

Informatik-Fachberichte 190

Herausgegeben von W. Brauer
im Auftrag der Gesellschaft für Informatik (GI)

Dieter Maurer

Relevanzanalyse

**Eine Kombination von Striktheits-
und Datenflußanalyse zur effizienten
Auswertung funktionaler Programme**



**Springer-Verlag Berlin
Heidelberg GmbH**

Autor

Dieter Maurer
HighTec EDV-Systeme GmbH
Neue Bahnhofstraße 71, D-6670 St. Ingbert

CR Subject Classifications (1987): D.3.2, D.3.4, F.3.2

CIP-Titelaufnahme der Deutschen Bibliothek.

Maurer, Dieter:

Relevanzanalyse: e. Kombination von Striktheits- u. Datenflußanalyse zur effizienten Auswertung funktionaler Programme / Dieter Maurer. – Berlin; Heidelberg; New York; London; Paris; Tokyo: Springer, 1988

(Informatik-Fachberichte; 190)

Zugl.: Saarbrücken, Univ., Diss.

ISBN 978-3-540-50429-0

ISBN 978-3-642-74212-5 (eBook)

DOI 10.1007/978-3-642-74212-5

NE: GT

Dieses Werk ist urheberrechtlich geschützt. Die dadurch begründeten Rechte, insbesondere die der Übersetzung, des Nachdrucks, des Vortrags, der Entnahme von Abbildungen und Tabellen, der Funksendung, der Mikroverfilmung oder der Vervielfältigung auf anderen Wegen und der Speicherung in Datenverarbeitungsanlagen, bleiben, auch bei nur auszugsweiser Verwertung, vorbehalten. Eine Vervielfältigung dieses Werkes oder von Teilen dieses Werkes ist auch im Einzelfall nur in den Grenzen der gesetzlichen Bestimmungen des Urheberrechtsgesetzes der Bundesrepublik Deutschland vom 9. September 1965 in der Fassung vom 24. Juni 1985 zulässig. Sie ist grundsätzlich vergütungspflichtig. Zuwiderhandlungen unterliegen den Strafbestimmungen des Urheberrechtsgesetzes.

© by Springer-Verlag Berlin Heidelberg 1988

Ursprünglich erschienen bei Springer-Verlag Berlin Heidelberg New York 1988

2145/3140 – 543210 – Gedruckt auf säurefreiem Papier

Vorwort

In dem Bericht wird eine Technik zur Analyse von Computerprogrammen entwickelt, die in einer funktionalen Programmiersprache mit verzögerter Auswertung geschrieben sind. Solche Programmiersprachen (z.B. lazy HOPE, lazy ML, MIRANDA) behandeln Funktionen als Datenobjekte und erlauben insbesondere ihre Berechnung. Sie trennen Definition und Auswertung von Datenobjekten, wodurch die einfache Handhabung unendlicher Datenstrukturen ermöglicht wird. Sie bieten neue Modularisierungsmöglichkeiten und erleichtern so die Entwicklung sicherer und leicht wartbarer Software. Funktionale Programme können ohne die Notwendigkeit expliziter Synchronisation parallel ausgeführt werden.

Funktionale Programme spezifizieren einen Berechnungsprozeß auf einer hohen Abstraktionsstufe. Die Ausnutzung spezieller Eigenschaften eines Programms kann die Effizienz des daraus abgeleiteten konkreten Berechnungsprozesses wesentlich erhöhen. Oft ist eine mehr oder weniger komplexe Programmanalyse notwendig, um solche Eigenschaften zu erkennen. In dem Bericht stelle ich ein derartiges Analyseverfahren vor. Seine Ergebnisse ermöglichen während der Programmausführung Rückschlüsse auf *relevante Auswertungen*. Diese Information unterstützt in einer parallelen Implementierung die Schedulingentscheidungen und kann in einer sequentiellen Implementierung dazu benutzt werden, für gewisse Datenobjekte auf die Trennung von Definition und Auswertung zu verzichten und so den damit verbundenen Overhead zu vermeiden. Zur Analyse verwende ich neben einer einfachen Datenflußanalyse ein von mir entwickeltes und in dem Bericht beschriebenes Verfahren zur Striktheitsanalyse.

Der Bericht ist eine leicht überarbeitete Fassung meiner Dissertation. Sie wurde im Rahmen des Sonderforschungsbereiches 124 "VLSI und Parallelität", Teilprojekt C1 "Rechnerarchitekturen für funktionale Programmiersprachen" an der Universität des Saarlandes angefertigt. Ich möchte mich sehr herzlich bei meinem Doktorvater Herrn Prof. Dr. Reinhard Wilhelm für die zahlreichen Anregungen und Ermutigungen sowie bei dem zweiten Berichtersteller Herrn Prof. Dr. Klaus Indermark für sein Interesse an meiner Arbeit bedanken. Mein Dank geht ebenfalls an meine derzeitige Arbeitgeberin, die Firma HighTec EDV-Systeme GmbH in St. Ingbert, die mir den nötigen Freiraum für die Überarbeitung gewährt hat, sowie an den Springer-Verlag für die Möglichkeit, meine Dissertation auf diesem Wege einer breiteren Öffentlichkeit zugänglich zu machen.

Inhalt

I	Einleitung	1
II	Notationen, Konventionen und Hilfsmittel	5
	§1 Notationen und Konventionen	5
	§2 Algebren	7
	1 Abstrakte Interpretation	10
	§3 CPO's und stetige Verbände	12
	§4 Kategorien	15
	§5 Induktion.	17
	§6 Noether'sche Relationen und Vertauschbarkeit.	19
	§7 Termersetzungssysteme	26
	§8 Erweiterter λ -Kalkül	33
	1 Syntax	34
	2 Substitution	36
	3 α -Konversion und β -Reduktion	37
	4 (Denotationelle) Semantik	38
	5 Vertauschbarkeit mit $\equiv_\alpha$	39
III	Problemstellung	46
	§1 Die Eingabesprache	50
	1 Syntax	51
	2 Denotationelle Semantik	53
	3 Operationelle Semantik	55
	§2 Der Relevanzbegriff	72
	§3 Relevanzanalyse	89
IV	Striktheitsausdrücke	100
	§1 Syntax und Semantik	102
	1 Syntax	102
	2 Semantik	104
	§2 Abstrakte Striktheitsinterpretation	108
	§3 Die Präordnung $\leq$	115
	§4 Typisierung	130
	§5 Approximative Auswertung von Striktheitsskripten	137
	1 Basisverfahren	139
	2 Terminierungssichernde approximierende Operatoren	141

3 Zerlegung von Striktheitsskripten	145
4 Heuristische Vereinfachungen	146
5 Algorithmus (Beispiel)	149
V Relevanzanalyse	151
§1 Algorithmus I	152
§2 Algorithmus II	159
§3 Erweiterungen	178
1 Relevanz von Teilausdrucksvorkommen	178
2 Mächtigere Datenflußanalyse	179
3 Strukturierte Datenobjekte	183
§4 Anwendungen	185
1 Parallele Programmauswertung	185
2 Priorisierung spekulativer Auswertungen	191
3 Lokale Call-by-Value Parameterübergabe	192
§5 Implementierung	199
§6 Vergleich mit anderen Ansätzen	205
1 Der Algorithmus von Mycroft	205
2 Der Algorithmus von Kersjes	207
3 Der Algorithmus von Wray	211
4 Der Algorithmus von Hudak und Young	212
5 Der Algorithmus von Burn, Hankin und Abramsky	214
6 Der Algorithmus von Kuo und Mishra	217
7 Vergleich	219
Literatur	227
Notationen	233
Definitionen	236