TECHNISCHE UNIVERSITÄT DRESDEN

Center for Information Services and High Performance Computing (ZIH)

Theorie und Einsatz von Verbindungseinrichtungen in parallelen Rechnersystemen

Routing in statischen Verbindungsnetzwerken

03. Juni 2011

Andy Georgi & Jens Domke

INF 1046
Nöthnitzer Straße 46
01187 Dresden

0351 - 463 38783

ZIH
Center for Information Services & High Performance Computing

Foliensatz

Verfügbarkeit der Folien

Vorlesungswebseite:
`http://tu-dresden.de/die_tu_dresden/zentrale_einrichtungen/zih/lehre/ss2011/tevpr`

Agenda

Agenda

Routing I

Definition

Im Allgemeinen wird die Festlegung des Weges einer Nachricht von einer Quelle zu einer oder mehrerer Senken als *Routing* bezeichnet. Dabei greifen verschiedene, voneinander unabhängige, Mechanismen ineinander und tragen zum Datentransport bei. Im engeren Sinne zählen zum *Routing* der *Verbindungsaufbau*, die *Dekodierung* von Adressen, eine geeignete *Datenflusssteuerung* und optional eine *alternative Wegewahl* zur Leistungsoptimierung.

Routing II

- Forderungen an die Datenübertragung:
  - Eintreffen der Nachricht an der Senke in endlicher Zeit
  - Minimale Übertragungszeit
- Realisierung der minimalen Übertragungszeit durch Auswahl der Route mit der
  - kürzesten physikalischen Entfernung,
  - kleinsten Konfliktwahrscheinlichkeit oder
  - der geringsten Auslastung

Metriken für Verbindungsnetzwerke

- Wichtige Eigenschaften
  - Niedrige Latenz
  - Hohe Bandbreiten für Übertragungen
  - Deadlockfreiheit des Routingalgorithmus
- Etablierte Metriken zur Bewertung
  - Latenz
  - Bandbreite
  - Bisektionsbandbreite
  - effektive Bandbreite
  - effektive Bisektionsbandbreite

Bisektionsbandbreite

Definition

Summe der Bandbreiten der Links welche durchtrennt werden müssen, um das Netzwerk in zwei gleich große Teile zu partitionieren.

- Verbreitete Metrik für Verbindungsnetzwerke/Topologien
- Kein einbeziehen des Routings
- Beispiele
  - Ring: 2×Linkbandbreite
  - Vollvermaschtes Netz (n Knoten): $(\frac{n}{2})^2$×Linkbandbreite

Effektive Bisektionsbandbreite (eBB)

Definition [HSL09]

Entspricht der mittleren Bandbreite für beliebige Kommunikationsmuster zwischen zwei gleich großen Partitionen des Netzwerks.

- Eher anwendungsorientiertes Maß, welches zusätzlich das verwendete Routing betrachtet
- $eBB \in [0, 1]$
- Beispiel: $eBB = \frac{1}{2} \implies$ das Netzwerk liefert im Mittel die halbe theoret. Bandbreite, wenn alle Knoten kommunizieren
- Simulation der *eBB* mittels ORCS [SHL09] für geg. Topologiedateien
- Gemessene Approximation der *eBB* mittels Netgauge [HML07]

Motivation

Generelle Probleme für die meisten Routingalgorithmen

- Netzverkehr wird nicht global balanciert ⇒ "congestions" (Überlastung) reduzieren die Bandbreite
- Nur für eine spezielle Menge von Topologien entwickelt (z.B. FatTree Algorithmus)
- Nicht deadlockfrei für jede Topologie
- Nicht nutzbar auf Produktionssystemen (wegen zu hoher Laufzeit)

Algorithmen sollten auch für irreguläre Topologien funktionieren, da

- HPC-Systeme während ihrer Lebenszeit wachsen
- Zusätzliche Knoten, wie I/O- oder Loginknoten, sind angeschlossen
- Netzwerkkomponenten können ausfallen

Agenda

Begriffserklärung

Definition Deadlock [Tan07]

Eine Menge von Prozessen befindet sich in einem Deadlock-Zustand, wenn jeder Prozess aus der Menge auf ein Ereignis wartet, welches nur ein anderer Prozess der Menge auslösen kann.

