Lehrstuhl für Informatik II

(Programmier- und Dialogsprachen sowie Compiler)

Leiter:

Prof. Dr. Hans Jürgen Schneider		

Mitarbeiter:

Arius, Peter, Dipl.-Inf.	(wiss. Angest., SFB)	
Behnsen, Wolfgang	(Programmierer)	
Betz, Wolfgang, Dipl.-Inf.	(wiss. Angest., SFB)	
Fischer, Ingrid, Dipl.-Inf.	(wiss. Hilfskr.)	
Frings, Sandra, Dipl.-Inf.	(wiss. Angest. - halbtags)	01. 06. - 31. 08. 94
Hasbargen, Thorsten, Dr.-Ing.	(wiss. Hilfskr.)	bis 31. 10. 94
Jacob, Christian, Dipl.-Inf.	(wiss. Hilfskr.)	
Keil, Martina, M.A.	(wiss. Hilfskr.)	
Lührs, Erni	(Sekretärin - halbtags)	
Minas, Mark, Dr.-Ing.	(wiss. Angest.)	beurlaubt: 01.09.93 - 31.08.94
Nilson, Jörg, Dipl.-Inf.	(wiss. Angest.)	ab 01. 06. 94
Schörmal, Elfriede	(Sekretärin - halbtags)	
Schorr, Ruth, Dr.-Ing.	(wiss. Angest.)	bis 31. 03. 94
Uebler, Manfred	(Programmierer)	
Viehstaedt, Gerhard, Dr.-Ing.	(wiss. Hilfskr.)	
Wilke, Peter, Dr.-Ing.	(Akad. Rat)	

Gastwissenschaftler:

Heidenreich, Georg, Dipl.-Inf.	(BMFT)	ab 18. 04. 94

Lehrbeauftragte:

Feder-Andres, Christiane, Dr.-Ing.	(Fa. sd&m GmbH, München)
Kips, Detlef, Dr.-Ing.	(Fa. BASYS, Erlangen)

1 Ein Generator für Diagrammeditoren

Ein Diagrammeditor ist ein syntaxgesteuerter Editor für eine bestimmte Klasse von Diagrammen. Ein Beispiel für eine Diagrammklasse sind Struktogramme, die auch als Nassi-Shneiderman-Diagramme bekannt sind. Die Verwendung eines Diagrammeditors für die Konstruktion eines Diagramms hat, verglichen mit der Benutzung eines allgemeinen Zeichenprogramms, zwei Vorteile. Zunächst können von einem Diagrammeditor spezifische Edieroperationen und Unterstützung für das Layout angeboten werden. Dies ermöglicht Benutzern ein wesentlich komfortableres Erstellen eines Diagramms. Außerdem steht in einem Diagrammeditor die formale Struktur des Diagramms zur Verfügung. Diese wird für die `Ausführung' eines Diagramms oder bei weiterer Verwendung des Diagramms in einem größeren Programmsystem gebraucht. Um den Aufwand für die Implementierung eines Diagrammeditors in akzeptablen Grenzen zu halten, wird ein Werkzeug benötigt. DiaGen (Diagram Editor Genenerator) ist ein solches Werkzeug und wurde am Lehrstuhl für Programmiersprachen konzipiert und implementiert.

Der Programmierer eines Diagrammeditors erstellt eine Spezifikation, die vom DiaGen-Parser verarbeitet wird und zusammen mit allgemeinen Laufzeitroutinen einen interaktiven Diagrammeditor ergibt. Die Spezifikation besteht dabei aus mehreren Teilen.

Eine Diagrammklasse wird durch eine (kontextfreie) Hyperkantengrammatik und eine Menge von (kontextsensitiven) Graphproduktionen spezifiziert. Ein bestimmtes Diagramm wird repräsentiert durch einen Ableitungsbaum, der auf der Hyperkantengrammatik basiert, und einen Graphteil, der sich durch Anwendung von Graphproduktionen ergibt. Terminalsymbole in der Repräsentation eines Diagramms werden auf dem Bildschirm abgebildet.

