RWTH Aachen

Department of Computer Science
Technical Report

Tangent-Linear Models by Augmented LL-Parsers

Uwe Naumann and Andre Vehreschild

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2005-19

RWTH Aachen · Department of Computer Science · September 2005

Tangent-Linear Models by Augmented LL-Parsers

Uwe Naumann and Andre Vehreschild

Department of Computer Science, RWTH Aachen University
52056 Aachen, Germany
naumann@stce.rwth-aachen.de, vehreschild@sc.rwth-aachen.de

Abstract. We describe a novel method for the generation of tangent-linear code by augmentation of LL-parsers generated by the software tool ANTLR. The main advantage of this approach to source code augmentation is the missing requirement for an internal representation of the original program. We consider this work as the basis for further investigations into how far this technique can be extended in the context of more sophisticated transformations, for example, the automatic generation of adjoint codes. Our prototype tool AD_C_ANTLR currently accepts a subset of the ANSI C standard. We discuss its theoretical basis, and we present case studies to underline the elegance of the parser-based approach to source augmentation.

1 Motivation and Summary of Results

What are tangent-linear models good for? Consider the elliptic partial differential equation

$$-\Delta y - \lambda e^y = 0 \tag{1}$$

on the unit square Ω where $y(x_1, x_2) = 0$ on $\partial\Omega$.[1] The Laplacian operator is defined as

$$\Delta \equiv \frac{\partial^2}{\partial x_1^2} + \frac{\partial^2}{\partial x_2^2} \quad .$$

It can be solved using a centered finite difference scheme by approximating the first derivative with respect to x_1 and x_2 at the midpoints of the discretization intervals

$$a = (\frac{x_i^1 - x_{i-1}^1}{2}, x_j^2), \quad b = (\frac{x_{i+1}^1 - x_i^1}{2}, x_j^2),$$

$$c = (x_i^1, \frac{x_j^2 - x_{j-1}^2}{2}), \text{ and } d = (x_i^1, \frac{x_{j+1}^2 - x_j^2}{2})$$

as

$$\frac{\partial y}{\partial x_1}(a) \approx \frac{y_{i,j} - y_{i-1,j}}{h}, \quad \frac{\partial y}{\partial x_1}(b) \approx \frac{y_{i+1,j} - y_{i,j}}{h}$$

and

$$\frac{\partial y}{\partial x_2}(c) \approx \frac{y_{i,j} - y_{i,j-1}}{h}, \quad \frac{\partial y}{\partial x_2}(d) \approx \frac{y_{i,j+1} - y_{i,j}}{h} \quad .$$

We have chosen uniform step size h both in the x_1 and x_2 directions such that $1/h \in I\!N$. Similarly, the second derivatives with respect to x_1 and x_2 at (x_i^1, x_j^2)

[1] Our intention is not to discuss state-of-the-art methods for solving this problem. We would rather like to give an intuitive example to motivate the need for tangent-linear models. Similar situations occur in the context of more sophisticated numerical methods.

can be approximated as

$$\frac{\partial^2 y}{\partial x_1^2}(x_i^1, x_j^2) \approx \frac{1}{h}\left[\frac{y_{i+1,j} - y_{i,j}}{h} - \frac{y_{i,j} - y_{i-1,j}}{h}\right]$$
$$= \frac{y_{i+1,j} - 2 \cdot y_{i,j} + y_{i-1,j}}{h^2}$$

and

$$\frac{\partial^2 y}{\partial x_2^2}(x_i^1, x_j^2) \approx \frac{1}{h}\left[\frac{y_{i,j+1} - y_{i,j}}{h} - \frac{y_{i,j} - y_{i,j-1}}{h}\right]$$
$$= \frac{y_{i,j+1} - 2 \cdot y_{i,j} + y_{i,j-1}}{h^2} \quad .$$

It follows that a numerical approximation to the solution of the partial differential equation can be computed by solving the discretized problem

$$\begin{aligned} f(y) = &- \frac{y_{i+1,j} - 2 \cdot y_{i,j} + y_{i-1,j}}{h^2} \\ &- \frac{y_{i,j+1} - 2 \cdot y_{i,j} + y_{i,j-1}}{h^2} - \lambda e^{y_{i,j}} = 0 \end{aligned} \tag{2}$$

for $i, j = 1, \ldots, 1/h - 1$. Since the values of y on $\partial\Omega$ are known Eqn. (2) is a system of $(1/h-1)^2$ nonlinear equations in the same number of unknowns. Given a good start estimate y^0 it can be solved by Newton's method with quadratic convergence as follows:

$$\delta y^i = -(f'(y^i))^{-1} \cdot f(y^i)$$
$$y^{i+1} = y^i + \delta y^i$$

for $i = 0, \ldots$ At each step the algorithm requires the Jacobian f' of f at the current estimate y^i. Alternatively the Newton step can be obtained as the solution of the linear system

$$f'(y^i)\delta y^i = -f(y^i) \tag{3}$$

at each Newton iteration. Direct methods may be prohibitive due to the size of the problem. Iterative methods are likely to be more suitable. For example, iterative refinement computes the iterates as

$$\delta y^{i+1} = \delta y^i + B(f(y^i) - f'(y^i)\delta y^i) \tag{4}$$

for a suitable preconditioner B. Note that Eqn. (4) involves the computation of a Jacobian-vector product. This is done by a tangent-linear code that can be generated efficiently from a routine implementing f by the source transformation technique proposed in this paper. There is no need to form the whole Jacobian explicitly.

Tools for *automatic differentiation (AD)* [25, 16, 10, 4, 9, 7] by source transformation such as ADIFOR2.0 [6], its successor ADIFOR3.0 [8], TAF [15], the commercial successor of TAMC [14] distributed by FastOpt GmbH, TAPENADE [19], the successor of Odyssee [13], ADiMat [5], and OpenAD [26] transform numerical simulation programs into an abstract internal representation (annotated parse tree, symbol tables, control flow graphs, call graph). The transformation algorithms operate on this often very complex data structure to obtain the semantically modified code that computes derivatives of various kinds. The intermediate representation allows for a large number of static analyses to be performed with the objective to optimize the resulting transformed code. However, the development of a complete compiler front-end is a difficult and time-consuming task. Many AD-related code transformations can be integrated directly with the parser. Parser generators such as Bison [12] and ANTLR [23] provide a very convenient platform for this integration.

Our main intention is to investigate the level of complexity in terms of semantic transformations that can be realized by a single-pass compiler through augmentation of the parser. Initially we focus on LL(k)-parsers [1] for a subset of ANSI C that are generated by ANTLR. Similar, investigations are underway for LR(1)-parsers generated by Bison. In this paper we present a feasibility study for the parser-based generation of tangent-linear code. After a brief introduction to the underlying principles of AD in Sec. 2 we present algorithmic details of the augmented LL-parser in Sec. 3. In Sec. 4 we present our prototype tangent-linear model compiler AD_C_ANTLR followed by case studies in Sec. 5. Conclusions are drawn in Sec. 6 together with an outlook to the possible application of similar techniques in the context of adjoint codes.

2 Background: Automatic Differentiation

For a given implementation of a vector function

$$f : I\!R^n \rightarrow I\!R^m$$

as a computer program[2] we use AD by source transformation to generate code for computing first derivatives of f. Therefore the computation of $\mathbf{y} = f(\mathbf{x})$ is expected to decompose into a sequence of elemental assignments

$$v_j = \varphi_j(v_i)_{i \prec j} \tag{5}$$

for $j = 1, \ldots, p+m$ and $i \prec j$ if and only if v_i is an argument of φ_j. Eqn. (5) is referred to as the *code list* of f. We set $v_{i-n} = x_i$ for $i = 1, \ldots, n$ and $v_{p+j} = y_j$ for $j = 1, \ldots, m$. The v_k, $k = 1-n, \ldots, p+m$, are called *code list variables*. Forward mode AD transforms f into the tangent-linear model $\dot{f}$ that computes a total (or directional) derivative $\dot{\mathbf{y}} = \dot{f}(\mathbf{x}, \dot{\mathbf{x}})$ as $\dot{\mathbf{y}} = f'\dot{\mathbf{x}}$. The $m \times n$ matrix $f' = f'(\mathbf{x})$ is the Jacobian of f with entries defined as

$$f'_{ji} = \frac{\partial y_j}{\partial x_i}$$

for $i = 1, \ldots, n$ and $j = 1, \ldots, m$. This transformation is achieved by applying well-known differentiation rules to the elemental functions

$$\varphi_j \in \{+, -, *, /, \sin, \exp, \ldots\}$$

followed by the application of the chain rule as

$$\dot{v}_j = \sum_{i \prec j} c_{ji} \dot{v}_i \tag{6}$$

[2] From now on we will use f to refer to this implementation as a computer program.

for $j = 1, \ldots, m$ and total derivatives $\dot{v}_k$ of the code list variables v_k, $k = 1-n, \ldots, p+m$. The elemental functions φ_j are assumed to be continuously differentiable in a neighborhood of the current argument. The corresponding local partial derivatives are denoted by

$$c_{ji} = \frac{\partial \varphi_j}{\partial v_i} \quad .$$

As a convenient alternative to source transformation, tools for the automatic differentiation of numerical programs can be implemented by operator overloading. Established representatives are ADOL-C [17], AD01 [24], and FADBAD [3].