Notwendige Kriterien [CoE71]

1. Betriebsmittel werden ausschließlich durch den Prozess freigegeben
2. Prozess fordert neue Betriebsmittel an, behält aber Zugriff auf aktuelle
3. Exklusiver Zugriff auf die Betriebsmittel
4. Zyklische Abhängigkeit der Betriebsmittel für zwei Prozesse

Deadlocks in Verbindungsnetzwerken

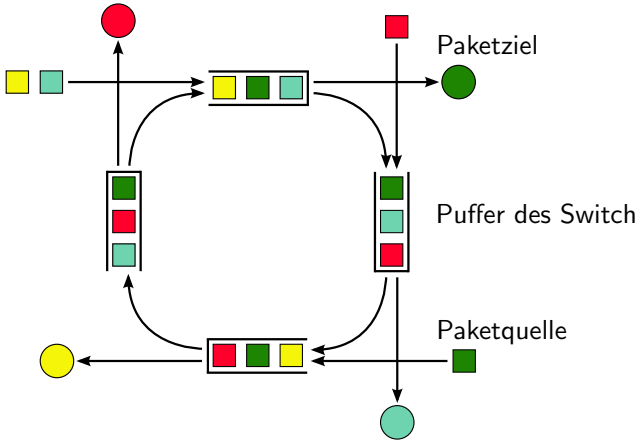


Abbildung: Deadlock Situation in einem Gitter-Netz

Ansätze für deadlockfreies Routing I

- Lebensdauer für Pakete (nur zum Auflösen)
- Controller Prinzip [Tou80]
- Up*/Down* [SBB91]
- Virtuelle Kanäle
  - Pro physischem Link mehrere virtuelle Kanäle
  - Dynamische Zuteilung von Datenrate und Speicherplatz
  - Reduzierung von Verzögerungszeiten und Ausschluss von Deadlocks möglich
  - Eliminierung von zyklischen Wartebedingungen durch Verteilung der Kommunikationen auf unterschiedliche virtuelle Kanäle
  - Formale Beschreibung mit Hilfe von *Kanalabhängigkeitsgraphen (engl. channel dependency graph)* [DaS87]

Ansätze für deadlockfreies Routing II

- Deadlockfreie Routingfunktion (z.B. Up*/Down*)
  - \+ Keine zeitaufwendige Zyklensuche nötig (s. Satz von Dally und Seitz [DaS87])
  - – Nicht für alle Topologien möglich (z.B. 2D-Torus)
  - – Einschränkung der Bandbreite durch nicht nutzbare Links
- Zyklen brechen, indem Pfade unterschiedlichen virtuelle Kanäle zugewiesen werden (z.B. LASH [SLT02])
  - \+ Höhere Bandbreite
  - \+ Bel. Topologien
  - – Zeitaufwendige Zyklensuche
  - – Anzahl der virtuelle Kanäle ist beschränkt

Kanalabhängigkeitsgraph

Definition 1

Eine **Route** $(c_1, c_2, \ldots, c_n)$ ist die Folge der Kanäle, auf denen ein Paket vom Sender zum Empfänger gelangt.
Eine **Routingfunktion** $R : C \times N \to C$, $R(c_i, n_d) := c_{i+1}$ weist jedem Paar aus aktuellem Kanal c_i und Zielknoten n_d den nächten Kanal zu, dabei gilt $c_i \neq c_{i+1}$.

Definition 2

Der **Kanalabhängigkeitsgraph** für eine Routingfunktion und ein Netzwerk ist der gerichtet Graph $CDG = G(C, E)$. Dabei gilt für die Kantenmenge E:

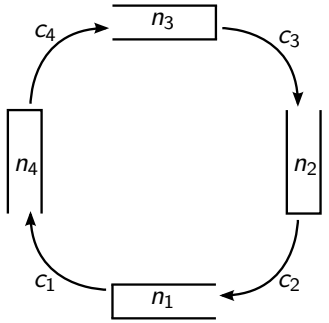
$$E = \{ (c_i, c_j) \mid R(c_i, n) = c_j \text{ für } n \in N \text{ bel. } \}$$

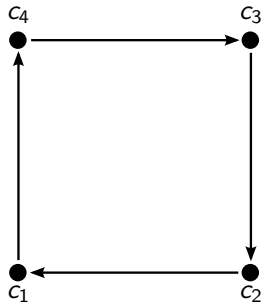
Satz von Dally und Seitz [DaS87]

Definition

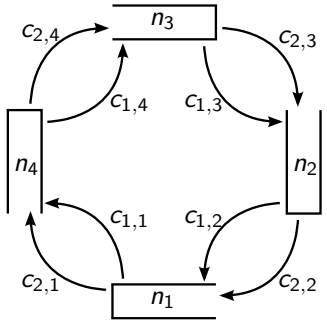
Eine Routingfunktion für ein Verbindungsnetzwerk ist genau dann deadlockfrei, wenn es keine Zyklen im zugehörigen Kanalabhängigkeitsgraphen gibt.

