RWTH Aachen

Department of Computer Science
Technical Report

Toward Adjoint OpenMP

Michael Förster and Uwe Naumann and Jean Utke

ISSN 0935–3232 · Aachener Informatik-Berichte · AIB-2011-13

RWTH Aachen · Department of Computer Science · July 2011

Toward Adjoint OpenMP

Michael Förster[1] and Uwe Naumann[1] and Jean Utke[2]

1: LuFG Informatik 12 - STCE, RWTH Aachen University, Germany
{foerster, naumann}@stce.rwth-aachen.de

2: MCS Division, Argonne National Laboratory, Argonne, Illinois, USA
utke@mcs.anl.gov

Abstract. Shared-memory multiprocessing is becoming increasingly important in high-performance scientific computing. Algorithmic differentiation provides accurate derivative values and better runtime performance of the adjoint model compared with finite differences. This paper presents a source-to-source transformation of OpenMP augmented code that can be used in source transformation tools for creating the adjoint code of a given input code. Only some directives of the OpenMP standard are considered here, namely, directives to parallelize loops.

1 Introduction

In numerical applications correct and accurate derivative values are essential. These values are computed by finite differences or by algorithmic differentiation (AD) [GW08]. AD has the advantage of providing accurate derivative values up to machine precision. The program code of a numerical application can be considered as a multivariate vector function

$$F : \mathbb{R}^n \to \mathbb{R}^m, \tag{1}$$

where F maps an input vector $\mathbf{x} \equiv (x_i)_{i=0,\dots,n-1}$ onto an output vector $\mathbf{y} \equiv (y_i)_{i=0,\dots,m-1}$; that is, $\mathbf{y} = F(\mathbf{x})$. The assumption that F is continuously differentiable in the neighborhood of all arguments implies the existence of all entries of the Jacobian matrix $\nabla F \equiv \nabla F(\mathbf{x}) \in \mathbb{R}^{m\times n}$. To compute ∇F with AD, we have the tangent-linear model and the adjoint model at our disposal. Both models can be implemented either in scalar mode or in vector mode. In scalar mode the Jacobian is computed column by column (tangent-linear) or row by row (adjoint), respectively. The vector mode computes the whole Jacobian in one evaluation but has the downside of using much more memory. The computational complexity of the Jacobian accumulation based on the tangent-linear model depends on the input size n. The corresponding complexity of the adjoint model depends on the output size m. Since in many applications the input size is much bigger than the output size, we focus on the adjoint model in this paper. The downside of the adjoint model is the necessary data flow reversal. This leads to the need for storing values in the forward section that are being overwritten later. These values are restored in the reverse section, making the adjoint model expensive because of its memory usage.

OpenMP [Ope08] is a good way to parallelize a numerical simulation merely by augmenting the code with compiler directives. The software engineer needs to be aware only of data dependencies in the numerical simulation instead of needing detailed insight into parallel programming as would be necessary for

POSIX threads [IEE92]. The OpenMP directives are also useful for AD. There is the problem of obtaining non-parallelizable code by reversing a parallelizable code. Because OpenMP is used in many numerical applications, it is important to investigate automatic adjoint code generation of OpenMP augmented code. This paper is the first step toward a source-to-source transformation for adjoining OpenMP augmented code.

The structure of this paper is as follows. Section 2 briefly discusses previous work with OpenMP and AD. Section 3 gives a brief overview of the basic principles of AD and the adjoint model. In Section 4 we show selected features of the OpenMP standard that occur in Section 5 where the source-to-source transformation is presented. Results are presented in Section 6. Section 7 closes the paper with a description of possible next steps.

2 Related Work

In [BLaMB01] the authors assume that the AD tool augments each assignment in the input code by a loop where the gradient of this assignment is computed. Since the computations of each gradient entry are mutually independent, the authors suggest to parallelize this loop with OpenMP. Based on this approach, [BLR+02] tries to improve synchronizing and workload issues. What both approaches have in common is the application of OpenMP to a sequential code generated by AD. Sequential in a way that the generated code does not contain any exploitation of parallelism. In [BRW04] the input code is assumed to have OpenMP directives. The authors suggest to create below this top-level parallel evaluation another parallel level that computes the gradient in parallel. This is possible with OpenMP by enabling nested parallelization what is disabled by default. Nested parallelization means that each thread of the current level creates another group of threads when it encounter the parallel directive of OpenMP. The authors mention that AD tools could be capable of handling programs with OpenMP directives but they do not describe how. Especially in the adjoint case the transformation of the OpenMP directives is not straightforward.

In contrast to the above work we want to focus mainly on the adjoint mode with its quality to be able to compute a whole gradient with only one evaluation as opposed to the componentwise gradient computation in tangent-linear mode. The original code is assumed to contain OpenMP directives. Their handling by an adjoint code generation algorithm is the subject of this paper.

Related work focusing on distributed parallelization with MPI can be found in [SNHU10,SNF10,UHH+09]. Common parallel techniques for AD are described in [VN07,Ben96,BGJ91,Bis91,Fis90] discussing how the structure of the Jacobian can be exploited by seeding the evaluation not only with one Cartesian basis vector but also with a sum of Cartesian basis vectors. This can be done in $\mathbb{R}^n$ for the tangent-linear model or in $\mathbb{R}^m$ when using the adjoint model, respectively. Here we focus on transforming an original code augmented with OpenMP directives into the adjoint model in scalar mode.

