www.teodosiu.net

Public Research and Development

1. Nikolaus Adrian Teodosiu: Development of a Parallel Logic Simulator based on the Time-Warp Optimistic Simulation Strategy. (in German) Master Thesis Nr. 989, Institut für parallele und verteilte Höchstleistungsrechner (IPVR) Stuttgart; Faculty of Computer Science, University of Stuttgart. Germany. April 13th 1993 (92 pages).

   Im Rahmen des PARASOL-Projekts am Institut für Parallele und Verteilte Höchstleistungsrechner entstehen zur Simulation von Digitalschaltungen nach verschiedenen Strategien arbeitende verteilte Simulatoren in einer Umgebung von unterstützenden Werkzeugen. Ziel der Diplomarbeit ist die Konzeption und Implementierung eines verteilten Simulators nach dem optimistischen Time Warp-Ansatz als auf einer Transputerkarte laufendem parallelen Prozeßsystem gewidmet. Die Implementierung wird konzeptionell detailliert vorgestellt.

   Die Arbeit ist in vier Kapitel gegliedert, denen zwei Verzeichnisse der verwendeten Abkürzungen und getroffenen Vereinbarungen vorangehen. Die verwendete Literatur wurde aus Gründen der spezifischen Bindung an recht unterschiedliche Gebiete auf die Kapitel verteilt. Die Darstellung ist folgendermaßen gegliedert:

   Kapitel A. Beschreibung des allgemeinen Problems der Simulation von Digitalschaltungen und der bestehenden Strategien zu ihrer Modellierung, mit Einstufung und Einführung in das Time Warp-Verfahren.

   Kapitel B. Beschreibung einzelner Aspekte des Verfahrens, mit Blick auf implementationsnahe Alternativen. Gleichzeitig wird ein Algorithmus neueren Datums zur effizienten Bestimmung der als Kontrollmechanismus des Verfahrens wichtigen Global Virtual Time vorgestellt.

   Kapitel C. Betrachtung im Einzelnen der Hardware-Umgebung und des eingesetzten Programmentwicklungs- und Botschaften-Systems, welche die Anwendung weitgehend bestimmen; Beschreibung der bestehenden, unterstützenden Werkzeuge innerhalb des PARASOL-Projekts und ihrer Schnittstelle zu den Simulatoren. Das Kapitel schließt mit einer einheitlichen Diskussion der statischen Modellierung innerhalb aller Simulatoren und ihrer spezifischen Implementierung für die Time Warp-Strategie.

   Kapitel D. Detaillierte Beschreibung der dynamischen Modellierungsaspekte innerhalb des vorliegenden Simulators und ihrer Implementierung, in Anlehnung an die Daten- und Kontrollstrukturen im Anhang.

   Jedes Kapitel fängt mit einer allgemeinen Darstellung der Aufgabe und der in den restlichen Abschnitten formulierten Problemen an. Abschnitte, welche für das direkte Verständnis der Anwendung bedeutend sind, wurden im Titel durch einen nachgestellten Stern hervorgehoben, die anderen können vorerst übersprungen werden.

2. Nilolaus Adrian Teodosiu: CHARTS - A Library of Graphics Classes for the Visualisation of Quantitative and Qualitative Relations. (in German) Study Thesis Nr. 799, Institut für Arbeitswirtschaft und Organisation (IAO), Frauenhofer Gesellschaft, Stuttgart; Faculty of Computer Science, University of Stuttgart. Germany. June 30th 1989 (33+69 pages).

   Die Graphik-Bibliothek CHARTS ist eine Sammlung von C++ Klassen zur Visualisierung von quantitativen und zeitlichen Zusammenhängen. Sie beinhaltet Klassen, die horizontale und vertikale Balkendiagramme, Kuchendiagramme und Funktionengraphdiagramme von mehreren zeitlichen Abhängigkeiten mit numerischer oder beliebiger Beschriftung der Abszisse realisieren.

