(19) Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 60 2004 013 169 T2** 2009.05.07

(12) # Übersetzung der europäischen Patentschrift

(97) **EP 1 665 678 B1**
(21) Deutsches Aktenzeichen: **60 2004 013 169.5**
(86) PCT-Aktenzeichen: **PCT/GB2004/004003**
(96) Europäisches Aktenzeichen: **04 768 549.0**
(87) PCT-Veröffentlichungs-Nr.: **WO 2005/032069**
(86) PCT-Anmeldetag: **17.09.2004**
(87) Veröffentlichungstag der PCT-Anmeldung: **07.04.2005**
(97) Erstveröffentlichung durch das EPA: **07.06.2006**
(97) Veröffentlichungstag der Patenterteilung beim EPA: **16.04.2008**
(47) Veröffentlichungstag im Patentblatt: **07.05.2009**

(51) Int Cl.[8]: ***H04L 12/56*** (2006.01)

(30) Unionspriorität:
0322491 **25.09.2003** **GB**

(73) Patentinhaber:
British Telecommunications Public Ltd. Co., Greater, London, GB

(74) Vertreter:
BEETZ & PARTNER Patentanwälte, 80538 München

(84) Benannte Vertragsstaaten:
AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LI, LU, MC, NL, PL, PT, RO, SE, SI, SK, TR

(72) Erfinder:
BONSMA, Erwin Rein, Ipswich Suffolk IP3 8HD, GB

(54) Bezeichnung: **VIRTUELLE NETZWERKE**

Beschreibung

[0001] Die vorliegende Erfindung betrifft virtuelle Netzwerke und insbesondere, aber nicht ausschließlich, in dem Kontext von verteilten Systemen, wie Peer-to-Peer-Systeme, insbesondere solche ohne zentralisierten Speicher oder Steuerung.

[0002] Das DART-System, wie beschrieben in US2002/0163889 weist ein Adhoc-Netzwerk einer Vielzahl von Knoten auf, die sich selbst in dem Adhoc-Netzwerk anordnen können. In diesem Netzwerk können sich Verbindungen zwischen Knoten ändern oder verschwinden aufgrund einer relativen Bewegung der Knoten oder eines Ausfalls eines Kommunikationspfads aus welchen Gründen auch immer. Das DART-System zielt darauf ab, die Änderungen zu bewältigen durch Verwendung einer Form eines Routing-Baum-Algorithmus. Der Routing-Baum-Algorithmus funktioniert, Knoten über die Präsenz von benachbarten Knoten zu benachrichtigen, und in diesem Prozess wird jeder neue Knoten versorgt mit einer Liste von Identifizierern, die Verbindungen zwischen dem benachrichtigenden Knoten zurück zu einem einzelnen Wurzel- bzw. Root-Knoten identifizieren. In dem DART-Netzwerk wird erwartet, dass jeder Knoten kontaktiert wird durch eine Vielzahl von Routing-Knoten und somit eine Information über eine signifikante Anzahl von Routen (dargestellt durch eine Liste von Verbindungen) zurück zu dem Wurzel-Knoten sammelt. Das DART-System hat den Vorteil, dass, wenn eine Verbindung zu einem Knoten ausfällt, der Knoten sich nur auf seine interne Liste von alternativen Routen beziehen muss, um eine funktionierende Route zu wählen, um auf einen bestimmten Knoten zuzugreifen.

[0003] Gemäß einem Aspekt der vorliegenden Erfindung ist vorgesehen ein Verfahren zum Betreiben eines virtuellen Netzwerks, das eine Vielzahl von Knoten hat, wobei jeder Knoten eine Liste zum Speichern bis zu einer vorgegebenen Anzahl von Adressen anderer derartiger Knoten hat, das aufweist:

(i) Empfangen einer Nachricht, die eine Verbindung zwischen einem ersten Knoten und einem zweiten Knoten anfordert;

(ii) Bestimmen, ob sowohl der erste Knoten als auch der zweite Knoten jeweils eine Anzahl von Adressen in seiner Liste hat, die geringer ist als die vorgegebene Anzahl;

(iii) in dem Fall, dass diese Bedingung erfüllt ist, Einfügen der Adresse des ersten Knotens in die Liste des zweiten Knotens und Einfügen der Adresse des zweiten Knotens in die Liste des ersten Knotens;

(iv) in dem Fall, dass diese Bedingung nicht erfüllt ist, Bestimmen, ob der erste Knoten eine Anzahl von Adressen in seiner Liste hat, die zumindest zwei weniger als die vorgegebene Anzahl ist, und wenn dem so ist

(a) Auswählen von der Liste des zweiten Knotens die Adresse eines dritten Knotens;

(b) Entfernen der Adresse des dritten Knotens von der Liste des zweiten Knotens;

(c) Entfernen der Adresse des zweiten Knotens von der Liste des dritten Knotens; und

(d) Einfügen der Adresse des zweiten Knotens in die Liste des ersten Knotens und Einfügen der Adresse des dritten Knotens in die List des ersten Knotens;

(e) Einfügen der Adresse des ersten Knotens in die Liste des zweiten Knotens und Einfügen der Adresse des ersten Knotens in die Liste des dritten Knotens.

[0004] In einem anderen Aspekt sieht die Erfindung einen Knoten eines virtuellen Netzwerks vor, wobei der Knoten durch Speichermittel, die eine Liste zum Speichern bis zu einer vorgegebenen Anzahl von Adressen anderer derartiger Knoten enthalten, und Verarbeitungsmittel definiert ist, die programmiert sind, die folgenden Operationen durchzuführen:

Empfangen von Nachrichten;

Antworten auf Nachrichten, die eine Information über den Inhalt der Liste anfordern;

Erfüllen der empfangenen Anforderungen, eine Adresse von der Liste zu entfernen;

Erfüllen der Nachricht, die ein Einfügen einer Adresse in die Liste anfordert; und

als Antwort auf einen Empfang einer Nachricht, die eine Verbindung zwischen dem Knoten und einem zweiten Knoten anfordert:

(A) Erzeugen einer Nachricht an den zweiten Knoten, die eine Information über den Inhalt seiner Liste anfordert;

(B) Bestimmen, ob sowohl der erste Knoten als auch der zweite Knoten jeweils eine Anzahl von Adressen in seiner Liste hat, die geringer als die vorgegebene Anzahl ist;

(C) in dem Fall, dass diese Bedingung erfüllt ist, Einfügen in seine Liste der Adresse des zweiten Knotens und Erzeugen einer Nachricht an den zweiten Knoten, die den zweiten Knoten auffordert, zu seiner Liste die Adresse des Knotens hinzuzufügen;

(D) in dem Fall, dass diese Bedingung nicht erfüllt ist, Bestimmen, ob der Knoten eine Anzahl von Adressen in seiner Liste hat, die zumindest um zwei weniger als die vorgegebene Anzahl ist, und wenn dem so ist

(a) Auswählen von der Liste des zweiten Knotens die Adresse eines dritten Knotens;

(b) Einfügen der Adresse des zweiten Knotens in die Liste des ersten Knotens und Einfügen der Adresse des dritten Knotens in die Liste des ersten Knotens;

(c) Erzeugen einer Nachricht an den zweiten Knoten, die das Entfernen der Adresse des dritten Knotens von der Liste des zweiten Knotens und ein Einfügen der Adresse des Knotens anfordert;

und
(d) Erzeugen einer Nachricht an den dritten Knoten, die das Entfernen der Adresse des zweiten Knotens von der Liste des dritten Knotens und ein Einfügen der Adresse des Knotens anfordert.

[0005] Andere bevorzugte Aspekte der Erfindung werden in den Ansprüchen definiert.

[0006] Einige Ausführungsbeispiele der Erfindung werden nun beschrieben, auf beispielhafte Weise unter Bezugnahme auf die beigefügten Zeichnungen, wobei:

[0007] Fig. 1 ein Blockdiagramm eines Computers ist, der in einem Ausführungsbeispiel der Erfindung verwendet wird;

[0008] Fig. 1A ein Ablaufdiagramm ist, das eine Daten-Abruf-Operation unter Verwendung von primären und sekundären virtuellen Netzwerken zeigt;

[0009] Fig. 2 eine schematische Diagrammdarstellung der Verwaltung von Verbindungen zwischen Knoten eines Computernetzwerks ist;

[0010] Fig. 3 bis Fig. 10 Ablaufdiagramme sind, die Aspekte des Betriebs eines Knotens des sekundären virtuellen Netzwerks zeigen;

[0011] Fig. 11 bis Fig. 14 und Fig. 16 Ablaufdiagramme sind, die Aspekte des Betriebs eines Knotens des primären virtuellen Netzwerks zeigen; und

[0012] Fig. 15 ein schematisches Diagramm ist, das den Fluss von Nachrichten während des Prozesses darstellt, der in Fig. 14 dargestellt wird.

KNOTEN

[0013] In dieser Beschreibung wird Bezug genommen auf berechnende Knoten, die Verarbeitungs-, Speicher- und Kommunikations-Fähigkeiten haben. Ein berechnender Knoten kann ein Computer oder eine sonstige Vorrichtung sein, oder – mit der Anmerkung, dass ein einzelner Computer eine Anzahl von unabhängigen Programmen oder Prozessen darauf laufen haben kann – kann solch ein Programm oder Prozess sein. Ein Element von gespeicherten Daten kann auch angesehen werden als ein eindeutiger Knoten, obwohl eine Anzahl von solchen Elementen durch ein einzelnes Programm oder einen Prozess bedient werden kann.

[0014] Diese Beschreibung nimmt an, dass jeder berechnende Knoten verbunden ist mit einer Kommunikationsinfrastruktur, die zum Beispiel ein Telekommunikationsnetzwerk sein kann, wie ein IP(Internet Protocol)-Netzwerk, so dass Nachrichten an ihn gesendet werden können. Somit bildet jeder berechnende Knoten auch einen Knoten in der Kommunikationsinfrastruktur.

[0015] Bezug wird auch genommen auf virtuelle Knoten, die zu einem virtuellen Netzwerk gehören. Die Unterscheidung ist wichtig, da ein berechnender Knoten zwei oder mehr virtuelle Knoten haben kann (die möglicherweise zu verschiedenen virtuellen Netzwerken gehören), die zu ihm gehören. Wie sein Name andeutet, existiert ein virtueller Knoten nicht in einem physikalischen Sinn: stattdessen, wie offensichtlich wird, wird seine Existenz hergestellt durch gespeicherte Daten, die Verbindungen zwischen virtuellen Knoten definieren und folglich auch das virtuelle Netzwerk definieren, zu dem sie gehören.

[0016] Notwendigerweise muss ein virtueller Knoten zu einem berechnenden Knoten gehören, der ihn mit Verarbeitungs-, Speicher- und Kommunikationsfähigkeiten vorsieht: Referenzen zum Senden, Empfangen und Verarbeiten von Nachrichten durch einen virtuellen Knoten betreffen solch ein Senden, Empfangen oder Verarbeiten durch den berechnenden Knoten im Namen des virtuellen Knotens.

[0017] Ein Beispiel wird in Fig. 1 gezeigt. Ein Computer hat die üblichen Komponenten, nämlich einen Prozessor **1**, Speicher **2**, Anzeige **3**, Tastatur **4** und eine Kommunikationsschnittstelle **5** für eine Kommunikation über ein Netzwerk **10**.

[0018] Der Speicher **2** enthält ein Betriebssystem und andere Programme (nicht gezeigt) und Datendateien, wie die gezeigte Textdatei **20**. Er hat auch einen Speicher **21**, der ein Label **21** enthält, das der Textdatei **20** und seiner eigenen Adresse **21b** entspricht. Zusätzlich hat er eine Adressenliste **22** und ein Unterstützungsprogramm **23**, die zusammen die Existenz eines Knotens eines virtuellen Netzwerks auf dem Computer definieren. Dieser Knoten hat eine Adresse **24**. Auch gezeigt werden eine Adressenliste **25** und ein Unterstützungsprogramm **26**, welche zusammen die Existenz eines Knotens eines anderen virtuellen Netzwerks auf dem Computer definieren. Dieser Knoten hat eine Adresse **27**. Die Adressen, die in den Listen **22**, **25** gespeichert sind, sind die Adressen von anderen Knoten in demselben virtuellen Netzwerk.

VERWEIS- BZW. LOOK-UP-SYSTEM

[0019] Wir beschreiben nun ein verteiltes Verweissystem, obwohl dies nur ein mögliches Beispiel einer Anwendung für die Erfindung ist. Dieses System ermöglicht Benutzern, Kommentare zu einer Webseite zu geben. Wenn ein Benutzer diese Seite besucht, hat er die Gelegenheit, auch die Kommentare zu betrachten, die andere Benutzer gemacht haben. Der Kommentar wird gespeichert auf dem Computer des Benutzers, der den Kommentar beigetragen hat (zum

Beispiel als eine Textdatei).

[0020] Der die Webseite betrachtende Benutzer (oder eher sein Computer) hat die URL (Uniform Resource Locator) der Webseite und was erforderlich ist, ist ein Mechanismus, wodurch er die Kommentare abrufen kann. In diesem Beispiel ist der Mechanismus wie folgt:
Die Textdatei ist auf dem Computer des Benutzers gespeichert, der den Kommentar beigetragen hat und gehört zu einem Knoten eines virtuellen Netzwerks des Typs, der in unserer internationalen Patentanmeldung Nummer WO 03/034669 beschrieben wurde [Ref. des Agenten A30044], wie es auch andere Textdateien, die Kommentare über andere Webseiten enthalten, und möglicherweise andere Dateien ohne Bezug ebenfalls. Dieses virtuelle Netzwerk (das in Zusammenhang mit der vorliegenden Beschreibung als das primäre virtuelle Netzwerk oder einfach das primäre Netzwerk bezeichnet wird) dient dazu, zu ermöglichen, eine Nachricht an einen Knoten zu senden, ohne seine Adresse zu kennen, vorausgesetzt, er hat ein identifizierendes Label. Obgleich dieser Typ eines Netzwerk funktionieren kann mit eindeutigen Labels (eines pro Knoten), sind in diesem Beispiel die Label nicht eindeutig: stattdessen haben alle Knoten, die zu Textdateien gehören, die Kommentare über eine bestimmte Webseite enthalten, dasselbe Label. Dieses Label ist eine Hash-Funktion der URL der Webseite. Dieses virtuelle Netzwerk bietet einen Abruf-Mechanismus, der nur einen Knoten erreicht.

[0021] Die Textdatei ist auch zugehörig zu einem Knoten eines zweiten virtuellen Netzwerks. Dieses (das sekundäre virtuelle Netzwerk) enthält nur Knoten, die zu Textdateien gehören, die Kommentare über die eine bestimmte Webseite enthalten.

[0022] Es ist jedoch anzumerken, dass, während die Verwendung eines primären Netzwerks in Übereinstimmung mit unserer vorgenannten internationalen Patentanmeldung bevorzugt ist, es nicht wesentlich ist. In der Tat ist es nicht wesentlich, überhaupt ein virtuelles Netzwerk zu verwenden; ein anderer primärer Abruf-Mechanismus, der ein Label empfängt und die Adresse eines Knotens zurücksendet, die diesem entspricht, kann stattdessen verwendet werden.

[0023] Der Computer, der einen Kommentar abgibt, ist wie in Fig. 1 gezeigt und muss

- einen Knoten in dem primären Netzwerk erzeugen. Dieser Knoten hat ein Label **21a** und eine Netzwerkadresse **24**.
- einen Knoten in dem sekundären Netzwerk erzeugen. Dieser Knoten hat eine Netzwerkadresse **27**.

[0024] Zuerst sind die Adressenlisten **22**, **25** leer, außer dass die Liste **22** Bootstrap- bzw. Start-Verbindungen enthält. Die Selbstorganisation der Netzwerke, um sicherzustellen, dass die Liste **22** die Label und Adressen einiger anderer Knoten des primären Netzwerks enthält und dass die Liste **25** die Adressen einiger anderer Knoten des sekundären Netzwerks enthält, wird später beschrieben. Vorläufig wird das System beschrieben unter der Annahme, dass diese Label und Adressen präsent sind.

[0025] Einige Worte über Adressen sind an diesem Punkt angebracht. Der Knoten, der durch die Textdatei **20** gebildet wird, der Knoten des primären virtuellen Netzwerks und der Knoten des zweiten virtuellen Netzwerks, während sie konzeptionell eine einzelne Identität haben, haben ihre eigenen Adressen. Es wäre möglich, jedem Knoten eine eindeutige Adresse in dem Kommunikationsnetzwerks **10** zuzuteilen, obgleich in der Praxis dies nicht besonders zweckmäßig ist. In unserer bevorzugten Implementierung hat jeder Knoten eine Adresse, die aus drei Teilen besteht:

- Eine Internet-Adresse, die den berechnenden Knoten "lokalisiert", zum Beispiel 130.146.209.15
- eine Anschlussnummer, die einen bestimmten Kommunikationsanschluss an dem berechnenden Knoten lokalisiert. Anschlüsse sind ein Standard-Teil von Internet-Adressen. Sie ermöglichen zum Beispiel, dass verschiedene unabhängige Anwendungsprogramme unabhängig Nachrichten senden und empfangen. Das heißt jedes würde Nachrichten an seinem eigenen Anschluss empfangen und würde nicht empfangen oder "verwirrt sein" durch Nachrichten, die für andere Anwendungsprogramme bestimmt sind. Die Internetadresse zusammen mit der Anschlussnummer kann betrachtet werden als die Netzwerkadresse (da sie ein Teil der Kommunikationsprotokolle ist, wie TCP/IP, die verwendet werden). Die Netzwerkadresse für alle primären und sekundären Knoten kann dieselbe sein, jedoch muss sie dies nicht sein. Zum Beispiel können alle Nachrichten für primäre Knoten an einem anderen Anschluss als von dem empfangen werden, an dem sekundäre Nachrichten empfangen werden (was ein Weg ist, um zwischen solchen Nachrichten zu unterscheiden).
- Ein Knoten-Identifizierer (ein Ganzzahl-Wert), der den spezifischen Knoten lokalisiert, für den die Nachricht vorgesehen ist. Wenn zum Beispiel alle Nachrichten auf dem primären Netzwerk an einem bestimmten Anschluss empfangen werden, gibt es noch einen lokal eindeutigen Identifizierer, der zu jedem Knoten gehört. Somit, wenn es mehrfache Knoten gibt, ist es offensichtlich, für welchen Knoten die Nachricht vorgesehen ist. Dieser Knoten-Identifizierer ist eine anwendungsspezifische Adresserweiterung (sie ist kein Teil des Standard-Internetprotokolls). Sie ist einfach enthalten in der Nachricht, die gesendet wird. Der Prozess, der sie empfangt, „weiß" dies und unter-

sucht diesen Knoten-Identifizierer, um zu bestimmen, an welchen Knoten die Nachricht weitergeleitet werden soll.

[0026] Es ist möglich, dass beide Knoten die gleiche Netzwerkadresse haben, aber dies ist nicht notwendigerweise so. Nicht jeder Knoten hat einen eigenen Anschluss (teilweise, da die Anzahl von verfügbaren Anschlüssen etwas begrenzt ist), aber einer kann auch zwei Anschlüsse haben (und folglich zwei verschiedene Netzwerkadressen): einen für das primäre Netzwerk und einen für das sekundäre Netzwerk. Typischerweise gibt es eine Anzahl von sekundären Netzwerken, die alle denselben Anschluss verwenden können.