3 Algorithmic Differentiation

The function $\bar{F} : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}^n$ defined as

$$\begin{aligned} \mathbf{y} &= F(\mathbf{x}) \\ \bar{\mathbf{x}} &= \bar{\mathbf{x}} + \nabla F(\mathbf{x})^T \cdot \bar{\mathbf{y}} \end{aligned} \tag{2}$$

is known as the adjoint model of F. $\bar{F}$ is obtained by augmenting F. The Jacobian matrix can be computed row by row in $\bar{\mathbf{x}} = (x_i)_{i=0,\dots,n-1}$ by initializing $\bar{x_i} = 0$ for $i = 0, \dots, n-1$ followed by letting $\bar{\mathbf{y}} = (y_i)_{i=0,\dots,m-1}$ range over the Cartesian basis vectors in $\mathbb{R}^m$. Hence, m evaluations are necessary to compute the whole Jacobian matrix. If $m = 1$, then we need only one evaluation to get all entries of the gradient instead of n evaluations of the tangent-linear model. The downside is the data flow reversal of F needed for the adjoint code in $\bar{F}$ [NULF04].

Let $F : \mathbb{R}^n \to \mathbb{R}$ be given as

$$\sum_{i=0}^{n-1} \left(\sum_{j=0}^{n-1} \sin\left(a_{i,j} \cdot x_j\right) \right)^2 \quad , \tag{3}$$

where $A = (a_{i,j})_{i,j=0,\dots,n-1}$, $\mathbf{x} = (x_j)_{j=0,\dots,n-1}$. A possible implementation of F is shown in Listing 1.1. We are interested in the derivative of y with respect to x. Applying (2) to Listing 1.1 leads to the adjoint code shown in Listing 1.2. The application of (2) to the assignment in Listing 1.1, line 9 leads to the code shown in Listing 1.2, line 13 and line 22. Since the value of z is overwritten in each iteration of the forward section while the value is needed in the reverse section (see Listing 1.2, line 19), we store this variable onto a stack in line 10 and restore the value in line 24.

```
   void f(int n, double** A, double* x, double& y)
 2 {
     int i=0, j=0;
 4   double z=0;
     y=0;
 6   for(i=0;i<n;i++) {
       z=0;
 8     for(j=0;j<n;j++) {
         z+=sin(A[i][j]*x[j]);
10     }
       y+=z*z;
12   }
   }
```

Listing 1.1. Example implementation of (3)

Since the iterations of the outer loop in Listing 1.1, line 6 are all independent of each other we can parallelize this loop. How this can be done with OpenMP is shown in Section 4. This parallelization not only gives better runtime performance but also reduces the memory requirement of the forward section of the adjoint code, as we will see in Section 5.

```
void a1_f(int n, double** A,
      double* x, double* a1_x,
      double& y, double& a1_y)
{
  int i=0, j=0;
  double z=0, a1_z=0;
  stack<double> fds;
  y=0;
  for(i=0;i<n;i++) {
    fds.push(z);
    z=0;
    for(j=0;j<n;j++) {
      z+=sin(A[i][j]*x[j]);
    }
    y+=z*z;
  }
  // reverse section
  for(i=n-1;i>=0;i--) {
    a1_z+=a1_y*2*z;
    for(j=n-1;j>=0;j--) {
      a1_x[j]+=
        a1_z*cos((A[i][j]*x[j]))*A[i][j];
    }
    z=fds.top(); fds.pop();
    a1_z=0;
  }
  a1_y=0;
}
```

Listing 1.2. Adjoint code of Listing 1.1 implementing (2)

4 OpenMP

OpenMP is a shared-memory parallel programming concept. As opposed to using POSIX threads, the user parallelizes code by augmenting it with OpenMP's directives telling the compiler which instructions to execute in parallel. The OpenMP API provides a collection of compiler directives. For the sake of brevity we will focus on selected OpenMP directives for C that are relevant to the following discussion. A detailed overview of OpenMP can be found in [CJP07,CDK+01].

The directive #pragma omp parallel { ... } declares the code between the curly brackets as executable in parallel. In the context of shared-memory parallelization OpenMP distinguishes between shared and private memory. By default all variables are shared; that is, all threads have access to the same variables. To define variables as private for each thread, OpenMP provides the private($<$ list_of_variables $>$) clause. For any variable declared in the list of variables each thread is provided with its own instance of this variable having the same type and size. The value of this variable is undefined when the thread starts its execution. In order to define a private copy of an original variable and to initialize it with the value of the original variable, the firstprivate ($<$ list_of_variables $>$) clause is used. In numerical programs there is often the need for a reduction operation, and for that reason OpenMP has a clause called reduction(operator :$<$ list_of_variables $>$), where operator is one of $\{+, -, *, \&, |, \hat{}, \&\&, ||\}$. For each list item a private copy is created in each thread and is initialized with the neutral element of the operation. At the end of the parallel region, the original variable of the list item is computed with the values of the private copies using the specified operator. OpenMP provides a synchronizing mechanism with the directives critical and atomic. The directive critical defines a code block as a critical section, whereas atomic defines only the statement following the directive to be executed by only one thread at a time.

We parallelize the execution of the loop in Listing 1.1 by augmenting the code with OpenMP directives. In Listing 1.3 the sequential part before line 5 and after line 13 is executed by the *initial thread* or *master*. When the master thread reaches the OpenMP parallel region, it creates a group of threads that processes the parallel region. By default all variables are shared. We have to declare i, j, z as private in order to give each thread its own instance. Otherwise all threads would write to the same variable in parallel. The reduction operation in line 12 is handled correctly by defining sy as part of a linear reduction; see the reduction clause in line 6.

As mentioned before, we need to reverse the entire data flow. Parallelizable code inside the implementation of F does not imply parallelizable code in the resulting implementation of $\bar{F}$. Next we investigate data races inside the reverse section of $\bar{F}$.

