



(19)
Bundesrepublik Deutschland
Deutsches Patent- und Markenamt

(10) **DE 601 19 331 T2** 2006.09.14

(12) **Übersetzung der europäischen Patentschrift**

(97) **EP 1 305 908 B1**

(21) Deutsches Aktenzeichen: **601 19 331.8**

(86) PCT-Aktenzeichen: **PCT/US01/24240**

(96) Europäisches Aktenzeichen: **01 955 034.2**

(87) PCT-Veröffentlichungs-Nr.: **WO 2002/011366**

(86) PCT-Anmeldetag: **31.07.2001**

(87) Veröffentlichungstag
der PCT-Anmeldung: **07.02.2002**

(97) Erstveröffentlichung durch das EPA: **02.05.2003**

(97) Veröffentlichungstag
der Patenterteilung beim EPA: **03.05.2006**

(47) Veröffentlichungstag im Patentblatt: **14.09.2006**

(51) Int Cl.⁸: **H04L 12/18** (2006.01)
H04L 12/56 (2006.01)

(30) Unionspriorität:

629576	31.07.2000	US
629570	31.07.2000	US
629577	31.07.2000	US
629575	31.07.2000	US
629572	31.07.2000	US
629023	31.07.2000	US
629043	31.07.2000	US
629024	31.07.2000	US
629042	31.07.2000	US

(73) Patentinhaber:

The Boeing Co., Seattle, Wash., US

(74) Vertreter:

**Grünecker, Kinkeldey, Stockmair &
Schwanhäusser, 80538 München**

(84) Benannte Vertragsstaaten:

**AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT,
LI, LU, MC, NL, PT, SE, TR**

(72) Erfinder:

**BOURASSA, E., Virgil, Bellevue, WA, US; HOLT, B.,
Fred, Seattle, WA 98105, US**

(54) Bezeichnung: **RUNDSENDENETZ**

Anmerkung: Innerhalb von neun Monaten nach der Bekanntmachung des Hinweises auf die Erteilung des europäischen Patents kann jedermann beim Europäischen Patentamt gegen das erteilte europäische Patent Einspruch einlegen. Der Einspruch ist schriftlich einzureichen und zu begründen. Er gilt erst als eingelegt, wenn die Einspruchsgebühr entrichtet worden ist (Art. 99 (1) Europäisches Patentübereinkommen).

Die Übersetzung ist gemäß Artikel II § 3 Abs. 1 IntPatÜG 1991 vom Patentinhaber eingereicht worden. Sie wurde vom Deutschen Patent- und Markenamt inhaltlich nicht geprüft.

Beschreibung

[0001] Die beschriebene Technologie betrifft im Allgemeinen ein Computernetzwerk und konkret einen Übertragungskanal für einen Teilsatz eines Computers eines zugrunde liegenden Netzwerks.

[0002] Es gibt eine Vielzahl verschiedener Computernetzwerk-Kommunikationsverfahren, z.B. Punkt-zu-Punkt-Netzwerkprotokolle, Client/Server-Middleware, Multicasting-Netzwerkprotokolle und Peer-to-Peer-Middleware. Jedes dieser Kommunikationsverfahren hat seine eigenen Vor- und Nachteile, keines eignet sich jedoch besonders für den gleichzeitigen Informationsaustausch zwischen Computern, die weit verteilt sind. Z.B. ist es bei Verarbeitungsanwendungen zum Zwecke der Zusammenarbeit, beispielsweise bei Netzwerkmeeting-Programmen notwendig, Informationen zeitnah an alle Teilnehmer zu verteilen, die sich geografisch an verschiedenen Orten befinden.

[0003] Die Punkt-zu-Punkt-Netzwerkprotokolle, z.B. UNIX-Leitungen, TCP/IP und UDP, ermöglichen es, dass Prozesse auf verschiedenen Computern über Punkt-zu-Punkt-Verbindungen miteinander kommunizieren. Die Verbindung aller Teilnehmer untereinander mithilfe der Punkt-zu-Punkt-Verbindungen funktioniert, obwohl theoretisch möglich, bei zunehmender Anzahl von Teilnehmern nicht sehr gut. Z.B. müsste jeder teilnehmende Prozess seine direkten Verbindungen zu allen anderen teilnehmenden Prozessen selbst verwalten. Programmierer finden es jedoch schon sehr schwierig, einzelne Verbindungen zu verwalten, und die Verwaltung zahlreicher Verbindungen ist noch viel komplexer. Darüber hinaus können die teilnehmenden Prozesse auf die Anzahl der direkten Verbindungen begrenzt sein, welche sie unterstützen. Dadurch wird die Anzahl möglicher Teilnehmer bei der gemeinsamen Nutzung von Informationen begrenzt.

[0004] Client/Server-Middleware-Systeme verfügen über einen Server, der die Verbindungen zwischen den einzelnen Clients koordiniert, die die Informationen gemeinsam nutzen. Der Server fungiert als zentrale Einrichtung zum Steuern des Zugriffs auf die gemeinsam genutzten Ressourcen. Zu Beispielen für Client/Server-Middleware-Systeme gehören Remote Procedure Calls („RPC“), Datenbank-Server und die Common Object Request Broker Architecture („CORBA“). Client/Server-Middleware-Systeme eignen sich nicht besonders gut für die gemeinsame Nutzung von Informationen zwischen vielen Teilnehmern. Wenn ein Client Informationen speichert, die gemeinsam auf dem Server genutzt werden sollen, müsste konkret jeder andere Client den Server regelmäßig abfragen um festzustellen, ob neue, gemeinsam zu nutzende Informationen vorliegen. Durch eine derartige Abfrage entsteht ein großer zusätzlicher Aufwand für das Kommunikationsnetzwerk. Als Alternative dazu kann sich jeder Client für den Rückruf durch den Server eintragen lassen, welchen der Server in Gang setzt, wenn neue Informationen für die gemeinsame Nutzung vorliegen. Ein derartiges Rückrufverfahren stellt einen Leistungsengpass dar, da ein einziger Server jeden Client zurückrufen muss, wenn neue Informationen für die gemeinsame Nutzung vorliegen. Außerdem hängt die Zuverlässigkeit der gesamten Informationsnutzung von der Zuverlässigkeit des einen Servers ab. Folglich würde ein Defekt an einem einzigen Computer (d.h. dem Server) Datenübertragungen zwischen sämtlichen Clients verhindern.

[0005] Die Multicasting-Netzwerkprotokolle ermöglichen das Senden von Nachrichten zu mehreren Empfängern eines Netzwerks. Bei den gegenwärtigen Implementierungen solcher Multicasting-Netzwerkprotokolle entsteht meist eine unannehmbare Zusatzbelastung für das dazugehörige Netzwerk. Z.B. würde das UPD-Multicasting das Internet überlasten, während es versucht, sämtliche möglichen Teilnehmer zu lokalisieren. Bei dem IP-Multicasting gibt es andere Probleme, zu denen die Notwendigkeit einer speziellen Infrastruktur (z.B. Router) für die Unterstützung der effizienten Informationsnutzung gehört.

[0006] Die Peer-to-Peer-Middleware-Kommunikationssysteme beruhen auf einem Multicasting-Netzwerkprotokoll oder einem Graphen von Punkt-zu-Punkt-Netzwerkprotokollen. Eine derartige Peer-to-Peer-Middleware wird durch den Internet-Standard T.120 geschaffen, der in Produkten wie Data Connection's D.C.-Share und Microsoft's NetMeeting zur Anwendung kommt. Bei diesen Peer-to-Peer-Middleware-Systemen erstellt ein Nutzer einen Punkt-zu-Punkt-Graphen der Verbindungen, die für die gemeinsame Informationsnutzung verwendet werden. Somit ist es weder günstig noch wünschenswert, Peer-to-Peer-Middleware-Systeme zum Einsatz zu bringen, wenn mehr als eine geringe Anzahl von Teilnehmern erwünscht ist. Darüber hinaus ist die zugrunde liegende Architektur des Internet-Standards T.120 eine Baumstruktur, wobei die Zuverlässigkeit des gesamten Netzwerks von der Zuverlässigkeit des Wurzelknotens des Baums abhängt. D.h. jede Nachricht muss den Wurzelknoten passieren, um von sämtlichen Teilnehmern empfangen zu werden.

[0007] Wünschenswert wäre es, ein zuverlässiges Kommunikationsnetzwerk zur Verfügung zu haben, welches sich für die gleichzeitige gemeinsame Nutzung von Informationen zwischen einer großen Anzahl von Prozessen eignet, die weit verstreut sind.

[0008] Da das Internet die elektronische Kommunikation zwischen Verkäufern und Käufern vereinfacht, wird es zunehmend für den „elektronischen Handel“ genutzt. Das Internet umfasst eine große Anzahl von Computern und Computernetzwerken, die über Kommunikationskanäle miteinander verbunden sind. Unter elektronischem Handel werden allgemein Geschäftsabschlüsse verstanden, die zumindest teilweise mithilfe von Computersystemen der an den Geschäftsabschlüssen beteiligten Seiten ausgeführt werden. Z.B. kann ein Käufer einen Personalcomputer verwenden, um sich über das Internet mit dem Computer des Verkäufers in Verbindung zu setzen. Anschließend kann der Käufer mit dem Computer des Verkäufers interagieren und das Geschäft abwickeln. Wenngleich viele Handelsaktivitäten heutzutage über den elektronischen Handel abgewickelt werden könnten, hängen die Akzeptanz und die Verbreitung des elektronischen Handels zum großen Teil davon ab, wie sich dieser elektronische Handel abwickeln lässt. Ist er leicht zu bewerkstelligen, dann wird sich sogar der Computerneuling auf den elektronischen Handel einlassen. Deshalb ist es wichtig Verfahren zu entwickeln, die den elektronischen Handel erleichtern.

[0009] Weiterhin wird das Internet auch für andere Arten von Handelsaktivitäten genutzt. Z.B. sind einige Server-Computersysteme zum Unterstützen von elektronischen Auktionen entwickelt worden. Für eine elektronische Auktion stellt der Verkäufer eines Artikels einem Server-Computersystem eine Definition der Auktion über Webseiten zur Verfügung. Diese Definition enthält eine Beschreibung des Artikels, eine Auktionszeit und wahlweise ein Mindestgebot. Anschließend führt das Server-Computersystem die Auktion während der angegebenen Zeitspanne durch. Potenzielle Käufer können das Server-Computersystem nach einer sie interessierenden Auktion durchsuchen. Wenn eine solche Auktion gefunden worden ist, kann der potenzielle Käufer die Angebotsübersicht für die Auktion betrachten und ein Gebot für den Artikel eingeben. Nach Beendigung der Auktion benachrichtigt das Server-Computersystem den Höchstbietenden und den Verkäufer (z.B. über E-Mail), sodass sie das Geschäft abwickeln können.

[0010] Obwohl derartige Auktionsserver die elektronische Durchführung von Auktionen erleichtern, hat dies mehrere Nachteile. Zuerst einmal hängt die Zuverlässigkeit des Auktionssystems von der Zuverlässigkeit des Auktionsservers selbst ab. Sollte der Auktionsserver ausfallen, könnten die Auktionen nicht durchgeführt werden. Somit kann ein Ausfall das gesamte Auktionssystem zum Erliegen bringen. Zweitens bilden die von Auktionsservern durchgeführten Auktionen die traditionellen Auktionen ohne Computer nicht nahe genug ab. Speziell enden elektronische Auktionen meist zu einem feststehenden Zeitpunkt, wohingegen eine Auktion ohne Computer im typischen Fall dann endet, wenn ein Auktionator feststellt, dass ein weiteres Gebot nicht wahrscheinlich ist. Z.B. kann eine elektronische Auktion verkünden, dass sie an einem bestimmten Tag um 17 Uhr schließt. Bis zu diesem Zeitpunkt können Bieter Gebote abgeben. Traditionelle Auktionen hingegen haben eine festgelegte Anfangszeit, ihr Schluss hingegen ist abhängig von der Bietaktivität. Außerdem ermöglichen diese elektronischen Auktionen, speziell wenn es internetbasierte sind, keine Echtzeitbenachrichtigung über die Gebote. Ein Bieter findet lediglich über mehrere Möglichkeiten heraus, ob er überboten worden ist. Dies kann geschehen, indem er regelmäßig auf die Seite der Auktion zugreift, um das aktuelle Höchstgebot zu sehen. Ein derartiger wiederholter Zugriff auf die Webseite der Auktion ist mühselig. Einige Auktionsserver versenden E-Mails, wenn jemand überboten wurde. Jedoch können solche elektronischen Mails mitunter nicht zeitig genug eintreffen, als dass der Bieter noch ein neues Gebot abgeben könnte.

[0011] Somit wäre ein elektronisches Auktionssystem wünschenswert, welches diese Nachteile der aktuellen serverbasierten Auktionssysteme umgeht und die traditionellen Auktionen ohne Computer genauer nachbildet.

[0012] ALAGER S. et al., "Reliable broadcast in mobile wireless networks", Military Communications Conference, IEEE San Diego, Nov. 1995, pp. 236–240, beschreibt das Modell eines Drahtlos-Netzwerkes, welches aus mehreren mobilen Hosts besteht, die über ein unregelmäßiges Gebiet verteilt sind. Ein mobiler Host erkennt seinen Nachbarn, indem er regelmäßig eine Suchnachricht sendet. Ein Host, der eine Suchnachricht hört, sendet dem suchenden Host eine Quittierungsmeldung. Jeder Host pflegt eine Liste von Nachbarn und aktualisiert diese Liste periodisch auf der Grundlage der empfangenen Quittierungsmeldungen. Wenn zwei Hosts Nachbarn werden, wird eine Drahtlosverbindung zwischen ihnen hergestellt, und sie führen eine Handshake-Prozedur aus. Als Teil der Handshake-Prozedur aktualisieren sie jeweils ihre Nachbarlisten. Um eine Nachricht zu senden, überträgt ein mobiler Host die Nachricht an alle seine Nachbarn. Nach Empfang einer Nachricht überträgt ein mobiler Zwischen-Host die Nachricht an all seine Nachbarn. Durch Zählung der Weiterübertragungen und einen Zeitstempel wird die Verteilung der Nachrichten in einem großen Netzwerk eingegrenzt.

[0013] US-A-5056085 beschreibt einen Flood-and-Forward-Routing-Algorithmus zum Senden von Paketen in Paketvermittlungsnetzwerken. Der Flood-and-Forward-Routing-Algorithmus umfasst das periodische Senden eines Sendepakets in einem eingeschränkten Flood Broadcast. Jeder Empfangsknoten stellt fest, ob er

dieses Sendepaket bereits zuvor gesehen hat. Falls ja, verwirft er das Paket. Falls nicht, sendet er eine Empfangsquittung für das Paket an den Knoten zurück, der das Paket an ihn gesendet hat, und sendet das Sendepaket entsprechend dem eingeschränkten Flood-Broadcast zu weiteren Knoten. Jeder Knoten zeichnet in einer Broadcast-Routing-Tabelle die anderen Knoten auf, von denen er eine Quittung empfängt, und sendet solange weitere Sendepakete zu diesem Knoten, bis von dem nächsten eingeschränkten Flood-Paket neue Routes festgelegt werden.

[0014] Die Aufgabe der vorliegenden Erfindung besteht in der Schaffung eines Computernetzwerks mit verbesserten und zuverlässigen Kommunikationsmöglichkeiten für den Informationsaustausch zwischen kommunizierenden Teilnehmern.

[0015] Die Aufgabe wird durch den Gegenstand des unabhängigen Patentanspruchs gelöst. Von den abhängigen Patentansprüchen werden bevorzugte Ausführungsformen der Erfindung definiert.

KURZE BESCHREIBUNG DER ZEICHNUNGEN

[0016] [Fig. 1](#) stellt einen Graphen dar, der 4-regulär und 4-zusammenhängend ist und einen Übertragungskanal darstellt.

[0017] [Fig. 2](#) stellt einen Graphen dar, der 20 Computer repräsentiert, die an einen Übertragungskanal angeschlossen sind.

[0018] [Fig. 3A](#) und [Fig. 3B](#) veranschaulichen den Prozess des Anschließens eines neuen Computers Z an den Übertragungskanal.

[0019] [Fig. 4A](#) veranschaulicht den Übertragungskanal aus [Fig. 1](#) mit einem hinzugefügten Computer.

[0020] [Fig. 4B](#) zeigt den Übertragungskanal aus [Fig. 4A](#) mit einem hinzugefügten Computer.

[0021] [Fig. 4C](#) zeigt ebenfalls den Übertragungskanal aus [Fig. 4A](#) mit einem hinzugefügten Computer.

[0022] [Fig. 5A](#) veranschaulicht das Abschalten eines Computers von dem Übertragungskanal in geplanter Art und Weise.

[0023] [Fig. 5B](#) veranschaulicht das Abschalten eines Computers von dem Übertragungskanal in ungeplanter Art und Weise.

[0024] [Fig. 5C](#) zeigt die Nachbarn mit leeren Ports.

[0025] [Fig. 5D](#) zeigt zwei Computer, die keine Nachbarn sind und jetzt leere Ports aufweisen.

[0026] [Fig. 5E](#) zeigt die Nachbarn mit leeren Ports im Small Regime.

[0027] [Fig. 5F](#) zeigt die Situation aus [Fig. 5E](#) im Large Regime.

[0028] [Fig. 6](#) ist ein Blockdiagramm, welches Komponenten eines Computers darstellt, der an einen Übertragungskanal angeschlossen ist.

[0029] [Fig. 7](#) ist ein Blockdiagramm, welches die Teilkomponenten der Senderkomponente in einer Ausführungsform veranschaulicht.

[0030] [Fig. 8](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungsroutine in einer Ausführungsform zeigt.

[0031] [Fig. 9](#) ist ein Flussdiagramm, welches die Verarbeitung der Portalcomputer-Suchroutine des Computers nach einer Ausführungsform zeigt.

[0032] [Fig. 10](#) ist ein Flussdiagramm, welches die Verarbeitung der Verarbeitungskontakt-Routine nach einer Ausführungsform veranschaulicht.

- [0033] [Fig. 11](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungsanforderungs-Routine nach einer Ausführungsform zeigt.
- [0034] [Fig. 12](#) ist ein Flussdiagramm der Verarbeitung der Überprüfung der externen Anrufroutine nach einer Ausführungsform.
- [0035] [Fig. 13](#) ist ein Flussdiagramm der Verarbeitung der Verbindungsherstell-Routine nach einer Ausführungsform.
- [0036] [Fig. 14](#) ist ein Flussdiagramm, welches die Verarbeitung der externen Dispatcher-Routine nach einer Ausführungsform zeigt.
- [0037] [Fig. 15](#) ist ein Flussdiagramm, das die Verarbeitung der Verbindungssuchanrufverarbeitungs-Routine nach einer Ausführungsform darstellt.
- [0038] [Fig. 16](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungsanforderungsanruf-Verarbeitungsroutine nach einer Ausführungsform darstellt.
- [0039] [Fig. 17](#) ist ein Flussdiagramm, welches die Verarbeitung der Nachbarhinzufügungs-Routine nach einer Ausführungsform veranschaulicht.
- [0040] [Fig. 18](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungskantensuchweiterleitungsroutine nach einer Ausführungsform veranschaulicht.
- [0041] [Fig. 19](#) ist ein Flussdiagramm, welches die Verarbeitung der Kantenvorschlagsanrufabwicklungs-Routine nach einer Ausführungsform veranschaulicht.
- [0042] [Fig. 20](#) ist ein Flussdiagramm, welches die Verarbeitung der Portverbindungsanrufabwicklungs-Routine nach einer Ausführungsform veranschaulicht.
- [0043] [Fig. 21](#) ist ein Flussdiagramm, welches die Verarbeitung der Lochauffüll-Routine nach einer Ausführungsform veranschaulicht.
- [0044] [Fig. 22](#) ist ein Flussdiagramm, welches die Verarbeitung der internen Dispatcher-Routine nach einer Ausführungsform veranschaulicht.
- [0045] [Fig. 23](#) ist ein Flussdiagramm, welches die Verarbeitung der Sendenachrichtabwicklungs-Routine nach einer Ausführungsform veranschaulicht.
- [0046] [Fig. 24](#) ist ein Flussdiagramm, welches die Verarbeitung der Sendenachrichtverteilungs-Routine nach einer Ausführungsform veranschaulicht.
- [0047] [Fig. 26](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungsportsuchanweisungsabwicklungs-Routine nach einer Ausführungsform veranschaulicht.
- [0048] [Fig. 27](#) ist ein Flussdiagramm, welches die Verarbeitung der Court Neighbor-Routine nach einer Ausführungsform veranschaulicht.
- [0049] [Fig. 28](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungskantensuchanrufabwicklungs-Routine nach einer Ausführungsform veranschaulicht.
- [0050] [Fig. 29](#) ist ein Flussdiagramm, welches die Verarbeitung der Verbindungskantensuchantwortabwicklungs-Routine nach einer Ausführungsform veranschaulicht.
- [0051] [Fig. 30](#) ist ein Flussdiagramm, welches die Verarbeitung der Übertragungs-Routine nach einer Ausführungsform veranschaulicht.
- [0052] [Fig. 31](#) ist ein Flussdiagramm, welches die Verarbeitung der Nachrichtenerfassungs-Routine nach einer Ausführungsform veranschaulicht.

[0053] [Fig. 32](#) ist ein Flussdiagramm, welches die Verarbeitung der Zustandsprüfnachrichtabwicklungs-Routine nach einer Ausführungsform veranschaulicht.

[0054] [Fig. 33](#) ist ein Flussdiagramm, welches die Verarbeitung der Zustandsbehebungsanweisungsabwicklungs-Routine nach einer Ausführungsform veranschaulicht.

[0055] [Fig. 34](#) ist ein Flussdiagramm, welches die Verarbeitung der Zustandsnachprüfabwicklungs-Routine nach einer Ausführungsform veranschaulicht.

[0056] [Fig. 35](#) ist ein Blockdiagramm, welches Komponenten des Auktionssystems nach einer Ausführungsform zeigt.

[0057] [Fig. 36](#) ist ein Blockdiagramm, welches die Komponenten eines teilnehmenden Computers nach einer Ausführungsform veranschaulicht.

[0058] [Fig. 37](#) ist ein Blockdiagramm, welches eine Anzeige aktueller Auktionen verdeutlicht. Fenster **300** wird von der Anzeigestatus-Routine angezeigt.

[0059] [Fig. 38](#) ist ein Diagramm, welches die Anzeige des gesamten auktionsspezifischen Fensters veranschaulicht.

[0060] [Fig. 39](#) ist ein Flussdiagramm der Routine zum Anfordern des aktuellen Status der Auktionen.

[0061] [Fig. 40](#) ist das Flussdiagramm der Routine, die eine angeforderte Nachricht über den aktuellen Status empfängt.

[0062] [Fig. 41](#) ist ein Flussdiagramm der Routine, die die Nachricht über den aktuellen Status empfängt.

[0063] [Fig. 42](#) ist ein Flussdiagramm, welches die Verarbeitung der Routine zur Abgabe eines Gebots nach einer Ausführungsform veranschaulicht.

[0064] [Fig. 43](#) ist ein Flussdiagramm, welches die Verarbeitung der Routine zum Empfangen der Gebotsnachricht nach einer Ausführungsform zeigt.

[0065] [Fig. 44](#) ist ein Flussdiagramm, welches eine Routine zeigt, die das Ablaufen der laufenden Zeitschaltuhr verarbeitet.

[0066] [Fig. 45](#) ist ein Flussdiagramm, welches eine Routine veranschaulicht, die eine empfangene Nachricht verarbeitet.

[0067] [Fig. 46](#) ist ein Flussdiagramm, welches eine Routine veranschaulicht, die das Ablaufen der abgelauenen Zeitschaltuhr verarbeitet.

[0068] [Fig. 47](#) ist ein Blockdiagramm, welches eine Routine darstellt, die eine empfangene Nachricht verarbeitet.

[0069] [Fig. 48](#) ist ein Flussdiagramm, welches einen Auktionsagenten nach einer Ausführungsform darstellt.

DETAILLIERTE BESCHREIBUNG

[0070] Es wird ein Übertragungsverfahren geschaffen, bei dem ein Übertragungskanal auf einem Punkt-zu-Punkt-Kommunikationsnetzwerk liegt. Das Übertragen einer Nachricht über den Übertragungskanal ist eigentlich ein Multicast zu jenen Computern des Netzwerks, die aktuell an den Übertragungskanal angeschlossen sind. In einer Ausführungsform erzeugt das Übertragungsverfahren einen logischen Übertragungskanal, an den die Host-Computer über die Ausführungsprozesse angeschlossen werden können. Jeder Computer, der an den Übertragungskanal angeschlossen ist, kann Nachrichten zu dem Übertragungskanal schicken und Nachrichten von ihm empfangen. Jeder Computer, der an den Übertragungskanal angeschlossen ist, empfängt sämtliche Nachrichten, die während der Verbindung gesendet werden. Der logische Übertragungskanal wird mithilfe eines dazugehörigen Netzwerkssystems (z.B. das Internet) implementiert, welches es jedem an das dazugehörige Netzwerkssystem angeschlossenen Computer ermöglicht, Nachrichten an jeden an-

deren angeschlossenen Computer mithilfe der jeweiligen Computeradresse zu senden. Folglich erzeugt das Übertragungsverfahren mithilfe eines dazugehörigen Netzwerksystems, welches Nachrichten von Punkt-zu-Punkt sendet, wirksam einen Übertragungskanal.

[0071] Das Übertragungsverfahren liegt auf dem zugrunde liegenden Netzwerksystem mit einem Graphen von Punkt-zu-Punkt-Verbindungen (d.h. Kanten) zwischen Host-Computern (d.h. Knoten), wodurch der Übertragungskanal implementiert wird. In einer Ausführungsform ist jeder Computer an vier weitere Computer angeschlossen, die als Nachbarn bezeichnet werden. (Eigentlich wird ein Prozess, der an einem Computer ausgeführt wird, mit vier weiteren Prozessen verbunden, die auf diesem oder vier anderen Computern ablaufen.) Um eine Nachricht zu übertragen, sendet der veranlassende Computer die Nachricht mithilfe der Punkt-zu-Punkt-Verbindungen zu jedem seiner Nachbarn. Jeder Computer, der die Nachricht empfängt, sendet die Nachricht anschließend über die Punkt-zu-Punkt-Verbindungen zu seinen drei anderen Nachbarn. Auf diese Weise wird die Nachricht über das zugrunde liegende Netzwerk zu jedem Computer geleitet, wodurch die Übertragung der Nachricht zu jedem Computer über einen logischen Übertragungskanal erfolgt. Ein Graph, bei dem jeder Knoten mit vier weiteren Knoten verbunden ist, wird als 4-regulär bezeichnet. Die Verwendung eines e-regulären Graphen bedeutet, dass ein Computer nur dann von dem Übertragungskanal getrennt werden würde, wenn alle vier Verbindungen zu seinen Nachbarn ausfallen. Der von dem Übertragungsverfahren genutzte Graph hat weiterhin die Eigenschaft, dass ein Defekt der vier Computer erforderlich wäre, um den Graphen in disjunkte Teilgraphen zu unterteilen, d.h. in zwei separate Übertragungskanäle. Diese Eigenschaft wird als 4-verbunden bzw. 4-zusammenhängend bezeichnet. Folglich ist der Graph sowohl 4-regulär als auch 4-zusammenhängend.

[0072] [Fig. 1](#) bildet einen Graphen ab, der 4-regulär und 4-zusammenhängend ist und den Übertragungskanal darstellt. Jeder der neun Knoten A–I stellt einen Computer dar, der an den Übertragungskanal angeschlossen ist, und jede der Kanten stellt eine „Kanten“-Verbindung zwischen zwei Computern des Übertragungskanals dar. Die für das Übertragen einer Nachricht zu jedem Computer auf dem Übertragungskanal erforderliche Zeit hängt von der Geschwindigkeit der Verbindungen zwischen den Computern und der Anzahl der Verbindungen zwischen dem Ausgangscomputer und jedem der anderen Computer auf dem Übertragungskanal ab. Die Mindestanzahl der Verbindungen, die eine Nachricht benötigen würde, um jedes Paar Computer zu überbrücken, ist der „Abstand“ zwischen den Computern (d.h. der kürzeste Weg zwischen den beiden Knoten des Graphen). Z.B. beträgt der Abstand zwischen den Computern A und F eins, da der Computer A direkt mit dem Computer F verbunden ist. Der Abstand zwischen den Computern A und B beträgt zwei, da es keine direkte Verbindung zwischen dem Computer A und B gibt, sondern der Computer F direkt mit dem Computer B verbunden ist. Folglich würde eine Nachricht, die von dem Computer A ausgeht, direkt zu Computer F gesendet werden und anschließend vom Computer F zu Computer B. Der Höchstabstand zwischen den Computern ist der „Durchmesser“ des Übertragungskanals. Der Durchmesser des Übertragungskanals, der in [Fig. 1](#) dargestellt ist, beträgt zwei. D.h., eine Nachricht, die von einem beliebigen Computer gesendet wird, würde nicht mehr als zwei Verbindungen überqueren, um jeden anderen Computer zu erreichen. [Fig. 2](#) bildet einen Graphen ab, der 20 Computer darstellt, die an einen Übertragungskanal angeschlossen sind. Der Durchmesser dieses Übertragungskanals beträgt 4. Konkret enthält der kürzeste Weg zwischen den Computern 1 und 3 vier Verbindungen (1-12, 12-15, 15-18 und 18-3).

[0073] Das Übertragungsverfahren umfasst (1) das Verbinden der Computer mit dem Übertragungskanal (d.h. Erstellen des Graphen), (2) das Senden der Nachrichten über den Übertragungskanal (d.h. Senden über den Graphen) und (3) das Trennen der Computer vom Übertragungskanal, der den Graphen bildet (d.h. Zerlegen des Graphen).

Erstellen des Graphen

[0074] Für eine Verbindung mit dem Übertragungskanal lokalisiert zuerst der Computer, der die Verbindung sucht, einen Computer, der gegenwärtig vollständig an den Übertragungskanal angeschlossen ist, und stellt anschließend eine Verbindung mit vier der Computer her, die bereits mit dem Übertragungskanal verbunden sind. (Es wird davon ausgegangen, dass bereits mindestens vier Computer an den Übertragungskanal angeschlossen sind. Wenn weniger als fünf Computer angeschlossen sind, kann der Übertragungskanal kein 4-regulärer Graph sein. In einem solchen Fall wird der Übertragungskanal als im „Small Regime“ [kleinen Betrieb] befindlich angesehen. Das Übertragungsverfahren für das Small Regime wird nachstehend genauer beschrieben. Wenn fünf oder mehr Computer angeschlossen sind, befindet sich der Übertragungskanal in dem so genannten „Large Regime“ [großen Betrieb]. Die vorliegende Beschreibung basiert auf der Annahme, dass sich der Übertragungskanal im Large Regime befindet, wenn nicht anderweitig angegeben.) Folglich umfasst der Prozess des Verbindens mit dem Übertragungskanal das Lokalisieren des Übertragungskanals, das Identifi-

zieren der Nachbarn für den anzuschließenden Computer und danach das Verbinden jedes festgestellten Nachbarn. Jeder Computer hat einen oder mehrere „Portal-Computer“, durch die der Computer den Übertragungskanal lokalisieren kann. Ein suchender Computer lokalisiert den Übertragungskanal, indem er sich so lange mit den Portal-Computern in Verbindung setzt, bis er einen findet, der gegenwärtig vollständig an den Übertragungskanal angeschlossen ist. Anschließend leitet der gefundene Portal-Computer die Identifizierung von vier Computern (d.h. die Nachbarn des suchenden Computers) weiter, mit denen der suchende Computer verbunden werden soll. Jeder dieser vier Computer wirkt daraufhin mit dem suchenden Computer zusammen, um die Verbindung des suchenden Computers mit dem Übertragungskanal herzustellen. Ein Computer, der den Prozess des Lokalisierens eines Portal-Computers in Gang gesetzt hat, aber noch keinen Nachbarn hat, befindet sich im „Verbindungssuchzustand“. Ein Computer, der an wenigstens einen Nachbarn angeschlossen ist, jedoch noch nicht an vier Nachbarn, befindet sich im „teilweise verbundenen Zustand“. Ein Computer, der aktuell an vier Nachbarn verbunden ist oder zuvor war, ist bzw. war im „vollständig verbundenen Zustand“.

[0075] Da der Übertragungskanal ein 4-regulärer Graph ist, ist jeder der identifizierten Computer bereits mit vier Computern verbunden. Folglich müssen einige Verbindungen zwischen Computern aufgehoben werden, sodass der suchende Computer mit vier Computern verbunden werden kann. Bei einer Ausführungsform identifiziert das Übertragungsverfahren zwei Paare von Computern, die aktuell miteinander verbunden sind. Jedes dieser Computerpaare unterbricht die Verbindung zwischen ihnen, und anschließend stellt jeder der vier Computer (zwei aus jedem Paar) eine Verbindung zu dem suchenden Computer her. Die [Fig. 3A](#) und [Fig. 3B](#) veranschaulichen den Prozess, bei dem ein neuer Computer Z eine Verbindung mit dem Übertragungskanal herstellt. [Fig. 3A](#) zeigt den Übertragungskanal, bevor der Computer Z angeschlossen ist. Die Computerpaare B und E sowie C und D sind die beiden Paare, die als Nachbarn für den neuen Computer Z identifiziert werden. Die Verbindungen zwischen jedem dieser Paare wird unterbrochen, und eine Verbindung zwischen dem Computer Z und jedem der Computer B, C, D und E wird hergestellt, wie durch [Fig. 3B](#) angegeben. Der Prozess des Aufhebens der Verbindung zwischen zwei Nachbarn und der erneuten Verbindung von jedem der früheren Nachbarn mit einem anderen Computer wird als „Edge Pinning“ bezeichnet, da man davon ausgehen kann, dass die Kante (Edge) zwischen zwei Knoten gestreckt und an einen neuen Knoten geheftet (Pinning) wird.

[0076] Jeder Computer, der mit dem Übertragungskanal verbunden ist, verfügt über fünf Kommunikations-Ports zur Datenübertragung mit anderen Computern. Vier der Ports werden als „interne“ Ports bezeichnet, da sie die Ports sind, über die die Nachrichten der Übertragungskanäle gesendet werden. Die Verbindungen zwischen den internen Ports der Nachbarn werden als „interne“ Verbindungen bezeichnet. Folglich bilden die internen Verbindungen des Übertragungskanals den 4-regulären und 4-zusammenhängenden Graphen. Der fünfte Port wird als „externer“ Port bezeichnet, da er zum Senden von nicht zu übertragenden Nachrichten zwischen zwei Computern verwendet wird. Nachbarn können nicht für die Übertragung vorgesehenen Nachrichten entweder durch ihre internen Ports der Verbindung oder durch ihre externen Ports senden. Ein suchender Computer verwendet die externen Ports, wenn er einen Portal-Computer lokalisiert.

[0077] Bei einer Ausführungsform stellt das Übertragungsverfahren die Computerverbindungen mithilfe des TCP/IP-Protokolls, das ein Punkt-zu-Punkt-Protokoll ist, als zugrunde liegendes Netzwerk her. Das TCP/IP-Protokoll sorgt für eine zuverlässige und ordnungsgemäße Bereitstellung von Nachrichten zwischen den Computern. Das TCP/IP-Protokoll stellt jedem Computer einen „Port-Platz“ zur Verfügung, der von allen Prozessen gemeinsam genutzt wird, die auf jenem Computer ausgeführt werden können. Die Ports sind mit Zahlen von 0 bis 65 535 gekennzeichnet. Die ersten 2 056 Ports sind für spezielle Anwendungen (z.B. Port 80 für HTTP-Nachrichten) reserviert). Die übrigen Ports sind Benutzer-Ports, die für alle Prozesse zur Verfügung stehen. Bei einer Ausführungsform kann eine Gruppe von Port-Nummern für die Nutzung durch den Computer reserviert werden, der mit dem Übertragungskanal verbunden ist. Bei einer alternativen Ausführungsform werden die Port-Nummern dynamisch von jedem Computer identifiziert. Jeder Computer identifiziert dynamisch einen zur Verfügung stehenden Port, der als Call-in-Port (Abruf-Port) verwendet werden soll. Dieser Call-in-Port wird zur Herstellung von Verbindungen mit dem externen Port und den internen Ports verwendet. Jeder Computer, der mit dem Übertragungskanal verbunden ist, kann über seinen externen Port nicht für die Übertragung gedachte Nachrichten empfangen. Ein suchender Computer versucht, die Port-Nummern der Portal-Computer so lange zu „wählen“, bis ein Portal-Computer einen Anruf auf seinem Call-in-Port „beantwortet“. Ein Portal-Computer antwortet, wenn er an den Übertragungskanal angeschlossen ist oder versucht, mit ihm eine Verbindung herzustellen, und dessen Call-in-Port gewählt wird. (Bei der vorliegenden Beschreibung wird die Metapher eines Telefons zur Beschreibung der Verbindungen verwendet.) Wenn ein Computer einen Anruf auf seinem Call-in-Port empfängt, überträgt er den Anruf zu einem anderen Port. Folglich kommuniziert der suchende Computer eigentlich über jene Übertragung mit dem Port, der den externen Port bildet. Der Anruf wird so übertragen, dass andere Computer Anrufe zu jenem Computer über den Call-in-Port tätigen können. Der suchende Computer kommuniziert anschließend über den externen Port und bittet den Portalcomputer um

Unterstützung bei dem Verbinden des suchenden Computers mit dem Übertragungskanal. Der suchende Computer könnte die Call-in-Port-Nummer des Portal-Computers identifizieren, indem er nacheinander jeden Port in der Reihenfolge der Port-Nummern anwählt. Wie nachstehend erörtert, wendet das Übertragungsverfahren einen Hashing-Algorithmus an, um die Reihenfolge der Port-Nummern auszuwählen, was eine verbesserte Leistung nach sich ziehen kann.