[0027] Es sollte angemerkt werden, dass im Folgenden Bezugnahmen auf die Adresse eines Knotens die vollständige Adresse dieses Knotens betreffen.

[0028] Ein besonders attraktiver Ansatz ist, vorzusehen, dass eine Textdatei und die primären und sekundären Knoten alle denselben Knoten-Identifizierer (und IP-Adresse) haben, wobei nur die Anschlussnummern verschieden sind. Solch ein Adressierungsprotokoll kann eine Gelegenheit bieten zum Vereinfachen eines Teils der Verarbeitung derart, dass, wenn man die Adresse eines Knotens hat und die Adresse eines anderen Knotens erfordert, der zu diesem gehört, die Adresse der letzten Knotens abgeleitet werden kann von der des ersteren, anstatt sie nachschlagen zu müssen. In der folgenden Beschreibung jedoch wurde keine solche Vereinfachung gemacht, so dass diese Verfahren mit jedem Adressprotokoll arbeiten.

[0029] Der Computer, der eine Webseite betrachtet, ruft die zugehörigen Kommentare ab durch

- Anwenden derselben Hash-Funktion auf den URL, um das Label zu erlangen
- Senden einer Abfrage (die das Label enthält) auf dem primären virtuellen Netzwerk, um die Adresse eines Knotens zu erlangen
- unter Verwendung der gefundenen Adresse Senden einer Abfrage auf dem zweiten virtuellen Netzwerk, um die Adressen von mehreren (oder sogar allen) anderen Knoten auf dem zweiten virtuellen Netzwerk zu erlangen
- Verwenden dieser Adressen, um die Kommentare zu Anzeige abzurufen.

[0030] Es ist anzumerken, dass der abrufende Computer nicht notwendigerweise Knoten der virtuellen Netzwerke enthalten muss; er kann ein herkömmlicher Computer sein, der mit Software geladen ist zur Implementierung des Abrufprozesses und mit einer Kommunikationsschnittstelle, so dass er mit den Computern kommunizieren kann, auf denen sich die Knoten der virtuellen Netzwerke befinden. Dieses Verfahren wird gezeigt in dem Ablaufdiagramm der Fig. 1A, und geht wie. folgt:

[0031] Schritt **30**: nachdem der Benutzer eine URL eingegeben hat (oder einen Hyperlink aufgerufen hat), ruft der Computer die entsprechende Webseite ab. Dieser Schritt ist völlig herkömmlich.

[0032] Schritt **31**: eine Hash-Funktion wird angewendet auf die URL, um ein Label zu erlangen. Wie in unserer früheren internationalen Patentanmeldung diskutiert, kann dies den SHA-1-Algorithmus verwenden.

[0033] Schritt **32**: eine „Finden"-Nachricht, die dieses Label und die Netzwerkadresse des abrufenden Computers enthält, wird an einen Knoten des primären Netzwerks gesendet. Offenkundig ist es notwendig für den Computer, in Besitz von zumindest einer solchen Adresse zu sein.

[0034] Schritt **33**: der abrufende Computer empfangt eine „Gefunden"-Nachricht von dem primären Netzwerk. Diese Nachricht enthält das Label und die Adresse eines Knotens, der gefunden wurde, sowie die Adressen des zugehörigen Knotens des sekundären Netzwerks und des Kommentars. Ein Timeout-Mechanismus kann enthalten sein, um den Prozess abzubrechen, wenn keine „Gefunden"-Nachricht in einer angemessenen Zeit gefunden wird.

[0035] Schritt **34**: in diesem Beispiel ist das primäre Netzwerk derart ausgebildet, dass es immer das Label und die Adresse des Knotens zurückgibt, der ein Label hat, das am nächsten ist zu dem Label, das in der „Finden"-Nachricht enthalten ist. So wird eine Überprüfung durchgeführt, um zu sehen, ob das Label, das zurückgesendet wird, das gleiche ist wie das nachgefragte, und wenn nicht, wird das Verfahren beendet. Siehe unten für eine Erklärung der Bedeutung von "nächste"

[0036] Schritt **35**: unter der Annahme, dass die Label übereinstimmen, führt der abrufende Computer einen Prozess aus (wird im Detail unten beschrieben), wobei er die Adresse verwendet, die durch die „Gefunden"-Nachricht zurückgesendet wird, um weitere Adressen unter Verwendung des zweiten Netzwerks abzurufen.

[0037] Schritt **36**: Diese Adressen werden dann verwendet, aus den "eintragenden bzw. posting" Computern die Textdateien abzurufen, welche die Kommentare enthalten.

DAS SEKUNDÄRE VIRTUELLE NETZWERK

[0038] Das Ziel dieses Netzwerks ist, eine Gruppe von Knoten in ein einzelnes virtuelle Netzwerk zu selbstorganisieren, das dann verwendet werden kann, um alle Knoten zu entdecken, die Teil der Grup-

pe sind. Die Hauptanforderung ist, dass das resultierende Netzwerk alle Knoten enthält. Eine weitere Anforderung ist, dass die Systembelastung, die erforderlich ist, um das Netzwerk erzeugen und zu unterhalten, gleichmäßig über alle Knoten verteilt ist. Dies ist nicht nur am „fairsten", das wichtig ist, wenn verschiedene Benutzer ihre Ressourcen zu einer verteilten Anwendung beitragen, es hilft auch, das System gegen Überlastung zu schützen.

[0039] Das Netzwerk hat folglich die folgenden Eigenschaften:

- Die Anzahl von Verbindungen, die durch jeden Knoten unterhalten werden, ist vorzugsweise dieselbe.
- Alle Verbindungen sind bidirektional. Infolgedessen ist die Anzahl von Verbindungen zu einem Knoten für jeden Knoten ebenso gleich. Dies ist wichtig, da es die Anzahl von Nachrichten beeinflusst, die ein Knoten empfangt und handhaben muss.
- Es hat eine "flache" Struktur. Die Knoten ordnen sich selbst nicht hierarchisch an. Infolgedessen ist die Systembelastung gleichmäßig über allen Knoten verteilt.

Struktur jedes Knotens

[0040] Jeder Knoten hat die folgenden Daten, die zu ihm gehören:

- Mehrere Verbindungen zu anderen Knoten. Jede Verbindung ist einfach die Adresse eines anderen Knotens. Zugehörig zu jeder Verbindung ist ein Status, der "bestätigt" oder "unbestätigt" sein kann. Jede Knoten kann nur eine maximale Anzahl von Verbindungen unterhalten, die gegeben ist durch den Systemweiten Parameter L. Ein typischer Wert für L ist zum Beispiel 6. Es ist nicht wesentlich, dass dieser Parameter derselbe ist für alle Knoten; aber es wird kein Vorteil erzielt, wenn sie verschieden sind.
- Eine Liste von Reserve-Verbindungen oder kurz Reserven. Jede Reserve ist einfach die Adresse eines anderen Knotens. Die Reserven werden verwendet durch das Selbstorganisations-Verfahren, um das virtuelle Netzwerk aufzubauen. Ein Knoten fügt andere Knoten als Reserven hinzu, wenn er benachrichtigt wird über einen Knoten, den er nicht als eine Verbindung hinzufügen kann, da er entweder bereits eine Verbindung zu dem Knoten hat oder er bereits die maximale Anzahl von Verbindungen hat. Die Anzahl der Reserven, die ein Knoten unterhalten kann, ist auch begrenzt, und wird gegeben durch den Systemweiten Parameter S. Ein typischer Wert für S ist zum Beispiel 3. Die Liste der Reserve-Verbindungen ist im Allgemeinen nicht wesentlich, ist aber sehr wertvoll beim Vorsehen eines zusätzlichen Mechanismus, wodurch eine Verbindung, die nicht lokal aufgenommen werden kann, an einen anderen Punkt in dem virtuellen Netzwerk verbreitet werden kann. Jedoch ist die Verwendung von Reserve-Verbindungen (oder eines ähnlichen Ausbreitungsmechanismus) notwendig in Systemen, in denen die ankommenden „Benachrichtigen"-Nachrichten immer am selben Knoten (oder an einem einer sehr kleinen Anzahl von Knoten) des sekundären Netzwerks ankommen.

Nachrichten

[0041] Um sich in ein Netzwerk selbstzuorganisieren und zu entdecken, welche Knoten Teil eines gegebenen Netzwerks sind, senden Knoten Nachrichten aneinander: Die folgenden Typen von Nachrichten werden verwendet durch das sekundäre Netzwerk:

- AddLink(VerbindungHinzufügen)-Nachricht mit:

- Adresse des Senders
- Adresse des Empfängers

[0042] Sie wird durch einen Knoten (Sender) zu einem anderen Knoten (Empfänger) gesendet zur Anforderung einer gegenseitigen Verbindung.

- ChangeLink(VerbindungÄndern)-Nachricht mit:

- Adresse des Senders
- Adresse der Empfängers
- Adresse des Betreffenden

[0043] Sie wird durch einen Knoten (X) zu einem anderen Knoten (Y) gesendet, um anzufordern, dass er eine seiner Verbindungen (Z) zu einer Verbindung zu sich selbst (X) ändert. Das Protokoll ist derart, dass X eine ähnliche Nachricht an Z sendet, die ihn auffordert, dass er seine Verbindung zu Y mit einer Verbindung zu sich selbst (X) ändert. So fordert tatsächlich X an, sich in die Verbindung einzusetzen, die momentan zwischen Y und Z besteht.

- LinkAdded(VerbindungHinzugefügt)-Nachricht mit:

- Adresse des Senders
- Adresse des Empfängers

[0044] Sie wird verwendet, um einen Knoten zu benachrichtigen, dass der Sender gerade eine Verbindung zu ihm hinzugefügt hat.

- LinkError(Verbindungsfehler)-Nachricht mit:

- Adresse der Senders
- Adresse der Empfängers
- Adresse des Betreffenden
- Fehlercode

[0045] Sie wird verwendet, um einen Knoten zu benachrichtigen, dass es ein Problem zu geben scheint mit einer seiner Verbindungen. Zum Beispiel kann der betreffende Knoten möglicherweise nicht antwor-

ten, oder die Verbindung kann nicht gegenseitig sein. Sie umfasst einen Fehlercode, um den Typ des Fehlers anzuzeigen.

- Links(Verbindungen)-Nachricht mit:

- Adresse der Senders
- Adresse des Empfängers
- Adressen aller Verbindungen
- Referenzwert
- die Verbindungen-Nachricht kann auch einige andere Daten von dem Senderknoten enthalten. In dem Beispiel des Webseitenkommentars ist dies die Adresse des zugehörigen Kommentars.

[0046] Sie enthält alle aktuellen Verbindungen des sendenden Knotens. Sie wird immer gesendet als Antwort auf eine LinksQuery-Nachricht. Die Referenz kann verwendet werden, um die spezifizierte Abfrage zu unterscheiden, auf die geantwortet wird.

- LinksQuery(VerbindungenAbfrage)-Nachricht mit:

- Adresse des Senders
- Adresse des Empfängers
- Referenzwert

[0047] Sie wird verwendet, um einen Knoten aufzufordern, eine Verbindungen-Nachricht als Antwort zu senden (die seine aktuellen Verbindungen enthält).

- Notify(Benachrichtigen)-Nachricht mit:

- Adresse des Senders
- Adresse des Empfängers
- Adresse des Betreffenden
- Benachrichtigungspegel

[0048] Sie wird verwendet, um einen Knoten über einen anderen Knoten in dem Netzwerk zu benachrichtigen. Der Benachrichtigungspegel wird verwendet zur Steuerung und Begrenzung der Ausbreitung von Benachrichtigungs-Nachrichten. Wie hier beschrieben, wird keine Senderadresse verwendet, ist aber nützlich zum Debugging oder, wenn gewünscht, zum Senden von Bestätigungen.

Aufbau des sekundären Netzwerks

[0049] Das System lässt eine Gruppe von Knoten sich selbstorganisieren in ein einzelnes virtuelles Netzwerk, so dass, wenn einer die Adresse eines Knotens hat, er die Adressen von anderen in der Gruppe finden kann. Dieser Abschnitt beschreibt, wie neue Verbindungen erzeugt werden, wenn Knoten, die zum selben sekundären Netzwerk gehören sollten, entdeckt werden. Zwei Teile können hier unterschieden werden:
Entdeckung von Paaren von Knoten, die in das gleiche sekundäre Netzwerk gehören sollten. Das Kriterium zum Gruppieren von Knoten in dasselbe Netzwerk ist anwendungsspezifisch. In dem Beispiel der Webseitenanmerkung sollten alle Knoten, die Kommentare über dieselbe URL repräsentieren, zusammen in einem sekundären Netzwerk gruppiert werden. Wie Knoten entdeckt werden, die zusammen gruppiert werden sollen, ist ebenfalls anwendungsspezifisch. Ein Beispiel wird in Kürze gegeben.

[0050] Aktualisierung/Erweiterung des sekundären Netzwerks als ein Resultat einer Knotenentdeckung. Wenn ein Paar von Knoten entdeckt wird, die zu demselben sekundären Netzwerk gehören sollten, kann das System infolgedessen eine oder mehrere neue Verbindung(en) bauen. Die neue Verbindung ist nicht notwendigerweise zwischen dem Paar von Knoten, sondern kann zum Beispiel zwischen Knoten sein, mit denen diese zwei Knoten verbunden sind. Wie neue Verbindungen erzeugt werden, wird im Detail später beschrieben.

Anfängliche Benachrichtigungs-Nachricht

[0051] Die Organisation des sekundären Netzwerks setzt die Existenz von ankommenden „Benachrichtigen"-Nachrichten voraus, die zum Beispiel ein existierendes und ein neues Mitglied der Gruppe identifizieren können (obwohl möglich ist, dass anfangs noch kein Knoten Teil der Gruppe ist, während später in dem Selbstorganisationsprozess beide Knoten bereits Teil der Gruppe sein können). Es gehört zu einem anderen Teil des Systems, das sekundäre Netzwerk über Knoten zu informieren, die zu ihm gehören sollten. Es gibt verschiedene Wege, auf die dies getan werden kann. Hier geben wir ein Beispiel, wie dies getan wird, wenn das sekundäre Netzwerk verwendet wird in Kombination mit einem primären Netzwerk des Typs, der in unserer früheren internationalen Patentanmeldung beschrieben wird. In dem Beispiel der Webseitenanmerkung veröffentlicht sich jeder Kommentar als ein Knoten in dem primären Netzwerk unter einem Label basierend auf der URL der entsprechenden Webseite. Auf diese Weise kann das primäre Netzwerk verwendet werden, um einen Kommentar für eine gegebene URL zu suchen, wenn einer existiert. Um alle Kommentare für eine gegebene URL zu zeigen, hat jeder Kommentar auch einen Knoten des sekundären Netzwerks, das zu ihm gehört. Knoten, die Kommentaren über dieselbe URL entsprechen, selbstorganisieren sich in ein sekundäres Netzwerk, das spezifisch ist für diese URL. Auf diese Weise kann, sobald das primäre Netzwerk verwendet wird, um einen einzelnen Kommentar über eine URL zu finden, das sekundäre Netzwerk verwendet werden, um andere Kommentare über die selbe URL zu finden.

[0052] Somit werden in diesem Fall Knoten des sekundären Netzwerks, die zusammen gruppiert werden sollten, jeweils unter demselben Label in dem primären Netzwerk veröffentlicht. Ein Mechanismus, wodurch in dem primären Netzwerk Knoten regelmä-

ßig eine „Verschieben bzw. Push"-Aktualisierung durchführen, um Verbindungen aufzubauen und zu unterhalten, werden unten beschrieben, einschließlich eine Modifizierung, so dass, wenn ein Knoten einen anderen Knoten erkennt, der unter dem selben Label veröffentlicht wird, die erforderliche Benachrichtigungs-Nachricht erzeugt wird.

Handhabung von Benachrichtigungs-Nachrichten

[0053] Wenn ein Knoten eine Benachrichtigungs-Nachricht über einen Knoten empfängst, mit dem er noch nicht in Verbindung steht, geschieht eines des folgenden:
Wenn der empfangende Knoten bereits die maximale Anzahl von erlaubten Verbindungen hat, fügt er ihn stattdessen als Reserve hinzu (es sei denn, er hat ihn bereits als Reserve). Wenn dadurch der Knoten seine maximale Anzahl an Reserven übersteigen würde, entfernt er eine Reserve. Er kann dann auch die Benachrichtigungs-Nachricht an die Reserve weiterleiten, die er entfernt hat. Ob er dies tut oder nicht, hängt von dem Wert des Benachrichtigungspegels ab. Der Benachrichtigungspegel wird jedes Mal verringert, um zu verhindern, dass sich Nachrichten endlos ausbreiten.

[0054] Andernfalls, wenn der betreffende Knoten noch nicht die maximale Anzahl der Verbindungen hat, versucht der empfangende Knoten, eine gegenseitige Verbindung zwischen beiden Knoten zu erzeugen. Dies wird dargestellt in Fig. 2, in den Diagrammen a und b. Hier ist L = 3 und Knoten 1 hat eine Benachrichtigungs-Nachricht über Knoten 2 empfangen. Da beide Knoten nur zwei Verbindungen hatten, wird eine Verbindung zwischen Knoten 1 und Knoten 2 erzeugt.

[0055] Andernfalls, wenn der betreffende Knoten bereits die maximale Anzahl der Verbindungen hat, ist es nicht möglich, einfach eine gegenseitige Verbindung zwischen beiden Knoten zu erzeugen. So was geschieht ist, dass der empfangende Knoten versucht, sich in eine vorhandene Verbindung einzufügen. Dies wird dargestellt in Fig. 2, in den Diagrammen c und d. Hier ist die Verbindung zwischen Knoten 2 und Knoten 3 unterbrochen, aber sie wird ersetzt durch zwei neue Verbindungen: eine Verbindung zwischen Knoten 1 und Knoten 2 und eine Verbindung zwischen Knoten 1 und Knoten 3. So wird die Gesamtzahl der Verbindungen um eins erhöht. Es funktioniert, obwohl Knoten 2 und Knoten 3 bereits die maximale Anzahl der Verbindungen hatten. Jedoch muss der Knoten 1 zwei neue Verbindungen erzeugen können, damit dies erfolgreich ist. Der Prozess wird detaillierter in den Ablaufdiagrammen von Fig. 3 bis Fig. 9 erläutert.

[0056] Fig. 3 zeigt, wie ein Knoten ankommende Benachrichtigungs-Nachrichten handhabt. Hier wird entschieden, ob eine neue Verbindung erzeugt werden soll, und wenn, dann wie (durch Hinzufügen einer neuen Verbindung oder durch Änderung einer vorhandenen Verbindung in zwei Verbindungen). Wenn keine neuen Verbindungen erzeugt werden, kann der Satz von Reserven aktualisiert werden und eine weitere Benachrichtigungs-Nachricht kann gesendet werden.

