RWTH Aachen

Department of Computer Science
*Technical Report*

# Secure Multi-Party Computation with Security Modules

Zinaida Benenson, Felix C. Gärtner and Dogan Kesdogan

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2004-10

RWTH Aachen · Department of Computer Science · December 2004

# Secure Multi-Party Computation with Security Modules

Zinaida Benenson[1], Felix C. Gärtner[2], and Dogan Kesdogan[3]

[1] Lehrstuhl für Informatik IV
Email: zina@i4.informatik.rwth-aachen.de
[2] Lehr- und Forschungsgebiet Informatik IV
Email: gaertner@informatik.rwth-aachen.de
[3] Lehrstuhl für Informatik IV
Email: kesdogan@informatik.rwth-aachen.de
RWTH Aachen, Germany

**Abstract.** We consider the problem of secure multi-party computation (SMC) in a new model where individual processes contain a tamper-proof security module. Security modules can be trusted by other processes and can establish secure channels between each other. However, their availability is restricted by their host, i.e., a corrupted party can stop the computation of its own security module as well as drop any message sent by or to its security module. In this model we show that SMC is solvable if and only if a majority of processes is correct. We prove this by relating SMC to the problem of Uniform Interactive Consistency among security modules (a variant of the Byzantine Generals Problem from the area of fault-tolerance). The obtained solutions to SMC for the first time allow to compute any function securely with a complexity which is polynomial only in the number of processes (i.e., the complexity does not depend on the function which is computed). We conclude that adding secure hardware does not improve the resilience of SMC but can effectively improve the efficiency.

## 1 Introduction

### 1.1 Motivation

The problem of *secure multi-party computation* (SMC) is one of the most fundamental problems in security. The setting is as follows: a set of $n$ parties jointly wants to compute the result of an $n$-ary function $F$. Every party provides its own (private) input to this function but the inputs should remain secret to the other parties, except for what can be derived from the result of $F$. The problem is easy to solve if you assume the existence of a trusted third party (TTP) which collects the inputs, computes $F$ and distributes the result to everyone. However, the problem is very challenging if you assume that there is no TTP available and parties can misbehave arbitrarily, i.e., they can send wrong messages or fail to send messages at all. Still, the protocol must correctly and securely compute $F$ "as if" a TTP were available.

The problem was initially proposed by Yao in 1982 [Yao82]. In the first solution in 1987, Goldreich, Micali and Wigderson [GMW87] showed that in a synchronous system with cryptography a majority of honest processes can simulate a centralized trusted third party. Goldwasser [Gol97] and Goldreich [Gol02] provide comprehensive overview articles on this topic.

The obvious drawback of the general solutions is efficiency: All proposed protocols for SMC suffer from high communication complexities. This follows from their approach which involves extensive use of secret sharing and agreement protocols. To give an example, the most efficient solution [HMP00] can compute any function $F$ defined over a finite field, but it requires communicating $O(m \cdot n^3)$ field elements ($m$ is the number of multiplication gates in $F$) and at least $O(n^2)$ rounds of communication (in fact, the round complexity also depends on $F$). In this paper we revisit the problem of SMC in a stronger albeit still practical model and achieve a significant reduction in the communication complexity.

The context of this paper remains a model with no centralized TTP, but the task of jointly simulating a TTP is alleviated by assuming that parties have access to a local *security module*. Recently, manufacturers have begun to equip hardware with such modules: these include for instance smart cards or special microprocessors. These are assumed to be *tamper proof* and run a certified piece of software. Examples include the "Embedded Security Subsystem" within the recent IBM Thinkpad or the IBM 4758 secure co-processor board [DLP$^+$01]. A large body of computer and device manufacturers has founded the Trusted Computing Group (TCG) [Tru03] to promote this idea. Security modules contain cryptographic keys so that they can set up secure channels with each other. However, they are dependant on their hosts to be able to communicate with each other.

## 1.2 Contributions

We formalize the system model of untrusted hosts and security modules and investigate the resilience and efficiency of SMC in that model. In this model the security modules and their communication network form a subnetwork with a more benign fault assumption, namely that of *general omission* [PT86]. In the general omission failure model processes may simply stop executing steps or fail to send or receive messages sent to them.