5 Adjoint OpenMP

It is important to know that parts of the input code are parallelizable. This is not only advantageous for the runtime efficiency but also for the memory consumption of the adjoint model. We assume a parallel code region p with parallel program paths $p_{1,\ldots,N}$ and input variables I. Each path p_i starts with

```
void f(int n, double** A, double* x, double& y)
{
  int i=0, j=0;
  double z=0, sy=0;
#pragma omp parallel for private(i,j,z) \
                         reduction(+:sy)
  for(i=0;i<n;i++) {
      z=0;
      for(j=0;j<n;j++) {
         z+=sin(A[i][j]*x[j]);
      }
      sy += z*z;
  }
  y=sy;
}
```

Listing 1.3. Augmenting Listing 1.1 with an OpenMP directive.

the same input data at program state s_1. There is no data flow between p_i and p_j with $i \neq j, i, j \in \{1, \ldots, N\}$; otherwise it would not be a parallel path. When the reverse section reaches the adjoint of p, say $\bar{p}$, all input variables in I must be guaranteed to have the same values as at time s_1. This requirement can be met by checkpointing these values.

A naive adjoint implements the adjoint of the loop as first executing all loop iterations p_i in the given sequence eventually followed by executing the adjoint iterations $\bar{p}_i$ in the reverse sequence. The mutual independence of the loop iterations can be exploited by a simple reordering of the p_i and the $\bar{p}_i$. Without the reordering all needed values of all iterations are written to the stack at once, making the stack size required for handling the loop proportional to the iteration count. see Figure 1 (left). By letting each $\bar{p}_i$ immediately follow its corresponding $\bar{p}$ that stack size requirement is decoupled from the iteration count. In other words, it yields an N-fold reduction in the stack size and only depends on the length of a single loop iteration, see Figure 1 (center). This does not require any parallel execution.

If the given hardware supports N-way parallelism, then one obviously obtains an execution time reduction roughly by a factor of N while the overall stack size remains the same as in the case without reordering, see Figure 1 (right). For k-way parallelism ($k < N$) the tradeoff for the lesser time reduction is a $\frac{N}{k}$-fold stack size reduction over the unreordered case.

The minimal stack size costs can be achieved by executing $\bar{p}$ sequentially as shown in Listing 1.4 for our example code in Listing 1.3. From line 10 to line 14 the recomputation of p_i is shown. Here we do not push any value onto the stack; in more complex code it would be necessary to store some values onto the stack for the execution of $\bar{p}_i$. The code for $\bar{p}_i$ is shown from line 15 to line 19 .

When processing $\bar{p}$ in parallel we encounter a data race for `a1_x[j]`. To resolve this problem, the OpenMP standard provides the constructs `atomic` and `critical` . In Listing 1.5 we see a parallel adjoint implementation of our example code. Line 23 contains the data race since threads are trying to write to the memory location `a1_x[j]` in parallel. The synchronization by the `atomic` construct solves this data race.

```
void a1_f(int n, double** A,
          double* x, double* a1_x,
          double& y, double& a1_y)
{
  int i=0, j=0;
  double z=0,a1_z=0;
  double sy=0,a1_sy=0;
  a1_sy+=a1_y; a1_y=0;
  for(i=0;i<n;i++) {
      z=0;
      for(j=0;j<n;j++) {
          z+=sin(A[i][j]*x[j]);
      }
      sy += z*z;
      a1_z+=a1_sy*2*z;
      for(j=n-1;j>=0;j--) {
          a1_x[j]+=a1_z*cos(A[i][j]*x[j])*A[i][j];
      }
      a1_z=0;
  }
}
```

Listing 1.4. Sequential execution of the adjoint code for Listing 1.3 by exploiting knowledge about the parallelizable loops in the original code.

```
void a1_f(int n, double** A,
          double* x, double* a1_x,
          double& y, double& a1_y)
{
  int i=0, j=0;
  double z=0,a1_z=0;
  double sy=0,a1_sy=0;
  a1_sy+=a1_y; a1_y=0;
#pragma omp parallel private(i,j,z) \
                     firstprivate(sy,a1_sy)
  {
      double a1_z=0;
#pragma omp for
      for(i=0;i<n;i++) {
          z=0;
          for(j=0;j<n;j++) {
              z+=sin(A[i][j]*x[j]);
          }
          sy += z*z;
          a1_z+=a1_sy*2*z;
          for(j=n-1;j>=0;j--) {
#pragma omp atomic
              a1_x[j]+=a1_z*cos(A[i][j]*x[j])*A[i][j];
          }
          a1_z=0;
      }
  }
}
```

Listing 1.5. Synchronization of parallel adjoint code by the OpenMP directive atomic.

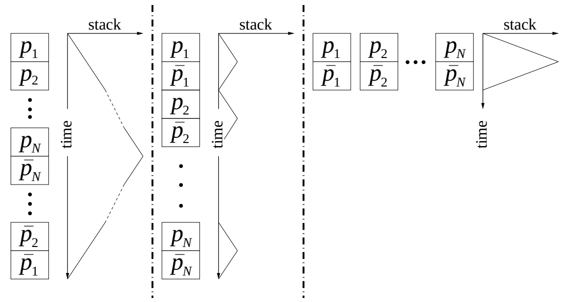

Fig. 1. Stack size vs. run time for original iteration ordering (left), reordered iterations (center), parallel execution (right).