[0057] In Schritt **300** wird eine Benachrichtigungs-Nachricht empfangen, welche die Adresse des Knotens enthält, die sie gesendet hat (Sender), die Adresse des betreffenden Knotens (subject node) und einen Ausbreitungsgrenzwert, notifylevel. Der empfangende Knoten prüft zuerst (**301**), ob er Raum zum Aufbau einer neuen Verbindung hat und wenn ja, ob (**302**) er bereits eine Verbindung zu dem betreffende Knoten hat. Wenn nicht, versucht er eine Verbindung mit dem Betreffenden (subject) aufzubauen.

[0058] In Schritt **303** sendet er eine LinksQuery-Nachricht an den betreffenden Knoten (subject node) und in **304** erwartet er eine Antwort. Sobald die Antwort – eine Verbindungen-Nachricht – empfangen wird, prüft er wieder (**305**), ob er noch Raum hat, um eine neue Verbindung aufzubauen (falls er mittlerweile andere Nachrichten empfangen und gehandhabt hat und Verbindungen erzeugt hat als ein Ergebnis). Wenn ja, überprüft (**306**) er dann die empfange Verbindungen-Nachricht, um zu prüfen, ob der betreffende Knoten den Raum hat zum Aufbau einer neuen Verbindung. Wenn ja, dann fügt in Schritt **307** und **308** der empfangende Knoten die Adresse des betreffenden Knotens zu seiner Liste von Verbindungen hinzu (aber als "unbestätigt" markiert) und sendet eine AddLink-Nachricht an den betreffenden Knoten.

[0059] Wenn jedoch in Schritt **306** bestimmt wird, dass der betreffende Knoten keine weiteren Verbindungen akzeptieren kann, versucht der empfangende Knoten, sich selbst in eine existierende Verbindung einzufügen, wie unter Bezugnahme auf Fig. 2 oben erwähnt. Der erste Schritt (**309**) ist, zu prüfen, ob der empfangende Knoten Raum für zwei Verbindungen hat; wenn nicht, wird das Verfahren beendet.

[0060] Wenn er jedoch Raum hat, dann wählt der empfangende Knoten eine Verbindung zufällig aus der Liste der Verbindungen in der empfangenen Verbindungen-Nachricht (aber keinen Knoten, zu dem der empfangende Knoten bereits eine Verbindung hat), das heißt, eine Verbindung zwischen dem betreffenden Knoten (subject node) und einem anderen Knoten, die hier als anderer (other) bezeichnet wird. Der empfangende Knoten versucht dann, sich selbst in diese Verbindung einzufügen durch:
311 Hinzufügen der Adresse des betreffenden (subject) Knotens (unbestätigt) zu seiner Liste von Verbindungen;
312 Hinzufügen der Adresse des anderen (other)

Knotens (unbestätigt) zu seiner Liste von Verbindungen;
313 Senden an den betreffenden Knoten eine ChangeLink-Nachricht, die die Adresse des anderen enthält;
314 Senden an den anderen Knoten eine ChangeLink-Nachricht, welche die Adresse des Betreffenden enthält.

[0061] Wird jedoch angenommen, dass in Schritt **301** bestimmt wird, dass der empfangende Knoten keinen Raum hat, um eine Verbindung hinzuzufügen, oder dass in Schritt **302** er bereits eine Verbindung zu dem betreffenden Knoten hat, dann prüft das Verfahren, ob der empfangende Knoten eine Verbindung zu seiner Liste der Reserve-Verbindungen hinzufügen sollte. In Schritt **315** endet das Verfahren, wenn herausgefunden wird, dass der betreffende Knoten bereits in der Reserven-Liste ist. Bei **316** wird überprüft, ob Raum vorhanden ist, um eine Verbindung zu der Reserven-Liste hinzuzufügen, und wenn dem so ist, wird sie bei **317** hinzugefügt. Wenn nicht, dann wird eine existierende der Reserve-Verbindungen zufällig gewählt bei **318**, und in Schritt **319** entfernt, so dass sie ersetzt werden kann durch eine Verbindung zu dem Betreffenden in Schritt **317**. Auch wird die Variable notifylevel dekrementiert bei **320** und wenn (Schritt **321**) sie ungleich Null bleibt, wird die ursprüngliche Benachrichtigungs-Nachricht – mit diesem neuen Wert von notifylevel – in Schritt **322** weitergeleitet an den Knoten (als Ersatz (replace) bezeichnet), geleitet durch die zufällig ausgewählte existierende Verbindung.

[0062] Der Effekt dieses Verfahrens ist, dass, wenn ein Knoten A, der bereits einen vollen Satz von Verbindungen hat, eine Benachrichtigungs-Nachricht empfängt, die ihn anweist, zu einem betreffenden Knoten B eine Verbindung aufzubauen, die Adresse von B aufgezeichnet wird als eine Reserve-Verbindung. Diese Verbindung bleibt schlafend, bis die Liste der Reserve-Verbindungen von A voll ist. Dann, wenn A eine spätere Benachrichtigungs-Nachricht empfängt, die ihn anweist, eine Verbindung zu dem Knoten C aufzubauen, und die Reserve-Verbindung zu dem Knoten B wird in Schritt **318** gewählt, ist die neue Benachrichtigungs-Nachricht, die in Schritt **322** erzeugt wird, tatsächlich eine Anforderung zu Knoten B, eine Verbindung von sich selbst zu dem Knoten C zu erzeugen.

[0063] Es wird auch ein Mechanismus vorgesehen – aber nicht gezeigt auf dem Ablaufdiagramm – durch den, wenn eine Verbindung unbestätigt ist und der empfangende Knoten keine Bestätigung (über eine LinkAdded-Nachricht, wie unten beschrieben mit Bezug auf Fig. 6) in einem gegebenen Zeitraum empfängt, die unbestätigte Verbindung gelöscht wird. Es ist anzumerken, dass, wenn der empfangende Knoten Verbindungen hat, die noch einen „unbestätigten" Status haben, er diese unbestätigten Verbindungen zurückgibt (sowie selbstverständlich die bestätigten) als Antwort auf LinksQuery-Nachrichten, was anderen Knoten ermöglicht, zu bestätigen, dass er versucht, die Verbindung aufzubauen.

[0064] In der Fig. 3 führen die "nein"-Ausgänge der Schritte **305** und **309** zur Beendigung des Verfahrens: jedoch, wenn gewünscht, können sie an das "Reserve-Verbindung"-Verfahren geleitet werden, das in Schritt **315** beginnt, mit einer leichten Verbesserung der Leistungsfähigkeit.

[0065] In den Schritten **309** bis **314** unterbricht der Knoten effektiv eine der Verbindungen des Betreffenden und fügt sich selbst dazwischen ein. Eine andere mögliche Option, nicht im Ablaufdiagramm gezeigt, ist für den Knoten eine seiner eigenen Verbindungen zu unterbrechen (selbstverständlich unter der Annahme, dass er bereits mindestens eine Verbindung hat) und den Betreffenden dazwischen einfügt. Diese Option, wenn implementiert, würde sofort nach dem „nein"-Ausgang von Schritt **301** versucht. Zuerst muss der empfangende Knoten prüfen, ob der Betreffende weniger als L-1 Verbindungen hat, zufällig eine seiner eigenen Verbindungen wählen (zu einem anderem Knoten), diese ersetzen mit einer unbestätigten Verbindung zu dem Betreffenden, und eine Add-Link-Nachricht an den Betreffenden senden. Um eine bidirektionale Verbindung zwischen dem Betreffenden (subject) und dem Anderen (other) herzustellen, sendet er (a) an den Betreffenden eine spezielle Add-Link-Nachricht, die von dem Betreffenden erfordert den Anderen als eine unbestätigte Verbindung zu seiner Liste von Verbindungen bedingungslos hinzuzufügen und (b) sendet er an den Anderen eine spezielle ChangeLink-Nachricht mit dem empfangende Knoten als der alten zu entfernenden Verbindung und benennt den Betreffenden als die neue hinzuzufügende Verbindung. Diese Option kann zusätzlich oder anstelle der Schritte **309** bis **314** enthalten sein.

[0066] Eine andere Option, damit der empfangende Knoten eine seiner eigenen Verbindungen unterbrechen kann, muss er (nachdem zuerst verifiziert wurde, dass der Betreffende weniger als L-1Verbindungen hat) an den Betreffenden eine Benachrichtigungs-Nachricht senden, die ihn selbst als den Betreffenden bezeichnet. Dieses hätte dieselben Ergebnisse, würde aber einen etwas größeren Messaging-Overhead mit sich bringen.

[0067] Fig. 4 zeigt, wie ein Knoten ankommende ChangeLink-Nachrichten handhabt. Diese Nachrichten werden gesendet, wenn ein Knoten X, der eine Benachrichtigungs-Nachricht empfangen hat, eine existierende Verbindung in zwei neue andern möchte (sehen Fig. 2). Der empfangende Knoten Y empfangt bei **400** eine Benachrichtigungs-Nachricht mit Knoten Z als den Betreffenden (subject), das heißt,

Knoten Y wird aufgefordert, seine existierende Verbindung zu Knoten Z mit einer zu Knoten X zu ersetzen. Wenn er bereits eine Verbindung zu X hat, unternimmt er nichts weiter (**401**), während, wenn (**402**) er tatsächlich keine Verbindung zu Knoten Z besitzt, sendet er **403** eine Fehlermeldung an den Sender, X.

[0068] Unter der Annahme, dass alles OK ist, sendet er (**404**) eine Links-Query-Nachricht an den Sender X und erwartet (**405**) eine Verbindungen-Nachricht als Antwort von dem sendenden Knoten X, um zu prüfen, dass letzterer in der Tat die zwei neuen Verbindungen erzeugt hat, die er vor Änderung der betreffenden (subject) Verbindung erzeugt haben sollte. Wenn diese Überprüfungen (**406**, **407**) erfolgreich sind, entfernt der empfangende Knoten seine Verbindung zu Z (**408**), fügt X als eine bestätigte Verbindung hinzu (**409**) und sendet eine LinkAdded-Nachricht an den Sender X zurück (**410**).

[0069] Fig. 5 zeigt, wie ein Knoten ankommende AddLink-Nachrichten handhabt. Diese Nachrichten werden gesendet, wenn ein Knoten wünscht, eine neue Verbindung mit einem Knoten zu erzeugen (siehe Fig. 1). Wenn die Nachricht bei **501** empfangen wird, prüft der Knoten in Schritt **502**, ob er Raum für eine weitere Verbindung hat und wenn nicht, sendet er eine Fehlermeldung bei **503** zurück. Andernfalls sendet er (**504**) eine LinksQuery-Nachricht an den Sender und erwartet (**505**) eine Verbindungen-Nachricht als Antwort von dem sendenden Knoten, damit er bei **506** prüfen kann, dass letzterer in der Tat eine Verbindung zu dem empfangenden Knoten erzeugt hat. Wenn nicht, weigert er sich, die Verbindung hinzuzufügen und endet, aber wenn ja, fügt er dann den Sender als eine bestätigte Verbindung hinzu (**507**) und sendet eine LinkAdded-Nachricht an den Sender zurück über eine Bestätigung (**508**).

[0070] Fig. 6 zeigt, wie ein Knoten ankommende LinkAdded-Nachrichten handhabt. Diese Nachrichten werden gesendet, wenn ein anderer Knoten eine Verbindung zu dem empfangende Knoten akzeptiert hat, entweder als Antwort auf ein ChangeLink- oder eine AddLink-Nachricht. Wenn die LinkAdded-Nachricht empfangen wird bei **600**, was anzeigt, dass eine Verbindung akzeptiert wurde, wird sein Status zu "bestätigt" in Schritt **601** geändert. Die Verbindung wird dann unterhalten, bis sie entweder geändert wird für eine neue Verbindung (als Antwort auf eine ChangeLink-Nachricht), oder die Verbindung unterbrochen wird.

[0071] Fig. 7 zeigt, wie ein Knoten ankommende LinkError-Nachrichten handhabt. Diese Nachrichten werden gesendet, wenn entweder ein Knoten nicht imstande ist, eine Verbindung zu dem empfangenden Knoten zu erzeugen, nachdem letzterer eine gegenseitige Verbindung angefordert hat (über eine ChangeLink- oder AddLink-Nachricht), oder eine Verbindung unterbrochen zu sein scheint (der Knoten an dem anderen Ende kann nicht auf Nachrichten antworten, oder die Verbindung kann nicht gegenseitig sein). Unterbrochene Verbindungen werden nicht erfasst durch den Selbstorganisations-Prozess, sondern wenn Clients das sekundäre Netzwerk durchqueren (wie später erläutert wird).

[0072] Nach dem Empfang der Nachricht bei **700** wird bestimmt (**701**), ob die Nachricht über einen Knoten ist, zu dem der empfangende Knoten eine unbestätigte Verbindung hat. Wenn dem so ist und (**702**) er einen Fehlercode trägt, der ein Fehlschlagen anzeigt, eine angeforderte Verbindung zu erzeugen, dann wird die Verbindung bei **703** entfernt. Wenn jedoch die Nachricht nicht über einen Knoten ist, zu dem der empfangende Knoten eine unbestätigte Verbindung hat, sendet (**704**) der empfangende Knoten eine LinksQuery-Nachricht an den Betreffenden (subject), erwartet (**705**) eine Verbindungen-Nachricht als Antwort, prüft die Antwort bei **706**, um zu prüfen, ob der Betreffende eine Verbindung zu ihm hat, und wenn nicht, entfernt dann in Schritt **703** seine Verbindung zu dem betreffenden Knoten.

[0073] Fig. 8 zeigt, wie ein Knoten ankommende LinksQuery-Nachrichten handhabt. Diese Nachrichten werden gesendet, wenn ein anderer Knoten wünscht, die Verbindungen des empfangende Knotens zu kennen und letzterer bei deren Empfang bei **800** folglich bei **801** mit einer Verbindungen-Nachricht antwortet.

[0074] Fig. 9 zeigt, wie ein Knoten ankommende Verbindungen-Nachrichten handhabt. Wie sie gehandhabt werden, hängt völlig davon ab, warum die entsprechende LinksQuery-Nachricht gesendet wurde. Dieses geschieht aus verschiedenen Gründen, wie unter anderem gezeigt wird in Fig. 3, in Fig. 4, in Fig. 5 und in Fig. 7. Wenn eine LinksQuery-Nachricht gesendet wird, wird ihre eine Referenz gegeben, die lokal eindeutig ist, und eine Nachrichten-Handhabungsvorrichtung wird der Referenz zugeordnet. Dann, wenn eine Verbindungen-Nachricht empfangen wird (**900**), wird die geeignete Nachrichten-Handhabungsvorrichtung identifiziert und die Nachricht wird in Schritt **902** an die geeignete Nachrichten-Handhabungsvorrichtung weitergeleitet, so dass die Nachricht auf die geeignete Weise behandelt wird.

[0075] Es kann selbstverständlich geschehen, dass keine Verbindungen-Nachricht überhaupt empfangen wird als Antwort auf eine LinksQuery, zum Beispiel da der empfangende Knoten abgeschaltet wurde. Folglich, wenn nach einer gegebenen Periode keine Verbindungen-Nachricht empfangen wurde, wird die entsprechende Nachrichten-Handhabungseinrichtung entfernt. Obgleich dies nicht ausdrücklich gezeigt wird in einem der Ablaufdiagramme, die hier disku-

tiert werden, bedeutet es einfach, dass, wenn eine Verbindungen-Abfrage abläuft, keine weitere Aktion unternommen wird und das gesamte Ablaufdiagramm beendet ist.

Abrufen von Knoten

[0076] Mit der Adresse eines einzelnen Knotens des sekundären Netzwerks ist es möglich, andere, möglicherweise alle, Knoten in dem Netzwerk zu erfassen. Wie dies getan werden kann, ist sehr einfach. Man sendet an den bekannten Knoten eine LinksQuery-Nachricht zur Anforderung aller seiner Verbindungen. Der Knoten antwortet mit einer Verbindungen-Nachricht, welche die Adresse aller Knoten enthält, mit denen er Verbindungen unterhält. Dann kann wiederum jeder dieser Knoten kontaktiert werden, ihre Verbindungen angefordert werden und so die Adressen aller ihrer Verbindungen erlangt werden. Auf diese Weise kann das Netzwerk durchquert werden und alle Knoten erfasst werden, die es enthält.

[0077] Fig. 10 zeigt das Verfahren detaillierter. Es sollte angemerkt werden, dass dies das Verfahren ist, das in dem Abrufen-Schritt **35** verwendet wird, der in der Fig. 1A gezeigt wird. Die Adressen aller bekannten Knoten, die erfolgreich kontaktiert wurde, werden in die "bestätigte" Liste eingegeben. Daten können gleichzeitig abgerufen werden. In dem Fall des Beispiels des "Webseiten-Kommentars" ist das relevante Datenelement die Adresse des Kommentars, und diese wird auch in die bestätigte Liste eingegeben zusammen mit der Knoten-Adresse. Die bestätigte Liste liefert dann die Adressen, die für den „Kommentar abrufen"-Schritt (**36**) in der Fig. 1A erforderlich ist. Die "unbestätigte" Liste enthält andererseits die Adressen der bekannten Knoten, die noch nicht kontaktiert wurden. Schließlich enthält die "bekannte" Liste die Adressen aller bekannten Knoten. Sie umfasst alle Adressen in der „bestätigten" und „unbestätigten" Liste, aber auch die Adressen der Knoten, die kontaktiert wurden und die nicht geantwortet haben. Die bekannte Liste hat auch für jede Adresse, die in sie eingegeben wurde, ein zusätzliches Feld zur Aufnahme einer Quell-Adresse – das heißt, die Adresse des Knotens, aus dessen Liste die Adresse, zu welcher der aktuelle Zeiger zeigt, erlangt wurde, für Fehler-Bericht-Zwecke.

[0078] Es ist nicht maßgeblich, wo das Abrufen-Verfahren stattfindet: an einem Knoten oder sonst irgendwo. In Schritt **1000** wird eine Anforderung, Knoten-Adressen abzurufen, empfangen zusammen mit einer Startadresse, das heißt, die Adresse eines Knotens, von dem festgestellt wurde, dass er zu dem fraglichen virtuellen Netzwerk gehört. In Schritt **1002** wird ein Adresszeiger, aktuell (current), anfangs auf diese Adresse gesetzt, während ein zweiter Adresszeiger, Quelle (source), anfangs Null ist (**1003**).

[0079] In den Schritten **1004** und **1005** wird eine LinksQuery-Nachricht gesendet an die Adresse, die durch aktuell (current) gegeben wird, und auf eine Antwort gewartet. Wenn eine Verbindungen-Nachricht empfangen wird, wird aktuell (current) zu der bestätigten (confirmed) Liste hinzugefügt (Schritt **1006**), zusammen mit der Kommentar-Adresse von der Verbindungen-Nachricht.

[0080] In Schritt **1007** wird ein Teilverfahren eingegangen, das durchgeführt wird für jede der Adressen, die in der Verbindungen-Nachricht enthalten sind. Wenn (**1008**) die Adresse bereits in der bekannten Liste ist, geht das Verfahren weiter zu der nächsten Adresse. Andernfalls wird die Adresse hinzugefügt zu der bekannten Liste und zu der unbestätigten Liste (Schritte **1009**, **1010**). Auch wird (**1011**) die Adresse in aktuell (current) in die bekannte Liste eingegeben als die Quelle der hinzugefügten Adresse.