We derive a novel solution to SMC in a modular way: We show that SMC is solvable if and only if the problem of *Uniform Interactive Consistency* (UIC) is solvable in the network of security modules. UIC is closely related to the problem of *Interactive Consistency* [PSL80], a classic fault-tolerance problem. From this "equivalence" we are able to derive a basic impossibility result for SMC in the new model: We then show that UIC requires a majority of correct processes and from this can conclude that SMC is impossible in the presence of a dishonest majority. This shows that, rather surprisingly, adding security modules cannot improve the resilience of SMC. However, we prove that adding security modules can considerably improve the efficiency of SMC protocols. We give a novel solution to SMC which uses security modules and requires only

$O(n)$ rounds of communication and $O(n^3)$ messages. This is the first solution for which the message and round complexity do not depend on the function which is computed.

### 1.3 Roadmap

We first present the model in Section 2. In Section 3 we define the problem of Secure Multi-Party Computation followed by the definition of Uniform Interactive Consistency in Section 4. We discuss security issues which arise if security modules should help to solve security problems in Section 5. Section 6 presents the main equivalence result and Section 7 concludes this paper with a discussion of efficiency issues and ideas for future work.

## 2 Model

### 2.1 Processes and channels

The system consists of a set of processes interconnected by a synchronous communication network with reliable secure bidirectional channels. Two processes connected by a channel are said to be adjacent.

A reliable secure channel connecting processes $P$ and $Q$ satisfies the following properties:

– (No Loss) No messages are lost during the transmission over the channel.
– (No Duplication) All messages are delivered at most once.
– (Authenticity) If a message is delivered at $Q$, then it was previously sent by $P$.
– (Integrity) Message contents are not tampered with during transmission, i.e., any change during transmission will be detected and the message will be discarded.
– (Confidentiality) Message contents remain secret from unauthorized entities.

In a synchronous network communication proceeds in rounds. In each round, a party first receives inputs from the user and all messages sent to it in the previous round (if any), processes them and may finally send some messages to other parties or give outputs to the user.

### 2.2 Untrusted hosts and security modules

The set of processes is divided into two disjoint classes: *untrusted hosts* (or simply *hosts*) and *security modules*. We assume that there exists a fully connected communication topology between the hosts, i.e., any two hosts are adjacent. We denote by $n$ the number of hosts in the system. Furthermore, we assume that every host process $H_A$ is adjacent to exactly one security module process $M_A$ (i.e., there is a bijective mapping between security modules and hosts). In this case we say that $H_A$ is *associated with* $M_A$ (i.e., $M_A$ is $H_A$'s associated security module). We call the part of the system consisting only of security modules and the communication links between them the *trusted system* (see Fig. 1).

We call the part of the system consisting only of hosts and the communication links between them the *untrusted system*. The notion of association can be extended to systems, meaning that for a given untrusted system, the *associated trusted system* is the system consisting of all security modules associated to any host in that untrusted system.

In some definitions we use the term "process" to refer to both a security module and a host. We do this deliberately to make the definitions applicable both in the trusted and the untrusted system.

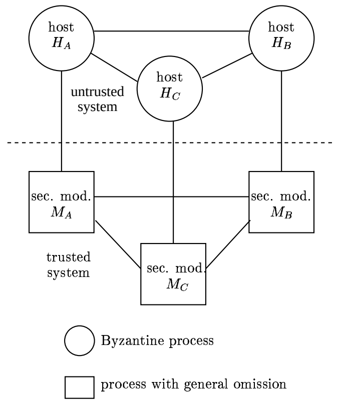

**Fig. 1.** Hosts and security modules.

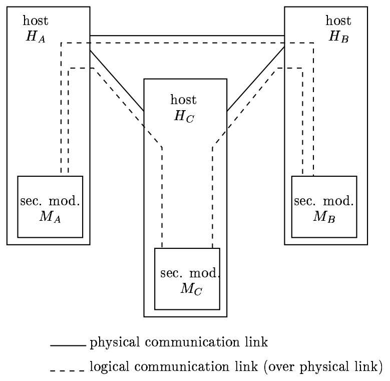

**Fig. 2.** Internet hosts with tamper-proof hardware corresponding to Fig. 1.

## 2.3 Relation to Systems with Trusted Hardware.

The model sketched above can be related to the setup in practice as follows: untrusted hosts model Internet hosts and their users, whereas security modules abstract tamper proof components of user systems (like smart cards, see Fig. 2). Intuitively, security modules can be trusted by other security modules or hosts, and hosts cannot be trusted by anybody. Hosts may be malicious, i.e., they may actively try to fool a protocol by not sending any message, sending wrong messages, or even sending the right messages at the wrong time.

