February 18, 2000

THE UNIVERSITY of MANCHESTER

Computer Science
University of Manchester

CASC-$16\frac{1}{2}$

Andrei Voronkov

Department of Computer Science
University of Manchester
Preprint Series
CSPP-4

CASC-16$\frac{1}{2}$

Andrei Voronkov*

February 18, 2000

Abstract

The recent CASC-16 competition of first-order theorem provers raised many questions about the competition design and aims. One of the raised questions is how to evaluate first-order theorem provers when only a library with a relatively small number of problems is available for benchmarks. There have been a lot of discussion about *fine-tuning* of theorem provers towards the problems from the library. In this preprint we introduce a transformation that changes first-order problems so that the transformed problems remain in a certain precise way *similar* to the original ones. However, after the transformation the provers cannot recognize particular formulas or axiomatizations. Then we define another transformation that also adds some "noise clauses" to the results of the first transformation. We discuss extensive experiments carried out on the transformed problems from CASC-16. We also discuss the future of TPTP and CASC.

Recent preprints issued by the Department of Computer Science, Manchester University, are available on WWW via URL `http://www.cs.man.ac.uk/preprints/index.html` or by ftp from `ftp.cs.man.ac.uk` in the directory `pub/preprints`.

---

*Department of Computer Science, University of Manchester. Partially supported by an EPSRC grant and a grant from the Faculty of Science and Technology.

Contents

1 Introduction **2**

2 Fine-tuning **2**

3 Problem transformations **3**
- 3.1 Extra argument transformation . . . 4
- 3.2 Preservation of orderings . . . 6
- 3.3 Other properties . . . 7
- 3.4 Noise transformation . . . 8

4 Experiments **8**

5 The CASC design and TPTP **14**
- 5.1 Two aims in theorem proving . . . 14
- 5.2 Wider range of problems . . . 14
- 5.3 Output of proofs and models . . . 15
- 5.4 Other remarks . . . 16

Bibliography **16**

A Appendix: CASC 16 and CASC $16\frac{1}{2}$: results **18**

Index **28**

List of Tables

1 Mixed category, original results, time limit 5 minutes . . . 19
2 Mixed division, extra argument transformation, time limit 10 minutes . . . 20
3 Mixed division, extra argument transformation, time limit 5 minutes . . . 21
4 Mixed division, extra argument transformation, time limit 3 minutes . . . 22
5 Mixed division, extra argument transformation, time limit 1 minutes . . . 23
6 Mixed category, noise transformation, time limit 5 minutes . . . 24
7 Unit equality division, original results, time limit 5 minutes . . . 25
8 Unit equality division, extra argument transformation, time limit 30 minutes . . . 25
9 Unit equality division, extra argument transformation, time limit 15 minutes . . . 26
10 Unit equality division, extra argument transformation, time limit 5 minutes . . . 26
11 Mixed category provers . . . 27
12 Unit equality category provers . . . 27

This preprint is unusual and may be perceived as controversial by other people. In fact, it is controversial even for myself. However, it is one of the aims of this paper to initiate discussion on all topics related to the CASC design.

1 Introduction

After the introduction of the CASC competitions, we have witnessed a considerable increase of power of first-order theorem provers. The winners of the first competition are hopelessly slow compared to the best CASC-16 systems. At the same time, the last CASC has raised questions about the reason for the increased power of provers: it was unclear to what extent this increase is due to their better design and theory, and to what extent it is due to *fine-tuning* towards the TPTP library. In order to answer these questions, we carried out experiments with the CASC-16 participants and modified versions of problems used at CASC-16.

This preprint has several aims:

1. to analyze state-of-the-art provers, their strength and weakness;
2. to understand to what extent fine-tuning has influenced the performance of provers at CASC-16;
3. to analyze possible changes in the CASC rules and TPTP maintenance.

It is not the aim of this paper to reconsider the results of CASC-16: all CASC participants followed the competition rules fairly. We do not aim at praising or discrediting any particular system: indeed, our system Vampire proved to be among the most unstable systems.

Since our experiments have been carried out between CASC-16 and CASC-17, we call them here CASC-16$\frac{1}{2}$.

CASC-16 is overviewed in [Sutcliffe 2000]. In this preprint we only consider experiments in two CASC divisions: the mixed division and the unit equality division. We are planning to make similar experiments in the SAT and FOF divisions, and also for the provers not considered here, including FDP and the TECH system consisting of two cooperating provers [Denzinger and Fuchs 1999].

Acknowledgments

Very special thanks are to Alexandre Riazanov who installed several CASC systems and wrote scripts for running them and collecting results. We acknowledge the help of the Department of Computer Science of Uppsala University who provided us with a SPARC station with 1G memory for our experiments. We thank everyone who helped us to install the systems at Uppsala, including Gernot Stenz and Dirk Fuchs.

We thank all participants of the pre-CADE discussion about CASC and fine-tuning, including Jürgen Avenhaus, Hans de Nivelle, Robert Nieuwenhuis, Stephan Schulz, Geoff Sutcliffe, Tanel Tammet, Christoph Weidenbach (even though some of them expressed quite radical views).

2 Fine-tuning

The question whether fine-tuning of provers towards problems in the TPTP library is acceptable (or to what extent it is acceptable) raised heavy debates before CASC-16. Many provers were run on the library dozens of times before the competition, then the information over different runs was collected and best ways to solve classes of problems or particular problems were identified. As a consequence, the provers supposed to solve problems within the time limit of 5 minutes, could use information about this (or similar) problems collected in previous runs with essentially unrestricted time. This is hardly a useful strategy for implementing provers which are intended for solving new problems, for example when a prover is used as an assistant in computer-aided verification.

We will briefly discuss several kinds of fine-tuning used by CASC-16 participants. The views expressed in this section belong to the author and are not necessarily shared by others.

Recognition of particular axioms. This is an extreme kind of fine-tuning. As far as we know, it was heavily used in Waldmeister, and maybe in some other systems, especially for selecting simplification orderings. In our opinion, recognition of particular axioms at CASC should only be allowed when this is based on appropriate theory. For example, systems implementing AC-unification should be allowed. Unfortunately, recognition of particular axioms was often used in the following form: "if such a set of axioms has been found, use this particular ordering and the pick-given ratio of 23". If this kind of fine-tuning is allowed, we would be interested to see a publication that proves that the pick-given ratio of 23 is the best on, say group theory problems, overall or at least in the average.

Recognition of problem sets based on rough syntactic characteristics, for example Horn sets or problems with equality. This kind of fine-tuning can be motivated by some general observations. For example, positive hyperresolution on Horn clauses derives only unit clauses, and processing unit clauses is much faster than processing multi-literal clauses. However, if this kind of fine-tuning is allowed without any limits, then nearly every single problem in TPTP can be recognized using criteria like percentage of Horn clauses, percentage or number of clauses with more than three literals etc. If this kind of tuning is allowed, one should ask how many sets of problems should provers be allowed to recognize.[1]

When particular problem sets are recognized, several provers used time-slicing, or scheduling: running a sequence (schedule) of different strategies, each with its own time limit. Scheduling is hardly very useful for the implementation of efficient provers, after all for applications one can run several strategies in parallel on several computers. Scheduling becomes much more interesting when some information collected by a prover running one strategy is used by the next strategy. Experimental evidence shows that this approach (which can be characterized as cooperation of provers) can be efficient in practice [Denzinger and Fuchs 1999]. We will make experiments with the system described in [Denzinger and Fuchs 1999] too, in order to establish its efficiency as compared to the CASC systems. The system description of Gandalf [CASC entrants 1999] suggests that it exchanges information across different runs, but there is no publication on the subject.

3 Problem transformations

In this section we define several problem transformations T used in our experiments. The main purpose of these transformations is to make provers unable to use statistical information collected during previous runs on the TPTP. At the same time, we wanted to preserve the

[1] We had to recompile the CASC version of SPASS because it rejected to run on a 233Mhz computer, there was an if-then-else statement in the code that terminated SPASS when the processor frequency is not between 300 and 333Mhz. When we recompiled SPASS, the source file containing the code for fine-tuning took several minutes to compile, due to the number of if-then-else statements it contained.

structure of the original problems as close as possible. We define two transformations having the following properties:

1. There exists a bisimulation of derivations between the original problems and the transformed ones: each resolution or paramodulation inference on the original problem corresponds to a resolution or paramodulation inference on the transformed problem, and vice versa.

2. There exists a simple and computable transformation that transforms any Herbrand model of the original problem into a Herbrand model of the transformed problem, and vice versa.

Moreover, our transformation is designed in such a way that for the standard reduction orderings (with rare exceptions) one can find isomorphic orderings on the transformed clauses. We omit some proofs in this section, but they can be easily reconstructed.

3.1 Extra argument transformation

This transformation works as follows. To each function and relation symbol p of the original problem (except for equality), we give an extra argument. Thus, a symbol of arity $k \geq 1$ becomes a symbol of arity $k+1$. Then we choose a random number n between 1 and $k+1$ and insert a new constant c_p as the nth argument to all occurrences of p.

Formally, the transformation is defined as follows. We assume that the signature of the problem is $\Sigma = \mathcal{F} \cup \mathcal{C}$, where $\mathcal{F}$ is the set of function and relation symbols (except for equality), and $\mathcal{C}$ is the set of constants occurring in the problem. We assume $\mathcal{F}$ and $\mathcal{C}$ be disjoint. We denote by *arity* the function from $\mathcal{F}$ to the set of positive integers, such that $arity(f)$ is the arity of f.

Let c be a one-to-one function defined on $\mathcal{F}$ such that $c(f)$ is a constant not occurring in $\mathcal{F} \cup \mathcal{C}$. We will write c_f instead of $c(f)$. Further, let *arg* be a function defined on $\mathcal{F}$ such that $1 \leq arg(f) \leq arity(f) + 1$. The *transformation* T maps terms, atoms, literals, clauses, and problems into terms, atoms, literals, clauses, and problems respectively as follows:

1. For terms and atoms different from equality we have

$$\begin{array}{rcl} T(v) & = & v, \text{ where } v \text{ is a variable;} \\ T(c) & = & c, \text{ where } c \text{ is a constant;} \\ T(f(t_1,\ldots,t_n)) & = & f(T(t_1),\ldots,T(t_{arg(f)-1}),c_f,T(t_{arg(f)}),\ldots,T(t_n)). \end{array}$$

2. For equality atoms we have

$$T(s \approx t) \quad = \quad T(s) \approx T(t).$$

3. For literals, clauses and sets of clauses we have

$$\begin{array}{rcl} T(\neg A) & = & \neg T(A); \\ T(L_1 \vee \ldots \vee L_k) & = & T(L_1) \vee \ldots \vee T(L_k); \\ T(\{C_1,\ldots,C_m\}) & = & \{T(C_1),\ldots,T(C_m)\}. \end{array}$$

EXAMPLE 3.1 Let us apply the transformation to the set of clauses comprising an axiomatization of group theory:

$$
\begin{array}{l}
((x,y),z) \approx *(x,*(y,z)); \\
*(x,e) \approx x; \\
*(x,i(x)) \approx e.
\end{array}
$$

The signature of this set of clauses has two function symbols $*$ and i and no relation symbols besides equality. So we introduce two new constants c_* and c_i and use them as additional arguments to $*$ and i, respectively. Suppose that the function *arg* selecting the random argument is defined by $arg(*) = 2$ and $arg(i) = 1$. Then the problem is transformed by T as follows.

$$
\begin{array}{l}
((x,c_*,y),c_*,z) \approx *(x,c_*,*(y,c_*,z)); \\
*(x,c_*,e) \approx x; \\
*(x,c_*,i(c_i,x)) \approx e.
\end{array}
$$

Let us see what properties of problems are preserved by this transformation and what properties are not. The most important for us property is preservation of inference rules.

THEOREM 3.2 *Let C_1 and C_2 be two clauses and C be their resolvent. Then $T(C)$ is a resolvent of $T(C_1)$ and $T(C_2)$. Vice versa, every resolvent of $T(C_1)$ and $T(C_2)$ has the form $T(C)$ for some resolvent C of C_1 and C_2.*

THEOREM 3.3 *Let C be a clause and D be a factor of C. Then $T(D)$ is a factor of $T(C)$. Vice versa, every factor of $T(C)$ has the form $T(D)$ for some factor D of C.* □

By a *paramodulation* we understand the following inference rule:

$$
\frac{s \approx t \vee C \quad A[s'] \vee D}{(A[t] \vee C \vee D)\theta}\ ,
$$

where s' is not a variable and θ is a most general unifier of s and s'. We call $(A[t] \vee C \vee D)\theta$ a *paramodulant* of $s \approx t \vee C$ and $A[s'] \vee D$.