[0081] Sobald dieses Teilverfahren abgeschlossen ist, dann wird (es sei denn, die unbestätigte Liste ist leer, in diesem Fall endet das Verfahren in Schritt **1012**) in Schritt **1013** eine Adresse zufällig gewählt aus der unbestätigten Liste. Diese Adresse wird die neue aktuelle Adresse und wird aus der unbestätigten Liste gelöscht. Der nächste Schritt (**1014**) ist, aktuell (current) in der bekannten Liste zu suchen, um die dazugehörige Quellenadresse abzurufen, und diese in den Quelle-Zeiger einzugeben. Die zufällige Auswahl ist nicht vorgeschrieben. Zum Beispiel kann aktuell (current) gewählt werden als der "älteste" Knoten in der unbestätigten Liste, oder die Liste kann mit einem andern Kriterium sortiert werden (zum Beispiel Adressen des Knotens) und aktuell (current) kann immer der „erste" Knoten in dieser Liste sein. Jedoch hat eine zufällige Wahl von aktuell (current) ihre Vorteile. Sie verteilt die Belastung in dem System (insbesondere, wenn nicht alle Knoten immer abgerufen werden) und verteilt auch das Testen der Verbindungen des Netzwerks, so dass unterbrochene Verbindungen schneller erfasst werden.

[0082] Das Verfahren fährt dann wieder von Schritt **1004** fort und iteriert, bis die unbestätigte Liste leer ist – das heißt, keine weiteren neuen Adressen können gefunden werden.

[0083] Ein Nebeneffekt des Abrufen-Verfahrens ist, dass es unterbrochene Verbindungen entdeckt. Zum Beispiel kann geschehen, dass ein Knoten nicht antwortet oder dass eine Verbindung nicht gegenseitig ist. Letzteres ist der Fall, wenn ein Knoten A eine Verbindung zu Knoten B hat, aber der Knoten B den Knoten A nicht in seiner Verbindungstabelle hat. Wenn eine unterbrochene Verbindung entdeckt wird, wird der Knoten, der die "Quelle" der Verbindung ist, benachrichtigt über eine LinkError-Nachricht. Wie Fig. 7 bereits gezeigt hat, kann dann der Quelle-Knoten die Verbindung selbst prüfen (um die Richtigkeit

des Fehlerberichts zu bestätigen) und kann als Ergebnis die Verbindung Entfernen. Ein Knoten, der nicht antwortet, wird erkannt durch den Ausfall in Schritt **1005**, eine Verbindungen-Nachricht in einer gesetzten Timeout-Periode zu empfangen, und in Schritt **1015** wird eine Fehlermeldung, welche die Adresse von aktuell (current) und einen "keine Antwort"-Fehlercode enthält, an die Quelle gesendet, worauf die Steuerung zurückkehrt zu Schritt **1012**. Die nicht-Gegenseitigkeit einer Verbindung wird erkannt durch Testen in Schritt **1016**, um zu bestimmen, ob die Verbindungen-Nachricht, die für aktuell (current) empfangen wird, die Adresse der Quelle enthält: wenn nicht, wird eine Fehlermeldung, welche die Adresse von aktuell (current) und einen "nicht-gegenseitig"-Fehlercodes enthält, an die Quelle gesendet (Schritt **1017**), aber das Abrufen-Verfahren geht weiter wie vorher, da es die Verantwortung des Quelle-Knotens ist, Abhilfeaktion zu unternehmen (in Übereinstimmung mit dem Verfahren von Fig. 7). Der Test in Schritt **1016** wird übersprungen, wenn die Quelle Null ist.

[0084] Es ist anzumerken, dass, obwohl mehrere bestätigte Knoten mit einem Knoten verbunden sein können, der nicht auf eine Verbindungen-Nachricht antwortet, nur der Knoten, der zuerst zu der Verbindung beigetragen hat (der Quelle-Knoten) benachrichtigt wird, dass es "keine Antwort" gegeben hat. Dies ist teilweise deswegen, da es das Ablaufdiagramm einfacher zu verstehen macht. Jedoch kann argumentiert werden, dass es einen anderen praktischen Vorteil gibt. Es kann der Fall sein, dass ein Knoten nicht (rechtzeitig) antwortet, da er vorübergehend überlastet ist. In diesem Fall ist es nicht erwünscht, dass mehrere Knoten an ihn gleichzeitig eine LinksQuery-Nachricht senden zum Testen, ob es einen Fehler gibt (wie in Fig. 7). So oder so ist, wenn gewünscht, es einfach, den Knoten-Abruf-Algorithmus zu aktualisieren, um alle bekannten Knoten zu benachrichtigen, die durch eine unterbrochene Verbindung beeinträchtigt sind, wenn eine derartige Verbindung entdeckt wird.

[0085] In Fig. 10 endet der Knoten-Abruf nicht, bis alle bekannten Knoten kontaktiert wurden. In der Praxis kann es wünschenswert sein, das Verfahren früher zu beenden. Zum Beispiel, wenn ein Benutzer nach einer Stelle sucht, von der er eine Datei herunterladen kann, kann es ausreichend sein, ihm oder ihr die Wahl von zehn möglichen Download-Adressen anzubieten, anstelle von beispielsweise allen Tausend.

[0086] Der Algorithmus in Fig. 10 wird gezeigt als vollständig seriell. Nur ein Knoten wird jeweils kontaktiert. Eine weitere LinksQuery-Nachricht wird nur gesendet, nachdem eine Antwort empfangen wurde auf die vorherige (oder die Zeit abgelaufen ist). In der Praxis gleichwohl bevorzugen wir, das Abrufen zu beschleunigen durch Ausgabe von mehreren LinksQuery-Nachrichten gleichzeitig. Es kann auch der Fall sein, dass an der Box **1000** mehrere Abruf-Anforderungen gleichzeitig gehandhabt werden durch mehrere Instanzen des Verfahrens von Fig. 10.

DISKUSSION

Erfolg einer Selbstorganisation

[0087] Das Ziel des sekundären virtuellen Netzwerks ist, alle Knoten zu selbstorganisieren, die zusammen gruppiert werden sollen in ein einzelnes Netzwerk, im Vergleich zu mehreren unverbundenen Netzwerken. Ob dies der Fall ist oder nicht, hängt zum großen Teil davon ab, wie die anfängliche Benachrichtigungs-Nachricht erzeugt wird. Wenn es zum Beispiel eine Gruppe mit zwölf Knoten gibt, die alle zusammen gruppiert werden sollen, aber von dieser Gruppe fünf Knoten nur Benachrichtigungen über andere Knoten in dieser Gruppe von fünf empfangen, und keiner der anderen sieben Knoten wird benachrichtigt über einen dieser fünf Knoten, ist es unmöglich für die Knoten, sich in ein einzelnes Netzwerk selbst zu organisieren. Stattdessen ordnen sie sich in zwei getrennten Netzwerken an, eines aus fünf Knoten und eines aus sieben Knoten. Solange jedoch die ursprünglichen Benachrichtigungen nicht derart sind, dass es unmöglich ist für Knoten, sich in ein einzelnes Netzwerk selbstzuorganisieren, ist das Selbstorganisations-Verfahren derart, dass es sehr unwahrscheinlich ist, dass sich Knoten nicht in ein einzelnes Netzwerk selbstorganisieren. Eine Berechnung der Wahrscheinlichkeit, dass die Selbstorganisation zu einem einzelnen Netzwerk führt, ist kompliziert und hängt von dem Mechanismus ab, durch den die anfänglichen Benachrichtigungen erzeugt werden. Jedoch haben wird in Simulationen experimentiert mit mehreren verschiedenen anfänglichen Benachrichtigungs-Mechanismus und bis jetzt ist es den Knoten nie misslungen, sich in ein einzelnes Netzwerk selbstzuorganisieren.

Robustheit gegenüber bösartigen Knoten

[0088] Bis jetzt wurde angenommen, dass alle Knoten das Protokoll befolgen. Jedoch ist möglich, dass es bösartige Knoten gibt, die sich nicht an die Regeln halten. Sie können versuchen, Verbindungen zu unterbrechen, die durch andere Knoten unterhalten werden und/oder versuchen, zu viele Verbindungen zu sich selbst zu erlangen. Es ist wünschenswert, dass das gesamte System so robust wie möglich gegenüber solchem Missbrauch ist.

[0089] Das bisher beschriebene System ist bereits ziemlich robust gegenüber bösartigen Knoten. Das ist, da jeder Knoten immer mit einem LinksQuery-Verbindungen-Nachricht-Austausch die Verbindungen prüft, die durch den anderen relevanten Kno-

ten unterhalten werden, bevor er seine eigenen Verbindungen ändert. Zum Beispiel, wenn ein Knoten eine AddLink-Nachricht empfängt (siehe Fig. 3), prüft er zuerst, dass der sendende Knoten in der Tat mit ihm verbunden ist, bevor er den Sender als seine eigene Verbindung hinzufügt.

[0090] Jedoch hat das System noch eine relative Schwäche. Soweit können Knoten einfach „lügen", wenn sie mit einer Verbindungen-Nachricht antworten. Häufig sendet ein Knoten eine LinksQuery-Nachricht, um zu prüfen, dass der empfangende Knoten eine Verbindung zu ihm hat. Mit diesem Wissen kann der empfangende Knoten mit einer gefälschten Verbindungen-Nachricht antworten, die derart modifiziert ist, dass sie immer den Sender der Verbindungen-Nachricht als eine Verbindung enthält. Dies ermöglicht einem Knoten, viel mehr als die erlaubte Anzahl von L Knoten zu haben, die mit ihm in Verbindung stehen. Dies würde infolgedessen die gesamte Anzahl von „guten" Verbindungen in dem System reduzieren.

[0091] Glücklicherweise gibt es einen Weg, um diese Schwäche zu adressieren. Dies kann getan werden, wenn Knoten ihre LinksQuery durch einen Proxy-Knoten senden. Diese Proxies werden jedes Mal zufällig gewählt, wenn ein Knoten eine Abfrage senden möchte. Jeder Knoten kann zum Beispiel die Knoten als Proxies verwenden, mit denen er momentan verbunden ist. Auf diese Weise ist der Knoten (A), der die Verbindungen eines anderen Knotens (B) wissen möchte, unbekannt für den Knoten B, da die LinksQuery-Nachrichten, die er von einem Proxy-Knoten (C) empfängt, und die Nachricht, die Knoten B von Knoten C empfängt, sich überhaupt nicht auf Knoten A bezieht. Folglich gibt es keinen guten Weg für Knoten B, gefälschte Nachrichten zu senden, die einen signifikanten Effekt auf das gesamte System haben.

[0092] Selbstverständlich gibt es die Frage, was der Effekt von bösartigen Proxies ist. Obgleich offensichtlich bösartige Proxies einen nachteiligen Effekt haben (es ist unvermeidlich, dass Knoten, die das Protokoll nicht befolgen, die Leistung zu einem gewissen Grad beeinflussen), ist dieser Effekt begrenzt. Der Grund ist, dass sie nur die LinksQuery bösartig handhaben können, die sei weiterleiten sollen, und diese Anforderungen sind ungefähr gleichmäßig über alle Knoten verteilt. Andererseits, wenn keine Proxies verwendet werden, können bösartige Knoten Verwüstungen anrichten, wenn sie sehr aktiv ist. Wenn diese Knoten viele falsche AddLink-Nachrichten senden und die vielen Verbindungen-Nachrichten fälschen, die sie danach senden, ist der Effekt auf das gesamte System viel größer.

PRIMÄRES VIRTUELLES NETZWERK

[0093] Das primäre Netzwerk wird im Detail beschrieben in unserer oben angeführten internationalen Patentanmeldung. Hier werden die grundlegenden Abruf- und Selbstorganisations-Mechanismen beschrieben, zusammen mit einer Modifizierung, welche die Erzeugung von Benachrichtigungs-Nachrichten ermöglicht zum Antreiben der Selbstorganisation des sekundären Netzwerks.

[0094] Zuerst ist es notwendig, das Konzept eines virtuellen Koordinaten-Raums zu erläutern, der durch diesen Mechanismus verwendet wird. Es wurde bereits erwähnt, dass jeder Knoten ein Label hat. Das Label wird in Koordinaten in einem virtuellen Raum übersetzt. Der Raum kann ein-, zwei- oder höher-dimensional sein. Der genaue Translationsmechanismus ist nicht sehr entscheidend: für einen eindimensionalen Raum kann das Label, das als eine Binärzahl betrachtet wird, direkt als die Koordinate verwendet werden. Für zwei oder mehr Dimensionen ist das bevorzugte Verfahren, dass das Label, das als ein String von Bits betrachtet wird, in zwei oder mehr gleiche Gruppen unterteilt wird, wobei jede Gruppe, als eine Binärzahl betrachtet, eine der Koordinaten bildet. Jede Koordinate (oder die Koordinate in einem eindimensionalen Raum) wird skaliert in den Bereich [0, 1].

[0095] Der Abstand zwischen zwei Label in diesem virtuellen Raum ist die Euklidische Entfernung zwischen den zwei Koordinatensätzen (obwohl andere Entfernungen, wie der Stadtblock(city block)-Abstand (oft als der Manhattan-Abstand bezeichnet) verwendet werden können, wenn gewünscht). Der Koordinaten-Raum ist verzerrt (wraps), so dass der Abstand in der X-Richtung zwischen x_1 und x_2 ist

$$\text{Min}\{(1 - |x_1 - x_2|), |x_1 - x_2|\}$$

und die Euklidische Entfernung in zwei Dimensionen zwischen den Punkten (x_1, y_1) und (x_2, y_2) folglich ist

$$\sqrt{\{[\text{Min}\{(1 - |x_1 - x_2|), |x_1 - x_2|\}]^2 + [\text{Min}\{(1 - |y_1 - y_2<), |y_1 - y_2|\}]^2\}}.$$

[0096] Wir erinnern uns an diesem Punkt auch, dass jeder Knoten eine Liste **22** (Fig. 1) mit einer Anzahl von Einträgen hat, die Verbindungen zu anderen Knoten darstellen. Jeder Eintrag besteht aus dem Label und der Adresse eines derartigen weiteren Knotens. Anfangs ist diese Liste leer und folglich hat der Knoten eine zweite ähnliche Liste von Start-Verbindungen – das heißt, wenige Verbindungen (typischerweise vier), damit er anfangs fähig ist, andere Knoten des Netzwerks zu kontaktieren. Wie die Verbindungen in der Liste **22** (bezeichnet als Nahbereichs-Verbindungen) kann der Knoten auch zusätzlich solche Listen haben, die hierarchisch angeordnet sind,

und/oder eine Liste von weitreichenden Verbindungen. Diese werden beschrieben in unserer früheren internationalen Patentanmeldung, werden aber, da sie optional sind, hier nicht beschrieben.

Nachrichten

[0097] Zuerst werden die folgenden Nachrichten verwendet (es ist anzumerken, dass die Nachrichten, die in dem primären virtuellen Netzwerk verwendet werden, verschieden sind von und vollständig unabhängig von den Nachrichten, die in dem sekundären virtuellen Netzwerk verwendet werden):
FINDEN-Nachrichten werden verwendet, um Knoten-Suchen zu initiieren und auszuführen und „PULL bzw. HOLEN"-Aktualisierungen zu unterstützen. Sie enthalten:

- das Label eines Zielknotens
- die Adresse des Knotens, der die Abfrage initiierte.

[0098] GEFUNDEN-Nachrichten werden verwendet, um die Ergebnisse von Abfragen zurückzugeben. Sie enthalten:

- das Label des Zielknotens
- das Label des Knotens, der gefunden wurde
- die Adresse des Knotens, der gefunden wurde
- die Adresse des Knotens des sekundären Netzwerks, das zu dem Knoten gehört, der gefunden wurde
- anwendungsspezifische Daten – in diesem Fall die Adresse des Kommentarknotens, der zu dem Knoten gehört, der gefunden wurde.

[0099] PUSH- bzw. VERSCHIEBEN-Nachrichten zeigen das Label eines Knotens den anderen Knoten an. Sie enthalten:

- das Label eines betreffenden (subject) Knotens
- die Adresse des betreffenden Knotens
- die Anzahl von „zu machenden Sprüngen (hops to go)", um einen Zielknoten zu erreichen.

[0100] BENACHRICHTIGUNGS-Nachrichten werden verwendet, um Verschieben-Aktualisierungen zu verbreiten. Sie enthalten:

- das Label eines betreffenden Knotens
- die Adresse des betreffenden Knotens.

Abrufen

[0101] Die Fig. 11 zeigt, wie jeder Knoten ankommende Finden-Nachrichten handhabt. Im Prinzip sucht der empfangende Knoten nach einem Knoten, der näher als er selbst zu dem Zielknoten ist, der in der Finden-Nachricht identifiziert wird, und wenn erfolgreich, leitet er die Finden-Nachricht weiter. Wenn nicht erfolgreich, sendet er seine eigene Adresse und Label zurück. Es tut dies durch Ausführen der folgenden Schritte:

[0102] Schritt **1100**: der Knoten empfangt eine Finden-Nachricht, die das Label eines Zielknotens und die Adresse eines initiierenden Knotens enthält;

[0103] Schritt **1105**: der Knoten übersetzt das Label des Zielknotens in Koordinaten in dem Label-Raum und berechnet, welche der Verbindungen (Knoten, die er aufgezeichnet hat, am nächsten zu dem Zielknoten in dem Label-Raum ist. Der relevante Knoten wird als nächster (nearest) Knoten bezeichnet;

[0104] Schritt **1110**: der Knoten vergleicht den Abstand zwischen seinen eigenen Koordinaten mit dem Abstand zwischen den Koordinaten des nächsten (nearest) Knotens und denen des Zielknotens;

[0105] Schritt **1115**: wenn der Abstand zwischen seinen eigenen Koordinaten und denen des Zielknotens geringer (oder gleich) ist, sendet der Knoten an den initiierenden Knoten über das Netzwerk **10** eine Gefunden-Nachricht, die sein eigenes Label und eigene Adresse enthält;

[0106] Schritt **1120**: wenn der Abstand zwischen den Koordinaten des nächsten Knotens und denen des Zielknotens geringer ist, leitet der Knoten die Finden-Nachricht an den nächsten Knoten weiter.

[0107] Die Adresse des Knotens, die in Schritt **1115** zurückgesendet wird, ist entweder die eines mit dem Ziel-Label oder nahe zu diesem in dem Label-Raum. Wenn das zurückgesendete Label nicht mit dem Ziel-Label übereinstimmt, kann dies bedeuten, dass entweder der Zielknoten nicht existiert oder dass das virtuelle Netzwerk nicht ausreichend selbstorganisiert ist.

Verschieben

[0108] Jeder Knoten initiiert spontan Verschieben- bzw. Push-Aktualisierungen. Zum Beispiel kann jeder Knoten regelmäßig einen Verschieben-Aktualisierungsprozess starten. In einer Verschieben-Aktualisierung sendet ein Knoten eine Verschieben-Nachricht mit seinem eigenen Label und Adresse durch eine zufällige Reihe von Knoten, wobei er eine Begrenzung auf die Länge der Reihe setzt. Der letzte Knoten in der Reihe sendet eine Benachrichtigungs-Nachricht zurück zu dem initiierenden Knoten. Die Fig. 12, Fig. 13 und Fig. 14 zeigen die verschiedenen Teile dieses Prozesses.