The reduction clause in Listing 1.3, line 6 becomes a firstprivate clause; see Listing 1.5, line 10. In the forward section the reduction operation implies that a private copy of sy is created, one for each implicit task, as if the private clause had been used. This private copy is initialized with zero in the case of a linear reduction. At the end of the OpenMP region (Listing 1.3, line 12), the original sy instance is updated by adding its original value to the final value of each of the private copies. This yields a data flow from all private copies to the original value sy, see Figure 2. Following the rule for the adjoint in (2) we know that for the gradient ∇F for a simple summation is the vector $[1, \ldots, 1]$. The x in (2) are the threadprivate sy_i and the y in (2) is the final sy value in the master thread after the summation. Following (2) the (thread private) adjoints $\overline{sy}_i$ that correspond to the thread private sy_i should be incremented by the value of the $\overline{sy}$ in the master thread. Because the thread private $\overline{sy}_i$ are initialized to 0 the increment amounts to an assignment and can therefore be implemented as a firstprivate clause (think broadcast) of $\overline{sy}$. Figure 3 illustrates the above and shows the data flow reversal in comparison to Figure 2.

Therefore, an adjoint variable of some reduction variable becomes a member of a firstprivate clause in the reverse section. In addition, the variable of the linear reduction itself, here sy, becomes part of a firstprivate clause as each thread recomputes the value for the private copy; see Listing 1.5, line 19. But this value does not have to be propagated outside of the parallel region p.

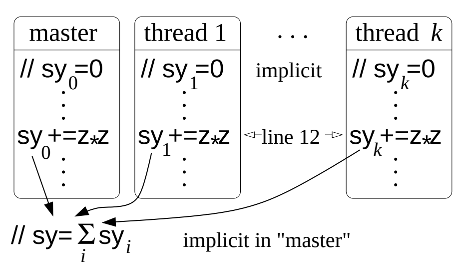

Fig. 2. Data flow for sum reduction, see lines 6 and 12 in Listing 1.3

Another possible source-to-source transformation for our example is based on the fact that one can derive the number of write accesses to a1_x[j] at compile time. This is implied by the assignment in Listing 1.3, line 10 inside the

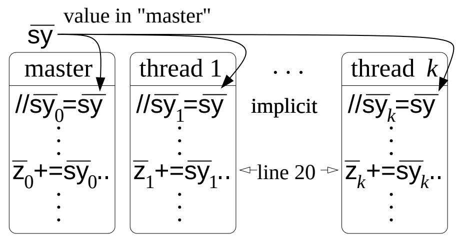

Fig. 3. Data flow for adjoint of sum reduction, see lines 10 and 20 in Listing 1.5

counting loop with induction variable j running from 0 to n−1. The fact that this loop references components of x with indices 0 to n−1 can be exploited for a1_x by giving each thread a private copy of a1_x. Each thread is able to write to this copy without any synchronization. Obviously there is an upper bound for the replication of data due to the memory bound. In practice there has to be a tradeoff between synchronizing the write accesses causing a data race by replicating data or by using the atomic construct. After the parallel section a reduction operation follows to bring the values from the copies to the original instance. Listing 1.6 shows this transformation. The private copies are allocated in line 18. Each thread writes to its own copy (see line 28). The reduction of all copies can be found in line 33 to line 36.

Clearly, the example we chose to illustrate the handling of some OpenMP pragmas is just that, an example. For instance, production reductions pose a more difficult problem when one aims at high efficiency of the adjoint. In a production reduction the ith element of the gradient ∇F can be written as $\prod_{j\neq i} x_j$, i.e. they are themselves product reductions and therefore there is no simple firstprivate solution as in the case of the summation reduction. These observations have been made for similar case of adjoint MPI reduction operations in [UHH$^+$09].

6 Experimental Results

The test system has two Intel Xeon X5570 CPUs. Each CPU has four cores, each with a clock speed of 2.93 GHz and the ability to run two hardware threads (hyperthreading). To show that the solutions in Listing 1.5 and in Listing 1.6 scale with an increasing number of threads, we ran a test with an input size of $n = 40 \cdot 10^3$. In Figure 4 we see that both scale almost identically. With 16 threads we achieve a speedup of 11. The solution with a private array for each thread scales similar to the original code from Listing 1.3 and almost perfectly until the number of threads reaches 8. At this point the physical bound of cores has been reached. More than 8 threads are executed with the help of hyperthreading.

Tests for the sequential solution in Listing 1.4 with an increasing input size n show run times between 3 seconds ($n = 5 \cdot 10^3$) and 209 seconds ($n = 40 \cdot 10^3$). The speedup results for the parallel solutions in Listing 1.5 and in Listing 1.6 with respect to the input size n are given in Figure 5. The number of threads was set to 16 for this test. The plot with square marks is for results using the synchronization construct critical instead of atomic. The overhead for this directive is so high that there is no gain in speed at all. On the other hand the synchronization with the atomic construct shows very good results in comparison

```
void a1_f(int n, double** A,
          double* x, double* a1_x,
          double& y, double& a1_y)
{
  int i=0, j=0;
  double z=0,a1_z=0;
  double sy=0,a1_sy=0;
  a1_sy+=a1_y; a1_y=0;
  int num_threads=omp_get_max_threads();
  double** a1_x_tmp = new double*[num_threads];
#pragma omp parallel private(i,j,z) \
                     firstprivate(sy,a1_sy)
  {
      double a1_z=0;
#pragma omp for
      for(i=0;i<num_threads;i++)
          a1_x_tmp[i]=NULL;
      a1_x_tmp[omp_get_thread_num()]=new double[n];
#pragma omp for
      for(i=0;i<n;i++) {
          z=0;
          for(j=0;j<n;j++) {
              z+=sin(A[i][j]*x[j]);
          }
          sy += z*z;
          a1_z+=a1_sy*2*z;
          for(j=n-1;j>=0;j--)
              a1_x_tmp[omp_get_thread_num()][j]+=
                  a1_z*cos(A[i][j]*x[j])*A[i][j];
          a1_z=0;
      }
#pragma omp for
      for(i=0;i<n;i++)
          for(j=0;j<num_threads;j++)
              if(a1_x_tmp[j])
                  a1_x[i]+=a1_x_tmp[j][i];
#pragma omp for
      for(j=0;j<num_threads;j++)
          delete [] a1_x_tmp[j];
  } // end of parallel region
  delete [] a1_x_tmp;
}
```

Listing 1.6. Synchronization of parallel adjoint code by memory extension.