[0078] Ein suchender Computer könnte sich mit dem Übertragungskanal verbinden, indem er eine Verbindung entweder zu den Computern herstellt, die direkt mit dem gefundenen Portal-Computer verbunden sind oder die direkt mit einem von dessen Nachbarn verbunden sind. Ein mögliches Problem bei einem derartigen Schema zum Erkennen der Nachbarn des suchenden Computers besteht darin, dass sich der Durchmesser des Übertragungskanals vergrößern kann, wenn jeder suchende Computer denselben gefundenen Portal-Computer nutzt und direkt über jenen gefundenen Portal-Computer eine Verbindung zu dem Übertragungskanal herstellt. Vom Konzept her verlängert sich der Graph in die Richtung, in der die neuen Knoten hinzugefügt werden. In den [Fig. 4A–Fig. 4C](#) ist dieses mögliche Problem dargestellt. [Fig. 4A](#) veranschaulicht den Übertragungskanal aus [Fig. 1](#) mit einem hinzugefügten Computer. Der Computer J wurde durch Edge Pinning der Kanten C-D und E-H mit dem Computer J an den Übertragungskanal angeschlossen. Der Durchmesser dieses Übertragungskanals beträgt noch immer zwei. [Fig. 4B](#) zeigt den Übertragungskanal aus [Fig. 4A](#) mit einem hinzugefügten Computer. Der Computer K wurde an den Übertragungskanal angeschlossen, indem die Kanten E-J und B-C mittels Edge Pinning mit dem Computer K verbunden wurden. Der Durchmesser dieses Übertragungskanals beträgt drei, da der kürzeste Weg von Computer G zu Computer K durch die Kanten G-A, A-E und E-K verläuft. Weiterhin zeigt [Fig. 4C](#) den Übertragungskanal aus [Fig. 4A](#) mit einem hinzugefügten Computer. Der Computer K wurde durch Edge Pinning der Kanten D-G und E-J an den Computer K mit dem Übertragungskanal verbunden. Der Durchmesser dieses Übertragungskanals beträgt jedoch noch immer zwei. Somit hat die Auswahl der Nachbarn einen Einfluss auf den Durchmesser des Übertragungskanals. Um den Durchmesser so gering wie möglich zu halten, nutzt das Übertragungsverfahren ein Zufallsauswahlverfahren zum Identifizieren der vier Nachbarn eines Computers im Verbindungssuchzustand. Bei dem Zufallsauswahlverfahren werden meist die Verbindungen zu neuen suchenden Computern über die gesamten Computer des Übertragungskanals hinweg verteilt, was zu kleineren Gesamtdurchmessern führen kann.

Übertragen durch den Graphen

[0079] Wie oben beschrieben, kann jeder Computer, der mit dem Übertragungskanal verbunden ist, Nachrichten zu dem Übertragungskanal senden und sämtliche Nachrichten, die über den Übertragungskanal gesendet werden, empfangen. Der Computer, von dem eine zu übertragende Nachricht ausgeht, sendet diese Nachricht mithilfe der internen Verbindungen zu jedem seiner vier Nachbarn. Wenn ein Computer eine Übertragungsnachricht von einem Nachbarn empfängt, sendet er die Nachricht zu seinen drei anderen Nachbarn. Jeder Computer auf dem Übertragungskanal wird somit, ausgenommen von dem Ursprungscomputer, eine Kopie jeder Übertragungsnachricht von jedem seiner vier Nachbarn empfangen. Jedoch sendet jeder Computer nur die erste Kopie der Nachricht, die er empfängt, zu seinen Nachbarn und lässt anschließend empfangene Kopien außer Acht. Folglich beträgt die Gesamtanzahl der Kopien einer Nachricht, die zwischen den Computern gesendet wird $3N + 1$, wobei N die Anzahl der mit dem Übertragungskanal verbundenen Computer ist. Jeder Computer sendet drei Kopien der Nachricht, ausgenommen davon ist der Ursprungscomputer, der vier Kopien der Nachricht sendet.

[0080] Die Redundanz der Nachrichtensendung trägt dazu bei, die Gesamtzuverlässigkeit des Übertragungskanals sicherzustellen. Da jeder Computer über vier Verbindungen zu dem Übertragungskanal verfügt, haben seine Nachbarn bei Ausfall eines Computers während des Sendens einer Nachricht drei andere Verbindungen zur Verfügung, über die sie Kopien der Übertragungsnachricht empfangen. Auch wenn die interne Verbindung zwischen zwei Computern langsam ist, stehen jedem Computer drei weitere Verbindungen zur Verfügung, über die er eine Kopie jeder Nachricht schneller empfangen kann.

[0081] Jeder Computer, der eine Nachricht verschickt, nummeriert seine Nachrichten fortlaufend. Aufgrund der Dynamik des Übertragungskanals und der Vielzahl möglicher Verbindungswege zwischen den Computern können die Nachrichten aber in anderer Reihenfolge empfangen werden. Z.B. kann der Abstand zwischen einem Ursprungscomputer und einem bestimmten Empfangscomputer vier betragen. Nach dem Senden der ersten Nachricht können der Ursprungscomputer und der Empfangscomputer Nachbarn werden, wodurch sich der Abstand zwischen ihnen auf eins ändert. Die erste Nachricht muss möglicherweise einen Abstand von vier überwinden, um den Empfangscomputer zu erreichen. Die zweite Nachricht hat lediglich einen Abstand von eins zurückzulegen. Folglich ist es möglich, dass die zweite Nachricht den Empfangscomputer vor der ersten Nachricht erreicht.

[0082] Wenn sich der Übertragungskanal im stabilen Zustand befindet (es werden keine Computer mit dem Übertragungskanal verbunden oder von ihm getrennt), stellt die falsche Reihenfolge der Nachrichten kein Problem dar, da jeder Computer letztendlich beide Nachrichten empfangen wird und die Nachrichten solange zurückhalten kann, bis alle früher in Auftrag gegebenen Nachrichten empfangen sind. Wenn sich jedoch der Übertragungskanal nicht in einem stabilen Zustand befindet, kann dieses Problem auftreten. Konkret kann ein Computer eine Verbindung zum Übertragungskanal aufbauen, nachdem die zweite Nachricht bereits empfangen und von seinen neuen Nachbarn weitergeleitet worden ist. Wenn ein neuer Nachbar schließlich die erste Nachricht empfängt, sendet er die Nachricht zu dem neu verbundenen Computer. Somit wird der neu verbundene Computer die erste Nachricht empfangen, die zweite Nachricht jedoch nicht. Wenn der neu angeschlossene Computer die Nachrichten in der richtigen Reihenfolge verarbeiten muss, wartet er vergeblich auf die zweite Nachricht.

[0083] Eine Lösung dieses Problems besteht darin, dass jeder Computer sämtliche Nachrichten, die er erhält, solange in die Warteschlange stellt, bis er sie in der richtigen Reihenfolge an seinen Nachbarn schicken kann. Allerdings wird durch diese Lösung meist die Ausbreitung von Nachrichten durch die Computer des Übertragungskanal verlangsamt. Eine weitere Lösung, die vielleicht weniger Einfluss auf die Ausbreitungsgeschwindigkeit hat, besteht darin, die Nachrichten nur bei den Computern in die Warteschlange zu stellen, die Nachbarn der neu angeschlossenen Computer sind. Jeder bereits angeschlossene Nachbar würde die Nachrichten so wie er sie empfängt an seine andere Nachbarn weiterleiten, die nicht neu angeschlossen sind, jedoch nicht zu dem neu angeschlossenen Nachbarn. Der bereits angeschlossene Nachbar würde nur dann Nachrichten von jedem Ursprungscomputer zu dem neu angeschlossenen Computer weiterleiten, wenn er gewährleisten kann, dass keine Lücken in den Nachrichten von dem Ursprungscomputer auftreten. Bei einer Ausführungsform kann der bereits angeschlossene Nachbar die höchste Nummer in der Sequenz der bereits empfangenen und weitergeleiteten Nachrichten von jedem Ursprungscomputer verfolgen. Der bereits angeschlossene Computer sendet lediglich die Nachrichten mit größerer Nummer von den Ursprungscomputern zu dem neu angeschlossenen Computer. Nachdem alle Nachrichten mit niedrigerer Nummer von sämtlichen Ursprungscomputern empfangen worden sind, kann der bereits angeschlossene Computer den neu angeschlossenen Computer genau wie seine anderen Nachbarn behandeln und jede Nachricht so wie sie empfangen wird einfach weiterleiten. Bei einer anderen Ausführungsform kann jeder Computer die Nachrichten in eine Warteschlange stellen und jene Nachrichten nur dann zu dem neu angeschlossenen Computer weiterleiten, wenn die Lücken zu füllen sind. Z.B. kann ein Computer die Nachrichten 4 und 5 empfangen und dann die Nachricht 3. In einem solchen Fall würde der bereits angeschlossene Computer die Nachrichten 4 und 5 in der Warteschlange weiterleiten. Wenn die Nachricht 3 schließlich empfangen worden ist, sendet der bereits angeschlossene Computer die Nachrichten 3, 4 und 5 zu dem neu angeschlossenen Computer. Wenn die Nachrichten 4 und 5 vor der Nachricht 3 zu dem neu angeschlossenen Computer gesendet wurden, dann würde der neu angeschlossene Computer die Nachrichten 4 und 5 verarbeiten und die Nachricht 3 außer Acht lassen. Da der bereits angeschlossene Computer die Nachrichten 4 und 5 in die Warteschlange stellt, ist der neu angeschlossene Computer in der Lage, Nachricht 3 zu verarbeiten. Es ist möglich, dass ein neu angeschlossener Computer einen Satz von Nachrichten von einem Ursprungscomputer über einen Nachbarn empfängt und anschließend einen anderen Satz Nachrichten von demselben Ursprungscomputer über einen anderen Nachbarn empfängt. Wenn der zweite Satz Nachrichten eine Nachricht enthält, die in der Reihenfolge vor den Nachrichten des zuerst empfangenen Satzes kommt, kann der neu angeschlossene Computer diese Nachricht, die in der Reihenfolge früher kommt, ignorieren, wenn der Computer die in der Reihenfolge weiter hinten stehenden Nachrichten bereits verarbeitet hat.

Zerlegen des Graphen

[0084] Ein angeschlossener Computer wird entweder geplant oder ungeplant vom Übertragungskanal getrennt. Wenn ein Computer geplant getrennt wird, sendet er eine Abschaltnachricht zu jedem seiner vier Nachbarn. Die Abschaltnachricht enthält eine Liste, die die vier Nachbarn des sich abschaltenden Computers identifiziert. Wenn ein Nachbar die Abschaltnachricht empfängt, versucht er, sich mit einem der Computer auf der Liste zu verbinden. Bei einer Ausführungsform versucht der erste Computer in der Liste, eine Verbindung zu dem zweiten Computer in der Liste herzustellen, und der dritte Computer in der Liste versucht, sich mit dem vierten Computer in der Liste zu verbinden. Wenn ein Computer keine Verbindung aufbauen kann (z.B. sind der erste und der zweite Computer bereits angeschossen), dann versuchen die Computer möglicherweise, sich in verschiedenen anderen Kombinationen miteinander zu verbinden. Wenn keine Verbindungen hergestellt werden können, sendet jeder Computer eine Nachricht, dass er eine Verbindung mit einem anderen Computer aufbauen muss. Empfängt ein Computer mit einem zur Verfügung stehenden internen Port die Nachricht, kann er eine Verbindung mit dem Computer aufbauen, der die Nachricht sendet. Die [Fig. 5A](#), [Fig. 5D](#) veranschaulichen das Trennen eines Computers von dem Übertragungskanal. [Fig. 5A](#) zeigt das Trennen eines

Computers von dem Übertragungskanal in geplanter Art und Weise. Wenn der Computer H entscheidet, die Verbindung zu unterbrechen, sendet er die Liste seiner Nachbarn an jeden seiner Nachbarn (Computer A, E, F und I) und trennt danach die Verbindung zu jedem seiner Nachbarn. Wenn die Computer A und I die Nachricht empfangen, stellen sie eine Verbindung zwischen sich her, wie durch die Strichlinie angegeben, Gleiches gilt für die Computer E und F.

[0085] Wenn sich ein Computer ungeplant abschaltet, z.B. infolge eines Stromausfalls, erkennen die mit dem abgeschalteten Computer verbundenen Nachbarn die Verbindungstrennung, wenn jeder versucht, seine nächste Nachricht an den jetzt abgeschalteten Computer zu senden. Jeder frühere Nachbar des abgeschalteten Computers erkennt, dass eine Verbindung fehlt (d.h., dass er ein Loch, bzw. einen leeren Port hat). Wenn ein angeschlossener Computer erkennt, dass einer seiner Nachbarn jetzt abgeschaltet ist, sendet er eine Portverbindungsanforderung zum Übertragungskanal, die anzeigt, dass ein interner Port eine Verbindung benötigt. Die Portverbindungsanforderung identifiziert den Call-in-Port des anfordernden Computers. Wenn ein angeschlossener Computer, dem ebenfalls eine Verbindung fehlt, die Verbindungsanforderung empfängt, kommuniziert er mit dem anfordernden Computer über den externen Port, um eine Verbindung zwischen den beiden Computern herzustellen. [Fig. 5B](#) zeigt das Abschalten eines Computers von dem Übertragungskanal in ungeplanter Weise. In der Darstellung ist der Computer H ungeplant abgeschaltet worden. Wenn jeder seiner Nachbarn, die Computer A, E, F und I, diese Abschaltung erkennen, sendet jeder Nachbar eine Portverbindungsanforderung, die anzeigt, dass ein leerer Port gefüllt werden muss. Wie durch die Strichlinien angegeben, reagieren die Computer F und I und die Computer A und E auf die gegenseitigen Anforderungen und stellen eine Verbindung her.

[0086] Es ist möglich, dass eine geplante oder ungeplante Abschaltung dazu führen kann, dass zwei Nachbarn jeweils einen leeren internen Port haben. Da sie bereits Nachbarn sind, sind sie in diesem Fall bereits miteinander verbunden und können ihre leeren Ports nicht durch Verbindung miteinander füllen. Ein solcher Zustand wird als Zustand mit "Nachbarn mit leeren Ports" bezeichnet. Wie oben beschrieben, sendet jeder Nachbar eine Portverbindungsanforderung, wenn er einen leeren Port bei sich entdeckt. Wenn ein Nachbar die Portverbindungsanforderung von dem anderen Nachbarn empfängt, erkennt er, dass sein Nachbar ebenfalls einen leeren Port hat. Ein solcher Zustand kann eintreten, wenn sich der Übertragungskanal im „Small Regime“ befindet. Dieser Zustand kann nur im „Large Regime“ korrigiert werden. Im „Small Regime“ hat jeder Computer weniger als vier Nachbarn. Um diesen Zustand im „Large Regime“ zu erkennen, was bei nicht einsetzender Behebung ein Problem wäre, erkennt der erste Nachbar, der die Portverbindungsanforderung empfängt, diesen Zustand und sendet eine Zustandsprüfnachricht an den anderen Nachbarn. Die Zustandsprüfnachricht enthält eine Liste der Nachbarn des absendenden Computers. Wenn der empfangene Computer die Liste empfängt, vergleicht er sie mit seiner eigenen Nachbarliste. Unterscheiden sich die Listen voneinander, ist der Zustand im „Large Regime“ eingetreten und muss behoben werden. Dazu sendet der empfangene Computer eine Zustandsbehebungsanforderung an einen der Nachbarn des absendenden Computers, der nicht bereits ein Nachbar des empfangenden Computers ist. Wenn der Computer die Zustandsbehebungsanforderung empfängt, trennt er sich von seinen Nachbarn (bis auf jenen, der in diesen Zustand eingezogen ist) und stellt eine Verbindung zu dem Computer her, der die Zustandsbehebungsanforderung sendet. Somit wird bei einem der ursprünglichen Nachbarn, die in diesen Zustand einbezogen waren, ein Port gefüllt. Allerdings müssen noch immer zwei Computer miteinander verbunden werden, der andere ursprünglichen Nachbar und der Computer, der jetzt von dem Computer getrennt ist, welcher die Zustandsbehebungsanforderung empfangen hat. Diese beiden Computer senden Portverbindungsanforderungen aus. Wenn jene beiden Computer keine Nachbarn sind, werden sie bei Empfang der Anforderungen eine Verbindung miteinander eingehen. Sind diese beiden Computer jedoch Nachbarn, wiederholen sie diesen Zustandsbehebungsprozess solange, bis zwei Nichtnachbarn eine Verbindung benötigen.

[0087] Es ist möglich, dass die beiden ursprünglichen Nachbarn in diesem Zustand die gleiche Gruppe von Nachbarn haben. Wenn der Nachbar, der die Zustandsnachprüfnachricht empfängt, feststellt, dass die Gruppen der Nachbarn identisch sind, sendet er eine Zustandsnachprüfnachricht an einen der Nachbarn, bei dem es sich nicht um jenen handelt, der den gleichen Zustand aufweist. Wenn der Computer die Zustandsnachprüfnachricht empfängt, stellt er fest, ob er dieselbe Gruppe von Nachbarn hat wie der sendende Computer. Falls ja, befindet sich der Übertragungskanal im Small Regime, und der Zustand bildet kein Problem. Sind die Gruppen der Nachbarn verschieden, sendet der Computer, der die Zustandsnachprüfnachricht empfangen hat, eine Zustandsprüfnachricht an die ursprünglichen Nachbarn in dem Zustand. Der Computer, der die Zustandsprüfnachricht empfängt, weist einen seiner Nachbarn an, eine Verbindung zu einem der ursprünglichen Nachbarn in diesem Zustand herzustellen, indem eine Zustandsbehebungsnachricht gesendet wird. Somit wird bei einem der ursprünglichen Nachbarn in diesem Zustand der Port aufgefüllt.

[0088] [Fig. 5C](#) zeigt die Nachbarn mit leeren Ports. In dieser Abbildung wurde der Computer H ungeplant abgeschaltet, doch die Computer F und I reagierten auf die Portverbindungsanforderung des anderen und sind jetzt miteinander verbunden. Die anderen früheren Nachbarn des Computers H, die Computer A und E, sind bereits Nachbarn, wodurch sie in den Zustand mit leeren Ports versetzt werden. Im vorliegenden Beispiel empfing der Computer E die Portverbindungsanforderung von Computer A, erkannte den möglichen Zustand und sendete (da sie über die interne Verbindung Nachbarn sind) eine Zustandsprüfnachricht mit einer Liste seiner Nachbarn zum Computer A. Als der Computer A die Liste empfing, erkannte er, dass Computer E eine andere Gruppe von Nachbarn hat (der Übertragungskanal befindet sich im Large Regime). Computer A wählte Computer D, der Nachbar von Computer E ist, und sendete eine Zustandsbehebungsanforderung. Als der Computer D die Zustandsbehebungsanforderung empfing, schaltete er sich von einem seiner Nachbarn ab (nicht von Computer E), bei diesem Beispiel von Computer G. Daraufhin stellte der Computer D eine Verbindung zum Computer A her. [Fig. 5D](#) stellt zwei Computer dar, die keine Nachbarn sind, jetzt aber leere Ports aufweisen. Die Computer E und G haben jetzt leere Ports und sind aktuell keine Nachbarn. Daher können die Computer E und G eine Verbindung zueinander aufbauen.

[0089] [Fig. 5E](#) und [Fig. 5F](#) zeigen weiterhin die Nachbarn mit leeren Ports. [Fig. 5E](#) zeigt die Nachbarn mit leeren Ports im Small Regime. Wenn Computer E in diesem Beispiel ungeplant abgeschaltet wird, sendet daraufhin jeder Computer eine Portverbindungsanforderung, wenn er die Abschaltung erkennt. Empfängt Computer A die Portverbindungsanforderung von Computer B, erkennt er die Nachbarn mit leeren Ports und sendet eine Zustandsprüfnachricht an Computer B. Computer B erkennt, dass er die gleiche Gruppe von Nachbarn (Computer C und D) wie Computer A hat und sendet eine Zustandsnachprüfnachricht an Computer C. Computer C erkennt, dass sich der Übertragungskanal im Small Regime befindet, da er die gleiche Gruppe von Nachbarn hat wie die Computer A und B, woraufhin Computer C eine Nachricht senden kann, die anzeigt, dass sich der Übertragungskanal im Small Regime befindet.

[0090] [Fig. 5F](#) zeigt die Situation von [Fig. 5E](#) im Large Regime. Wie oben erörtert, empfing Computer C die Zustandsnachprüfnachricht von Computer D. In diesem Fall erkennt Computer C, dass sich der Übertragungskanal im Large Regime befindet, da er eine andere Gruppe Nachbarn hat als Computer B. Die sich vom Computer C und D nach oben erstreckenden Kanten geben Verbindungen zu anderen Computern an. Danach sendet Computer C eine Zustandsprüfnachricht an Computer B. Wenn Computer B die Zustandsprüfnachricht empfängt, sendet er eine Zustandsbehebungsnachricht an einen der Nachbarn von Computer C. Der Computer, der die Zustandsbehebungsnachricht empfängt, trennt sich von einem seiner Nachbarn, nicht vom Computer C, und versucht, eine Verbindung zu Computer B herzustellen, und der Nachbar, von dem er sich getrennt hat, versucht eine Verbindung zu Computer A herzustellen.

Portauswahl

[0091] Wie oben erwähnt, bezeichnet das TCP/IP-Protokoll Ports oberhalb der Nummer 2 056 als Benutzer-Ports. Das Übertragungsverfahren nutzt fünf Benutzer-Port-Nummern an jedem Computer: einen externen Port und vier interne Ports. Allgemein können die Benutzer-Ports nicht statisch einem Anwendungsprogramm zugeordnet werden, da andere Anwendungsprogramme, die auf demselben Computer laufen, nicht gleichzeitig zulässige Port-Nummern verwenden können. Deshalb weisen in einer Ausführungsform die an den Übertragungskanal angeschlossenen Computer ihre Port-Nummern dynamisch zu. Jeder Computer könnte einfach versuchen, den ungenutzten Port mit der niedrigsten Nummer an jenem Computer zu lokalisieren und diesen Port als Call-in-Port zu nutzen. Ein suchender Computer kennt jedoch nicht im Voraus die Call-in-Port-Nummer der Portal-Computer, wenn die Port-Nummern dynamisch zugewiesen werden. Folglich muss ein suchender Computer die Ports eines Portal-Computers anwählen, beginnend mit der niedrigsten Port-Nummer, wenn er den Call-in-Port eines Portal-Computers lokalisieren will. Wenn der Portal-Computer an den Übertragungskanal angeschlossen ist (oder versucht, eine Verbindung zu ihm herzustellen), dann würde der suchende Computer schließlich den Call-in-Port finden. Ist der Portal-Computer nicht angeschlossen, würde der suchende Computer schließlich jeden Benutzer-Port anwählen. Wenn jedes Anwendungsprogramm an einen Computer versuchen würde, die Port-Nummern von unten ausgehend zuzuweisen, hätte ein Portal-Computer zusätzlich am Ende für seinen Call-in-Port nur noch einen Port mit höherer Nummer, da viele der Port-Nummern im unteren Bereich von anderen Anwendungsprogrammen genutzt werden würden. Da das Einwählen eines Portes relativ langsam abläuft, würde der suchende Computer lange Zeit benötigen, um den Call-in-Port eines Portal-Computers zu lokalisieren. Um diese Zeit so gering wie möglich zu halten, nutzt das Übertragungsverfahren einen Port-Ordnungs-Algorithmus, der die Port-Nummern-Reihenfolge festlegt, die ein Portal-Computer nutzen sollte, wenn er einen verfügbaren Port für seinen Call-in-Port findet. Bei einer Ausführungsform wendet das Übertragungsverfahren einen Hashing-Algorithmus zum Identifizieren der Portreihenfolge an. Dieser Algorithmus verteilt die Reihenfolge der Port-Nummern vorzugsweise zufällig über den gesamten Nummernraum

der Benutzer-Ports und wählt jede Port-Nummer nur einmal aus. Jedes Mal, wenn der Algorithmus auf einem Computer für eine bestimmte Kanalart und Kanalinstanz ausgeführt wird, erzeugt er die gleiche Port-Reihenfolge. Wie nachstehend beschrieben wird, ist es möglich, dass ein Computer mit mehreren Übertragungskanälen verbunden wird, die durch den Kanaltyp und die Kanalinstanz eindeutig identifiziert sind. Dieser Algorithmus kann mit dem Kanaltyp und der Kanalinstanz versehen werden, um eine eindeutige Reihenfolge der Port-Nummern für jeden Übertragungskanal zu erzeugen. Somit wählt ein suchender Computer die Ports eines Portal-Computers in der gleichen Reihenfolge an wie der Portal-Computer beim Zuweisen seines Call-in-Ports.

[0092] Wenn viele Computer gleichzeitig versuchen, eine Verbindung zu einem Übertragungskanal über einen einzigen Portal-Computer herzustellen, können die Ports des Portal-Computers belegt sein, wenn sie von den suchenden Computern angerufen werden. Meist muss der suchende Computer einen besetzten Port mehrfach anwählen. Durch dieses wiederholte Anwählen kann das Lokalisieren eines Call-in-Ports wesentlich verlangsamt werden. Bei einer Ausführungsform kann jeder suchende Computer die ersten Port-Nummern, die von dem Hashing-Algorithmus erzeugt werden, neu ordnen. Z.B. könnte jeder suchende Computer die ersten acht Port-Nummern neu ordnen, die vom Hashing-Algorithmus erzeugt werden. Die willkürliche Reihenfolge könnte auch gewichtet werden, wobei die erste Port-Nummer, die von dem Hashing-Algorithmus erzeugt wird, mit 50%iger Sicherheit bei der Neuordnung an erster Stelle stehen würde, die zweite Port-Nummer mit 25%iger Wahrscheinlichkeit zuerst in der Neuordnung käme usw. Da die suchenden Computer verschiedene Reihenfolgen verwenden würden, verringert sich die Wahrscheinlichkeit, dass man auf einen besetzten Port stößt. Wenn beispielsweise die ersten acht Port-Nummern zufällig ausgewählt werden, ist es möglich, dass acht suchende Computer gleichzeitig Ports in unterschiedlichen Sequenzen anwählen, wodurch sich die Wahrscheinlichkeit verringert, dass ein besetzter Port angewählt wird.

Lokalisieren eines Portal-Computers

[0093] Jeder Computer, der sich an den Übertragungskanal anschließen kann, hat eine Liste von einem oder mehreren Portal-Computern, über die er sich mit dem Übertragungskanal verbinden kann. Bei einer Ausführungsform hat jeder Computer die gleiche Gruppe von Portal-Computern. Ein suchender Computer lokalisiert einen Portal-Computer, der an den Übertragungskanal angeschlossen ist, indem er nacheinander die Ports von jedem Portal-Computer in der von dem Algorithmus angegebenen Reihenfolge anwählt. Ein suchender Computer könnte den ersten Portal-Computer auswählen und anschließend all seine Ports anwählen, bis ein Call-in-Port eines Computers gefunden worden ist, der vollständig an den Übertragungskanal angeschlossen ist. Wenn kein Call-in-Port gefunden wird, wählt der suchende Computer den nächsten Portal-Computer aus und wiederholt den Prozess solange, bis ein Portal-Computer mit einem derartigen Call-in-Port gefunden worden ist. Ein Problem bei einem derartigen Suchverfahren besteht darin, dass sämtliche Benutzer-Ports jedes Portal-Computers solange angewählt werden, bis ein Portal-Computer gefunden worden ist, der vollständig an den Übertragungskanal angeschlossen ist. Bei einer anderen Ausführungsform wählt der suchende Computer eine Port-Nummer entsprechend dem Algorithmus aus und wählt anschließend jeden Portal-Computer auf dieser Port-Nummer an. Wenn kein annehmbarer Call-in-Port für den Übertragungskanal gefunden wird, wählt der suchende Computer anschließend die nächste Port-Nummer aus und wiederholt den Prozess. Da den Call-in-Ports wahrscheinlich Port-Nummern mit niedrigerer Zahl zugeordnet sind, wählt der suchende Computer zuerst die Port-Nummern, die am wahrscheinlichsten die Call-in-Ports des Übertragungskanals sind. Die suchenden Computer können eine maximale Suchtiefe aufweisen, bei der es sich um die Anzahl der Ports handelt, die beim Suchen eines vollständig angeschlossenen Portal-Computers angewählt wird. Wenn der suchende Computer seine Suchtiefe ausgeschöpft hat, dann ist entweder der Übertragungskanal noch nicht hergestellt, oder wenn der suchende Computer ebenfalls ein Portal-Computer ist, kann er den Übertragungskanal mit sich selbst als dem ersten vollständig angeschlossenen Computer aufbauen.

[0094] Lokalisiert ein suchender Computer einen Portal-Computer, der selbst nicht vollständig angeschlossen ist, stellen die beiden Computer keine Verbindung her, wenn sie einander zuerst lokalisieren, da der Übertragungskanal möglicherweise schon aufgebaut ist und über eine höhere Port-Nummer an einem anderen Portal-Computer zugänglich ist. Wenn die beiden suchenden Computer miteinander verbunden werden würden, entstünden zwei disjunkte Übertragungskanäle. Jeder suchende Computer kann seine Erfahrung bei dem Versuch, einen Portal-Computer zu lokalisieren, anderen suchenden Computern mitteilen. Konkret dann, wenn ein suchender Computer alle Portal-Computer bis zu einer Tiefe von acht durchsucht hat, kann der eine suchende Computer einem anderen suchenden Computer mitteilen, dass er bis zu einer Tiefe von acht gesucht hat. Wenn jener andere suchende Computer bis zu einer Tiefe von beispielsweise lediglich vier die Suche ausgeführt hat, kann er die Suche zwischen der Tiefe fünf bis acht auslassen, sodass er seine Suche auf einer Tiefe von neun fortsetzt.

[0095] Bei einer Ausführungsform kann jeder Computer eine andere Gruppe von Portal-Computern und eine unterschiedliche maximale Suchtiefe aufweisen. In einer solchen Situation ist es möglich, dass zwei disjunkte Übertragungskanäle gebildet werden, da ein suchender Computer einen Computer mit vollständig angeschlossenen Port auf höherer Tiefe nicht lokalisieren kann. Ebenso würden dann, wenn die Gruppe der Portal-Computer disjunkt ist, zwei separate Übertragungskanäle entstehen.

Erkennen der Nachbarn eines suchenden Computers

[0096] Wie oben beschrieben, werden die Nachbarn eines neu angeschlossenen Computers vorzugsweise zufällig aus der Gruppe der aktuell angeschlossenen Computer ausgewählt. Ein Vorteil des Übertragungskanals besteht jedoch darin, dass kein Computer ein globales Wissen über den Übertragungskanal besitzt. Vielmehr verfügt jeder Computer über lokales Wissen über sich selbst und seine Nachbarn. Dieses begrenzte lokale Wissen hat den Vorteil, dass alle angeschlossenen Computer gleichrangig (Peers) sind (was das Senden betrifft), und der Ausfall eines Computers (eigentlich von drei beliebigen Computern, wenn eine 4-reguläre und 4-zusammenhängende Form vorliegt) keinen Ausfall des Übertragungskanals zur Folge hat. Durch dieses lokale Wissen ist es für einen Portal-Computer schwierig, zufällig vier Nachbarn für einen suchenden Computer auszuwählen.

[0097] Zum Auswählen der vier Computer sendet ein Portal-Computer eine Kantenverbindungs-Anforderungsnachricht über eine seiner internen Verbindungen, die zufällig ausgewählt wird. Der empfangende Computer sendet wiederum die Kantenverbindungs-Anforderungsnachricht über eine seiner internen Verbindungen, die zufällig ausgewählt wird. Dieses Senden der Nachricht entspricht einem Zufallslauf durch den Graphen, der den Übertragungskanal darstellt. Schließlich entscheidet ein Empfangscomputer, dass die Nachricht weit genug gelangt ist und repräsentiert einen zufällig ausgewählten Computer. Jener Empfangscomputer bietet die interne Verbindung, auf der er die Kantenverbindungs-Anforderungsnachricht empfangen hat, dem suchenden Computer zum Edge Pinning an. Wenn entweder die Computer am Ende der angebotenen internen Verbindung bereits Nachbarn des suchenden Computers sind, dann kann der suchende Computer natürlich keine Verbindung über jene interne Verbindung aufbauen. Der Computer, der entschieden hat, dass die Nachricht bereits weit genug transportiert wurde, erkennt diesen Zustand, dass er bereits Nachbar ist, und sendet die Nachricht zu einem zufällig ausgewählten Nachbarn.

[0098] Bei einer Ausführungsform wird von dem Portal-Computer ermittelt, dass die Entfernung, die die Kantenverbindungs-Anforderungsnachricht zurücklegt, annähernd das Doppelte des geschätzten Durchmessers des Übertragungskanals beträgt. Die Nachricht enthält einen Hinweis auf die Entfernung, die zurückzulegen ist. Jeder Empfangscomputer verringert die noch zu absolvierende Entfernung, bevor die Nachricht weitergesendet wird. Der Computer, der die Nachricht bei einer noch zurückzulegenden Entfernung von null empfängt, wird als der zufällig ausgewählte Computer betrachtet. Wenn jener zufällig ausgewählte Computer keine Verbindung zu dem suchenden Computer aufbauen kann (z.B. weil er bereits an ihn angeschlossen ist), dann sendet der zufällig ausgewählte Computer die Kantenverbindungsanforderung zu einem seiner Nachbarn mit einer neu zurückzulegenden Entfernung weiter. Bei einer Ausführungsform schaltet der weiterleitende Computer die neue zurückzulegende Entfernung zwischen null und eins um, wodurch verhindert werden soll, dass zwei Computer die Nachricht zwischen einander hin- und herschicken.

[0099] Aufgrund der lokalen Begrenztheit der Informationen jedes Computers, der an den Übertragungskanal angeschlossen ist, benötigen die Computer im Allgemeinen keine Kenntnis vom Durchmesser des Übertragungskanals. Bei einer Ausführungsform hat jede Nachricht, die über den Übertragungskanal gesendet wird, ein Feld der zurückgelegten Entfernung. Jeder Computer, der eine Nachricht weiterleitet, vergrößert das Feld der zurückgelegten Entfernung. Darüber hinaus behält jeder Computer einen geschätzten Durchmesser des Übertragungskanals bei. Wenn ein Computer eine Nachricht empfängt, die eine Entfernung zurückgelegt hat, welche anzeigt, dass der geschätzte Durchmesser zu gering ist, aktualisiert er den geschätzten Durchmesser und sendet die Nachricht bezüglich des geschätzten Durchmessers. Empfängt ein Computer eine Nachricht von einem geschätzten Durchmesser, die anzeigt, dass ein Durchmesser größer ist als der eigene geschätzte Durchmesser, aktualisiert er den eigenen geschätzten Durchmesser. Dieser geschätzte Durchmesser wird dazu verwendet, die Entfernung festzulegen, die eine Kantenverbindungs-Anforderungsnachricht zurücklegen sollte.

Externe Datendarstellung

[0100] Die an den Übertragungskanal angeschlossenen Computer können ihre Daten intern in verschiedenen Formaten speichern. Beispielsweise kann ein Computer 32-Bit-Ganzzahlen und ein anderer Computer

64-Bit-Ganzzahlen verwenden. Als weiteres Beispiel kann ein Computer ASCII zur Darstellung von Text verwenden und ein anderer Computer Unicode. Um Datenübertragungen zwischen heterogenen Computern zu ermöglichen, können die über den Übertragungskanal gesendeten Nachrichten das Format XDR (eXternal Data Representation) nutzen.

[0101] Das zugrunde liegende Peer-to-Peer-Kommunikationsprotokoll kann mehrere Sendungen in einem einzigen Nachrichtenstrom senden. Das traditionelle Verfahren zum Abrufen von Nachrichten aus einem Strom bestand darin, dass wiederholt eine Betriebssystemroutine aufgerufen wurde, um die nächste Nachricht im Strom abzurufen. Der Abruf jeder Nachricht kann zwei Anrufe an das Betriebssystem erforderlich machen: einen zum Abrufen der Größe der nächsten Nachricht und den anderen zum Abrufen der Anzahl von Bytes, angegeben durch die abgerufene Größe. Im Vergleich zu den Abrufen lokaler Routinen können derartige Anrufe an das Betriebssystem jedoch sehr langsam sein. Zur Überwindung der mangelnden Leistungsfähigkeit bei derartig wiederholten Anrufen nutzt das Übertragungsverfahren bei einer Ausführungsform das XDR zum Identifizieren der Nachrichtengrenzen in einem Nachrichtenstrom. Das Übertragungsverfahren kann das Betriebssystem auffordern, die nächsten beispielsweise 1.024 Bytes aus dem Strom zur Verfügung zu stellen. Anschließend kann das Übertragungsverfahren wiederholt die XDR-Routinen aufrufen, um Nachrichten abzurufen, und durch den Erfolg oder das Scheitern jedes Aufrufs feststellen, ob ein weiterer Block von 1.024 Bytes von dem Betriebssystem abgerufen werden muss. Der Abruf der XDR-Routinen beinhaltet keine Systemanrufe und ist daher effizienter als wiederholte Systemanrufe.