Security modules are supposed to be cheap devices without their own source of power. They rely on power supply from their hosts. In principle, a host may cut off the power supply to its security module whenever he chooses, thereby preventing the security module from continuing to execute steps. Instead of doing this, a host may inhibit some or even *all* communication between its associated security module and the outside world.

### 2.4 Trust and adversary model

The setting described above is formalized using distinct failure models for different parts of the system. We assume that nodes in the untrusted system can act arbitrarily, i.e., they follow the Byzantine failure model [LSP82]. In particular, the incorrect processes can act together according to some sophisticated strategy, and they can pool all information they possess about the protocol execution. We assume however that hosts are computationally bounded, i.e., brute force attacks on secure channels are not possible.

For the trusted system we assume the failure model of *general omission* [PT86], i.e., processes can crash or fail by not sending messages or not receiving messages.

A process is *faulty* if it does not correctly follow the prescribed protocol. In particular, a security module is faulty if it crashes or commits send or receive omissions. Otherwise a process is said to be *correct*. In a system with $n$ processes, we use $t$ to denote a bound on the number of hosts which are allowed to be faulty. In general, $t$ should be at least 1 and at most $n$. Sometimes we restrict our attention to the case where $t < n/2$, i.e., where a majority of hosts is guaranteed not to misbehave. We call this the *correct majority assumption.*

## 3 Secure Multi-Party Computation

In *secure multi-party computation* (SMC), a set of processes $p_1, \ldots, p_n$, each starting with an input value $x_i$, wants to compute the result of a deterministic function $F$, i.e., $r = F(x_1, \ldots, x_n)$. Result $r$ should be computed reliably and securely, i.e., as if they were using a trusted third party (TTP). This means that the individual inputs remain secret to other processes (apart from what is given away by $r$) and that malicious processes can neither prevent the computation from taking place nor influence $r$ in favorable ways.

We assume that $F$ is a well-known deterministic function with input domain $X$ and output domain $Y$ upon which all processes have agreed upon beforehand and that all correct processes jointly begin the protocol. We say that $r$ is an *$F$-result* if $r$ was computed using $F$. Since faulty processes cannot be forced to submit their input value, $F$ may be computed using a special value $\perp \notin X$ instead of the input value of a faulty process.

Instead of defining SMC using a TTP [Gol02, Mau03], we now define SMC using a set of abstract properties.

**Definition 1 (secure multi-party computation).** *A protocol solves* secure multi-party computation (SMC) *if it satisfies the following properties:*

- *(SMC-Validity) If a process receives an $F$-result, then $F$ was computed with at least the inputs of all correct processes.*
- *(SMC-Agreement) If some process $p_i$ receives $F$-result $r_i$ and some process $p_j$ receives $F$-result $r_j$ then $r_i = r_j$.*

– *(SMC-Termination) Every correct process eventually receives an $F$-result.*
– *(SMC-Privacy) Faulty processes learn nothing about the input values of correct processes (apart from what is given away by the result $r$ and the input values of all faulty processes).*

Definition 1 defines SMC in terms of safety, liveness, and information-flow (security) properties, three distinct classes of system properties in the context of security [McL96]. From a security protocols perspective, the above definition can be considered slightly stronger than the usual (cryptographic) definitions of SMC since it demands that SMC-properties hold without any restriction. In the literature it is often stated that the probability of a violation of SMC-properties can be made arbitrarily small.

The properties of SMC are best understood by comparing them to a solution based on a TTP. There, the TTP waits for the inputs of all $n$ processes and computes the value of $F$ on all those inputs which it received. Since all correct processes send their input value to the TTP, $F$ is computed on at least those values, which motivates SMC-Validity. After computing $F$, the TTP sends the result back to all processes. Hence, all correct processes eventually receive that result (SMC-Termination). Additionally, if a process receives a result from the TTP, then it will be the same result which any other process (whether correct or faulty) will receive. This motivates SMC-Agreement.