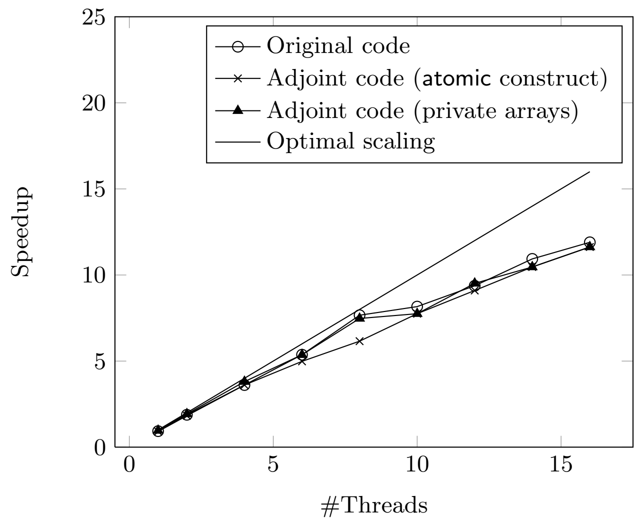

Fig. 4. Speedup for parallel adjoint code with respect to number of threads.

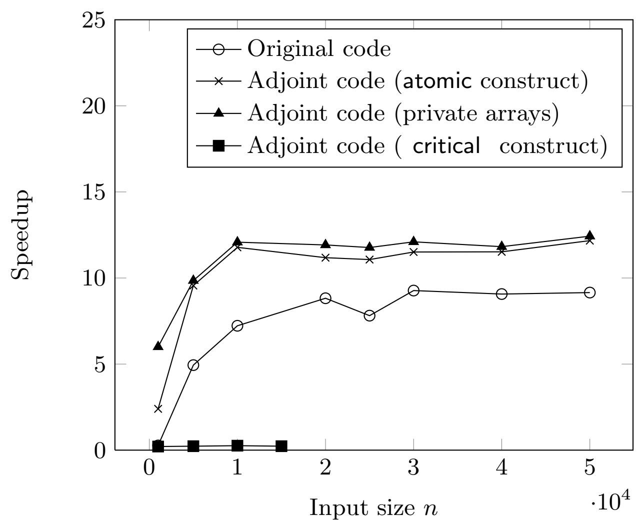

Fig. 5. Speedup for parallel adjoint code with respect to the input size n.

with the solution with a private copy for each thread. Both solutions have almost the same speedup results up to an input size of $n = 50 \cdot 10^3$. We note that the speedup results for the adjoint code are better than those for the original code, likely because of the omitted reduction clause in the adjoint code.

7 Outlook

Despite the fact that the adjoint obtained from a parallel code may end up in a non-parallelizable code, the results for our simple example look promising. We saw that the summation reduction clause inside the original code becomes a firstprivate clause in the adjoint code through the implicit data flow of the reduction clause. This is an illustration for building a body of rules and conditions to allow an automatic adjoint transformation of OpenMP pragmas. We are investigating how other OpenMP clauses, for instance the lastprivate clause, should be treated in adjoint code. Another aspect of the ongoing work is how to exploit all facts about data dependencies implied in the OpenMP pragmas for the adjoint construction and how to automatically ascertain that the prerequisites for parallelizing the adjoint code that is the result of a transformation are satisfied.

A common way to implement OpenMP is a source-to-source transformation that replaces the original OpenMP part with a call to an external function; see [LHC+07,LQPdS10,Nov10]. This approach is called *outlining*. The external function contains the code from the OpenMP part, handles the memory-sharing issues according to the clauses used in the OpenMP directive, and creates the threads to process the parallel code. We intend to implement adjoint OpenMP in the same way by outlining the parallel regions. Targets for this implementation are `dcc` [SNF10] and OpenAD [NUW+06].

References

[Ben96] J. Benary. Parallelism in the reverse mode. In *Computational Dierentiation: Techniques, Applications, and Tools. SIAM Philadelphia*, pages 137–148. SIAM, 1996.

[BGJ91] C. Bischof, A. Griewank, and D. Juedes. Exploiting parallelism in automatic differentiation. In *Proceedings of the 5th international conference on Supercomputing*, ICS '91, pages 146–153, New York, 1991. ACM.

[Bis91] C. Bischof. Issues in parallel automatic differentiation. In *Proceedings of the 1991 International Conference on Supercomputing*, pages 146–153. ACM Press, 1991.

[BLaMB01] M. Bücker, B. Lang, D. an Mey, and C. Bischof. Bringing together automatic differentiation and OpenMP. In *ICS '01: Proceedings of the 15th international conference on Supercomputing*, pages 246–251, New York, 2001. ACM.

[BLR+02] M. Bücker, B. Lang, A. Rasch, C. Bischof, and D. an Mey. Explicit Loop Scheduling in OpenMP for Parallel Automatic Differentiation. *High Performance Computing Systems and Applications, Annual International Symposium on*, 0:121, 2002.