Für die Interaktion eines Benutzers mit dem Diagramm wurde ein Modell entwickelt, dem endliche Automaten zugrunde liegen. Dabei wurden insbesondere zwei bei der Interaktion mit Diagrammen charakteristische Probleme berücksichtigt, nämlich die Überlappung von Diagrammelementen und Abstraktion bei der Beschreibung zulässiger Benutzeraktionen.

Änderungen der Diagrammstruktur werden durch zwei Arten von Transformationen spezifiziert, die den Ableitungsbaum bzw. Graphteil in der Repräsentation eines Diagramms modifizieren. Das Hauptaugenmerk bei Transformationen in DiaGen liegt auf einer unabhängigen Manipulation des Ableitungsbaumes und des Graphteils eines Diagramms sowie auf einem relativ einfachen Modell für beide Arten von Transformationen. Bisher wurden ein Editor für Struktogramme und ein einfacher Editor für endliche Automaten mit DiaGen generiert. Erste Erfahrungen zeigen, daß sehr komfortable und mächtige Edieroperationen spezifiziert werden können.

(Viehstaedt)

2 Zyklisches Debugging paralleler Programme

Debugger sind Werkzeuge, die bei der Fehlersuche, Fehlerlokalisierung und Fehlerbehebung in Programmen unverzichtbar sind. Das gilt bei sequentiellen, aber um so mehr bei parallelen Programmen, da Parallelität, Nichtdeterminismus, komplexe Kausalität etc. zusätzliche Probleme hervorrufen. Im konkreten Fall der parallelen objektorientierten Programmiersprache pSather, die am ICSI in Berkeley, Kalifornien, entwickelt wurde, ist das Debugging paralleler Programme untersucht worden.

Da bei sequentiellen Programmen zyklisches Debugging weit verbreitet und die fast ausschließlich angewandte Debugging-Methode ist, sollte untersucht werden, inwieweit man dieses Verfahren auch bei pSather anwenden kann. Unter zyklischem Debugging versteht man dabei das Einkreisen von Programmfehlern durch mehrmaliges Wiederholen desselben Programmlaufs mit Hilfe eines Breakpoint-Debuggers. Voraussetzung ist allerdings ein deterministisches Programmverhalten, was für parallele und insbesondere pSather-Programme nicht der Fall ist.

Die zur Zeit einzig bekannte Vorgehensweise, den Nichtdeterminismus auszuschließen, ist deterministisches Replay: Während eines ersten Programmlaufs zeichnet das instrumentierte Programm genügend Information auf, so daß in den folgenden Programmausführungen das aufgezeichnete Programmverhalten an Hand dieser Daten exakt reproduziert werden kann. Das hinsichtlich Speicheraufwand günstigste der bekannten Verfahren, Instant Replay, ist nicht direkt bei pSather anwendbar: Prozesse können dynamisch und in beliebiger Zahl gestartet werden, sie können in beliebiger Weise zwischen den Rechnerknoten migrieren, und da alle Objekte in einem allen Rechnerknoten gemeinsamen Adreßraum liegen, ist jeder Objektzugriff im Grunde ein Kommunikationsvorgang. Der Speicheraufwand zum Aufzeichnen der Kommunikationsereignisse wäre unvertretbar hoch. Deshalb wurde ein modifiziertes Instant Replay-Verfahren vorgeschlagen und implementiert, das durch Bindung von Ereignisspuren an Rechnerknoten und eine Lauflängenkodierung eine effiziente Aufzeichnung erlaubt. Dadurch ist es nun möglich, vergleichsweise lang laufende Programme aufzuzeichnen (typische pSather-Programme auf einer CM-5 erzeugen Ereignisspuren von größenordnungsmäßig 100 kB pro Minute und Rechnerknoten) und pSather zyklisch zu debuggen.

Literatur:

T.J. LeBlanc/J.M. Mellor-Crummey: "Debugging parallel programs with Instant Replay", *IEEE Transactions on Computers*, C-36(4), 1987, S. 471-482.