3 Augmented LL-Parsing

We present an algorithm that augments scalar assignments described by the context-free grammar

$$G = (N, T, S, P)$$

with non-terminals

$$N = \{ASGN, EXPR, EXPR\_LIST\} \quad ,$$

terminals

$$T = \{'=', \ '(', \ \varphi, \ ')', \ ','\} \quad ,$$

where $\varphi \in \{\varphi_j, j = 1, \ldots, p+1\}$, start symbol

$$S = ASGN \quad ,$$

and production rules

$$P = \left\{ \begin{array}{r} ASGN := var \ \ '=' \ \ EXPR \\ EXPR := var \\ EXPR := \varphi \ '(' \ EXPR\_LIST \ ')' \\ EXPR\_LIST := EXPR \\ EXPR\_LIST := EXPR \ \ ',' \ \ EXPR\_LIST \end{array} \right\} \quad .$$

LL-parsers traverse the abstract syntax tree (ast) in depth-first order, making recursive calls to "matching" routines defined for each non-terminal and terminal symbol. For example, the simple assignment $y = x_1 * x_2$ is parsed as follows:

$ASGN$
- $var(y)$
- $EXPR(*)$
  - $EXPR\_LIST$
    - $EXPR$
      - $var(x_1)$
    - $EXPR\_LIST$
      - $EXPR$
        - $var(x_2)$

We associate post-order actions A, Φ, and V with $ASGN$, $EXPR$, and var, respectively. In order to relate the indexing to the code list in Eqn. (5) we set $\Phi \equiv \Phi(j)$ and $V \equiv V(j)$ in the following definition.

A : **CALL** $\Phi(p+1)$
EMIT($\dot{y} = \dot{v}_{p+1}$)
EMIT($y = v_{p+1}$)

$\Phi \equiv \Phi(j)$: **CALL** $\Phi(i)$, $\forall i \prec j$
$P :=$ POP($|\{i : i \prec j\}|$)
if ($P = \emptyset$) **then**
  CALL $V(v_j = x_k)$
else
  EMIT($\dot{v}_j = \sum_{k \in P} c_{jk} \cdot \dot{v}_k$)
  EMIT($v_j = \varphi_j(v_k)_{k \in P}$)
  PUSH(j)
end if

$V \equiv V(j)$: **if** NEW(x) **then**
  EMIT($v_j = x$)
  PUSH(j)
else
  PUSH(ID(x))
end if

Processing an assignment starts with processing its right-hand-side whose value is stored in v_{p+1} as in Eqn. (5). Hence, the required assignments of the value v_{p+1} and its directional derivative $\dot{v}_{p+1}$ to the corresponding left-hand-side variables are generated by calling the EMIT function. Expressions are processed recursively. A stack is used to PUSH and POP the indexes of the respective code list variables. When matching a variable x the boolean function NEW is called to check if x has already been assigned to a code list variable. If so, then ID(x) returns the corresponding index. According to the assumption that $V = V(j)$ is called with the correct code list variable index we have implicitly ID(x) $= j$. However, the emission of the assignment $v_j = x$ is omitted in this case. The algorithm is illustrated with the help of a simple example in Fig. 1.

The augmentation can be performed irrespective of the flow of control. Hence, an existing parser can easily be extended to become a tangent-linear code generator.

4 Implementation

A first prototype has been implemented using ANTLR. The grammar of the prototype is based on the GNU C grammar provided by John Mitchell and Monty Zukowski provided on the website of ANTLR at `www.antlr.org`. The portion of the grammar implementing ANSI C was separated from code for building the AST, so that only the rules for recognizing C input remained. Support for the

[1] A :
[2] . Φ :
[3] . .Φ :
[4] . . . V :
[4] . . . . EMIT($\dot{v}_0 = \dot{x}_2$)
[5] . . . . EMIT($v_0 = x_2$)
[6] . . . . PUSH(0)
[7] . .Φ :
[8] . . . Φ :
[9] . . . . V :
[9] . . . . . EMIT($\dot{v}_{-1} = \dot{x}_1$)
[10] . . . . . EMIT($v_{-1} = x_1$)
[11] . . . . . PUSH(-1)
[12] . . . Φ :
[13] . . . . Φ :
[14] . . . . . V :
[14] . . . . . . PUSH(0)
[15] . . . . $P :=$ POP(1) $= \{0\}$
[16] . . . . EMIT($\dot{v}_1 = -\sin(v_0) * \dot{v}_0$)
[17] . . . . EMIT($v_1 = \cos(v_0)$)
[18] . . . . PUSH(1)
[19] . . . $P :=$ POP(2) $= \{1, -1\}$
[20] . . . EMIT($\dot{v}_2 = v_{-1} * \dot{v}_1 + v_1 * \dot{v}_{-1}$)
[21] . . . EMIT($v_2 = v_{-1} * v_1$)
[22] . . . PUSH(2)
[23] . .$P :=$ POP(2) $= \{2, 0\}$
[24] . .EMIT($\dot{v}_3 = 1/v_2 * \dot{v}_0 - v_0/v_2^2 * \dot{v}_2$)
[25] . .EMIT($v_3 = v_0/v_2$)
[26] . .PUSH(3)
[27] . $P :=$ POP(1) $= \{3\}$
[28] . EMIT($\dot{y} = \dot{v}_3$)
[29] . EMIT($y = v_3$)

The following tangent-linear code is generated:

$\dot{v}_0 = \dot{x}_2$
$v_0 = x_2$
$\dot{v}_{-1} = \dot{x}_1$
$v_{-1} = x_1$
$\dot{v}_1 = -\sin(v_0) * \dot{v}_0$
$v_1 = \cos(v_0)$
$\dot{v}_2 = v_{-1} * \dot{v}_1 + v_1 * \dot{v}_{-1}$
$v_2 = v_{-1} * v_1$
$\dot{v}_3 = 1/v_2 * \dot{v}_0 - v_0/v_2^2 * \dot{v}_2$
$v_3 = v_0/v_2$
$\dot{y} = \dot{v}_3$
$y = v_3$

Fig. 1. Example: $y = x_2/(x_1 * \cos(x_2))$

symbol table, scoping, and unparsing was dropped completely. The implementation language was changed from Java to C++. The full source of the current prototype can be downloaded from `www.stce.rwth-aachen.de/research/antlr`. Below we briefly explain the implementation of the actions associated with assignments, operators or function calls, and variable references.

All grammar rules handling expressions take a flag as argument that selectively activates the generation of tangent linear code. For example, the generation is switched off if indexes of arrays are computed, or when parsing the conditions in loops or in branch statements. A pair of identifiers is returned consisting of the names of the variables that the result of the last computation and the value of the corresponding total derivative are assigned to, respectively.

Assignments are implemented as described in Sec. 3. Derivative code is generated only if the type of the variable on the left-hand side is `float` or `double`.. The prototype stores the association of variables and their types in a simple map that is far from a full featured symbol-table.

The right-hand side of the assignment is known to return a pair of names of temporary variables. The output generated is the assignment of the temporary variable to the variable on the left-hand side. The C-compiler's copy propagation algorithm [2, 20] is set in charge of removing these obsolete assignments.

Multiplications and other operators and calls to intrinsic functions are treated by distinct grammar rules preserving priority. After such a rule is matched completely, two new temporary variables are allocated. The declaration of these variables is written to a second stream which is merged to the output stream when the current statement list is closed, i.e. when the associated closing curly bracket is parsed. The expressions to compute the derivative is usually written to the output stream first followed by the expression to compute the original expression. The order of the derivative and original computation is arbitrary. Only for the division operator and the exponential the order is exchanged to take advantage of the computed original result in the derivative expression. An illustration of these statements can be found in Sec. A.

Variables are looked up in the map of variables. A derivative variable is generated and its declaration is emitted depending on their type.

5 Case Studies

In this section we present two tangent-linear codes that were generated by AD_C_ANTLR. First we consider a very simple example that is similar to the one used in Sec. 3. The original input code is given as follows.

```
void f(double *x) {
  x[2]= x[1]/(x[0] * exp(x[1]));
}
```

It implements a scalar function $f : I\!R^2 \rightarrow I\!R^1$. AD_C_ANTLR produces the following tangent-linear code.

```
void f(double * x, double * g_x){
   double v_00;
   double g_v_00;
```

```
   double v_01;
   double g_v_01;
   double v_02;
   double g_v_02;
   v_00= exp(x[1]);
   g_v_00= g_x[1]* exp(x[1]);
   g_v_01= g_x[0]* v_00+ x[0]* g_v_00;
   v_01= x[0]* v_00;
   v_02= x[1]/ v_01;
   g_v_02= (g_x[1]- v_02* g_v_01)/ v_01;
   g_x[2]= g_v_02;
   x[2]= v_02;
   ;
}
```

The gradient $f' \in I\!R^2$ can be computed by two executions of the tangent-linear code initializing `g_x[0]=1`, `g_x[1]=0` and `g_x[0]=0`, `g_x[1]=1`, respectively. These initializations are performed within the following driver routine.

```
#include <stdio.h>

void f(double * x, double * g_x);

int main(int argc, char **argv) {

  double x[]={2.0 , 4.0, 0.0};
  double g_x[]={0,0,0};
  int j;

  for(j=0;j<2; ++j) {
    g_x[j]=1;
    f(x, g_x);
    printf("%f\n", g_x[2]);
    g_x[j]=0;
  }

  return 0;
}
```

Running the corresponding executable results in the following output

```
-0.018316
-0.027473
```

that is exactly equal to the gradient of `x[2]` with respect to `x[0]` and `x[1]` at the point $(2.0, 4.0)$.