[BRW04] M. Bücker, A. Rasch, and A. Wolf. A class of OpenMP applications involving nested parallelism. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 220–224, New York, 2004. ACM.

[CDK+01] R. Chandra, L. Dagum, D. Kohr, D. Maydan, J. McDonald, and R. Menon. *Parallel Programming in OpenMP*. Morgan Kaufmann Publishers Inc., San Francisco, 2001.

[CJP07] B. Chapman, G. Jost, and R. Pas. *Using OpenMP: Portable Shared Memory Parallel Programming (Scientific and Engineering Computation)*. MIT Press, 2007.

[Fis90] H. Fischer. Automatic differentiation: parallel computation of function, gradient, and Hessian matrix. *Parallel Computing*, 13(1):101–110, 1990.

[GW08] A. Griewank and A. Walter. *Evaluating Derivatives. Principles and Techniques of Algorithmic Differentiation (2nd ed.)*. SIAM, Philadelphia, 2008.

[IEE92] IEEE. IEEE: Threads Extension for Portable Operating Systems (Draft 6). Specification, 1992.

[LHC+07] C. Liao, O. Hernandez, B. Chapman, W. Chen, and W. Zheng. OpenUH: An optimizing, portable OpenMP compiler. *Concurr. Comput. : Pract. Exper.*, 19:2317–2332, December 2007.

[LQPdS10] C. Liao, D. Quinlan, T. Panas, and B. de Supinski. A ROSE-Based OpenMP 3.0 research compiler supporting multiple runtime libraries. In *IWOMP*, pages 15–28, 2010.

[Nov10] D. Novillo. OpenMP and automatic parallelization in GCC. Developers' Summit, 2010.

[NULF04] 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. IEEE Computer Society, 2004.

[NUW+06] U. Naumann, J. Utke, C. Wunsch, C. Hill, P. Heimbach, M. Fagan, N. Tallent, and M. Strout. Adjoint code by source transformation with OpenAD/F. In *Proceedings of the European Conference on Computational Fluid Dynamics (ECCOMAS CFD 2006)*. TU Delft, 2006.

[Ope08] OpenMP Architecture Review Board. OpenMP Application Program Interface. Specification, 2008.

[SNF10] M. Schanen, U. Naumann, and M. Förster. Second-order adjoint algorithmic differentiation by source transformation of MPI code. In *Recent Advances in the Message Passing Interface, Lecture Notes in Computer Science*, pages 257–264. Springer, 2010.

[SNHU10] M. Schanen, U. Naumann, L. Hascoët, and J. Utke. Interpretative adjoints for numerical simulation codes using MPI. In *Procedia Computer Science*, pages 1819–1827. Elsevier, 2010.

[UHH+09] J. Utke, L. Hascoët, C. Hill, P. Hovland, and U. Naumann. Toward adjoinable MPI. In *Proceedings of IPDPS 2009*, 2009.

[VN07] E. Varnik and U. Naumann. Parallel Jacobian accumulation. In G. Joubert, C. Bischof, F. Peters, T. Lippert, M. Bücker, P. Gibbon, and B. Mohr, editors, *Parallel Computing: Architectures, Algorithms and Applications. Proceedings of the International Conference ParCo 2007, Sep. 2007*, pages 311–318, 2007.

Aachener Informatik-Berichte

This is the list of all technical reports since 1987. To obtain copies of 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-Red^+ - 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, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed 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"

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht über den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Köln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Röglin, Berthold Vöcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-01 * Fachgruppe Informatik: Jahresbericht 2005

2006-02 Michael Weber: Parallel Algorithms for Verification of Large Systems

2006-03 Michael Maier, Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulation

2006-05 Uwe Naumann, Jean Utke, Patrick Heimbach, Chris Hill, Derya Ozyurt, Carl Wunsch, Mike Fagan, Nathan Tallent, Michelle Strout: Adjoint Code by Source Transformation with OpenAD/F

2006-06 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Divide-and-Color

2006-07 Thomas Colcombet, Christof Löding: Transforming structures by set interpretations

2006-08 Uwe Naumann, Yuxiao Hu: Optimal Vertex Elimination in Single-Expression-Use Graphs

2006-09 Tingting Han, Joost-Pieter Katoen: Counterexamples in Probabilistic Model Checking

2006-10 Mesut Günes, Alexander Zimmermann, Martin Wenig, Jan Ritzerfeld, Ulrich Meis: From Simulations to Testbeds - Architecture of the Hybrid MCG-Mesh Testbed

2006-11 Bastian Schlich, Michael Rohrbach, Michael Weber, Stefan Kowalewski: Model Checking Software for Microcontrollers

2006-12 Benedikt Bollig, Joost-Pieter Katoen, Carsten Kern, Martin Leucker: Replaying Play in and Play out: Synthesis of Design Models from Scenarios by Learning

2006-13 Wong Karianto, Christof Löding: Unranked Tree Automata with Sibling Equalities and Disequalities

2006-14 Danilo Beuche, Andreas Birk, Heinrich Dreier, Andreas Fleischmann, Heidi Galle, Gerald Heller, Dirk Janzen, Isabel John, Ramin Tavakoli Kolagari, Thomas von der Maßen, Andreas Wolfram: Report of the GI Work Group "Requirements Management Tools for Product Line Engineering"

2006-15 Sebastian Ullrich, Jakob T. Valvoda, Torsten Kuhlen: Utilizing optical sensors from mice for new input devices

2006-16 Rafael Ballagas, Jan Borchers: Selexels: a Conceptual Framework for Pointing Devices with Low Expressiveness

2006-17 Eric Lee, Henning Kiel, Jan Borchers: Scrolling Through Time: Improving Interfaces for Searching and Navigating Continuous Audio Timelines