   Die Implementierung ist in der objektorientierten Programmiersprache C++ erfolgt, mit dem ebenfalls in C++ geschriebenen Graphik-Paket InterViews als Basis für das Darstellen auf dem Bildschirm. (siehe Referenzen am Ende der Dokumentation für C++ und InterViews). Die Diagramme von Charts werden auf Sun-Computern mit graphischen Terminals visualisiert. Die Erstellung der Programme ist unter dem Betriebssystem UNIX erfolgt, mit dem Editor MAXed vom Betriebssystem SINIX der Firma Siemens. Das Compilieren und Linken der Klassen erfolgt mit dem Tool make, mit Hilfe einer Steuerdatei Makefile. Die Verwaltung der Source- und Objekt-Files wurde unter dem Mehrbenutzersystem SCCS vorgenommen. Diese Dokumentation wurde auf einem Macintosh unter Word 3.0 erstellt, und das Manual für die Klassen der Bibliothek mit troff auf der Sun unter UNIX, da es ein Teil der Bibliothek auf der Platte ist.

   Jeder Klasse entsprechen ein Header- und ein Implementierungsfile, wie allgemein in C++, die die Definition der Klasse, beziehungsweise den Source-Code der Funktionen enthalten. Die Files tragen meistens den Namen der Klasse, in der Form «Klasses.h, beziehungsweise <Klasse>.C Jedem Diagramm (chart) entspricht eine Klasse, und diese Klassen haben die gemeinsame Super-Klasse Chart. Der Rahmen des Charts wird von der Klasse FramedChart mit Scroll- und Zoom-Möglichkeiten des Charts realisiert. Die Balkendiagramme haben zusätzlich noch eine gemeinsame Superklasse, BarChart, die Funktionengraphen die Superklasse GraphChart. Die einzelnen Diagrammklassen auf der letzten Ebene sind HBarChart und VBarChart für die horizontalen und vertikalen Balkendiagramme, PieChart für die Kuchendiagramme und ValGraphChart und LabGraphChart für die Funktionengraphen. Die Files graphobj.h und graphobj.C enthalten Klassen, die beim Aufbau dieser Diagramme benutzt werden, wie horizontale und vertikale Strings und x- und y-Achsen. Zusätzlich zu den Bibliotheksklassen enthalt die Bibliothek eine Demo-Rahmenfunktion main, die die einzelnen Möglichkeiten der

Bibliotheksklassen darstellen soll. Die Klasse ChartsDemo enthält das Menu für die Steuerung der Insertionen dieser Diagramme auf dem Bildschirm in main und das Beenden der Demo.

3. Nilolaus Adrian Teodosiu: Message Passing and Routing in Processor Networks. (in German) Proseminararbeit, Institut für parallele und verteilte Höchstleistungsrechner (IPVR) Stuttgart; Faculty of Computer Science, University of Stuttgart. Germany. June 21st 1990 (33 pages).

   Parallele Rechnerarchitekturen, so wie wir sie hier ganz allgemein betrachten, gehen von Fernrechnernetzen (Wide Area Networks, WAN) und lokalen Rechnernetzen (Lokal Area Networks, LAN) über Multicomputersysteme (MCS) mit getrennten Speichersätzen bis zu Multiprozessorensystemen (MPS) mit gemeinsamem Speicher, siehe Ungerer (1989). Ein Spezialfall dieser Kategorie der MPS mit getrennten Speichersätzen sind die Transputernetze.

   Eine erster unterschied zwischen diesen Strukturen besteht in der Kopplungsentfernung. Ein zweiter, weitaus gewichtigerer Unterschied besteht in der Art der Kopplung. In ihrer Klassifikation unterscheidet man zwischen dem Schemata eines speichergekoppelten, bzw. eines nachrichtengekoppelten Systems. WANs, LANs, MCSs und einfache Transputernetze sind Beispiele von nachrichtengekoppelten Systemen, bei denen die Kommunikation zwischen Prozessen und Recheneinheiten durch Austausch von Nachrichten über ein Verbindungsnetz lauft. Diese Systeme sind für unsere Betrachtungen über die Laufwegbestimmung von Nachrichten (Routing) zunächst relevant. Andere Systeme, wie MPSs, sind speichergekoppelte Parallelstrukturen, bei denen die Kommunikation über Schreiben und Lesen in den gemeinsam zur Verfügung stehenden Speicher realisiert wird. Hybride Strukturen mit hardwaremäßiger Verteilung der Speichereinheiten zwischen den Prozessoren und einheitlicher Verwaltung, durch das Betriebssystem realisiert, werden hier ebenfalls nicht betrachtet.