(Minas)

3 NeuroGraph - Simulation neuronaler Netze und genetischer Algorithmen

Der Simulator für neuronale Netze (NeuroGraph) wurde weiterentwickelt. Die wichtigsten neuen Komponenten sind die Parallelisierung und Genetische Algorithmen.

Die Komponente für die genetischen Algorithmen erlaubt die interaktive Konstruktion genetischer Algorithmen auf der Basis vordefinierter genetischer Operatoren. Die graphische Bedienoberfläche zeigt dabei sowohl die Erzeugung der Individuen wie auch den Ablauf des Algorithmus an. Die Kombination von genetischen Algorithmen mit neuronalen Netzwerken ermöglicht die Optimierung der Topologie der neuronalen Netze. Dazu kann der genetische Algorithmus so konfiguriert werden, daß er eine Sequenz von Konstruktionsschritten eines neuronalen Netzes erzeugt. Diese wird dann von der Komponente für neuronale Netze ausgewertet und ein entsprechendes Netz generiert und dessen Leistungsfähigkeit ermittelt. Diese wird an den genetischen Algorithmus zurückgemeldet, der dann - abhängig von den Ergebnissen - eine neue Generation neuronaler Netze erzeugt, die durch Vererbung und genetische Operationen aus der Vorgängergeneration hervorgehen. Im Laufe der Zeit setzen sich dann die guten Eigenschaften durch und man erhält ein gegenüber der Ausgangssituation optimiertes neuronales Netz.

Die Parallelisierungskomponente ermöglicht die Beschleunigung der Lernphase durch Abbildung der neuronalen Netze und genetischen Algorithmen auf ein (auch heterogenes) Netzwerk von Workstations und PCs unter Unix oder ähnlichen Betriebssystemen. Dabei wird zunächst der Grad der möglichen Parallelisierung ermittelt und dem Benutzer eine Auswahl präsentiert, wobei unter verschiedenen Parallelisierungsstrategien ausgewählt werden kann. (Musterverteilung, Netzaufteilung uvm.). Praktische Bedeutung hat diese Komponente besonders dadurch, daß keine teuren Spezialrechner verwendet werden müssen, sondern die wesentlich häufiger anzutreffenden Workstation-Cluster und PC-Netzwerke zum Einsatz kommen.

(Nilson/Wilke)

4 Evolutionäres Programmieren mit MathEvolvica

Aus der natürlichen Evolution, die eine unglaubliche Vielfalt von zum Teil sehr spezialisierten Lebewesen mit erstaunlichen Fähigkeiten hervorgebracht hat, lassen sich Strategien zum Lösen von Lernaufgaben und Optimierungsproblemen ableiten: Wie sich Populationen von Organismen durch Evolution an veränderte Umweltbedingungen anpassen, so können Problemlösungsstrategien selektiv modifiziert und an eine veränderte Problemstellung adaptiert werden.

Mit Hilfe "evolutionärer" bzw. "genetischer" Programmiermethoden wird versucht, eine - zugegebenermaßen noch sehr eingeschränkte - Auswahl der aus der Natur bekannten evolutionären Verfahren und Operatoren nachzubilden und für die Evolution problemspezifischer Programme gewinnbringend einzusetzen. Eine Grundvoraussetzung der Programmevolution ist eine einheitliche Repräsentationsform der Programmelemente und Daten - in Analogie zur Kodierung der Erbinformationen von Lebewesen in der DNA-Doppelhelix. Neben der Programmiersprache LISP bzw. Scheme, in der die Liste als die Standardrepräsentation Verwendung findet, ist auch Mathematica eine funktionale Sprache, deren Elemente durchgehend aus Expressions (symbolischen Ausdrücken) aufgebaut sind. Weil das Computeralgebrasystem Mathematica darüber hinaus viele Annehmlichkeiten hinsichtlich Dokumentation und Visualisierung bietet, haben wir uns für Mathematica als Basissprache für unser System MathEvolvica zur Programmevolution entschieden.