2007-01 * Fachgruppe Informatik: Jahresbericht 2006

2007-02 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, and Harald Zankl: SAT Solving for Termination Analysis with Polynomial Interpretations

2007-03 Jürgen Giesl, René Thiemann, Stephan Swiderski, and Peter Schneider-Kamp: Proving Termination by Bounded Increase

2007-04 Jan Buchholz, Eric Lee, Jonathan Klein, and Jan Borchers: coJIVE: A System to Support Collaborative Jazz Improvisation

2007-05 Uwe Naumann: On Optimal DAG Reversal

2007-06 Joost-Pieter Katoen, Thomas Noll, and Stefan Rieger: Verifying Concurrent List-Manipulating Programs by LTL Model Checking

2007-07 Alexander Nyßen, Horst Lichter: MeDUSA - MethoD for UML2-based Design of Embedded Software Applications

2007-08 Falk Salewski and Stefan Kowalewski: Achieving Highly Reliable Embedded Software: An empirical evaluation of different approaches

2007-09 Tina Kraußer, Heiko Mantel, and Henning Sudbrock: A Probabilistic Justification of the Combining Calculus under the Uniform Scheduler Assumption

2007-10 Martin Neuhäußer, Joost-Pieter Katoen: Bisimulation and Logical Preservation for Continuous-Time Markov Decision Processes

2007-11 Klaus Wehrle (editor): 6. Fachgespräch Sensornetzwerke

2007-12 Uwe Naumann: An L-Attributed Grammar for Adjoint Code

2007-13 Uwe Naumann, Michael Maier, Jan Riehme, and Bruce Christianson: Second-Order Adjoints by Source Code Manipulation of Numerical Programs

2007-14 Jean Utke, Uwe Naumann, Mike Fagan, Nathan Tallent, Michelle Strout, Patrick Heimbach, Chris Hill, and Carl Wunsch: OpenAD/F: A Modular, Open-Source Tool for Automatic Differentiation of Fortran Codes

2007-15 Volker Stolz: Temporal assertions for sequential and concurrent programs

2007-16 Sadeq Ali Makram, Mesut Güneç, Martin Wenig, Alexander Zimmermann: Adaptive Channel Assignment to Support QoS and Load Balancing for Wireless Mesh Networks

2007-17 René Thiemann: The DP Framework for Proving Termination of Term Rewriting

2007-18 Uwe Naumann: Call Tree Reversal is NP-Complete

2007-19 Jan Riehme, Andrea Walther, Jörg Stiller, Uwe Naumann: Adjoints for Time-Dependent Optimal Control

2007-20 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, and Verena Wolf: Three-Valued Abstraction for Probabilistic Systems

2007-21 Tingting Han, Joost-Pieter Katoen, and Alexandru Mereacre: Compositional Modeling and Minimization of Time-Inhomogeneous Markov Chains

2007-22 Heiner Ackermann, Paul W. Goldberg, Vahab S. Mirrokni, Heiko Röglin, and Berthold Vöcking: Uncoordinated Two-Sided Markets

2008-01 * Fachgruppe Informatik: Jahresbericht 2007

2008-02 Henrik Bohnenkamp, Marielle Stoelinga: Quantitative Testing

2008-03 Carsten Fuhs, Jürgen Giesl, Aart Middeldorp, Peter Schneider-Kamp, René Thiemann, Harald Zankl: Maximal Termination

2008-04 Uwe Naumann, Jan Riehme: Sensitivity Analysis in Sisyphe with the AD-Enabled NAGWare Fortran Compiler

2008-05 Frank G. Radmacher: An Automata Theoretic Approach to the Theory of Rational Tree Relations

2008-06 Uwe Naumann, Laurent Hascoet, Chris Hill, Paul Hovland, Jan Riehme, Jean Utke: A Framework for Proving Correctness of Adjoint Message Passing Programs

2008-07 Alexander Nyßen, Horst Lichter: The MeDUSA Reference Manual, Second Edition

2008-08 George B. Mertzios, Stavros D. Nikolopoulos: The λ-cluster Problem on Parameterized Interval Graphs

2008-09 George B. Mertzios, Walter Unger: An optimal algorithm for the k-fixed-endpoint path cover on proper interval graphs

2008-10 George B. Mertzios, Walter Unger: Preemptive Scheduling of Equal-Length Jobs in Polynomial Time

2008-11 George B. Mertzios: Fast Convergence of Routing Games with Splittable Flows

2008-12 Joost-Pieter Katoen, Daniel Klink, Martin Leucker, Verena Wolf: Abstraction for stochastic systems by Erlang's method of stages

2008-13 Beatriz Alarcón, Fabian Emmes, Carsten Fuhs, Jürgen Giesl, Raúl Gutiérrez, Salvador Lucas, Peter Schneider-Kamp, René Thiemann: Improving Context-Sensitive Dependency Pairs

2008-14 Bastian Schlich: Model Checking of Software for Microcontrollers

2008-15 Joachim Kneis, Alexander Langer, Peter Rossmanith: A New Algorithm for Finding Trees with Many Leaves

2008-16 Hendrik vom Lehn, Elias Weingärtner and Klaus Wehrle: Comparing recent network simulators: A performance evaluation study

2008-17 Peter Schneider-Kamp: Static Termination Analysis for Prolog using Term Rewriting and SAT Solving

2008-18 Falk Salewski: Empirical Evaluations of Safety-Critical Embedded Systems

2008-19 Dirk Wilking: Empirical Studies for the Application of Agile Methods to Embedded Systems