[0109] Fig. 12 zeigt, wie ein Knoten eine Verschieben-Aktualisierung unter Verwendung; der folgenden Schritten initiiert:

[0110] Schritt **1200**: der Knoten wählt eine Verbindung zufällig aus seinen Start-Verbindungen aus und gibt die Adresse des Knotens, der durch die gewählte Verbindung identifiziert wird, als eine Weiterlei-

tungs-Adresse für eine nächste Nachricht ein;

[0111] Schritt **1205**: der Knoten gibt eine kleine positive Zufallszahl für das Feld „zu machende Sprünge (hops to go)" in der Verschieben-Nachricht ein;

[0112] Schritt **1210**: der Knoten gibt sein eigenes Label und seine Adresse als die des betreffenden Knotens in die Verschieben-Nachricht ein und sendet die Verschieben-Nachricht an den Knoten an der Weiterleitungs-Adresse unter Verwendung des Netzwerks **10**.

[0113] Die Fig. 13 und Fig. 14 zeigen, wie Nahbereichs-Verbindungen aktualisiert werden. Verschieben-Nachrichten werden zusammen mit Benachrichtigungs-Nachrichten verwendet, um Nahbereichs-Verbindungen zu aktualisieren. Es gibt zwei Phasen bei dieser Aktualisierung. In einer ersten Phase leitet jeder Knoten zufällig die Verschieben-Nachricht weiter, bis der Wert in „zu machende Sprünge" in der Nachricht, wie empfangen, „0" ist. Wenn der Wert in „zu machende Sprünge" „0" ist, startet der empfangende Knoten die zweite Phase der Verschieben-Aktualisierung durch das Senden einer Benachrichtigungs-Nachricht. In der zweiten Phase wird die Benachrichtigungs-Nachricht sukzessive weitergeleitet an Knoten, deren Label stufenweise näher an dem des betreffende Knotens in dem virtuellen Raum sind. Wenn kein Knoten mit einem näheren Label gefunden werden kann, dann werden, wenn notwendig, die Verbindungen für den letzten gefundenen Knoten aktualisiert. Dies ist immer der Fall, wenn es ansonsten unmöglich ist, den gegebenen betreffende Knoten zu finden, zum Beispiel da er noch keine Nahbereichs-Verbindungen aufgebaut hat. Der letzte gefundene Knoten sendet dann auch zusätzliche Benachrichtigungs-Nachrichten an Knoten, die möglicherweise ihre Verbindungssätze genauso verbessern könnten.

[0114] Unter Bezugnahme auf Fig. 13 umfasst die erste Phase einer Verschieben-Aktualisierung, die ankommende Verschieben-Nachrichten handhabt, die folgenden Schritte:

[0115] Schritt **1300**: ein Knoten empfangt eine Verschieben-Nachricht. Die Verschieben-Nachricht enthält das Label und die Adresse eines initiierenden Knotens als der betreffende Knoten und hat einen Wert in dem Feld „zu machende Sprünge";

[0116] Schritt **1305**: der empfangende Knoten wählt eine Verbindung zufällig aus seinen Start-Verbindungen und gibt die Adresse des Knotens, der durch die gewählte Verbindung identifiziert wird, als eine Weiterleitungs-Adresse für eine nächste Nachricht ein;

[0117] Schritte **1310** und **1315**: der empfangende Knoten verringert den Wert in dem Feld "zu machende Sprünge" um 1 und prüft, ob der verringerte für „zu machende Sprünge" weiter größer als Null ist;

[0118] Schritt **1320**: wenn der verringerte Wert noch größer als Null ist, leitet der Knoten die Verschieben-Nachricht an die Weiterleitungs-Adresse weiter, die er eingegeben hat;

[0119] Schritt **1325**: wenn der Wert Null ist, gibt der Knoten stattdessen das Label und die Adresse des initiierenden Knotens (gegeben in der empfangenen Verschieben-Nachricht) als den betreffenden Knoten in einer Benachrichtigungs-Nachricht ein und sendet die Benachrichtigungs-Nachricht an die Weiterleitungs-Adresse, die er eingegeben hat.

[0120] Unter Bezugnahme auf Fig. 14 umfasst die zweite Phase der Handhabung von Verschieben-Aktualisierungen und der Handhabung von Benachrichtigungs-Nachrichten die folgenden Schritte:

[0121] Schritt **1400** ein Knoten empfangt eine Benachrichtigungs-Nachricht, welche das Label und die Adresse eines Knotens als den betreffenden Knoten enthält;

[0122] Schritt **1401**: der empfangende Knoten prüft, ob der Betreffende der Benachrichtigungs-Nachricht das gleiche Label hat wie der empfangende Knoten;

[0123] Schritt **1402**: wenn ja, prüft der empfangende Knoten, ob der Betreffende der Benachrichtigungs-Nachricht die gleiche Adresse wie der empfangende Knoten hat. In diesem Fall unternimmt er nichts weiter.

[0124] Wenn jedoch der Betreffende der Benachrichtigungs-Nachricht einen Knoten mit demselben Label wie der empfangende Knoten hat, aber eine Adresse, die von diesem verschieden ist, dann finden zwei Ereignisse staut. Erstens (Schritt **1403**) sendet der empfangende Knoten an den betreffenden Knoten der ankommenden Benachrichtigungs-Nachricht eine Benachrichtigungs-Nachricht, die den Betreffenden einen zufällig gewählten Knoten aus der eigenen Liste von Nahbereichs-Verbindungen des empfangenden Knotens benennt. Zweitens veranlasst der Schritt **1404** die Erzeugung einer Benachrichtigungs-Nachricht zur Aktion durch das sekundäre Netzwerk. Jedoch kann der empfangende Knoten eine solche Nachricht nicht direkt erzeugen. Im Allgemeinen bevorzugen wir, ein Senden von Nachrichten über das Kommunikationsnetzwerk zwischen verschiedenen virtuellen Netzwerken zu vermeiden, aber das Hauptproblem ist, dass der empfangende Knoten nicht nur die Adresse seines eigenen Knotens des sekundären Netzwerks benötigt, sondern auch die Adresse des Knotens des sekundären Knotens, der zu dem betreffenden Knoten gehört. Der empfangende Knoten hat diese Adresse nicht. Folg-

lich wird ein zweistufiges Verfahren verwendet.

[0125] Zuerst sendet der empfangende Knoten eine bestimmte CrossNotify- bzw. Kreuzbenachrichtigungs-Nachricht an den Knoten des primären Netzwerks, der als der Betreffende in der ankommenden Benachrichtigungs-Nachricht spezifiziert wird. Diese Nachricht enthält:

- eine Sender-Adresse, die auf die Adresse des empfangenden Knotens gesetzt ist (das heißt der Knoten, der die ankommende Nachricht (des primären Netzwerks) empfängt);
- eine Empfänger(oder Ziel)-Adresse, die auf die Adresse gesetzt ist, die in der ankommenden Benachrichtigungs-Nachricht enthalten ist;
- eine Betreffender-Adresse, die auf die Adresse des Knotens des sekundären Netzwerks gesetzt ist, der zu dem empfangenden Knoten gehört.

[0126] Es ist anzumerken, dass die ersten zwei Adressen die Adressen von Knoten auf dem primären Netzwerk sind und die dritte Adresse die Adresse eines Knotens auf dem sekundären Netzwerk ist.

[0127] Zweitens leitet der Knoten des primären Netzwerks, der die CrossNotify-Nachricht tatsächlich empfängt, diese an den zugehörigen Knoten des sekundären Netzwerks weiter. Wenn erforderlich, kann der weiterleitende Knoten die Nachricht in das Format umformatieren, das auf dem sekundären Netzwerk verwendet wird, und die Empfänger-Adresse (des primären Netzwerks) mit der Adresse des zugehörigen Knotens des sekundären Netzwerks ersetzen. Die Nachricht wird dann gehandhabt wie in Fig. 3 gezeigt. Der Grund, warum wir „tatsächlich" sagen, ist, dass wir in der Praxis bevorzugen, dass der Knoten des primären Netzwerks, der die CrossNotify-Nachricht empfängt, an seinen zugehörigen Knoten des sekundären Netzwerks eine einfache lokale Nachricht sendet, welche die Adresse enthält, die in dem Betreffender-Feld der CrossNotify-Nachricht spezifiziert wird. In diesem Fall wird das Verfahren von Fig. 3 modifiziert, um den Schritt der Einstellung von notifylevel auf einen geeigneten Wert zu umfassen.

[0128] Dieses Verfahren wird gezeigt mittels eines Beispiels, unter Bezugnahme auf Fig. 15, wo die Kästchen Knoten darstellen und die Pfeile Nachrichten darstellen. Es wird angenommen, dass ein Knoten P1 des primären Netzwerks in Schritt **1400** von Fig. 14 eine Benachrichtigungs-Nachricht empfangt, die das Label LP2 und die Adresse A_{P2} des Knotens P2 des primären Netzwerks als den Betreffenden (subject) enthält. An dem Knoten P1 wird erkannt (Schritte **1401**, **1402** in Fig. 14), dass der betreffende Knoten das gleiche Label wie P1 hat (das heißt $L_{P1} = L_{P2}$), aber eine andere Adresse ($A_{P1} \neq A_{P2}$). Der Knoten P1 kennt die Adresse AS1 seines sekundären Netzwerkknotens S1 und erzeugt (in Schritt **1404** in Fig. 14) eine CrossNotify-Nachricht mit Sender-Adresse A_{P1}, Empfänger-Adresse A_{P2} und Betreffender-Adresse A_{S1}. Diese Nachricht wird an dem Knoten P2 des primären Netzwerks empfangen und dieser sendet eine lokale Benachrichtigungs-Nachricht, mit der Adresse A_{S1}, an den zugehörigen Knoten S2 des sekundären Netzwerks. Alternativ kann der Knoten S2 des sekundären Netzwerks bei Empfang der lokalen Benachrichtigungs-Nachricht, anstelle die Verbindung selbst zu erzeugen entsprechend dem Verfahren von Fig. 3, eine weitere Benachrichtigungs-Nachricht erzeugen (des sekundären Netzwerks) (gezeigt durch die gestrichelte Linie in Fig. 12), die er an den Knoten S1 sendet, und sich selbst als den Betreffenden bezeichnet. Die Benachrichtigungs-Nachricht wird dann an dem Knoten S1 verarbeitet, der dann das Verfahren von Fig. 3 verwendet. Diese Option umfasst eine zusätzliche Nachricht, hat aber den Vorteil, dass, wenn das Verfahren von Fig. 3 ausgeführt wird, die Benachrichtigungs-Nachricht tatsächlich gesendet wird durch den Knoten, dessen Adresse in dem Betreffender-Feld der Nachricht ist, und der betreffende Knoten wurde folglich inhärent bestätigt als noch existierend.

[0129] Zurück nun zu Fig. 14: Schritt **1405**: der empfangende Knoten übersetzt das Label des betreffenden Knotens in Koordinaten und berechnet, welche der Nahbereichs-Verbindungen, die er aufgezeichnet hat, zu einem Knotenlabel führen, dessen Koordinaten am nächsten sind zu denen des betreffenden Knotens in dem virtuellen Raum. Der relevante Knoten wird als der nächste (nearest) Knoten identifiziert;

[0130] Schritt **1415**: der empfangende Knoten vergleicht den Abstand zwischen seinen eigenen Koordinaten und den Koordinaten für den betreffenden Knoten mit dem Abstand zwischen den Koordinaten für den nächsten Knoten und den Koordinaten für den betreffenden Knoten.

[0131] Wenn, in Schritt **1415**, der Abstand zwischen dem empfangenden Knoten und dem betreffenden Knoten als gleiche oder geringer erfasst wird, fügt der empfangende Knoten das Label und die Adresse des betreffenden Knotens als eine Verbindung zu seinem eigenen Nahbereichs-Verbindungs-Satz hinzu ((Schritt **1420**): dieses Verfahren wird unten weiter diskutiert mit Bezug auf Fig. 16), sendet an den betreffenden Knoten eine Benachrichtigungs-Nachricht, die das Label und die Adresse des empfangenden Knotens enthält (Schritt **1430**), und sendet an den nächsten Knoten eine Benachrichtigungs-Nachricht, welche das Label und die Adresse des betreffenden Knotens enthält (Schritt **1435**).

[0132] Wenn, in Schritt **1415**, der Abstand zwischen dem nächsten Knoten und dem betreffenden Knoten als größer erfasst wird, kehrt der empfangende Kno-

ten zurück zu Schritt **1435** dadurch, dass er an den nächsten Knoten eine Benachrichtigungs-Nachricht sendet, welche das Label und die Adresse des betreffenden Knotens enthält.

[0133] Fig. 16 zeigt im Detail, wie sich ein Knoten verhält, wenn er seine Nahbereichs-Verbindungen aktualisiert. Er fügt die neue Verbindung zu seinen Nahbereichs-Verbindungen hinzu und entfernt alle Nahbereichs-Verbindungen, die durch diese Verbindung verdrängt werden.

[0134] Unter Bezugnahme auf Fig. 16 kann ein Knoten eine neue Verbindung zu seiner Liste von Nahbereichs-Verbindungen hinzufügen müssen, zum Beispiel als ein Resultat von Schritt **1420** in Fig. 14.

[0135] Schritt **1600**: der aktualisierende Knoten (das heißt, ein Knoten, der eine Aktualisierung seines Nahbereichs-Verbindungs-Satzes durchführt) hat das Label und die Adresse eines Knotens für eine neue Verbindung;

[0136] Schritt **1605**: der aktualisierende Knoten identifiziert alle existierenden Verbindungen, die in Bezug zu Knoten sind, die näher zu dem neuen Knoten sind als zu dem aktualisierenden Knoten. Diese identifizierten Verbindungen sind zu ersetzen. Um diese Verbindungen zu identifizieren, berechnet der aktualisierende Knoten für jede existierende Verbindung die Abstände zwischen den Koordinaten für den neuen Knoten und den Koordinaten für die Knoten, die in seinen existierenden Verbindungen spezifiziert werden. Er vergleicht jeden dieser Abstände mit dem Abstand zwischen seinen eigenen Koordinaten und den Koordinaten für den Knoten, der in der jeweiligen existierenden Verbindung spezifiziert wird;

[0137] Schritt **1610**: alle Verbindungen, wo der Abstand in Bezug zu dem neuen Knoten geringer ist als der Abstand in Bezug zu dem aktualisierenden Knoten werden aus den Nahbereichs-Verbindungen entfernt;

[0138] Schritt **1620**: der aktualisierende Knoten fügt eine Verbindung für den neuen Knoten zu seinen Nahbereichs-Verbindungen hinzu.

Patentansprüche

1. Verfahren zum Betreiben eines virtuellen Netzwerks, das eine Vielzahl von Knoten hat, wobei jeder Knoten eine Liste zum Speichern bis zu einer vorgegebenen Anzahl von Adressen anderer derartiger Knoten hat, das aufweist:
(i) Empfangen einer Nachricht, die eine Verbindung zwischen einem ersten Knoten und einem zweiten Knoten anfordert;
(ii) Bestimmen, ob sowohl der erste Knoten als auch der zweite Knoten jeweils eine Anzahl von Adressen in seiner Liste hat, die geringer ist als die vorgegebene Anzahl;
(iii) in dem Fall, dass diese Bedingung erfüllt ist, Einfügen der Adresse des ersten Knotens in die Liste des zweiten Knotens und Einfügen der Adresse des zweiten Knotens in die Liste des ersten Knotens;
(iv) in dem Fall, dass diese Bedingung nicht erfüllt ist, Bestimmen, ob der erste Knoten eine Anzahl von Adressen in seiner Liste hat, die zumindest zwei weniger als die vorgegebene Anzahl ist, und wenn dem so ist
(a) Auswählen von der Liste des zweiten Knotens die Adresse eines dritten Knotens;
(b) Entfernen der Adresse des dritten Knotens von der Liste des zweiten Knotens;
(c) Entfernen der Adresse des zweiten Knotens von der Liste des dritten Knotens; und
(d) Einfügen der Adresse des zweiten Knotens in die Liste des ersten Knotens und Einfügen der Adresse des dritten Knotens in die Liste des ersten Knotens;
(e) Einfügen der Adresse des ersten Knotens in die Liste des zweiten Knotens und Einfügen der Adresse des ersten Knotens in die Liste des dritten Knotens.

2. Verfahren gemäß Anspruch 1, wobei die Nachricht, die eine Verbindung anfordert, an dem ersten Knoten empfangen wird, und wobei in Schritt (iii):
die Adresse des zweiten Knotens in die Liste des ersten Knotens eingefügt wird, begleitet von einer Markierung, die anzeigt, dass sie unbestätigt ist;
eine Nachricht von dem ersten Knoten an den zweiten Knoten gesendet wird, die den zweiten Knoten auffordert, die Adresse des ersten Knotens zu den Verbindungen des zweiten Knotens hinzuzufügen;
an dem zweiten Knoten die Adresse so hinzugefügt wird und eine Bestätigungsnachricht an den ersten Knoten gesendet wird; und an dem ersten Knoten bei Empfang der Bestätigungsnachricht die "unbestätigt"-Markierung entfernt wird.

3. Verfahren gemäß Anspruch 2, wobei ein Knoten, bei Empfang einer Nachricht, die anfordert, dass er zu seiner Liste die Adresse eines spezifizierten Knotens hinzufügt, zuerst eine Nachricht an den spezifizierten Knoten sendet, die eine Kopie der Liste des spezifizierten Knotens anfordert, und dann die Hinzufügen-Anforderung nur erfüllt, wenn er von dem spezifizierten Knoten eine Liste empfängt, welche die Adresse des Knotens enthält, der die Anforderung empfängt.

4. Verfahren gemäß einem der vorhergehenden Ansprüche, wobei die Nachricht, die eine Verbindung anfordert, an dem ersten Knoten empfangen wird, und wobei
der erste Knoten an den zweiten Knoten eine Anforderung für eine Kopie der Liste des zweiten Knotens sendet;
der zweite Knoten die angeforderte Kopie an den ers-

ten Knoten sendet;
der Schritt (iv)(a) der Auswahl aus der Liste der Adresse eines dritten Knotens an dem ersten Knoten durchgeführt wird; und
die Schritte (iv) (a) und (b) derart durchgeführt werden, dass:
der erste Knoten die Adresse des zweiten Knotens und die Adresse des dritten Knotens zu der Liste des ersten Knotens hinzufügt, jeweils begleitet von einer Markierung, die anzeigt, dass sie unbestätigt ist;
der erste Knoten an den zweiten Knoten eine Nachricht sendet, die anfordert, dass er von seiner Liste die Adresse des dritten Knotens entfernt und sie durch die Adresse des ersten Knotens ersetzt;
der erste Knoten an den dritten Knoten eine Nachricht sendet, die anfordert, dass er von seiner Liste die Adresse des zweiten Knotens entfernt und sie durch die Adresse des ersten Knotens ersetzt;
der zweite Knoten bei Empfang einer derartigen Nachricht von seiner Liste die Adresse des dritten Knotens entfernt, sie durch die Adresse des ersten Knotens ersetzt und eine Bestätigungsnachricht an den ersten Knoten sendet;
der dritte Knoten bei Empfang einer derartigen Nachricht von seiner Liste die Adresse des zweiten Knotens entfernt, sie durch die Adresse des ersten Knotens ersetzt und eine Bestätigungsnachricht an den ersten Knoten sendet;
der erste Knoten bei Empfang der Bestätigungsnachricht von dem zweiten oder dritten Knoten die jeweilige "unbestätigt"-Markierung von seiner Liste entfernt.