Our second example code implements the discretized differential equation describing an explosion. Both the original source code for the corresponding vector function $F : I\!R^7 \rightarrow I\!R^7$ and parts of the tangent-linear code are shown in the appendix. We are primarily interested in the derivative of the output $\mathtt{f} \in I\!R^7$ with respect to the input $\mathtt{x} \in I\!R^7$. The nonzero entries of the tridiagonal 7×7 Jacobian can be computed by three evaluations of the tangent-linear code using Curtis-Powell-Reid seeding [11]. We use the following seed matrix

$$S^T = \begin{pmatrix} 1\,0\,0\,1\,0\,0\,1 \\ 0\,1\,0\,0\,1\,0\,0 \\ 0\,0\,1\,0\,0\,1\,0 \end{pmatrix}$$

to obtain the compressed Jacobian $(F')^T$

$$\begin{pmatrix} -1.89 & 1.01 & 1.01 & -1.87 & 1.01 & 1.01 & -1.89 \\ 1.01 & -1.87 & 1.01 & 1.01 & -1.87 & 1.01 & 0.00 \\ 0.00 & 1.01 & -1.87 & 1.01 & 1.01 & -1.87 & 1.01 \end{pmatrix} .$$

Decompression using a simple substitution algorithm delivers the Jacobian matrix F'

$$\begin{pmatrix} -1.89 & 1.01 & 0.00 & 0.00 & 0.00 & 0.00 & 0.00 \\ 1.01 & -1.87 & 1.01 & 0.00 & 0.00 & 0.00 & 0.00 \\ 0.00 & 1.01 & -1.87 & 1.01 & 0.00 & 0.00 & 0.00 \\ 0.00 & 0.00 & 1.01 & -1.87 & 1.01 & 0.00 & 0.00 \\ 0.00 & 0.00 & 0.00 & 1.01 & -1.87 & 1.01 & 0.00 \\ 0.00 & 0.00 & 0.00 & 0.00 & 1.01 & -1.87 & 1.01 \\ 0.00 & 0.00 & 0.00 & 0.00 & 0.00 & 1.01 & -1.89 \end{pmatrix}$$

that we were looking for. Details can be found in Sec. A.

6 Conclusion and Outlook

An approach to the automatic generation of tangent-linear code was presented that does not rely on the construction of an intermediate representation of the original program. Special post-order traversal actions were associated with an LL(k)-parser generated by ANTLR to cover a subset of the ANSI C standard. The simplicity of the implementation is apparent from the freely available source code. Feasibility and correctness of this approach was illustrated by two case studies.

An obvious disadvantage of the single-pass approach is the missing infrastructure for static data-flow analysis [18]. Nevertheless we believe that the parser-based source transformation represents a reasonable trade-off between efficiency and ease of implementation.

We are currently working on the extension of AD_C_ANTLR to the parser-based generation of adjoint code defined by

$$\bar{v}_j = \sum_{k:j \prec k} c_{kj} \bar{v}_k, \quad \text{for } j = p \ldots, 1-n.$$

Adjoint code requires the reversal of the flow of control [21]. The straight-forward approach is to augmented the original code with statements to PUSH values of overwritten variables and indexes of basic blocks [22]. These values are POPed in the adjoint section of the code to achieve the control-flow reversal and to compute the local partial derivatives c_{kj}.

References

1. J. Ullman A. Aho. *The Theory of Parsing, Translation, and Compiling. Volume 1: Parsing.* Prentice Hall, 1972.
2. J. Ullman A. Aho. *The Theory of Parsing, Translation, and Compiling. Volume 2: Compiling.* Prentice Hall, 1972.
3. C. Bendtsen and O. Stauning. FADBAD, a flexible C++ package for automatic differentiation. Technical Report IMM–REP–1996–17, Department of Mathematical Modelling, Technical University of Denmark, Lyngby, Denmark, August 1996.
4. M. Berz, C. Bischof, G. Corliss, and A. Griewank, editors. *Computational Differentiation: Techniques, Applications, and Tools*, Proceedings Series, Philadelphia, 1996. SIAM.
5. C. Bischof, M. Bücker, B. Lang, A. Rasch, and A. Vehreschild. Combining source transformation and operator overloading techniques to compute derivatives for MATLAB programs. In *Proceedings of the Second IEEE International Workshop on Source Code Analysis and Manipulation (SCAM 2002)*, pages 65–72, Los Alamitos, CA, USA, 2002. IEEE Computer Society.
6. C. Bischof, A. Carle, P. Khademi, and A. Maurer. The ADIFOR 2.0 system for Automatic Differentiation of Fortran 77 programs. *IEEE Comp. Sci. & Eng.*, 3(3):18–32, 1996.

7. M. Bücker, G. Corliss, P. Hovland, U. Naumann, and B. Norris, editors. *Automatic Differentiation: Applications, Theory, and Tools*, Lecture Notes in Computational Science and Engineering, Berlin, 2005. Springer. To appear.
8. A. Carle and M. Fagan. ADIFOR 3.0 overview. Technical Report CAAM-TR-00-02, Rice University, 2000.
9. G. Corliss, C. Faure, A. Griewank, L. Hascoet, and U. Naumann, editors. *Automatic Differentiation of Algorithms – From Simulation to Optimization*, New York, 2002. Springer.
10. G. Corliss and A. Griewank, editors. *Automatic Differentiation: Theory, Implementation, and Application*, Proceedings Series, Philadelphia, 1991. SIAM.
11. A. Curtis, M. Powell, and J. Reid. On the estimation of sparse Jacobian matrices. *J. Inst. Math. Appl.*, 13:117–119, 1974.
12. C. Donnelly and R. Stallman. *The Bison Manual*. Free Software Foundation, 2003.
13. C. Faure and Y. Papegay. Odyssée User's Guide. Version 1.7. Rapport technique RT–0224, INRIA, Sophia-Antipolis, France, September 1998.
14. R. Giering and T. Kaminski. Recipes for Adjoint Code Construction. *ACM Trans. Math. Software*, 24(4):437–474, 1998.
15. R. Giering and T. Kaminski. Applying TAF to generate efficient derivative code of Fortran 77-95 programs. In *Proceedings of GAMM 2002, Augsburg, Germany*, 2002.
16. A. Griewank. *Evaluating Derivatives. Principles and Techniques of Algorithmic Differentiation*. Number 19 in Frontiers in Applied Mathematics. SIAM, Philadelphia, 2000.
17. A. Griewank, D. Juedes, and J. Utke. ADOL–C, a package for the automatic differentiation of algorithms written in C/C++. *ACM Trans. Math. Software*, 22(2):131–167, 1996.
18. L. Hascoët, U. Naumann, and V. Pascual. TBR analysis in reverse-mode Automatic Differentiation. In M. Bücker, editor, *Future Generation Computer Systems – Special Issue on Automatic Differentiation*. Elsevier, 2004. To appear.
19. L. Hascoët and V. Pascual. Tapenade 2.1 user's guide. Technical report 300, INRIA, 2004.
20. S. Muchnick. *Advanced Compiler Design and Implementation*. Morgan Kaufmann Publishers, San Francisco, 1997.
21. U. Naumann, J. Utke, A. Lyons, and M. Fagan. Control flow reversal for adjoint code generation. In *Proceedings of the Fourth IEEE International Workshop on Source Code Analysis and Manipulation (SCAM 2004)*, pages 55–64, Los Alamitos, CA, USA, 2004. IEEE Computer Society.
22. U. Naumann, J. Utke, A. Lyons, and M. Fagan. Control flow reversal for adjoint code generation. *Science of Computer Programming*, 2005. Invited extended version of the SCAM 2004 paper. Under review.
23. T. Parr and R. Quong. Antlr: A predicated-ll(k)-parser generator. *Software Practice and Experience*, 7(25):789–810, 1995.
24. J. Pryce and J. Reid. ADO1, a Fortran 90 code for automatic differentiation. Technical Report RAL-TR-1998-057, Rutherford Appleton Laboratory, Chilton, Didcot, Oxfordshire, OX11 OQX, England, 1998.
25. L. Rall. *Automatic Differentiation: Techniques and Applications*, volume 120 of *LNCS*. Springer, Berlin, 1981.
26. C. Wunsch, C. Hill, P. Heimbach, U. Naumann, J. Utke, M. Fagan, and N. Tallent. Openad. Preprint ANL/MCS-P1230-0205, Argonne National Laboratory, Argonne, IL, February 2005.