The information-flow property SMC-Privacy defines the data-security requirements of SMC. It covers *all* possible sources of non-authorized information-flow about input values from one process to another. This covers direct as well as indirect information-flow. Direct information flow exists if a process receives the secret input of another process directly in a message. Indirect information flow occurs if some process can, for example, by observing another process' external behavior deduce the first bit of that process' input value.

Looking at a TTP-based solution again, SMC-Privacy is motivated by the fact that the TTP does all the processing and channels to the TTP are confidential: no information about other processes' input values leaks from this idealized entity, apart of what the result of $F$ gives away when it is finally received by the processes.

## 4 Uniform Interactive Consistency

The problem of *Interactive Consistency* (IC) was introduced by Pease, Shostak and Lamport in 1980 [PSL80]. It is one of the classical problems of reliable distributed computing since solutions to this problem can be used to implement almost any type of fault-tolerant service [Sch90]. In this problem, every process starts with an initial value $v_i$. To solve the problem, the set of processes needs to agree on a vector $D$ of values, one per process (Agreement property). Once vector $D$ is output by process $p$, we say that *$p$ decides $D$*. The $i$-th component of this vector should be $v_i$ if $p_i$ does not fail, and can be $\perp$ otherwise. IC is equivalent to the (also classic) *Byzantine Generals Problem* [LSP82].

The original version of IC was defined for the Byzantine failure model, i.e., where faulty processes can act arbitrarily. In such systems the Agreement property must be restricted to the set of correct processes. Definition 2 considers the version of IC in the context of the general omission failure model which allows to achieve a stronger agreement property called *Uniform Agreement.* Uniform Agreement demands that *all* processes should decide the same (if they decide) — it does not matter whether they are correct or faulty.

**Definition 2 (uniform interactive consistency).** *A protocol solves* uniform interactive consistency (UIC) *if it satisfies the following properties:*

- *(UIC-Termination) Every correct process eventually decides.*
- *(UIC-Validity) The decided vector $D$ is such that $D[i] \in \{v_i, \perp\}$, and is $v_i$ if $p_i$ is not faulty.*
- *(UIC-Uniform Agreement) No two different vectors are decided.*

Parvédy and Raynal [PR03] studied the problem of UIC in the context of general omission failures. They give an algorithm that solves UIC in such systems provided a majority of processes is correct. Since their system model is the same as the one used for trusted systems in this paper, we conclude:

**Theorem 1 ( [PR03]).** *If $t < n/2$ then UIC is solvable in the trusted system.*

Parvédy and Raynal also show that Uniform Consensus (UC), a problem closely related to UIC, can be solved in the omission failure model only if $t < n/2$. In Uniform Consensus, instead of agreeing on a vector of input values like in UIC, all processes have to decide on a single value which must be input value of some process. Given a solution to UIC, the solution to UC can be constructed in the following way: the processes first solve UIC on their input values and then output the first non-$\perp$ element of the decided vector as the result of UC. Thus, we conclude:

**Theorem 2.** *UIC is solvable in the trusted system only if $t < n/2$.*

*Proof.* As shown above, UC can be reduced to UIC. UC can be solved in the trusted system only if $t < n/2$ [PR03]. If we assume that UIC can be solved for $t \geq n/2$, then UC also can be solved, a contradiction. Therefore, there is no solution to UIC if $t \geq n/2$. □

## 5 Maintaining Secrecy in Trusted Systems

The problem of UIC, which was introduced in the previous section, will be used as a building block in our solution to SMC. The idea is that a protocol for SMC will delegate certain security-critical actions to the trusted system in which the UIC protocol runs. One could think that the security features of the tamper proof devices together with the confidentiality of secure channels between security modules offer sufficient protection from evesdropping on the data exchanged within the trusted system. Since critical data items (like the hosts' inputs to SMC) are never transmitted in cleartext, how can a host learn anything about these items? In this section we argue that we have to carefully analyse the security properties of the protocols which run within the trusted system. We show that, in general, there can be non-negligible information flow out of the trusted system and what can be done to prevent it.