M-regulär

[0102] Bei der oben beschriebenen Ausführungsform hat jeder vollständig angeschlossene Computer vier interne Verbindungen. Das Übertragungsverfahren kann aber auch mit einer anderen Anzahl von internen Verbindungen zum Einsatz kommen. Beispielsweise könnte jeder Computer 6, 8 oder jede andere gerade Anzahl von internen Verbindungen aufweisen. Mit zunehmender Anzahl interner Verbindungen nimmt meist der Durchmesser des Übertragungskanals ab und folglich auch die Verbreitungszeit für eine Nachricht. Die für den Port eines suchenden Computers an den Übertragungskanal erforderliche Zeit kann jedoch mit zunehmender Anzahl interner Verbindungen größer werden. Wenn die Anzahl interner Verbindungen gerade ist, kann der Übertragungskanal als m-regulär und m-zusammenhängend aufrechterhalten werden (in einem stabilen Zustand). Ist die Anzahl interner Verbindungen ungerade, hat in dem Fall, in dem eine ungerade Anzahl von Computern an den Übertragungskanal angeschlossen ist, einer der Computer weniger als diese ungerade Anzahl von internen Verbindungen. Hierbei ist das Sendernetzwerk weder m-regulär noch m-zusammenhängend. Wenn sich der nächste Computer an den Übertragungskanal anschließt, wird er wieder m-regulär und m-zusammenhängend. Bei einer ungeraden Anzahl interner Verbindungen schwankt der Übertragungskanal folglich immer zwischen dem Zustand, in dem er m-regulär und m-zusammenhängend ist und einem Zustand, in dem er dies nicht ist, hin und her.

Komponenten

[0103] [Fig. 6](#) ist ein Blockdiagramm, welches Komponenten eines Computers darstellt, der an einen Übertragungskanal angeschlossen ist. In der obigen Beschreibung wird generell angenommen, dass lediglich ein Übertragungskanal vorhanden war und jeder Computer nur eine Verbindung zu dem Übertragungskanal hatte. Allgemein kann jedoch ein Computernetzwerk mehrere Übertragungskanäle aufweisen. Jeder Computer kann an mehr als einen Übertragungskanal angeschlossen sein und jeder Computer kann mehrere Verbindungen zu demselben Übertragungskanal aufweisen. Der Übertragungskanal ist für Computerprozesse (z.B. Anwendungsprogramme) gut geeignet, die gemeinsam laufen, z.B. Programme für Netzwerk-Meetings. Jeder Computerprozess kann eine Verbindung zu einem oder mehreren Übertragungskanälen aufbauen. Die Übertragungskanäle können über den Kanaltyp (z.B. Name des Anwendungsprogramms) und die Kanalinstanz identifiziert werden, die separate Übertragungskanäle für jenen Kanaltyp repräsentieren. Wenn ein Prozess versucht, eine Verbindung zu einem Übertragungskanal herzustellen, sucht er einen aktuell mit dem Übertragungskanal verbundenen Prozess, der auf einem Portal-Computer ausgeführt wird. Bei dem Suchprozess wird der Übertragungskanal durch den Kanaltyp und die Kanalinstanz identifiziert.

[0104] Der Computer **600** enthält mehrere Anwendungsprogramme **601**, die als separate Prozesse ablaufen. Jedes Anwendungsprogramm bildet eine Schnittstelle zu einer Senderkomponente **602** für jeden Übertragungskanal, mit dem es verbunden ist. Die Senderkomponente kann als ein Objekt implementiert sein, welches innerhalb des Prozessraums des Anwendungsprogramms instantiiert wird. Als Alternative dazu kann die Senderkomponente als ein separater Prozess oder Programmbaustein aus dem Anwendungsprogramm ausgeführt werden. Bei einer Ausführungsform stellt die Senderkomponente Funktionen zur Verfügung (z.B. Klas-

sen-Verfahren), die von den Anwendungsprogrammen aufgerufen werden können. Zu den wichtigsten Funktionen gehört eine Verbindungsfunktion, die ein Anwendungsprogramm aufrufen kann, indem eine Kennung (ID) des Übertragungskanals weitergeleitet wird, mit dem das Anwendungsprogramm eine Verbindung aufbauen möchte. Das Anwendungsprogramm kann eine Rückroutine bereitstellen, welche die Senderkomponente aufruft, um das Anwendungsprogramm darüber zu benachrichtigen, dass die Verbindung hergestellt ist und dass der Prozess in den vollständig verbundenen Zustand eintritt. Die Senderkomponente kann weiterhin eine Nachrichtenerfassungsfunktion schaffen, welche das Anwendungsprogramm aufrufen kann, um die nächste Nachricht abzurufen, die auf dem Übertragungskanal gesendet wird. Als Alternative dazu kann das Anwendungsprogramm eine Rückroutine bereitstellen (bei der es sich um eine virtuelle Funktion des Anwendungsprogramms handeln kann), die die Senderkomponente aufruft, um das Anwendungsprogramm zu benachrichtigen, dass eine gesendete Nachricht empfangen wurde. Jede Senderkomponente weist mithilfe des Hashing-Algorithmus einen Call-in-Port zu. Wenn die Anrufe am Call-in-Port beantwortet worden sind, werden sie zu anderen Ports übertragen, die als externe und interne Ports dienen.

[0105] Die an den Übertragungskanal angeschlossenen Computer können eine zentrale Verarbeitungseinheit, einen Speicher, Eingabevorrichtungen (z.B. Tastatur und Zeigervorrichtung), Ausgabevorrichtungen (z.B. Anzeigevorrichtung) und Speichervorrichtungen (z.B. Plattenspeicherlaufwerke) umfassen. Bei dem Speicher und den Speichervorrichtungen handelt es sich um computerlesbare Medien, die Computerbefehle enthalten können, welche die Senderkomponente implementiert. Darüber hinaus können Datenstrukturen und Nachrichtenstrukturen über ein Signal, das auf einem computerlesbaren Medium übertragen wird, z.B. einer Kommunikationsverbindung, gespeichert oder übertragen werden.

[0106] [Fig. 7](#) ist ein Blockdiagramm, welches die Teilkomponenten der Senderkomponente in einer Ausführungsform veranschaulicht. Die Senderkomponente enthält eine Verbindungskomponente **701**, einen externen Dispatcher **702**, einen internen Dispatcher **703** für jede interne Verbindung, eine Nachrichtenerfassungskomponente **704** und eine Sendekomponente **712**. Das Anwendungsprogramm kann eine Rückrufverbindungskomponente **710** und eine Antwortempfangskomponente **711** zur Verfügung stellen, die von der Senderkomponente aufgerufen werden. Das Anwendungsprogramm ruft die Verbindungskomponente auf, um eine Verbindung zu einem angegebenen Übertragungskanal herzustellen. Die Verbindungskomponente identifiziert den externen Port und installiert den externen Dispatcher für die Verarbeitung von Nachrichten, die auf dem externen Port empfangen werden. Die Verbindungskomponente ruft die Portalcomputer-Suchkomponente **705** auf, einen Portal-Computer zu identifizieren, der mit dem Übertragungskanal verbunden ist, und ruft die Verbindungsanforderungskomponente **706** auf, um den Portal-Computer (falls vollständig verbunden) aufzufordern, Nachbarprozesse für den neu angeschlossenen Prozess auszuwählen. Der externe Dispatcher empfängt externe Nachrichten, identifiziert den Nachrichtentyp und ruft die geeignete Abwicklungsroutine **707** auf. Der interne Dispatcher empfängt die internen Nachrichten, identifiziert den Nachrichtentyp und ruft die geeignete Abwicklungsroutine **708** auf. Die empfangenen Nachrichten werden in der Warteschlange für die Sendenachrichten **709** gespeichert. Zum Abrufen von Nachrichten aus der Sendewarteschlange wird die Nachrichtenerfassungskomponente aufgerufen. Von dem Anwendungsprogramm wird die Sendekomponente aufgerufen, um Nachrichten auf dem Übertragungskanal zu senden.

Informationsbereitstellungsdienst

[0107] Bei einer Ausführungsform wird mithilfe des Übertragungskanals eine Informationsbereitstellungsdienstanwendung implementiert. Mit dem Informationsbereitstellungsdienst ist es Teilnehmern möglich, Nachrichten zu überwachen, während sie auf dem Übertragungskanal gesendet werden. Jeder Teilnehmer kann als Informationserzeuger, als Informationsverbraucher oder beides fungieren. Die Erzeuger senden Nachrichten zum Übertragungskanal und die Verbraucher empfangen gesendete Nachrichten. Z.B. kann ein Sportkanal zum Verbreiten der Ergebnisse von Sportereignissen verwendet werden. Bestimmte Organisationen, z.B. die National Football League (Nationale Fußballliga), kann autorisiert sein, Ergebnisse von Sportveranstaltungen auf dem Übertragungskanal zu übertragen. Die Betreiber des Übertragungskanals können Abonnements für den Übertragungskanal an Sportbegeisterte verkaufen. Der Informationsbereitstellungsdienst kann zum Verteilen einer Vielzahl verschiedener Inhalte zum Einsatz kommen, einschließlich von Nachrichten, Aktienpreisen, Wettermeldungen, medizinischen Informationen, Verkehrsberichten usw.

[0108] Der Informationsbereitstellungsdienst kann eine Verzeichnis-Webseite bereitstellen, auf der Verbraucher für sie interessante Übertragungskanäle finden und abonnieren können. Das Verzeichnis kann einen hierarchischen Aufbau von Themen der einzelnen Übertragungskanäle aufweisen. Wenn sich ein Benutzer entschließt, einen Übertragungskanal zu abonnieren, können die Senderkomponente und das Anwendungsprogramm für den Informationsbereitstellungsdienst auf den Benutzercomputer heruntergeladen werden, falls die-

se dort nicht schon vorhanden sind. Des Weiteren können der zu dem Übertragungskanal gehörende Kanaltyp und die Kanalinstanz sowie die Kennung der Portal-Computer für den Übertragungskanal auf den Computer des Teilnehmers heruntergeladen werden. Darüber hinaus kann der Informationsbereitstellungsdienst eine Teilnehmerkennung erzeugen, die von einem Portal-Computer verwendet wird, um den Zugriff zu genehmigen oder um nachzuvollziehen, wer sich an den Übertragungskanal angeschlossen hat.

[0109] Die Webseite des Informationsbereitstellungsdienstes kann es des Weiteren einer Organisation ermöglichen, neue Übertragungskanäle zu erzeugen. Z.B. kann die NFL einen Übertragungskanal haben wollen, der zur Verbreitung von Informationen unter ihrer Kontrolle vorgesehen ist. In diesem Fall würde sich die Organisation mit der Webseite in Verbindung setzen, um den Übertragungskanal zu erzeugen. Die Erstellung des Übertragungskanals würde die Erzeugung eines Kanaltyps und einer Kanalinstanz, die Festlegung der Sicherheitsebene (z.B. verschlüsselte Nachrichten), die Festlegung der Bedingungen für die Teilnehmer usw. nach sich ziehen.

[0110] Ein Benutzer kann einen Übertragungskanal für ein einzelnes Thema abonnieren, was einem Blattknoten in der Hierarchie entspricht, oder er kann eine Themenkategorie abonnieren, was einem Nichtblattknoten in der Hierarchie entspricht. Z.B. kann ein Benutzer die Kategorie "Sportergebnisse" abonnieren oder das Thema NFL-Ergebnisse. Bei einer Ausführungsform hätte jedes Thema seinen eigenen Übertragungskanal. Das Abonnement für eine Themenkategorie würde demnach das Abo für zahlreiche Übertragungskanäle bedeuten. Als Alternative dazu kann eine Themenkategorie einen einzigen Übertragungskanal haben. Wenn ein Benutzer lediglich ein Thema in der Kategorie abonniert hat, würde das Anwendungsprogramm für den Informationsbereitstellungsdienst, welches auf dem Computer des Teilnehmers läuft, einfach die nicht zu dem Thema gehörenden Nachrichten außer Acht lassen.

[0111] Von dem Informationsbereitstellungsdienst können viele verschiedene Gebührenstrukturen genutzt werden. So kann einem Abonnenten eine feststehende Gebühr pro Monat für ein Thema berechnet werden. Als Alternative dazu kann einem Abonnenten auch die tatsächliche Verbindungszeit in Rechnung gestellt werden. Wenn der Computer eines Abonnenten angeschlossen ist, könnte er beispielsweise etwa einmal pro Stunde eine Identifizierungsnachricht senden. Auf der Grundlage der Identifizierungsnachrichten könnte ein Abrechnungscomputer die Übertragung überwachen und die Verbindungszeit aufzeichnen. Empfängt der Abrechnungscomputer über einen bestimmten Zeitraum keine Identifizierungsnachricht, geht er davon aus, dass sich der Abonnent mit seinem Computer abgeschaltet hat. Weiterhin kann der Betreiber des Übertragungskanals Einnahmen aus Werbesendungen über den Übertragungskanal erzielen. Die Gebühr für die Werbung auf einem Übertragungskanal kann ausgehend von der Anzahl der Abonnenten, die zum Zeitpunkt der Werbesendung an den Übertragungskanal angeschlossen sind, variieren.

Ein verteiltes Konferenzsystem

[0112] Bei einer Ausführungsform wird mithilfe des Übertragungskanals ein Konferenzsystem implementiert. Jeder Teilnehmer an einer Konferenz stellt eine Verbindung zu dem Übertragungskanal der Konferenz her und ein Teilnehmer wird als Sprecher festgelegt. Das Konferenz-Anwendungsprogramm kann eine Sprecherkomponente und eine Zuhörerkomponente enthalten. Die Sprecherkomponente überträgt die Konferenzereignisse über den Übertragungskanal. Jede Zuhörerkomponente empfängt die Konferenzereignisse und zeigt die Ergebnisse der Konferenzereignisse an. Z.B. kann ein Sprecher auf der Konferenz Folien präsentieren und jede einzelne beschreiben. Jeder Zuhörer kann vor der Konferenz eine elektronische Kopie der Folien erhalten. Zum geplanten Zeitpunkt der Konferenz kommen der Sprecher und jeder Zuhörer der Konferenz zusammen, indem sie eine Verbindung zu dem Übertragungskanal der Konferenz aufbauen. Die Sprecherkomponente ermöglicht es dem Sprecher anzuzeigen, wann welche Folie aufzulegen ist. Wenn eine neue Folie gezeigt wird, sendet die Sprecherkomponente die Nachricht 'Neue Folie'. Wenn die Zuhörerkomponente die Nachricht 'Neue Folie' empfängt, zeigt sie dem Teilnehmer die neue Folie an. Außerdem ermöglicht es die Sprecherkomponente dem Sprecher, mit einem Stift oder einer anderen Zeigevorrichtung auf einer Folie zu zeichnen. Daraufhin sendet die Sprecherkomponente die gezeichneten Nachrichten über den Übertragungskanal, sodass die Zuhörerkomponente die Zeichnung für die Zuhörer anzeigen kann. Das Konferenzsystem kann ebenso Sprache in Text und Text in Sprache umwandeln und so die Anmerkungen des Sprechers an alle Zuhörer weiterleiten.

[0113] Das Konferenzsystem kann eine Verzeichnis-Webseite zur Verfügung stellen, auf der sich Teilnehmer umschauen und für eine sie interessierende Konferenz eintragen können. Das Verzeichnis kann einen hierarchischen Aufbau geplanter Konferenzen haben. Wenn ein Benutzer beschließt, sich für eine Konferenz registrieren zu lassen, kann der Web-Server die Senderkomponente und das Konferenz-Anwendungsprogramm

auf den Computer des Zuhörers herunterladen, falls diese nicht schon auf dem Computer des Zuhörers gespeichert sind. Zudem lädt der Web-Server den Kanaltyp und die Kanalinstanz des Übertragungskanals für die Konferenz sowie die Kennung der Portal-Computer für den Übertragungskanal herunter. Der Web-Server kann darüber hinaus Folien oder andere Inhalte herunterladen, die den Teilnehmern während der Konferenz angezeigt werden sollen.

[0114] Das Konferenzsystem kann es einer Organisation ermöglichen, Konferenzen mithilfe der Web-Seite zu planen. Z.B. kann es sein, dass ein Softwareunternehmen eine Konferenz abhalten möchte, um ein neues Produkt anzukündigen. Die Schaffung der Konferenz würde die Erzeugung eines Kanaltyps und einer Kanalinstanz nach sich ziehen sowie die Festlegung einer Sicherheitsebene (z.B. verschlüsselte Nachrichten), die Festlegung der Auswahlkriterien für die Teilnehmer, eine Beschreibung und die geplante Zeit der Konferenz, die Angabe des Inhalts, der an die Teilnehmer zu verteilen ist usw. Möglicherweise möchte der Sprecher bei der Konferenz den eigentlichen Inhalt (z.B. Folien) nicht im Voraus veröffentlichen. Dann kann z.B. der Inhalt beim Verteilen an die Teilnehmer verschlüsselt sein und während der Konferenz von dem Sprecher ein Schlüssel zum Entschlüsseln des Inhalts verteilt werden. Z.B. kann jede Folie für die Bekanntmachung des Softwareunternehmens mit einem anderen Schlüssel verschlüsselt sein, und der geeignete Schlüssel mit der Nachricht für jede neue Folie gesendet werden.

[0115] Das Konferenzsystem kann es auch Teilnehmern ermöglichen, Anmerkungen auf dem Übertragungskanal zu übertragen. Die Zeiten, zu denen ein Teilnehmer Anmerkungen senden kann, können von dem Sprecher kontrolliert werden. So kann beispielsweise die Sprecherkomponente eine Anmerkungsurlaubsnachricht senden und eine Anmerkungsverbotsnachricht, um die Zeiten, in denen Anmerkungen zulässig sind, zu begrenzen. Anmerkungen, die außerhalb dieser Zeiten gesendet werden, können ignoriert werden. Als Alternative dazu kann es den Teilnehmern erlaubt sein, jederzeit Kommentare zu senden, die anderen Teilnehmer können diese Kommentare jedoch erst sehen, wenn der Sprecher eine Genehmigungsnachricht sendet, die anzeigt, dass die Zuhörerkomponente eine bestimmte Anmerkung anzeigen kann.

[0116] Das Konferenzsystem kann es jedem Teilnehmer ermöglichen, sich während der Konferenz nach Belieben zu dem Konferenzkanal zuzuschalten oder von ihm abzuschalten. Darüber hinaus kann es das Konferenzsystem mehreren Sprechern gleichzeitig ermöglichen, das „Podium“ zu bilden. Die Sprecher können untereinander ein Sprecher-Berechtigungszeichen herumgehen lassen, welches anzeigt, wer gerade spricht und somit die Konferenz leitet. Ein Teilnehmer, der sich erst später zu der Konferenz dazuschaltet, kann sich durch Zugriff auf einen Konferenzüberwachungs-Webserver mit der Konferenz synchronisieren. Der Überwachungs-Webserver kann an den Konferenzkanal angeschlossen sein und den aktuellen Status der Konferenz überwachen. Wenn ein Teilnehmer erst später hinzustößt, kann der Überwachungs-Webserver dem Teilnehmer den aktuellen Status der Konferenz mitteilen. Ab dem Zeitpunkt kann der Teilnehmer auf dem Übertragungskanal zuhören und dem Fortgang der Konferenz folgen. Des Weiteren kann es die Zuhörerkomponente dem Teilnehmer ermöglichen, andere Teile der Präsentation zu betrachten als die, die gegenwärtig gezeigt werden. Somit kann ein Teilnehmer auf andere Teile der Präsentation zurück- oder vorgreifen.

Eine verteilte Spiele-Umgebung

[0117] Bei einer Ausführungsform wird mithilfe von Übertragungskanälen eine Spiele-Umgebung implementiert. Die Spiele-Umgebung wird von einem Spiele-Anwendungsprogramm erzeugt, welches auf dem Computer jedes Spielers läuft und mit einer Senderkomponente interagiert. Jeder Spieler schließt sich einem Spiel an (z.B. einem First Person Shooter Game – FPS), in dem er sich auf dem Übertragungskanal zuschaltet, auf dem das Spiel gespielt wird. Bei jeder Handlung eines Spielers in dem Spiel wird eine Nachricht auf dem Übertragungskanal des Spiels gesendet, die diese Aktion darstellt. Zusätzlich kann ein Spieler Nachrichten (z.B. Strategieinformationen) durch Senden einer Nachricht an einen oder mehrere andere Spieler schicken. Wenn das Spiele-Anwendungsprogramm einen Hinweis auf eine Aktion empfängt, entweder über den Übertragungskanal empfangen oder von dem Spieler an diesem Computer erzeugt, aktualisiert es den aktuellen Stand des Spiels. Das Spiel kann beendet werden, wenn einer der Spieler einen bestimmten Punktestand erreicht, sämtliche anderen Spieler besiegt hat, alle Spieler das Spiel verlassen, usw.

[0118] Um die Erzeugung von Spielen für die Spiele-Umgebung zu erleichtern, gibt es eine Anwendungs-Programmierschnittstelle „API“, die die Spiele-Entwickler unterstützt. Die API kann High-Level-Spielefunktionen erzeugen, die von den meisten Arten der FPS genutzt werden. Z.B. kann die API Funktionen zum Anzeigen, dass sich ein Spieler zu einer neuen Position bewegt hat, zum Schießen in eine andere Richtung, zum Melden eines Ergebnisses, zum Bekanntgeben des Hinzukommens/Ausscheidens von Spielern, zum Senden einer Nachricht zu einem anderen Spieler, usw. enthalten.

[0119] Die Spiele-Umgebung kann eine Spieler-Webseite erzeugen, über die sich Spieler den Stand der aktuellen Spiele ansehen und für neue Spiele eintragen können. Auf dem Spiele-Webserver würde eine Abbildung zwischen jedem Spiel und dem Übertragungskanal erfolgen, auf dem das Spiel gespielt werden soll. Zur Teilnahme an einem Spiel würde der Benutzer die Senderkomponente und das Spiele-Anwendungsprogramm vom Webserver herunterladen. Ebenso würde der Spieler die Beschreibung des Spiels herunterladen, in der auch die Grafiken für das Spiel enthalten sein können. Zudem würde der Webserver auch den Kanaltyp und die Kanalinstanz für das Spiel sowie die Kennung der Portal-Computer für das Spiel bereitstellen. Die Spiele-Umgebung kann weiterhin einen Spiele-Überwachungscomputer aufweisen, der mit jedem Spiel verbunden ist, die Aktivitäten des Spiels überwacht und sie an den Webserver meldet. Mit dieser Information über die Aktivitäten kann der Webserver Informationen über den aktuellen Stand (z.B. Anzahl der Spieler) jedes Spiels geben.

[0120] Ebenso kann die Spiele-Umgebung auch für andere Spieler als FPS verwendet werden. Z.B. können verschiedene Rollenspiele gespielt werden, bei denen sich Spieler für unterschiedliche Rollen eintragen. Wenn eine Rolle nicht besetzt wird oder ein Spieler in jener Rolle nicht spielt, kann ein automatischer Spieler die Rolle übernehmen.

[0121] In der nachfolgenden Tabelle sind Nachrichten aufgeführt, die von den Senderkomponenten gesendet werden.

EXTERNE NACHRICHTEN

Nachrichtentyp	Beschreibung
seeking_connection_call	Zeigt an, dass ein Suchprozess wissen möchte, ob der Empfangsprozess vollständig mit dem Übertragungskanal verbunden ist
connection_request_call	Zeigt an, dass der Sendeprozess möchte, dass der Empfangsprozess eine Verbindung des Sendeprozesses zu dem Übertragungskanal herstellt
edge_proposal_call	Zeigt an, dass der Sendeprozess eine Kante vorschlägt, durch die der Empfangsprozess eine Verbindung zum Über-
	tragungskanal aufbauen kann (d.h. Edge Pinning)
port_connection_call	Zeigt an, dass der Sendeprozess einen Port vorschlägt, durch den der Empfangsprozess eine Verbindung zum Übertragungskanal aufbauen kann
connected_stmt	Zeigt an, dass der Sendeprozess mit dem Übertragungskanal verbunden ist
condition_repair_stmt	Zeigt an, dass sich der Empfangsprozess von einem seiner Nachbarn trennen und mit einem der Prozesse verbinden sollte, in dem sich die Nachbarn mit leeren Ports befinden

INTERNE NACHRICHTEN

Nachrichtentyp	Beschreibung
broadcast_stmt	Zeigt eine Nachricht an, die über den Übertragungs-kanal für die Anwendungsprogramme gesendet wird
connection_port_search_stmt	Zeigt an, dass der festgelegte Prozess einen Port sucht, über den er sich mit dem Übertragungskanal verbinden kann
connection_edge_search_call	Zeigt an, dass der Anforderungsprozess eine Kante sucht, über die er eine Verbindung zum Übertra-gungskanal aufbauen kann
connection_edge_search_resp	Zeigt an, ob die Kante zwischen diesem Prozess und dem sendenden Nachbarn von der anfordernden Sei-te akzeptiert wurde
diameter_estimate_stmt	Zeigt einen geschätzten Durchmesser des Übertra-gungskanals an
diameter_reset_stmt	Zeigt an, dass der geschätzte Durchmesser auf an-gezeigten Durchmesser zurückzusetzen ist
disconnect_stmt	Zeigt an, dass der sendende Nachbar vom Übertra-gungskanal getrennt ist
condition_check_stmt	Zeigt an, dass Nachbarn mit leeren Ports entdeckt wurden
condition_double_check_stmt	Zeigt an, dass die Nachbarn mit leeren Ports diesel-ben Nachbarn haben
shutdown_stmt	Zeigt an, dass der Übertragungskanal geschlossen wird

Flussdiagramme

[0122] Die [Fig. 8](#) bis [Fig. 34](#) zeigen Flussdiagramme, welche die Verarbeitung der Senderkomponente bei einer Ausführungsform veranschaulichen. [Fig. 8](#) ist ein Flussdiagramm, das die Verarbeitung der Verbindungs-routine bei einer Ausführungsform zeigt. Dieser Routine wird auch einem Kanaltyp (z.B. Anwendungsname) und eine Kanalinstanz (z.B. Sitzungskennung) zugeschrieben, die den Übertragungskanal angibt, mit dem der Prozess verbunden werden soll. Der Routine wird ebenfalls eine Zusatzinformation zugeordnet, die eine Liste von Portal-Computern und eine Verbindungsrückruf-Routine enthält. Wenn die Verbindung hergestellt ist, wird die Verbindungsrückruf-Routine aufgerufen, um das Anwendungsprogramm zu benachrichtigen. Wenn dieser Prozess die Routine aufruft, befindet sie sich im Verbindungssuchzustand. Ist ein Portal-Computer gefunden, der angeschlossen ist, und stellt diese Routine die Verbindung zu wenigstens einem Nachbarn her, tritt dieser Prozess in den teilweise verbundenen Zustand ein, und wenn der Prozess schließlich mit vier Nachbarn ver-bunden ist, gelangt er in den vollständig verbundenen Zustand. Im Small Regime kann ein vollständig verbun-dener Prozess weniger als vier Nachbarn haben. In Block **801** öffnet die Routine den Call-in-Port, durch den der Prozess beim Aufbau externer und interner Verbindungen mit anderen Prozessen kommunizieren soll. Der Port wird mithilfe des oben beschriebenen Hashing-Algorithmus als erster verfügbarer Port ausgewählt. In Block **802** setzt die Routine die Verbindungszeit auf die aktuelle Zeit. Die Verbindungszeit wird zum Identifizie-ren der Instanz des Prozesses verwendet, der über diesen externen Port verbunden ist. Ein Prozess kann un-ter Verwendung eines Call-in-Ports eine Verbindung mit einem Übertragungskanal eines bestimmten Kanaltyps und einer Kanalinstanz aufbauen und diese anschließend aufheben, woraufhin ein anderer Prozess die Verbindung zu demselben Übertragungskanal mithilfe desselben Call-in-Ports aufbauen kann. Bevor der an-dere Prozess vollständig verbunden ist, kann ein weiterer Prozess versuchen, mit ihm zu kommunizieren, wo-bei angenommen wird, dass es sich um den vollständig verbundenen alten Prozess handelt. In diesem Fall kann die Verbindungszeit zum Erkennen dieser Situation verwendet werden. In Block **803** ruft die Routine die Portal-Computer-Suchroutine auf, die den Kanaltyp und die Kanalinstanz weiterleitet. Die Portal-Compu-ter-Suchroutine versucht, einen Portal-Computer zu lokalisieren, über den sich dieser Prozess mit dem Über-tragungskanal für den angegebenen Typ und die Instanz verbinden kann. Wenn die Portal-Computer-Suchrou-tine bei der Suche nach einem vollständig verbundenen Prozess an jedem Portal-Computer im Entscheidungs-block **804** erfolgreich ist, geht die Routine weiter zu Block **805**, ansonsten gibt die Routine einen Hinweis auf ein erfolgloses Ergebnis aus. Wenn im Entscheidungsblock **805** kein anderer Portal-Computer als jener gefun-

den wurde, an dem der Prozess ausgeführt wurde, dann ist dies der erste Prozess zum vollständigen Verbinden mit dem Übertragungskanal, und die Routine geht weiter zu Block **806**, ansonsten geht es weiter in Block **808**. In Block **806** ruft die Routine die VerbindungsaufbauRoutine auf, um den Zustand dieses Prozesses in den vollständig verbundenen zu ändern. In Block **807** installiert die Routine den externen Dispatcher zum Verarbeiten von Nachrichten, die durch den externen Port für den angegebenen Kanaltyp und die Kanalinstanz empfangen wurden. Wenn eine Nachricht durch jenen externen Port empfangen wurde, wird der externe Dispatcher aufgerufen. Anschließend kehrt die Routine wieder an den Ausgangspunkt zurück. In Block **808** installiert die Routine einen externen Dispatcher. In Block **809** ruft die Routine die VerbindungsanforderungsRoutine auf, um den Prozess des Identifizierens von Nachbarn für den suchenden Computer einzuleiten. Anschließend kehrt die Routine wieder an den Anfang zurück.

[0123] [Fig. 9](#) ist ein Flussdiagramm, das die Verarbeitung der Portal-Computer-Suchroutine nach einer Ausführungsform zeigt. Dieser Routine wird der Kanaltyp und die Kanalinstanz des Übertragungskanals zugeleitet, mit dem sich dieser Prozess verbinden möchte. Diese Routine überprüft für jede Suchtiefe (z.B. Port-Nummer) die Portal-Computer auf jener Suchtiefe. Wenn auf jener Suchtiefe ein Portal-Computer gefunden wird, bei dem ein Prozess vollständig mit dem Übertragungskanal verbunden ist, gibt diese Routine einen Erfolgshinweis aus. In den Blöcken **902–911** wird dieser Vorgang wahlweise auf jeder Suchtiefe solange wiederholt, bis ein Prozess gefunden worden ist. In Block **902** wählt die Routine die nächste Suchtiefe mithilfe eines Port-Nummern-Ordnungsalgorithmus aus. Wenn im Entscheidungsblock **903** alle Suchtiefen bereits während der Ausführung dieser Wiederholungsschleife ausgewählt worden sind, d.h. für die aktuell ausgewählte Tiefe, dann gibt die Routine eine Fehlermeldung aus, ansonsten geht die Routine weiter zu Block **904**. In den Blöcken **904–911** absolviert die Routine wieder schleifenförmig die Auswahl jedes Portal-Computers und stellt fest, ob ein Prozess von jenem Portal-Computer mit dem Übertragungskanal verbunden ist (oder eine Verbindung herzustellen versucht), der den zugewiesenen Kanaltyp und die Kanalinstanz aufweist. In Block **904** wählt die Routine den nächsten Portal-Computer aus. In dem Entscheidungsblock **905** kehrt die Routine, wenn bereits alle Portal-Computer ausgewählt worden sind, zu Block **902** zurück und wählt die nächste Suchtiefe aus, ansonsten geht die Routine weiter zu Block **906**. In Block **906** wählt die Routine den ausgewählten Portal-Computer durch den Port aus, der von der Suchtiefe dargestellt wird. Im Entscheidungsblock **907** geht die Routine anschließend, falls die Anwahl erfolgreich war, weiter zu Block **908**, ansonsten kehrt sie wieder zurück zu Block **904** und wählt den nächsten Portal-Computer aus. Die Anwahl ist erfolgreich, wenn der angewählte Port der Call-in-Port des Übertragungskanals mit dem zugewiesenen Kanaltyp und der Kanalinstanz eines Prozesses ist, der auf jenem Portal-Computer ausgeführt wird. In Block **908** ruft die Routine eine Prozesskontakt-Routine auf, die den Antwortprozess des Portal-Computers durch den angewählten Port kontaktiert und feststellt, ob jener Prozess vollständig mit dem Übertragungskanal verbunden ist. In Block **909** verbindet sich die Routine mit dem ausgewählten Portal-Computer. Im Entscheidungsblock **910** gibt die Routine anschließend, falls der Antwortprozess vollständig mit dem Übertragungskanal verbunden ist, eine Erfolgsmeldung aus, ansonsten setzt die Routine bei Block **911** fort. In Block **911** ruft die Routine die Überprüfung der externen RufRoutine auf, um festzustellen, ob ein externer Ruf für diesen Prozess als Portal-Computer erfolgt ist, und verarbeitet jenen Ruf. Danach kehrt die Routine schleifenförmig zu Block **904** zurück und wählt den nächsten Portal-Computer aus.

[0124] [Fig. 10](#) ist ein Flussdiagramm, das die Verarbeitung der ProzesskontaktRoutine in einer Ausführungsform veranschaulicht. Die Routine stellt fest, ob der Prozess des ausgewählten Portal-Computers, der den Ruf zum ausgewählten Port beantwortet hat, vollständig mit dem Übertragungskanal verbunden ist. In Block **1001** sendet die Routine eine externe Nachricht (d.h. `seeking_connection_call`) an den Antwortprozess, die angibt, dass ein Suchprozess wissen will, ob der Antwortprozess vollständig mit dem Übertragungskanal verbunden ist. In Block **1002** empfängt die Routine die externe Antwortnachricht von dem Antwortprozess. Im Entscheidungsblock **1003** setzt die Routine dann, wenn die externe Antwortnachricht erfolgreich empfangen wurde (d.h. `seeking_connection_resp`), in Block **1004** fort, ansonsten kehrt die Routine an den Anfang zurück. Wo immer die Sendekomponente den Empfang einer externen Nachricht anfordert, legt sie eine Ablaufzeit fest. Wenn die externe Nachricht nicht innerhalb der Ablaufzeit empfangen wird, überprüft die Senderkomponente ihren eigenen Call-in-Port um festzustellen, ob ein anderer Prozess sie ruft. Konkret kann der angewählte Prozess den Wählprozess rufen, was zu einer Blockadesituation führen kann. Die Senderkomponente kann die Empfangsanforderung mehrmals wiederholen. Wenn die erwartete Nachricht nicht empfangen wird, verarbeitet die Sendekomponente den Fehler in angemessener Weise. Wenn der Antwortprozess im Entscheidungsblock **1004** in seiner Antwortnachricht angibt, dass er vollständig mit dem Übertragungskanal verbunden ist, geht die Routine weiter zu Block **1005**, ansonsten zu **1006**. In Block **1005** fügt die Routine den ausgewählten Portal-Computer zu einer Liste angeschlossener Portal-Computer hinzu und kehrt anschließend wieder zurück. In Block **1006** fügt die Routine den Antwortprozess zu einer Liste weiterer Suchprozesse hinzu und kehrt wieder zurück.

[0125] Fig. 11 ist ein Flussdiagramm, das die Verarbeitung der Verbindungsanforderungs-Routine nach einer Ausführungsform zeigt. Diese Routine fordert einen Prozess eines Portal-Computers an, der als vollständig mit dem Übertragungskanal verbunden identifiziert worden ist, um die Verbindung dieses Prozesses mit dem Übertragungskanal einzuleiten. Wenn im Entscheidungsblock **1101** wenigstens ein Prozess eines Portal-Computers gefunden wurde, der vollständig mit dem Übertragungskanal verbunden ist, geht die Routine weiter zu Block **1103**, ansonsten zu Block **1102**. Ein Prozess des Portal-Computers kann nicht mehr in der Liste stehen, wenn er vor kurzem vom Übertragungskanal getrennt wurde. Bei einer Ausführungsform kann ein Suchcomputer immer die gesamte Suchtiefe durchsuchen und mehrere Portal-Computer finden, durch die er eine Verbindung mit dem Übertragungskanal aufbauen kann. In Block **1102** startet die Routine erneut den Prozess des Verbindens mit dem Übertragungskanal und kehrt anschließend wieder zurück. In Block **1103** wählt die Routine diesen Prozess von einem der gefundenen Portal-Computer über den Call-in-Port an. Im Entscheidungsblock **1104** geht die Routine dann, wenn das Anwählen erfolgreich ist, weiter zu Block **1105**, ansonsten zu Block **1113**. Die Anwahl kann erfolglos sein, wenn beispielsweise der angewählte Prozess vor kurzem von dem Übertragungskanal getrennt wurde. In Block **1105** sendet die Routine eine externe Nachricht zu dem angewählten Prozess, in der eine Verbindung zum Übertragungskanal angefordert wird (d.h. `connection_request_call`). In Block **1106** empfängt die Routine die Antwortnachricht (d.h. `connection_request_resp`). Wenn die Antwortnachricht im Entscheidungsblock **1107** erfolgreich empfangen wurde, geht die Routine weiter zu Block **1108**, ansonsten zu Block **1113**. In Block **1108** setzt die Routine die erwartete Anzahl von Löchern (d.h. leere interne Verbindungen) für diesen Prozess ausgehend von der empfangenen Antwort. Im Large Regime ist die erwartete Anzahl von Löchern gleich null. Im Small Regime schwankt die erwartete Anzahl von Löchern zwischen eins und drei. In Block **1109** legt die Routine den geschätzten Durchmesser des Übertragungskanals ausgehend von der empfangenen Antwort fest. Wenn im Entscheidungsblock **1111** der angewählte Prozess bereit ist, eine Verbindung mit diesem Prozess herzustellen, wie durch die Antwortnachricht angegeben, geht die Routine weiter zu Block **1112**, ansonsten zu Block **1113**. In Block **1112** ruft die Routine die hinzugefügte Nachbarroutine auf, um den Antwortprozess als Nachbarn dieses Prozesses hinzuzufügen. Das Hinzufügen des Antwortprozesses erfolgt meist, wenn sich der Übertragungskanal im Small Regime befindet. Im Large Regime wird die zufällige Suche nach einem Nachbarn durchgeführt. In Block **1113** beendet die Routine die externe Verbindung mit dem Antwortprozess-Computer und kehrt wieder zurück.