A Explosion Example

A.1 Original Code

```
void ex(int n, double *x, double *p, double *f) {
  int i,v;
  double h;

  h = 2.0/ (n + 1);
  f[0] = -(2*x[0]) + h*h*p[0]/12*
         (1 + 10*(exp(x[0]/(1 + p[1]*x[0]))));
  f[1] = x[0] + h*h*p[0]/12*
         (exp(x[0]/(1 + p[1]*x[0])));
```

```
  v= n-1;
  for(i=1;i<v;++i) {
    f[i-1 ] = f[i - 1] + x[i] + h*h*p[0]/12*
         (exp(x[i]/(1 + p[1]*x[i])));
    f[i] = f[i] - 2*x[i] + h*h*p[0]/1.2*
         (exp(x[i]/(1+ p[1]*x[i])));
    f[i+1] = x[i] + h*h*p[0]/12*
         (exp(x[i]/(1 + p[1]*x[i])));
  }
  f[n-2] = f[n-2] + x[n-1] + h*h*p[0]/12*
         (exp(x[n-1]/(1 + p[1]*x[n-1])));
  f[n-1] = f[n-1] - 2*x[n-1];
  f[n-1] = f[n-1] + h*h*p[0]/12*(1 + 10*
         (exp(x[n-1]/(1 + p[1]*x[n-1]))));
}
```

A.2 Tangent-Linear Code

```
void ex(int n, double * x, double * g_x,
               double * p, double * g_p,
               double * f, double * g_f) {
   double g_h;
   double v_00;
   double g_v_00;
   double v_01;
   double g_v_01;
   ...
   double v_92;
   double g_v_92;

   int i,v;
   double h;

   g_v_00=0;
   v_00= n+1;
   v_01= 2.0/v_00;
   g_v_01= (0.0F- v_01* g_v_00)/v_00;
   g_h= g_v_01;
   h= v_01;
   // The total derivative g_h of h is zero
   // since h is passive. Static activity
   // is required to avoid the generation
   // of this obsolete code section

   g_v_02= 0*x[0]+ 2* g_x[0];
   v_02= 2* x[0];
   // The partial derivative wrt. a constant is
   // zero.  Hence, the generation of 0*x[0]
   // could be avoided. Code optimizations of
   // this type are the subject of ongoing work.

   ...

   g_v_07= g_p[1]* x[0]+ p[1]* g_x[0];
   v_07= p[1]* x[0];
   // A typical example for the product rule.

   ...
```

```
    v_09= x[0]/ v_08;
    g_v_09= (g_x[0]- v_09* g_v_08)/ v_08;
    // The result of a division can be used
    // to compute the partial derivatives more
    // efficiently. Hence, the computation of
    // the intermediate function value precedes
    // the computation of the derivative in this
    // case. A similar statement applies to the
    // exp intrinsic ...
    v_10= exp(v_09);
    g_v_10= g_v_09* v_10;

    ...

    g_v_14= g_v_03+ g_v_13;
    v_14= v_03+ v_13;
    // A typical addition followed by
    // a typical assignment ...
    g_f[0]= g_v_14;
    f[0]= v_14;

    // The flow of control remains unchanged.
    for(i= 1;i<v;++i) {
       double v_25;
       ...

       double v_57;
       double g_v_57;

       ...

       g_v_28= g_h* h+ h* g_h;
       v_28= h* h;
       // Algebraic simplifications such as
       // h*h=h**2 are beyond the scope of
       // the current prototype.
    }

    ...

    g_f[(int)v_78]= g_v_92;
    f[(int)v_78]= v_92;
    // Currently, all temporary variables
    // have type double. Hence, a explicit
    // cast to integer is required when using
    // temporary variables inside of array index
    // expressions.
}
```

A.3 Driver Routine

```
#include <stdio.h>

void ex(int dim, double * x, double * g_x,
                 double * p, double * g_p,
                 double * f, double * g_f);

int main(int argc, char **argv) {
```

```
   double x[]={1.72, 3.45, 4.16, 4.87,
               4.16, 3.45, 1.72};

   // seed matrix
   double g_x_col1[]={1,0,0,1,0,0,1};
   double g_x_col2[]={0,1,0,0,1,0,0};
   double g_x_col3[]={0,0,1,0,0,1,0};

   double f[7];
   double g_f[7];

   double p[]={1.3, 0.245828};
   double g_p[]={0.0,0.0};
   int i;

   ex(7, x, g_x_col1, p, g_p, f, g_f);
   for(i=0; i<7; ++i)
      printf("%f ", g_f[i]);
   printf("\n");

   ex(7, x, g_x_col2, p, g_p, f, g_f);
   for(i=0; i<7; ++i)
      printf("%f ", g_f[i]);
   printf("\n");

   ex(7, x, g_x_col3, p, g_p, f, g_f);
   for(i=0; i<7; ++i)
      printf("%f ", g_f[i]);
   printf("\n");

  return 0;
}
```

A.4 Screen Dump

```
-1.887 1.011 1.012 -1.871 1.012 1.011 -1.887
1.012 -1.871 1.012 1.012 -1.870 1.012 0.000
0.000 1.012 -1.870 1.012 1.012 -1.871 1.012
```

We have decreased the precision of the displayed floating-point values to make them fit on the page.

Aachener Informatik-Berichte

This is a list of recent technical reports. To obtain copies of technical reports please consult http://aib.informatik.rwth-aachen.de/ or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

1987-01 *	Fachgruppe Informatik: Jahresbericht 1986
1987-02 *	David de Frutos Escrig, Klaus Indermark: Equivalence Relations of Non-Deterministic Ianov-Schemes
1987-03 *	Manfred Nagl: A Software Development Environment based on Graph Technology
1987-04 *	Claus Lewerentz, Manfred Nagl, Bernhard Westfechtel: On Integration Mechanisms within a Graph-Based Software Development Environment
1987-05 *	Reinhard Rinn: Über Eingabeanomalien bei verschiedenen Inferenzmodellen
1987-06 *	Werner Damm, Gert Döhmen: Specifying Distributed Computer Architectures in AADL*
1987-07 *	Gregor Engels, Claus Lewerentz, Wilhelm Schäfer: Graph Grammar Engineering: A Software Specification Method
1987-08 *	Manfred Nagl: Set Theoretic Approaches to Graph Grammars
1987-09 *	Claus Lewerentz, Andreas Schürr: Experiences with a Database System for Software Documents
1987-10 *	Herbert Klaeren, Klaus Indermark: A New Implementation Technique for Recursive Function Definitions
1987-11 *	Rita Loogen: Design of a Parallel Programmable Graph Reduction Machine with Distributed Memory
1987-12	J. Börstler, U. Möncke, R. Wilhelm: Table compression for tree automata
1988-01 *	Gabriele Esser, Johannes Rückert, Frank Wagner: Gesellschaftliche Aspekte der Informatik
1988-02 *	Peter Martini, Otto Spaniol: Token-Passing in High-Speed Backbone Networks for Campus-Wide Environments
1988-03 *	Thomas Welzel: Simulation of a Multiple Token Ring Backbone
1988-04 *	Peter Martini: Performance Comparison for HSLAN Media Access Protocols
1988-05 *	Peter Martini: Performance Analysis of Multiple Token Rings
1988-06 *	Andreas Mann, Johannes Rückert, Otto Spaniol: Datenfunknetze
1988-07 *	Andreas Mann, Johannes Rückert: Packet Radio Networks for Data Exchange
1988-08 *	Andreas Mann, Johannes Rückert: Concurrent Slot Assignment Protocol for Packet Radio Networks
1988-09 *	W. Kremer, F. Reichert, J. Rückert, A. Mann: Entwurf einer Netzwerktopologie für ein Mobilfunknetz zur Unterstützung des öffentlichen Straßenverkehrs
1988-10 *	Kai Jakobs: Towards User-Friendly Networking
1988-11 *	Kai Jakobs: The Directory - Evolution of a Standard
1988-12 *	Kai Jakobs: Directory Services in Distributed Systems - A Survey
1988-13 *	Martine Schümmer: RS-511, a Protocol for the Plant Floor

1988-14 * U. Quernheim: Satellite Communication Protocols - A Performance Comparison Considering On-Board Processing

1988-15 * Peter Martini, Otto Spaniol, Thomas Welzel: File Transfer in High Speed Token Ring Networks: Performance Evaluation by Approximate Analysis and Simulation

1988-16 * Fachgruppe Informatik: Jahresbericht 1987

1988-17 * Wolfgang Thomas: Automata on Infinite Objects

1988-18 * Michael Sonnenschein: On Petri Nets and Data Flow Graphs

1988-19 * Heiko Vogler: Functional Distribution of the Contextual Analysis in Block-Structured Programming Languages: A Case Study of Tree Transducers

1988-20 * Thomas Welzel: Einsatz des Simulationswerkzeuges QNAP2 zur Leistungsbewertung von Kommunikationsprotokollen

1988-21 * Th. Janning, C. Lewerentz: Integrated Project Team Management in a Software Development Environment

1988-22 * Joost Engelfriet, Heiko Vogler: Modular Tree Transducers

1988-23 * Wolfgang Thomas: Automata and Quantifier Hierarchies

1988-24 * Uschi Heuter: Generalized Definite Tree Languages

1989-01 * Fachgruppe Informatik: Jahresbericht 1988

1989-02 * G. Esser, J. Rückert, F. Wagner (Hrsg.): Gesellschaftliche Aspekte der Informatik

1989-03 * Heiko Vogler: Bottom-Up Computation of Primitive Recursive Tree Functions

1989-04 * Andy Schürr: Introduction to PROGRESS, an Attribute Graph Grammar Based Specification Language

1989-05 J. Börstler: Reuse and Software Development - Problems, Solutions, and Bibliography (in German)

1989-06 * Kai Jakobs: OSI - An Appropriate Basis for Group Communication?