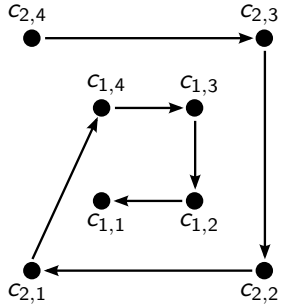
Virtuelle Kanäle und Kanalabhängigkeitsgraphen I



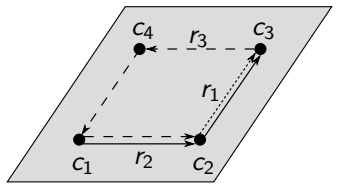


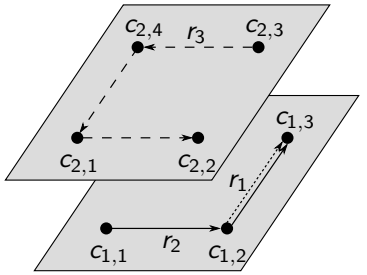
Virtuelle Kanäle und Kanalabhängigkeitsgraphen II





Virtuelle Kanäle und Kanalabhängigkeitsgraphen III





Livelock

Definition

Livelocks verhindern wie *Deadlocks* eine Datenübertragung in endlicher Zeit. Dabei verharren die Prozesse jedoch nicht in einem Zustand, sondern ändern diesen permanent, ohne das dabei der Zielzustand erreicht wird.

Agenda

Source Routing vs. Distributed Routing

Source Routing

Wird die Verbindung zwischen Quelle und Senke ausschließlich von der Quelle definiert, so spricht man vom *Source Routing*.

Distributed Routing

Beim *Distributed Routing* hingegen, wird lokal an jedem Zwischenknoten über den weiteren Verlauf der Nachricht entschieden.

Deterministisches vs. adaptives Routing

Deterministisches Routing

Deterministische Routing Verfahren verwenden ausschließlich die Adressen von Quelle und Senke als Berechnungsgrundlage. Somit ist der Weg einer Nachricht zwischen einem Sender und einem Empfänger durch das Verbindungsnetzwerk stets identisch.

Adaptives Routing

Adaptive Routing Verfahren beziehen hingegen weitere Informationen mit in die Berechnung des Wegs ein, so dass dieser zwischen zwei Kommunikationspartnern erst dynamisch zur Laufzeit definiert wird.

Progressives Routing vs. Backtracking

Progressives Routing

Bei Einsatz eines *progressiven Routing Verfahrens* ist eine getroffene Entscheidung endgültig, unabhängig vom weiteren Verlauf.

Backtracking

Backtracking Verfahren bieten die Möglichkeit beim Auftreten definierter Ereignisse die Rückverfolgung des bisher zurückgelegten Weges und die Auswahl alternativer Routen.

Umwegloses vs. umwegbehaftetes Routing

Umwegloses Routing

Bei Einsatz eines *umweglosen Routing Verfahrens* wird mit jedem Schritt die Entfernung zur Senke reduziert. Trifft die Nachricht auf eine Blockierung, so kann sie nur entlang solcher Verbindungsleitungen weiter vermittelt werden, die zu einem Weg gleicher Länge führen.

Umwegbehaftetes Routing

Bei Verwendung eines *umwegbehafteten Routing Verfahrens* wird die Beschränkung auf minimale Weglänge aufgehoben und ein Umweg einer Blockierung vorgezogen. Somit kann die Weglänge über dem Mindestabstand von Quelle und Senke liegen.

Vollständige vs. eingeschränkte Adaption

Vollständige Adaption

Die *Adaption* definiert Klassen von Wegen durch das Verbindungsnetzwerk, welche vom Suchalgorithmus genutzt werden können. Bei einer *vollständigen adaptiven Wegsuche* kann der Algorithmus beliebige Wege aus dieser Klasse nutzen.