[0126] Fig. 12 ist ein Flussdiagramm der Verarbeitung zum Überprüfen der externen Rufoutine nach einer Ausführungsform. Diese Routine wird aufgerufen um festzustellen, ob ein weiterer Suchprozess versucht, über diesen Prozess eine Verbindung mit dem Übertragungskanal herzustellen. In Block **1201** versucht die Routine, einen Anruf auf dem Call-in-Port zu beantworten. Wenn die Antwort im Entscheidungsblock **1202** erfolgreich ist, geht die Routine weiter zu Block **1203**, ansonsten kehrt sie wieder zurück. In Block **1203** empfängt die Routine die externe Nachricht von dem externen Port. Wenn der Nachrichtentyp im Entscheidungsblock **1204** anzeigt, dass ein Suchprozess anruft (d.h. `seeking_connection_call`), dann geht die Routine weiter zu Block **1205**, ansonsten kehrt sie wieder zurück. In Block **1205** sendet die Routine eine externe Nachricht (d.h. `seeking_connection_resp`) zu dem anderen suchenden Prozess, was anzeigt, dass dieser Prozess ebenfalls eine Verbindung sucht. Wenn im Entscheidungsblock **1206** das Senden der externen Nachricht erfolgreich ist, geht die Routine zu Block **1207**, ansonsten kehrt die Routine wieder zurück. In Block **1207** fügt die Routine den anderen Suchprozess zu einer Liste weiterer Suchprozesse hinzu und kehrt anschließend wieder zurück. Diese Liste kann verwendet werden, wenn der Prozess keinen Prozess findet, der vollständig mit dem Übertragungskanal verbunden ist. In diesem Fall kann dieser Prozess überprüfen, ob ein anderer Suchprozess erfolgreich eine Verbindung mit dem Übertragungskanal hergestellt hat. Z.B. kann ein anderer Suchprozess der erste Prozess werden, der vollständig mit dem Übertragungskanal verbunden ist.

[0127] Fig. 13 ist ein Flussdiagramm zur Verarbeitung der Routine zum Fertigstellen der Verbindung nach einer Ausführungsform. Diese Routine legt den Zustand des Prozesses fest, um eine vollständige Verbindung mit dem Übertragungskanal herzustellen, und ruft eine Rückrufoutine auf, um das Anwendungsprogramm darüber zu benachrichtigen, dass der Prozess jetzt vollständig mit dem angeforderten Übertragungskanal verbunden ist. In Block **1301** setzt die Routine den Verbindungsstatus dieses Prozesses auf vollständig verbunden. In Block **1302** benachrichtigt die Routine die anderen Suchprozesse darüber, dass ein vollständiger Port besteht, in dem eine externe Verbindungsnachricht an sie gesendet wird (d.h. `connected_stmt`). In Block **1303** ruft die Routine die Rückrufverbindungs-Routine auf, um das Anwendungsprogramm zu benachrichtigen und kehrt danach wieder zurück.

[0128] Fig. 14 ist ein Flussdiagramm, das die Verarbeitung der externen Dispatcher-Routine nach einer Ausführungsform zeigt. Diese Routine wird aufgerufen, wenn der externe Port eine Nachricht empfängt. Diese Routine empfängt die Nachrichten, identifiziert den Typ der externen Nachricht und ruft die geeignete Routine

für die Verarbeitung der Nachricht auf. Diese Routine führt schleifenartig die Verarbeitung jeder Nachricht so lange durch, bis alle empfangenen Nachrichten verarbeitet worden sind. In Block **1401** antwortet die Routine auf den externen Port (z.B. nimmt den Hörer ab) und ruft eine externe Nachricht ab. Im Entscheidungsblock **1402** geht die Routine dann, wenn eine Nachricht abgerufen wurde, weiter zu Block **1403**, ansonsten endet die Routine bei dem externen Port in Block **1415** und kehrt anschließend zurück. Wenn der Nachrichtentyp in dem Entscheidungsblock **1403** ein Prozess zum Suchen einer Verbindung ist (d.h. `seeking_connection_call`), dann ruft die Routine in Block **1404** die Verbindungssuchrufabwicklungs-Routine auf, ansonsten geht sie weiter zu Block **1405**. Wenn der Nachrichtentyp im Entscheidungsblock **1405** ein Verbindungsanforderungsruf ist (d.h. `connection_request_call`), dann ruft die Routine im Block **1406** die Verbindungsanforderungsrufabwicklungs-Routine auf, ansonsten geht die Routine weiter zu Block **1407**. Wenn es sich im Entscheidungsblock **1407** um den Nachrichtentyp Kantenvorschlagsruf handelt (d.h. `edge_proposal_call`), dann ruft die Routine im Block **1408** die Kantenvorschlagsrufabwicklungs-Routine auf, ansonsten geht die Routine weiter zu Block **1409**. Wenn im Entscheidungsblock **1409** der Nachrichtentyp ein Port-Verbindungsruf ist (d.h. `port_connect_call`), dann ruft die Routine in Block **1410** die Port-Verbindungsrufabwicklungs-Routine auf, ansonsten geht die Routine weiter zu Block **1411**. Wenn der Nachrichtentyp im Entscheidungsblock **1411** eine Port-Anweisung ist (d.h. `connected_stmt`), ruft die Routine in Block **1112** die Verbundabwicklungsanweisung auf, ansonsten geht die Routine weiter zu Block **1212**. Wenn der Nachrichtentyp im Entscheidungsblock **1412** eine Zustandsbehebungsanweisung ist (d.h. `condition_repair_stmt`), dann ruft die Routine in Block **1413** die Zustandsbehebungsabwicklungs-Routine auf, ansonsten geht die Routine schleifenförmig weiter zu Block **1414**, um die nächste Nachricht zu verarbeiten. Nachdem jede Bearbeitungs-Routine aufgerufen worden ist, absolviert sie die Schleife zu Block **1414**. In Block **1414** schaltet sich die Routine vom externen Port ab und macht bei Block **1401** weiter, um die nächste Nachricht zu empfangen.

[0129] [Fig. 15](#) ist ein Flussdiagramm, das die Verarbeitung der Verbindungssuchanrufabwicklungs-Routine nach einer Ausführungsform darstellt. Diese Routine wird aufgerufen, wenn sich ein Suchprozess meldet, der einen Portal-Computer identifizieren will, über den er eine Verbindung zum Übertragungskanal aufbauen kann. Wenn dieser Prozess im Entscheidungsblock **1501** aktuell vollständig mit dem Übertragungskanal verbunden ist, der in der Nachricht angegeben ist, geht die Routine weiter zu Block **1502**, ansonsten zu Block **1503**. In Block **1502** setzt die Routine eine Nachricht, die anzeigt, dass dieser Prozess vollständig mit dem Übertragungskanal verbunden ist, und geht weiter zu Block **1505**. In Block **1503** erstellt die Routine eine Nachricht, die anzeigt, dass dieser Prozess nicht vollständig verbunden ist. In Block **1504** fügt die Routine die Kennung des Suchprozesses zu einer Liste weiterer Suchprozesse hinzu. Wenn dieser Prozess nicht vollständig verbunden ist, versucht er, sich mit dem Übertragungskanal zu verbinden. In Block **1505** sendet die Routine die externe Nachrichtenantwort (d.h. `seeking_connection_resp`) an den Suchprozess und kehrt anschließend wieder zurück.

[0130] [Fig. 16](#) ist ein Flussdiagramm, das die Verarbeitung der Abwicklungsroutine für den Verbindungsanforderungsruf nach einer Ausführungsform darstellt. Diese Routine wird aufgerufen, wenn der Rufprozess will, dass dieser Prozess die Verbindung des Prozesses mit dem Übertragungskanal einleitet. Diese Routine ermöglicht es entweder, dass der Rufprozess eine interne Verbindung mit diesem Prozess herstellt (z.B. im Small Regime) oder setzt den Prozess des Identifizierens eines Prozesses in Gang, mit dem sich der Rufprozess verbinden kann. Wenn es sich bei diesem Prozess in dem Entscheidungsblock **1601** um einen handelt, der aktuell vollständig mit dem Übertragungskanal verbunden ist, geht die Routine weiter zu Block **1603**, ansonsten endet sie am externen Port in Block **1602** und kehrt wieder zurück. In Block **1603** legt die Routine die Anzahl von Löchern fest, die der Rufprozess in der Antwortnachricht erwarten sollte. In Block **1604** legt die Routine den geschätzten Durchmesser in der Antwortnachricht fest. In Block **1605** gibt die Routine an, ob dieser Prozess bereit ist für eine Verbindung mit dem Rufprozess. Dieser Prozess ist verbindungsbereit, wenn die Anzahl seiner Löcher größer ist als null und der Rufprozess kein Nachbar dieses Prozesses ist. In Block **1606** sendet die Routine dem Rufprozess eine externe Nachricht, die auf den Verbindungsanforderungsruf reagiert (d.h. `connection_request_resp`). In Block **1607** notiert die Routine die Anzahl von Löchern, die der Rufprozess füllen muss, wie in der Anforderungsnachricht angegeben. Wenn dieser Prozess im Entscheidungsblock **1608** bereit für die Verbindung mit dem Rufprozess ist, geht die Routine weiter zu Block **1609**, ansonsten zu Block **1611**. In Block **1609** ruft die Routine die Nachbarhinzufügroutine auf, um den Rufprozess als Nachbarn hinzuzufügen. In Block **1610** zählt die Routine die Anzahl von Löchern, die der Rufprozess füllen muss, herunter und geht weiter zu Block **1611**. In Block **1611** trennt sich die Routine vom externen Port. Wenn dieser Prozess in dem Entscheidungsblock **1612** keine Löcher aufweist oder der geschätzte Durchmesser größer ist als eins (d.h. im Large Regime), dann geht die Routine weiter zu Block **1613**, ansonsten zu Block **1616**. In den Blöcken **1613-1615** leitet die Routine schleifenartig eine Kantenanforderung weiter, über die sie eine Verbindung mit dem Rufprozess des Übertragungskanals aufbauen kann. Für jedes Paar Löcher des Rufprozesses, das gefüllt werden muss, wird eine Anforderung weitergeleitet. Wenn die Anzahl der zu füllenden Löcher des

Rufprozesses im Entscheidungsblock **1613** größer als oder gleich zwei ist, geht die Routine weiter zu Block **1614**, ansonsten zu Block **1616**. In Block **1614** ruft die Routine die Verbindungskantensuchweiterleitungs-Routine auf. Diese aufgerufene Routine wird einer Anzeige des Rufprozesses und dem zufälligen Abstand zugeleitet. Bei einer Ausführungsform ist der Abstand zweimal so groß wie der geschätzte Durchmesser des Übertragungskanals. In Block **1614** zählt die Routine die zum Füllen verbliebenen Löcher um zwei zurück und geht schleifenartig zu Block **1613**. Wenn im Entscheidungsblock **1616** noch immer ein Loch zu füllen ist, geht die Routine weiter zu Block **1617**, ansonsten kehrt sie an den Anfang zurück. In Block **1617** ruft die Routine die Lochauffüllroutine auf, wobei die Kennung des Rufprozesses angegeben wird. Die Lochfüllroutine sendet eine Verbindungsport-Suchanweisung (d.h. `connection_port_search_stmt`) für ein Loch eines verbundenen Prozesses, über den sich der Rufprozess mit dem Übertragungskanal verbinden kann. Anschließend kehrt die Routine wieder zurück.

[0131] [Fig. 17](#) ist ein Flussdiagramm, das die Verarbeitung der Nachbarhinzufügungs-Routine aus einer Ausführungsform darstellt. Diese Routine fügt den Prozess des Anrufes des externen Ports als Nachbarn zu diesem Prozess hinzu. In Block **1701** identifiziert die Routine den Rufprozess am externen Port. In Block **1702** setzt die Routine ein Flag, welches anzeigt, dass der Nachbar die gesendeten Nachrichten von diesem Prozess noch nicht empfangen hat. Mit diesem Flag wird sichergestellt, dass es keine Lücken bei den Nachrichten gibt, die anfangs zu dem neuen Nachbarn gesendet werden. Bei dieser Verbindung wird aus dem externen Port der interne Port. Wenn es sich im Entscheidungsblock **1703** bei diesem Prozess um den Verbindungszustand handelt, verbindet sich dieser Prozess mit seinem ersten Nachbarn und die Routine geht weiter zu Block **1704**, ansonsten zu Block **1705**. In Block **1704** setzt die Routine den Verbindungsstatus dieses Prozesses auf teilweise verbunden. In Block **1705** fügt die Routine den Rufprozess zu der Liste von Nachbarn dieses Prozesses hinzu. In Block **1706** installiert die Routine einen internen Dispatcher für den neuen Nachbarn. Der interne Dispatcher wird aufgerufen, wenn über den internen Port jenes neuen Nachbarn eine Nachricht von ihm empfangen wird. Wird im Entscheidungsblock **1707** festgestellt, dass dieser Prozess Nachrichten gespeichert hat, während er nicht vollständig verbunden war, geht die Routine weiter zu Block **1708**, ansonsten zu Block **1709**. Bei einer Ausführungsform kann ein Prozess, der teilweise verbunden ist, die Nachrichten zwischenspeichern, die er über eine interne Verbindung empfängt, sodass er diese Nachrichten senden kann, wenn er mit seinen neuen Nachbarn verbunden ist. In Block **1708** sendet die Routine die gespeicherten Nachrichten über den internen Port zu dem neuen Nachbarn. Wenn die Anzahl der Löcher dieses Prozesses im Entscheidungsblock **1709** genauso groß ist wie die erwartete Lochanzahl, dann ist dieser Prozess vollständig verbunden und die Routine geht weiter zu Block **1710**, ansonsten zu Block **1711**. In Block **1710** ruft die Routine die VerbindungsaufbauRoutine auf um anzuzeigen, dass dieser Prozess vollständig verbunden ist. Wenn im Entscheidungsblock **1711** die Lochanzahl für diesen Prozess null beträgt, dann geht die Routine weiter zu Block **1712**, ansonsten kehrt sie an den Anfang zurück. In Block **1712** löscht die Routine sämtliche wartenden Kanten und kehrt anschließend zurück. Eine wartende Kante ist eine Kante, die für diesen Prozess jedoch nicht mehr zum Edge Pinning benötigt wird.

[0132] [Fig. 18](#) ist ein Flussdiagramm, welches die Verarbeitung der Weiterleitungs-Verbindungskantensuchweiterleitungs-Routine nach einer Ausführungsform veranschaulicht. Diese Routine ist verantwortlich für die Übertragung einer Anforderung zum Verbinden eines Anforderungsprozesses mit einem zufällig ausgewählten Nachbarn dieses Prozesses über den internen Port des ausgewählten Nachbarn, wobei es sich um einen Teil des Zufallslaufs handelt. Wenn der Weiterleitungsabstand im Entscheidungsblock **1801** größer als null bleibt, geht die Routine weiter zu Block **1804**, ansonsten zu Block **1802**. Wenn die Anzahl der Nachbarn dieses Prozesses im Entscheidungsblock **1802** größer ist als eins, geht die Routine zu Block **1804** weiter, ansonsten befindet sich der Übertragungskanal im Small Regime und die Routine setzt bei Block **1803** fort. Wenn der Anforderungsprozess im Entscheidungsblock **1803** ein Nachbar dieses Prozesses ist, kehrt die Routine an den Anfang zurück, ansonsten geht sie weiter zu Block **1804**. In den Blöcken **1804–1807** versucht die Routine schleifenartig, eine interne Verbindungskanten-Suchrufnachricht (d.h. `connection_edge_search_call`) an einen zufällig ausgewählten Nachbarn zu senden. In Block **1804** wählt die Routine zufällig einen Nachbarn dieses Prozesses aus. Wenn im Entscheidungsblock **1805** sämtliche Nachbarn dieses Prozesses bereits ausgewählt wurden, kann die Routine die Nachricht nicht weiterleiten und kehrt an den Anfang zurück, ansonsten setzt sie bei Block **1806** fort. In Block **1806** sendet die Routine eine interne Verbindungskanten-Suchrufnachricht an den ausgewählten Nachbarn. Wenn im Entscheidungsblock **1807** das Senden der Nachricht erfolgreich ist, geht die Routine weiter zu Block **1808**, ansonsten kehrt die Routine schleifenartig zu Block **1804** zurück, um den nächsten Nachbarn auszuwählen. Ist das Senden einer internen Nachricht nicht erfolgreich, kann sich der Nachbar ungeplant von dem Übertragungskanal getrennt haben. Wann immer eine solche Situation von der Senderkomponente erkannt wird, versucht diese, einen anderen Nachbarn durch Aufruf der Lochfüllroutine zu finden, um ein einziges Loch zu füllen, oder durch Aufruf der Verbindungskantensuchweiterleitungs-Routine um zwei Löcher zu füllen. In Block **1808** bemerkt die Routine, dass der kürzlich gesendete Verbindungskan-

ten-Suchruf noch nicht quittiert worden ist, und zeigt an, dass die Kante zu diesem Nachbarn reserviert ist, wenn der verbleibende Weiterleitungsabstand kleiner als oder gleich eins ist. Reserviert wird sie, weil der ausgewählte Nachbar diese Kante dem Anforderungsprozess für das Edge Pinning anbieten kann. Anschließend kehrt die Routine wieder zurück.

[0133] [Fig. 19](#) ist ein Flussdiagramm, welches die Verarbeitung der Kantenvorschlagsrufabwicklungs-Routine darstellt. Diese Routine wird aufgerufen, wenn eine Nachricht von einem Vorschlagsprozess empfangen wird, der die Verbindung einer Kante zwischen dem vorgeschlagenen Prozess und einem seiner Nachbarn dieses Prozesses für das Edge Pinning vorschlägt. Wenn die Anzahl der Löcher dieses Prozesses abzüglich der Anzahl der wartenden Kanten im Entscheidungsblock **1901** größer als oder gleich eins ist, dann weist dieser Prozess noch immer Löcher auf, die zu schließen sind, und die Routine geht weiter zu Block **1902**, ansonsten zu Block **1911**. Wenn der vorgeschlagene Prozess oder sein Nachbar in dem Entscheidungsblock **1902** ein Nachbar dieses Prozesses ist, geht die Routine weiter zu Block **1911**, ansonsten zu Block **1903**. In Block **1903** gibt die Routine an, dass die Kante zwischen diesem Prozess und dem vorgeschlagenen Prozess noch nicht belegt ist. Wenn im Entscheidungsblock **1904** ein vorgeschlagener Nachbar bereits als ein vorgeschlagener Nachbar ansteht, dann geht die Routine zu Block **1911**, ansonsten zu Block **1907**. In Block **1907** sendet die Routine eine Kantenvorschlagsantwort als externe Nachricht zu dem vorgeschlagenen Prozess (d.h. `edge_proposal_resp`), was anzeigt, dass die vorgeschlagene Kante akzeptiert wird. Wenn das Senden der Nachricht im Entscheidungsblock **1908** erfolgreich war, setzt die Routine bei Block **1909** fort, ansonsten kehrt sie zurück. In Block **1909** fügt die Routine die Kante als wartende Kante hinzu. In Block **1910** ruft die Routine die Nachbarhinzufügens-Routine auf, um den vorgeschlagenen Prozess am externen Port als Nachbarn hinzuzufügen. Anschließend kehrt die Routine wieder zurück. In Block **1911** sendet die Routine eine externe Nachricht (d.h. `edge_proposal_resp`), die anzeigt, dass diese vorgeschlagene Kante nicht akzeptiert wird. Wenn die Anzahl von Löchern in dem Entscheidungsblock **1912** ungerade ist, geht die Routine weiter zu Block **1913**, ansonsten kehrt sie wieder zurück. In Block **1913** ruft die Routine die Lochauffüllroutine auf und kehrt anschließend zurück.

[0134] [Fig. 20](#) ist ein Flussdiagramm, das die Verarbeitung der Abwicklungsroutine für den Portverbindungs-ruf nach einer Ausführungsform darstellt. Diese Routine wird aufgerufen, wenn eine externe Nachricht empfangen wird, die anzeigt, dass der Sendeprozess eine Verbindung zu einem Loch dieses Prozesses aufbauen will. Wenn die Anzahl der Löcher in diesem Prozess im Entscheidungsblock **2001** größer ist als null, geht die Routine weiter zu Block **2002**, ansonsten zu Block **2003**. Wenn im Entscheidungsblock **2002** der Sendeprozess kein Nachbar ist, geht die Routine weiter zu Block **2004**, ansonsten zu Block **2003**. In Block **2003** sendet die Routine eine externe Portverbindungs-Antwortnachricht (d.h. `port_connection_resp`) an den Sendeprozess, die anzeigt, dass einer Verbindung mit diesem Prozess nicht zugestimmt wird. Anschließend kehrt die Routine wieder zurück. In Block **2004** sendet die Routine eine externe Portverbindungs-Antwortnachricht an den Sendeprozess, die anzeigt, dass eine Verbindung mit diesem Prozess genehmigt wird. Wenn das Senden der Nachricht im Entscheidungsblock **2005** erfolgreich war, geht die Routine weiter zu Block **2006**, ansonsten zu Block **2007**. In Block **2006** ruft die Routine die Nachbarhinzufügens-Routine auf, um den Sendeprozess als einen Nachbarn dieses Prozesses hinzuzufügen, und kehrt anschließend wieder zurück. In Block **2007** beendet die Routine die externe Verbindung. In Block **2008** ruft die Routine die Verbindungsanforderungs-Routine auf, um eine Prozessverbindung mit einem der Löcher dieses Prozesses anzufordern. Anschließend kehrt die Routine wieder zurück.

[0135] [Fig. 21](#) ist ein Flussdiagramm, das die Verarbeitung der Lochauffüll-Routine nach einer Ausführungsform veranschaulicht. Dieser Routine wird ein Hinweis auf den Anforderungsprozess zugeführt. Wenn dieser Prozess das Füllen eines Loches anfordert, sendet diese Routine eine interne Nachricht zu anderen Prozessen. Wenn ein anderer Prozess das Füllen eines Loches anfordert, dann ruft diese Routine die Routine auf, eine Verbindungsport-Suchanforderung abzuwickeln. In Block **2101** initialisiert die Routine eine interne Verbindungsport-Suchanweisungsnachricht (d.h. `connection_port_search_stmt`). Wenn dieser Prozess im Entscheidungsblock **2102** der anfordernde Prozess ist, dann geht die Routine weiter zu Block **2103**, ansonsten zu Block **2104**. In Block **2103** verteilt die Routine die Nachricht über die internen Ports zu den Nachbarn dieses Prozesses und kehrt anschließend zurück.

[0136] In Block **2104** ruft die Routine die Verbindungsport-Suchverarbeitungsroutine auf und kehrt anschließend zurück. [Fig. 22](#) ist ein Flussdiagramm, das die Verarbeitung der internen Dispatcheroutine nach einer Ausführungsform zeigt. Diese Routine erhält einen Hinweis auf den Nachbarn, der die interne Nachricht gesendet hat. In Block **2201** empfängt die Routine die interne Nachricht. Diese Routine identifiziert den Nachrichtentyp und ruft die geeignete Routine zum Abwickeln der Nachricht auf. In Block **2202** beurteilt die Routine, ob der geschätzte Durchmesser des Übertragungskanal ausgehend von der Information in der empfangenen

Nachricht zu ändern ist. Wenn im Entscheidungsblock **2203** festgestellt wird, dass dieser Prozess der Ursprungsprozess der Nachricht ist oder die Nachricht bereits empfangen wurde (d.h. ein Duplikat), dann ignoriert die Routine die Nachricht und geht weiter zu Block **2208**, ansonsten zu Block **2203A**. Wenn der Prozess im Entscheidungsblock **2203A** teilweise verbunden ist, setzt die Routine fort in Block **2203B**, ansonsten bei Block **2204**. In Block **2203B** fügt die Routine die Nachricht zu dem wartenden Verbindungspuffer hinzu und geht weiter zu Block **2204**. In den Entscheidungsblöcken **2204–2207** decodiert die Routine den Nachrichtentyp und ruft die geeignete Routine zum Abwickeln der Nachricht auf. Wenn es sich bei dem Nachrichtentyp im Entscheidungsblock **2204** beispielsweise um eine Übertragungsanweisung (d.h. `broadcast_stmt`) handelt, ruft die Routine dann in Block **2205** die Übertragungsnachrichtabwicklungs-Routine auf. Nach Aufruf der geeigneten Verarbeitungsroutine geht es weiter zu Block **2208**. Wenn im Entscheidungsblock **2208** der teilweise verbundene Puffer voll ist, geht die Routine weiter zu Block **2209**, ansonsten zu Block **2210**. Die Senderkomponente sammelt sämtliche internen Nachrichten in einem Puffer, während dieser teilweise verbunden ist, sodass er die Nachrichten weiterleiten kann, wenn er mit neuen Nachbarn verbunden ist. Wenn jedoch jener Puffer voll wird, geht der Prozess davon aus, dass er jetzt vollständig verbunden ist und dass die erwartete Anzahl von Verbindungen zu groß war, da sich der Übertragungskanal jetzt im Small Regime befindet. In Block **2209** ruft die Routine die Verbindungsherstellungs-Routine auf und geht anschließend weiter zu Block **2210**. Wenn im Entscheidungsblock **2210** die Anwendungsprogramm-Nachrichtenwarteschlange leer ist, kehrt die Routine zurück, ansonsten geht es weiter zu Block **2212**. In Block **2212** ruft die Routine die Antwortempfangsroutine auf, die die erhaltene Nachricht weiterleitet und kehrt wieder zurück. Die Antwortempfangsroutine ist eine Rückruf-Routine des Anwendungsprogramms.

[0137] [Fig. 23](#) ist ein Flussdiagramm, das die Verarbeitung der Sendenachrichtabwicklungs-Routine nach einer Ausführungsform zeigt. Dieser Routine wird ein Hinweis auf den Ursprungsprozess zugeführt, ein Hinweis auf den Nachbarn, der die Sendenachricht gesendet hat, und die Sendenachricht selbst. In Block **2301** führt die Routine die Verarbeitung dieser Nachricht in ungeordneter Reihenfolge durch. Die Senderkomponente stellt Nachrichten von jedem Ursprungsprozess solange in eine Warteschlange, bis sie sie in der richtigen Reihenfolge an das Anwendungsprogramm senden kann. In Block **2302** ruft die Routine die Sendenachrichtverteilungs-Routine auf, um die Nachricht zu den Nachbarn dieses Prozesses weiterzuleiten. Wenn im Entscheidungsblock **2303** ein neu angeschlossener Nachbar auf den Empfang von Nachrichten wartet, dann geht die Routine weiter zu Block **2304**, ansonsten kehrt sie wieder zurück. In Block **2304** sendet die Routine die Nachrichten, falls möglich, für jeden Ursprungsprozess in der richtigen Reihenfolge und kehrt anschließend zurück.

[0138] [Fig. 24](#) ist ein Flussdiagramm, das die Verarbeitung der Sendenachrichtverteilungs-Routine nach einer Ausführungsform veranschaulicht. Diese Routine sendet die Übertragungsnachricht zu jedem Nachbarn dieses Prozesses mit Ausnahme des Nachbarn, der die Nachricht zu diesem Prozess gesendet hat. In Block **2401** wählt die Routine den nächsten Nachbarn aus, bei dem es sich um einen anderen Nachbarn handelt als den, der die Nachricht gesendet hat. Wenn im Entscheidungsblock **2402** festgestellt wird, dass bereits all diese Nachbarn ausgewählt wurden, kehrt die Routine wieder zurück. In Block **2403** sendet die Routine die Nachricht zu dem ausgewählten Nachbarn und kehrt schleifenförmig zu Block **2401** zurück, um den nächsten Nachbarn auszuwählen.

[0139] [Fig. 26](#) ist ein Flussdiagramm, das die Verarbeitung der Abwicklungsroutine für die Verbindungsport-Suchanweisung nach einer Ausführungsform darstellt. Diese Routine erhält einen Hinweis auf den Nachbarn, der die Nachricht gesendet hat, und die Nachricht selbst. In Block **2601** ruft die Routine die Verteilungsroutine für die interne Nachricht auf, welche die Nachricht zu jedem der Nachbarn bis auf den Absender sendet. Wenn im Entscheidungsblock **2602** die Anzahl von Löchern dieses Prozesses größer als null ist, dann geht die Routine weiter zu Block **2603**, ansonsten kehrt sie an den Anfang zurück. Wenn der anfordernde Prozess im Entscheidungsblock **2603** ein Nachbar ist, geht die Routine weiter zu Block **2605**, ansonsten zu Block **2604**. In Block **2604** ruft die Routine die Court Neighbor-Routine auf und kehrt anschließend wieder an den Anfang zurück. Falls möglich, verbindet die Court Neighbor-Routine diesen Prozess mit dem anfordernden Prozess. Wenn dieser Prozess in Block **2605** ein Loch hat, dann sind Nachbarn mit leeren Ports vorhanden, und die Routine geht weiter zu **2606**, ansonsten kehrt sie an den Anfang zurück. In Block **2606** erzeugt die Routine eine Zustandsprüfnachricht (d.h. `condition_check`), die eine Liste der Nachbarn dieses Prozesses enthält. In Block **2607** sendet die Routine die Nachricht zu den anfordernden Nachbarn.

[0140] [Fig. 27](#) ist ein Flussdiagramm, dass die Verarbeitung der Court Neighbor-Routine nach einer Ausführungsform zeigt. Diese Routine erhält einen Hinweis über den potenziellen Nachbarn für diesen Prozess. Wenn dieser Prozess eine Verbindung zu dem potenziellen Nachbarn aufbauen kann, sendet er eine externe Port-Verbindungsrufrnachricht zu dem potenziellen Nachbarn und fügt diesen als Nachbarn hinzu. Wenn der potenzielle Nachbar im Entscheidungsblock **2701** bereits Nachbar ist, kehrt die Routine wieder zurück, an-

sonsten setzt sie bei Block **2702** fort. In Block **2702** wählt die Routine den potenziellen Nachbarn an. Wenn die Anzahl der Löcher dieses Prozesses im Entscheidungsblock **2703** größer als null ist, geht die Routine weiter zu Block **2704**, ansonsten zu Block **2706**. In Block **2704** sendet die Routine eine externe Port-Verbindungsanrufnachricht (d.h. `port_connection_call`) an den potenziellen Nachbarn und empfängt dessen Antwort (d.h. `port_connection_resp`). Angenommen, die Antwort wird erfolgreich empfangen, dann fügt die Routine in Block **2705** den potenziellen Nachbarn als Nachbarn dieses Prozesses hinzu, indem eine Nachbarhinzufügens-Routine aufgerufen wird. In Block **2706** beendet die Routine die Verbindung mit den potenziellen Nachbarn und kehrt danach zurück.

[0141] [Fig. 28](#) ist ein Flussdiagramm, das die Verarbeitung der Verbindungskantensuchanrufabwicklungs-Routine nach einer Ausführungsform zeigt. Diese Routine erhält einen Hinweis auf den Nachbarn, der die Nachricht gesendet hat, und die Nachricht selbst. Diese Routine leitet die Nachricht entweder weiter zu einem Nachbarn oder schlägt dem anfordernden Prozess diese Kante zwischen dem vorliegenden Prozess und dem sendenden Nachbarn für das Edge Pinning vor. Wenn es sich im Entscheidungsblock **2801** nicht um den anfordernden Prozess handelt oder die Anzahl von Löchern des anfordernden Prozesses noch immer größer oder gleich zwei ist, geht die Routine weiter zu Block **2802** ansonsten zu Block **2813**. Wenn der Weiterleitungsabstand im Entscheidungsblock **2802** größer ist als null, dann ist dieser Zufallslauf nicht vollständig und die Routine geht zu Block **2803**, ansonsten zu Block **2804**. In Block **2803** ruft die Routine die Verbindungskantensuchweiterleitungs-Routine auf, die die Kennung des anfordernden Prozesses und den heruntergezählten Weiterleitungsabstand angibt. Danach setzt die Routine bei Block **2815** fort. Wenn im Entscheidungsblock **2804** der anfordernde Prozess ein Nachbar oder die Kante zwischen diesem Prozess und dem sendenden Nachbar reserviert ist, da sie bereits einem Prozess angeboten worden ist, dann geht die Routine zu Block **2805**, ansonsten zu Block **2806**. In Block **2805** ruft die Routine die Verbindungskantensuchweiterleitungs-Routine auf, die einen Hinweis auf die anfordernde Seite und einen Umschaltindikator enthält, der alternativ anzeigt, dass der Zufallslauf für einen oder mehrere Computer fortgesetzt werden soll. Anschließend setzt die Routine bei Block **2815** fort. In Block **2806** wählt die Routine den anfordernden Prozess über den Call-in-Port an. In Block **2807** sendet die Routine eine externe Kantenvorschlagsanrufnachricht (d.h. `edge_proposal_call`) und empfängt die Antwort (d.h. `edge_proposal_resp`). Unter der Annahme, dass die Antwort erfolgreich empfangen wurde, geht die Routine weiter zu Block **2808**. Wenn die Antwort im Entscheidungsblock **2808** anzeigt, dass die Kante für den anfordernden Prozess akzeptabel ist, geht die Routine weiter zu Block **2809**, ansonsten zu Block **2812**. In Block **2809** reserviert die Routine die Kante zwischen diesem Prozess und dem sendenden Nachbarn. In Block **2810** fügt die Routine den anfordernden Prozess als Nachbarn hinzu, indem die Nachbarhinzufügens-Routine aufgerufen wird. In Block **2811** entfernt die Routine den sendenden Nachbarn als Nachbarn. In Block **2812** beendet die Routine die Verbindung mit dem externen Port und geht weiter zu Block **2815**. Wenn dieser Prozess im Entscheidungsblock **2813** der anfordernde Prozess ist und die Anzahl von Löchern dieses Prozesses gleich eins ist, geht die Routine zu Block **2814**, ansonsten zu Block **2815**. In Block **2814** ruft die Routine die Lochauffüll-Routine auf. In Block **2815** sendet die Routine eine Verbindungskantensuch-Antwortnachricht (d.h. `connection_edge_search_response`) an den sendenden Nachbarn, die eine Bestätigung angibt, und kehrt danach zurück. Die Graphen sind paritätsempfindlich. D.h. alle möglichen Pfade, die von einem Knoten ausgehen und an jenem Knoten enden, haben eine gleiche Länge, falls nicht der Graph einen Kreis aufweist, dessen Länge ungerade ist. Die Senderkomponente verwendet einen Umschaltindikator, um den Zufallslaufabstand zwischen geraden und ungeraden Abständen zu verändern.

[0142] [Fig. 29](#) ist ein Flussdiagramm, das die Verarbeitung der Verbindungskantensuchantwortabwicklungs-Routine nach einer Ausführungsform darstellt. Diese Routine erhält einen Hinweis auf den anfordernden Prozess, den sendenden Nachbarn und die Nachricht. In Block **2901** vermerkt die Routine, dass die Verbindungskanten-Suchantwort (d.h. `connection_edge_search_resp`) empfangen wurde, und wenn der Weiterleitungsabstand kleiner oder gleich eins ist, hebt sie die Reservierung der Kante zwischen diesem Prozess und dem sendenden Nachbarn auf. Wenn der anfordernde Prozess im Entscheidungsblock **2902** anzeigt, dass die Kante akzeptabel ist, wie in der Nachricht angegeben, dann setzt die Routine bei Block **2903** fort, ansonsten kehrt sie zurück. In Block **2903** reserviert die Routine die Kante zwischen diesem Prozess und dem sendenden Nachbarn. In Block **2904** entfernt die Routine den sendenden Nachbarn als einen Nachbarn. In Block **2905** ruft die Routine die Court Neighbor-Routine auf, um eine Verbindung mit dem anfordernden Prozess herzustellen. Wenn die aufgerufene Routine im Entscheidungsblock **2906** erfolglos war, geht die Routine weiter zu Block **2907**, ansonsten kehrt sie wieder zurück. Wenn die Anzahl der Löcher in diesem Prozess im Entscheidungsblock **2907** größer als null ist, geht die Routine weiter zu Block **2908**, ansonsten kehrt sie zurück. In Block **2908** ruft die Routine die Lochschließroutine auf und kehrt zurück.