### 5.1 Examples of Unauthorized Information Flow

Assume that a host $H_A$ hands its input value $v_A$ of SMC to its associated security module $M_A$ and that $M_A$ sends $v_A$ over a secure channel to the security module $M_B$ of host $H_B$. Since all communication travels through $H_B$ (see Fig. 2), $H_B$ can derive some information about $v_A$ even if all information is encrypted. For example, if no special care is taken, $H_B$ could deduce the size (number of bits) of $v_A$ by observing the size of the ciphertext of $v_A$. This may be helpul to exclude certain choices of $v_A$ and narrow down the possibilities in order to make a brute-force attack feasible.

As another example, suppose $M_A$ only sends $v_A$ to $M_B$ if $v_A$ (interpreted as a binary number) is even. Since we must assume that $H_B$ knows the protocol which is executed on $M_A$ and $M_B$, observing (or not observing) a message on the channel at the right time is enough for $M_B$ to deduce the lowerest order bit of $v_A$. In this example, the control flow of the algorithm (exhibited by the message pattern) unintentionally leaks information about secrets.

In the Privacy property of SMC we are required to prevent any type of unauthorized information flow about the input values of processes. So if the security properties of SMC depend on secrets being confined within the trusted system, we need to prevent any such information flow by carefully devising protocols that run in the trusted system.

### 5.2 Security Properties of Protocols in the Trusted System

A protocol running in the trusted system needs to satisfy two properties to be of use as a building block in SMC:

– (Content Secrecy) Hosts cannot learn any useful information about other hosts' inputs from observing the *messages* in transit.
– (Control Flow Secrecy) Hosts cannot learn any useful information about other host's inputs from observing the *message pattern*.

Both types of secrecy can be implemented in two ways. Either we give an algorithm that transforms any algorithm in the trusted system into an algorithm that satisfies the two secrecy properties, or we can adapt a known algorithm to satisfy these properties.

We now show that the UIC protocol of Parvédy and Raynal [PR03] can be modified to fulfill the two above secrecy properties. At the end of this section we will briefly sketch how to transform any UIC algorithm to one that satisfies the two secrecy properties.

### 5.3 Making Parvédy and Raynal's UIC Protocol Secure

We first analyze the UIC algorithm given by Parvédy and Raynal [PR03] and show that this algorithm already satisfies the required properties, if following changes are made to it:

- All sent messages must be end-to-end encrypted such that the consecutive encryption of the same input (i.e., plain text) results in different outputs (cipher text).
- All sent messages must have the same length.

The first change is easy fulfilled by our formal model (see Section 2), since we consider only end-to-end encrypted messages between the processes and any standard probabilistic encryption scheme that prevents replay attacks satisfies this property [MOV97]. The second change can be simply achieved by giving a constant maximum length of messages, so that all messages have to be padded (before encryption) to the given maximum length.

We now briefly explain the protocol by Parvédy and Raynal: It goes through $t+1$ synchronous rounds:

- In the first round every process $p_i$ sends his initial value $v_i$ to all other processes and initializes the $n$-ary vector $D_i$ to hold the result of the protocol, i.e. $D_i[i] := v_i$, $D_i[j] = \perp$ for all $i \neq j$. Upon receiving values from other processes, $p_i$ includes their values into $D_i$.
- In all subsequent rounds, if $p_i$ receives some value $v_j$ for the first time, it includes this value into $D_i[]$ and relays his knowledge to all processes which are not *suspected*. In case $p_i$ does not receive any new values in some round, it just sends an "empty" message meaning that he has not failed yet.

In each round each process keeps a record of those processes which he *suspects* to have failed. These are all processes from which $p_i$ did not receive any messages in some round. If $p_i$ suspects more than $t$ processes, it realizes that he himself must be faulty and quits the protocol without deciding on any vector. Otherwise, $p_i$ outputs $D_i$ at the end of round $t+1$.