By noting that T changes variables into themselves, one can modify the proof of Theorem 3.2 to obtain the following.

THEOREM 3.4 *Let C_1 and C_2 be two clauses and C be their paramodulant. Then $T(C)$ is a paramodulant of $T(C_1)$ and $T(C_2)$. Vice versa, every paramodulate of $T(C_1)$ and $T(C_2)$ has the form $T(C)$ for some paramodulant C of C_1 and C_2.* □

Theorems 3.2, 3.3, and 3.4 show that the most important inference rules used in resolution-based theorem-proving: resolution, factoring, and paramodulation, are preserved by our transformation. In the same way one can prove that model elimination inferences, including model elimination with tabulation, are also preserved.

For problems without equality, one can show a simple transformation of Herbrand models of the original problems into Herbrand models of the transformed problem, and vice versa. Such a transformation also exists for problems with equality, but it is more complex.

3.2 Preservation of orderings

However, all systems based on resolution and paramodulation use ordered versions of resolution and paramodulation, often with negative selection [Bachmair and Ganzinger 2000, Niewenhuis and Rubio 2000]. So it would be desirable to show that reduction orderings are also preserved. Since our transformation changes the signature and in addition changes the arity of function and relation symbols, we simply cannot use orderings on the original signature as orderings on the transformed one. However, in many cases there exists a very precise correspondence between orderings. There are two kinds of orderings used in resolution-based theorem proving: recursive path orderings and Knuth-Bendix orderings. We will now prove that such orderings, with one exception, can be changed into isomorphic orderings on the transformed terms.

This means that any system which uses intelligent criteria for selecting orderings, should be able to find the ordering isomorphic to the one originally found by the system. However, as far as we know, all systems either used a predefined set of orderings or selected orderings based on axioms found in problems (for example, special orderings when a group theory axiomatization has been found).

Lexicographic path ordering

We call a *precedence relation* on Σ is any total ordering on Σ. Let $\succ_\Sigma$ be a precedence relation on Σ. In this section we consider, for simplicity, constants as function symbols of arity 0. The *lexicographic path ordering* $\succ$ [Dershowitz 1982] induced by $\succ_\Sigma$ is the ordering on terms defined recursively as follows:

$t \succ x$ if $x \neq t$ and x occurs in t;

It is not the case that $x \succ t$;

Let $s = f(s_1, \ldots, s_m)$ and $t = g(t_1, \ldots, t_n)$. Then $s \succ t$ if one of the following is true:

$s_i \succeq t$, for some i with $1 \leq i \leq m$;

$f \succ_\Sigma g$ and $s \succ t_j$, for all j with $1 \leq j \leq n$;

$f = g$, $(s_1, \ldots, s_m) \succ^{lex} (t_1, \ldots, t_n)$ and $s \succ t_j$, for all j with $1 \leq j \leq n$,

where $(s_1, \ldots, s_m) \succ^{lex} (t_1, \ldots, t_m)$ if there is $j \leq m$ such that $s_j \succ t_j$ and for all $i < j$ we have $s_i = t_i$.

Denote by Σ^+ the extended signature obtained by adding the constants c_f for all $f \in \mathcal{F}$.

THEOREM 3.5 *Let $\succ_{\Sigma^+}$ be any precedence relation on Σ^+ extending a precedence relation $\succ_\Sigma$ on Σ. Let $\succ^+$ and $\succ$ be the lexicographic path orderings induced by $\succ_{\Sigma^+}$ and $\succ_\Sigma$, respectively. Then for all terms s and t of the signature Σ we have $s \succ t$ if and only if $T(s) \succ^+ T(t)$.* □

Knuth-Bendix ordering

The definition of Knuth-Bendix ordering [Knuth and Bendix 1970] is parametrized by a *weight function* on Σ, i.e., a function w from Σ to the set of natural numbers, and a precedence relation $\succ_\Sigma$ on Σ. We require of a weight function the following: if $w(f) = 0$ and f is unary, then f must be the greatest w.r.t. $\succ_\Sigma$ in Σ, and $w(c) > 0$ for all constants c.

Given a weight function w on Σ, we define the *weight* $|t|$ of term t as follows: $|x| = 1$ for every variable x $|c| = w(c)$ for every constant c and $|f(t_1, \ldots, t_n)| = w(f) + \sum_{i=1}^{n} |t_i|$.

For every variable x and term t denote by $|t|_x$ the number of occurrences of x in t.

Given a weight function w and a precedence relation $\succ_\Sigma$ on Σ, the *Knuth-Bendix ordering based on w and $\succ_\Sigma$* is the binary relation $\succ_{kb}$ on terms of Σ defined as follows (in the definition we consider constants as function symbols of 0 arguments).

1. For every variable x and term t it is not true that $x \succ_{kb} t$;

2. For every term t and variable x, $t \succ_{kb} x$ if x is a proper subterm of t.

3. For any terms $t = f(t_1, \ldots, t_n)$ and $s = g(s_1, \ldots, s_k)$ we have $f(t_1, \ldots, t_n) \succ_{kb} g(s_1, \ldots, s_k)$ if for every variable x we have $|t|_x \geq |s|_x$ and one of the following conditions holds:

   (a) $|f(t_1, \ldots, t_n)| > |g(s_1, \ldots, s_k)|$

   or

   (b) $|f(t_1, \ldots, t_n)| = |g(s_1, \ldots, s_k)|$ and one of the following holds:

      i. $f \succ_\Sigma g$ or

      ii. $f = g$ and for some $1 \leq i \leq n$ we have $t_1 = s_1, \ldots, t_{i-1} = s_{i-1}$ and $t_i \succ_{kb} s_i$.

THEOREM 3.6 *Let $\succ_\Sigma$ be a precedence relation on Σ and w be a weight function on Σ such that $w(f) \neq 0$ for all $f \in \Sigma$. Let $\succ_{\Sigma^+}$ be any extension of $\succ_\Sigma$ to a precedence relation on Σ^+. Define the weight function w^+ as follows: $w^+(c) = 2w(c)$ for all constants $c \in \mathcal{C}$, $w^+(f) = 2w(f) - 1$ for all function symbols $f \in \mathcal{F}$, and $w^+(c_f) = 1$ for every $f \in \mathcal{F}$. Denote by $\succ_{kb}$ the Knuth-Bendix ordering based on w and $\succ_\Sigma$ and by $\succ_{kb}^+$ the Knuth-Bendix ordering based on w^+ and $\succ_{\Sigma^+}$. Then for all terms s and t of the signature Σ we have $s \succ_{kb} t$ if and only if $T(s) \succ_{kb}^+ T(t)$.*
$\square$

This theorem is not true when function symbols of the weight 0 are allowed for the following reason. Take the unary function symbol f of weight 0 and a unary function symbol g of weight 1. Then $g(g(c)) \succ_{kb} f^n(c)$ for every constant c and natural number n. Therefore, there exists an infinite number of ground terms less than $g(g(c))$. After the transformation, the signature contains no unary function symbols. Then for every ground term t there exists only a finite number of terms that are less than t.

3.3 Other properties

Let us summarize some other properties preserved by the extra argument transformation.

1. The transformation preserves groundness: a clause C is ground if and only if so is $T(C)$. Moreover, the set of variables of a literal L coincides with the set of variables of $T(L)$. This means that provers that use splitting rules or optimized subsumption for clauses that can be divided in variable-disjoint parts, can also use these splitting rules on the transformed problems.

2. Horn clauses are transformed into Horn clauses, and non-Horn clauses into non-Horn. Unit (nonunit) clauses are transformed into unit (respectively nonunit) clauses.

3. A clause contains equality if and only if the transformed clause contains equality. This property and the previous one imply that the transformation preserves all categories of problems used at CASC-16 (for example, Horn equality).

4. A clause contains function symbols if and only if the transformed clause contains function symbols. This means that the transformation preserves the decidable class of problems without function symbols.

There are several natural properties of problems that are not preserved by the transformation.

1. Signatures are not preserved. However, we believe that it should be generally forbidden to distinguish signatures, since then too many single TPTP problems can be uniquely identified by their signature. Strangely enough, the fact that the signatures are not preserved influenced the prover E in such a way that it could solve more problems, because of a specific Knuth-Bendix ordering used in E.

2. Vampire (and maybe some other provers) has an optimization that eliminates, under some circumstances, clauses "defining" predicates, i.e. clauses of the form $(\neg)P(\bar{x}) \vee C$, where $\bar{x}$ is a sequence of variable. Since after the transformation each atom has non-variable arguments, Vampire could no more recognize such "definitions". However, elimination of definitions can hardly be regarded as tuning towards TPTP.

3. Of course, concrete clauses are not preserved. This means that the provers recognizing particular axiomatizations, like Waldmeister, cannot recognize them any more.

3.4 Noise transformation

This transformation extends the extra argument transformation by adding two "noise" clauses. These clauses are written in a signature disjoint from the problem's signature. One of the noise clauses contains equality, and the other one is non-Horn. Thus, the transformation is only applicable to the mixed division problems. The noise clauses use function symbols. After the addition of the noise clauses, all systems believe that all the problems contain equality, function symbols, and are non-Horn. Thus, all five subcategories of the mixed division fall into one: non-Horn with equality. This transformation was intended to observe behavior of provers which choose strategies depending on the syntactic form of the problem when no information on the syntactic structure is available.

4 Experiments

In this section we discuss our experiments. First, we briefly describe the design of the experiments, and then discuss how particular provers were affected by the problem transformations. We ran all provers on the same computer on the transformed problems taken from CASC-16 using several time limits. For the mixed division, we used the extra argument transformation with time limits of 1,3,5, and 10 minutes and the noise transformation with the time limit of 5 minutes. Our computer was slightly slower than the computer used at CASC, and transformed problems are a bit heavier than the original ones, so one can roughly estimate the CASC-16 time limit of 5 minutes in the mixed division corresponding to the time limit of 10 minutes at

CASC-16$\frac{1}{2}$. In the unit equality division we would estimate 5 minutes of CASC-16 equivalent to 15 minutes in CASC-16$\frac{1}{2}$, since the cost for processing heavier terms is much higher than for the mixed division, where other costs dominate the performance.

For the unit equality division we used the extra argument transformation with the time limits of 5, 15 and 30 minutes. As we mentioned, the noise transformation is not applicable to this division. In the appendix we give tables with the results of all runs.

Let us now briefly describe all systems that took part in CASC-16$\frac{1}{2}$. We try to emphasize their original features as compared to other systems. Before our experiments, we asked all CASC-16 participants if there are versions of their systems that can behave better on new problems than the CASC-16 versions. Based on their replies, we used two versions of the provers SPASS and E. The order of provers in this section is purely alphabetic. We overview the results obtained by the provers at CASC16$\frac{1}{2}$. We emphasize results when they are quite different compared to the original CASC-16 results. We also emphasize results that looked interesting or strange for us. It is indeed very difficult to present the results impartially. In several rare cases we could obtain explanations of results from the system implementors.

Bliksem 1.10

Bliksem [de Nivelle 2000] is developed at MPI für Informatik. It is based on resolution and superposition. Its original features are nonliftable orderings [De Nivelle 1994] and decision procedures for some decidable classes of first-order formulas [e.g. de Nivelle 1998*b*]. According to de Nivelle [CASC entrants 1999], the decidable classes are unlikely to appear at CASC. However, there were several problems belonging to the decidable class of clauses without function symbols, ironically not solved by Bliksem. It seems that Bliksem also uses some original datastructures [de Nivelle 1998*a*]. Bliksem was very stable over CASC-16$\frac{1}{2}$ and showed essentially the same performance as at CASC-16.

E 0.5

E is an equational theorem prover [Schulz 1999] developed at Munich. There is not much literature on E. It seems that the strong sides of E are an efficient implementation and intelligent literal selection. We used both the CASC version of E and a newer version. The CASC version of E was disqualified because the system was found unsound (but the bug found did not influence the performance of E at CASC-16 at all). At CASC-16$\frac{1}{2}$ we used the option which turned off an optimization that causes unsoundness.

E proved to be extremely efficient over CASC-16$\frac{1}{2}$ in the mixed division for the extra arguments transformation. For the time limits of 5 minutes and more it was second after E-SETHEO, and ahead of Vampire. However, for the noise transformation, it went down from 45 solved problems to only 24. (The results of the noise transformation were a complete surprise for us compared to the extra argument transformation in general.) Some decrease of performance of E for the noise transformation could be predicted because E was stated to be very efficient on Horn problems in [CASC entrants 1999]. But the decrease was very sharp, since E changed from solving nearly all Horn problems to solving nearly none of them.