   Diese Unterteilung kann auf Grund des Nachrichtenvermittlungsprinzips weiter verfeinert werden. Wir unterscheiden zwischen festen Verbindungen, die zeitinvariant sind, wie im Falle der Rechnernetze, und dynamischen Verbindungen, die per Software zur Laufzeit verändert werden, wie Kreuzschienenschalter. Letztere treten im Falle der Transputernetze mit Verteilerkopplung (Network Communication Unit, NCU) auf. Schließlich sind von den festen Verbindungen nur die Punkt-zu-Punkt-Verbindungen (P2P Connections) für uns relevant, die nach dem bekannten 'Store-and-Forward'-Prinzip arbeiten. Bussysteme, bei denen eine gesendete Nachricht nach dem 'Broadcast'-Prinzip von allen Beteiligten zunächst abgehört wird, und nachher dem Zielrechner zur Bearbeitung überlassen wird, sind nicht interessant, da die Laufwegbestimmungsalgorithmen somit entfallen. Punkt-zu-Punkt-Verbindungen treten in WANs und einigen MCPs und in den einfachen Transputernetzen ohne Verteilerkopplungen (NCUs) auf, Bussysteme meistens in LANs. In Abbildungen stellen wir die beiden Varianten von Nachrichtenvermittlung im Falle der Rechnernetze, und eine Kreuzschienenkopplung bei Speicher- und Nachrichtengekoppelten Systemen dar.

   Schließlich sei hier noch der Unterschied zwischen synchronem und asynchronem Kommunikationsprinzip erwähnt, auf den wir hier nicht näher eingehen. Rechnernetze und allgemeine MCPs arbeiten meist in asynchroner Kommunikation, Transputernetze, hardwaremäßig zumindest, wie wir hier zeigen werden, mit synchroner Kommunikation.

4. Nilolaus Adrian Teodosiu: Inverse Resolution in Deductive Systems. (in German) Hauptseminararbeit, International Business Machines (IBM) EMEA Böblingen; Faculty of Computer Science, University of Stuttgart. Germany. June 8th 1990 (24 pages).

   Die diesem Papier zugrundeliegenden Arbeiten von Muggleton & Buntine (1988) und Wirth (1989) sind mit der Frage beschäftigt, für wissensbasierte Systeme innerhalb der Prädikatenlogik erster Ordnung eine gewisse Form von Lernfähigkeit durch Umkehrung der logischen Schlussregeln zu kreieren. Dieser Aspekt des Lernens ist somit induktiv, im Gegensatz zu den weit verbreiteten deduktiven Systemen, die von den Hypothesen aus mittels Schlussregeln syntaktisch fundierte Schlusse ziehen, und geht vielmehr von zu lernenden Beispielen in Richtung hierfür notwendiger Voraussetzungen. Das entspricht der Auffassung von Michalski (1983) von Induktion als Umkehrung der Deduktion. Hierbei findet eine Komprimierung des Wissens statt. Die Repräsentation der Beispiele, des Hintergrundwissens und der induzierten Hypothesen bedient sich der in der logischen Programmierung weit verbreiteten Hornklauseln.

   In den praktisch arbeitenden Deduktionssystemen wird als logische Schlussregel die von Robinson (1965) eingeführte Resolution verwendet. Die natürliche Umkehrung der Resolution, die als inverse Resolution bezeichnet wird, bildet somit die Grundlage des induktiven Lernens von logischen Formeln durch Generalisierung, in der Form von Hornklauseln.

   Das allgemeine Paradigma induktiven Lernens wurde von Michalski (1983) folgendermaßen formuliert. Gegeben sind eine Menge von Aussagen über Beobachtungen (Fakten), als spezifisches Wissen über Objekte, F; eine vorläufige induktive Behauptung, die inkrementell während des Lernprozesses verbessert wird, I; eine Menge von Aussagen als Hintergrundwissen, B. Gesucht wird eine induktive Hypothese, H, die die Fakten impliziert und dem Hintergrundwissen nicht widerspricht. Das entspricht der Formel (B & H) |= F, oder B |= (H =>F), also die Hypothese ist eine semantische Generalisierung der Fakten im Raum (in der Theorie) des Hintergrundwissens.

   Da es um die Lernbarkeit von logischen Formeln geht, sind die vorgegebenen Fakten hier positive Beispiele von Grund-Hornklauseln, das Hintergrundwissen eine Menge von Regeln als Hornklauseln und die induzierte Hypothese eine Menge von logischen Formeln in Hornklauselform.