**Lemma 1.** *The augmented protocol of Parvédy and Raynal [PR03] satisfies Content Secrecy.*

*Proof.* Since all messages have the same length and are encrypted using replay-safe probabilistic encryption, observation of a ciphertext message does not leak any information about the contents of the corresponding plaintext message (even if the same message contents are sent twice). □

**Lemma 2.** *The augmented protocol of Parvédy and Raynal [PR03] satisfies Control Flow Secrecy.*

*Proof.* The message pattern of the protocol does not depend on the inputs of the processes in any way. So while a host may infer information about the program counter, it cannot infer any information about values exchanged within the protocol. □

### 5.4 Making an Arbitrary UIC Protocol Secure

To provide the two secrecy properties in general, it is sufficient to use a communication protocol between the processes that ensures *unobservability.* Unobservability refers to the situation when an adversary cannot distinguish meaningful protocol actions from "random noise" [PK01]. In particular, unobservability assumes that an adversary knows the protocol which is running in the underlying network. It demands that despite this knowledge and despite observing the messages and the message pattern on the network it is impossible for the adversary to figure out in what state the protocol is. The term "state" refers to all protocol variables including the program counter, e.g., the mere fact whether the protocol has started or has terminated must remain secret.

**Definition 3 (unobservability).** *A protocol satisfies* unobservability *if an unauthorized entity which knows the protocol cannot learn any information about the state of the protocol.*

Obviously, if unobservability is fulfilled during the application of UIC then an adversary cannot obtain any information which may be derived from the control flow of the algorithm and therefore Content Secrecy and Control Flow Secrecy are fulfilled.

There are known techniques in the area of unobservable communication that guarantee perfect unobservability [PK01]. It is out of scope of this paper to present these techniques or to give a new technique providing unobservability. It goes without saying that unobservability techniques are not for free. However, the cost does not depend on the function $F$ and depends only polynomially on the number of processes (i.e. number of real messages).

## 6 Solving SMC with Security Modules

In this section we prove the main result of this paper. The following theorem shows that SMC and UIC are "equivalent" in their respective worlds. Like in the example of the protocol by Parvédy and Raynal given a Section 5, we use similar constructions to transform any UIC protocol into a protocol that provides Content Secrecy and Control Flow Secrecy.

**Theorem 3.** *SMC is solvable for any deterministic $F$ in the untrusted system if and only if UIC is solvable in the associated trusted system.*

```
**SMC**(*input* $x_i$)
  $D := secureUIC(x_i)$
  **return** $F(D)$
```

**Fig. 3.** Implementing SMC using UIC on security modules. Code for the security module of host $H_i$. The term "secure UIC" refers to a UIC protocol that satisfies Content Secrecy and Control Flow Secrecy.

*Proof.* ($\Leftarrow$) We first prove that the solvability of UIC in the trusted system implies the solvability of SMC in the untrusted system. Fig. 3 shows the transformation protocol which is executed within the security module. The hosts first give their inputs for SMC to their security modules. Security modules run "secure UIC" (i.e., UIC which satisfies Message Secrecy and Control Flow Secrecy) on these inputs, compute $F$ on the decided vector and give the result to their hosts. We prove that the properties of SMC are achieved.

First consider *SMC-Validity.* UIC-Termination guarantees that all correct processes eventually decide on some vector $D$. UIC-Validity guarantees that $D$ contains the inputs of all correct processes, and hence, SMC-Validity holds for the output of the transformation algorithm.

Consider *SMC-Agreement.* From UIC-Uniform Agreement it follows that all processes decide on the same vector. As $F$ is deterministic, all processes compute the same $F$-result if they compute such a result.

*SMC-Termination* follows immediately from UIC-Termination.

Now consider *SMC-Privacy.* Since secure UIC is executed (i.e., the construction and proof techniques presentes in Section 5 (Lemmas 1 and 2) have been applied to ensure Content Secrecy and Control Flow Secrecy) and because security modules are tamper-proof, we conclude that there is no unauthorized information flow from within the trusted system to the outside (i.e., to the untrusted system). The only (authorized) flow of information occurs at the interface of the security modules when they output the result of computing $F$. SMC-Privacy easily follows from this observation.

($\Rightarrow$) We now prove that if SMC is solvable in the untrusted system, then UIC is solvable in the trusted system.

First note that if SMC is solvable in the untrusted system, then SMC is trivially also solvable in the trusted system. This is because the assumptions available to the protocol are much stronger (general omission failures instead of Byzantine).

To solve UIC, we let the processes compute the function $F(v_1, \ldots, v_n) = (v_1, \ldots, v_n)$ (see Fig. 4). We now show that the properties of UIC follow from the properties of SMC.