1989-07 * Kai Jakobs: ISO's Directory Proposal - Evolution, Current Status and Future Problems

1989-08 * Bernhard Westfechtel: Extension of a Graph Storage for Software Documents with Primitives for Undo/Redo and Revision Control

1989-09 * Peter Martini: High Speed Local Area Networks - A Tutorial

1989-10 * P. Davids, Th. Welzel: Performance Analysis of DQDB Based on Simulation

1989-11 * Manfred Nagl (Ed.): Abstracts of Talks presented at the WG '89 15th International Workshop on Graphtheoretic Concepts in Computer Science

1989-12 * Peter Martini: The DQDB Protocol - Is it Playing the Game?

1989-13 * Martine Schümmer: CNC/DNC Communication with MAP

1989-14 * Martine Schümmer: Local Area Networks for Manufactoring Environments with hard Real-Time Requirements

1989-15 * M. Schümmer, Th. Welzel, P. Martini: Integration of Field Bus and MAP Networks - Hierarchical Communication Systems in Production Environments

1989-16 * G. Vossen, K.-U. Witt: SUXESS: Towards a Sound Unification of Extensions of the Relational Data Model

1989-17 * J. Derissen, P. Hruschka, M.v.d. Beeck, Th. Janning, M. Nagl: Integrating Structured Analysis and Information Modelling

1989-18 A. Maassen: Programming with Higher Order Functions

1989-19 * Mario Rodriguez-Artalejo, Heiko Vogler: A Narrowing Machine for Syntax Directed BABEL

1989-20 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Graph-based Implementation of a Functional Logic Language

1990-01 * Fachgruppe Informatik: Jahresbericht 1989

1990-02 * Vera Jansen, Andreas Potthoff, Wolfgang Thomas, Udo Wermuth: A Short Guide to the AMORE System (Computing Automata, MOnoids and Regular Expressions)

1990-03 * Jerzy Skurczynski: On Three Hierarchies of Weak SkS Formulas

1990-04 R. Loogen: Stack-based Implementation of Narrowing

1990-05 H. Kuchen, A. Wagener: Comparison of Dynamic Load Balancing Strategies

1990-06 * Kai Jakobs, Frank Reichert: Directory Services for Mobile Communication

1990-07 * Kai Jakobs: What's Beyond the Interface - OSI Networks to Support Cooperative Work

1990-08 * Kai Jakobs: Directory Names and Schema - An Evaluation

1990-09 * Ulrich Quernheim, Dieter Kreuer: Das CCITT - Signalisierungssystem Nr. 7 auf Satellitenstrecken; Simulation der Zeichengabestrecke

1990-11 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Lazy Narrowing in a Graph Machine

1990-12 * Kai Jakobs, Josef Kaltwasser, Frank Reichert, Otto Spaniol: Der Computer fährt mit

1990-13 * Rudolf Mathar, Andreas Mann: Analyzing a Distributed Slot Assignment Protocol by Markov Chains

1990-14 A. Maassen: Compilerentwicklung in Miranda - ein Praktikum in funktionaler Programmierung (written in german)

1990-15 * Manfred Nagl, Andreas Schürr: A Specification Environment for Graph Grammars

1990-16 A. Schürr: PROGRESS: A VHL-Language Based on Graph Grammars

1990-17 * Marita Möller: Ein Ebenenmodell wissensbasierter Konsultationen - Unterstützung für Wissensakquisition und Erklärungsfähigkeit

1990-18 * Eric Kowalewski: Entwurf und Interpretation einer Sprache zur Beschreibung von Konsultationsphasen in Expertensystemen

1990-20 Y. Ortega Mallen, D. de Frutos Escrig: A Complete Proof System for Timed Observations

1990-21 * Manfred Nagl: Modelling of Software Architectures: Importance, Notions, Experiences

1990-22 H. Fassbender, H. Vogler: A Call-by-need Implementation of Syntax Directed Functional Programming

1991-01 Guenther Geiler (ed.), Fachgruppe Informatik: Jahresbericht 1990

1991-03 B. Steffen, A. Ingolfsdottir: Characteristic Formulae for Processes with Divergence

1991-04 M. Portz: A new class of cryptosystems based on interconnection networks

1991-05 H. Kuchen, G. Geiler: Distributed Applicative Arrays
1991-06 * Ludwig Staiger: Kolmogorov Complexity and Hausdorff Dimension
1991-07 * Ludwig Staiger: Syntactic Congruences for w-languages
1991-09 * Eila Kuikka: A Proposal for a Syntax-Directed Text Processing System
1991-10 K. Gladitz, H. Fassbender, H. Vogler: Compiler-based Implementation of Syntax-Directed Functional Programming
1991-11 R. Loogen, St. Winkler: Dynamic Detection of Determinism in Functional Logic Languages
1991-12 * K. Indermark, M. Rodriguez Artalejo (Eds.): Granada Workshop on the Integration of Functional and Logic Programming
1991-13 * Rolf Hager, Wolfgang Kremer: The Adaptive Priority Scheduler: A More Fair Priority Service Discipline
1991-14 * Andreas Fasbender, Wolfgang Kremer: A New Approximation Algorithm for Tandem Networks with Priority Nodes
1991-15 J. Börstler, A. Zündorf: Revisiting extensions to Modula-2 to support reusability
1991-16 J. Börstler, Th. Janning: Bridging the gap between Requirements Analysis and Design
1991-17 A. Zündorf, A. Schürr: Nondeterministic Control Structures for Graph Rewriting Systems
1991-18 * Matthias Jarke, John Mylopoulos, Joachim W. Schmidt, Yannis Vassiliou: DAIDA: An Environment for Evolving Information Systems
1991-19 M. Jeusfeld, M. Jarke: From Relational to Object-Oriented Integrity Simplification
1991-20 G. Hogen, A. Kindler, R. Loogen: Automatic Parallelization of Lazy Functional Programs
1991-21 * Prof. Dr. rer. nat. Otto Spaniol: ODP (Open Distributed Processing): Yet another Viewpoint
1991-22 H. Kuchen, F. Lücking, H. Stoltze: The Topology Description Language TDL
1991-23 S. Graf, B. Steffen: Compositional Minimization of Finite State Systems
1991-24 R. Cleaveland, J. Parrow, B. Steffen: The Concurrency Workbench: A Semantics Based Tool for the Verification of Concurrent Systems
1991-25 * Rudolf Mathar, Jürgen Mattfeldt: Optimal Transmission Ranges for Mobile Communication in Linear Multihop Packet Radio Networks
1991-26 M. Jeusfeld, M. Staudt: Query Optimization in Deductive Object Bases
1991-27 J. Knoop, B. Steffen: The Interprocedural Coincidence Theorem
1991-28 J. Knoop, B. Steffen: Unifying Strength Reduction and Semantic Code Motion
1991-30 T. Margaria: First-Order theories for the verification of complex FSMs
1991-31 B. Steffen: Generating Data Flow Analysis Algorithms from Modal Specifications
1992-01 Stefan Eherer (ed.), Fachgruppe Informatik: Jahresbericht 1991
1992-02 * Bernhard Westfechtel: Basismechanismen zur Datenverwaltung in strukturbezogenen Hypertextsystemen
1992-04 S. A. Smolka, B. Steffen: Priority as Extremal Probability
1992-05 * Matthias Jarke, Carlos Maltzahn, Thomas Rose: Sharing Processes: Team Coordination in Design Repositories