Eingeschränkte Adaption

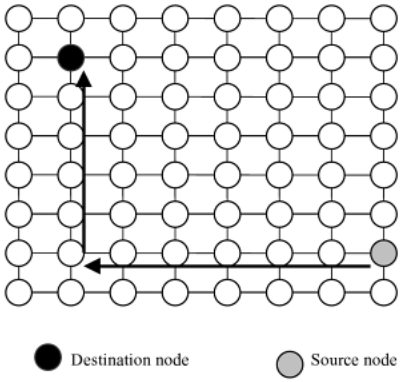
Die *eingeschränkte Adaption* erlaubt lediglich die Nutzung einer Untermenge aller verfügbaren Wege durch das Verbindungsnetzwerk.

Agenda

4. Deterministisches Routing
   - XY-Routing (Dimension Order Routing)
   - e-Cube-Routing (Verallgemeinerung von XY-Routing)
   - MinHop Routing
   - Single-Source-Shortest-Path Routing [HSL09]
   - Weitere Routingalgorithmen

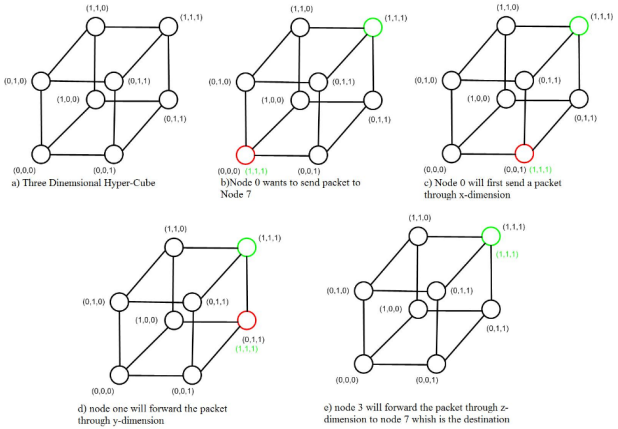
XY-Routing (Dimension Order Routing)

- Horizontale Vermittlung der Nachrichten bis zum Erreichen der Zielspalte
- Vertikale Transfers im Anschluss bis die Zielreihe erreicht ist
- Impliziert Deadlock-Freiheit



e-Cube-Routing (Verallgemeinerung von XY-Routing)

- Entwurf des *e-Cube-Algorithmus* für binäre n-Cube-Netze
- Nachricht durchläuft die Dimensionen eines Netzes stets in der gleichen Reihenfolge
- Keine zyklischen Abhängigkeiten



MinHop Routing

- Routet entlang der kürzesten Pfade im Netzwerk
- Basiert auf dem Dijkstra Algorithmus
- Balanciert Nachrichten nur auf multiplen Links zwischen zwei Switchen aber nicht global

Single-Source-Shortest-Path Routing [HSL09]

Algorithmus 1 SSSP Routingalgorithmus

Eingabe: Kontext des DFSSSP Routing
Ausgabe: Linear Forwarding Tabellen
/ N-zu-N, Multigraph Dijkstra Algorithmus */*
for all Port $\in$ Subnetz **do**
Dijkstra(...) für den Port als Quelle
Aktualisierung alle Linear Forwarding Tabellen
Verändere die Kantengewichte
end for

- Routet entlang der kürzesten Pfade im Netzwerk
- Basiert auf dem Dijkstra Algorithmus
- Update der Kantengewichte führt zur globalen Balancierung $\Rightarrow$ höhere eBB

Weitere Routingalgorithmen

- Up*/Down*
- Fat-Tree
- LASH
- ...

(Klassifikation der vorgestellten Routingalgorithmen: Selbststudium!)