E-SETHEO 99 csp

E-SETHEO [Moser, Ibens, Letz, Steinbach, Goller, Schumann and Mayr 1997] is developed in Munich University of Technology. It consists of 3 components (the equational prover E, the model-elimination based prover SETHEO, and the propositional satisfiability checker Semprop) plus a controlling script that decides which components and strategies should be used to solve a particular problem. In addition, in the FOF category E-SETHEO used the system FLOTTER developed at MPI [Weidenbach, Gaede and Rock 1996]. The original features of E-SETHEO are its two components: SETHEO and Semprop. SETHEO was the only model-elimination based prover at CASC-16, and Semprop was the only propositional component used at the mixed and first-order division. The system description [CASC entrants 1999] says that E-SETHEO "contains transformation techniques which may split the formula into independent subparts or which may perform a ground instantiation". It seems to be an original, but unpublished, feature.

The performance of E-SETHEO was stable for the extra argument transformation, where it was the best in all time limits except for 3 minutes. For short time limits it was closely followed by E and Vampire. However, the performance for the noise transformation was much worse. For the nonHorn problems with equality and function symbols (resulting from the noise transformation) E-SETHEO delegated most of its work to E (usually after running the SETHEO component for a few seconds). In CASC-16, E-SETHEO solved several hard problems without function symbols using the propositional component, but for the noise transformation these problems were not recognized any more, causing a performance degradation.

E-SETHEO has had another problem: on 10 different unsatisfiable problems with different time limits (from 3 to 30 minutes) it reported "model found". We communicated with Gernot Stenz, and he received different results on a computer in Munich. It seems that some strategies in the schedule under certain circumstances read the problem file before it was completely written by a syntax translator. However, SETHEO does not display the found models, which raises the issue of the competition design discussed below.

FDP 0.9

FDP [Baumgartner 2000] is developed at the University of Koblenz-Landau. It is based on a highly original calculus: first-order version of Davis-Putnam-Logeman-Loveland procedure. Other provers (E-SETHEO, SPASS and Vampire) implement some form of splitting, but FDP is the only prover that is solely based on such a rule. FDP has no built-in equality rules, equality is treated using a preprocessing transformation similar to Brand's [1975].

We did not run FDP because of installation problems, but we plan to run it later.

Fiesta 2

Fiesta implements unfailing completion. The Fiesta's original feature is the basic restriction [Degtyarev 1979, Bachmair, Ganzinger, Lynch and Snyder 1992, Nieuwenhuis and Rubio 1992]. The CASC version of Fiesta "does not adjust any system parameters according to the given axiom set or conjecture" [CASC entrants 1999]. Fiesta showed a very stable performance in the unit equality division.

Gandalf c 1.8c

Gandalf [Tammet 1997] has been developed at Chalmers University, and later at Tallinn University of Technology. At CASC-16, it was represented by two versions: the CASC-15 version was entered as the winner of CASC-15. Gandalf, as well as E-SETHEO, heavily uses *time-slicing*, or *scheduling*: it distinguishes several kinds of problems and runs a specific schedule on each kind.

Each schedule is a sequence of time slices given to a specific strategy. The system description of Gandalf says that it can combine "different strategies, in particular forward and backward reasoning, in one run". It seems to be the most original feature of Gandalf, although no published details on this subject are available. Evidently, in Gandalf some efforts are invested in efficient datastructures [Tammet 1998].

Both versions of Gandalf showed an extremely stable performance on the extra argument transformation. The noise transformation gave astonishing results: Gandalf solved essentially the same number of problems as for the extra argument transformation and was ahead of all other provers by at least 8 problems.

MUSCADET 2.1

Unlike all other systems, MUSCADET [Pastre 1989] is based on natural deduction. Not much is known about this system, though its CASC results, especially for problems with equality, were quite impressive for a non-resolution based system. We could not install the system for CASC$16\frac{1}{2}$.

OtterMACE 437

Otter is a well-known prover developed at Argonne National Laboratory based on resolution and paramodulation. It is not clear from [CASC entrants 1999] what kind of options were used by Otter for its auto-mode at CASC-16. MACE searches for finite models by fixing the model size, expressing the existence of a model by a set of propositional clauses so that any satisfying assignment for this set of clauses gives a model of the original formula. Satisfiability is checked using a DLL procedure. If the propositional formula is unsatisfiable, MACE tries to find a model of a larger size.

Otter's results were quite interesting. In the mixed division, Otter solved more transformed problems than the original ones for the extra argument transformation. When the noise transformation has been applied, Otter solved 21 problems with the time limit of 5 minutes, while at CASC it solved 16 problems with the same time limit on a faster computer. This probably shows that Otter's auto-mode is far from perfect, and it can gain from investing into strategies. In the unit equality division, the situation was similar. Otter was better on the transformed problems than on the original ones.

Another interesting observation about Otter is that it chronically terminates before the time limit (for example in the unit equality category Otter terminated on the problem `col003-1` in 112 seconds under the time limit of 30 minutes). With the time limit 15 minutes Otter solves 5 unit equality problems, while with the time limit of 30 minutes it solves 7 unit equality problems, and each one in less than 12 minutes. Otter's problems are similar to Vampire's problems, but Vampire's limited resource strategy [Riazanov and Voronkov 2000] is more stable in practice: Vampire rarely terminates before the time limit. This shows that Otter has high potential if it changes the memory limit strategy to a strategy used by other provers (scheduling, weight increment, or the limited resource strategy).

SCOTT IV

SCOTT is developed from OTTER. It is based on the idea of guiding the proof search by finding "maximally consistent sets of clauses". SCOTT is based on several heuristics for clause selection based on these maximally consistent sets of clauses. The system description of SCOTT [CASC entrants 1999] says that in the new version of SCOTT "our naive notions of semantic guidance have been superseded by those with genuine theoretical support". However, all accessible papers on SCOTT only describe heuristics and use arguments that can hardly be called theoretical. Even the notion of a maximally consistent set of clauses is used in an approximate meaning, since in reality a set of clauses is consistent if and only if it coincides with its maximally consistent subset, so there is no algorithm to check whether a given set of clauses is maximally consistent in another set of clauses.

However, the idea underlying SCOTT is very interesting and seems to be deeper than the ideas used for clause selection by other provers. When we began experiments, we were curious whether SCOTT uses a predefined set of models collected in previous runs over TPTP. However, the results of SCOTT in CASC-$16\frac{1}{2}$ were so good that there was no question of fine-tuning. In the mixed category, SCOTT was slightly better on transformed problems than on the original ones. The only problem SCOTT had was the time limit of 1 minute (3 problems versus 19 problems with the time limit of 3 minutes), but this was predicted in [CASC entrants 1999]: "searching for models and testing clauses against them can be costly, and so SCOTT may perform poorly if the time limit is too severe".

The performance of SCOTT in the unit equality division was brilliant: it lost by only a very small margin to Waldmeister'798, but was far ahead of all other provers. Since SCOTT uses the inference engine of Otter, which is believed to be slower than that of Waldmeister, the results of SCOTT show high potential of its very original approach.

SPASS 1.0.0

SPASS [Weidenbach, Afshordel, Brahm, Cohrs, Engel, Keen, Theobalt and Topic 1999] is developed at MPI für Informatik, Saarbrücken. The original features of SPASS are the use of a splitting rule [Weidenbach 2000] and an original concept of sorts [Ganzinger, Meyer and Weidenbach 1997]. Since the CASC version of SPASS was specifically tuned towards TPTP, we also used the original version from the SPASS homepage. In FOF category, SPASS is used together with the intelligent clausifier FLOTTER [Nonnengart and Weidenbach 2000], that has also been used by E-SETHEO.

The results of SPASS were a bunch of surprises, which we cannot explain. First, the performance of the CASC version of SPASS was not very different from that of the standard version. Second, the extra argument transformation resulted in a sharp decrease of performance (down to 29 problems from 50 in the mixed division). In the unit equality division SPASS could not solve any transformed problem at all, even with the time limit of 30 minutes, while it solved 7 original unit equality problems in 5 minutes. Third, the noise transformation did not change the performance at all, so after the noise transformation SPASS solved the same number of problems that E-SETHEO and Vampire, and was only behind both versions of Gandalf. Compared to CASC, SPASS had losses in nearly all categories of the mixed division, so it is difficult for us to explain its behavior. The absolutely worst results in the unit equality division seems to show that SPASS has quite a slow inference engine.

Vampire 0.0

Vampire [Riazanov and Voronkov 1999] is developed in the University of Manchester. It is a resolution-based prover with several nonstandard features. One is Limited Resource Strategy [Riazanov and Voronkov 2000] that allows one to dynamically discard clauses from the search space that are believed not to be processed by the end of the time limit. Another original feature in splitting without backtracking, that is a first-order analogue of the propositional splitting rule. Vampire has very efficient datastructures for performing some set-at-a-time operations, like forward subsumption. However, the new version of Vampire [Riazanov and Voronkov 1999] has been developed in one year from the previous version [Voronkov 1995], so several important algorithms were not implemented at all, resulting in a quite unstable performance. In addition, our limited resource strategy [Riazanov and Voronkov 2000] (which is, as we believe, one of the main reasons for our success at CASC-16) is still far from perfection. For example, there are problems that we can solve in less than 5 minutes when we set the time limit to 10 minutes, but cannot solve them with the time limit of 5 minutes.

When we began experiments, we were sure that the performance of Vampire will not be much different from the performance at CASC. At CASC we used positive and negative hyperresolution for Horn problems without equality and binary resolution (with superposition) for all other. This was motivated by the observation that positive hyperresolution on Horn clauses is very efficient because only unit clauses are generated, and indexing on unit clauses is much simpler and faster than on multiliteral clauses. Negative hyperresolution is known to be very much goal-oriented, so we decided to use it as well. However, we were confident that even with binary resolution on Horn clauses we will not lose a lot.

Binary resolution did perform reasonably well, but all of a sudden we got a few surprises. First of all, Vampire could not solve 6 combinatory logic problems easily solved before transformation. The problem (as we mentioned above) was our definition elimination optimization (which can hardly be classified as fine-tuning). After the extra argument transformation Vampire solved 43 problems as compared to 51 at CASC-16. The second surprise was the behavior of Vampire on the noise transformation: it went down to 30 problems. In addition to the combinatory logic problems we could not solve several planning problems (solved by negative hyperresolution before) and a bunch or problems obtained by translation of the knowledge representation formalism ALC into classical logic. It took us three days to understand what has happened. For problems without equality, we used a nonstandard ordering on literal, with the intention of restricting applicability of resolution. We do not know whether this ordering is complete for logic with equality, so we used a randomly chosen ordering and a very naive selection function for problems with equality. After the noise transformation, all problems had equality, so we had to use the naive ordering.

Waldmeister 798 and 799

Waldmeister [Hillenbrand, Buch, Vogt and Löchner 1997] is a superposition-based prover with an extremely fast inference engine. In our experiments, it was not unusual that Waldmeister used over 700M memory during a 30 minutes run. Two versions of Waldmeister entered CASC-16, version 798 entered as the winner of CASC-15, version 799 is the new one. The system description of Waldmeister at [CASC entrants 1999] says "we now focus on how to guide the proof search". It seems that Waldmeister also implements an approximation to hereditary ordering constraints.

Waldmeister is believed to be highly fine-tuned towards particular problems: it recognizes particular axiom sets and even reports finding concrete axioms. So Waldmeister was a particularly interesting object of study for us. At CASC-16 Waldmeister 799 solved 30 problems out of 30, while Waldmeister 798 solved 19. After the extra argument transformation Waldmeister 799 could solve only 6 problems, while Waldmeister 798 could still solve 14. Waldmeister 799 lost by a very large margin to SCOTT and was slightly ahead of E. It is very difficult for us to explain what has happened to Waldmeister. [CASC entrants 1999] says: "this year's version of our unit equality theorem prover is in most parts identical to Waldmeister 798 that won the CASC-15 UEQ competition with an enormous lead ... no major changes took place. Some portions of code have been slightly revised, e.g., a number of time-consuming loops have carefully been reworked". This seems to completely contradict to the results of CASC16$\frac{1}{2}$ since the new version of Waldmeister performed much worse than the previous year version.

5 The CASC design and TPTP

In this section we will make some remarks about design of future CASC competitions. In our opinion, the competitions should continue, as they certainly motivate further progress in the area. At the same time we need to ensure that they really motivate progress, but not regress, and discourage participants from fine-tuning. The questions of the CASC design are closely connected to the maintenance of the TPTP library, so we also discuss the future of TPTP.