Die Evolution der "Programmindividuen" läuft folgendermaßen ab: Aus einer Menge von parametrisierten, problemspezifischen Programmelementen wird eine Population von interpretierbaren Programmen erzeugt. Über mehrere Generationen werden die Programmkodierungen durch "genetische" Operatoren modifiziert oder miteinander rekombiniert (z. B. durch Mutation, Kreuzung, Kapselung, Verdoppelung, Löschung) und konkurrieren dabei untereinander um die beste Problemlösung. Die Selektion bzw. das Überleben der einzelnen Programme von einer Generation zur nächsten wird dabei von problemspezifischen Bewertungsfunktionen gesteuert, die auch die Interpretation hinsichtlich der vorgegebenen Problemstellung definieren.

Dieser Ansatz zur evolutionären Programmierung wurde bis jetzt auf folgenden Gebieten mit dem Ziel eingesetzt, unterschiedliche mit evolutionären Mechanismen verknüpfte Entwicklungsprogramme miteinander zu vergleichen:

- Topologieevolution neuronaler Netze mit integrierter Anpassung der Neuronenfunktionalitäten und Gewichte auf Basis evolutionärer Algorithmen,
- Nachbildung von Entwicklungsvorgängen in Zellenverbänden und künstlichen Pflanzen durch Evolution von L-Systemen (parallele Ersetzungssysteme),
- Simulation koevolutiver Mechanismen in einer zwei-dimensionalen Welt (BlocksWorld) mit interagierenden Organismen.

(Jacob)

Dissertation:

Viehstaedt, G.:
A Generator for Diagram Editors

Diplomarbeiten:

Abhau, D.:
Vergleichende Untersuchung objektorientierter Software-Entwicklungsverfahren
Andres, M.:
Eine Sprache für kontextsensitive Transformationen in Diagrammeditoren
Barthel, S.:
Ein Werkzeug zur Generierung der Interaktionssteuerung für Diagrammeditoren
Bauer, B.:
Entwurf und Implementierung von Programmen zur komfortablen Bedienung eines Simulatorsystems (NeuroGraph) unter X und Motif
Burghof, A.:
Entwurf und Implementierung einer objektorientierten Bedienoberfläche für evolutionäre Programmiersysteme
Burkert, A.:
Implementierung eines Resolvierers auf Klauselgrammatiken in C
Dormeyer, R.:
Konzeption geeigneter Datenstrukturen und Funktionen für die Lexikondatenbank
Felbinger, C.:
Füllstandsregelung schäumender Flüssigkeiten mit Fuzzy Control - Fuzzy Zapfhahn
Frings, S.:
Implementierung einer Tutorial-Komponente für einen Simulator für neuronale Netze (NeuroGraph)
Klein, W.:
Erweiterung der Ausgabe und Benutzeroberfläche für Diagrammeditoren
Krzymek, M.:
Strukturen des Wissens im Bereich Informatik bei Schülern und Lehrern der Sekundarstufe I im Verhältnis zu den Grundbegriffen der Informatik - Konsequenzen für die Lehreraus- und -fortbildung und für die Unterrichtsgestaltung im Fach Informatik -
Mansfeld, C.:
Entwurf und Implementierung einer Komponente für Genetische Algorithmen und Fuzzy-Logik eines Simulatorsystems für künstliche neuronale Netze (NeuroGraph)
Martin, D.:
Portierung und Weiterentwicklung eines Generators für grafische Oberflächen
Mohraz, K.:
Neuronale Netze mit dynamischer Architektur
Moll, R.:
Optimierungsprobleme bei der Programmierung von Steuerungen mit C++
Nilson, J.:
Entwurf und Implementierung eines Funktionen-Editors für ein Simulatorsystem für künstliche neuronale Netze (NeuroGraph)
Pohl, M.:
Entwurf und Implementierung eines anpaßbaren Disassemblers

Reitberger, G.:
Vergleichende Untersuchung der funktionalen (SA/SD) und der objektorientierten (OOA/OOD nach Rumbaugh) Vorgehensweise bei Systemanalyse und Systementwurf