Agenda

InfiniBand Verbindungsnetzwerk

- Technologie basiert auf einem offenem Standard, herausgegeben von der InfiniBand Trade Association
- Eine der gängigsten Netzwerktechnologien im HPC-Bereich

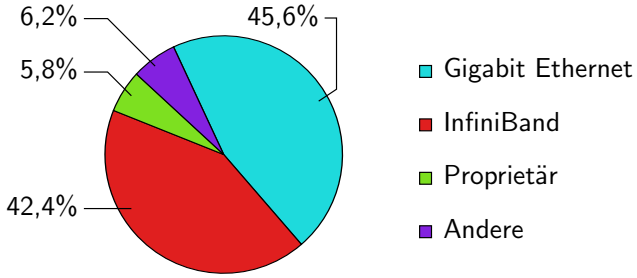
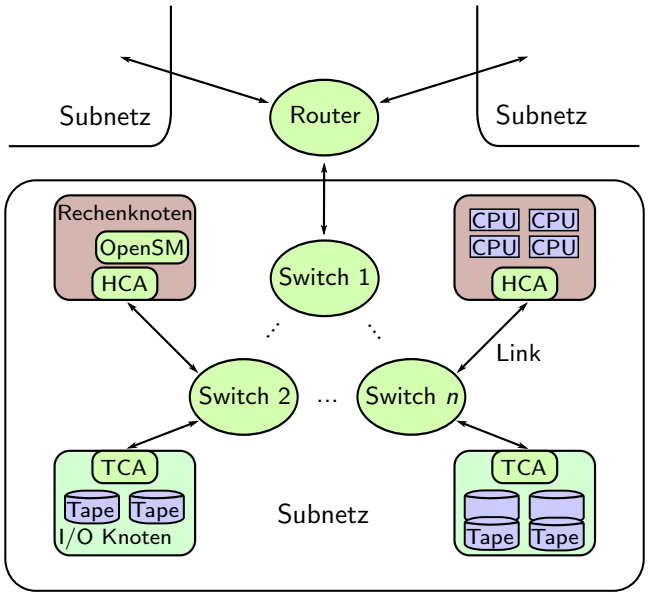


Abbildung: Top500 Liste, Verbindungsnetzwerke, Stand Nov. 2010

InfiniBand Verbindungsnetzwerk



InfiniBand Subnet Manager – OpenSM

Aufgaben

- Scannen des IB Subnetz
- Initialisieren der IB Ports
- Rekonfigurieren bei Änderungen des Subnetz

Arbeitsweise des OpenSM

- Verbindet sich mit dem IB Port auf der lokalen Maschine
- Scannt von da aus das angeschlossene Subnetz
  1. Sende Managementpaket zu jedem bekannten, nicht bearbeiteten Port
  2. Empfange jeweils eine Portinformation
  3. Auswerten der Portinformation
  4. Neue Ports zur Liste der bekannten Ports hinzufügen; Gehe zu 1.
- Berechnet die Routen für jedes Portpaar im Subnetz
- Schreibt die Linear Forwarding Tabellen der Switche
- Konfiguriert die IB Ports, unteranderem mit QoS Einstellungen
- Wartet im Leerlauf auf weitere Aufgaben

InfiniBand Subnet Manager – OpenSM

Implementierte Routingalgorithmen

- MinHop
- Up*/Down*
- Fat-Tree
- LASH
- DOR
- SSSP [HSL09]
- DFSSSP [Dom11]

Deadlockfreier SSSP Routingalgorithmus

Algorithmus 2 DFSSSP Routingalgorithmus

```
/* Phase 1: Identifikation der Komponenten */
Scan(...)
/* Phase 2: Routenberechnung */
SSSP(...)
/* Phase 3: Verteilung auf virtuelle Schichten */
DeadlocksBeseitigen(...)
/* Phase 4: Balancierung der virtuellen Schichten */
Balancierung(...)
```


Deadlockfreier SSSP Routingalgorithmus

Algorithmus 3 Entferne Deadlocks aus Kanalabhängigkeitsgraph (Phase 3)

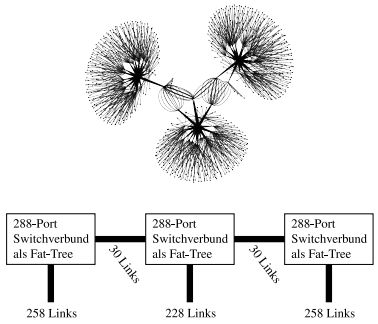
Eingabe: Linear Forwarding Tabellen
Ausgabe: Zuordnung der Routen zu virtuellen Schichten

```
/* Initialisierung der Schicht 1 */
for all Portpaare (Quelle, Ziel) do
    Aktualisiere CDG[1] mit Route von Quelle zum Ziel
end for
/* Suche Zyklen in den Kanalabhängigkeitsgraphen */
for i = 1, ..., max − 1 do
    repeat
        Führe Zyklensuche in CDG[i] aus
        Identifiziere "schwächste" Kante des Zyklus
        Verschiebe Portpaare/Routen der Kante in CDG[i + 1]
    until kein Zyklus gefunden in CDG[i]
end for
Führe Zyklensuche in CDG[max] aus
```