5.1 Two aims in theorem proving

Automated theorem proving has at least two main aims. One is to use provers as human's assistants for finding proofs of open problems. Finding proofs of open problems is a long process which may require several years of work, the best example is the famous McCune's solution of the Robbins Algebra Problem. A proof was found by the prover in a 9-days run, but various experiments were carried out for several years before finding the proof. The current CASC rules are not intended for encouraging research in such kind of theorem proving. It seems interesting to have another competition in solving new problems that will be run for the whole year between each two CADE's. The winner will be the system that solved the largest number of previously open TPTP problems. Any kind of hardware or tuning should be allowed for this kind of competition. Having such a competition will stimulate research in solving open problems. Experimental papers about solving new problems should be encouraged at CADE[2], maybe a special category of paper should be introduced in addition to full papers and system abstracts.

The second aim of automatic theorem proving is to solve problems automatically when the time for solving them is limited. Such a use of provers is required in many applications, most notably computer-aided verification. The current CASC rules seem to be ideally suited to encourage development of provers for such applications, except that more emphasis should be put on new problems and practical problems.

5.2 Wider range of problems

One of the main problems of the competition design is the relatively low number of TPTP problems eligible for the competition (i.e. not too hard and not too easy). This problem can be overcome in two ways. One immediate solution was to change one item in the CASC-16 rules, which says that "nonstandard" versions of problems (whatever it means) are not eligible. For example, TPTP contains hundreds of problems from the field theory. Many of these problems have the right degree of difficulty for the modern provers. Some of them can only be solved

[2]We personally doubt that a paper which describes such an experiment would be accepted by any of the last CADE's.

by one prover. However, all these problems were claimed not eligible! At the same time 6 combinatory logic problems with the same axiomatization have been selected for the mixed division.

The second solution is to search for the new problems to be included. This can be done in at least three ways:

1. Many people use first-order provers in experiments, especially in computer-aided verification. We should invest efforts in collecting these problems for TPTP. It is especially important that these problems come from practice. For example, [Denzinger and Fuchs 1999] mentions a large number of problems from the domain of software components retrieval, but these problems are still not part of TPTP. In the verification community combination of model checking and theorem proving is now a hot issue, so there should be a better cooperation between theorem proving and verification communities.

2. It is possible to formalize parts of modern mathematics in a way suitable for first-order and other kinds of provers. Experiments with such formalizations and automated theorem proving can be subjects of undergraduate theses. Even a chapter of a textbook can give us a substantial set of new problems to experiment with.

3. The idea of problem generators is appealing, but nobody has ever succeeded in problem generation in first-order logic. However, in some cases we can generate problems automatically, for example, using translations of other logics (most notably description logics) into predicate logics. Another example where problem transformers can be useful is the area of relational databases, where many problems related to database schemata can be specified in first-order logic. One can automatically generate many interesting first-order problems from the schemata of particular databases.

It also seems that the random problem selection for CASC should be changed to allow for more diversity.

5.3 Output of proofs and models

The current CASC rules allow the provers not to output proofs or models. This resulted in the controversial decision to disqualify *E* and E-SETHEO after the competition, despite that the solutions found by these provers at CASC-16 were probably correct. Strangely enough, the current CASC policy discourages systems from outputting proofs and models, indeed the proof or model output amounts in extra implementation overhead and slowdown (which can be especially noticeable on unit equality problems).

However, it is difficult to imagine applications where output of models or proofs is not important. In verification, if a (counter)model is found, this model should be presented in some comprehensive way, in order to understand what is wrong with the software or hardware design. If we prove theorems in mathematics, a sheer statement of provability without displaying a proof can hardly be regarded as serious.

In our opinion, output of proofs and models must be one of the requirements at CASC. In addition to the pragmatic aspect of this requirement, it can motivate further research in model building, proof checking, and proof explanation.

Some systems may have difficulties with proof or model output. For example, provers that use saturation algorithms for checking satisfiability of a set of clauses simply claim satisfiability when no new clauses can be derived. This is hardly useful for applications, we think that some kind of model should be presented. Further research in model building using saturation procedures may be required. Another problem arises when propositional provers are used for proving a first-order theorem, by using some ground instantiation of the original problem. Propositional proofs of problems by resolution or DLL have, in the worst case, exponentially long proofs, so it may be difficult for provers with propositional components to output proofs. For such provers, one can consider the ground instantiation to be a proof.

5.4 Other remarks

Upper bound on fine-tuning. We think that the rules of CASC should be very explicit as to what kind of fine-tuning is allowed. If CASC entrants are allowed to distinguish several categories of problems (using any criteria), the (small) upper bound on the number of such categories should be posed. Otherwise, the competition does not make much sense, because each and every single problem can then be recognized. When every single problem is recognized, it is unclear where is the bound between storing detailed information about a proof of this problem and storing the proof itself.

Extending TPTP. Automated deduction is not only about first-order theorem proving. Other kinds of theorem proving, especially used in applications, should also be encouraged. We propose to extend TPTP by benchmarks in other areas, especially coming from applications, for example, induction, propositional logic, higher-order logic, Presburger's arithmetic, term algebras.

References

BACHMAIR L. AND GANZINGER H. [2000], A theory of resolution, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Elsevier Science and MIT Press. To appear.

BACHMAIR L., GANZINGER H., LYNCH C. AND SNYDER W. [1992], Basic paramodulation and superposition, *in* D. Kapur, ed., '11th International Conference on Automated Deduction', Vol. 607 of *Lecture Notes in Artificial Intelligence*, Springer Verlag, Saratoga Springs, NY, USA, pp. 462–476.

BAUMGARTNER P. [2000], FDPLL — a first-order Davis-Putnam-Logeman-Loveland procedure, *in* 'CADE'2000'. submitted.

BRAND D. [1975], 'Proving theorems with the modification method', *SIAM Journal of Computing* **4**, 412–430.

CASC ENTRANTS [1999], 'CASC-16 entrants system descriptions'.
URL: *http://www.???*

DE NIVELLE H. [1994], Resolution games and non-liftable orderings, *in* E. Börger, Y. Gurevich and K. Meinke, eds, 'Computer Science Logic, 7th Workshop, CSL'93', Vol. 832 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 279–293.

DE NIVELLE H. [1998*a*], 'An algorithm for the retrieval of unifiers from discrimination trees', *Journal of Automated Reasoning* **20**(1/2), 5–25.

DE NIVELLE H. [1998*b*], A resolution decision procedure for the guarded fragment, *in* C. Kirchner and H. Kirchner, eds, 'Automated Deduction — CADE-15. 15th International Conference on Automated Deduction', Vol. 1421 of *Lecture Notes in Artificial Intelligence*, Springer Verlag, Lindau, Germany, pp. 191–204.

DE NIVELLE H. [2000], *Bliksem 1.10 User's Manual*, MPI für Informatik, Saarbrücken.

DEGTYAREV A. [1979], 'The strategy of strict paramodulation (in Russian)', *Semiotika i informatica* (12).

DENZINGER J. AND FUCHS D. [1999], Cooperation of heterogeneous provers, *in* T. Dean, ed., 'Proc. of the Sixteenth International Joint Conference on Artificial Intelligence (IJCAI-99)', Vol. 1, Stockholm, pp. 10–15.

DERSHOWITZ N. [1982], 'Orderings for term rewriting systems', *Theoretical Computer Science* **17**, 279–301.

GANZINGER H., MEYER C. AND WEIDENBACH C. [1997], Soft typing for ordered resolution, *in* W. McCune, ed., 'Automated Deduction — CADE-14. 14th International Conference on Automated Deduction.', Vol. 1249 of *Lecture Notes in Artificial Intelligence*, Townsville, Australia, pp. 321–335.

HILLENBRAND T., BUCH A., VOGT R. AND LÖCHNER B. [1997], 'Waldmeister: High-performance equational deduction', *Journal of Automated Reasoning* **18**(2), 265–270.

KNUTH D. AND BENDIX P. [1970], Simple word problems in universal algebras, *in* J. Leech, ed., 'Computational Problems in Abstract Algebra', Pergamon Press, Oxford, pp. 263–297.

MOSER M., IBENS O., LETZ R., STEINBACH J., GOLLER C., SCHUMANN J. AND MAYR K. [1997], 'SETHEO and E-SETHEO — the CADE-13 systems', *Journal of Automated Reasoning* **18**, 237–246.

NIEUWENHUIS R. AND RUBIO A. [1992], Basic superposition is complete, *in* 'ESOP'92', Vol. 582 of *Lecture Notes in Computer Science*, Springer Verlag, pp. 371–389.

NIEWENHUIS R. AND RUBIO A. [2000], Paramodulation-based theorem proving, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Elsevier Science and MIT Press. To appear.

NONNENGART A. AND WEIDENBACH C. [2000], Computing small clause normal forms, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Elsevier Science and MIT Press. To appear.

PASTRE D. [1989], 'MUSCADET: an automatic theorem proving system using knowledge and metaknowledge in mathematics', *Artificial Intelligence* **38**, 257–318.

RIAZANOV A. AND VORONKOV A. [1999], Vampire, *in* H. Ganzinger, ed., 'Automated Deduction—CADE-16. 16th International Conference on Automated Deduction', Lecture Notes in Artificial Intelligence, Trento, Italy, pp. 292–296.

RIAZANOV A. AND VORONKOV A. [2000], How to make your prover solve in 5 minutes what it cannot solve in one hour, *in* 'AAAI'2000'. submitted.

SCHULZ S. [1999], System abstract: E 0.3, *in* H. Ganzinger, ed., 'Automated Deduction—CADE-16. 16th International Conference on Automated Deduction', Lecture Notes in Artificial Intelligence, Trento, Italy, pp. 297–301.

SUTCLIFFE G. [2000], 'The CADE-16 ATP system competition', *Journal of Automated Reasoning* . to appear.

TAMMET T. [1997], 'Gandalf', *Journal of Automated Reasoning* **18**(2), 199–204.

TAMMET T. [1998], Towards efficient subsumption, *in* C. Kirchner and H. Kirchner, eds, 'Automated Deduction — CADE-15. 15th International Conference on Automated Deduction', Vol. 1421 of *Lecture Notes in Artificial Intelligence*, Springer Verlag, Lindau, Germany, p. ???

VORONKOV A. [1995], 'The anatomy of Vampire: Implementing bottom-up procedures with code trees', *Journal of Automated Reasoning* **15**(2), 237–265.

WEIDENBACH C. [2000], Combining superposition, sorts and splitting, *in* A. Robinson and A. Voronkov, eds, 'Handbook of Automated Reasoning', Elsevier Science and MIT Press. To appear.

WEIDENBACH C., AFSHORDEL B., BRAHM U., COHRS C., ENGEL T., KEEN E., THEOBALT C. AND TOPIC D. [1999], System description: SPASS version 1.0.0, *in* H. Ganzinger, ed., 'Automated Deduction—CADE-16. 16th International Conference on Automated Deduction', Lecture Notes in Artificial Intelligence, Trento, Italy, pp. 378–382.

WEIDENBACH C., GAEDE B. AND ROCK G. [1996], Spass & flotter. version 0.42, *in* M. McRobbie and J. Slaney, eds, 'Automated Deduction — CADE-13', Vol. 1104 of *Lecture Notes in Computer Science*, New Brunswick, NJ, USA, pp. 141–145.

A Appendix: CASC 16 and CASC $16\frac{1}{2}$: results

In this appendix we give tables with statistical information from CASC-16 and CASC-$16\frac{1}{2}$.

Table 1 presents the original CASC results in the mixed division. It is followed by Tables 2–5 corresponding to the extra argument transformation with time limits of 10, 5, 3 and 1 minutes. Table 6 presents the results obtained on the noise transformation with the time limit of 5 minutes.

Table 7 presents the original CASC results in the unit equality division. It is followed by Tables 8–10 corresponding to the extra argument transformation with time limits of 30, 15, and 5 minutes.