Roloff, A.:
Theoretische Grundlagen des Unterrichtsfaches Informatik in Schulen - Probleme einer anschaulichen Darstellung unter Berücksichtigung der schulischen Erfordernisse auf der Sekundarstufe I -

Ruttkamp, U. A.:
Erarbeiten eines Konzepts zur Realisierung eines interaktiven Interpretersystems für Bedien- und Beobachtungsapplikationen

Schmid, B.:
BlocksWorld - eine Simulationsumgebung zur genetischen Programmierung

Schneider, A.:
Unterstützung des systematischen Modultests in C++

Scholz, A.:
Ein Werkzeug zur Generierung der Operationen für strukturelle Änderungen in Diagrammeditoren

Scholz, R.:
Implementierung und Vergleich von Algorithmen zur Simulation neuronaler Netze auf verteilten Rechnersystemen

Seifert, D.:
Konzeption und Implementierung einer zentralen Steuerung zu Phraseo-Lex

Waldmüller, G.:
Entwicklung und Implementierung eines komfortablen Hilfesystems zu Phraseo-Lex

---

Studienarbeiten:

Adelhardt, J.:
Ein quellsprachorientierter Editor für die objektorientierte Programmiersprache HERAKLIT - Realisierung eines Modulkonzepts -

Bauer, W.:
Korrektur des kontextfreien Teiles der Syntax und Überarbeitung des Typkonzeptes der IEC IS 1131-3

Evert, S.:
Graphische Darstellung des HERAKLIT-Systembauminhalts - Browser -

Faltin, N.:
Eine verteilte Datenbank zur Speicherung von Ereignisnetzen in HERAKLIT

Fischer, F.:
Eine Sprache für die Interaktion in Diagrammeditoren

Günthner, O.:
Evolutionäre Entwicklung neuronaler Netze durch stochastische Regelsysteme - Eine Implementierung in Mathematica -

Haake, B.:
Programm zur Fehlererkennung bei Speicherzugriffen mittels malloc() und free() bei C-Compilern

Haug, E.:
Optimierung des auf regulären Ausdrücken basierenden Mustervergleichs in IMP und MAGIC

Holland, B.:
Eine komfortable Bedienungsschnittstelle für HERAKLIT

Kraska, D.:
Implementierung einer Testumgebung zur regelbasierten, interaktiven Evolution neuronaler Netze

Lattka, M.:
Untersuchung und Manipulation des Ausführungszustandes in HERAKLIT - Maßnahmen zur Debugzeit -

Lehmkuhl, R.:
Entwicklung und Implementierung eines Algorithmus zur Generierung von Kantenzügen für einen interaktiven grafischen Editor

Lesch, R.:
Vergleichende Analyse verschiedener Methoden zur kurz- und mittelfristigen Prognose von Zeitreihen im Finanzbereich unter Verwendung Neuronaler Netze

Löser, A.:
Entwicklung und Implementierung des Grammatikregelsystems Phraseo-Gramm zur Verarbeitung verbaler Phraseologismen

Spilker, J.:
Ein neuronales Netz zur Diagnose eines Blutschwamms im Gehirn anhand von Gehirndurchblutungsmessungen

Stumpp, M.:
Nutzung der Flowback-Analyse in HERAKLIT

Wagner, D.:
Untersuchung und Manipulation des Ausführungszustandes in HERAKLIT -

Maßnahmen zur Übersetzungszeit -

Woithe, R.:

Entwurf und Implementierung einer Graphikoberfläche für eine linguistische Datenbank

Vorträge:

Fischer, I.:
"*Die inkrementelle Bildung von Diskursrepräsentationsstrukturen über einer Chart"*, KI-94, Workshop 'Prozedurale Anforderungen an die Sprachverarbeitung', Saarbrücken,
21. 09. 1994