[0143] [Fig. 30](#) ist ein Ablaufdiagramm, das die Verarbeitung der Übertragungs-Routine nach einer Ausführungsform zeigt. Diese Routine wird von den Anwendungsprogrammen aufgerufen, um eine Nachricht auf

dem Übertragungskanal zu übertragen. Dieser Routine wird die zu sendende Nachricht zugeführt. Wenn in dem Entscheidungsblock **3001** dieser Prozess wenigstens einen Nachbarn hat, geht die Routine weiter zu Block **3002**, ansonsten kehrt sie an den Anfang zurück, weil es der einzige Prozess ist, der mit dem Übertragungskanal verbunden ist. In Block **3002** erzeugt die Routine eine interne Nachricht über den Typ der Senderanweisung (d.h. `broadcast_stmt`). In Block **3003** legt die Routine die Sequenznummer der Nachricht fest. In Block **3004** ruft die Routine die Verteilungsroutine für die internen Nachrichten auf, um die Nachricht auf dem Übertragungskanal zu übertragen. Anschließend kehrt die Routine an den Anfang zurück.

[0144] [Fig. 31](#) ist ein Flussdiagramm, das die Verarbeitung der Nachrichtenerfassungs-Routine nach einer Ausführungsform zeigt. Die Nachrichtenerfassungs-Routine kann von dem Anwendungsprogramm oder von einer Rückruf-Routine, die von dem Anwendungsprogramm geschaffen wird, aufgerufen werden. Diese Routine gibt eine Nachricht aus. In Block **3101** ruft die Routine die Nachricht aus der Nachrichtenwarteschlange des Übertragungskanals ab. Wenn die Nachricht abgerufen wurde, gibt die Routine in dem Entscheidungsblock **3102** eine Erfolgsmeldung aus, ansonsten gibt die Routine eine Fehlermeldung aus.

[0145] Die [Fig. 32–Fig. 34](#) sind Flussdiagramme, die die Verarbeitung von Nachrichten veranschaulichen, die zu Nachbarn mit leeren Ports gehören. [Fig. 32](#) ist ein Flussdiagramm, das die Verarbeitung der Zustandsprüfnachricht nach einer Ausführungsform zeigt. Diese Nachricht wird von einem Nachbarprozess gesendet, der ein Loch hat und eine Anforderung zum Verbinden eines Loches dieses Prozesses empfangen hat. Wenn die Anzahl an Löchern dieses Prozesses im Entscheidungsblock **3201** gleich eins ist, dann geht die Routine weiter zu Block **3202**, ansonsten sind keine Nachbarn mit leeren Ports mehr vorhanden und die Routine kehrt an den Anfang zurück. Wenn der sendende Nachbar und dieser Prozess die gleiche Gruppe von Nachbarn haben, was im Entscheidungsblock **3202** festgestellt wird, geht die Routine weiter zu Block **3203**, ansonsten zu Block **3205**. In Block **3203** initialisiert die Routine eine Zustandsnachprüfnachricht (d.h. `condition_double_check`) mit der Liste von Nachbarn dieses Prozesses. In Block **3204** sendet die Routine die Nachricht intern zu einem anderen Nachbarn als dem sendenden Nachbarn. Danach kehrt die Routine an den Anfang zurück. In Block **3205** wählt die Routine einen Nachbarn des Sendeprozesses aus, bei dem es sich nicht um einen Nachbarn dieses Prozesses handelt. In Block **3206** sendet die Routine eine Zustandsbehebungsnachricht (d.h. `condition_repair_stmt`) extern an den ausgewählten Prozess. In Block **3207** ruft die Routine die Nachbarhinzufügens-Routine auf, um den ausgewählten Nachbarn als einen Nachbarn dieses Prozesses hinzuzufügen, und kehrt anschließend zurück.

[0146] [Fig. 33](#) ist ein Flussdiagramm, das die Verarbeitung der Zustandsbehebungsanweisungsabwicklungs-Routine nach einer Ausführungsform zeigt. Diese Routine entfernt einen vorhandenen Nachbarn und stellt die Verbindung zu dem Prozess her, der die Nachricht gesendet hat. Wenn dieser Prozess im Entscheidungsblock **3301** keine Löcher hat, geht die Routine weiter zu Block **3302**, ansonsten zu Block **3304**. In Block **3302** wählt die Routine einen Nachbarn aus, der nicht unter den Nachbarn mit leeren Ports ist. In Block **3303** entfernt die Routine den ausgewählten Nachbarn als Nachbarn dieses Prozesses. Somit hat der Prozess, der die Routine ausführt, jetzt wenigstens ein Loch. In Block **3304** ruft die Routine die Nachbarhinzufügens-Routine auf, um den Prozess, der die Nachricht gesendet hat, als Nachbarn dieses Prozesses hinzuzufügen. Danach kehrt die Routine wieder zurück.

[0147] [Fig. 34](#) ist ein Flussdiagramm, das die Verarbeitung der Zustandsnachprüfabwicklungs-Routine veranschaulicht. Diese Routine legt fest, ob die Nachbarn mit leeren Ports wirklich ein Problem darstellen oder ob sich der Übertragungskanal im Small Regime befindet. Wenn dieser Prozess im Entscheidungsblock **3401** ein Loch hat, geht die Routine weiter zu Block **3402**, ansonsten zu Block **3403**. Hat dieser Prozess kein Loch, dann ist die Gruppe der Nachbarn dieses Prozesses nicht identisch mit jener des sendenden Prozesses. Wenn dieser Prozess und der Sendeprozess im Entscheidungsblock **3402** die gleiche Gruppe von Nachbarn haben, befindet sich der Übertragungskanal nicht im Small Regime und die Routine geht weiter zu Block **3403**, ansonsten zu Block **3406**. Wenn dieser Prozess keine Löcher hat, kehrt die Routine im Entscheidungsblock **3403** wieder zurück, ansonsten geht es weiter bei Block **3404**. In Block **3404** setzt die Routine den geschätzten Durchmesser für diesen Prozess auf eins. In Block **3405** sendet die Routine eine interne Durchmesserzurücksetznachricht (d.h. `diameter_reset`), die angibt, dass der geschätzte Durchmesser eins ist, und kehrt dann wieder zurück. In Block **3406** erzeugt die Routine eine Liste der Nachbarn dieses Prozesses. In Block **3407** sendet die Routine die Zustandsprüfnachricht (d.h. `condition_check_stmt`) mit der Liste der Nachbarn an den Nachbarn, der die Zustandsnachprüfnachricht gesendet hat, und kehrt anschließend zurück.

[0148] Aus der obigen Beschreibung wird deutlich, dass zwar spezifische Ausführungsformen der Technologie dargelegt wurden, jedoch verschiedene Modifizierungen durchgeführt werden können, ohne vom Sinn und Schutzzumfang der Erfindung abzuweichen. Z.B. können die Datenübertragungen auf dem Übertragungskanal

verschlüsselt sein. Weiterhin kann die Kanalinstanz bzw. die Sitzungskennung eine sehr große Zahl sein (z.B. 128 Bits), um dazu beizutragen, dass nicht ein unberechtigter Nutzer einen Übertragungskanal in boshafter Absicht anzapft. Der Portal-Computer kann ebenfalls die Sicherheit verstärken und verhindern, dass sich ein nicht berechtigter Benutzer an den Übertragungskanal anschließt. Dementsprechend unterliegt die Erfindung keinerlei Einschränkungen bis auf jene aus den Ansprüchen.

Verteiltes Auktionssystem

[0149] Es werden ein Verfahren und ein System zum Ausführen elektronischer Auktionen mit einem verteilten Auktionator geschaffen. Bei einer Ausführungsform enthält der Computer jedes Teilnehmers eine Auktionator-komponente zum Eröffnen von Auktionen, zum Annehmen von Geboten und zum Schließen von Auktionen. Folglich hängt das Auktionssystem nicht von einem zentralen Auktionsserver zum Koordinieren des Bietvorgangs bei einer Auktion ab. Das Auktionssystem ist gewissermaßen serverlos. Mit einem Übertragungskanal kommuniziert das Auktionssystem zwischen den Teilnehmern einer Auktion. Der Computer jedes Teilnehmers ist an den Übertragungskanal angeschlossen und lässt ein Auktionsteilnehmerprogramm ablaufen. Das Auktionsteilnehmerprogramm ermöglicht es einem Teilnehmer, ein Gebot auf einen zu versteigernden Artikel abzugeben und zu empfangen, Gebote anderer Teilnehmer anzuzeigen und den Abschluss der Auktion zu koordinieren. Wenn ein Teilnehmer ein Gebot auf einen Artikel abgibt, der versteigert wird, sendet das Auktionsteilnehmerprogramm eine Gebotsnachricht über den Übertragungskanal. Jedes Auktionsteilnehmerprogramm, das mit dem Übertragungskanal verbunden ist, empfängt die Gebotsnachricht und zeigt dem Teilnehmer das aktuelle Höchstgebot an. Das Auktionsteilnehmerprogramm, dessen Teilnehmer das höchste Gebot abgegeben hat, koordiniert den Abschluss der Auktion entsprechend den Abschlussregeln. Wenn beispielsweise das Auktionsteilnehmerprogramm feststellt, dass der jeweilige Teilnehmer über eine bestimmte Zeit nicht mehr überboten wurde, kann das Auktionsteilnehmerprogramm die Nachricht „Zum Ersten“ senden. Diese Nachricht „Zum Ersten“ entspricht einem Auktionator, der die Teilnehmer warnt, dass die Auktion dem Ende entgegengeht. Stellt das Auktionsteilnehmerprogramm fest, dass der entsprechende Teilnehmer nach dem Senden der Nachricht „Zum Ersten“ eine bestimmte zeitlang nicht mehr überboten wurde, kann das Auktionsteilnehmerprogramm die Nachricht „Zum Dritten“ senden. Wenn die Auktionsteilnehmerprogramme die Nachricht „Zum Dritten“ empfangen, können sie ihre Teilnehmer informieren, dass die Auktion geschlossen ist. Das Auktionsteilnehmerprogramm, dessen Teilnehmer das höchste Gebot abgegeben hat, setzt sich anschließend mit einem Auktionslisting-Server in Verbindung, um das Geschäft abzuschließen. Da die Teilnehmer über einen Übertragungskanal angeschlossen sind, empfängt jeder Teilnehmer eine Benachrichtigung über jedes Gebot, wenn es abgegeben wird. Darüber hinaus hängt die Zuverlässigkeit des Auktionssystems nicht von einem zentralen Auktionsserver ab. Wenn einer der teilnehmenden Computer ausfällt, können die anderen Teilnehmer die Auktion fortsetzen. Bei einer Ausführungsform ist das Auktionssystem unter Verwendung des hier beschriebenen Übertragungskanals implementiert. Fachleuten ist jedoch klar, dass das Auktionssystem auch mit anderen zugrunde liegenden Kommunikationsnetzwerken genutzt werden kann.

[0150] Das Auktionssystem kann einen Auktionslisting-Servercomputer, einen Auktions-Überwachungscomputer und Teilnehmercomputer umfassen. Der Auktionslisting-Servercomputer kann eine Webseite zur Verfügung stellen, über die Verkäufer ihre zu versteigernden Artikel einstellen können. Wenn ein Artikel gelistet ist, kann der Verkäufer ein Bild des zu versteigernden Artikels (falls angemessen) bereitstellen, das Mindestgebot für den Artikel und eine Startzeit für die Auktion. Potenzielle Bieter können auf die Webseiten des Auktionslisting-Servers zugreifen, um die aufgelisteten Auktionen anzusehen. Weiterhin können potenzielle Bieter auch das Auktionsteilnehmerprogramm von dem Auktionslisting-Server auf ihre Computer herunterladen. Wenn ein Nutzer an einer bestimmten Auktion teilnehmen möchte, lässt der Teilnehmer das Auktionsteilnehmerprogramm ablaufen, das eine Aufstellung der aktuellen Auktionen bietet, die durchgeführt werden, sowie den Stand jeder Auktion. Der Teilnehmer kann eine bestimmte Auktion auswählen und bei dieser Auktion ein Gebot abgeben. Da möglicherweise zwei Teilnehmer etwa zur gleichen Zeit denselben Betrag für einen Artikel bieten, erteilt das Auktionsteilnehmerprogramm dem Teilnehmer auf der Grundlage einer zufälligen Zahl, die von dem Auktionsteilnehmerprogramm des Bieters erzeugt wird, den Zuschlag. Wenn ein Gebot abgegeben wird, erzeugt das Auktionsteilnehmerprogramm beim Senden automatisch eine Zufallszahl und sendet diese mit. Immer wenn ein Auktionsteilnehmerprogramm ein Gebot für den gleichen Betrag als höchstes aktuelles Gebot empfängt, erteilt das Auktionsteilnehmerprogramm dem Teilnehmer mit der höchsten zufällig erzeugten Nummer den Zuschlag. Der Auktionsüberwachungs-Computer kann ebenfalls mit dem Übertragungskanal verbunden sein. Der Auktionsüberwachungs-Computer verfolgt den Status der Auktion durch Überwachung der abgegebenen Gebote. Der Auktionsüberwachungscomputer kann dem Auktionslisting-Server den Stand der Auktionen mitteilen und auch den Auktionsteilnehmerprogrammen, wenn sie sich an der Auktion beteiligen.

[0151] [Fig. 35](#) ist ein Blockdiagramm, das Komponenten des Auktionssystems nach einer Ausführungsform

zeigt. Das Auktionssystem enthält einen Auktionslisting-Server **3501**, Teilnehmercomputer **3502** und eine Auktionsüberwachungseinrichtung **3503**. Jeder Computer kann eine zentrale Verarbeitungseinheit, einen Speicher, Eingabevorrichtungen (z.B. eine Tastatur und Zeigervorrichtung), Ausgabevorrichtungen (z.B. Anzeigevorrichtungen) und Speichervorrichtungen (z.B. Plattenspeicherlaufwerke) umfassen. Der Speicher und die Speichervorrichtungen sind computerlesbare Medien, die Computerbefehle enthalten können, welche das Auktionssystem implementieren. Des Weiteren können die computerlesbaren Medien Computerdaten-Übertragungsmedien aufweisen, z.B. verdrahtete oder drahtlose Kommunikationsmechanismen. Die teilnehmenden Computer können einen Browser für den Zugriff auf Webseiten aufweisen, die von dem Auktionslisting-Server bereitgestellt werden. Die teilnehmenden Computer und die Auktionsüberwachungseinrichtung sind mit dem Übertragungskanal **3505** verbunden. Die teilnehmenden Computer, die Auktionsüberwachungseinrichtung und der Auktionslisting-Server sind über das Internet **3504** miteinander verbunden. Die teilnehmenden Computer können einen Browser nutzen, um auf die von dem Auktionslisting-Server bereitgestellten Auktionsinformationen zuzugreifen. Der Auktionslisting-Server kann eine Web-Maschine **3506**, eine Auktionserzeugungskomponente **3507**, eine Auktionsbeendigungskomponente **3508** und eine Auktionsdatenbank **3509** enthalten. Die Auktionserzeugungskomponente wird von einem Verkäufer für die Schaffung einer Versteigerung für einen Artikel verwendet. Die Auktionsbeendigungskomponente wird von einem Höchstbietenden verwendet, um die Zahlung für den gekauften Artikel abzuwickeln. Des Weiteren kann der Auktionslisting-Server eine Komponente zum Registrieren von Teilnehmern und eine Teilnehmerdatenbank aufweisen. Die Auktionsdatenbank definiert die Auktionen und kann den aktuellen Status der Auktion enthalten, der von der Auktionsüberwachungseinrichtung bereitgestellt wird. Für Fachleute liegt es auf der Hand, dass verschiedene andere Kommunikationsmechanismen von dem Auktionssystem genutzt werden können. Z.B. kann der Übertragungskanal auch mithilfe des Internets selbst implementiert werden. Darüber hinaus können mehrere Auktionen gleichzeitig auf dem Übertragungskanal ausgeführt werden. In derartigen Fällen enthält jede Nachricht, die gesendet wird, eine Auktionskennung. Als Alternative dazu kann jede Auktion ihren eigenen Übertragungskanal haben. Dann stellt der Auktionslisting-Server jedem Auktionsteilnehmerprogramm Übertragungskanalinformationen (z.B. Anwendungs- und Sitzungskennung) zur Verfügung. Die Nachrichten können verschlüsselt oder anderweitig gesichert sein, so dass nur ein berechtigtes Auktionsteilnehmerprogramm an einer Auktion teilnehmen kann.

[0152] [Fig. 36](#) ist ein Blockdiagramm, das die Komponenten des Computers eines Teilnehmers nach einer Ausführungsform zeigt. Der teilnehmende Computer enthält eine Senderkomponente **3601**, ein Auktionsteilnehmerprogramm **3602** und eine Auktionsdatenbank **3603**. Die Senderkomponente steuert die Verbindung zu dem Übertragungskanal, das Senden von Nachrichten zum Übertragungskanal und das Empfangen von Nachrichten von ihm. Das Auktionsteilnehmerprogramm steuert die Teilnahme an einer Auktion durch Senden von Nachrichten zu dem Übertragungskanal und empfangen von Nachrichten vom Übertragungskanal mithilfe der Senderkomponente. Die Auktionsdatenbank enthält aktuelle Statusinformationen über die Auktionen. Das Auktionsteilnehmerprogramm enthält eine Überwachungs-Teilkomponente **3605**, eine Nachrichtenbearbeitungseinheit **3606**, eine Anzeigestatus-Teilkomponente **3607** und eine Teileinheitsgebots-Teilkomponente **3608**. Die Überwachungs-Teilkomponente überwacht die Nachrichten, die auf dem Übertragungskanal gesendet werden, und ruft die geeignete Nachrichtenabwicklungsroutine auf. Die Anzeigestatus-Teilkomponente zeigt den aktuellen Status der Auktionen an. Die Angebotsabgabe-Teilkomponente wird aufgerufen, wenn ein Teilnehmer ein Gebot bei einer Versteigerung abgeben will.

[0153] [Fig. 37](#) ist ein Blockdiagramm, das eine Anzeige aktuell definierter Auktionen darstellt. Das Fenster **3700** wird von einer Anzeigestatusroutine angezeigt. Für jede Auktion enthält das Fenster ein Teilfenster **3701**. Jedes Teilfenster kann Informationen über die Auktion enthalten. Wenn ein Benutzer ein Teilfenster auswählt, zeigt die Anzeigestatusroutine ein auktionsspezifisches Fenster an.

[0154] [Fig. 38](#) ist ein Diagramm, das die Anzeige eines auktionsspezifischen Fensters verdeutlicht. Das Fenster **3800** enthält ein Artikelbild **3801**, einen Artikelbeschreibungsbereich **3802**, einen Auktionsbeschreibungsbereich **3803** und eine Gebotsabgabe-Schaltfläche **3804**. Der Artikelbeschreibungsbereich enthält eine Beschreibung des Artikels, der versteigert wird. Der Auktionsbeschreibungsbereich enthält Informationen über den aktuellen Status der Auktion. Z.B. kann der tatsächliche Auktionsstatus der Anfang der Auktion sein, ein Hinweis, dass die Auktion gerade läuft, ein Hinweis, dass die Auktion dem Ende entgegengeht („Zum Ersten“) und ein Hinweis, dass die Auktion beendet ist. Der Auktionsbeschreibungsbereich kann weiterhin das Mindestgebot, das aktuelle Gebot und den vorgeschlagenen Gebotsbetrag enthalten, der überboten werden kann. Wenn der Teilnehmer die Gebotsabgabeschaltfläche auswählt, gibt das Auktionsteilnehmerprogramm das Gebot ab.

[0155] Die [Fig. 39–Fig. 48](#) sind Flussdiagramme, die die Verarbeitung des Auktionsteilnehmerprogramms

veranschaulichen. Die Verarbeitung dieser Flussdiagramme ist anhand einer einzelnen Auktion dargestellt. Für Fachleute ist es klar, dass die Verarbeitung so abgewandelt werden könnte, dass sich auch mehrere Auktionen gleichzeitig abwickeln lassen. [Fig. 39](#) ist ein Flussdiagramm einer Routine zur Abfrage des aktuellen Status der Auktion. Diese Routine kann aufgerufen werden, wenn das Auktionsteilnehmerprogramm zum ersten Mal in Gang gesetzt wird. Wenn das Auktionsteilnehmerprogramm startet, kann es sich mit dem Auktionslisting-Server in Verbindung setzen, um den aktuellen Status der Auktion abzurufen. Als Alternative dazu kann die Statusabfrageroutine wie in Block **3901** eine Anforderungsnachricht über den aktuellen Status auf dem Übertragungskanal übertragen. Als Antwort darauf empfängt das Auktionsteilnehmerprogramm eine Meldung über den aktuellen Status der Auktion. Das Auktionsteilnehmerprogramm speichert diese Statusinformation in seiner Auktionsdatenbank. [Fig. 40](#) ist ein Flussdiagramm der Routine, die eine Anforderungsnachricht über den aktuellen Status empfängt und verarbeitet. Jedes Auktionsteilnehmerprogramm kann diese Anforderung außer Acht lassen, wenn die Auktionsüberwachungseinheit so ausgelegt ist, dass sie auf diese Anforderung reagiert. Als Alternative dazu kann das Auktionsteilnehmerprogramm mit dem aktuell höchsten Gebot in der Auktion durch Senden einer Nachricht reagieren, die den aktuellen Stand der Auktion enthält. Wenn dieser Teilnehmer in dem Entscheidungsblock **4001** aktuell das höchste Gebot hat, geht die Routine weiter zu Block **4002**, ansonsten kehrt sie an den Anfang zurück. In Block **4002** sendet die Routine den Status der Auktion und mehr anschließend zurück. [Fig. 41](#) ist ein Flussdiagramm einer Routine, die die aktuelle Statusnachricht empfängt. In Block **4101** aktualisiert die Routine den Auktionsstatus in der Auktionsdatenbank und kehrt wieder zurück.

[0156] [Fig. 42](#) ist ein Flussdiagramm, das die Verarbeitung der Gebotsabgaberroutine nach einer Ausführungsform verdeutlicht. Diese Routine validiert den Betrag des Gebots und sendet anschließend jenes Gebot. Weiterhin stellt die Routine eine Zeitschaltuhr, um anzuzeigen, wann eine Nachricht zum Beenden der Versteigerung gesendet werden sollte und andere Teilnehmer darüber informiert, dass die Auktion endet, wenn kein Teilnehmer ein höheres Gebot abgibt. Wenn das Gebot im Entscheidungsblock **4201** gültig ist, geht die Routine weiter zu Block **4202**, ansonsten kehrt die Routine zurück. Ob ein Gebot gültig ist, stellt die Routine fest, indem sie sicherstellt, dass das Gebot höher ist als das aktuelle Höchstgebot. Die Routine kann auch überprüfen, ob die Auktion noch läuft. Die Auktion kann beendet worden sein, nachdem der Teilnehmer die Gebotsabgabeschaltfläche ausgewählt hatte. In Block **4202** erzeugt die Routine eine Zufallszahl, die in die Gebotsnachricht aufgenommen wird. Diese Zufallszahl wird von den empfangenden Teilnehmern für den Fall genutzt, dass zwei Gebote mit demselben Betrag von jenen Teilnehmern eingegangen sind. Sollte dies der Fall sein, erteilen die Teilnehmer dem Bieter mit der höchsten Zufallszahl den Zuschlag. In Block **4203** erzeugt die Routine eine Gebotsnachricht, die die Kennung des Teilnehmers, die Gebotshöhe und die Zufallszahl enthält. Wenn auf dem Übertragungskanal gerade Nachrichten für mehrere Auktionen übertragen werden, kann die Gebotsnachricht auch die Auktionskennung enthalten. In Block **4204** überträgt die Routine die Gebotsnachricht auf dem Übertragungskanal. In Block **4205** startet die Routine eine Zeitschaltuhr zum Senden der Nachricht „Zum Ersten“. Anschließend kehrt die Routine an den Anfang zurück.

[0157] [Fig. 43](#) ist ein Flussdiagramm, das die Verarbeitung der Gebotsempfangsnachricht-Routine nach einer Ausführungsform veranschaulicht. Diese Routine wird aufgerufen, wenn das Auktionsteilnehmerprogramm eine Gebotsnachricht von dem Übertragungskanal empfängt. Zur Routine validiert das Gebot, aktualisiert den Auktionsstatus und löscht alle Zeitvorgaben. Wenn im Entscheidungsblock **4301** die Auktion gerade läuft, geht die Routine weiter zu Block **4302**, ansonsten kehrt sie wieder zurück. Wenn im Entscheidungsblock **4302** das empfangene Gebot größer ist als oder genauso groß wie das aktuelle Höchstgebot, geht die Routine weiter zu Block **4303**, ansonsten ist das empfangene Gebot bereits überboten worden und die Routine kehrt an den Anfang zurück. Wenn im Entscheidungsblock **4303** das empfangene Gebot genauso hoch ist wie das gegenwärtige Höchstgebot, dann haben zwei Teilnehmer denselben Betrag geboten und die Routine setzt bei Block **4304** fort, ansonsten bei Block **4305**. Wenn im Entscheidungsblock **4304** die Zufallszahl in der empfangenen Gebotsnachricht größer ist als die Zufallszahl, die in der Gebotsnachricht mit dem aktuellen Höchstgebot enthalten war, dann erhält jener Teilnehmer den Zuschlag, der die Gebotsnachricht gesendet hat, und die Routine geht weiter zu Block **4305**, ansonsten kehrt die Routine an den Anfang zurück. In Block **4305** ersetzt die Routine das aktuelle Höchstgebot in der Auktionsdatenbank und kann die Anzeige aktualisieren. In Block **4306** löscht die Routine alle Zeitschaltuhren, die vielleicht gestellt worden sind, um das Ende der Auktion anzuzeigen. Anschließend kehrt die Routine an den Anfang zurück.

[0158] [Fig. 44](#) ist ein Flussdiagramm, das eine Routine zeigt, die das Ablaufen der Zeitschaltuhr verarbeitet. In Block **4401** sendet die Routine eine Nachricht, dass die Auktion dem Ende entgegengeht. Diese Nachricht kann den Teilnehmer und das höchste aktuelle Gebot angeben. In Block **4402** stellt die Routine eine Zeitschaltuhr zum Senden der Nachricht „Zum Dritten“, die anzeigt, dass die Auktion jetzt beendet ist. Anschließend kehrt die Routine an den Anfang zurück. [Fig. 45](#) ist ein Flussdiagramm, das eine Routine zeigt, die eine emp-

fangene Nachricht über das angekündigte Ende der Auktion verarbeitet. Im Entscheidungsblock **4501** entspricht diese Nachricht einem Gebot, das bereits überboten worden ist, dann kehrt die Routine wieder zurück, ansonsten geht sie weiter zu Block **4502**. In Block **4502** aktualisiert die Routine den Status der Auktion, wozu auch das Aktualisieren der Anzeige gehören kann. Anschließend kehrt die Routine an den Ausgangspunkt zurück.

[0159] **Fig. 46** ist ein Flussdiagramm, das eine Routine zeigt, die das Ablaufen der Zeitschaltuhr für das Auktionsende verarbeitet. In Block **4601** sendet die Routine eine Auktionsabschlussnachricht, die den Teilnehmer angeben kann, der die Nachricht zusammen mit der Gebotshöhe sendet. In Block **4602** aktualisiert die Routine den Status der Auktion und zeigt an, dass sie beendet ist. **Fig. 47** ist ein Blockdiagramm, das eine Routine veranschaulicht, die eine empfangene Auktionsabschlussnachricht verarbeitet. In Block **4702** aktualisiert die Routine den Status der Auktion und zeigt an, dass sie beendet ist. Bei einer Ausführungsform kann das Auktionsteilnehmerprogramm ebenfalls eine Unterdrückungsnachricht vor dem Senden der Auktionsbeendigungsnachricht senden. Wenn ein Teilnehmer eine Unterdrückungsnachricht empfängt, kann er bei jener Auktion kein weiteres Gebot abgeben. Wenn ein Teilnehmer, der die Unterdrückungsnachricht gesendet hat, von einem anderen Teilnehmer über einen bestimmten Zeitraum keine weitere Gebotsnachricht empfangen hat, kann er die Auktionsabschlussnachricht senden. Empfängt ein Teilnehmer nach dem Empfang der Unterdrückungsnachricht jedoch keine Auktionsabschlussnachricht innerhalb einer bestimmten Zeitspanne, kann er davon ausgehen, dass die Auktion noch läuft.

[0160] **Fig. 48** ist ein Flussdiagramm, das einen Auktionsagenten nach einer Ausführungsform zeigt. Der Auktionsagent ist ein Programm, das es einem Teilnehmer ermöglicht, ein Höchstgebot anzugeben, das er für einen bestimmten Artikel abgeben möchte. Der Auktionsagent überwacht automatisch die Auktion und gibt Gebote im Namen des Teilnehmers bis zu dem Maximalgebot ab. Der Auktionsagent kann verschiedene Verfahren nutzen, um vor den anderen Teilnehmern geheim zu halten, dass es sich um einen automatischen Agenten handelt. Z.B. kann der Auktionsagent das Abgeben eines neuen Gebots verzögern, wenn er überboten wurde. Diese Verzögerung kann eine willkürlich gewählte Zeitspanne oder nach Regeln des Teilnehmers festgelegt sein. Darüber hinaus kann der Auktionsagent solange warten, bis er die Nachricht „Zum Ersten“ empfängt, und erst dann ein neues Gebot abgeben. In Block **4801** ruft die Routine das aktuelle Höchstgebot von der Auktionsdatenbank ab. Wenn das aktuelle Höchstgebot im Entscheidungsblock **4802** bereits größer ist als das Maximalgebot, das der Agent abgeben darf, geht die Routine weiter zu Block **4803**, ansonsten zu Block **4804**. In Block **4803** benachrichtigt die Routine den Teilnehmer, dass dieser bei der Auktion überboten wurde, und kehrt anschließend wieder zurück. In Block **4804** gibt die Routine ein Gebot ab, das das aktuelle Gebot plus einer Mindestgebotserhöhung beträgt. Das abgegebene Gebot wird auf dem Übertragungskanal übertragen. In Block **4805** wartet die Routine auf eine Nachricht, die für die Auktion gesendet werden soll. Gegebenenfalls sendet dieses Auktionsteilnehmerprogramm auch die Nachrichten „Zum Ersten“, „Zum Zweiten“ und „Zum Dritten“. Wenn die Nachricht im Entscheidungsblock **4806** angibt, dass ein neues Gebot abgegeben wurde, das über dem aktuellen Gebot liegt, dann geht die Routine weiter zu Block **4807**, ansonsten zu Block **4808**, da dieses Auktionsteilnehmerprogramm die Nachricht „Zum Dritten“ gesendet hat. In Block **4807** fügt die Routine wahlweise Verzögerungen ein und kehrt anschließend schleifenartig zu Block **4802** zurück, um ein neues Gebot abzugeben. In Block **4808** benachrichtigt die Routine den Teilnehmer darüber, dass er den Zuschlag in der Versteigerung erhalten hat, und kehrt anschließend an den Anfang zurück.

[0161] Zwar wurden zur Verdeutlichung spezielle Ausführungsformen der Erfindung beschrieben, dennoch wird aus der Beschreibung deutlich, dass verschiedene Abwandlungen vorgenommen werden können, ohne vom Schutzzumfang der Erfindung abzuweichen. Dementsprechend ist die Erfindung nur durch die beigefügten Patentansprüche begrenzt.

Patentansprüche

1. Computernetzwerk, welches eine Vielzahl von Teilnehmern (A, B, C, D, E, F, G, H, I) aufweist, wobei jeder Teilnehmer mit mindestens drei Teilnehmern verbunden ist, die als Nachbarn dieses Teilnehmers bezeichnet werden, wobei ein veranlassender Teilnehmer eingerichtet ist, Daten zu den anderen Teilnehmern durch Senden der Daten über jede seiner Verbindungen zu seinen Nachbarn zu senden, und wobei jeder Teilnehmer eingerichtet ist, von einem Nachbar empfangene Daten zu seinen anderen Nachbarn zu senden, dadurch charakterisiert, dass das Netzwerk m-regulär und m-verbunden ist, wobei m die Anzahl der Nachbar-Teilnehmer von jedem Teilnehmer darstellt.

2. Computernetzwerk nach Anspruch 1, wobei jeder Teilnehmer mit vier anderen Teilnehmern verbunden ist.

3. Computernetzwerk nach Anspruch 1 oder 2, wobei jeder Teilnehmer mit einer geradzahligen Anzahl von anderen Teilnehmern verbunden ist.
4. Computernetzwerk nach einem der Ansprüche 1 bis 3, wobei alle Teilnehmer Peers sind.
5. Computernetzwerk nach einem der Ansprüche 1 bis 4, wobei die Verbindungen Peer-to-Peer Verbindungen sind.
6. Computernetzwerk nach einem der Ansprüche 1 bis 5, wobei die Verbindungen TCP/IP Verbindungen sind.
7. Computernetzwerk nach einem der Ansprüche 1 bis 6, wobei das Computernetzwerk eingerichtet ist, einen Teil des Internets zu bilden.
8. Computernetzwerk nach einem der Ansprüche 1 bis 7, wobei jeder Teilnehmer eingerichtet ist, einem auf einem Computer ausgeführten Prozess zugeordnet zu sein.
9. Computernetzwerk nach einem der Ansprüche 1 bis 7, wobei jeder Teilnehmer eingerichtet ist, einem Computer-Thread zugeordnet zu sein.
10. Computernetzwerk nach einem der Ansprüche 1 bis 7, wobei jeder Teilnehmer ein Computer ist.
11. Computernetzwerk nach einem der Ansprüche 1 bis 7, wobei ein Computer mehr als einen Teilnehmer beherbergt.
12. Computernetzwerk nach einem der Ansprüche 1 bis 11, wobei jeder Teilnehmer eingerichtet ist, zu jedem seiner Nachbarn nur eine Kopie der Daten zu senden.
13. Computernetzwerk nach einem der Ansprüche 1 bis 12, wobei das Netzwerk eingerichtet ist, einen logischen Rundfunkkanal für jeden Teilnehmer zur Verfügung zu stellen, um Nachrichten von einem Teilnehmer zu allen anderen Teilnehmern zu übertragen.