	E-Se	Vam	SP-C	E-C	Gan98	Gan99	Bli	Sco	Ot-M	FDP
ana002-3	-	178.5	97.1	-	-	-	-	-	-	-
ana002-4	9.3	108.8	37.0	-	76.8	163.5	-	-	-	214.1
cat005-3	108.4	0.0	0.1	0.0	115.1	37.6	0.0	-	-	0.7
cid003-1	-	-	37.8	12.6	45.4	63.8	-	95.6	36.5	-
cid003-2	89.9	-	36.8	13.4	39.0	55.5	-	47.1	0.8	-
col003-7	24.2	41.8	-	103.5	-	-	-	-	-	-
col003-8	157.2	34.4	-	79.4	-	-	-	-	-	-
col003-9	29.4	37.8	-	79.9	-	-	-	-	-	-
col003-10	160.4	37.8	-	79.7	-	-	-	-	-	-
col006-2	181.0	-	-	136.1	-	-	-	136.4	-	-
col042-2	138.6	9.9	80.6	60.9	-	-	-	90.1	-	-
col042-3	250.0	13.6	86.8	165.3	-	-	-	89.7	-	-
col044-3	-	-	-	3.5	-	-	-	91.4	-	-
col044-4	-	-	-	83.2	-	-	-	87.0	-	-
col044-5	-	-	-	83.4	-	-	-	102.8	-	-
col078-1	-	-	-	-	-	-	-	-	-	-
col081-1	27.9	68.0	10.6	2.1	44.6	38.4	-	6.6	0.7	-
geo010-1	112.9	12.8	4.4	1.8	108.7	108.4	-	85.5	126.7	-
geo065-2	110.7	0.0	0.2	0.1	5.2	5.9	2.2	111.7	21.5	-
grp050-1	45.7	3.0	26.0	99.1	9.1	11.3	5.7	105.1	29.4	-
grp051-1	72.1	3.0	15.0	178.0	108.2	-	83.8	-	-	-
grp052-1	44.8	8.7	180.6	271.4	13.1	17.6	6.2	-	-	-
grp056-1	98.2	1.2	-	7.1	-	-	3.9	-	-	-
grp062-1	41.8	3.4	-	2.0	30.1	38.9	4.5	-	-	-
grp071-1	41.4	3.4	15.9	0.9	-	-	3.3	-	-	-
grp072-1	45.8	2.9	18.4	15.6	-	-	3.8	-	-	-
grp077-1	42.7	2.4	15.9	2.1	113.6	152.6	1.0	-	98.3	-
grp078-1	41.2	0.3	1.9	0.1	3.7	5.2	0.1	201.6	38.2	-
grp083-1	45.0	3.7	72.3	7.3	266.9	-	8.5	-	-	-
grp105-1	41.8	41.3	11.4	0.1	6.1	8.3	6.9	-	-	-
grp108-1	41.3	-	3.2	0.2	17.9	24.8	26.8	-	-	-
lat005-3	-	-	-	-	-	-	-	-	-	-
lat005-6	179.2	61.3	35.5	-	55.5	-	-	-	209.9	-
lat030-1	42.9	-	-	80.4	36.5	-	6.8	-	-	-
lcl002-1	-	-	-	-	173.9	186.4	-	-	-	-
lcl017-1	107.7	38.5	64.1	32.4	239.7	257.6	192.8	-	-	-
lcl021-1	-	67.1	-	236.0	-	-	-	-	-	-
lcl064-1	7.9	52.3	19.1	-	2.0	1.9	-	-	-	-
lcl105-1	261.1	-	-	195.7	-	-	-	-	-	-
lcl109-4	103.7	45.0	-	36.7	-	-	-	-	-	-
lcl121-1	75.9	33.6	47.2	1.3	22.5	24.8	4.2	-	134.3	-
lcl124-1	76.9	-	-	1.9	-	-	-	-	-	-
lcl128-1	102.2	-	8.4	26.8	33.4	36.4	3.1	-	146.8	-
lcl191-1	-	31.4	-	-	-	-	26.0	-	274.0	-
lcl222-1	-	66.8	-	-	-	-	-	187.6	-	-
lcl223-1	-	53.2	-	-	-	53.4	-	101.5	-	-
num094-1	-	-	-	-	69.2	99.1	-	-	-	-
pla009-1	1.9	0.0	0.7	11.5	54.8	37.2	-	-	-	-
pla015-1	33.3	0.8	3.5	37.5	58.8	123.4	-	-	-	-
pla018-1	33.2	0.3	0.6	1.2	36.1	51.0	-	-	-	-
pla023-1	33.4	0.6	1.5	1.2	36.9	39.0	-	-	-	-
prv008-1	8.5	-	0.2	0.0	0.4	0.3	0.1	51.4	1.1	-
puz017-1	-	0.1	3.7	-	-	-	-	-	-	-
rng041-1	1.0	0.0	0.7	0.0	0.8	0.8	0.5	80.9	24.7	-
set012-1	0.9	-	2.4	2.8	44.4	38.3	1.6	-	-	-
set013-2	-	-	174.5	-	0.5	0.4	-	-	-	6.4
set031-4	4.4	-	10.9	-	47.6	39.1	-	-	-	-
set095-6	44.6	10.5	1.4	0.5	58.8	39.2	0.0	198.7	53.3	-
set130-6	191.8	21.6	27.9	79.3	64.1	40.0	-	-	-	-
set135-6	-	23.0	64.9	-	-	46.9	-	-	-	-
set186-6	-	-	-	-	69.3	-	-	-	-	-
set189-6	-	-	132.5	-	58.9	41.1	-	-	6.7	-
set236-6	3.4	-	100.0	-	58.8	39.3	-	-	-	-
set287-6	-	61.3	-	14.8	113.1	115.3	-	-	-	-
syn015-1	109.3	0.3	0.5	0.2	-	-	1.8	-	-	-
syn067-1	-	-	99.0	-	-	-	-	-	-	-
syn067-2	-	-	-	-	-	-	-	-	-	145.7
syn436-1	16.4	166.0	54.1	-	-	-	-	-	-	-
syn448-1	21.6	13.6	37.3	-	-	-	-	-	-	217.3
syn461-1	22.0	284.8	61.8	-	-	-	-	-	-	219.1
syn469-1	45.2	17.0	86.9	-	-	-	-	-	-	211.0
syn477-1	115.4	8.1	95.2	-	-	-	-	-	-	-
syn486-1	49.7	16.9	56.0	-	-	-	-	-	-	-
syn489-1	50.5	18.1	44.4	-	-	-	-	-	-	211.0
syn500-1	52.1	7.9	60.8	-	-	-	-	-	-	-
total	54	51	50	47	39	36	24	18	16	8

Table 1: Mixed category, original results, time limit 5 minutes

	E-Se	Vam	SP	SP-C	E	E-C	Gan98	Gan99	Bli	Sco	Ot-M
ana002-3	-	-	-	-	-	-	-	-	-	-	-
ana002-4	16.3	100.8	-	-	-	-	217.8	-	-	-	-
cat005-3	216.1	0.0	0.0	0.1	0.0	0.0	231.2	37.5	0.0	-	-
cid003-1	43.0	125.5	-	-	0.8	0.8	89.6	65.3	-	38.0	10.5
cid003-2	27.0	13.4	-	-	3.2	3.1	68.7	54.8	-	0.2	0.3
col003-10	-	-	-	-	-	-	-	-	-	-	-
col003-7	-	-	-	-	-	-	-	-	-	-	-
col003-8	-	-	-	-	-	-	-	-	-	-	-
col003-9	-	-	-	-	-	-	-	-	-	-	-
col006-2	192.3	-	-	-	245.9	238.2	-	-	-	117.0	-
col042-2	-	-	-	-	25.9	22.4	-	-	-	84.6	-
col042-3	-	-	-	-	27.3	24.9	-	-	-	80.5	-
col044-3	405.7	-	-	-	4.9	4.8	-	-	-	113.4	-
col044-4	395.6	-	-	-	129.8	121.5	-	-	-	117.2	-
col044-5	395.2	-	-	-	128.3	121.4	-	-	-	100.9	-
col078-1	-	-	-	-	-	-	-	-	-	34.1	-
col081-1	89.3	165.2	-	-	74.0	72.1	88.4	38.6	-	253.0	0.3
geo010-1	227.1	18.7	6.6	6.3	1.8	1.8	217.6	106.9	-	84.6	61.3
geo065-2	225.0	0.0	0.1	0.1	0.1	0.1	6.1	7.6	2.9	86.5	7.5
grp050-1	22.4	3.3	-	-	41.7	38.4	13.8	16.3	12.4	101.6	18.1
grp051-1	138.7	13.9	18.7	31.4	165.9	150.3	221.9	-	81.1	-	-
grp052-1	23.1	11.0	-	-	241.7	223.9	15.2	19.8	14.6	-	-
grp056-1	163.5	1.8	25.9	-	11.8	11.0	-	-	6.2	-	-
grp062-1	15.0	11.8	-	-	3.3	3.1	38.2	87.4	8.3	-	-
grp071-1	19.5	6.3	122.5	120.7	0.5	0.5	-	-	5.1	-	-
grp072-1	17.7	5.0	36.3	36.1	12.6	11.9	-	-	4.8	-	-
grp077-1	18.3	5.8	46.7	48.5	5.2	4.9	301.3	395.7	1.7	117.3	-
grp078-1	11.1	0.5	2.9	2.7	0.1	0.1	3.8	5.3	0.2	87.4	19.5
grp083-1	27.4	4.4	-	-	12.9	12.0	-	-	-	-	-
grp105-1	15.8	54.0	-	-	5.5	5.1	4.9	6.8	25.0	-	-
grp108-1	14.0	-	-	-	3.5	3.2	21.7	29.1	197.2	-	-
lat005-3	-	-	-	-	-	-	-	-	-	-	-
lat005-6	165.0	56.5	-	-	312.6	288.4	55.9	-	-	111.6	73.5
lat030-1	14.1	-	-	-	160.9	150.1	57.4	-	15.7	-	-
lcl002-1	-	-	-	-	-	-	552.6	182.5	-	-	-
lcl017-1	61.4	70.6	-	-	38.7	37.1	507.5	254.4	274.7	569.1	-
lcl021-1	343.5	110.3	-	-	289.6	276.8	-	-	-	-	-
lcl064-1	17.0	55.1	-	-	19.1	18.4	3.5	4.3	-	151.3	69.1
lcl105-1	393.7	-	-	-	325.6	320.3	-	-	-	373.5	-
lcl109-4	59.1	-	-	-	87.3	87.4	-	-	-	-	-
lcl121-1	22.7	64.6	-	-	1.9	1.8	34.3	36.6	5.9	167.0	116.8
lcl124-1	23.3	-	-	-	2.7	2.7	-	-	-	-	-
lcl128-1	21.3	63.3	-	-	0.8	0.8	51.7	51.2	144.5	164.4	109.3
lcl191-1	-	28.6	-	-	-	-	-	-	25.6	347.0	201.0
lcl222-1	-	-	-	-	207.0	-	-	-	-	-	-
lcl223-1	-	90.6	-	-	-	-	-	53.2	-	-	-
num094-1	-	-	-	-	-	-	126.6	79.3	-	331.6	195.5
pla009-1	1.9	0.1	0.9	0.8	0.4	0.7	89.2	37.0	-	-	-
pla015-1	11.0	0.9	2.9	2.6	6.1	3.3	90.2	128.8	-	-	-
pla018-1	10.7	0.3	0.8	0.8	0.3	0.2	70.3	57.3	-	-	-
pla023-1	11.0	0.6	1.9	1.7	0.4	0.4	70.0	38.2	-	-	-
prv008-1	20.3	-	0.0	0.0	0.0	0.0	0.5	0.6	5.4	565.3	0.2
puz017-1	-	0.1	-	-	-	-	-	-	-	-	-
rng041-1	0.9	0.0	0.6	0.6	0.0	0.0	0.7	0.9	0.7	94.9	11.5
set012-1	0.9	-	5.2	5.3	2.5	2.1	87.0	37.1	1.9	-	-
set013-2	-	349.6	248.0	243.3	29.4	-	0.5	0.5	-	-	-
set031-4	8.1	-	23.5	23.6	-	-	91.2	38.1	-	-	-
set095-6	107.1	6.0	113.5	115.5	21.4	5.4	67.1	35.0	16.3	-	42.1
set130-6	-	21.8	-	-	-	-	324.8	58.8	-	-	-
set135-6	-	19.2	-	-	-	-	381.7	-	-	-	-
set186-6	-	-	-	-	-	-	-	-	-	-	-
set189-6	-	-	403.2	403.1	-	-	117.4	43.0	-	-	69.9
set236-6	4.6	-	162.6	159.7	96.2	-	116.2	38.7	-	-	8.1
set287-6	393.8	33.0	-	-	-	41.2	-	-	-	-	-
syn015-1	-	0.2	0.5	0.5	0.3	0.2	-	-	1.2	-	-
syn067-1	210.0	-	129.7	128.3	322.9	294.5	-	-	-	-	-
syn067-2	-	-	-	-	-	-	-	-	-	-	-
syn436-1	18.8	134.8	77.2	92.0	-	-	-	97.5	-	-	-
syn448-1	68.6	19.2	52.2	59.3	-	-	220.1	85.7	-	-	-
syn461-1	67.3	-	71.2	82.6	-	-	-	-	-	-	-
syn469-1	29.6	17.5	92.6	107.6	-	-	-	-	-	-	-
syn477-1	208.8	16.0	87.4	97.2	-	-	-	94.3	-	-	-
syn486-1	77.3	18.0	63.5	73.1	-	-	223.8	80.8	-	-	-
syn489-1	77.6	7.4	71.7	78.6	-	-	-	-	-	-	-
syn500-1	81.1	11.8	87.7	94.2	-	-	-	185.8	-	-	-
total	53	45	31	30	47	45	39	38	23	25	18