Deimos

Hochleistungsrechner am ZIH

- LNXII PC-Farm Deimos (13.9 TFlop/s)
- 726 Rechenknoten über 108 IB Switchen verbunden
- 2,6 GHz AMD Opteron X85 Dual Core
- 1, 2 oder 4 Prozessoren pro Knoten
- 2 GByte RAM pro Core



Netgauge [HML07]

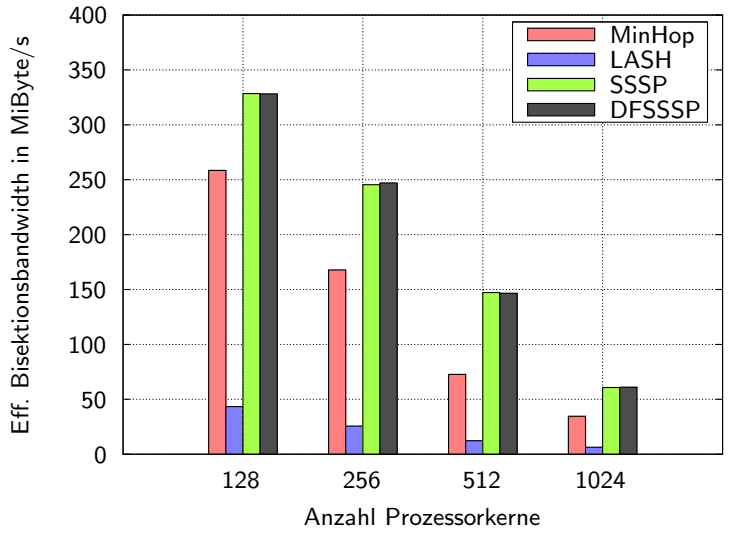


Abbildung: Approximation mittels 1000 Zufallsbisektionen

BenchIT [JBK04]

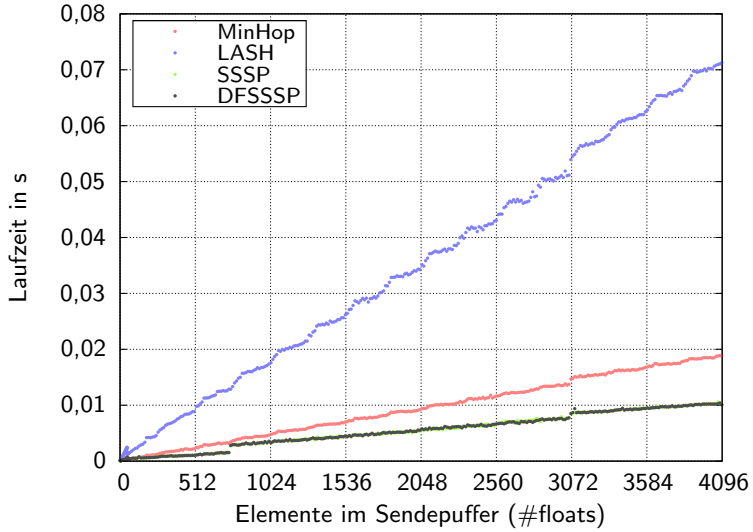


Abbildung: Kollektive *N*-zu-*N* MPI-Operation auf 128 Knoten

NAS Parallel Benchmarks [BHS95]