*UIC-Termination* follows immediately from SMC-Termination.

To see *UIC-Validity*, consider the decided vector $D = (d_1, \ldots, d_n)$. SMC-Validity and the construction of Fig. 4 guarantee that $D$ contains the inputs of all correct processes. Consider a faulty process $p_j$ with input value $v_j$. Then either $F$ was computed using its input, and then $d_j = v_j$, or, according to our definition of SMC, function $F$ was computed using a special input value $\perp$ instead of $v_j$, and then, $d_j = \perp$.

To see *UIC-Uniform Agreement* follows directly from SMC-Agreement.

This concludes the proof. □

```
**UIC**(input $v_i$)
  $D := SMC_F\,(v_i)$
  **return** $D$
```

**Fig. 4.** Implementing UIC on security modules using SMC for the function $F(v_1, \ldots, v_n) = (v_1, \ldots, v_n)$ in the untrusted system. Code for the security module of host $H_i$.

Theorem 3 allows us to derive a lower bound on the resilience of SMC in the given system model.

**Corollary 1.** *There is no solution to SMC in the untrusted system if* $t \geq n/2$.

*Proof.* Follows from Theorems 3 and 2. □

## 7 Analysis and Conclusions

Theorem 3 and Corollary 1 show that adding security modules cannot improve the resilience of SMC compared to the standard model without trusted hardware [GMW87]. For the model of perfect security (i.e., systems without cryptography) our secure hardware has a potential to improve the resilience from a two-thirds majority [BOGW88, CCD88] to a simple majority. However, this would rely on the assumption that security modules can withstand *any* side-channel attack, an assumption which can hardly be made in practice.

Since $F$ is computed locally, the main efficiency metric for our solution is message and round complexity of the underlying UIC protocol. We estimate the worst case message complexity of the protocol of Parvédy and Raynal [PR03] (see Section 5). Each process forwards its input value and input values of other processes at most once which means that at most $n^2$ messages containing input values are sent during the protocol. Additionally, each process sends "I'm still correct"-messages to all unsuspected processes in each of $t+1$ rounds if it has not received any new input values in the previous round. Thus $O(n)$ processes send $O(n)$ "I'm still correct"-messages during the $O(n)$ rounds. Therefore, the worst case message complexity of the protocol is $O(n^3)$. This worst case complexity also holds for the private and fair modification introduced in Section 5.

Thus, our solution requires $O(n)$ rounds and $O(n^3)$ messages, whereas the most efficient solution to SMC without secure hardware requires at least $O(n^2)$

rounds and $O(mn^3)$ messages where $m$ is the number of multiplications in $F$ [HMP00].

This analysis shows the main benefit of using secure hardware: In contrast to previous solutions to SMC the message and round complexity does not depend on $F$ anymore.

Recently, Avoine *et al.* [AGGV04] studied the problem of Fair Exchange in the same system model as used in this paper. Similar to the technique in this paper, they showed that Fair Exchange is equivalent to a problem called *Biased Consensus*. Using this equivalence they showed that Fair Exchange is possible if and only if $t < n/2$. In a sense, Theorem 3 is a generalization of their findings.

Avoine *et al.* [AGGV04] also present a solution to Fair Exchange for a dishonest majority of processes in which the probability of unfairness can be made arbitrarily small. It would be interesting to derive a similar result for SMC. Also interesting future work is to extend the work in this paper to asynchronous systems in which SMC has been already studied in the general model [BOCG93].

**Acknowledgments**

We wish to thank Neeraj Mittal for helpful comments regarding the security properties of SMC. Zinaida Benenson was supported by Deutsche Forschungsgemeinschaft as part of the Graduiertenkolleg "Software for mobile communication systems" at RWTH Aachen University.

## References

[AGGV04] Gildas Avoine, Felix C. Gärtner, Rachid Guerraoui, and Marko Vukolic. Gracefully degrading fair exchange with security modules. Technical Report 200426, Swiss Federal Institute of Technology (EPFL), School of Computer and Communication Sciences, Lausanne, Switzerland, March 2004.