Table 2: Mixed division, extra argument transformation, time limit 10 minutes

	E-Se	Vam	SP	SP-C	E	E-C	Gan98	Gan99	Bli	Sco	Ot-M
ana002-3	-	-	-	-	-	-	-	-	-	-	-
ana002-4	9.3	79.7	-	-	-	-	-	-	-	-	-
cat005-3	145.3	0.0	0.0	0.1	0.0	0.0	113.7	37.3	0.0	-	-
cid003-1	45.5	118.9	-	-	0.8	0.9	45.4	66.4	-	37.2	11.0
cid003-2	27.3	13.5	-	-	3.2	3.4	42.7	55.2	-	0.2	0.4
col003-10	-	-	-	-	-	-	-	-	-	-	-
col003-7	-	-	-	-	-	-	-	-	-	-	-
col003-8	-	-	-	-	-	-	-	-	-	-	-
col003-9	-	-	-	-	-	-	-	-	-	-	-
col006-2	193.2	-	-	-	243.4	240.8	-	-	-	118.8	-
col042-2	-	-	-	-	25.3	23.0	-	-	-	84.7	-
col042-3	-	-	-	-	27.3	26.0	-	-	-	78.7	-
col044-3	-	-	-	-	4.9	5.0	-	-	-	113.4	-
col044-4	-	-	-	-	127.7	129.1	-	-	-	114.4	-
col044-5	-	-	-	-	127.1	128.3	-	-	-	100.6	-
col078-1	-	-	-	-	-	-	-	-	-	33.7	-
col081-1	-	-	-	-	76.0	77.7	44.2	38.7	-	249.6	0.3
geo010-1	154.5	17.3	6.2	6.1	1.9	1.9	107.6	107.6	-	86.5	57.3
geo065-2	152.2	0.0	0.1	0.1	0.1	0.1	6.1	7.6	2.9	84.9	8.0
grp050-1	23.2	3.3	-	-	40.3	40.8	14.1	16.5	12.5	101.7	18.4
grp051-1	-	14.1	19.3	31.7	155.5	161.4	-	-	81.5	-	-
grp052-1	23.8	11.2	-	-	240.8	243.2	15.4	20.8	14.6	-	-
grp056-1	237.8	1.8	26.0	-	11.5	11.7	-	-	6.3	-	-
grp062-1	15.3	11.9	-	-	3.3	3.5	38.7	85.6	8.2	-	-
grp071-1	20.0	6.4	123.0	122.7	0.5	0.5	-	-	5.2	-	-
grp072-1	18.4	5.0	36.6	36.2	12.1	12.8	-	-	4.8	-	-
grp077-1	18.9	5.9	48.6	47.7	5.3	5.2	299.7	300.0	1.7	117.0	-
grp078-1	11.1	0.5	2.9	2.7	0.1	0.1	3.9	5.4	0.2	87.2	19.6
grp083-1	29.3	4.4	-	-	12.7	12.8	-	-	-	-	-
grp105-1	16.2	54.4	-	-	5.4	5.5	4.8	7.2	25.1	-	-
grp108-1	14.1	-	-	-	3.3	3.3	21.8	31.6	198.6	-	-
lat005-3	-	-	-	-	-	-	-	-	-	-	-
lat005-6	200.4	57.3	-	-	-	-	55.2	-	-	109.6	71.7
lat030-1	14.6	-	-	-	155.7	158.2	-	-	15.6	-	-
lcl002-1	-	-	-	-	-	-	172.7	182.9	-	-	-
lcl017-1	67.7	40.5	-	-	39.0	39.3	254.2	254.7	279.1	-	-
lcl021-1	-	80.6	-	-	286.4	286.5	-	-	-	-	-
lcl064-1	17.9	34.8	-	-	19.4	19.7	3.7	4.2	-	144.2	69.8
lcl105-1	-	-	-	-	-	-	-	-	-	-	-
lcl109-4	58.8	-	-	-	79.9	83.2	-	-	-	-	-
lcl121-1	22.7	33.7	-	-	1.9	1.9	33.4	36.5	5.6	171.5	110.2
lcl124-1	23.7	-	-	-	2.6	2.8	-	-	-	-	-
lcl128-1	21.6	33.1	-	-	0.8	0.8	59.8	49.7	142.9	164.1	107.3
lcl191-1	-	27.8	-	-	-	-	-	-	25.7	-	201.1
lcl222-1	-	-	-	-	209.4	-	-	-	-	-	-
lcl223-1	-	-	-	-	-	-	-	53.1	-	-	-
num094-1	-	-	-	-	-	-	68.8	79.9	-	-	-
pla009-1	2.0	0.1	0.9	0.9	0.5	0.8	55.3	36.5	-	-	-
pla015-1	11.0	0.9	2.9	2.6	6.1	3.4	55.7	121.2	-	-	-
pla018-1	10.8	0.3	0.8	0.8	0.3	0.3	36.4	57.1	-	-	-
pla023-1	11.0	0.6	1.9	1.8	0.4	0.4	37.3	38.3	-	-	-
prv008-1	16.5	-	0.0	0.1	0.0	0.0	0.5	0.6	5.4	-	0.2
puz017-1	-	0.1	-	-	-	-	-	-	-	-	-
rng041-1	1.0	0.0	0.6	0.6	0.0	0.0	0.8	1.0	0.7	91.4	11.0
set012-1	0.9	-	4.8	4.5	2.2	1.6	43.2	37.6	1.8	-	-
set013-2	-	262.4	248.3	241.6	29.2	-	0.4	0.5	-	-	-
set031-4	8.5	-	23.2	23.1	-	-	46.9	38.7	-	-	-
set095-6	86.7	6.2	114.5	116.6	20.8	5.6	42.6	35.2	16.4	-	43.5
set130-6	-	20.2	-	-	-	-	162.4	57.9	-	-	-
set135-6	-	19.7	-	-	-	-	-	-	-	-	-
set186-6	-	-	-	-	-	-	-	-	-	-	-
set189-6	-	-	-	-	-	-	58.5	45.0	-	-	72.1
set236-6	4.8	-	162.8	161.7	94.7	-	58.7	38.9	-	-	8.2
set287-6	-	33.1	-	-	-	44.3	-	-	-	-	-
syn015-1	145.4	0.2	0.4	0.5	0.3	0.2	-	-	1.2	-	-
syn067-1	181.0	-	120.7	125.0	298.1	-	-	-	-	-	-
syn067-2	-	-	-	-	-	-	-	-	-	-	-
syn436-1	18.5	131.8	75.4	90.5	-	-	-	95.6	-	-	-
syn448-1	48.5	18.8	51.7	58.6	-	-	-	85.7	-	-	-
syn461-1	45.7	-	69.9	80.8	-	-	-	-	-	-	-
syn469-1	28.4	17.3	92.7	104.9	-	-	-	-	-	-	-
syn477-1	117.3	15.9	85.8	95.9	-	-	-	95.6	-	-	-
syn486-1	56.6	17.7	63.8	72.5	-	-	-	81.5	-	-	-
syn489-1	55.8	7.1	68.9	77.2	-	-	-	-	-	-	-
syn500-1	60.6	11.6	85.1	92.5	-	-	-	183.8	-	-	-
total	46	43	30	29	45	42	33	38	23	20	17

Table 3: Mixed division, extra argument transformation, time limit 5 minutes

	E-Se	Vam	SP	SP-C	E	E-C	Gan98	Gan99	Bli	Sco	Ot-M
ana002-3	-	-	-	-	-	-	-	-	-	-	-
ana002-4	6.3	-	-	-	-	-	-	-	-	-	-
cat005-3	102.8	0.0	0.0	0.1	0.0	0.0	65.0	36.9	0.0	-	-
cid003-1	44.7	102.2	-	-	1.0	0.8	32.1	66.1	-	36.6	11.6
cid003-2	-	14.0	-	-	3.3	3.1	26.3	53.7	-	0.2	0.4
col003-10	-	-	-	-	-	-	-	-	-	-	-
col003-7	-	-	-	-	-	-	-	-	-	-	-
col003-8	-	-	-	-	-	-	-	-	-	-	-
col003-9	-	-	-	-	-	-	-	-	-	-	-
col006-2	-	-	-	-	-	-	-	-	-	116.5	-
col042-2	-	-	-	-	26.4	22.8	-	-	-	84.6	-
col042-3	-	-	-	-	27.6	25.8	-	-	-	79.4	-
col044-3	-	-	-	-	5.1	4.8	-	-	-	113.5	-
col044-4	-	-	-	-	129.6	126.0	-	-	-	112.8	-
col044-5	-	-	-	-	133.0	124.9	-	-	-	100.7	-
col078-1	-	-	-	-	-	-	-	-	-	33.4	-
col081-1	-	-	-	-	78.5	74.9	25.3	39.7	-	-	0.2
geo010-1	108.6	18.1	6.4	6.3	1.9	1.8	60.5	107.0	-	86.0	62.1
geo065-2	108.6	0.0	0.1	0.1	0.1	0.1	6.5	7.6	2.9	85.5	8.0
grp050-1	23.3	3.5	-	-	41.6	40.8	14.5	17.5	13.2	102.2	18.9
grp051-1	-	14.7	20.3	31.8	171.1	158.3	-	-	84.5	-	-
grp052-1	23.4	11.6	-	-	-	-	16.2	20.4	15.1	-	-
grp056-1	-	1.9	28.9	-	12.6	12.0	-	-	6.8	-	-
grp062-1	15.2	12.4	-	-	3.4	3.3	63.1	87.9	8.4	-	-
grp071-1	19.7	6.7	127.4	128.6	0.6	0.6	-	-	5.5	-	-
grp072-1	17.9	5.2	38.8	37.9	12.9	12.3	-	-	5.0	-	-
grp077-1	20.4	6.4	50.8	51.2	5.7	5.1	180.0	180.0	1.7	115.9	-
grp078-1	11.2	0.5	3.0	2.9	0.1	0.2	4.1	5.2	0.2	86.4	22.3
grp083-1	29.0	4.7	-	-	13.4	12.5	-	-	-	-	-
grp105-1	16.1	53.4	-	-	5.7	5.5	5.3	7.4	26.4	-	-
grp108-1	14.1	-	-	-	3.3	3.2	22.7	29.4	-	-	-
lat005-3	-	-	-	-	-	-	-	-	-	-	-
lat005-6	175.7	58.5	-	-	-	-	-	-	-	109.3	-
lat030-1	14.3	-	-	-	168.5	154.5	-	-	16.0	-	-
lcl002-1	-	-	-	-	-	-	84.3	179.0	-	-	-
lcl017-1	63.6	29.1	-	-	40.0	38.1	159.4	-	-	-	-
lcl021-1	-	68.6	-	-	-	-	-	-	-	-	-
lcl064-1	17.2	22.5	-	-	20.7	19.3	3.7	4.5	-	149.5	-
lcl105-1	-	-	-	-	-	-	-	-	-	-	-
lcl109-4	56.3	-	-	-	85.0	79.2	-	-	-	-	-
lcl121-1	-	21.9	-	-	1.9	1.9	28.6	37.3	5.9	162.8	-
lcl124-1	23.5	-	-	-	2.9	2.7	-	-	-	-	-
lcl128-1	21.4	21.2	-	-	0.9	0.8	32.2	49.4	147.0	154.3	111.5
lcl191-1	-	29.2	-	-	-	-	-	-	26.5	-	-
lcl222-1	-	-	-	-	-	-	-	-	-	-	-
lcl223-1	-	-	-	-	-	-	-	53.1	-	-	-
num094-1	-	-	-	-	-	-	51.1	91.8	-	-	-
pla009-1	2.0	0.1	0.9	0.9	0.4	0.7	67.0	36.4	-	-	-
pla015-1	10.8	1.1	2.9	2.7	6.3	3.4	69.8	127.6	-	-	-
pla018-1	10.7	0.3	0.8	0.8	0.3	0.2	29.4	56.8	-	-	-
pla023-1	11.0	0.5	1.8	1.8	0.4	0.4	30.2	38.2	-	-	-
prv008-1	10.6	-	0.0	0.0	0.0	0.0	0.5	0.6	5.5	-	0.2
puz017-1	-	0.1	-	-	-	-	-	-	-	-	-
rng041-1	0.9	0.0	0.6	0.6	0.0	0.0	0.8	0.9	0.7	95.5	10.9
set012-1	0.9	-	5.2	4.9	2.5	1.5	24.9	36.5	1.9	-	-
set013-2	-	-	-	-	30.0	-	0.4	0.5	-	-	-
set031-4	8.1	-	23.8	23.5	-	-	29.1	38.5	-	-	-
set095-6	80.4	6.3	122.6	124.6	21.2	5.7	22.3	33.2	17.2	-	-
set130-6	-	21.0	-	-	-	-	96.0	57.2	-	-	-
set135-6	-	20.3	-	-	-	-	-	-	-	-	-
set186-6	-	-	-	-	-	-	-	-	-	-	-
set189-6	-	-	-	-	-	-	34.9	43.8	-	-	74.2
set236-6	4.9	-	173.4	173.0	99.8	-	34.4	38.8	-	-	8.9
set287-6	-	34.1	-	-	-	42.9	-	-	-	-	-
syn015-1	104.1	0.2	0.5	0.5	0.3	0.2	-	-	1.3	-	-
syn067-1	-	-	122.8	128.8	-	-	-	-	-	-	-
syn067-2	-	-	-	-	-	-	-	-	-	-	-
syn436-1	13.2	136.7	77.8	95.3	-	-	-	97.3	-	-	-
syn448-1	32.4	22.3	53.9	64.5	-	-	-	87.3	-	-	-
syn461-1	32.5	-	72.0	84.5	-	-	-	-	-	-	-
syn469-1	34.1	18.3	96.0	107.6	-	-	-	-	-	-	-
syn477-1	76.9	16.7	88.1	97.6	-	-	-	95.7	-	-	-
syn486-1	41.8	18.3	64.9	75.7	-	-	-	82.2	-	-	-
syn489-1	41.6	7.3	72.0	79.4	-	-	-	-	-	-	-
syn500-1	46.2	12.7	87.0	96.7	-	-	-	180.0	-	-	-
total	41	41	29	28	40	39	32	37	21	19	12