Es folgen 53 Blatt Zeichnungen

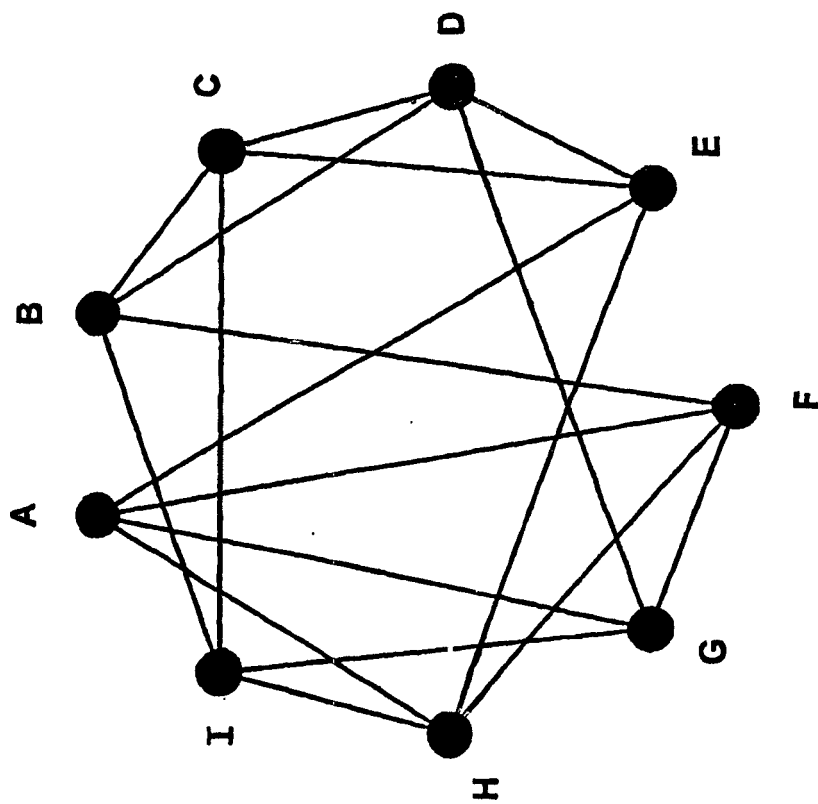


Fig. 1

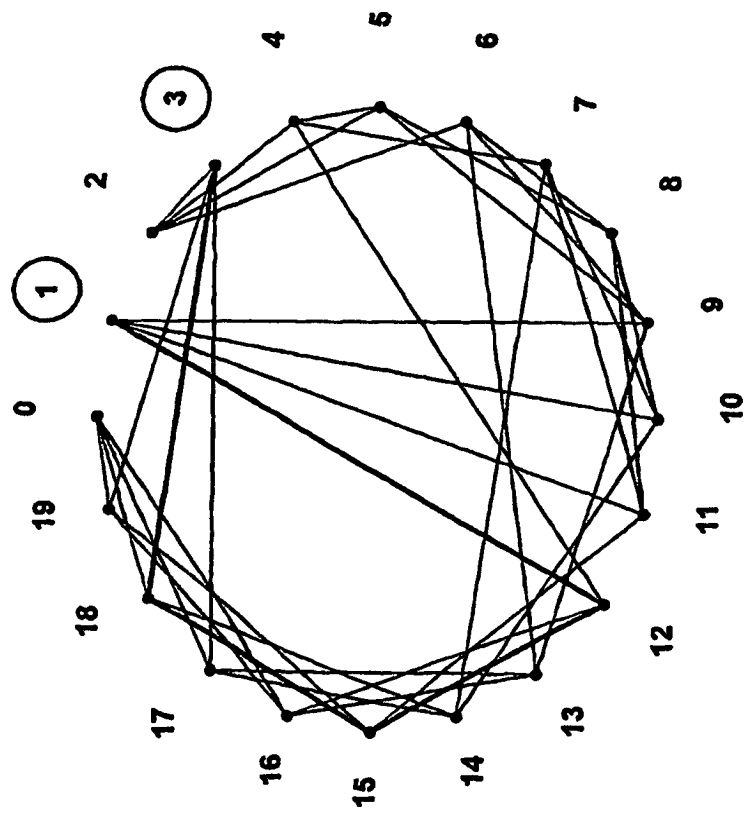


Fig. 2

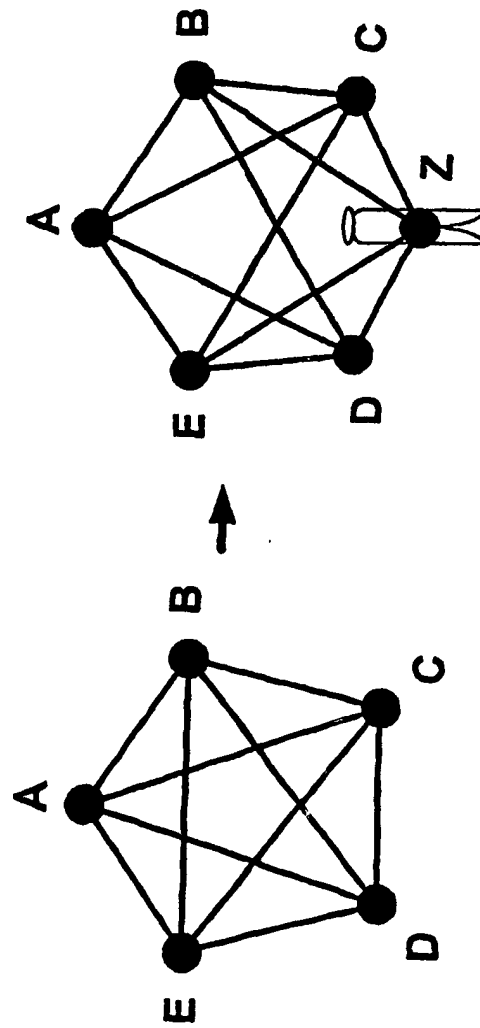


Fig. 3B

Fig. 3A

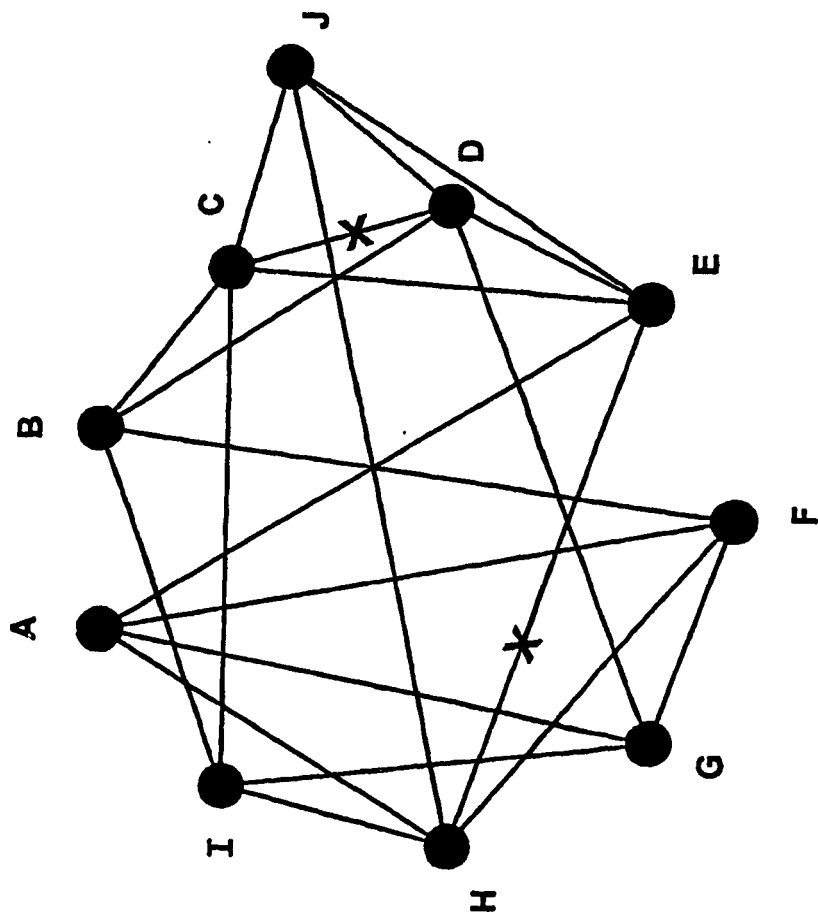


Fig. 4A

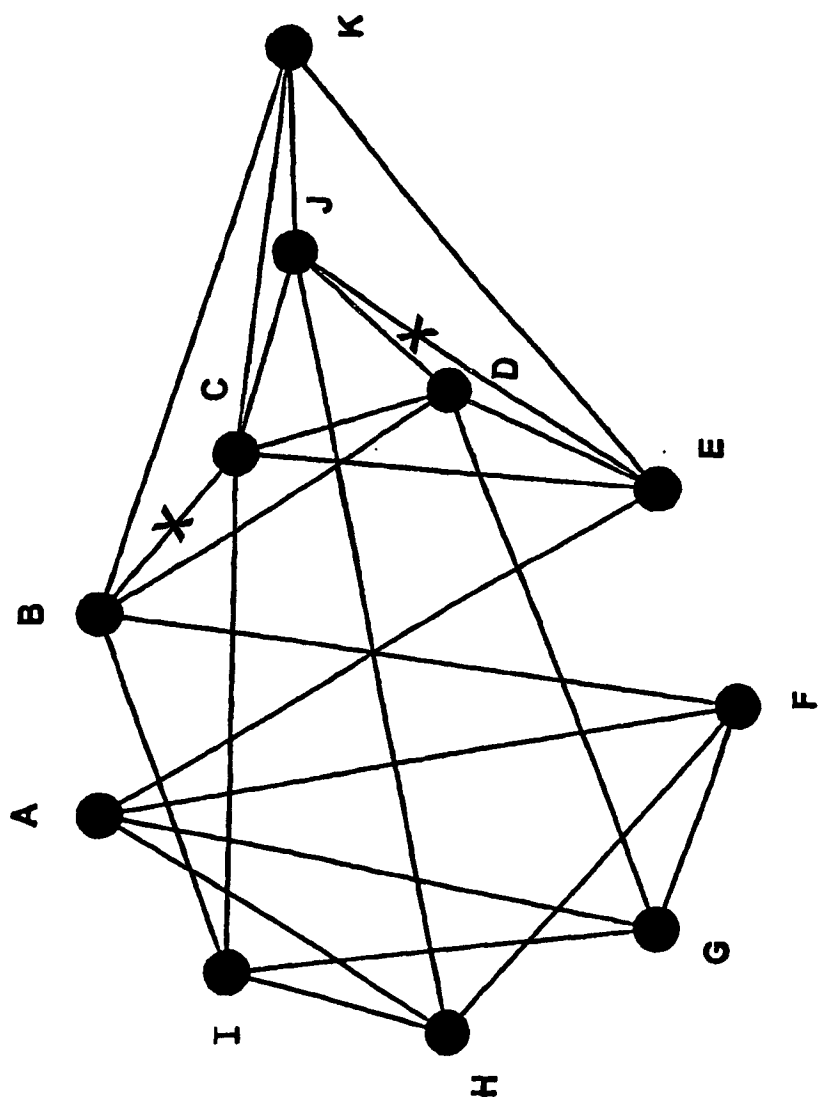


Fig. 4B

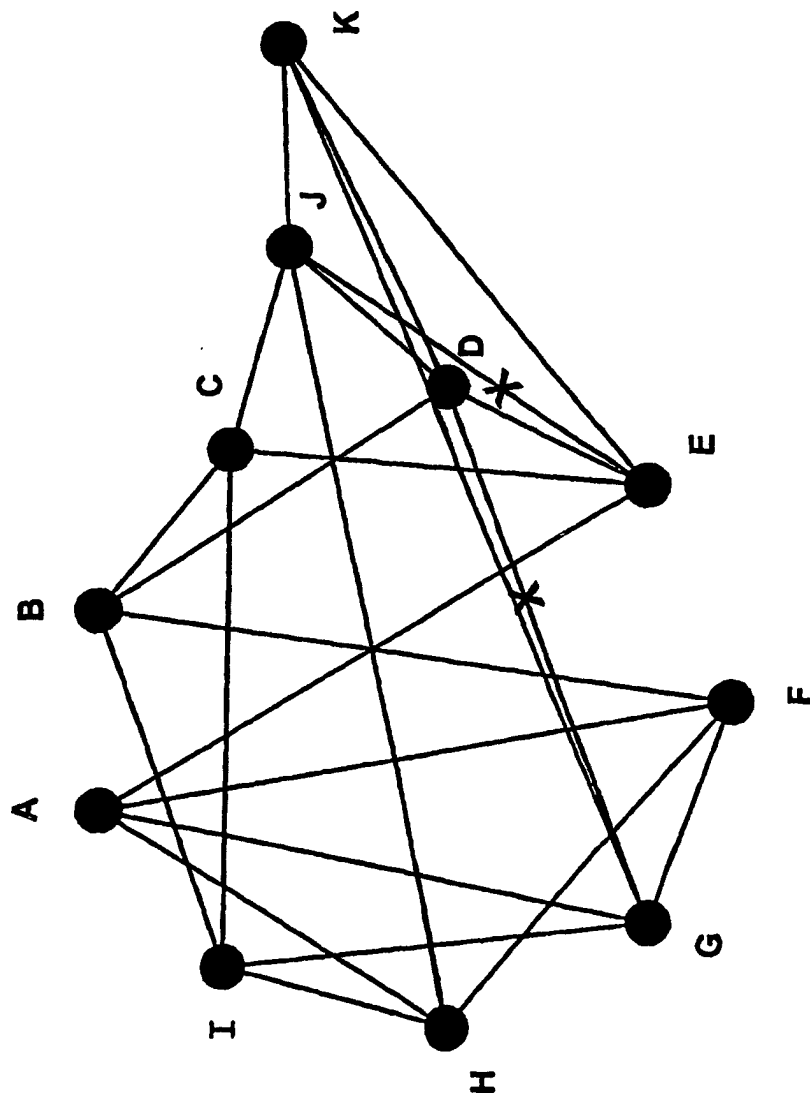


Fig. 4C

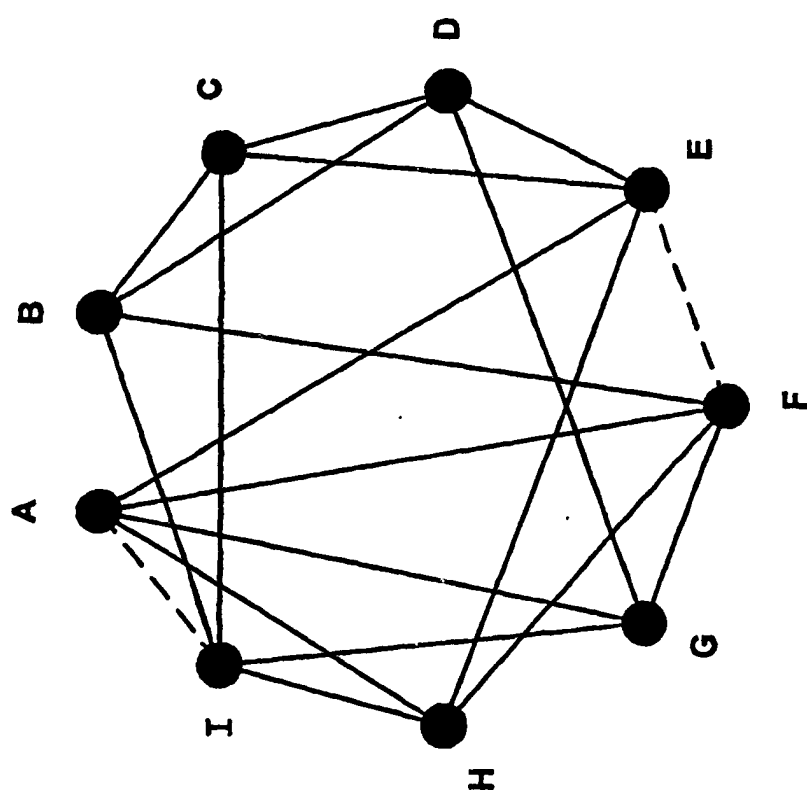


Fig. 5A

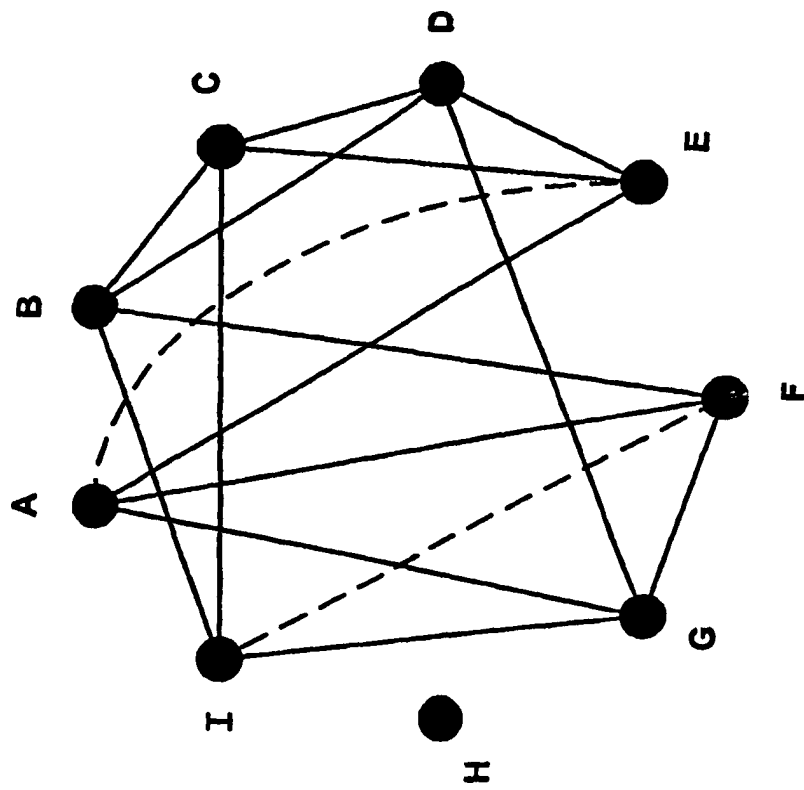


Fig. 5B

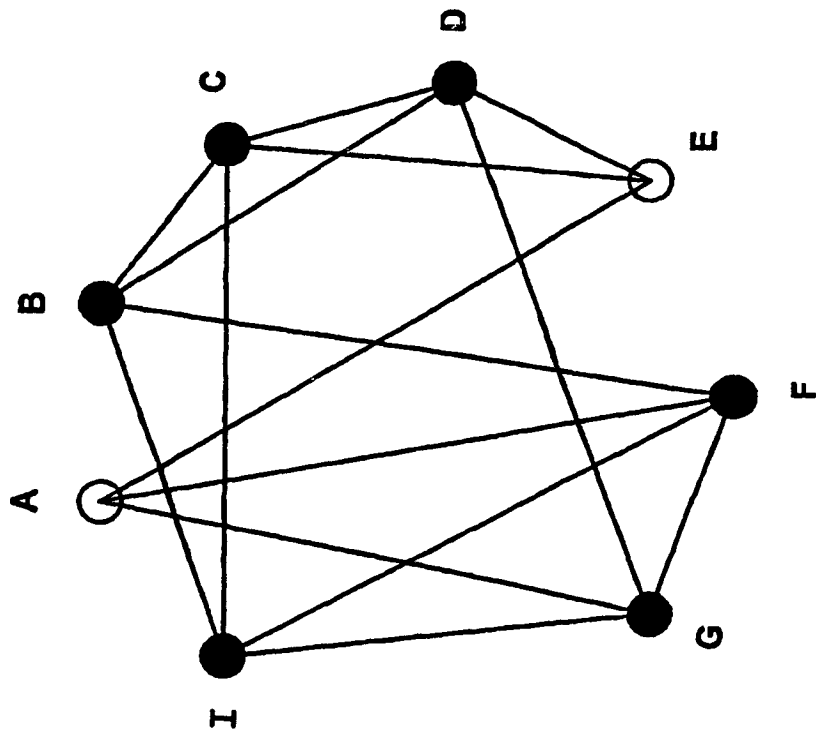


Fig. 5C

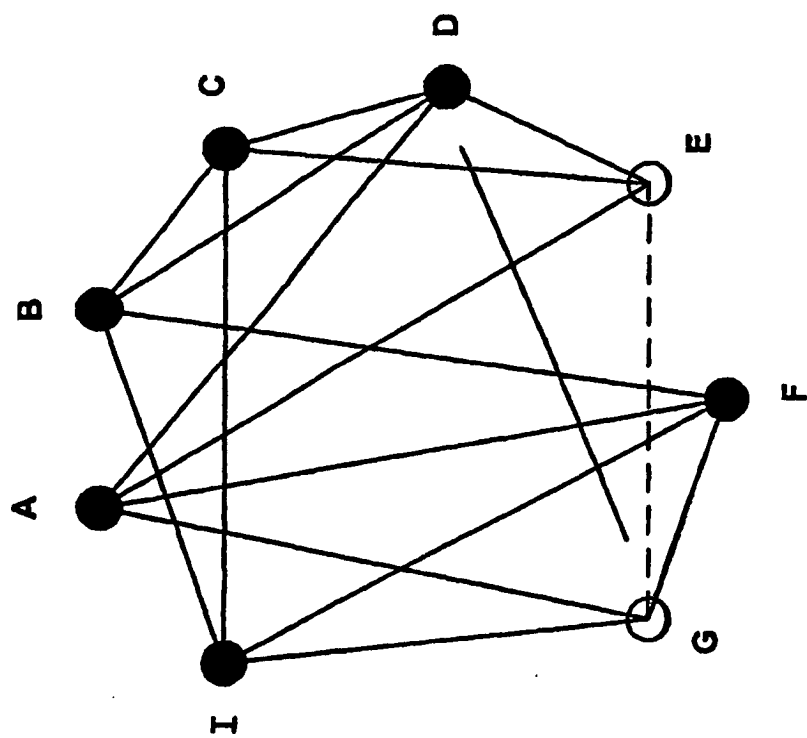


Fig. 5D

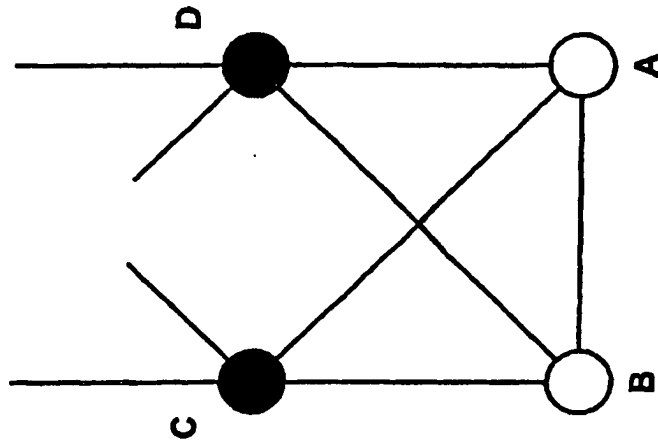


Fig. 5F

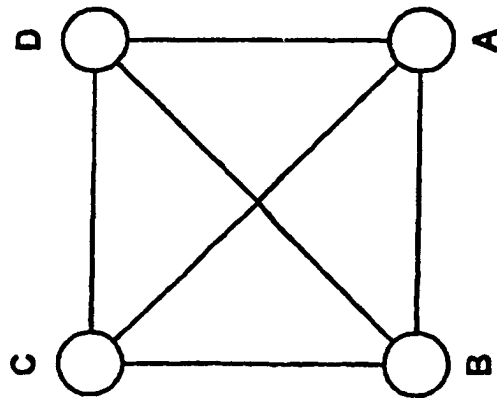


Fig. 5E

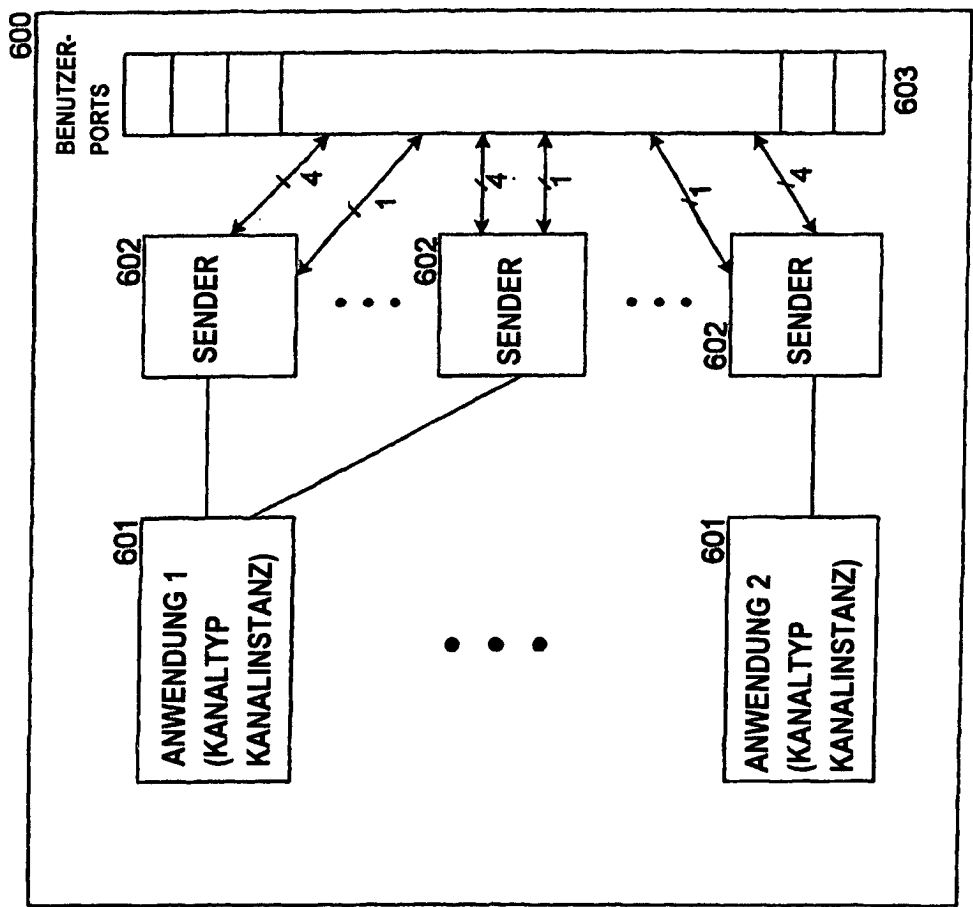


Fig. 6

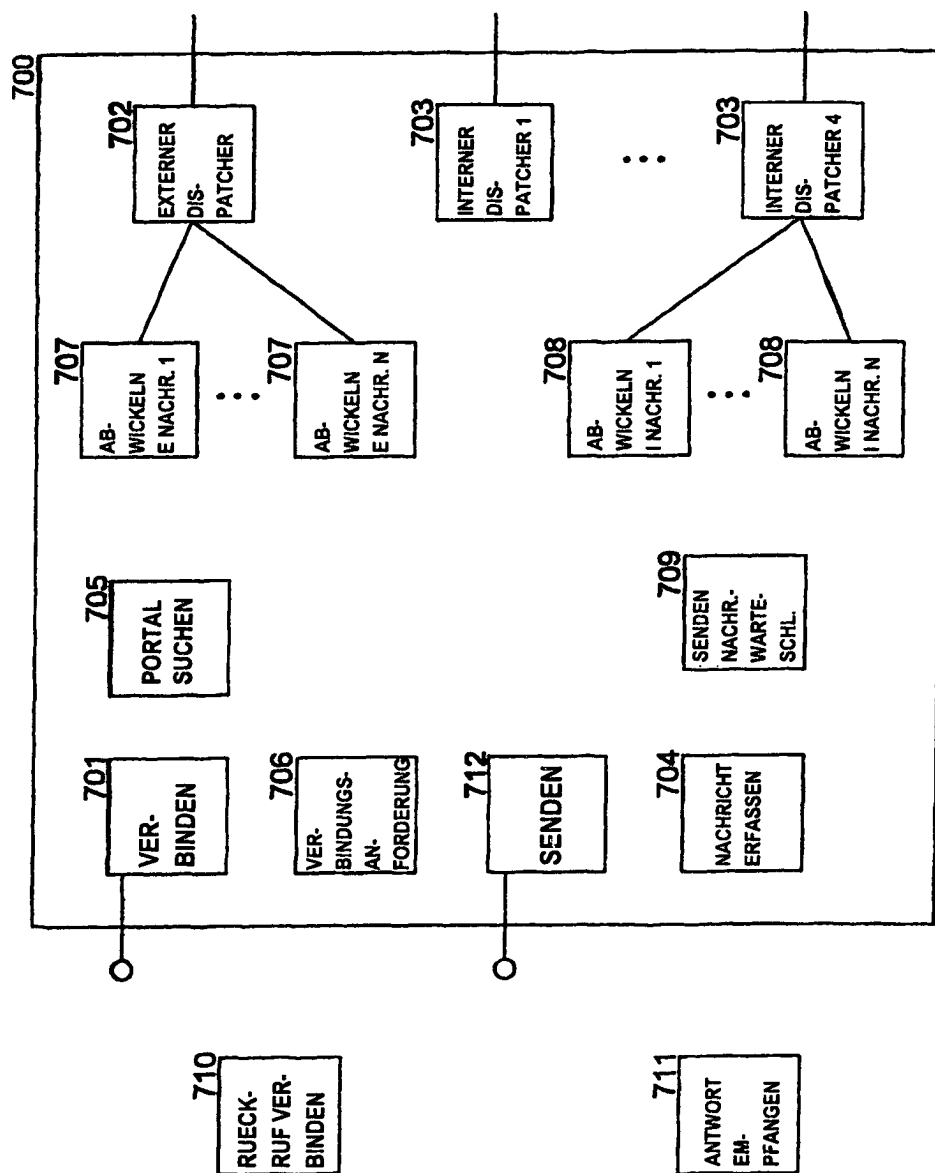
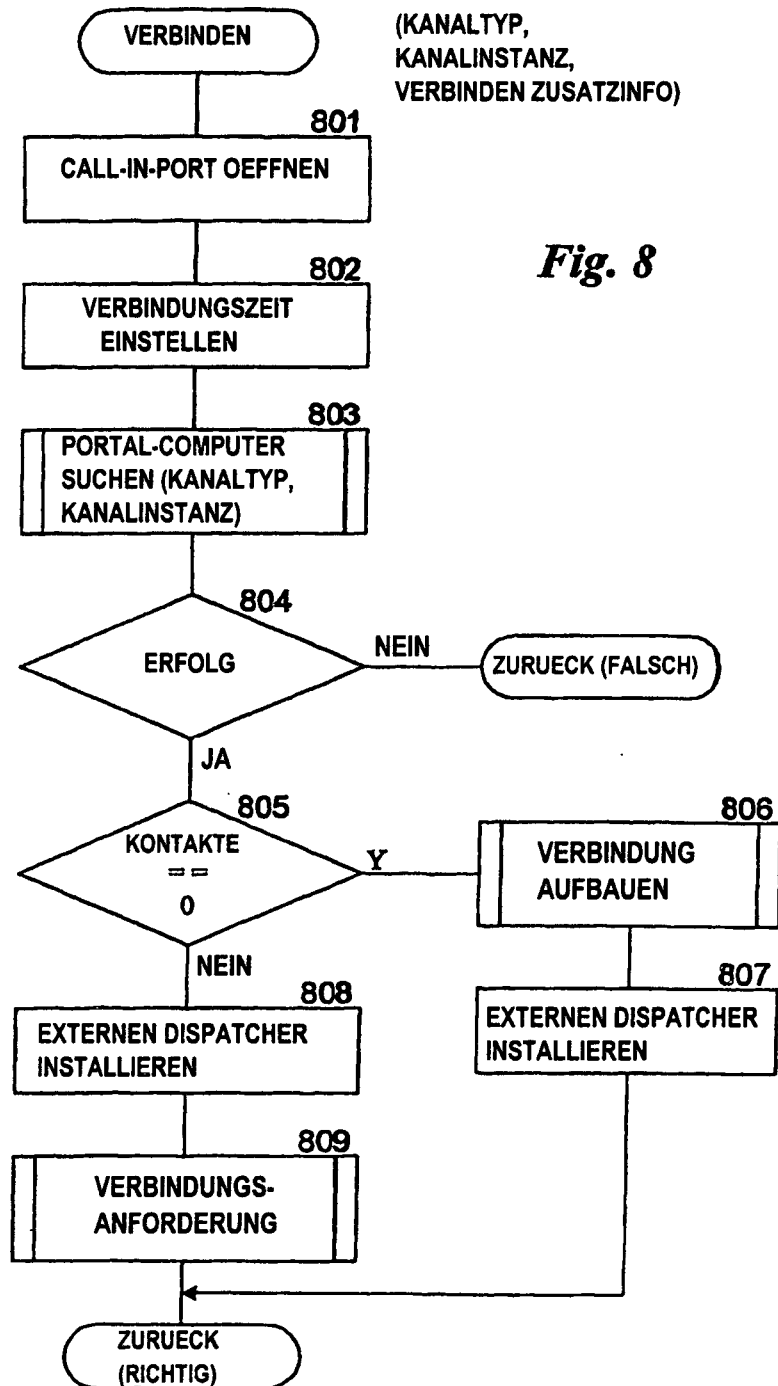
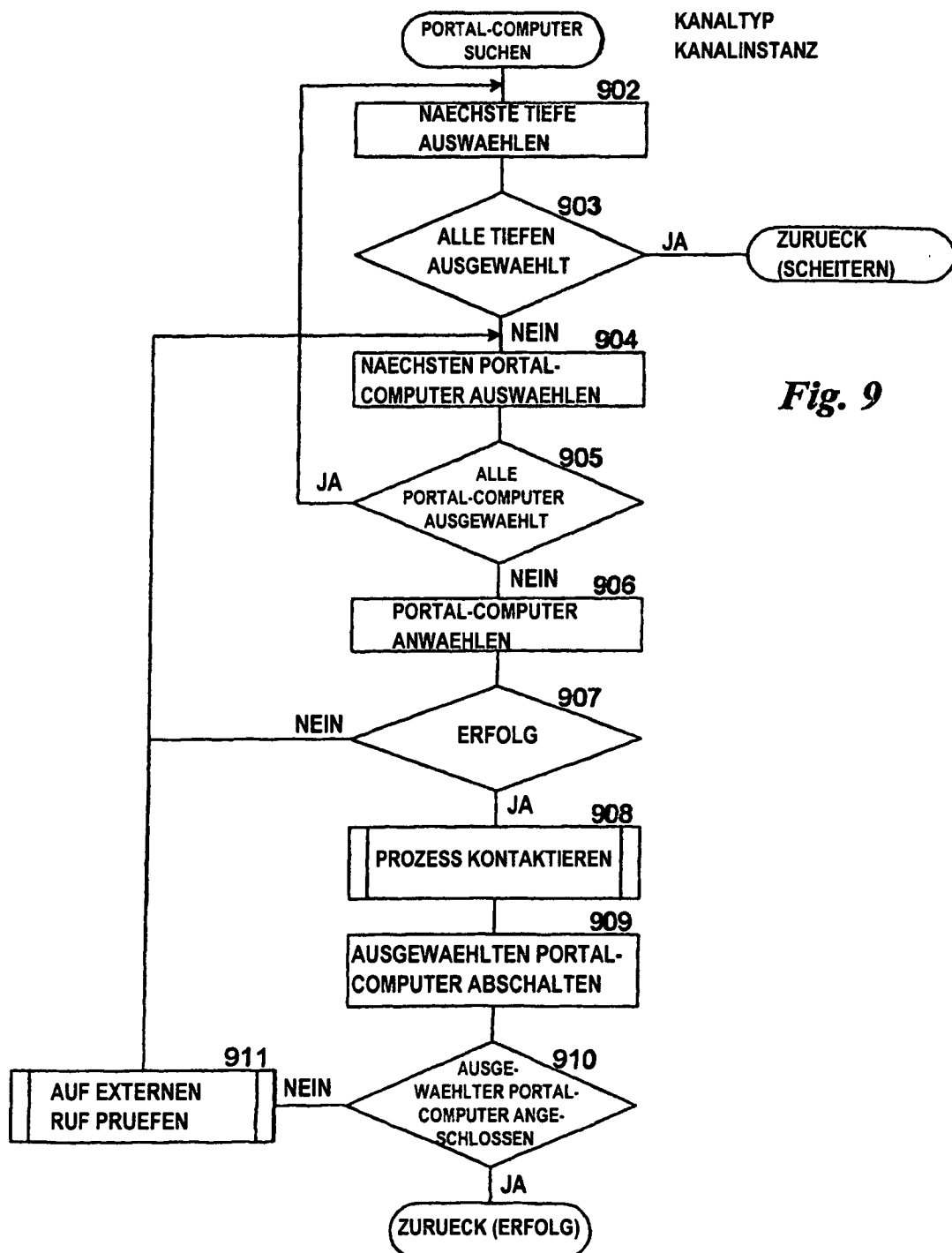


Fig. 7





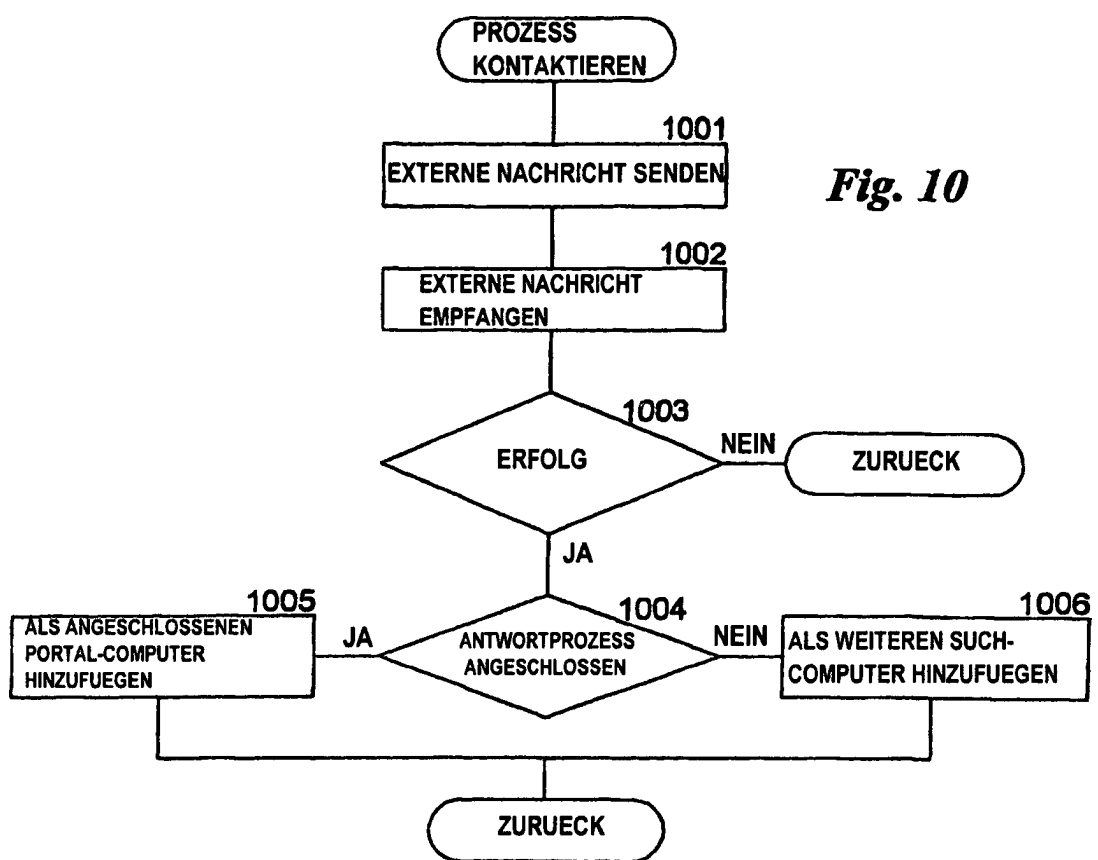
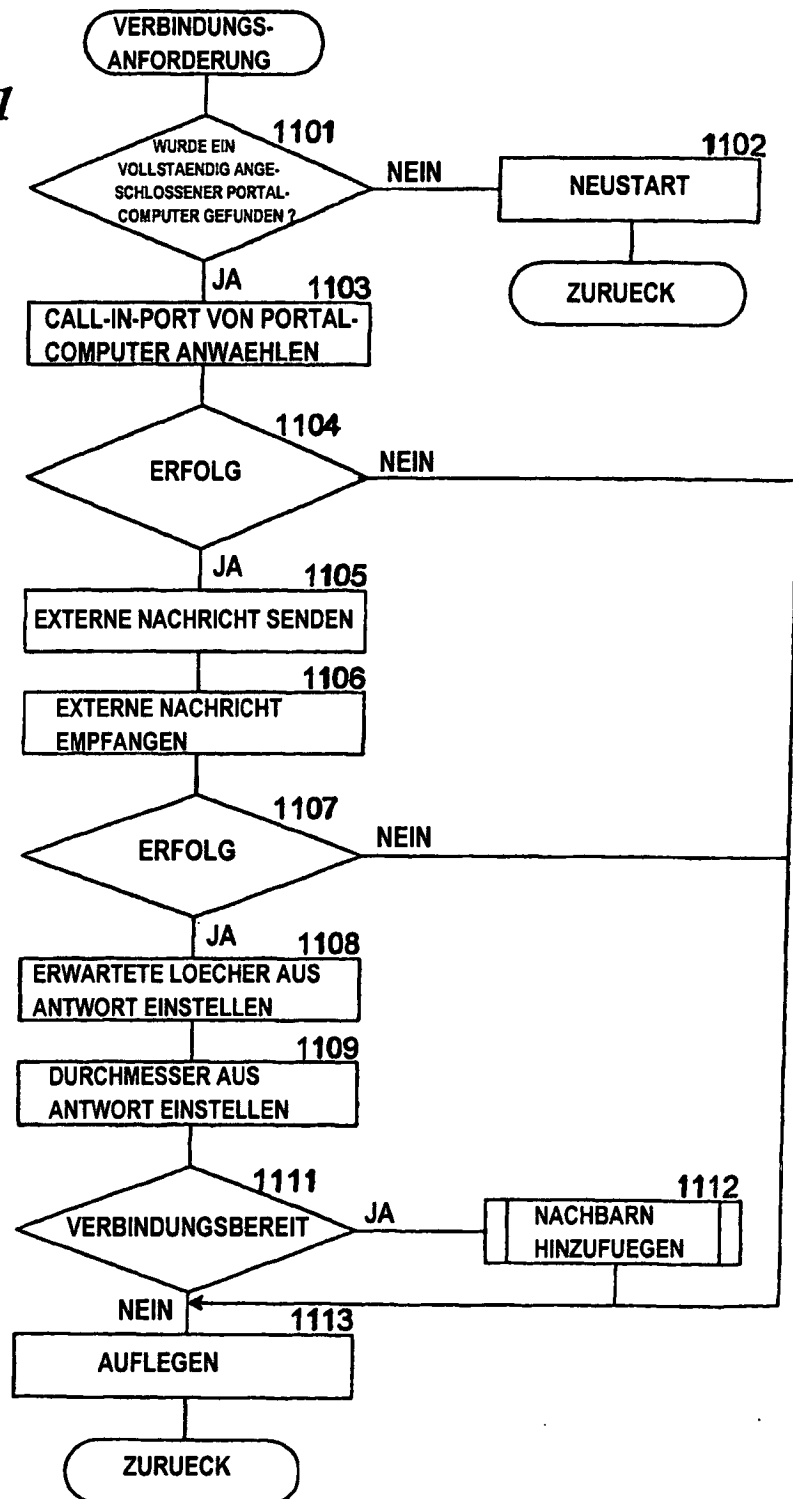


