*, S. 177–192. Springer-Verlag, Heidelberg.

[SMITH et al. 1994] SMITH, D.C., A. CYPHER und J. SPOHRER (1994). *KidSim: Programming Agents Without a Programming Language*. Communications of the ACM, 37(7):55–67.

[SMITHSON und MOREAU 2002] SMITHSON, A. und L. MOREAU (2002). *Engineering an Agent-Based Peer-To-Peer Resource Discovery System*. In: *Proceedings of the First International Workshop on Agents and Peer-To-Peer Computing*, Bd. 2530.

[SORENSEN und ELLIGOTT 1995] SORENSEN, H. und M. ELLIGOTT (1995). *PSUN: A Profiling System for Usenet News*. In: *Proceedings of the CIKM'95 Workshop on Intelligent Information Agents*.

[SPAMASSASSIN 2002] SPAMASSASSIN (2002). *SpamAssassin Homepage*. http://www.spamassassin.org/index.html.

[SPAMBOUNCER 2002] SPAMBOUNCER (2002). *SpamBouncer Version 1.4 Content Page*. http://www.spambouncer.org.

[SPAMBUSTER 2002] SPAMBUSTER (2002). *Contact Plus Corporation: SpamBuster Homepage*. http://www.contactplus.com/spam/spam.htm.

[SPAMEATER 2002] SPAMEATER (2002). *High Mountain Software: Spameater Homepage*. http://www.hms.com/spameater.asp.

[SPAMKILLER 2002] SPAMKILLER (2002). *Mc Afee.com: Spamkiller Homepage*. http://www.mcafee.com/myapps/msk/default.asp.

[SPAMVACCINE 2002] SPAMVACCINE (2002). *matterform media: How Spam Vaccine Works*. http://www.matterform.com/index.php?page=/spamvaccine/how.html.

[STATISTIC 2002] STATISTIC, SPAM (2002). *Spam received since 1996 listed by an user of email-accounts*. http://www.raingod.com/angus/Computing/Internet/Spam/Statistics/index.html.

[STROGATZ und WATTS 1998] STROGATZ, S.H. und D. WATTS (1998). *Collective dynamics of 'small-world' networks*. Nature, 393:440–442.

[SUKANEN 2002] SUKANEN, J. (2002). *Peer-to-peer communication*. Technischer Bericht Seminar on Internetworking, Helsinki University of Technology, Telecommunications Software and Multimedia Laboratory.

[SYCARA 1998] SYCARA, K.P. (1998). *Multiagent Systems*. AI Magazine, 10(2):79–93.

[TEL 1994] TEL, G. (1994). *Introduction to Distributed Algorithms*. Cambridge University Press.

[TRAVERS 1988] TRAVERS, M. (1988). *Animal Construction Kit*. Artificial Life, Addison-Wesley, S. 421–442.

[VAPNIK et al. 1999] VAPNIK, V., D. WU und H. DRUCKER (1999). *Support Vector Machines for Spam Categorization*. IEEE Transactions on Neural Networks, 10(5):1048–1054.

[VAPNIK 1995] VAPNIK, V.N. (1995). *The Nature of Statistical Learning Theory*. Springer Verlag, Heidelberg.

[VAPNIK und WU 1998] VAPNIK, V.N. und D. WU (1998). *Support Vector Machine for Text Categorization*. AT&T Research Labs, http://citeseer.nj.nec.com/347263.htm.

[Vipul's Razor 2002] VIPUL'S RAZOR (2002). *Vipul's Razor Homepage*. http://razor.sourceforge.net.

[WEISS 1999] WEISS, G. (1999). *Multiagent Systems: A Modern Approach To Distributed Artificial Intelligence*. MIT Press, Cambridge, MA.

[WEINSTEIN 1992] WEINSTEIN, S. (1992). *The Elm Filter System Guide*. Technischer Bericht http://www.colostate.edu/Services/acns/Filter.pdf, Datacomp Systems Inc.

[WIENER et al. 1995] WIENER, E., J. PEDERSON und A. WEIGEND (1995). *A Neural Network Approach to Topic Spotting in Text*. In: *Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval*, S. 317–332.

[WITTEN und FRANK 1999] WITTEN, I.H. und E. FRANK (1999). *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann.

[WOOLDRIDGE und JENNINGS 1995] WOOLDRIDGE, M. und N. JENNINGS (1995). *Intelligent Agents: Theory and Practice*. In Knowledge Engineering Review, Cambridge University Press, 10(2):115–152.

[WOOLDRIDGE et al. 2000] WOOLDRIDGE, M., N. JENNINGS und D. KINNY (2000). *The Gaia Methodology for Agent-Orientated Analysis and Design*. Autonomous Agents and Multi-Agent Systems, 3(3):285–312.

[YANG 1994] YANG, Y. (1994). *Expert Network: Effective and Efficient Learning from Human Decisions in Text Categorization and Retrieval*. In: *Proceedings of the Seventeenth Annual International Conference on Research and Development in Information Retrieval*, S. 13–22. Springer Verlag, Heidelberg.

[YANG 1999] YANG, Y. (1999). *An Evaluation of Statistical Approaches to Text Categorization*. Information Retrieval, 1(1-2):69–90.

[YANG und LIU 1999] YANG, Y. und X. LIU (1999). *A re-examination of text categorization methods*. In: *Proceedings of ACM SIGIR Conference on Research and Development in Information Retrieval*, S. 42–49. ACM Press, New York, US.

Index