5. Verfahren gemäß Anspruch 4, wobei ein Knoten bei Empfang einer Nachricht, die anfordert, dass er von seiner Liste die Adresse eines anderen Knotens entfernt und sie durch die Adresse eines spezifizierten Knotens ersetzt, zuerst eine Nachricht an den spezifizierten Knoten sendet, die eine Kopie der Liste des spezifizierten Knotens anfordert, und dann die Anforderung nur erfüllt, wenn er von dem spezifizierten Knoten eine Liste empfängt, welche die Adresse des Knotens enthält, der die Anforderung empfängt.

6. Verfahren gemäß Anspruch 3 oder 5, wobei die Listenanforderungsnachricht an den spezifizierten Knoten und eine Antwort auf eine solche Nachricht über einen Zwischenknoten derart gesendet werden, dass die Adresse des Knotens, der die Listenanforderungsnachricht sendet, nicht an den spezifizierten Knoten kommuniziert wird.

7. Verfahren gemäß einem der vorhergehenden Ansprüche, wobei jeder Knoten auch Mittel zum Speichern von zumindest einer Ersatzverbindung hat, und aufweist:
in dem Fall eines Empfangs einer Nachricht, die eine Verbindung zwischen einem ersten Knoten und einem zweiten Knoten anfordert, wenn der erste Knoten eine Anzahl von Adressen in seiner Liste hat, die gleich der vorgegebenen Anzahl ist, Einfügen der Adress: des zweiten Knotens in den Ersatzverbindungsspeicher; und
bei Empfang einer späteren Nachricht, die eine Verbindung zwischen dem ersten Knoten und einem anderen Knoten anfordert, Weiterleiten dieser Nachricht an die oder eine Adresse, die aus Ersatzverbindungs-Speichermittel des ersten Knotens abgerufen wird.

8. Virtuelles Netzwerk mit einer Vielzahl von Knoten, wobei jeder Knoten Speichermittel, die eine Liste zum Speichern bis zu einer vorgegebenen Anzahl von Adressen von anderen derartigen Knoten enthalten, und Verarbeitungsmittel hat, wobei die Verarbeitungsmittel ausgebildet sind, in Betrieb alle Schritte des Verfahrens gemäß einem der Ansprüche 1 bis 7 durchzuführen.

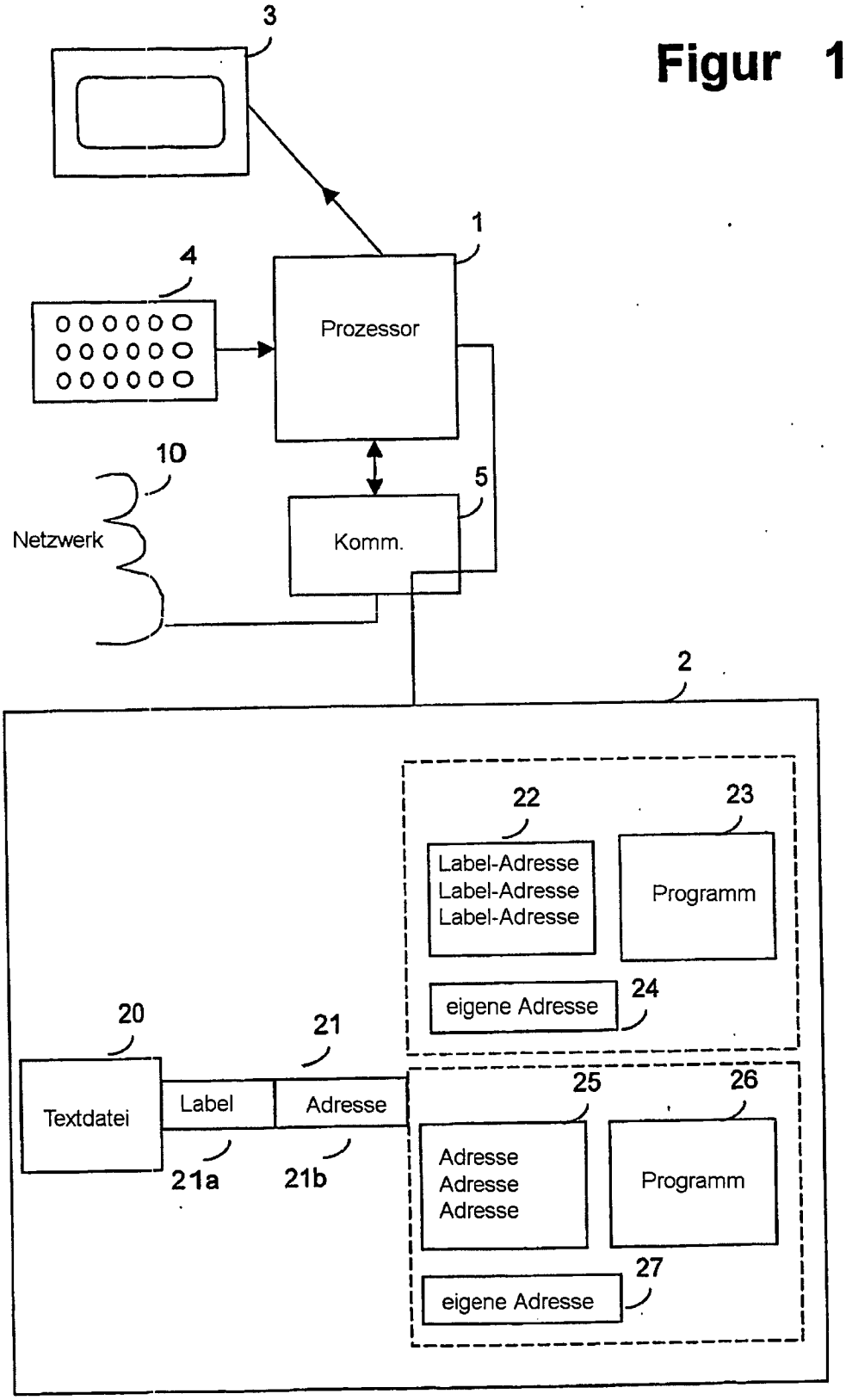
9. Knoten eines virtuellen Netzwerks, wobei der Knoten durch Speichermittel, die eine Liste zum Speichern bis zu einer vorgegebenen Anzahl von Adressen anderer derartiger Knoten enthalten, und Verarbeitungsmittel definiert ist, die programmiert sind, die folgenden Operationen durchzuführen:
Empfangen von Nachrichten;
Antworten auf Nachrichten, die eine Information über den Inhalt der Liste anfordern;
Erfüllen der empfangenen Anforderungen, eine Adresse von der Liste zu entfernen;
Erfüllen der Nachricht, die ein Einfügen einer Adresse in die Liste anfordert; und
als Antwort auf einen Empfang einer Nachricht, die eine Verbindung zwischen dem Knoten und einem zweiten Knoten anfordert:
(A) Erzeugen einer Nachricht an den zweiten Knoten, die eine Information über den Inhalt seiner Liste anfordert;
(B) Bestimmen, ob sowohl der erste Knoten als auch der zweite Knoten jeweils eine Anzahl von Adressen in seiner Liste hat, die geringer als die vorgegebene Anzahl ist;
(C) in dem Fall, dass diese Bedingung erfüllt ist, Einfügen in seine Liste der Adresse des zweiten Knotens und Erzeugen einer Nachricht an den zweiten Knoten, die den zweiten Knoten auffordert, zu seiner Liste die Adresse des Knotens hinzuzufügen;
(D) in dem Fall, dass diese Bedingung nicht erfüllt ist, Bestimmen, ob der Knoten eine Anzahl von Adressen in seiner Liste hat, die zumindest um zwei weniger als die vorgegebene Anzahl ist, und wenn dem so ist
(a) Auswählen von der Liste des zweiten Knotens die Adresse eines dritten Knotens;
(b) Einfügen der Adresse des zweiten Knotens in die Liste des ersten Knotens und Einfügen der Adresse des dritten Knotens in die Liste des ersten Knotens;
(c) Erzeugen einer Nachricht an den zweiten Knoten, die das Entfernen der Adresse des dritten Knotens von der Liste des zweiten Knotens und ein Einfügen der Adresse des Knotens anfordert;

und
(d) Erzeugen einer Nachricht an den dritten Knoten, die das Entfernen der Adresse des zweiten Knotens von der Liste des dritten Knotens und ein Einfügen der Adresse des Knotens anfordert.

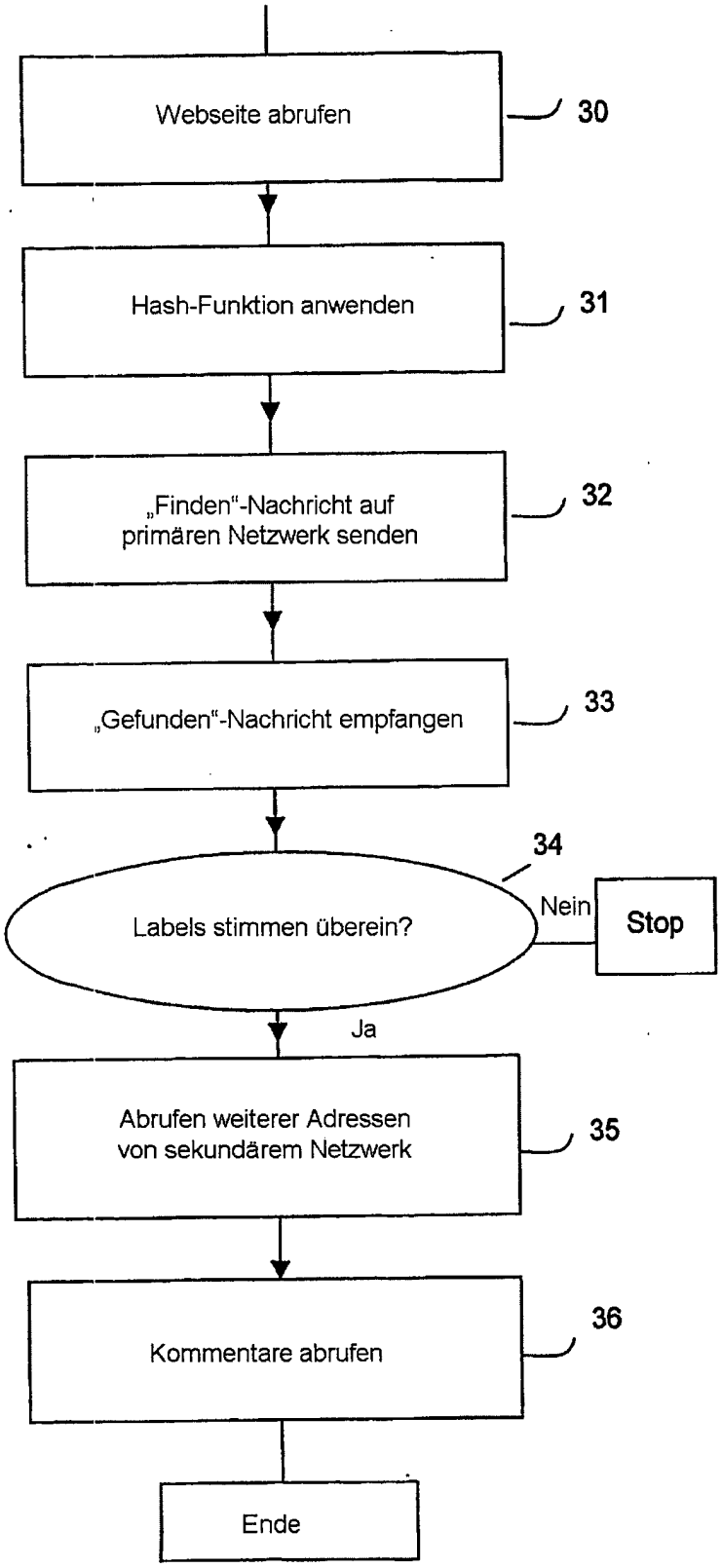
Es folgen 14 Blatt Zeichnungen

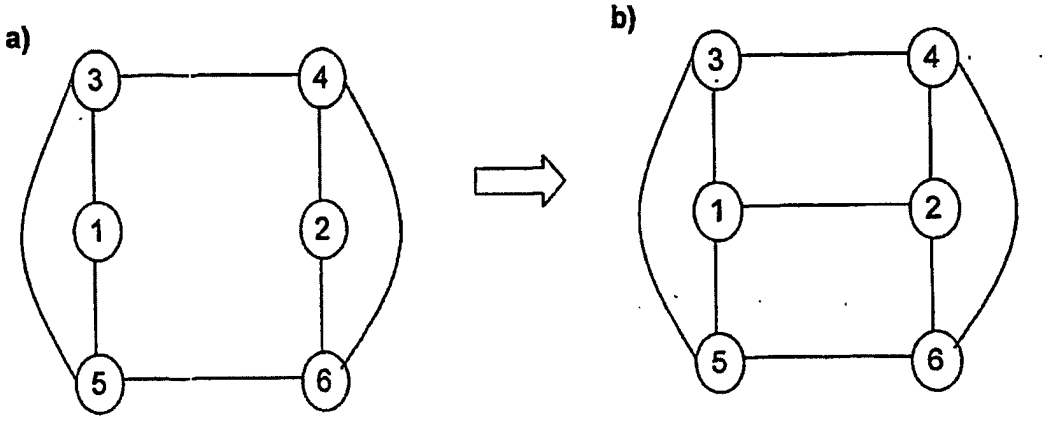
Anhängende Zeichnungen

Figur 1

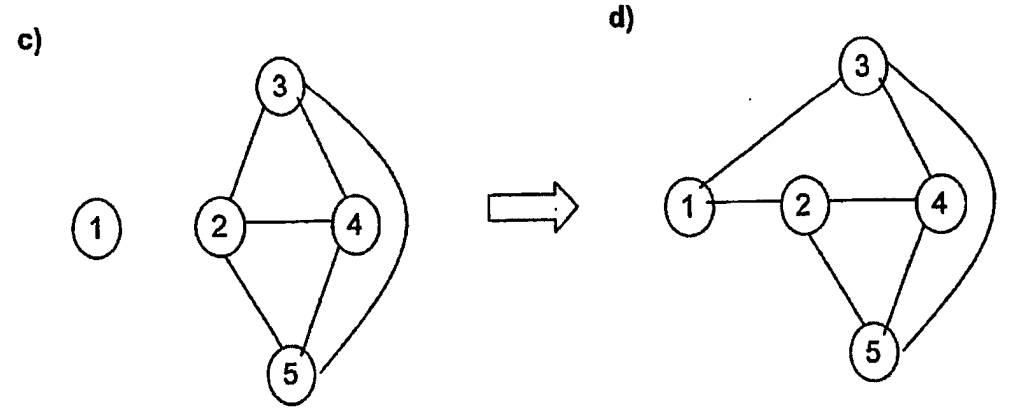


Figur 1A


Webseite abrufen
30
Hash-Funktion anwenden
31
„Finden"-Nachricht auf primären Netzwerk senden
32
„Gefunden"-Nachricht empfangen
33
34
Labels stimmen überein?
Nein
Stop
Ja
Abrufen weiterer Adressen von sekundärem Netzwerk
35
Kommentare abrufen
36
Ende

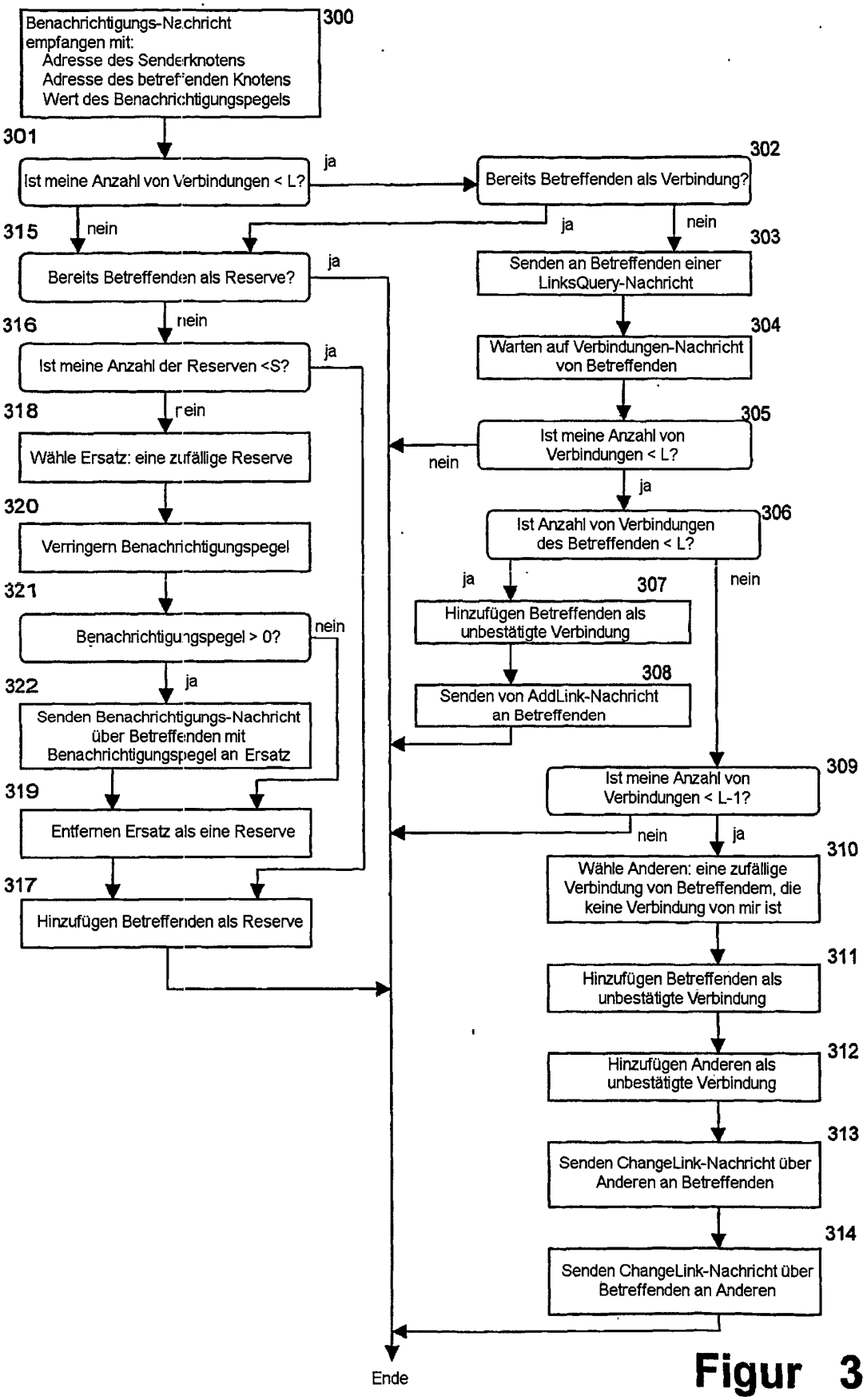


Wie eine neue Verbindung erzeugt wird zwischen zwei Knoten (Knoten 1 und 2), wenn beide Knoten weiter eine neue Verbindung erzeugen können. a) Das Netzwerk, bevor Knoten 1 eine Benachrichtigungs-Nachricht über Knoten 2 empfangen hat, und b) Das Netzwerk, nachdem Knoten 1 erfolgreich eine neue Verbindung erzeugt hat als Antwort auf die Benachrichtigungs-Nachricht.