Fischer, I.:
"*Die kompositionelle Bildung von Diskursrepräsentationsstrukturen über einer Chart*", (Posterpräsentation), KONVENS '94, Konferenz 'Verarbeitung natürlicher Sprache', Wien,
28. 09. 1994

Jacob, C.:
"*Mathematica - Ein leistungsfähiger Taschenrechner für den Mathematik- und Informatikunterricht*", Tag der Erlanger Informatik, Erlangen,
29. 04. 1994

Jacob, C.:
"*Evolutionäres Programmieren*", Forum Fuzzy, FORWISS, Erlangen,
17. 05. 1994

Jacob, C.:
"*Evolution of Typed Expressions describing Artificial Nervous Systems*", (Posterpräsentation), International Conference on Artificial Neural Networks, Sorrento, Italien,
25. 05. 1994

Jacob, C.:
"*XNeuroGene: A system for evolving artificial neural networks*", (Posterpräsentation), International Conference on Artificial Neural Networks, Sorrento, Italien,
26. 05. 1994

Jacob, C.:
"*Typed Expressions Evolution of Artificial Nervous Systems*", IEEE World Congress on Computational Intelligence, Orlando, USA,
30. 06. 1994

Jacob, C.:
"*Evolutionary Computation with Mathematica*", 1994 European Mathematica Conference, Oxford, U.K.,
19. 09. 1994

Jacob, C.:
"*Evolutionary Programming of Artificial Nervous Systems*", 39. Internationales Wissenschaftliches Kolloquium der TU Ilmenau,
27. 09. 1994

Jacob, C.:
"*Genetic L-System Programming*", Parallel Problem Solving from Nature - PPSN III, International Conference on Evolutionary Computation, Jerusalem, Israel,
13. 10. 1994

Keil, M.:

"*Phraseo-Lex - eine Lexikondatenbank zur systematischen Repräsentation von Phraseologismen*", Arbeitsgemeinschaft Sprache und Computer, Institut für Linguistische Informatik der Universität Erlangen,
22. 02. 1994

Keil, M.:
"A*nalyse von Partikeln für ein sprachverstehendes System - am Beispiel telefonischer Zugauskunftsdialoge*", VERBMOBIL-Workshop zum Thema "Partikeln im Diskurs", Universität Hamburg,
09. 05. 1994

Keil, M.:
"*Phraseo-Lex - eine Lexikondatenbank zur systematischen Repräsentation von Phraseologismen*", IBM Heidelberg,
20. 05. 1994

Keil, M.:
"*Systematische Repräsentation verbaler Phraseologismen und deren Eigenschaften im Lexikon*", KONVENS '94, Konferenz 'Verarbeitung natürlicher Sprache', Wien,
28. 09. 1994

Minas, M.:
"*Debugging of pSather, a parallel object-oriented programming language*", First General Meeting of the Parallel Tools Consortium, NASA Ames Research Center, Moffett Field, California,
09. 06. 1994

Minas, M.:
"*Fault detection for sequentially controlled machines using temporal constraint nets*", IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes (Safeprocess '94), Helsinki, Finnland
15. 06. 1994

Minas, M.:
"*Cyclic debugging for pSather, a parallel object-oriented programming language*", DAGS'94 Symposium, Dartmouth Institute for Advanced Graduate Studies in Parallel Computation, Hanover, NH, USA
07. 07. 1994

Minas, M.:
"*Debugging paralleler Systeme*", BASYS-Betriebsseminar, Schwend/Bayern
25. 11. 1994

Pohl, M.:
"*Building Safety Tools by Generators*", Workshop 'Semantikgestützte Analyse, Entwicklung und Generierung von Programmen' der GI-Fachgruppe 2.1.3, Schloß Rauischholzhausen,
11. 03. 1994

Schneider, H. J.:
"*Graph Grammars as a Tool to Define the Behaviour of Process Systems: From Petri Nets to Linda*", 5th Int. Workshop on Graph Grammars, Williamsburg, Va./USA,
14. 11. 1994

Viehstaedt, G.:
"*Interaction in really graphical user interfaces*", 1994 IEEE Symposium on Visual