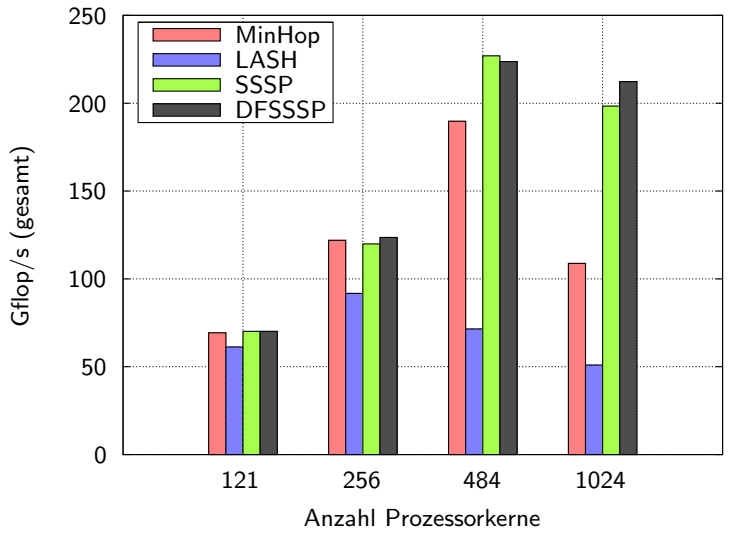


Abbildung: BT, Klasse C – Gleichungssystemlöser

Agenda

Idle-Algorithmus

- Grundlage des *Idle-Algorithmus* [GrR89] bildet ein deterministisches Routing-Verfahren
- Tritt eine Blockierung auf, wird der nächste Ausgang gewählt, welcher die Entfernung zur Senke reduziert
- Existiert dieser nicht, wird die bestehende Verbindung blockiert oder der Vermittlungsversuch beendet
- Umsetzung bspw. im Mad-Postman-Routing [Jel93]

Double-Y-Channel-Routing

- *Double-Y-Channel-Routing* [NiM93] beruht auf der Zuordnung der verfügbaren Kanäle zu mehreren unabhängigen Subnetzen
- Auswahl des entsprechenden Subnetzes für jede Verbindung
- Erfordert mehrere physische oder virtuelle Kanäle zwischen den Knoten

Duato's Methodik

- Erweiterung des Kanalabhängigkeitsgraphen 1991 durch *Duato* [Dua91a]
- Existiert eine Untermenge R_1 aller Routing-Kanäle, deren Abhängigkeitsgraph keine Schleifen aufweist, so ist der Algorithmus frei von Deadlocks

Turn-Modell

- Ziel des *Turn-Modells* [GIN92]: Zulassung von Umwegen, bei gleichzeitigem Ausschluss von Deadlocks
- Vorgehensweise:
  - Aufbrechen von Zyklen durch Unterbindung eines Wechsels in eine definierte Richtung
  - Erfordert das zunächst alle Transfers in die untersagte Richtung ausgeführt werden

A1-Algorithmus

- Ziel des A1-Algorithmus [ChS90]: Wegfindung innerhalb eines Cube-Netzes, insofern einer existiert
- Mitführung zusätzlicher Informationen notwendig:
  - Verbleibende Weglänge k
  - Noch zu durchlaufende Dimensionen d
  - Umwegvektor u
- Priorisierung der Wege minimaler Länge, andernfalls Markierung des Umwegs und der blockierten Verbindung im Umwegvektor u

Backtracking

- Rückverfolgung des bereits zurückgelegten Wegs im Falle einer Blockierung
- Mitführung, Speicherung und Auswertung von Informationen über bereits untersuchte Verbindungen
- Frei von Dead- und Livelocks

Vollständige Suche

Vollständige umwegbehaftete Suche

Das Verfahren der *vollständigen umwegbehafteten Suche* bezieht alle Wege zwischen Quelle und Senke in die Suche mit ein und garantiert damit, dass ein Weg gefunden wird, insofern einer existiert.

Vollständige profitable Suche

Bei dem Verfahren der *vollständigen profitablen Suche* werden alle Wege zwischen Quelle und Senke berücksichtigt, welche über die minimale Weglänge verfügen.

Eingeschränkte Suche

k-Familie

Bei Protokollen der *k-Familie* wird die Wegsuche durch die Unterteilung in zwei Phasen beschleunigt. Ist die Nachricht weiter als k Schritte von der Quelle entfernt, wird eine heuristische Suche eingesetzt. Erst im Anschluss wird die vollständige Suche verwendet.

Two-Phase Misroute Backtracking

Bei dem *Two-Phase Misroute Backtracking* erfolgt die Berechnung des Wegs in zwei Phasen. Ist der Abstand zur Quelle größer als eine definierte Konstante wird eine vollständige profitable Suche, andernfalls eine vollständige umwegbehaftete Suche durchgeführt.