1992-06 O. Burkart, B. Steffen: Model Checking for Context-Free Processes
1992-07 * Matthias Jarke, Klaus Pohl: Information Systems Quality and Quality Information Systems
1992-08 * Rudolf Mathar, Jürgen Mattfeldt: Analyzing Routing Strategy NFP in Multihop Packet Radio Networks on a Line
1992-09 * Alfons Kemper, Guido Moerkotte: Grundlagen objektorientierter Datenbanksysteme
1992-10 Matthias Jarke, Manfred Jeusfeld, Andreas Miethsam, Michael Gocek: Towards a logic-based reconstruction of software configuration management
1992-11 Werner Hans: A Complete Indexing Scheme for WAM-based Abstract Machines
1992-12 W. Hans, R. Loogen, St. Winkler: On the Interaction of Lazy Evaluation and Backtracking
1992-13 * Matthias Jarke, Thomas Rose: Specification Management with CAD
1992-14 Th. Noll, H. Vogler: Top-down Parsing with Simultaneous Evaluation on Noncircular Attribute Grammars
1992-15 A. Schuerr, B. Westfechtel: Graphgrammatiken und Graphersetzungssysteme(written in german)
1992-16 * Graduiertenkolleg Informatik und Technik (Hrsg.): Forschungsprojekte des Graduiertenkollegs Informatik und Technik
1992-17 M. Jarke (ed.): ConceptBase V3.1 User Manual
1992-18 * Clarence A. Ellis, Matthias Jarke (Eds.): Distributed Cooperation in Integrated Information Systems - Proceedings of the Third International Workshop on Intelligent and Cooperative Information Systems
1992-19-00 H. Kuchen, R. Loogen (eds.): Proceedings of the 4th Int. Workshop on the Parallel Implementation of Functional Languages
1992-19-01 G. Hogen, R. Loogen: PASTEL - A Parallel Stack-Based Implementation of Eager Functional Programs with Lazy Data Structures (Extended Abstract)
1992-19-02 H. Kuchen, K. Gladitz: Implementing Bags on a Shared Memory MIMD-Machine
1992-19-03 C. Rathsack, S.B. Scholz: LISA - A Lazy Interpreter for a Full-Fledged Lambda-Calculus
1992-19-04 T.A. Bratvold: Determining Useful Parallelism in Higher Order Functions
1992-19-05 S. Kahrs: Polymorphic Type Checking by Interpretation of Code
1992-19-06 M. Chakravarty, M. Köhler: Equational Constraints, Residuation, and the Parallel JUMP-Machine
1992-19-07 J. Seward: Polymorphic Strictness Analysis using Frontiers (Draft Version)
1992-19-08 D. Gärtner, A. Kimms, W. Kluge: pi-Redˆ+ - A Compiling Graph-Reduction System for a Full Fledged Lambda-Calculus
1992-19-09 D. Howe, G. Burn: Experiments with strict STG code
1992-19-10 J. Glauert: Parallel Implementation of Functional Languages Using Small Processes
1992-19-11 M. Joy, T. Axford: A Parallel Graph Reduction Machine
1992-19-12 A. Bennett, P. Kelly: Simulation of Multicache Parallel Reduction

1992-19-13 K. Langendoen, D.J. Agterkamp: Cache Behaviour of Lazy Functional Programs (Working Paper)

1992-19-14 K. Hammond, S. Peyton Jones: Profiling scheduling strategies on the GRIP parallel reducer

1992-19-15 S. Mintchev: Using Strictness Information in the STG-machine

1992-19-16 D. Rushall: An Attribute Grammar Evaluator in Haskell

1992-19-17 J. Wild, H. Glaser, P. Hartel: Statistics on storage management in a lazy functional language implementation

1992-19-18 W.S. Martins: Parallel Implementations of Functional Languages

1992-19-19 D. Lester: Distributed Garbage Collection of Cyclic Structures (Draft version)

1992-19-20 J.C. Glas, R.F.H. Hofman, W.G. Vree: Parallelization of Branch-and-Bound Algorithms in a Functional Programming Environment

1992-19-21 S. Hwang, D. Rushall: The nu-STG machine: a parallelized Spineless Tagless Graph Reduction Machine in a distributed memory architecture (Draft version)

1992-19-22 G. Burn, D. Le Metayer: Cps-Translation and the Correctness of Optimising Compilers

1992-19-23 S.L. Peyton Jones, P. Wadler: Imperative functional programming (Brief summary)

1992-19-24 W. Damm, F. Liu, Th. Peikenkamp: Evaluation and Parallelization of Functions in Functional + Logic Languages (abstract)

1992-19-25 M. Kesseler: Communication Issues Regarding Parallel Functional Graph Rewriting

1992-19-26 Th. Peikenkamp: Charakterizing and representing neededness in functional loginc languages (abstract)

1992-19-27 H. Doerr: Monitoring with Graph-Grammars as formal operational Models

1992-19-28 J. van Groningen: Some implementation aspects of Concurrent Clean on distributed memory architectures

1992-19-29 G. Ostheimer: Load Bounding for Implicit Parallelism (abstract)

1992-20 H. Kuchen, F.J. Lopez Fraguas, J.J. Moreno Navarro, M. Rodriguez Artalejo: Implementing Disequality in a Lazy Functional Logic Language

1992-21 H. Kuchen, F.J. Lopez Fraguas: Result Directed Computing in a Functional Logic Language

1992-22 H. Kuchen, J.J. Moreno Navarro, M.V. Hermenegildo: Independent AND-Parallel Narrowing

1992-23 T. Margaria, B. Steffen: Distinguishing Formulas for Free

1992-24 K. Pohl: The Three Dimensions of Requirements Engineering

1992-25 * R. Stainov: A Dynamic Configuration Facility for Multimedia Communications

1992-26 * Michael von der Beeck: Integration of Structured Analysis and Timed Statecharts for Real-Time and Concurrency Specification

1992-27 W. Hans, St. Winkler: Aliasing and Groundness Analysis of Logic Programs through Abstract Interpretation and its Safety

1992-28 * Gerhard Steinke, Matthias Jarke: Support for Security Modeling in Information Systems Design

1992-29 B. Schinzel: Warum Frauenforschung in Naturwissenschaft und Technik

1992-30 A. Kemper, G. Moerkotte, K. Peithner: Object-Orientation Axiomatised by Dynamic Logic

1992-32 * Bernd Heinrichs, Kai Jakobs: Timer Handling in High-Performance Transport Systems

1992-33 * B. Heinrichs, K. Jakobs, K. Lenßen, W. Reinhardt, A. Spinner: Euro-Bridge: Communication Services for Multimedia Applications

1992-34 C. Gerlhof, A. Kemper, Ch. Kilger, G. Moerkotte: Partition-Based Clustering in Object Bases: From Theory to Practice

1992-35 J. Börstler: Feature-Oriented Classification and Reuse in IPSEN

1992-36 M. Jarke, J. Bubenko, C. Rolland, A. Sutcliffe, Y. Vassiliou: Theories Underlying Requirements Engineering: An Overview of NATURE at Genesis

1992-37 * K. Pohl, M. Jarke: Quality Information Systems: Repository Support for Evolving Process Models

1992-38 A. Zuendorf: Implementation of the imperative / rule based language PROGRES

1992-39 P. Koch: Intelligentes Backtracking bei der Auswertung funktional-logischer Programme

1992-40 * Rudolf Mathar, Jürgen Mattfeldt: Channel Assignment in Cellular Radio Networks

1992-41 * Gerhard Friedrich, Wolfgang Neidl: Constructive Utility in Model-Based Diagnosis Repair Systems

1992-42 * P. S. Chen, R. Hennicker, M. Jarke: On the Retrieval of Reusable Software Components

1992-43 W. Hans, St.Winkler: Abstract Interpretation of Functional Logic Languages

1992-44 N. Kiesel, A. Schuerr, B. Westfechtel: Design and Evaluation of GRAS, a Graph-Oriented Database System for Engineering Applications

1993-01 * Fachgruppe Informatik: Jahresbericht 1992

1993-02 * Patrick Shicheng Chen: On Inference Rules of Logic-Based Information Retrieval Systems

1993-03 G. Hogen, R. Loogen: A New Stack Technique for the Management of Runtime Structures in Distributed Environments

1993-05 A. Zündorf: A Heuristic for the Subgraph Isomorphism Problem in Executing PROGRES

1993-06 A. Kemper, D. Kossmann: Adaptable Pointer Swizzling Strategies in Object Bases: Design, Realization, and Quantitative Analysis

1993-07 * Graduiertenkolleg Informatik und Technik (Hrsg.): Graduiertenkolleg Informatik und Technik

1993-08 * Matthias Berger: k-Coloring Vertices using a Neural Network with Convergence to Valid Solutions

1993-09 M. Buchheit, M. Jeusfeld, W. Nutt, M. Staudt: Subsumption between Queries to Object-Oriented Databases

1993-10 O. Burkart, B. Steffen: Pushdown Processes: Parallel Composition and Model Checking

1993-11 * R. Große-Wienker, O. Hermanns, D. Menzenbach, A. Pollacks, S. Repetzki, J. Schwartz, K. Sonnenschein, B. Westfechtel: Das SUKITS-Projekt: A-posteriori-Integration heterogener CIM-Anwendungssysteme