2009-02 Taolue Chen, Tingting Han, Joost-Pieter Katoen, Alexandru Mereacre: Quantitative Model Checking of Continuous-Time Markov Chains Against Timed Automata Specifications

2009-03 Alexander Nyßen: Model-Based Construction of Embedded Real-Time Software - A Methodology for Small Devices

2009-04 Daniel Klünder: Entwurf eingebetteter Software mit abstrakten Zustandsmaschinen und Business Object Notation

2009-05 George B. Mertzios, Ignasi Sau, Shmuel Zaks: A New Intersection Model and Improved Algorithms for Tolerance Graphs

2009-06 George B. Mertzios, Ignasi Sau, Shmuel Zaks: The Recognition of Tolerance and Bounded Tolerance Graphs is NP-complete

2009-07 Joachim Kneis, Alexander Langer, Peter Rossmanith: Derandomizing Non-uniform Color-Coding I

2009-08 Joachim Kneis, Alexander Langer: Satellites and Mirrors for Solving Independent Set on Sparse Graphs

2009-09 Michael Nett: Implementation of an Automated Proof for an Algorithm Solving the Maximum Independent Set Problem

2009-10 Felix Reidl, Fernando Sánchez Villaamil: Automatic Verification of the Correctness of the Upper Bound of a Maximum Independent Set Algorithm

2009-11 Kyriaki Ioannidou, George B. Mertzios, Stavros D. Nikolopoulos: The Longest Path Problem is Polynomial on Interval Graphs

2009-12 Martin Neuhäußer, Lijun Zhang: Time-Bounded Reachability in Continuous-Time Markov Decision Processes

2009-13 Martin Zimmermann: Time-optimal Winning Strategies for Poset Games

2009-14 Ralf Huuck, Gerwin Klein, Bastian Schlich (eds.): Doctoral Symposium on Systems Software Verification (DS SSV'09)

2009-15 Joost-Pieter Katoen, Daniel Klink, Martin Neuhäußer: Compositional Abstraction for Stochastic Systems

2009-16 George B. Mertzios, Derek G. Corneil: Vertex Splitting and the Recognition of Trapezoid Graphs

2009-17 Carsten Kern: Learning Communicating and Nondeterministic Automata

2009-18 Paul Hänsch, Michaela Slaats, Wolfgang Thomas: Parametrized Regular Infinite Games and Higher-Order Pushdown Strategies

2010-02 Daniel Neider, Christof Löding: Learning Visibly One-Counter Automata in Polynomial Time

2010-03 Holger Krahn: MontiCore: Agile Entwicklung von domänenspezifischen Sprachen im Software-Engineering

2010-04 René Wörzberger: Management dynamischer Geschäftsprozesse auf Basis statischer Prozessmanagementsysteme

2010-05 Daniel Retkowitz: Softwareunterstützung für adaptive eHome-Systeme

2010-06 Taolue Chen, Tingting Han, Joost-Pieter Katoen, Alexandru Mereacre: Computing maximum reachability probabilities in Markovian timed automata

2010-07 George B. Mertzios: A New Intersection Model for Multitolerance Graphs, Hierarchy, and Efficient Algorithms

2010-08	Carsten Otto, Marc Brockschmidt, Christian von Essen, Jürgen Giesl: Automated Termination Analysis of Java Bytecode by Term Rewriting
2010-09	George B. Mertzios, Shmuel Zaks: The Structure of the Intersection of Tolerance and Cocomparability Graphs
2010-10	Peter Schneider-Kamp, Jürgen Giesl, Thomas Ströder, Alexander Serebrenik, René Thiemann: Automated Termination Analysis for Logic Programs with Cut
2010-11	Martin Zimmermann: Parametric LTL Games
2010-12	Thomas Ströder, Peter Schneider-Kamp, Jürgen Giesl: Dependency Triples for Improving Termination Analysis of Logic Programs with Cut
2010-13	Ashraf Armoush: Design Patterns for Safety-Critical Embedded Systems
2010-14	Michael Codish, Carsten Fuhs, Jürgen Giesl, Peter Schneider-Kamp: Lazy Abstraction for Size-Change Termination
2010-15	Marc Brockschmidt, Carsten Otto, Christian von Essen, Jürgen Giesl: Termination Graphs for Java Bytecode
2010-16	Christian Berger: Automating Acceptance Tests for Sensor- and Actuator-based Systems on the Example of Autonomous Vehicles
2010-17	Hans Grönniger: Systemmodell-basierte Definition objektbasierter Modellierungssprachen mit semantischen Variationspunkten
2010-18	Ibrahim Armaç: Personalisierte eHomes: Mobilität, Privatsphäre und Sicherheit
2010-19	Felix Reidl: Experimental Evaluation of an Independent Set Algorithm
2010-20	Wladimir Fridman, Christof Löding, Martin Zimmermann: Degrees of Lookahead in Context-free Infinite Games
2011-02	Marc Brockschmidt, Carsten Otto, Jürgen Giesl: Modular Termination Proofs of Recursive Java Bytecode Programs by Term Rewriting
2011-03	Lars Noschinski, Fabian Emmes, Jürgen Giesl: A Dependency Pair Framework for Innermost Complexity Analysis of Term Rewrite Systems
2011-04	Christina Jansen, Jonathan Heinen, Joost-Pieter Katoen, Thomas Noll: A Local Greibach Normal Form for Hyperedge Replacement Grammars
2011-11	Nils Jansen, Erika Ábrahám, Jens Katelaan, Ralf Wimmer, Joost-Pieter Katoen, Bernd Becker: Hierarchical Counterexamples for Discrete-Time Markov Chains

* These reports are only available as a printed version.

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