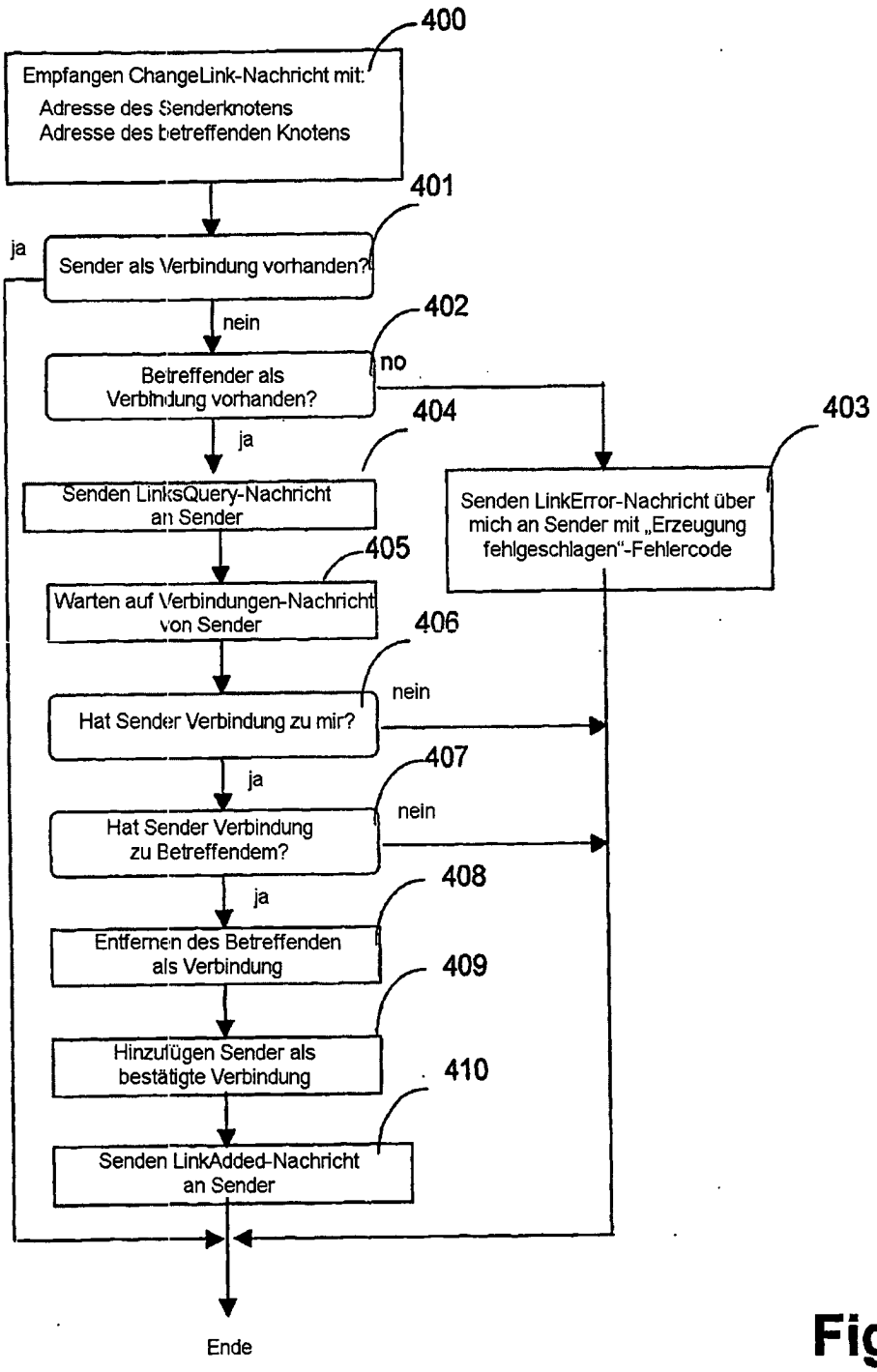


Wie sich ein Knoten (Knoten 1) einfügt in eine existierende Verbindung (zwischen Knoten 2 und Knoten 3), da Knoten 2 bereits die maximale Anzahl von Verbindungen hat. c) Das Netzwerk, bevor Knoten 1 eine Benachrichtigungs-Nachricht über Knoten 2 empfangen hat, und d) Das Netzwerk, nachdem Knoten 1 erfolgreich eine existierende Verbindung in zwei neue geändert hat als Antwort auf die Benachrichtigungs-Nachricht.

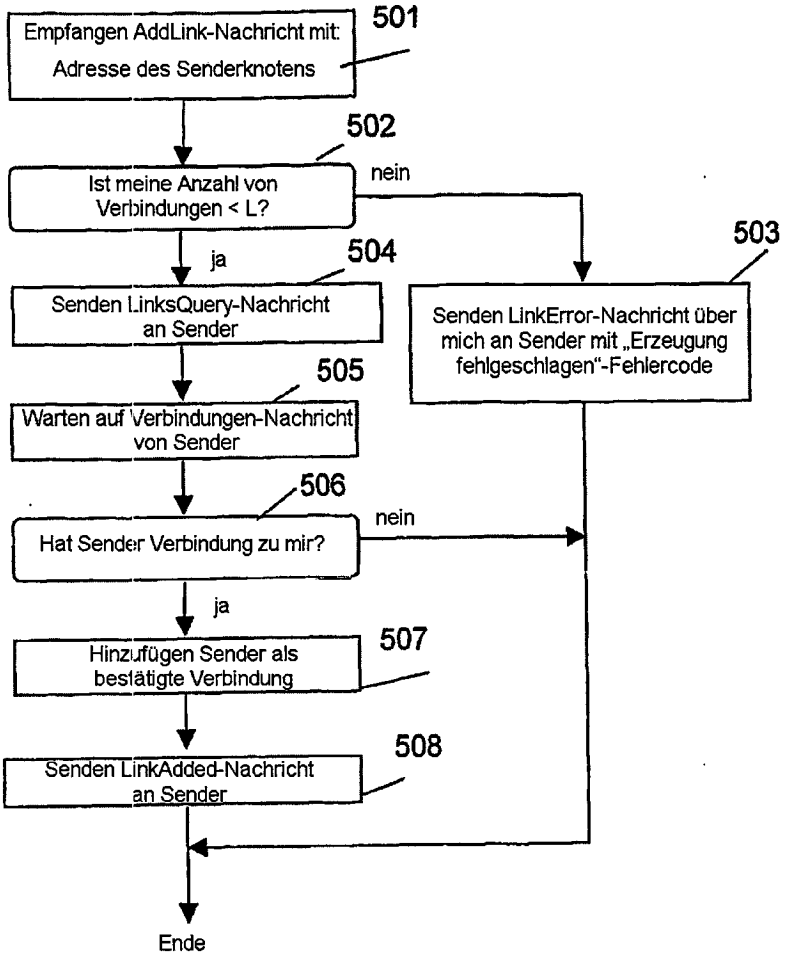
Figur 2


Benachrichtigungs-Nachricht empfangen mit:
Adresse des Senderknotens
Adresse des betreffenden Knotens
Wert des Benachrichtigungspegels
300
301
Ist meine Anzahl von Verbindungen < L?
ja
302
Bereits Betreffenden als Verbindung?
nein
ja
nein
315
Bereits Betreffenden als Reserve?
ja
303
Senden an Betreffenden einer LinksQuery-Nachricht
nein
316
Ist meine Anzahl der Reserven <S?
ja
304
Warten auf Verbindungen-Nachricht von Betreffenden
nein
318
Wähle Ersatz: eine zufällige Reserve
305
Ist meine Anzahl von Verbindungen < L?
nein
ja
320
Verringern Benachrichtigungspegel
306
Ist Anzahl von Verbindungen des Betreffenden < L?
ja
nein
321
Benachrichtigungspegel > 0?
nein
307
Hinzufügen Betreffenden als unbestätigte Verbindung
ja
322
Senden Benachrichtigungs-Nachricht über Betreffenden mit Benachrichtigungspegel an Ersatz
308
Senden von AddLink-Nachricht an Betreffenden
319
Entfernen Ersatz als eine Reserve
309
Ist meine Anzahl von Verbindungen < L-1?
nein
ja
317
Hinzufügen Betreffenden als Reserve
310
Wähle Anderen: eine zufällige Verbindung von Betreffendem, die keine Verbindung von mir ist
311
Hinzufügen Betreffenden als unbestätigte Verbindung
312
Hinzufügen Anderen als unbestätigte Verbindung
313
Senden ChangeLink-Nachricht über Anderen an Betreffenden
314
Senden ChangeLink-Nachricht über Betreffenden an Anderen
Ende

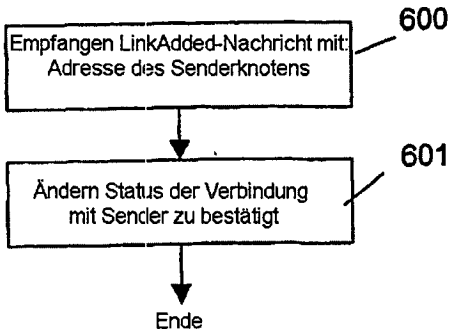
Figur 3


400
Empfangen ChangeLink-Nachricht mit:
Adresse des Senderknotens
Adresse des betreffenden Knotens
401
Sender als Verbindung vorhanden?
ja
nein
402
Betreffender als Verbindung vorhanden?
no
403
Senden LinkError-Nachricht über mich an Sender mit „Erzeugung fehlgeschlagen"-Fehlercode
ja
404
Senden LinksQuery-Nachricht an Sender
405
Warten auf Verbindungen-Nachricht von Sender
406
Hat Sender Verbindung zu mir?
nein
ja
407
Hat Sender Verbindung zu Betreffendem?
nein
ja
408
Entfernen des Betreffenden als Verbindung
409
Hinzufügen Sender als bestätigte Verbindung
410
Senden LinkAdded-Nachricht an Sender
Ende

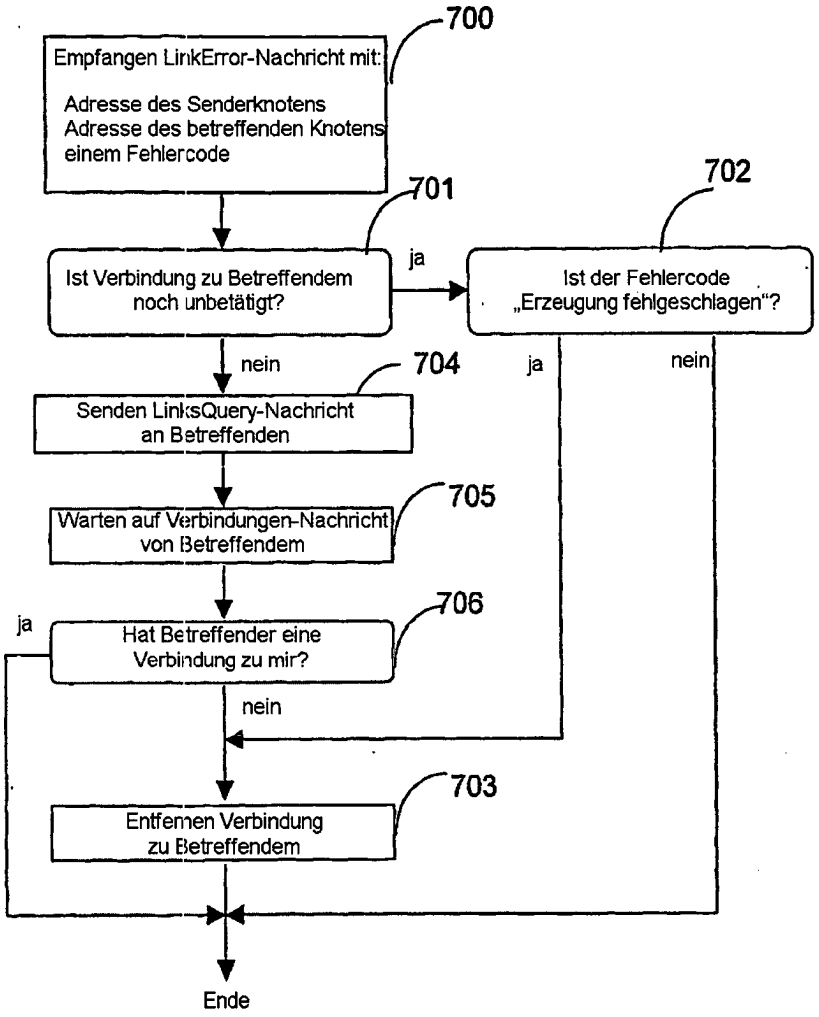
Figur 4



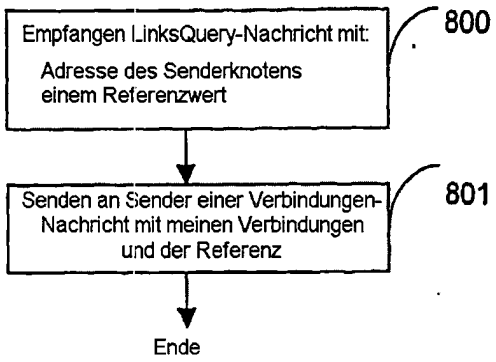
Figur 5



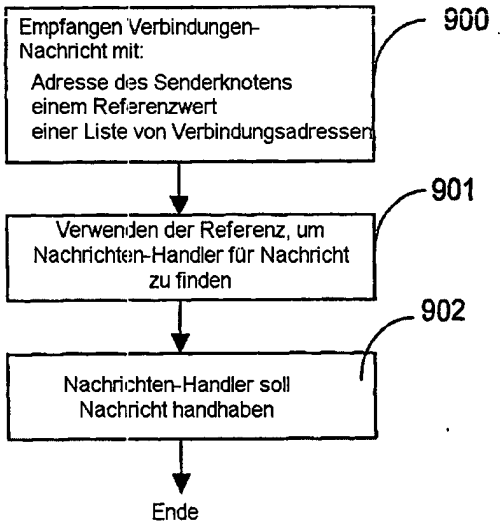
Figur 6


700
Empfangen LinkError-Nachricht mit:
Adresse des Senderknotens
Adresse des betreffenden Knotens
einem Fehlercode
701
Ist Verbindung zu Betreffendem noch unbetätigt?
ja
702
Ist der Fehlercode „Erzeugung fehlgeschlagen"?
nein
704
Senden LinksQuery-Nachricht an Betreffenden
ja
nein
705
Warten auf Verbindungen-Nachricht von Betreffendem
706
ja
Hat Betreffender eine Verbindung zu mir?
nein
703
Entfernen Verbindung zu Betreffendem
Ende