Fig. 11

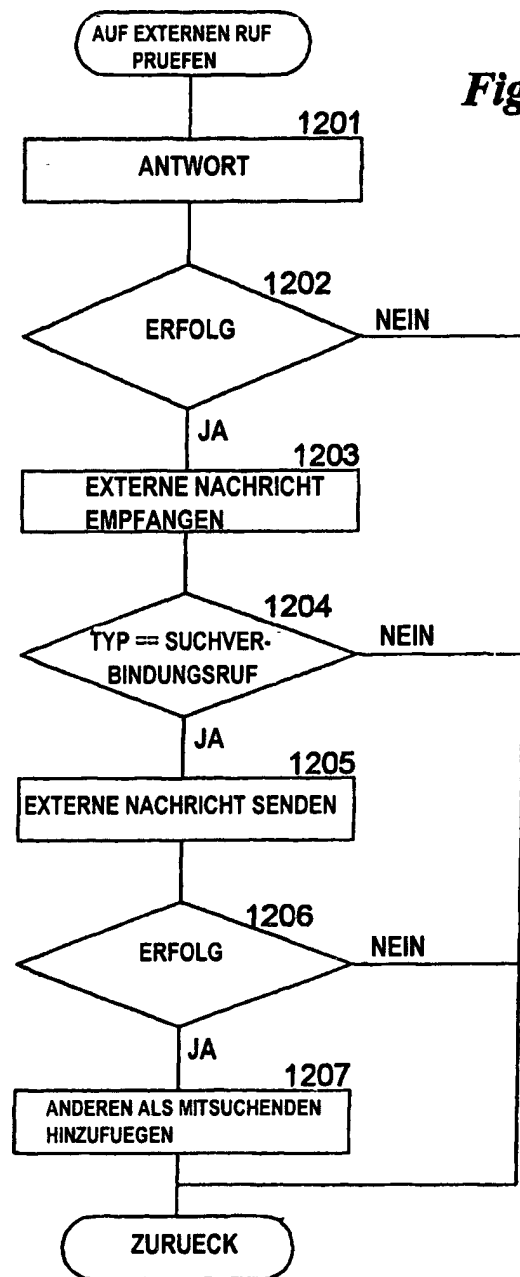


Fig. 12

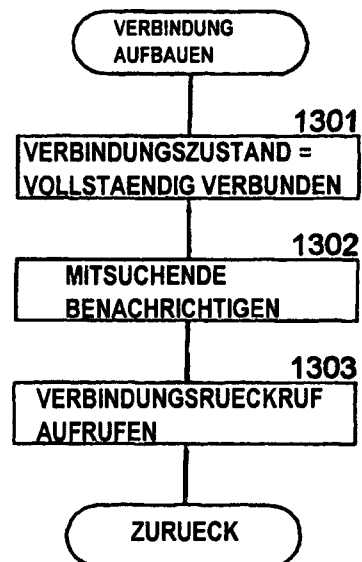


Fig. 13

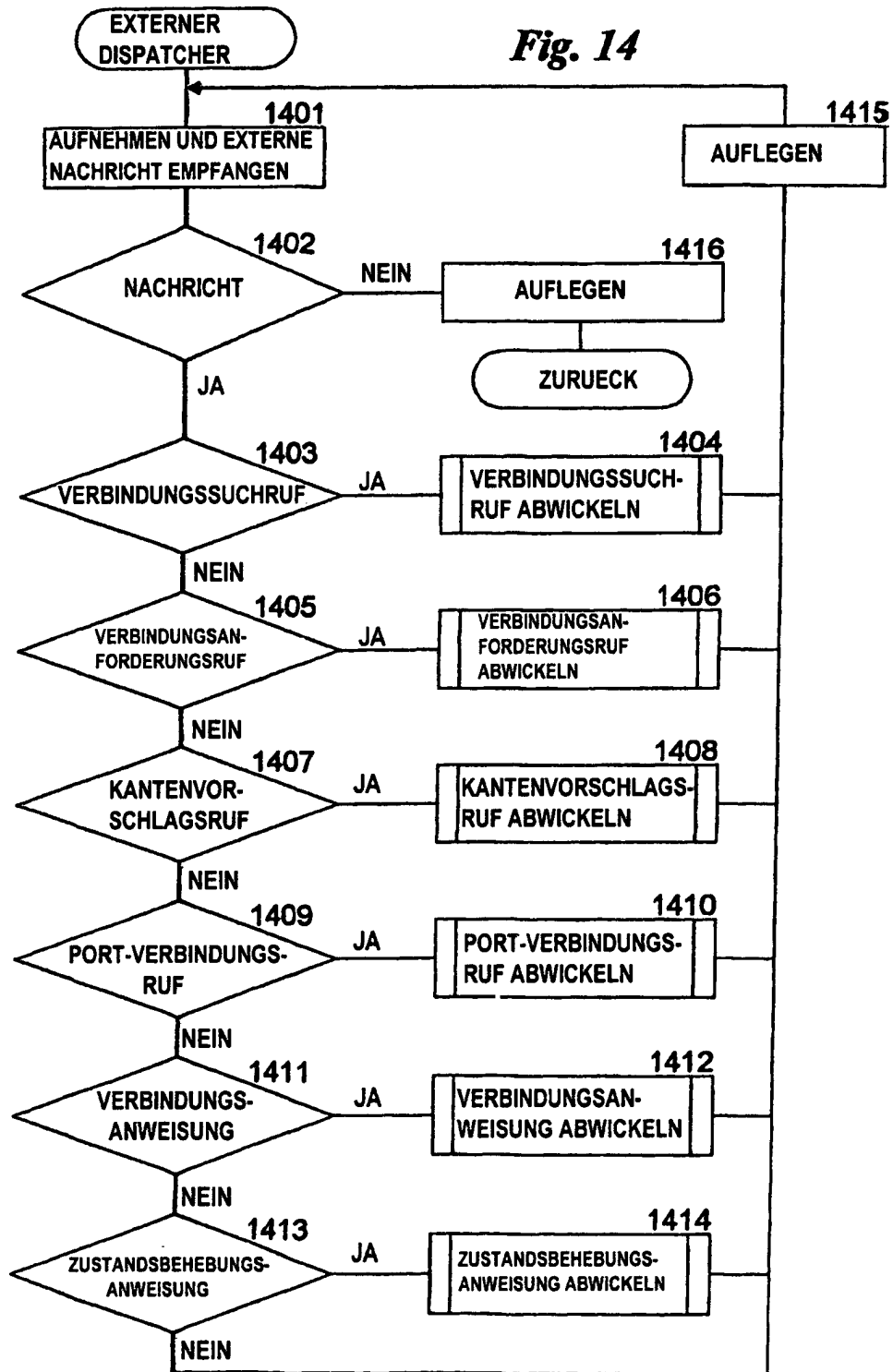
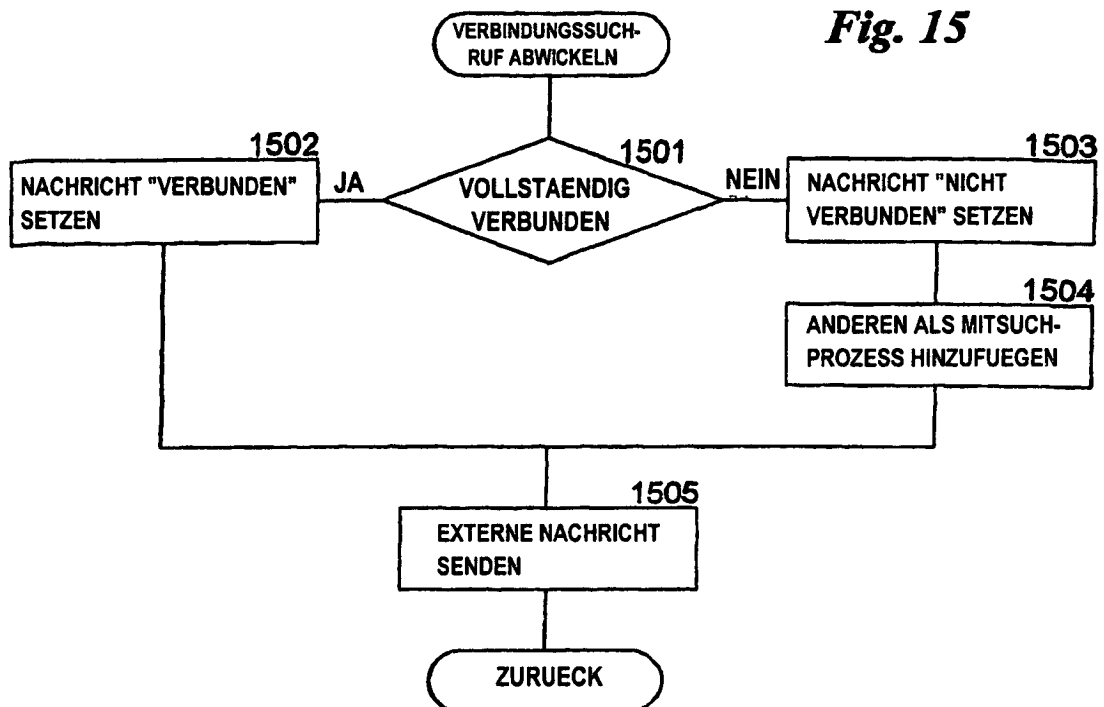
Fig. 14

Fig. 15



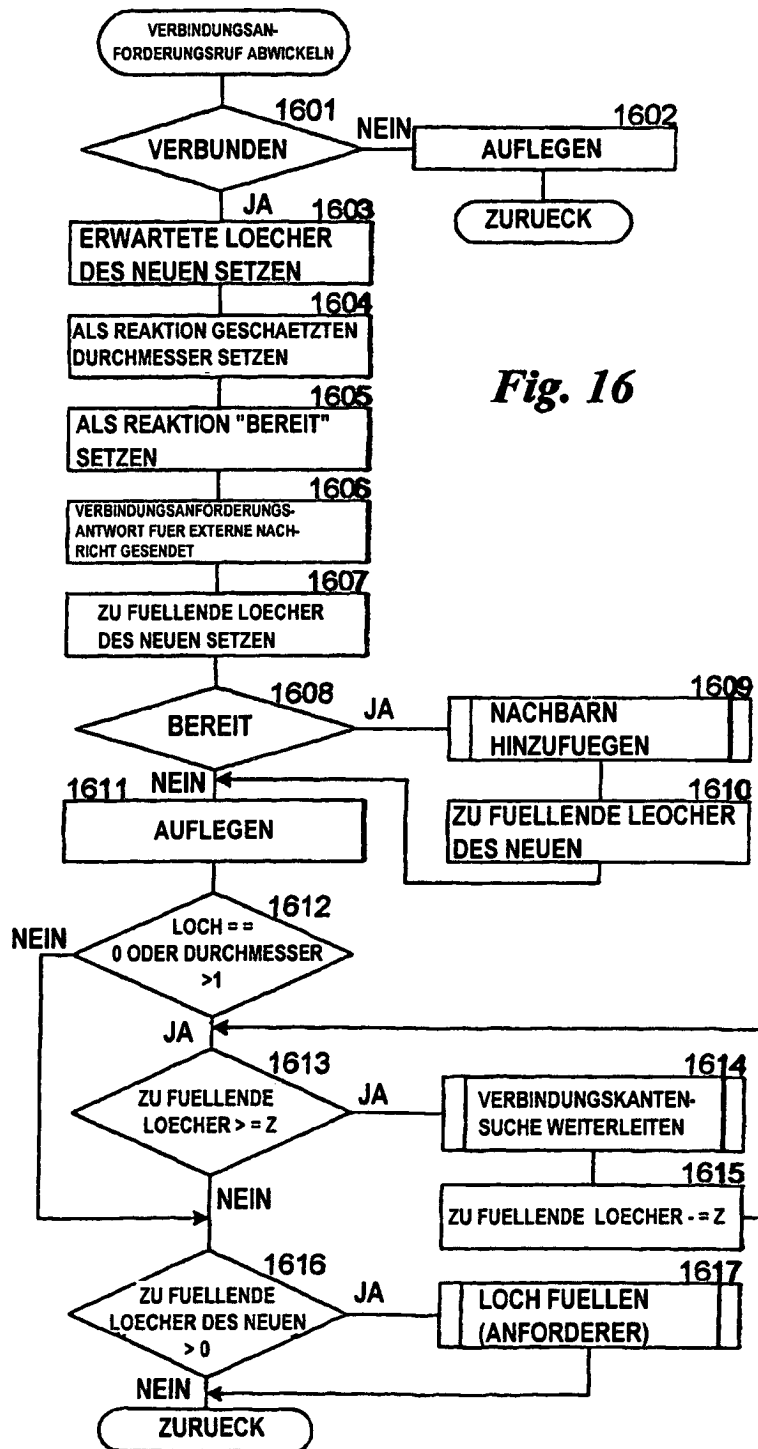


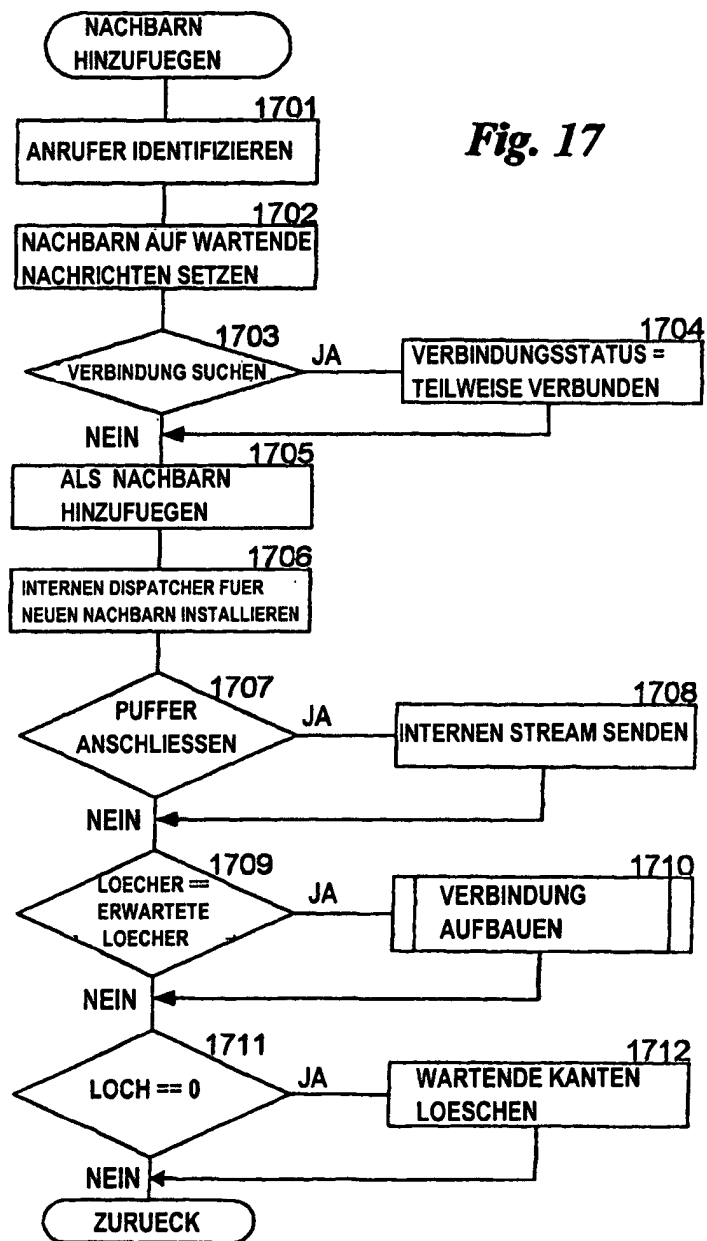
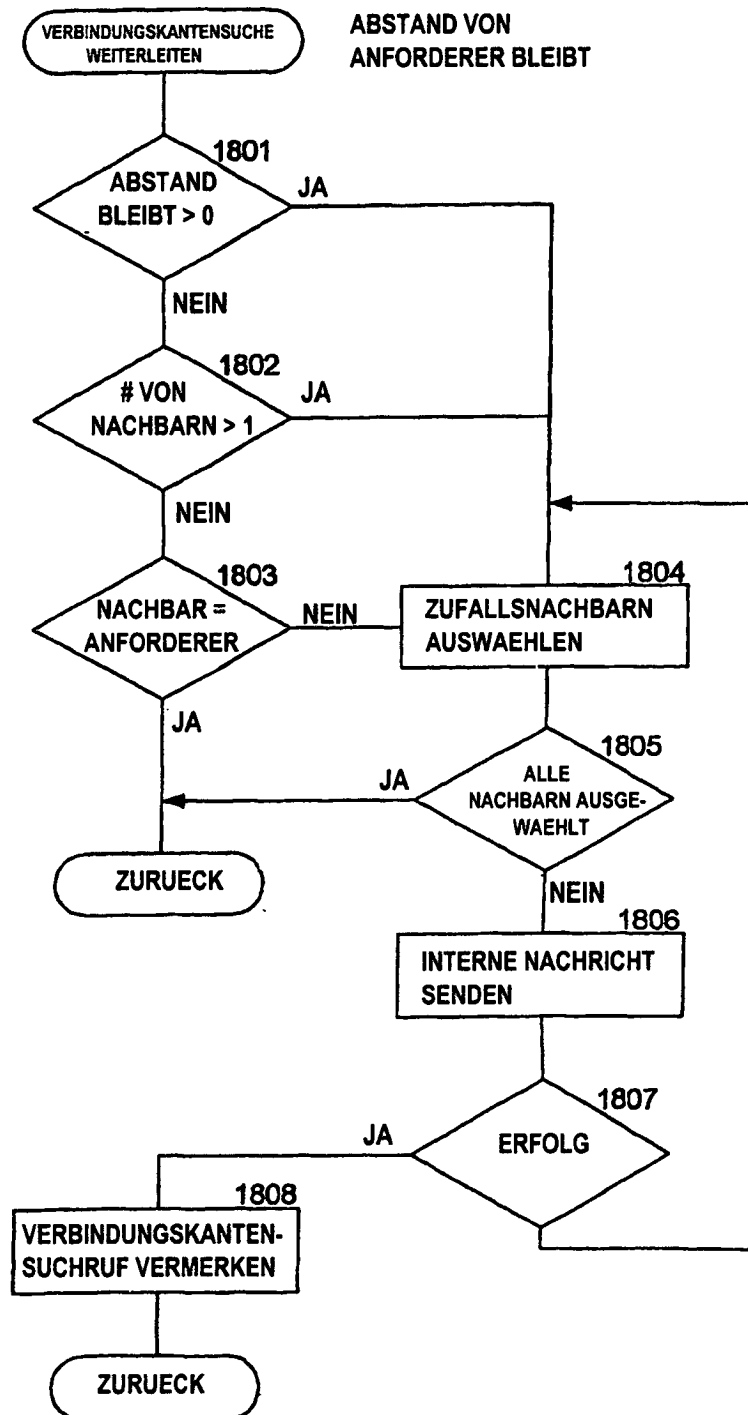
Fig. 17

Fig. 18

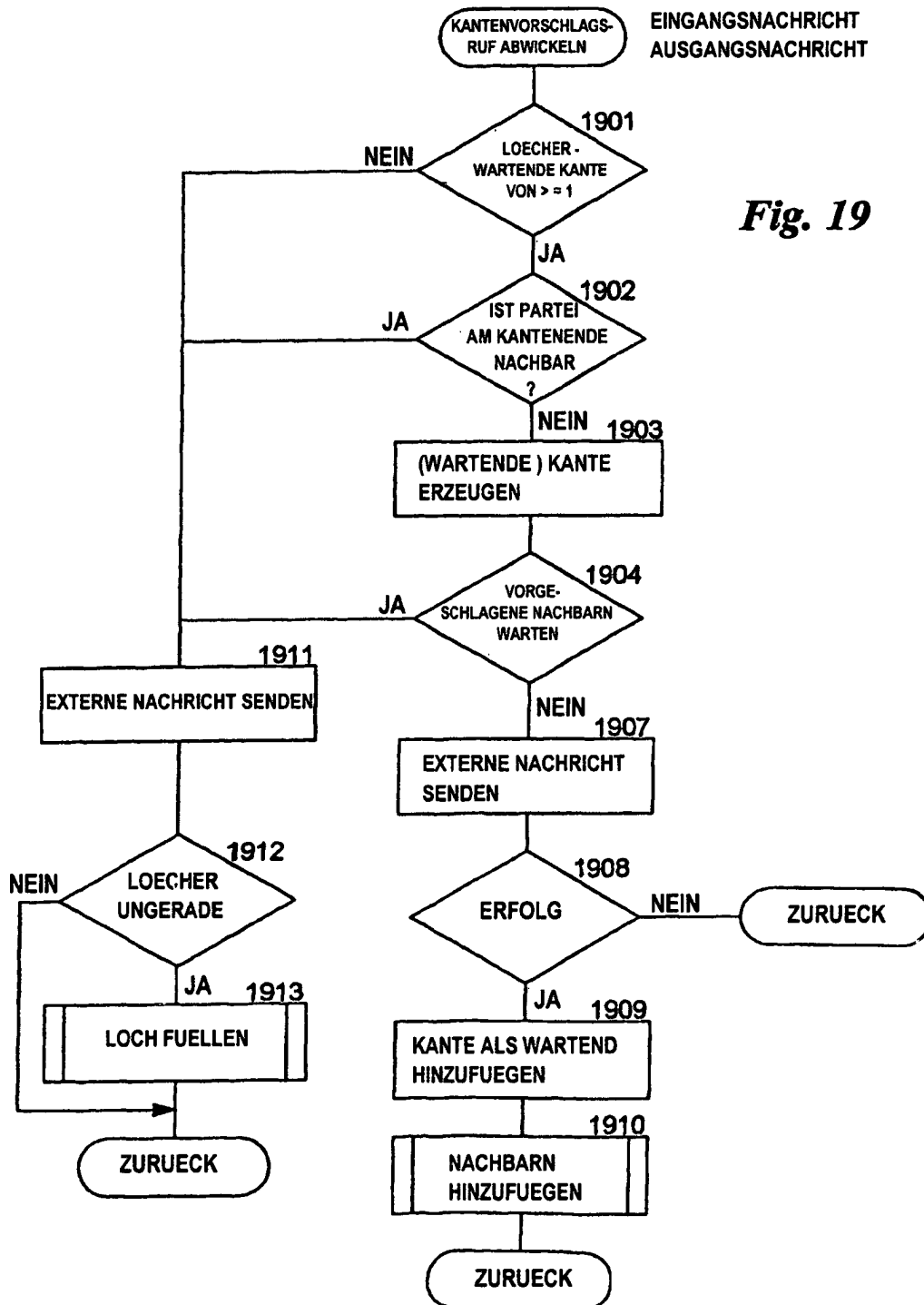


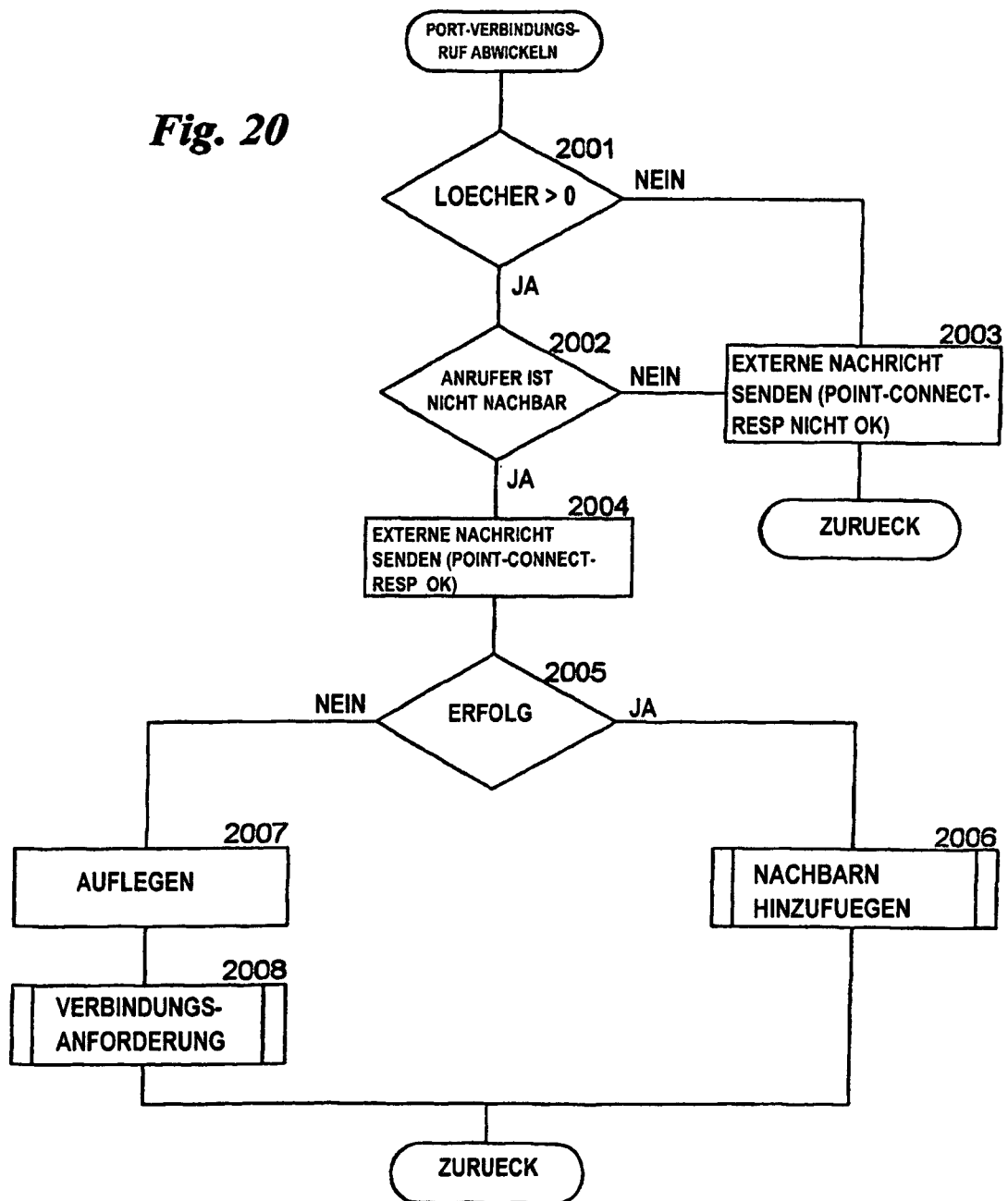
Fig. 20

Fig. 21

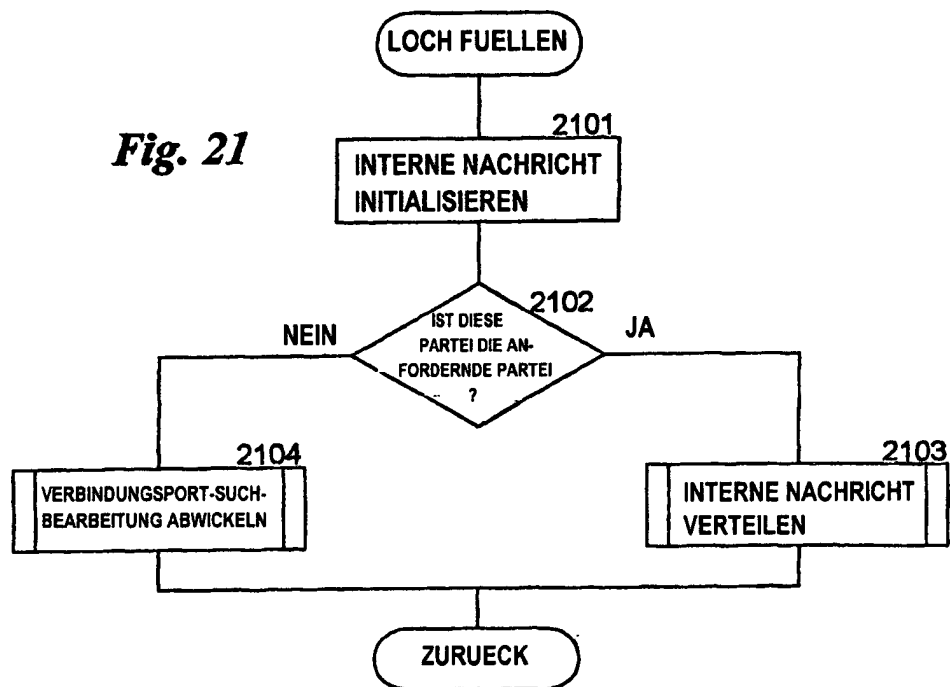


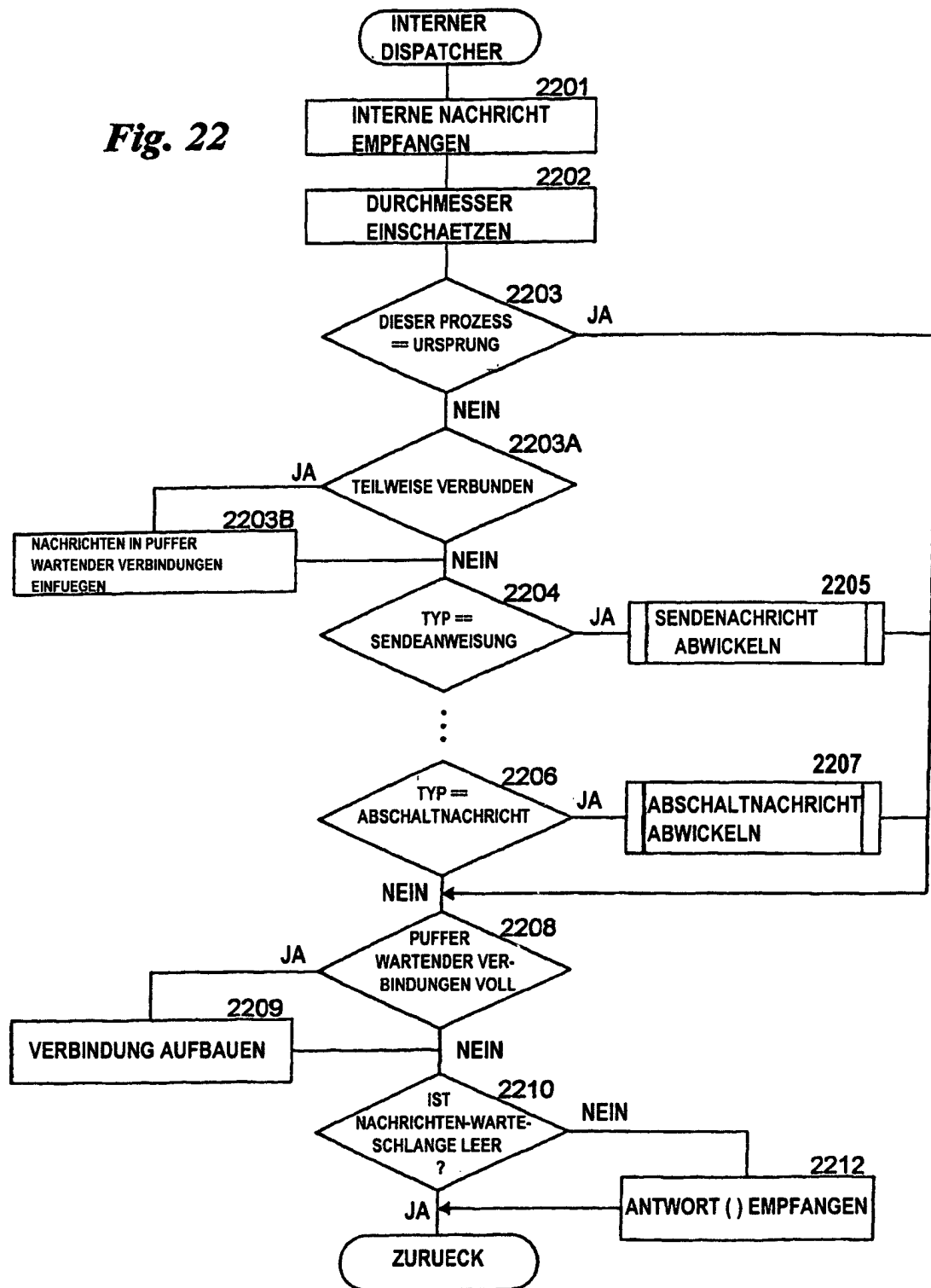
Fig. 22

Fig. 23

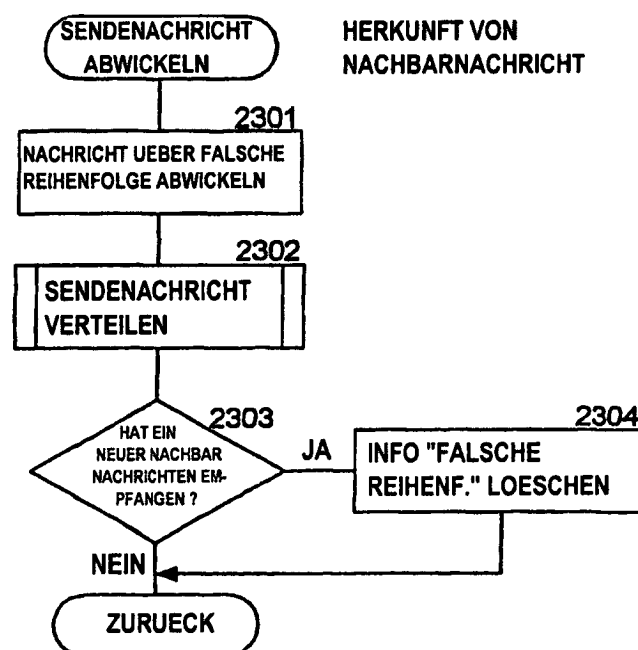
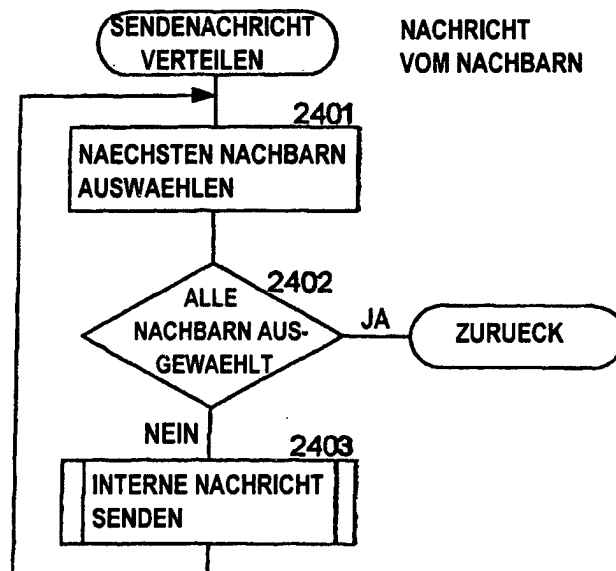