Agenda

Literaturverzeichnis I

[Dom11] J. Domke, T. Hoefler, W. E. Nagel
Deadlock-Free Oblivious Routing for Arbitrary Topologies, May. 2011.
Proceedings of the 25th IEEE International Parallel & Distributed Processing Symposium (IPDPS), S. 613–624

[HSL09] T. Hoefler, T. Schneider, A. Lumsdaine
Optimized Routing for Large-Scale InfiniBand Networks, 2009.
17th Annual IEEE Symposium on High Performance Interconnects

[SHL09] T. Schneider, T. Hoefler, A. Lumsdaine
ORCS: An Oblivious Routing Congestion Simulator, Feb. 2009.
Techreport 675, Indiana University Computer Science

[HML07] T. Hoefler, T. Mehlan, A. Lumsdaine, W. Rehm
Netgauge: A Network Performance Measurement Framework, Sep. 2007.
Proceedings of High Performance Computing and Communications, HPCC'07, Springer Verlag, S. 659-671

Literaturverzeichnis II

[Tan07] A. Tanenbaum
Modern Operating Systems, Band 3, 2007.

[JBK04] G. Juckeland, S. Börner, M. Kluge, S. Kölling, W.E. Nagel, S. Pflüger, H. Röding, S. Seidl, T. William, R. Wloch
BenchIT – Performance measurement and comparison for scientific applications, 2004.
Parallel Computing - Software Technology, Algorithms, Architectures and Applications, S. 501–508

[SLT02] T. Skeie, O. Lysne, I. Theiss
Layered Shortest Path (LASH) Routing in Irregular System Area Networks, 2002.
IPDPS '02: Proceedings of the 16th International Parallel and Distributed Processing Symposium, S. 194-

Literaturverzeichnis III

- [BHS95] D. Bailey, T. Harris, W. Saphir, R. Wijngaart, A. Woo, M. Yarrow
  The NAS Parallel Benchmarks 2.0, 1995.
  Techreport NAS-95-020, NASA Ames Research Center
- [NiM93] L. M. Ni, P. K. McKinley
  A survey of wormhole routing techniques in direct networks, Feb 1993.
  IEEE Computer, Band 26, Nr. 2, S. 62-76
- [Jel93] C. R. Jesshope, C. Izu
  The MP1 Network Chip and its Application to Parallel Computers, 1993.
  The Computer Journal, Band 36, Nr. 8
- [GIN92] C. J. Glas, L. M. Ni
  The turn model for adaptive routing, May 1992.
  19th International Symposium on Computer Architecture, S. 278-287

Literaturverzeichnis IV

- [SBB91] M. Schroeder, A. Birrell, M. Burrows, H. Murray, R. Needham, T. Rodeheffer, E. Satterthwaite, C. Thacker
  Autonet: A High-Speed, Self-Configuring Local Area Network Using Point-to-Point Links, 1991.
  IEEE Journal on Selected Areas in Communications, Vol. 9, Nr. 8, S. 1318-1335
- [Dua91a] J. Duato
  Deadlock-free adaptive routing algorithms for multicomputers: Evaluation of a new algorithm, 1991.
  Third IEEE Symposium on Parallel and Distributed Processing, S. 840-847
- [Dua91b] J. Duato
  On the design of deadlock-free adaptive routing algorithms for multicomputers: Theoretical aspects, Apr 1991
  Distributed Memory Computing, Springer Verlag, S. 234-243

Literaturverzeichnis V

- [ChS90] M.-S. Chen, G. K. Shin
  Adaptive fault-tolerant routing in hypercube multicomputers, Dec 1990.
  IEEE Transactions on Computers, Band C-39, Nr. 12, S. 1406-1415
- [Tou80] S. Toueg
  Deadlock- and livelock-free packet switching networks, 1980.
  STOC '80: Proceedings of the twelfth annual ACM symposium on Theory of computing, S. 94-99
- [GrR89] D. Grunwald, D. Reed
  Analysis of Backtracking routing in binary hypercube computers, 1989.
  Technical Report UIUC-DCS-R-89-1486, University of Illinois
- [DaS87] W. J. Dally, C. L. Seitz
  Deadlock-free message routing in multi-processor interconnection networks, May 1987.
  IEEE Transactions on Computers, Band C-36, Nr. 5, S. 547-553

Literaturverzeichnis VI

- [CoE71] E. G. Coffman, M. J. Elphick, A. Shoshani
  System deadlocks, 1971.
  ACM Computer Surveys, Band 3, S. 67-78