[BOCG93] Michael Ben-Or, Ran Canetti, and Oded Goldreich. Asynchronous secure computations. In *Proceedings of the 25th Annual Symposium on Theory of Computing (STOC)*, pages 52–61, San Diego, CA, USA, May 1993. ACM Press.

[BOGW88] Michael Ben-Or, Shafi Goldwasser, and Ari Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the 20th Annual Symposium on Theory of Computing (STOC)*, pages 1–10, Chicago, IL USA, May 1988. ACM Press.

[CCD88] David Chaum, Claude Crepeau, and Ivan Damgård. Multiparty unconditionally secure protocols. In Richard Cole, editor, *Proceedings of the 20th Annual ACM Symposium on the Theory of Computing*, pages 11–19, Chicago, IL, May 1988. ACM Press.

[DLP$^+$01] Joan G. Dyer, Mark Lindemann, Ronald Perez, Reiner Sailer, Leendert van Doorn, Sean W. Smith, and Steve Weingart. Building the IBM 4758 secure coprocessor. *IEEE Computer*, 34(10):57–66, October 2001.

[GMW87] Oded Goldreich, S. Micali, and A. Wigderson. How to play any mental game — a completeness theorem for protocols with honest majority. In *Proceedings of the 19th ACM Symposium on the Theory of Computing (STOC)*, pages 218–229, 1987.

[Gol97] Shafi Goldwasser. Multi party computations: Past and present. In *Proceedings of the sixteenth annual ACM Symposium on Principles of Distributed Computing*, pages 1–6. ACM Press, 1997.

[Gol02] Oded Goldreich. Secure multi-party computation. Internet: `http://www.wisdom.weizmann.ac.il/~oded/pp.html`, 2002.

[HMP00] Martin Hirt, Ueli Maurer, and Bartosz Przydatek. Efficient secure multi-party computation. In *Proceedings of Asiacrypt*, 2000.

[LSP82] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.

[Mau03] Ueli Maurer. Secure multi-party computation made simple. In G. Persiano, editor, *Third Conference on Security in Communication Networks (SCN'02)*, volume 2576 of *Lecture Notes in Computer Science*, pages 14–28. Springer-Verlag, 2003.

[McL96] John McLean. A general theory of composition for a class of "possibilistic" properties. *IEEE Transactions on Software Engineering*, 22(1):53–67, January 1996. Special Section—Best Papers of the IEEE Symposium on Security and Privacy 1994.

[MOV97] Alfred J. Menezes, Paul C. Van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography.* CRC Press, Boca Raton, FL, 1997.

[PK01] Andreas Pfitzmann and Marit Köhntopp. Anonymity, unobservability, and pseudonymity — A proposal for terminology. In H. Federrath, editor, *Anonymity 2000*, number 2009 in Lecture Notes in Computer Science, pages 1–9, 2001.

[PR03] Philippe Raïpin Parvédy and Michel Raynal. Uniform agreement despite process omission failures. In *17th International Parallel and Distributed Processing Symposium (IPDPS 2003).* IEEE Computer Society, April 2003. Appears also as IRISA Technical Report Number PI-1490, November 2002.

[PSL80] M. Pease, R. Shostak, and L. Lamport. Reaching agreements in the presence of faults. *Journal of the ACM*, 27(2):228–234, April 1980.

[PT86] Kenneth J. Perry and Sam Toueg. Distributed agreement in the presence of processor and communication faults. *IEEE Transactions on Software Engineering*, 12(3):477–482, March 1986.

[Sch90] Fred B. Schneider. Implementing fault-tolerant services using the state machine approach: A tutorial. *ACM Computing Surveys*, 22(4):299–319, December 1990.

[Tru03] Trusted Computing Group. Trusted computing group homepage. Internet: `https://www.trustedcomputinggroup.org/`, 2003.

[Yao82] A. C. Yao. Protocols for secure computations (extended abstract). In *23th Annual Symposium on Foundations of Computer Science (FOCS '82)*, pages 160–164, Los Alamitos, Ca., USA, November 1982. IEEE Computer Society Press.

## 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-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. Zuendorf: 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 |
| 1998-08 * | H. Aust: Sprachverstehen und Dialogmodellierung in natürlichsprachlichen Informationssystemen |
| 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 |
| 2000-04 | 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 2004 |
| 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 |

* These reports are only available as a printed version.

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