Fig. 24



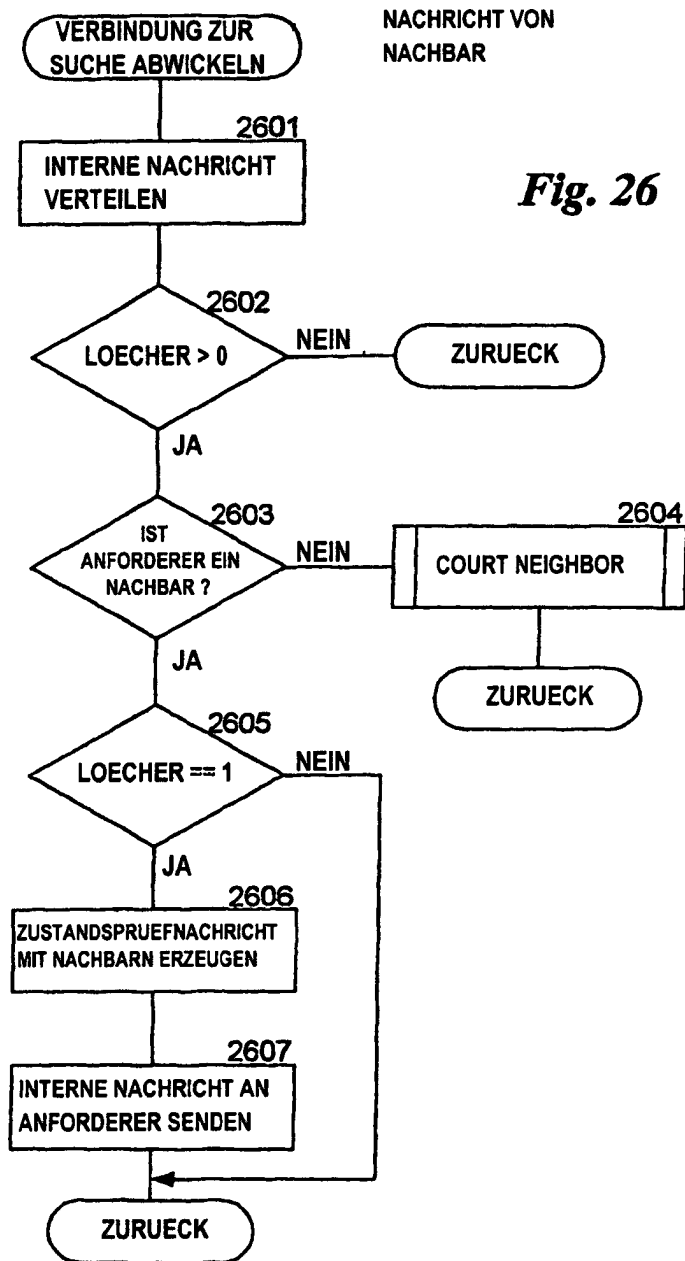


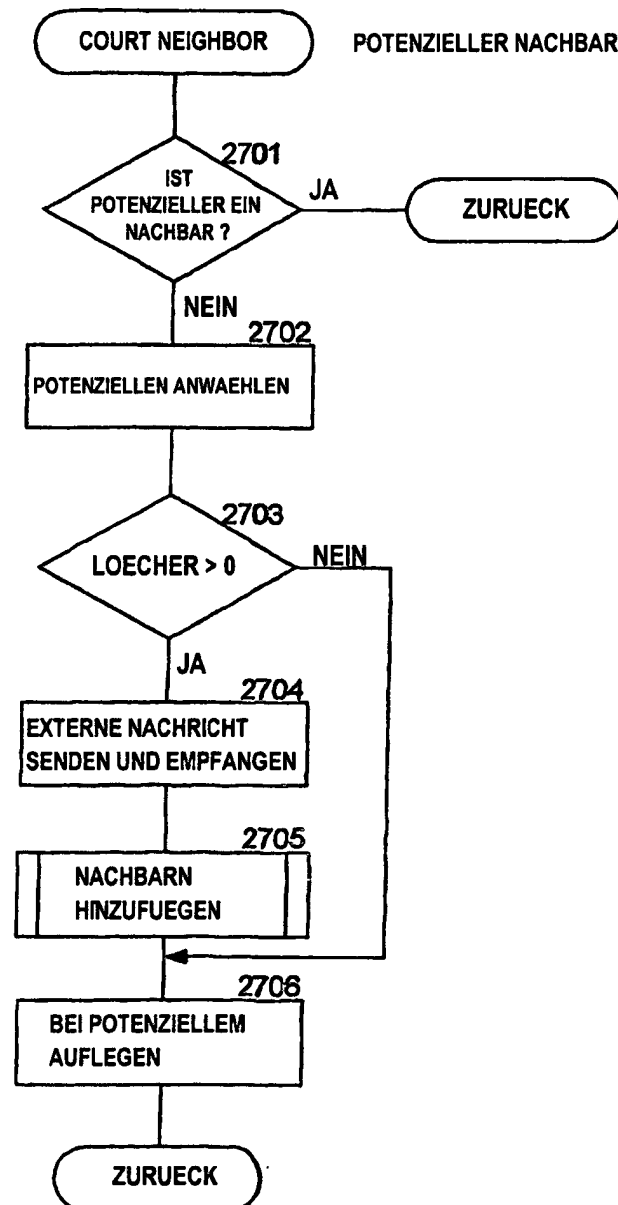
Fig. 27

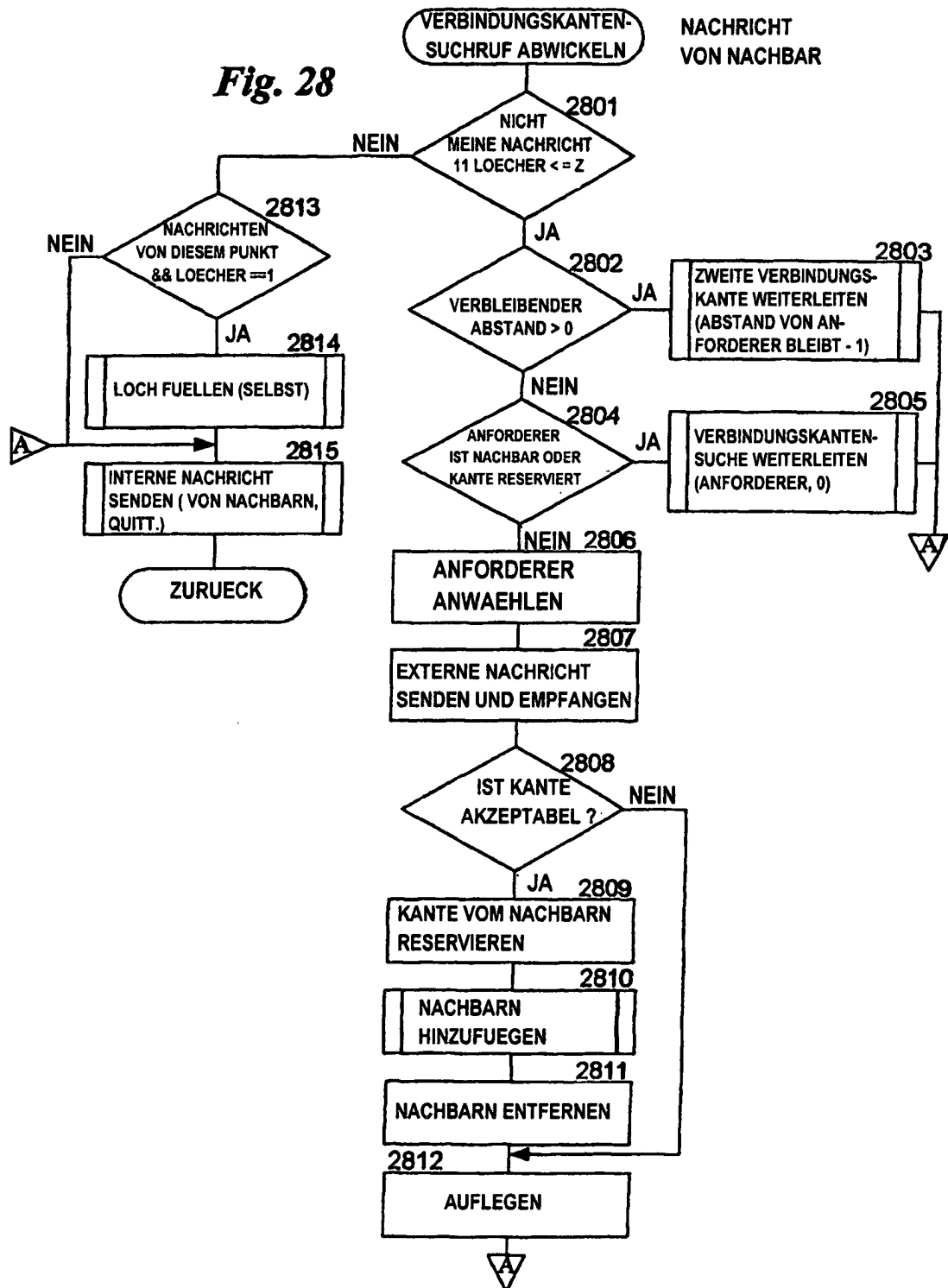
Fig. 28

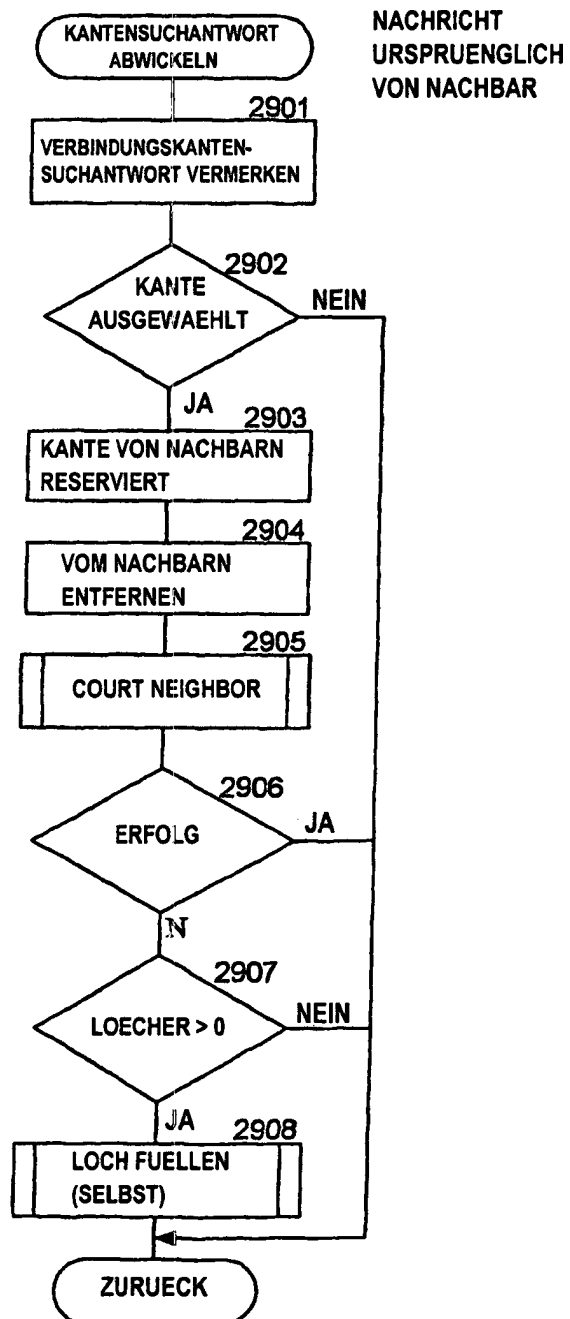
Fig. 29

Fig. 30

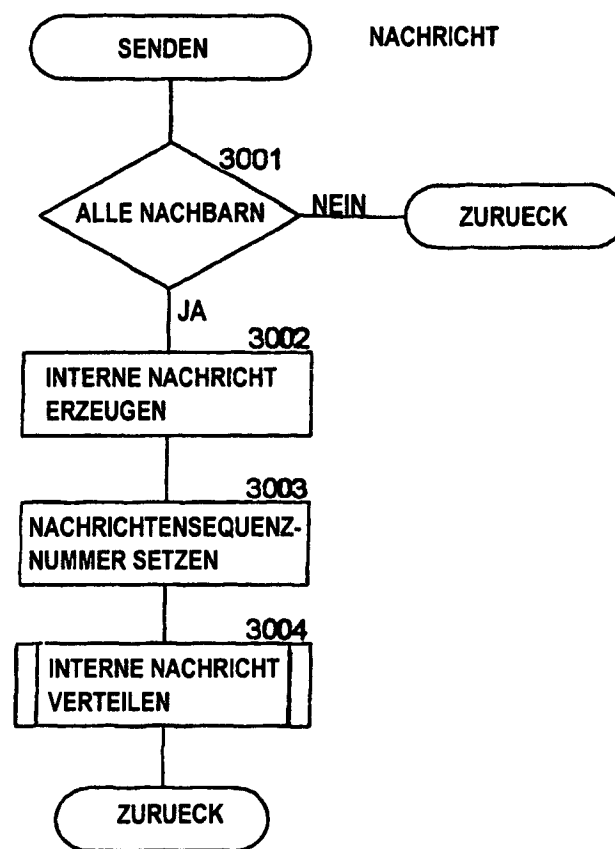


Fig. 31

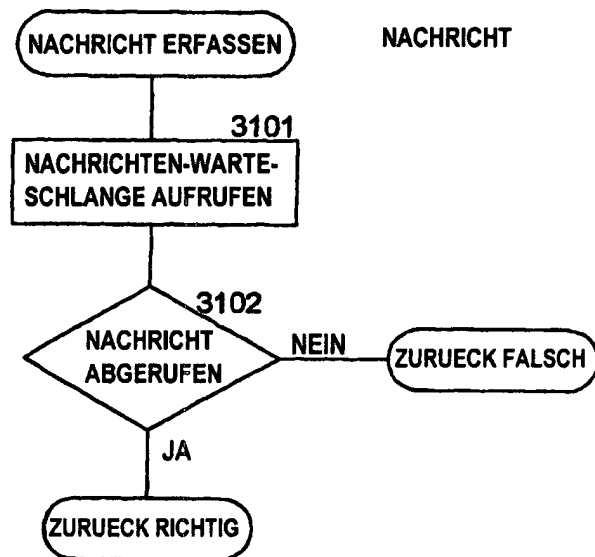


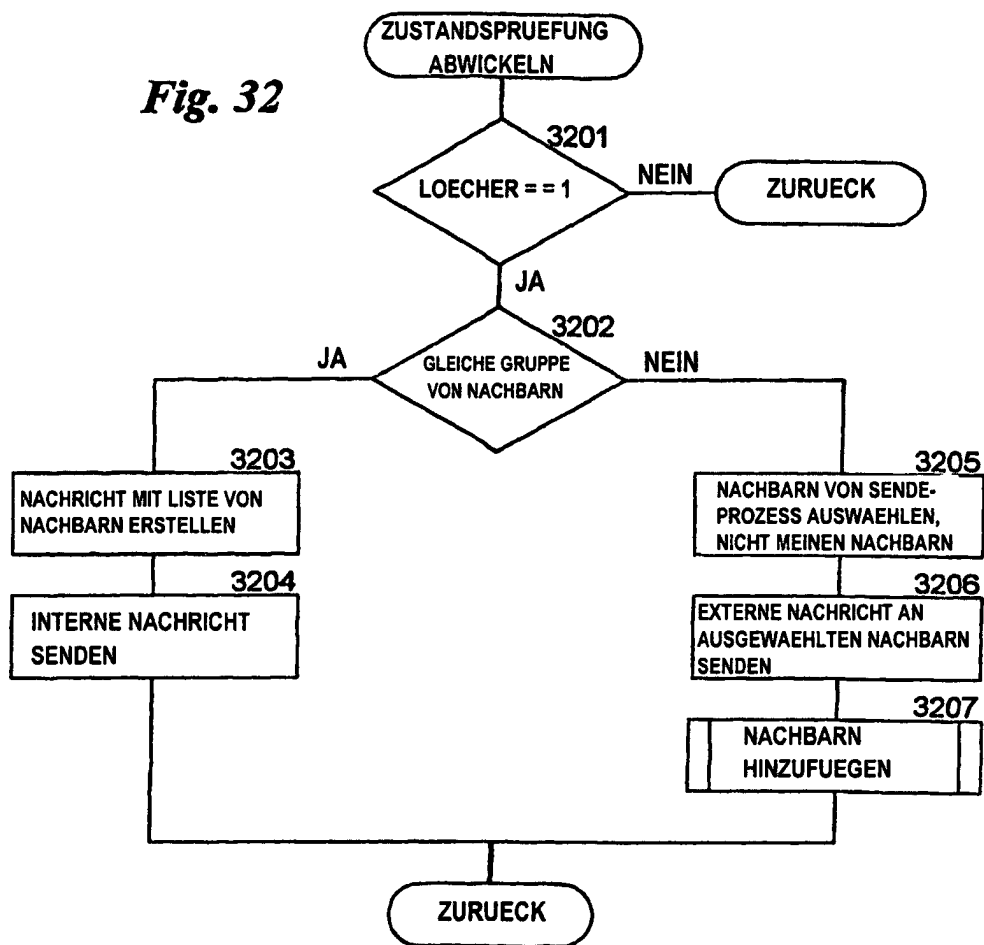
Fig. 32

Fig. 33

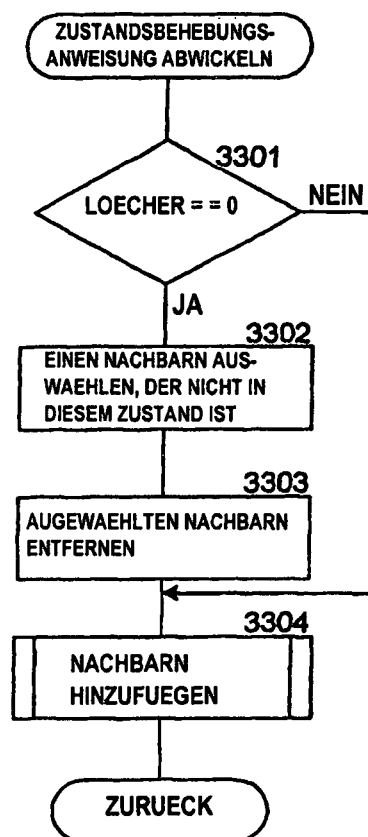
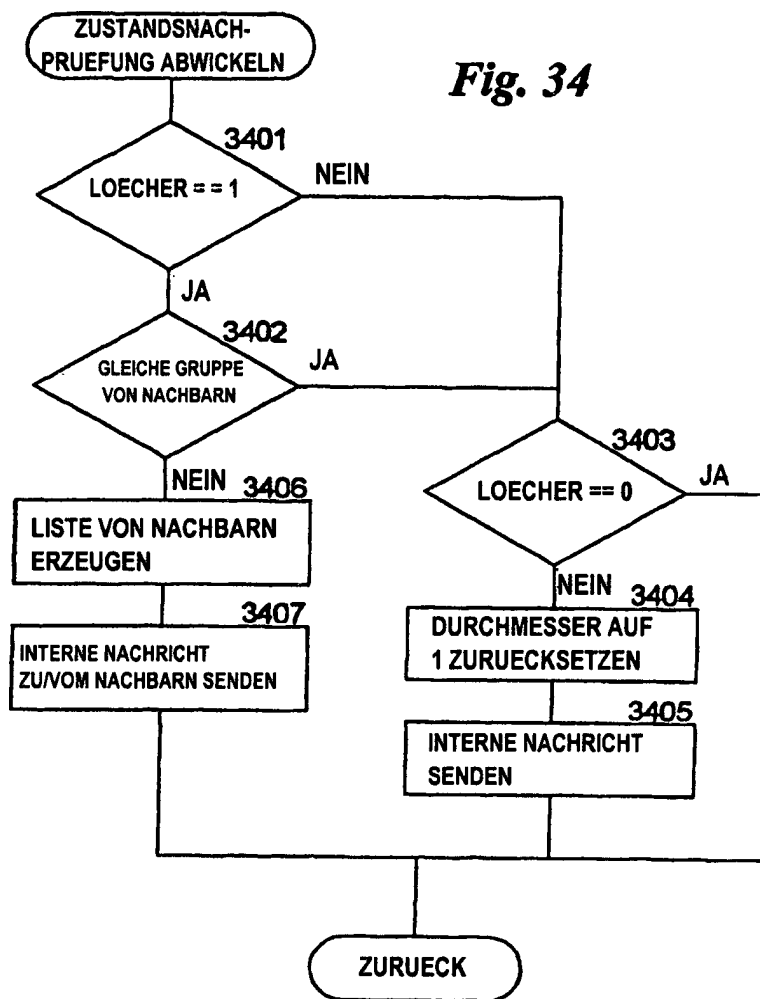


Fig. 34

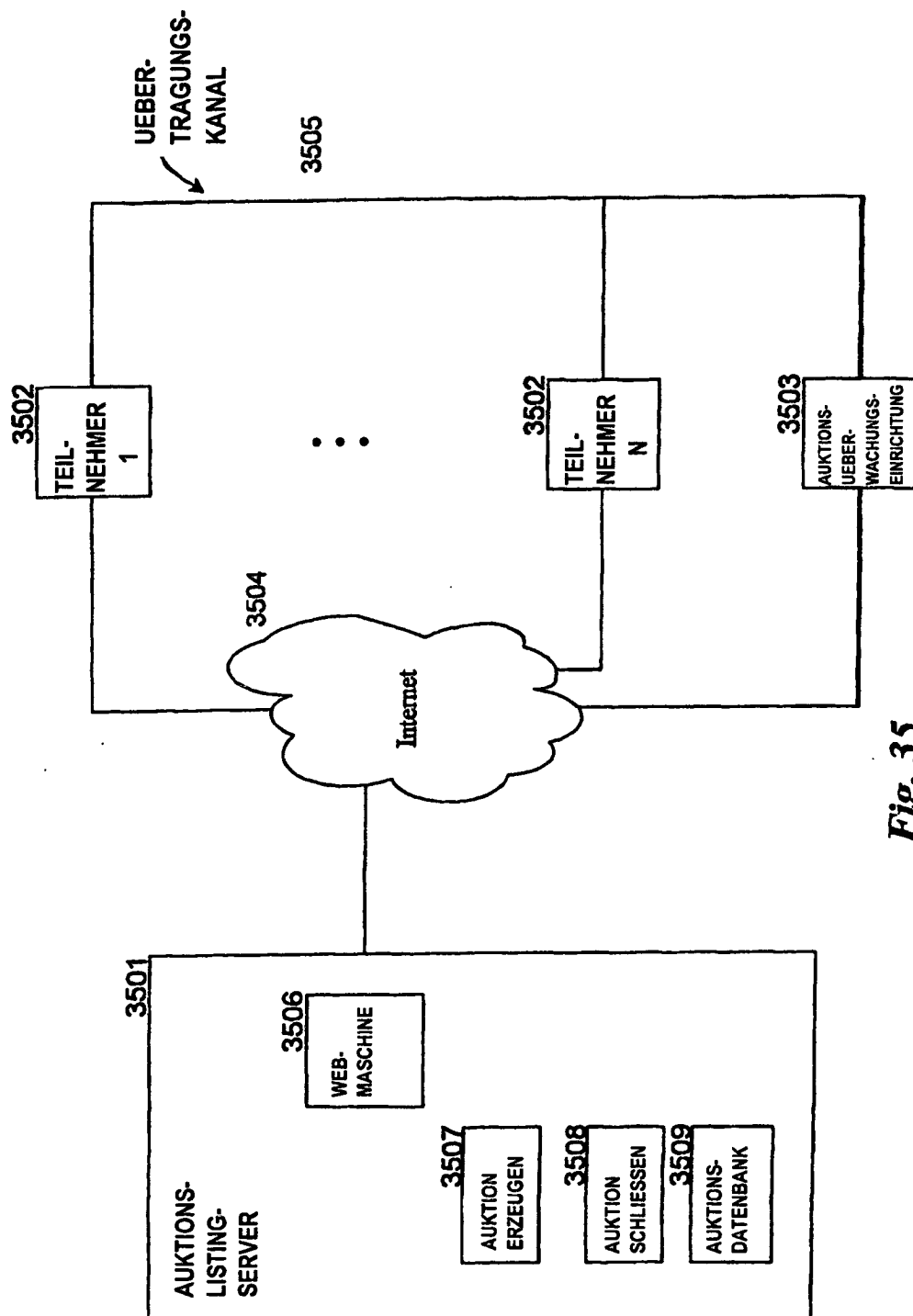


Fig. 35

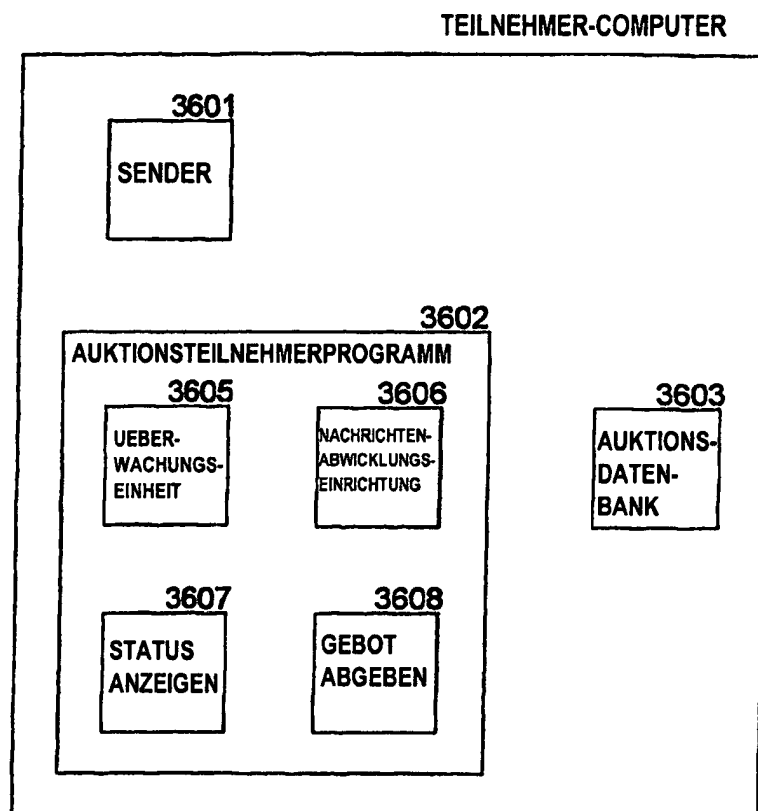


Fig. 36

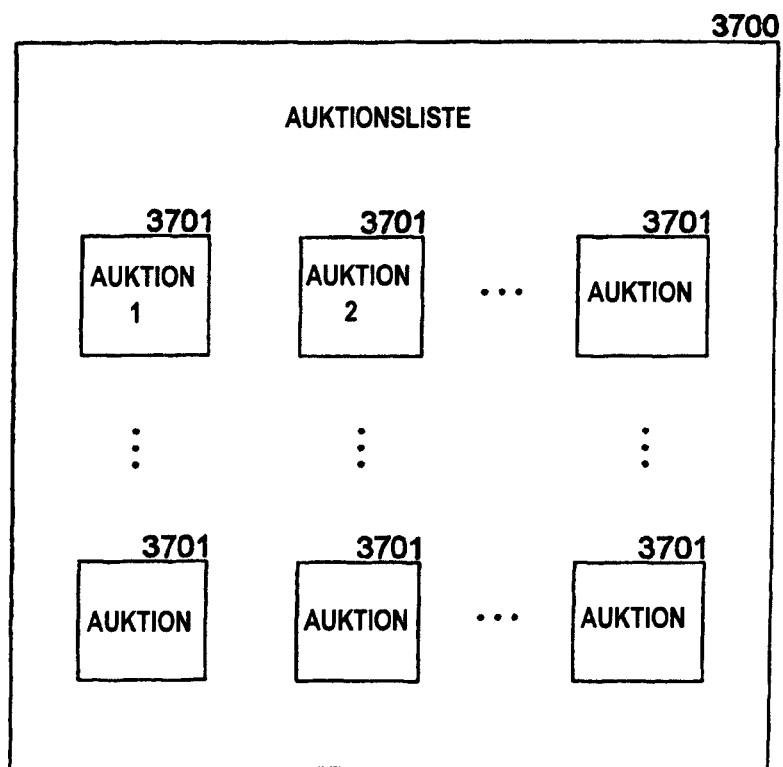


Fig. 37

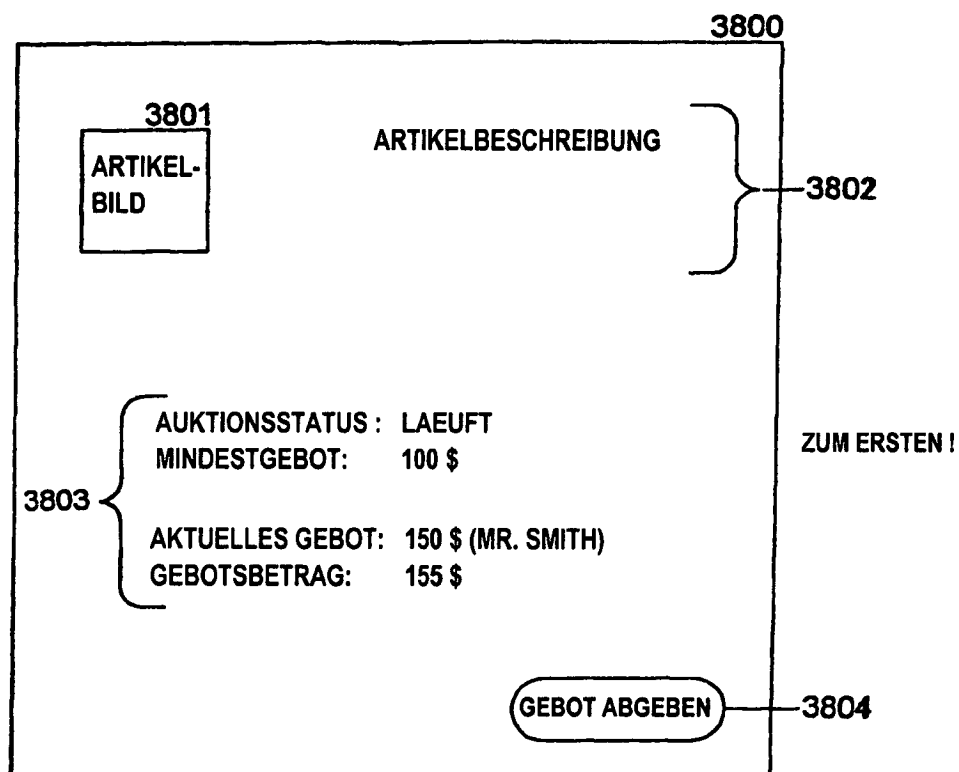


Fig. 38

Fig. 39

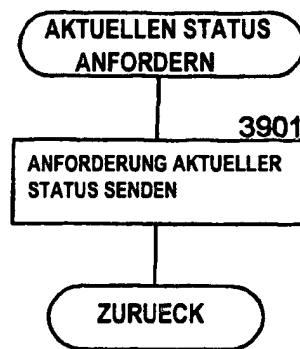


Fig. 40

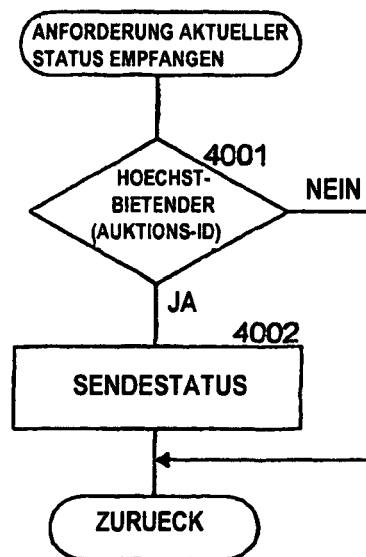


Fig. 41

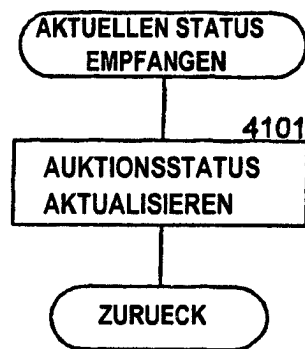
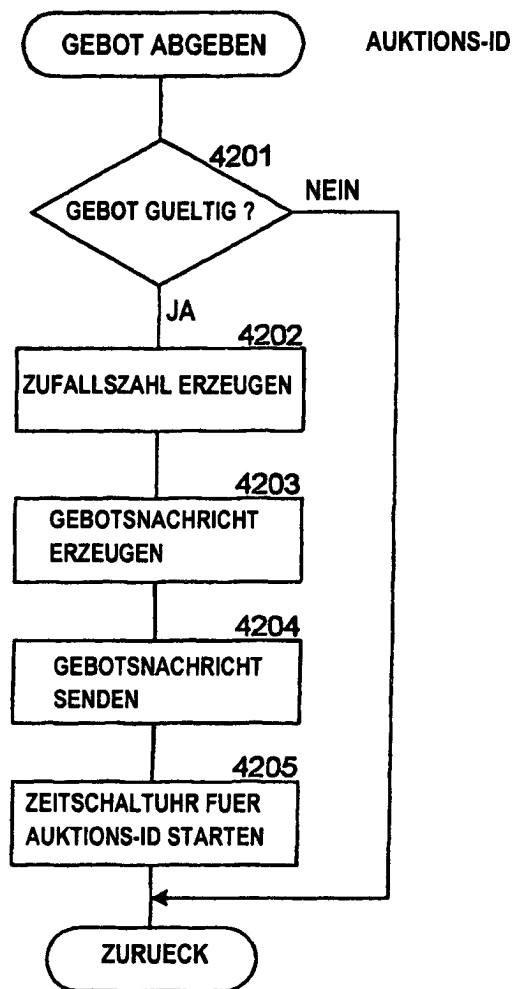


Fig. 42



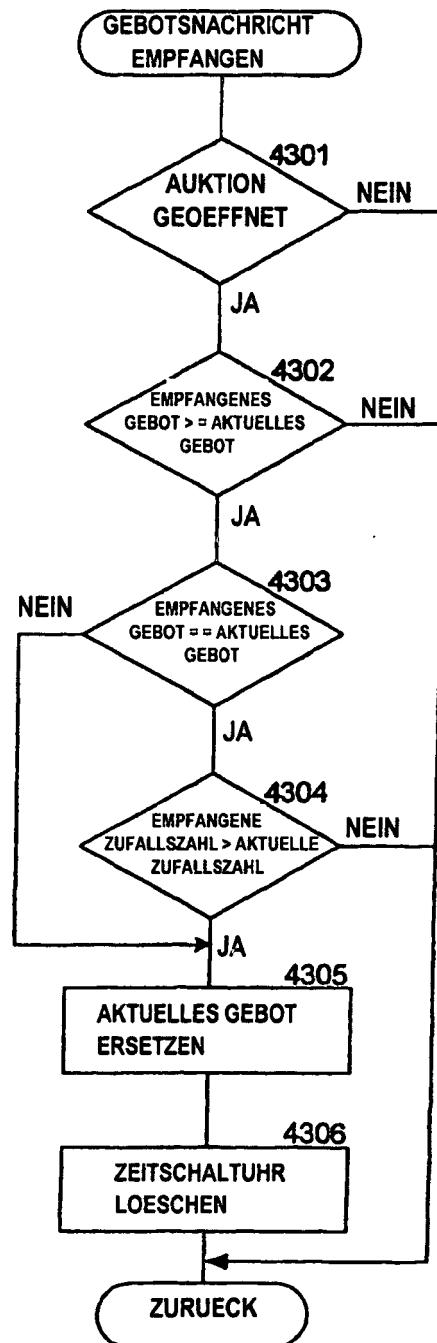
**Fig. 43**

Fig. 44

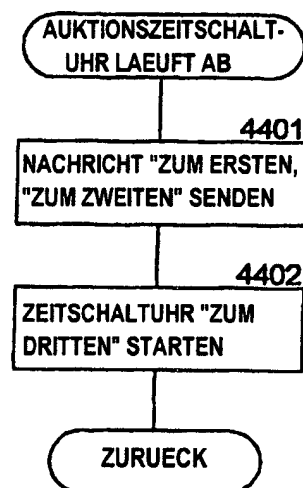


Fig. 45

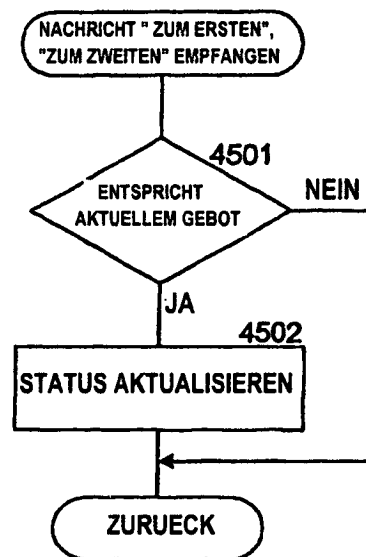


Fig. 46

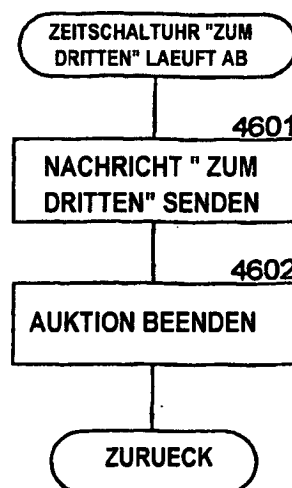


Fig. 47

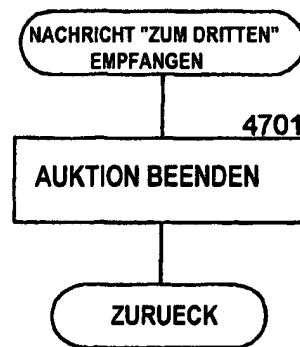


Fig. 48