Languages, St. Louis, USA
07. 10. 1994

Wilke, P.:
"*Neuronale Netze: Elektronengehirn oder Nürnberger Trichter?*", Ringvorlesung "Chaos, Selbstorganisation, Neuronale Netze", Universität Passau, Passau,
11. 01. 1994

Wilke, P.:
"*Einführung in Neuronale Netze*", Carl-Cranz-Gesellschaft, Oberpfaffenhofen,
12. 07. 1994

Wilke, P.:
"*Die Kohonen-Karte*", Carl-Cranz-Gesellschaft, Oberpfaffenhofen,
13. 07. 1994

Wilke, P.:
"*Marktübersicht Simulationswerkzeuge für neuronale Netzwerke*", Carl-Cranz-Gesellschaft, Oberpfaffenhofen,
14. 07. 1994

Wilke, P.:
"*The NeuroGraph Neural Network Simulator*", Fa. Parallel Performance Group, Sedona, AZ, USA,
16. 09. 1994

Wilke, P.:
"*The NeuroGraph Neural Network Simulator*", Port Authority New York, New York, NY, USA,
19. 09. 1994

Wilke, P.:
"*The NeuroGraph Neural Network Simulator*", Fa. Entropic, Washington, DC, USA,
24. 09. 1994

Wilke, P.:
"*Verteilte Simulation Neuronaler Netze mit NeuroGraph V3.0*", Transputer Anwender-Treffen TAT 94, Aachen,
27. 09. 1994

Wilke, P.:
"*NeuroGraph - A Neural Network Simulation Environment with Components for Genetic Algorithms and Fuzzy Logic*", 39. Internationales Wissenschaftliches Kolloquium der TU Ilmenau,
29. 09. 1994

Wilke, P.:
"*Simulation und Visualisierung neuronaler Netze und genetischer Algorithmen mit NeuroGraph*", Universität Tübingen, Tübingen,
26. 10. 1994

Wilke, P.:
"*Simulation und Visualisierung neuronaler Netze und genetischer Algorithmen mit NeuroGraph*", Universität Stuttgart, Stuttgart,
09. 11. 1994

Veröffentlichungen:

Arius, P.:
"Dynamischer Objektaustausch im HERAKLIT-System. In: Wedekind, H. (Hrsg.): "*Verteilte Systeme: Grundlagen und zukünftige Entwicklungen*". BI-Wissenschaftsverlag, Mannheim, 1994, S. 343 - 346.

Fischer, I.:
"Die inkrementelle Bildung von Diskursrepräsentationsstrukturen über einer Chart". - In: '*KI-94 Workshops - Extended Abstracts*', J. Kunze/ H. Stoyan (Hrsg.), Workshop 'Prozedurale Anforderungen an die Sprachverarbeitung', Saarbrücken, 21./22. 09. 1994. - Bonn: Gesellschaft für Informatik e.V., 1994, S. 174 u. 175.

Fischer, I.:
"Die kompositionelle Bildung von Diskursrepräsentationsstrukturen über einer Chart". - In: *Tagungsband KONVENS '94 - Verarbeitung natürlicher Sprache*, H. Trost (Hrsg.), Wien, 28. - 30. 09. 1994. - Berlin: Springer, 1994, S. 419 - 422.

Geiger, H. / Greska, W. / Wilke, P. / Berentz, J.:
"Design and Implementation of Neural Networks for Matching Contours in CAD Models", *Proc. ICANN'94 Int. Conference on Artificial Neural Networks*, Amsterdam, Niederlande, Springer-Verlag, 1994

Heidenreich, G. / Kips, D.:
"Projektflußgraphen - ein Meta-Modell zur Unterstützung der Qualitätssicherung im Software-Engineering". - In: *Proc. 'QUALITY'94 - Fünfte Intern. Fachmesse und Kongreß für Qualitätssicherung'*, Stuttgarter Messe- & Kongreßgesellschaft mbH (Hrsg.), Stuttgart, 17. 05. - 20. 05. 1994 - Ulm, Müller Adreß- und Neue Medien GmbH, 1994, S. 282 - 287.