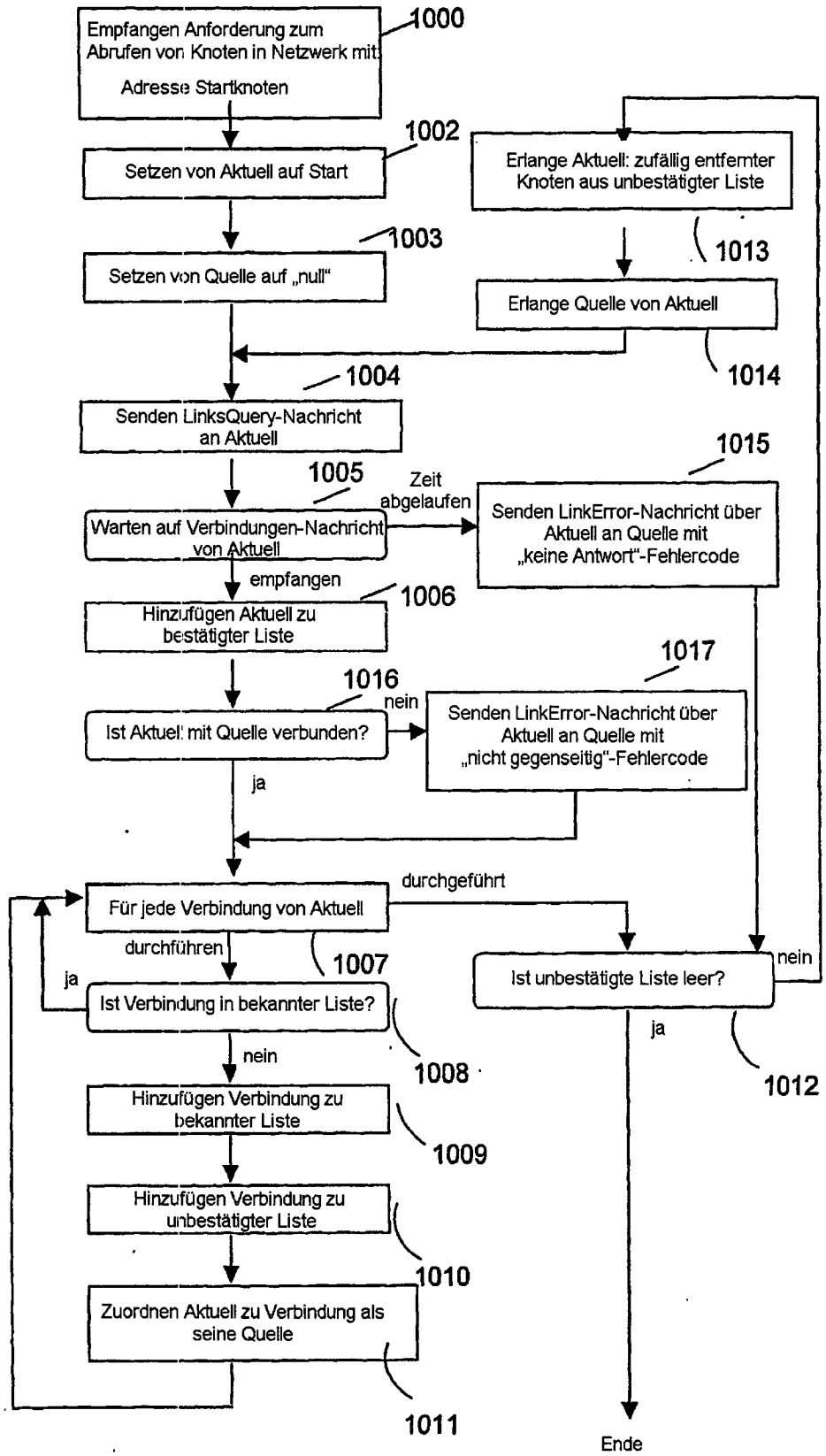
Figur 7



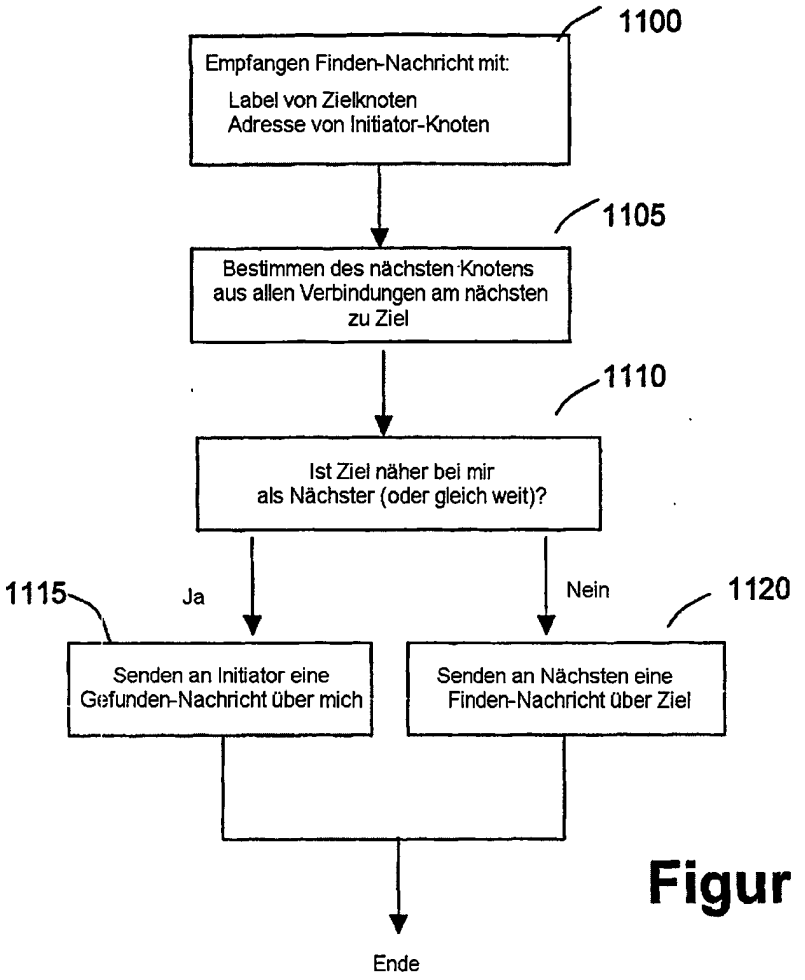
Figur 8



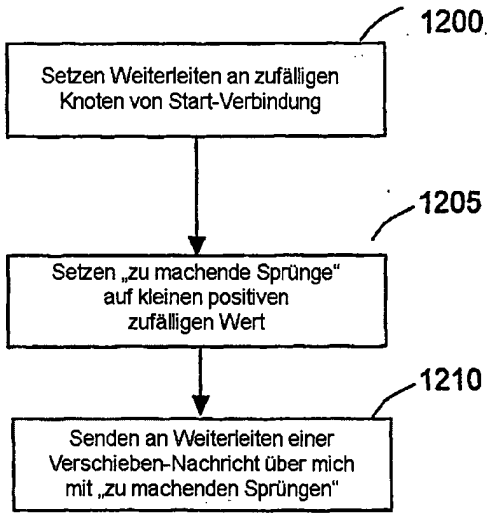
Figur 9


Empfangen Anforderung zum Abrufen von Knoten in Netzwerk mit Adresse Startknoten
1000
Setzen von Aktuell auf Start
1002
Setzen von Quelle auf „null"
1003
Senden LinksQuery-Nachricht an Aktuell
1004
Warten auf Verbindungen-Nachricht von Aktuell
1005
Zeit abgelaufen
empfangen
Hinzufügen Aktuell zu bestätigter Liste
1006
Ist Aktuell mit Quelle verbunden?
1016
nein
ja
Senden LinkError-Nachricht über Aktuell an Quelle mit „keine Antwort"-Fehlercode
1015
Senden LinkError-Nachricht über Aktuell an Quelle mit „nicht gegenseitig"-Fehlercode
1017
Für jede Verbindung von Aktuell
durchgeführt
durchführen
Ist Verbindung in bekannter Liste?
1007
ja
nein
1008
Hinzufügen Verbindung zu bekannter Liste
1009
Hinzufügen Verbindung zu unbestätigter Liste
1010
Zuordnen Aktuell zu Verbindung als seine Quelle
1011
Ist unbestätigte Liste leer?
nein
ja
1012
Erlange Aktuell: zufällig entfernter Knoten aus unbestätigter Liste
1013
Erlange Quelle von Aktuell
1014
Ende

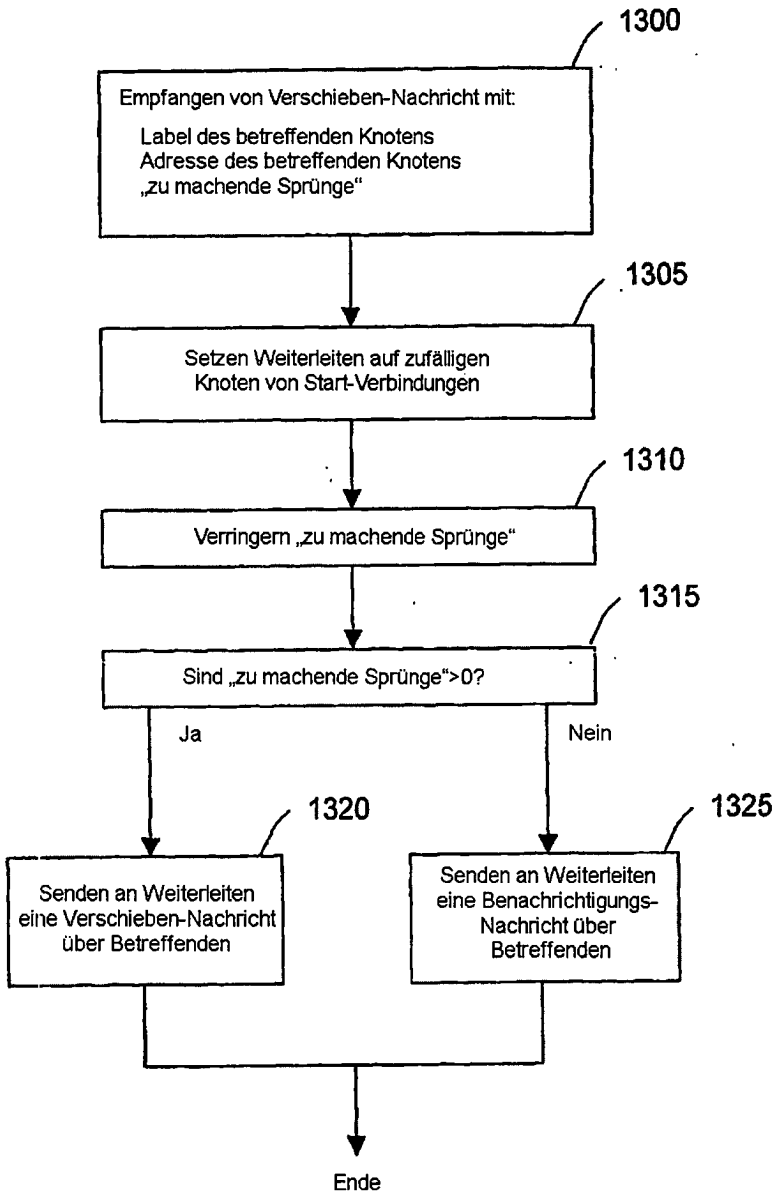
Figur 10



Figur 11



Figur 12


1300
Empfangen von Verschieben-Nachricht mit:
Label des betreffenden Knotens
Adresse des betreffenden Knotens
„zu machende Sprünge"
1305
Setzen Weiterleiten auf zufälligen Knoten von Start-Verbindungen
1310
Verringern „zu machende Sprünge"
1315
Sind „zu machende Sprünge">0?
Ja
Nein
1320
Senden an Weiterleiten eine Verschieben-Nachricht über Betreffenden
1325
Senden an Weiterleiten eine Benachrichtigungs-Nachricht über Betreffenden
Ende

Figur 13

Figur 14

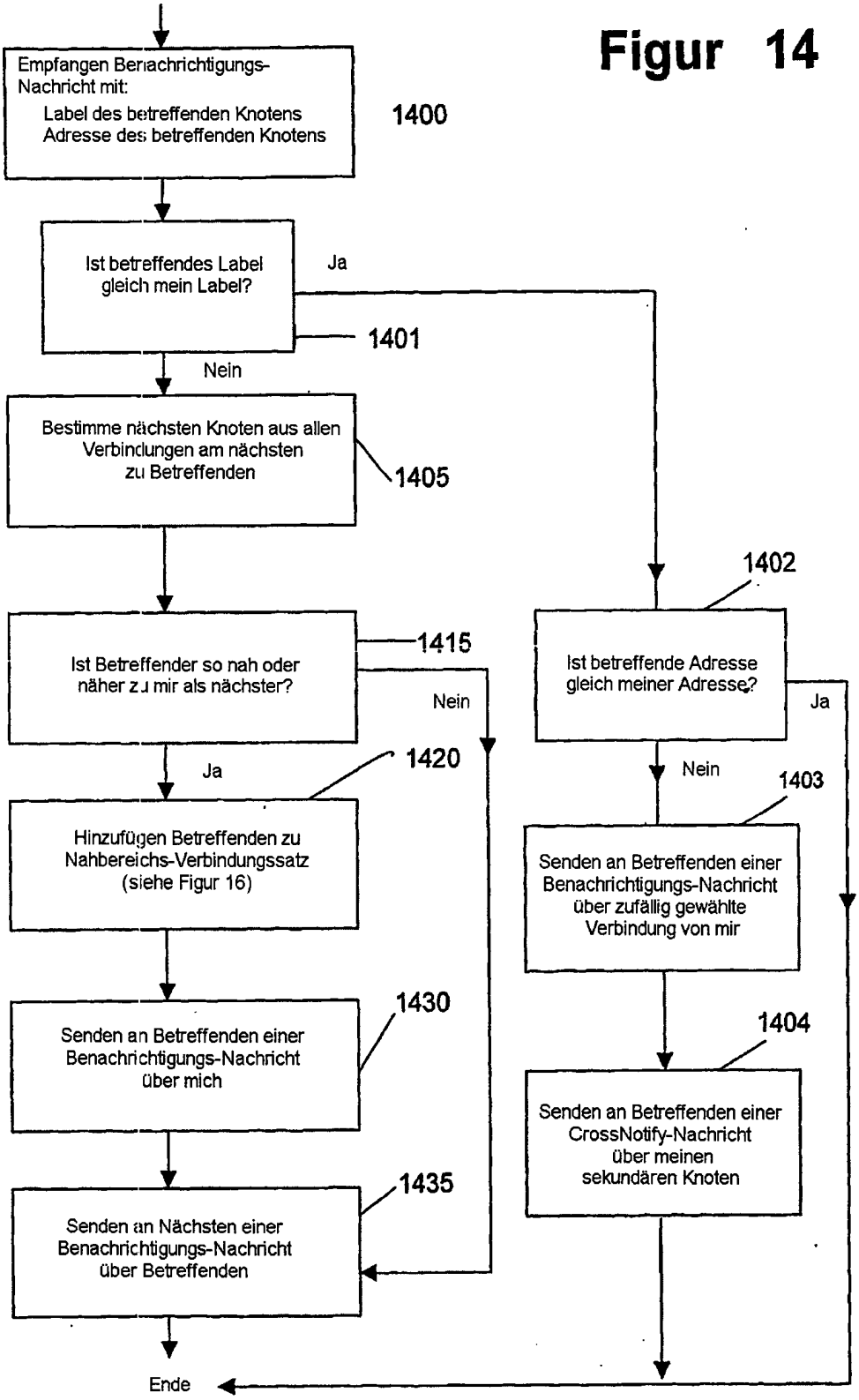
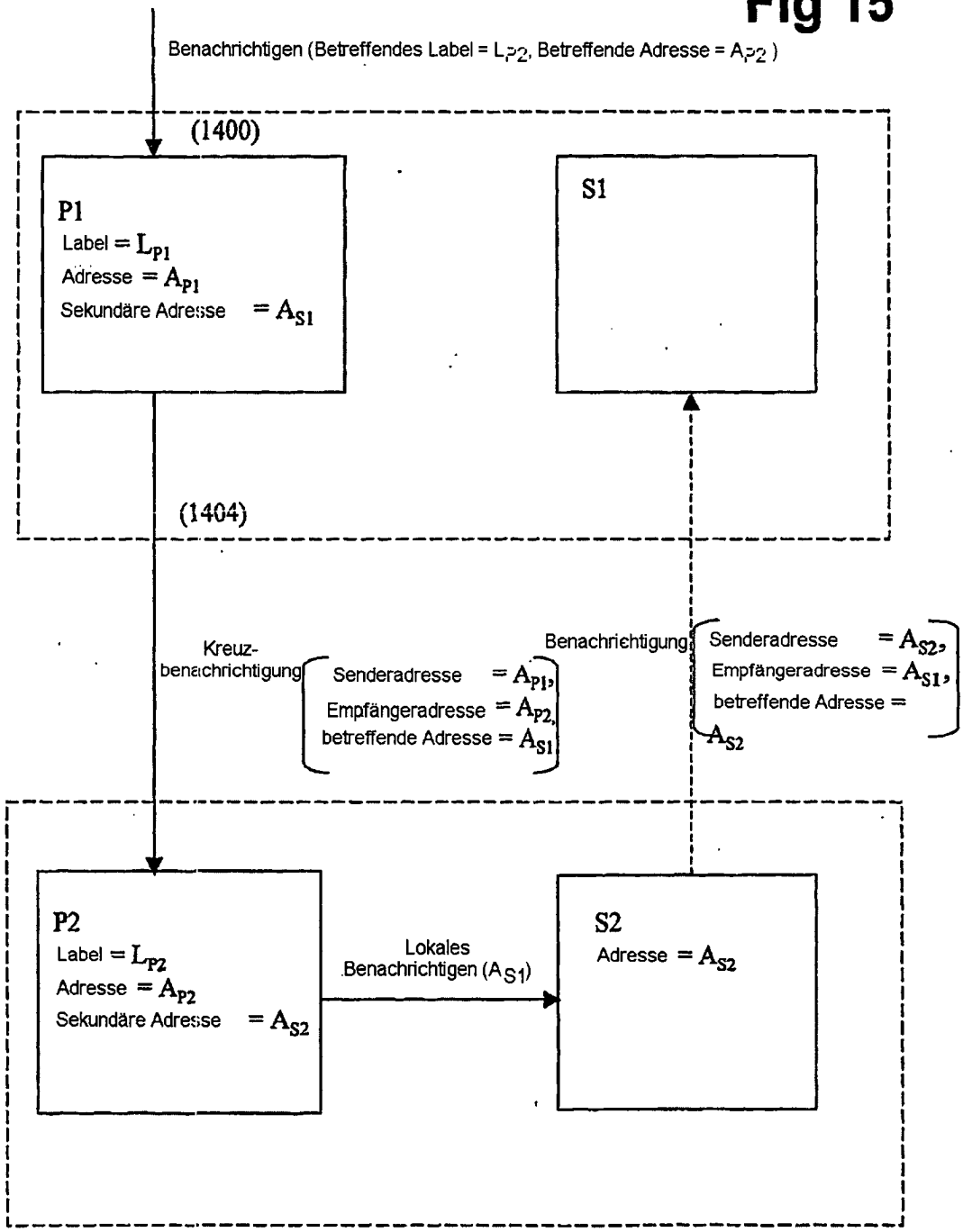
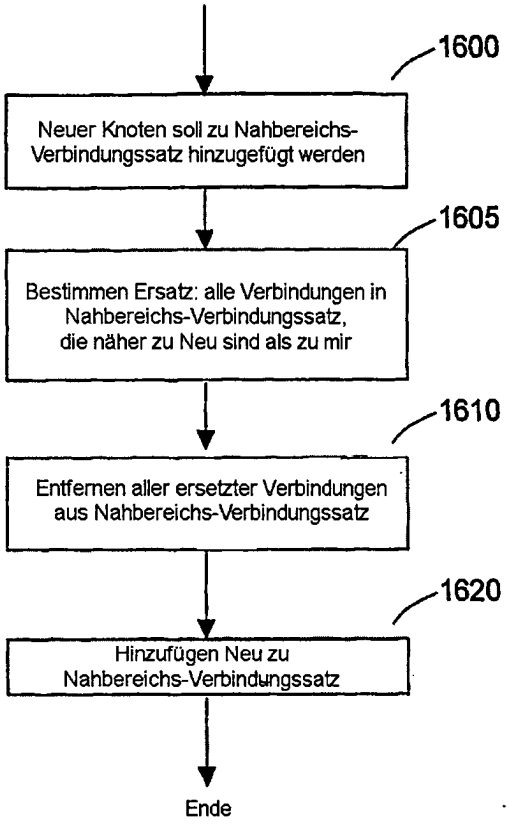

Empfangen Benachrichtigungs-Nachricht mit:
Label des betreffenden Knotens
Adresse des betreffenden Knotens
1400
Ist betreffendes Label gleich mein Label?
Ja
1401
Nein
Bestimme nächsten Knoten aus allen Verbindungen am nächsten zu Betreffenden
1405
1402
Ist Betreffender so nah oder näher zu mir als nächster?
1415
Nein
Ja
Ist betreffende Adresse gleich meiner Adresse?
Ja
Nein
1420
1403
Hinzufügen Betreffenden zu Nahbereichs-Verbindungssatz (siehe Figur 16)
Senden an Betreffenden einer Benachrichtigungs-Nachricht über zufällig gewählte Verbindung von mir
1430
1404
Senden an Betreffenden einer Benachrichtigungs-Nachricht über mich
Senden an Betreffenden einer CrossNotify-Nachricht über meinen sekundären Knoten
1435
Senden an Nächsten einer Benachrichtigungs-Nachricht über Betreffenden
Ende

Fig 15


Benachrichtigen (Betreffendes Label = L_{P2}, Betreffende Adresse = A_{P2})
(1400)
P1
Label = L_{P1}
Adresse = A_{P1}
Sekundäre Adresse = A_{S1}
S1
(1404)
Kreuz-benachrichtigung
Senderadresse = A_{P1},
Empfängeradresse = A_{P2},
betreffende Adresse = A_{S1}
Benachrichtigung
Senderadresse = A_{S2},
Empfängeradresse = A_{S1},
betreffende Adresse = A_{S2}
P2
Label = L_{P2}
Adresse = A_{P2}
Sekundäre Adresse = A_{S2}
Lokales Benachrichtigen (A_{S1})
S2
Adresse = A_{S2}


1600
Neuer Knoten soll zu Nahbereichs-Verbindungssatz hinzugefügt werden
1605
Bestimmen Ersatz: alle Verbindungen in Nahbereichs-Verbindungssatz, die näher zu Neu sind als zu mir
1610
Entfernen aller ersetzter Verbindungen aus Nahbereichs-Verbindungssatz
1620
Hinzufügen Neu zu Nahbereichs-Verbindungssatz
Ende

Figur 16