Table 4: Mixed division, extra argument transformation, time limit 3 minutes

	E-Se	Vam	SP	SP-C	E	E-C	Gan98	Gan99	Bli	Sco	Ot-M
ana002-3	-	-	-	-	-	-	-	-	-	-	-
ana002-4	3.4	-	-	-	-	-	-	-	-	-	-
cat005-3	36.2	0.0	0.0	0.1	0.0	0.0	18.4	37.4	0.0	-	-
cid003-1	37.8	-	-	-	0.9	0.9	9.9	-	-	37.3	10.5
cid003-2	21.1	12.5	-	-	3.3	3.3	8.5	55.0	-	0.2	0.3
col003-10	-	-	-	-	-	-	-	-	-	-	-
col003-7	-	-	-	-	-	-	-	-	-	-	-
col003-8	-	-	-	-	-	-	-	-	-	-	-
col003-9	-	-	-	-	-	-	-	-	-	-	-
col006-2	-	-	-	-	-	-	-	-	-	-	-
col042-2	-	-	-	-	25.9	23.1	-	-	-	-	-
col042-3	-	-	-	-	27.1	26.8	-	-	-	-	-
col044-3	-	-	-	-	4.9	4.9	-	-	-	-	-
col044-4	-	-	-	-	-	-	-	-	-	-	-
col044-5	-	-	-	-	-	-	-	-	-	-	-
col078-1	-	-	-	-	-	-	-	-	-	34.3	-
col081-1	-	-	-	-	-	-	8.0	39.9	-	-	0.2
geo010-1	41.6	17.0	6.2	6.0	1.8	1.8	20.4	-	-	-	-
geo065-2	39.8	0.0	0.1	0.2	0.0	0.1	11.3	7.5	2.9	-	7.4
grp050-1	14.9	3.4	-	-	40.3	41.0	31.0	16.6	12.5	-	18.4
grp051-1	-	14.2	18.9	31.8	-	-	-	-	-	-	-
grp052-1	13.0	11.5	-	-	-	-	-	20.8	14.8	-	-
grp056-1	-	1.8	26.9	-	11.9	12.0	-	-	6.4	-	-
grp062-1	10.2	11.9	-	-	3.3	3.3	-	-	8.1	-	-
grp071-1	9.4	6.4	-	-	0.5	0.5	-	-	5.2	-	-
grp072-1	13.1	5.1	37.6	36.8	12.6	12.8	-	-	5.0	-	-
grp077-1	11.7	6.0	47.7	47.9	5.2	5.2	60.0	60.0	1.7	-	-
grp078-1	9.0	0.5	2.9	2.7	0.2	0.1	4.0	5.7	0.2	-	20.0
grp083-1	18.0	4.5	-	-	12.9	13.1	-	-	-	-	-
grp105-1	11.1	39.8	-	-	5.5	5.4	4.9	6.9	26.4	-	-
grp108-1	10.0	-	-	-	3.4	3.4	10.9	29.2	-	-	-
lat005-3	-	-	-	-	-	-	-	-	-	-	-
lat005-6	-	42.7	-	-	-	-	-	-	-	-	-
lat030-1	11.9	-	-	-	-	-	-	-	15.7	-	-
lcl002-1	-	-	-	-	-	-	-	-	-	-	-
lcl017-1	57.5	15.9	-	-	38.5	39.2	-	-	-	-	-
lcl021-1	-	-	-	-	-	-	-	-	-	-	-
lcl064-1	13.1	10.4	-	-	19.6	19.2	3.6	4.2	-	-	-
lcl105-1	-	-	-	-	-	-	-	-	-	-	-
lcl109-4	52.3	-	-	-	-	-	-	-	-	-	-
lcl121-1	16.5	9.5	-	-	1.9	1.9	8.2	36.4	5.7	-	-
lcl124-1	17.2	-	-	-	2.8	2.8	-	-	-	-	-
lcl128-1	15.4	9.8	-	-	0.9	0.8	9.0	49.7	-	-	-
lcl191-1	-	16.8	-	-	-	-	-	-	25.9	-	-
lcl222-1	-	-	-	-	-	-	-	-	-	-	-
lcl223-1	-	-	-	-	-	-	-	53.0	-	-	-
num094-1	-	-	-	-	-	-	17.1	-	-	-	-
pla009-1	2.0	0.1	0.9	0.8	0.4	0.7	24.6	36.3	-	-	-
pla015-1	6.8	0.9	2.8	2.6	6.2	3.4	-	-	-	-	-
pla018-1	6.7	0.3	0.8	0.8	0.3	0.2	10.9	57.2	-	-	-
pla023-1	6.9	0.5	1.9	1.7	0.4	0.4	11.6	38.3	-	-	-
prv008-1	5.0	-	0.0	0.0	0.0	0.0	0.5	0.6	5.4	-	0.2
puz017-1	-	0.1	-	-	-	-	-	-	-	-	-
rng041-1	1.0	0.0	0.6	0.6	0.0	0.0	0.7	0.9	0.7	-	10.2
set012-1	0.9	-	4.9	4.5	2.4	1.6	7.9	37.6	1.8	-	-
set013-2	-	-	-	-	29.9	-	0.4	0.5	-	-	-
set031-4	-	-	23.3	23.0	-	-	10.6	38.7	-	-	-
set095-6	49.0	6.0	-	-	21.1	5.8	10.8	35.2	16.4	-	-
set130-6	-	19.8	-	-	-	-	30.2	58.2	-	-	-
set135-6	-	19.2	-	-	-	-	-	-	-	-	-
set186-6	-	-	-	-	-	-	-	-	-	-	-
set189-6	-	-	-	-	-	-	10.6	44.3	-	-	-
set236-6	18.7	-	-	-	-	-	10.7	38.8	-	-	7.7
set287-6	-	-	-	-	-	43.4	-	-	-	-	-
syn015-1	36.1	0.2	0.4	0.5	0.3	0.2	-	-	1.2	-	-
syn067-1	-	-	-	-	-	-	-	-	-	-	-
syn067-2	-	-	-	-	-	-	-	-	-	-	-
syn436-1	28.5	-	-	-	-	-	-	-	-	-	-
syn448-1	21.2	19.1	52.2	57.9	-	-	-	-	-	-	-
syn461-1	20.3	-	-	-	-	-	-	-	-	-	-
syn469-1	36.7	17.3	-	-	-	-	-	-	-	-	-
syn477-1	38.1	16.1	-	-	-	-	-	-	-	-	-
syn486-1	-	17.7	-	-	-	-	-	-	-	-	-
syn489-1	-	7.1	-	-	-	-	-	-	-	-	-
syn500-1	-	11.6	-	-	-	-	-	-	-	-	-
total	38	37	18	17	33	33	27	26	19	3	9

Table 5: Mixed division, extra argument transformation, time limit 1 minutes

	E-Se	Vam	SP	SP-C	E	E-C	Gan98	Gan99	Bli	Sco	Ot-M
ana002-3	-	-	-	-	-	-	-	-	-	-	-
ana002-4	-	-	-	-	-	-	-	197.9	-	-	-
cat005-3	144.0	0.1	0.0	0.1	0.0	0.0	112.7	37.3	0.0	-	-
cid003-1	161.4	107.9	-	-	2.0	1.7	47.2	61.0	-	37.1	14.3
cid003-2	182.9	42.1	-	-	17.2	16.3	42.2	52.8	-	0.2	0.4
col003-10	-	-	-	-	-	-	-	-	-	-	-
col003-7	-	-	-	-	-	-	-	-	-	-	-
col003-8	-	-	-	-	-	-	-	-	-	-	-
col003-9	-	-	-	-	-	-	-	-	-	-	-
col006-2	-	-	-	-	-	-	-	27.9	-	159.0	-
col042-2	-	-	-	-	-	-	-	-	-	124.4	-
col042-3	-	-	-	-	-	-	34.9	52.6	-	93.1	-
col044-3	-	-	-	-	-	-	-	-	-	-	-
col044-4	-	-	-	-	-	-	-	-	-	102.9	-
col044-5	-	-	-	-	-	-	-	-	-	95.4	-
col078-1	-	-	-	-	-	-	-	-	-	41.1	-
col081-1	274.7	-	-	-	78.1	73.6	41.6	25.8	-	-	0.4
geo010-1	153.1	13.9	6.9	6.6	2.0	1.9	107.5	106.1	-	86.0	68.8
geo065-2	152.2	0.0	0.1	0.1	0.1	0.1	78.6	25.9	3.1	85.1	7.4
grp050-1	170.4	4.1	-	-	-	25.8	8.4	13.1	13.9	102.8	19.2
grp051-1	228.2	15.6	-	44.9	50.3	50.0	-	-	77.5	165.6	-
grp052-1	-	12.8	-	-	-	272.4	12.2	15.8	12.9	108.7	161.7
grp056-1	-	1.9	-	-	85.5	32.9	-	-	187.6	-	-
grp062-1	155.9	7.5	-	-	8.3	12.6	66.6	173.3	8.2	-	-
grp071-1	143.7	7.0	108.9	112.8	4.7	2.6	-	-	6.0	-	-
grp072-1	160.7	5.4	28.7	28.8	1.1	2.9	-	-	6.9	101.4	24.9
grp077-1	148.3	6.7	57.9	59.2	8.3	4.0	32.9	43.3	2.2	123.3	24.9
grp078-1	142.7	0.5	5.9	5.9	0.4	0.3	4.0	5.4	0.2	87.0	7.7
grp083-1	209.8	9.3	-	-	204.8	41.2	-	-	-	-	-
grp105-1	144.0	56.5	12.0	9.2	0.9	0.8	9.1	13.8	26.8	-	77.5
grp108-1	145.2	-	-	-	1.4	1.1	24.2	20.9	128.3	162.9	35.3
lat005-3	-	-	-	-	-	-	-	-	-	-	-
lat005-6	-	61.9	-	-	-	-	-	-	-	-	54.4
lat030-1	-	-	-	-	-	-	-	-	85.8	-	-
lcl002-1	-	280.9	-	-	-	-	299.0	211.7	9.8	-	-
lcl017-1	-	9.4	-	-	-	-	171.7	165.2	23.5	-	-
lcl021-1	-	36.0	-	-	-	-	-	-	-	-	-
lcl064-1	20.6	4.7	-	-	-	-	5.1	6.1	1.9	-	101.7
lcl105-1	-	-	-	-	-	-	-	-	-	-	-
lcl109-4	-	-	-	-	-	-	-	-	-	-	-
lcl121-1	170.5	4.0	-	-	26.1	26.0	23.1	21.5	-	-	168.4
lcl124-1	-	-	-	-	-	-	-	-	-	-	-
lcl128-1	283.1	5.7	-	-	222.0	213.4	26.9	22.2	-	-	25.7
lcl191-1	-	-	-	-	-	-	-	-	-	-	-
lcl222-1	-	-	-	-	-	-	-	-	-	-	-
lcl223-1	-	-	-	-	-	-	43.9	37.8	-	-	-
num094-1	-	-	-	-	-	-	69.2	89.9	-	-	-
pla009-1	45.3	-	1.0	0.9	-	-	28.5	26.5	1.2	-	-
pla015-1	45.6	-	3.0	2.8	-	-	55.9	68.2	1.2	-	-
pla018-1	45.3	-	0.9	0.9	-	-	32.2	44.3	0.9	-	-
pla023-1	46.0	-	1.4	1.4	-	-	33.3	26.5	0.5	-	-
prv008-1	21.3	-	0.0	0.0	0.1	0.0	1.0	0.7	5.6	105.0	0.2
puz017-1	-	0.1	-	-	-	-	-	-	-	-	-
rng041-1	1.0	0.0	0.6	0.6	0.1	0.0	0.8	0.9	0.9	-	3.8
set012-1	45.6	-	5.3	5.1	0.8	0.8	42.8	36.8	-	-	-
set013-2	-	-	261.1	251.8	-	-	133.4	141.7	-	-	-
set031-4	8.3	-	21.2	21.2	-	-	47.4	38.2	-	-	-
set095-6	87.7	6.2	130.3	124.8	21.0	6.0	56.3	34.0	17.5	-	45.5
set130-6	-	21.1	-	-	-	-	161.4	57.1	-	-	-
set135-6	-	20.9	-	-	-	-	-	-	-	-	-
set186-6	-	-	-	-	-	-	-	-	-	-	-
set189-6	-	-	-	-	-	-	58.2	44.6	-	-	77.0
set236-6	4.7	-	167.4	172.4	80.8	-	58.3	39.0	-	-	8.9
set287-6	-	34.2	-	-	-	44.8	-	-	-	-	-
syn015-1	144.7	0.2	0.7	0.6	0.3	0.3	-	-	1.5	-	-
syn067-1	-	-	138.9	146.4	-	-	-	-	-	-	-
syn067-2	-	-	-	-	-	-	-	-	-	-	-
syn436-1	-	-	128.4	142.9	-	-	-	-	-	-	-
syn448-1	-	-	58.5	64.0	-	-	73.8	93.2	-	-	-
syn461-1	-	-	76.8	87.1	-	-	-	-	-	-	-
syn469-1	-	-	88.2	101.7	-	-	-	-	-	-	-
syn477-1	-	-	195.7	212.8	-	-	73.2	87.5	-	-	-
syn486-1	-	-	64.5	74.3	-	-	-	-	-	-	-
syn489-1	-	-	87.6	96.7	-	-	-	-	-	-	-
syn500-1	-	-	85.8	89.5	-	-	-	-	-	-	-
total	30	30	29	30	24	26	36	38	25	18	21