Hindel, B. / Pohl, M.:
"Building Safety Tools by Generators". In: '*Semantikgestützte Analyse, Entwicklung und Generierung von Programmen'*, G. Snelting/U. Meyer (Hrsg.), Workshop der GI-Fachgruppe 2.1.3, Schloß Rauischholzhausen, 10./11. 03. 1994. - Berichte der ARBEITSGRUPPE INFORMATIK 9402, 1994, S. 155 - 165.

Jacob, C.:
"Evolution of Typed Expressions describing Artificial Nervous Systems", *Proc. International Conference on Artificial Neural Networks*, Sorrento, Italien (1994), Springer-Verlag, Berlin, 1994, S. 258 - 262.

Jacob, C.:
"XNeuroGene: A system for evolving artificial neural networks", *Proc. International Conference on Artificial Neural Networks*, Sorrento, Italien (1994), Springer-Verlag, Berlin, 1994, S. 739 - 742.

Jacob, C.:
"Typed Expressions Evolution of Artificial Nervous Systems", *Proc. First IEEE Conference on Evolutionary Computation,* Orlando, Florida, USA, 1994, S. 502 - 507.

Jacob, C.:
"Evolutionary Programming of Artificial Nervous Systems", *Tagungsband 39. Internationales Wissenschaftliches Kolloquium der TU Ilmenau*, Vortragsreihen

der TU Ilmenau, 1994, S. 31 - 38.

Jacob, C.:
"Genetic L-System Programming", *Parallel Problem Solving from Nature - PPSN III, International Conference on Evolutionary Computation*, Lecture Notes in Computer Science 866, Springer Verlag, Berlin, 1994, S. 334 - 343.

Keil, M.:
"Systematische Repräsentation verbaler Phraseologismen und deren Eigenschaften im Lexikon". - In: *Tagungsband KONVENS '94 - Verarbeitung natürlicher Sprache*, H. Trost (Hrsg.), Wien, 28. - 30. 09. 1994. - Berlin: Springer, 1994, S. 181 - 190.

Minas, M.:
"Fault Detection for Sequentially Controlled Machines Using Temporal Constraint Nets". - In: *Preprints of the 1994 IFAC Symposium on Fault Detection and Safety for Technical Processes (Safeprocess '94)*, Helsinki, 13. - 16. June 1994. - Finnish Society of Automation, 1994, S. 323 - 328.

Minas, M.:
"Cyclic debugging for pSather, a parallel object-oriented programming language". In: *Proceedings of the DAGS'94 Symposium*, Hanover, NH, USA, July 1994. Dartmouth Institute for Advanced Graduate Studies in Parallel Computation.

Schneider, H. J.:
"Programmiersprachen zur Beschreibung paralleler Algorithmen". - Erscheint in: '*Parallelrechner: Architekturen, Systeme, Werkzeuge*', K. Waldschmidt et al. (Hrsg.), Stuttgart: Teubner-Verlag

Schneider, H. J. / Ehrig, H. (Hrsg.):
"*Graph Transformations in Computer Science*", Lecture Notes in Computer Science 776, Berlin: Springer, 1994. 395 S.

Viehstaedt, G. / Minas, M.:
"Interaction in really graphical user interfaces". In: *Proc. 1994 IEEE Symposium on Visual Languages,* Los Alamitos, CA, USA, Oct. 1994. IEEE Computer Society Press, 1994, S. 270 - 277.

Wilke, P.:
"NeuroGraph - A Neural Network Simulation Environment with Components for Genetic Algorithms and Fuzzy Logic", *Tagungsband 39. Internationales Wissenschaftliches Kolloquium der TU Ilmenau*, Vortragsreihen der TU Ilmenau, 1994, S. 153 - 158.

Wilke, P. / Scholz, R.:
"Verteilte Simulation Neuronaler Netze mit Neurograph V3.0", *Proc. Transputer Anwender-Treffen TAT94*, 1994