1993-12 *	Rudolf Mathar, Jürgen Mattfeldt: On the Distribution of Cumulated Interference Power in Rayleigh Fading Channels
1993-13	O. Maler, L. Staiger: On Syntactic Congruences for omega-languages
1993-14	M. Jarke, St. Eherer, R. Gallersdoerfer, M. Jeusfeld, M. Staudt: ConceptBase - A Deductive Object Base Manager
1993-15	M. Staudt, H.W. Nissen, M.A. Jeusfeld: Query by Class, Rule and Concept
1993-16 *	M. Jarke, K. Pohl, St. Jacobs et al.: Requirements Engineering: An Integrated View of Representation Process and Domain
1993-17 *	M. Jarke, K. Pohl: Establishing Vision in Context: Towards a Model of Requirements Processes
1993-18	W. Hans, H. Kuchen, St. Winkler: Full Indexing for Lazy Narrowing
1993-19	W. Hans, J.J. Ruz, F. Saenz, St. Winkler: A VHDL Specification of a Shared Memory Parallel Machine for Babel
1993-20 *	K. Finke, M. Jarke, P. Szczurko, R. Soltysiak: Quality Management for Expert Systems in Process Control
1993-21	M. Jarke, M.A. Jeusfeld, P. Szczurko: Three Aspects of Intelligent Cooperation in the Quality Cycle
1994-01	Margit Generet, Sven Martin (eds.), Fachgruppe Informatik: Jahresbericht 1993
1994-02	M. Lefering: Development of Incremental Integration Tools Using Formal Specifications
1994-03 *	P. Constantopoulos, M. Jarke, J. Mylopoulos, Y. Vassiliou: The Software Information Base: A Server for Reuse
1994-04 *	Rolf Hager, Rudolf Mathar, Jürgen Mattfeldt: Intelligent Cruise Control and Reliable Communication of Mobile Stations
1994-05 *	Rolf Hager, Peter Hermesmann, Michael Portz: Feasibility of Authentication Procedures within Advanced Transport Telematics
1994-06 *	Claudia Popien, Bernd Meyer, Axel Kuepper: A Formal Approach to Service Import in ODP Trader Federations
1994-07	P. Peters, P. Szczurko: Integrating Models of Quality Management Methods by an Object-Oriented Repository
1994-08 *	Manfred Nagl, Bernhard Westfechtel: A Universal Component for the Administration in Distributed and Integrated Development Environments
1994-09 *	Patrick Horster, Holger Petersen: Signatur- und Authentifikationsverfahren auf der Basis des diskreten Logarithmusproblems
1994-11	A. Schürr: PROGRES, A Visual Language and Environment for PROgramming with Graph REwrite Systems
1994-12	A. Schürr: Specification of Graph Translators with Triple Graph Grammars
1994-13	A. Schürr: Logic Based Programmed Structure Rewriting Systems
1994-14	L. Staiger: Codes, Simplifying Words, and Open Set Condition
1994-15 *	Bernhard Westfechtel: A Graph-Based System for Managing Configurations of Engineering Design Documents
1994-16	P. Klein: Designing Software with Modula-3
1994-17	I. Litovsky, L. Staiger: Finite acceptance of infinite words

1994-18 G. Hogen, R. Loogen: Parallel Functional Implementations: Graphbased vs. Stackbased Reduction

1994-19 M. Jeusfeld, U. Johnen: An Executable Meta Model for Re-Engineering of Database Schemas

1994-20 * R. Gallersdörfer, M. Jarke, K. Klabunde: Intelligent Networks as a Data Intensive Application (INDIA)

1994-21 M. Mohnen: Proving the Correctness of the Static Link Technique Using Evolving Algebras

1994-22 H. Fernau, L. Staiger: Valuations and Unambiguity of Languages, with Applications to Fractal Geometry

1994-24 * M. Jarke, K. Pohl, R. Dömges, St. Jacobs, H. W. Nissen: Requirements Information Management: The NATURE Approach

1994-25 * M. Jarke, K. Pohl, C. Rolland, J.-R. Schmitt: Experience-Based Method Evaluation and Improvement: A Process Modeling Approach

1994-26 * St. Jacobs, St. Kethers: Improving Communication and Decision Making within Quality Function Deployment

1994-27 * M. Jarke, H. W. Nissen, K. Pohl: Tool Integration in Evolving Information Systems Environments

1994-28 O. Burkart, D. Caucal, B. Steffen: An Elementary Bisimulation Decision Procedure for Arbitrary Context-Free Processes

1995-01 * Fachgruppe Informatik: Jahresbericht 1994

1995-02 Andy Schürr, Andreas J. Winter, Albert Zündorf: Graph Grammar Engineering with PROGRES

1995-03 Ludwig Staiger: A Tight Upper Bound on Kolmogorov Complexity by Hausdorff Dimension and Uniformly Optimal Prediction

1995-04 Birgitta König-Ries, Sven Helmer, Guido Moerkotte: An experimental study on the complexity of left-deep join ordering problems for cyclic queries

1995-05 Sophie Cluet, Guido Moerkotte: Efficient Evaluation of Aggregates on Bulk Types

1995-06 Sophie Cluet, Guido Moerkotte: Nested Queries in Object Bases

1995-07 Sophie Cluet, Guido Moerkotte: Query Optimization Techniques Exploiting Class Hierarchies

1995-08 Markus Mohnen: Efficient Compile-Time Garbage Collection for Arbitrary Data Structures

1995-09 Markus Mohnen: Functional Specification of Imperative Programs: An Alternative Point of View of Functional Languages

1995-10 Rainer Gallersdörfer, Matthias Nicola: Improving Performance in Replicated Databases through Relaxed Coherency

1995-11 * M.Staudt, K.von Thadden: Subsumption Checking in Knowledge Bases

1995-12 * G.V.Zemanek, H.W.Nissen, H.Hubert, M.Jarke: Requirements Analysis from Multiple Perspectives: Experiences with Conceptual Modeling Technology

1995-13 * M.Staudt, M.Jarke: Incremental Maintenance of Externally Materialized Views

1995-14 * P.Peters, P.Szczurko, M.Jeusfeld: Oriented Information Management: Conceptual Models at Work

1995-15 * Matthias Jarke, Sudha Ram (Hrsg.): WITS 95 Proceedings of the 5th Annual Workshop on Information Technologies and Systems
1995-16 * W.Hans, St.Winkler, F.Saenz: Distributed Execution in Functional Logic Programming
1996-01 * Jahresbericht 1995
1996-02 Michael Hanus, Christian Prehofer: Higher-Order Narrowing with Definitional Trees
1996-03 * W.Scheufele, G.Moerkotte: Optimal Ordering of Selections and Joins in Acyclic Queries with Expensive Predicates
1996-04 Klaus Pohl: PRO-ART: Enabling Requirements Pre-Traceability
1996-05 Klaus Pohl: Requirements Engineering: An Overview
1996-06 * M.Jarke, W.Marquardt: Design and Evaluation of Computer–Aided Process Modelling Tools
1996-07 Olaf Chitil: The Sigma-Semantics: A Comprehensive Semantics for Functional Programs
1996-08 * S.Sripada: On Entropy and the Limitations of the Second Law of Thermodynamics
1996-09 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP96 - Fifth International Conference on Algebraic and Logic Programming
1996-09-0 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP 96 - Fifth International Conference on Algebraic and Logic Programming: Introduction and table of contents
1996-09-1 Ilies Alouini: An Implementation of Conditional Concurrent Rewriting on Distributed Memory Machines
1996-09-2 Olivier Danvy, Karoline Malmkjær: On the Idempotence of the CPS Transformation
1996-09-3 Víctor M. Gulías, José L. Freire: Concurrent Programming in Haskell
1996-09-4 Sébastien Limet, Pierre Réty: On Decidability of Unifiability Modulo Rewrite Systems
1996-09-5 Alexandre Tessier: Declarative Debugging in Constraint Logic Programming
1996-10 Reidar Conradi, Bernhard Westfechtel: Version Models for Software Configuration Management
1996-11 * C.Weise, D.Lenzkes: A Fast Decision Algorithm for Timed Refinement
1996-12 * R.Dömges, K.Pohl, M.Jarke, B.Lohmann, W.Marquardt: PRO-ART/CE* — An Environment for Managing the Evolution of Chemical Process Simulation Models
1996-13 * K.Pohl, R.Klamma, K.Weidenhaupt, R.Dömges, P.Haumer, M.Jarke: A Framework for Process-Integrated Tools
1996-14 * R.Gallersdörfer, K.Klabunde, A.Stolz, M.Eßmajor: INDIA — Intelligent Networks as a Data Intensive Application, Final Project Report, June 1996
1996-15 * H.Schimpe, M.Staudt: VAREX: An Environment for Validating and Refining Rule Bases
1996-16 * M.Jarke, M.Gebhardt, S.Jacobs, H.Nissen: Conflict Analysis Across Heterogeneous Viewpoints: Formalization and Visualization
1996-17 Manfred A. Jeusfeld, Tung X. Bui: Decision Support Components on the Internet

1996-18 Manfred A. Jeusfeld, Mike Papazoglou: Information Brokering: Design, Search and Transformation

1996-19 * P.Peters, M.Jarke: Simulating the impact of information flows in networked organizations

1996-20 Matthias Jarke, Peter Peters, Manfred A. Jeusfeld: Model-driven planning and design of cooperative information systems

1996-21 * G.de Michelis, E.Dubois, M.Jarke, F.Matthes, J.Mylopoulos, K.Pohl, J.Schmidt, C.Woo, E.Yu: Cooperative information systems: a manifesto

1996-22 * S.Jacobs, M.Gebhardt, S.Kethers, W.Rzasa: Filling HTML forms simultaneously: CoWeb architecture and functionality

1996-23 * M.Gebhardt, S.Jacobs: Conflict Management in Design

1997-01 Michael Hanus, Frank Zartmann (eds.): Jahresbericht 1996

1997-02 Johannes Faassen: Using full parallel Boltzmann Machines for Optimization

1997-03 Andreas Winter, Andy Schürr: Modules and Updatable Graph Views for PROgrammed Graph REwriting Systems

1997-04 Markus Mohnen, Stefan Tobies: Implementing Context Patterns in the Glasgow Haskell Compiler

1997-05 * S.Gruner: Schemakorrespondenzaxiome unterstützen die paargrammatische Spezifikation inkrementeller Integrationswerkzeuge

1997-06 Matthias Nicola, Matthias Jarke: Design and Evaluation of Wireless Health Care Information Systems in Developing Countries

1997-07 Petra Hofstedt: Taskparallele Skelette für irregulär strukturierte Probleme in deklarativen Sprachen

1997-08 Dorothea Blostein, Andy Schürr: Computing with Graphs and Graph Rewriting

1997-09 Carl-Arndt Krapp, Bernhard Westfechtel: Feedback Handling in Dynamic Task Nets

1997-10 Matthias Nicola, Matthias Jarke: Integrating Replication and Communication in Performance Models of Distributed Databases

1997-11 * R. Klamma, P. Peters, M. Jarke: Workflow Support for Failure Management in Federated Organizations

1997-13 Markus Mohnen: Optimising the Memory Management of Higher-Order Functional Programs

1997-14 Roland Baumann: Client/Server Distribution in a Structure-Oriented Database Management System

1997-15 George Botorog: High-Level Parallel Programming and the Efficient Implementation of Numerical Algorithms

1998-01 * Fachgruppe Informatik: Jahresbericht 1997

1998-02 Stefan Gruner, Manfred Nagel, Andy Schürr: Fine-grained and Structure-Oriented Document Integration Tools are Needed for Development Processes

1998-03 Stefan Gruner: Einige Anmerkungen zur graphgrammatischen Spezifikation von Integrationswerkzeugen nach Westfechtel, Janning, Lefering und Schürr

1998-04 * O. Kubitz: Mobile Robots in Dynamic Environments

1998-05 Martin Leucker, Stephan Tobies: Truth - A Verification Platform for Distributed Systems

1998-06 * Matthias Oliver Berger: DECT in the Factory of the Future
1998-07 M. Arnold, M. Erdmann, M. Glinz, P. Haumer, R. Knoll, B. Paech, K. Pohl, J. Ryser, R. Studer, K. Weidenhaupt: Survey on the Scenario Use in Twelve Selected Industrial Projects
1998-09 * Th. Lehmann: Geometrische Ausrichtung medizinischer Bilder am Beispiel intraoraler Radiographien
1998-10 * M. Nicola, M. Jarke: Performance Modeling of Distributed and Replicated Databases
1998-11 * Ansgar Schleicher, Bernhard Westfechtel, Dirk Jäger: Modeling Dynamic Software Processes in UML
1998-12 * W. Appelt, M. Jarke: Interoperable Tools for Cooperation Support using the World Wide Web
1998-13 Klaus Indermark: Semantik rekursiver Funktionsdefinitionen mit Striktheitsinformation
1999-01 * Jahresbericht 1998
1999-02 * F. Huch: Verifcation of Erlang Programs using Abstract Interpretation and Model Checking — Extended Version
1999-03 * R. Gallersdörfer, M. Jarke, M. Nicola: The ADR Replication Manager
1999-04 María Alpuente, Michael Hanus, Salvador Lucas, Germán Vidal: Specialization of Functional Logic Programs Based on Needed Narrowing
1999-05 * W. Thomas (Ed.): DLT 99 - Developments in Language Theory Fourth International Conference
1999-06 * Kai Jakobs, Klaus-Dieter Kleefeld: Informationssysteme für die angewandte historische Geographie
1999-07 Thomas Wilke: CTL+ is exponentially more succinct than CTL
1999-08 Oliver Matz: Dot-Depth and Monadic Quantifier Alternation over Pictures
2000-01 * Jahresbericht 1999
2000-02 Jens Vöge, Marcin Jurdzinski: A Discrete Strategy Improvement Algorithm for Solving Parity Games
2000-03 D. Jäger, A. Schleicher, B. Westfechtel: UPGRADE: A Framework for Building Graph-Based Software Engineering Tools
2000-04 Andreas Becks, Stefan Sklorz, Matthias Jarke: Exploring the Semantic Structure of Technical Document Collections: A Cooperative Systems Approach
2000-05 Mareike Schoop: Cooperative Document Management
2000-06 Mareike Schoop, Christoph Quix (eds.): Proceedings of the Fifth International Workshop on the Language-Action Perspective on Communication Modelling
2000-07 * Markus Mohnen, Pieter Koopman (Eds.): Proceedings of the 12th International Workshop of Functional Languages
2000-08 Thomas Arts, Thomas Noll: Verifying Generic Erlang Client-Server Implementations
2001-01 * Jahresbericht 2000
2001-02 Benedikt Bollig, Martin Leucker: Deciding LTL over Mazurkiewicz Traces
2001-03 Thierry Cachat: The power of one-letter rational languages

2001-04	Benedikt Bollig, Martin Leucker, Michael Weber: Local Parallel Model Checking for the Alternation Free mu-Calculus
2001-05	Benedikt Bollig, Martin Leucker, Thomas Noll: Regular MSC Languages
2001-06	Achim Blumensath: Prefix-Recognisable Graphs and Monadic Second-Order Logic
2001-07	Martin Grohe, Stefan Wöhrle: An Existential Locality Theorem
2001-08	Mareike Schoop, James Taylor (eds.): Proceedings of the Sixth International Workshop on the Language-Action Perspective on Communication Modelling
2001-09	Thomas Arts, Jürgen Giesl: A collection of examples for termination of term rewriting using dependency pairs
2001-10	Achim Blumensath: Axiomatising Tree-interpretable Structures
2001-11	Klaus Indermark, Thomas Noll (eds.): Kolloquium Programmiersprachen und Grundlagen der Programmierung
2002-01 *	Jahresbericht 2001
2002-02	Jürgen Giesl, Aart Middeldorp: Transformation Techniques for Context-Sensitive Rewrite Systems
2002-03	Benedikt Bollig, Martin Leucker, Thomas Noll: Generalised Regular MSC Languages
2002-04	Jürgen Giesl, Aart Middeldorp: Innermost Termination of Context-Sensitive Rewriting
2002-05	Horst Lichter, Thomas von der Maßen, Thomas Weiler: Modelling Requirements and Architectures for Software Product Lines
2002-06	Henry N. Adorna: 3-Party Message Complexity is Better than 2-Party Ones for Proving Lower Bounds on the Size of Minimal Nondeterministic Finite Automata
2002-07	Jörg Dahmen: Invariant Image Object Recognition using Gaussian Mixture Densities
2002-08	Markus Mohnen: An Open Framework for Data-Flow Analysis in Java
2002-09	Markus Mohnen: Interfaces with Default Implementations in Java
2002-10	Martin Leucker: Logics for Mazurkiewicz traces
2002-11	Jürgen Giesl, Hans Zantema: Liveness in Rewriting
2003-01 *	Jahresbericht 2002
2003-02	Jürgen Giesl, René Thiemann: Size-Change Termination for Term Rewriting
2003-03	Jürgen Giesl, Deepak Kapur: Deciding Inductive Validity of Equations
2003-04	Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Improving Dependency Pairs
2003-05	Christof Löding, Philipp Rohde: Solving the Sabotage Game is PSPACE-hard
2003-06	Franz Josef Och: Statistical Machine Translation: From Single-Word Models to Alignment Templates
2003-07	Horst Lichter, Thomas von der Maßen, Alexander Nyßen, Thomas Weiler: Vergleich von Ansätzen zur Feature Modellierung bei der Softwareproduktlinienentwicklung
2003-08	Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Mechanizing Dependency Pairs
2004-01 *	Fachgruppe Informatik: Jahresbericht 2003

2004-02	Benedikt Bollig, Martin Leucker: Message-Passing Automata are expressively equivalent to EMSO logic
2004-03	Delia Kesner, Femke van Raamsdonk, Joe Wells (eds.): HOR 2004 – 2nd International Workshop on Higher-Order Rewriting
2004-04	Slim Abdennadher, Christophe Ringeissen (eds.): RULE 04 – Fifth International Workshop on Rule-Based Programming
2004-05	Herbert Kuchen (ed.): WFLP 04 – 13th International Workshop on Functional and (Constraint) Logic Programming
2004-06	Sergio Antoy, Yoshihito Toyama (eds.): WRS 04 – 4th International Workshop on Reduction Strategies in Rewriting and Programming
2004-07	Michael Codish, Aart Middeldorp (eds.): WST 04 – 7th International Workshop on Termination
2004-08	Klaus Indermark, Thomas Noll: Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information
2004-09	Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Parameterized Power Domination Complexity
2004-10	Zinaida Benenson, Felix C. Gärtner, Dogan Kesdogan: Secure Multi-Party Computation with Security Modules
2005-01 *	Fachgruppe Informatik: Jahresbericht 2004
2005-02	Maximillian Dornseif, Felix C. Gärtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"
2005-03	Jürgen Giesl, René Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions
2005-04	Daniel Mölle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem
2005-05	Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots
2005-06	Simon Fischer, Berthold Vöcking: Adaptive Routing with Stale Information
2005-07	Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks
2005-08	Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut
2005-09	Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures
2005-10	Benedikt Bollig: Automata and Logics for Message Sequence Charts
2005-11	Simon Fischer, Berthold Vöcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture
2005-12	Neeraj Mittal, Felix Freiling, Subbarayan Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Crash-Prone Systems
2005-13	Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments
2005-14	Felix C. Freiling, Sukumar Ghosh: Code Stabilization
2005-15	Uwe Naumann: The Complexity of Derivative Computation

2005-16	Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)
2005-17	Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)
2005-18	Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlußbericht des GI-Arbeitskreises "Features"

* These reports are only available as a printed version.

Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.