Table 6: Mixed category, noise transformation, time limit 5 minutes

	Wal99	Wal98	Sco	E-C	SP-C	Fie	Ot-M	E-Se	Gan99	Vam	Bli
col003-1	1.4	-	90.2	31.3	-	-	-	-	-	-	-
col006-5	0.0	2.1	104.5	296.9	-	-	-	-	-	-	-
col006-6	0.0	0.0	-	-	-	-	-	-	-	-	-
col006-7	0.0	0.5	-	-	-	-	-	-	-	-	-
col042-1	16.6	-	-	2.9	-	-	-	108.5	-	-	-
col044-6	0.0	0.0	11.1	-	-	-	-	-	-	-	-
col044-7	0.0	0.0	103.4	-	-	-	-	-	-	-	-
col044-8	0.0	0.0	115.2	-	-	-	-	-	-	-	-
col044-9	0.0	0.0	102.9	-	-	-	-	-	-	-	-
col066-1	1.3	1.2	-	155.4	-	-	-	41.2	-	-	-
grp024-5	15.0	54.5	-	-	-	-	-	-	-	-	-
grp164-1	236.0	-	-	-	-	-	-	-	-	-	-
grp164-2	236.0	-	-	-	-	-	-	-	-	-	-
grp185-1	0.6	0.6	-	-	0.2	-	-	-	76.3	-	-
grp185-3	0.1	0.1	104.7	-	0.9	0.0	-	-	53.6	-	-
lat022-1	11.6	106.6	185.9	246.3	-	175.0	-	-	-	-	-
lat023-1	11.9	48.7	150.5	86.3	-	-	-	149.6	-	-	-
lcl109-2	2.0	2.1	-	283.9	-	91.3	-	-	188.5	27.8	61.9
rng019-6	0.0	0.0	2.9	-	0.6	63.6	0.9	-	-	-	-
rng021-6	0.0	0.0	53.2	-	0.7	-	2.1	-	-	-	-
rng025-4	1.2	-	98.8	-	1.6	-	32.2	-	-	-	-
rng025-6	0.1	3.1	-	42.3	1.1	-	-	-	-	-	-
rng026-6	0.0	9.7	-	-	-	-	-	-	-	-	-
rng027-8	22.0	-	-	-	-	-	-	-	-	-	-
rng028-5	25.6	-	-	-	-	-	-	-	-	-	-
rng028-8	25.5	-	-	-	-	-	-	-	-	-	-
rng029-5	17.9	-	-	-	-	-	-	-	-	-	-
rng029-6	18.0	-	-	-	-	-	-	-	-	-	-
rng035-7	36.1	-	-	288.6	-	-	-	-	-	-	-
rob005-1	2.5	3.0	114.5	163.9	219.4	60.0	143.4	-	-	-	-
total	30	19	13	10	7	5	4	3	3	1	1

Table 7: Unit equality division, original results, time limit 5 minutes

	Wal99	Wal98	Sco	E	E-C	SP	SP-C	Fie	Ot-M	E-Se	Gan99	Vam	Bli
col003-1	17.3	-	72.8	47.7	46.0	-	-	-	-	-	-	-	-
col006-5	-	14.0	-	578.3	560.2	-	-	-	-	-	-	-	-
col006-6	-	0.0	-	-	-	-	-	-	-	-	-	-	-
col006-7	-	0.7	-	-	-	-	-	-	-	-	-	-	-
col042-1	-	-	-	-	4.1	-	-	-	-	-	-	-	-
col044-6	-	0.0	11.5	-	-	-	-	-	-	-	-	868.1	-
col044-7	-	0.0	111.3	-	-	-	-	-	-	-	-	811.2	-
col044-8	-	0.0	116.3	-	-	-	-	-	-	-	-	490.8	-
col044-9	-	0.0	108.5	-	-	-	-	-	-	-	-	-	-
col066-1	-	-	-	730.3	602.2	-	-	-	-	-	-	-	-
grp024-5	90.9	112.7	-	170.8	159.3	-	-	-	-	699.8	-	-	-
grp164-1	1050.2	1289.2	-	-	-	-	-	-	-	-	-	-	-
grp164-2	1114.2	1316.6	-	-	-	-	-	-	-	-	-	-	-
grp185-1	-	0.2	72.2	-	-	-	-	-	-	-	73.6	-	-
grp185-3	-	1.4	179.5	-	-	-	-	0.0	301.5	-	52.8	-	-
lat022-1	89.3	111.2	123.4	378.4	367.9	-	-	313.8	678.7	-	-	353.3	-
lat023-1	61.5	76.8	113.9	137.7	130.9	-	-	281.7	269.9	94.9	-	-	-
lcl109-2	1.6	1.9	1374.7	16.4	16.2	-	-	1357.8	-	26.5	50.9	78.3	101.2
rng019-6	-	-	8.9	-	-	-	-	65.0	0.5	-	-	-	-
rng021-6	-	-	8.7	-	-	-	-	-	8.3	-	-	-	-
rng025-4	-	-	-	-	-	-	-	-	14.4	-	-	-	-
rng025-6	704.2	872.7	484.1	1568.4	1520.7	-	-	-	-	513.9	-	-	-
rng026-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng027-8	-	-	-	-	-	-	-	-	-	-	-	-	-
rng028-5	-	-	-	-	-	-	-	-	-	-	-	-	-
rng028-8	-	-	-	-	-	-	-	-	-	-	-	-	-
rng029-5	-	-	-	-	-	-	-	-	-	-	-	-	-
rng029-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng035-7	800.4	1033.7	-	1163.0	1037.8	-	-	-	-	-	-	-	-
rob005-1	67.1	58.3	123.0	-	-	-	-	482.2	77.5	-	-	-	-
total	10	18	14	9	10	0	0	6	7	4	3	5	1

Table 8: Unit equality division, extra argument transformation, time limit 30 minutes

	Wal99	Wal98	Sco	E	E-C	SP	SP-C	Fie	Ot-M	E-Se	Gan99	Vam	Bli
col003-1	16.6	-	67.4	46.6	45.7	-	-	-	-	-	-	-	-
col006-5	-	15.1	-	575.2	570.2	-	-	-	-	-	-	-	-
col006-6	-	0.0	-	-	-	-	-	-	-	-	-	-	-
col006-7	-	0.7	-	-	-	-	-	-	-	-	-	-	-
col042-1	-	-	-	-	4.1	-	-	-	-	-	-	-	-
col044-6	-	0.0	11.5	-	-	-	-	-	-	-	-	-	-
col044-7	-	0.0	116.1	-	-	-	-	-	-	-	-	-	-
col044-8	-	0.0	120.7	-	-	-	-	-	-	-	-	-	-
col044-9	-	0.0	113.7	-	-	-	-	-	-	-	-	-	-
col066-1	-	-	-	711.4	609.9	-	-	-	-	-	-	-	-
grp024-5	90.7	113.0	-	167.3	160.8	-	-	-	-	681.7	-	-	-
grp164-1	-	-	-	-	-	-	-	-	-	-	-	-	-
grp164-2	-	-	-	-	-	-	-	-	-	-	-	-	-
grp185-1	-	0.1	77.2	-	-	-	-	-	-	-	74.8	-	-
grp185-3	-	1.3	183.9	-	-	-	-	0.0	-	-	53.0	-	-
lat022-1	88.3	110.3	125.9	372.0	363.3	-	-	315.1	-	-	-	-	-
lat023-1	60.1	77.2	103.0	144.4	134.9	-	-	277.6	265.5	108.9	-	-	-
lcl109-2	1.5	1.8	-	16.5	16.3	-	-	-	-	28.8	51.8	76.7	98.9
rng019-6	-	-	8.8	-	-	-	-	64.8	0.3	-	-	-	-
rng021-6	-	-	8.7	-	-	-	-	-	8.0	-	-	-	-
rng025-4	-	-	-	-	-	-	-	-	14.0	-	-	-	-
rng025-6	694.9	859.2	528.8	-	-	-	-	-	-	515.6	-	-	-
rng026-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng027-8	-	-	-	-	-	-	-	-	-	-	-	-	-
rng028-5	-	-	-	-	-	-	-	-	-	-	-	-	-
rng028-8	-	-	-	-	-	-	-	-	-	-	-	-	-
rng029-5	-	-	-	-	-	-	-	-	-	-	-	-	-
rng029-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng035-7	790.4	-	-	-	-	-	-	-	-	-	-	-	-
rob005-1	67.3	59.8	128.9	-	-	-	-	487.2	77.0	-	-	-	-
total	8	15	13	7	8	0	0	5	5	4	3	1	1

Table 9: Unit equality division, extra argument transformation, time limit 15 minutes

	Wal99	Wal98	Sco	E	E-C	SP	SP-C	Fie	Ot-M	E-Se	Gan99	Vam	Bli
col003-1	16.9	-	66.1	47.8	44.7	-	-	-	-	-	-	-	-
col006-5	-	14.4	-	-	-	-	-	-	-	-	-	-	-
col006-6	-	0.0	-	-	-	-	-	-	-	-	-	-	-
col006-7	-	0.7	-	-	-	-	-	-	-	-	-	-	-
col042-1	-	-	-	-	4.1	-	-	-	-	-	-	-	-
col044-6	-	0.0	11.4	-	-	-	-	-	-	-	-	-	-
col044-7	-	0.0	109.6	-	-	-	-	-	-	-	-	-	-
col044-8	-	0.0	115.9	-	-	-	-	-	-	-	-	-	-
col044-9	-	0.0	108.4	-	-	-	-	-	-	-	-	-	-
col066-1	-	-	-	-	-	-	-	-	-	-	-	-	-
grp024-5	88.7	111.7	-	163.6	163.9	-	-	-	-	-	-	-	-
grp164-1	-	-	-	-	-	-	-	-	-	-	-	-	-
grp164-2	-	-	-	-	-	-	-	-	-	-	-	-	-
grp185-1	-	0.2	72.5	-	-	-	-	-	-	-	76.2	-	-
grp185-3	-	1.4	180.2	-	-	-	-	0.0	-	-	52.7	-	-
lat022-1	87.6	111.5	123.6	-	-	-	-	-	-	-	-	-	-
lat023-1	60.8	79.5	114.3	142.3	135.9	-	-	278.8	-	95.9	-	-	-
lcl109-2	1.5	1.8	-	17.1	17.1	-	-	-	-	26.6	54.1	75.8	101.1
rng019-6	-	-	8.9	-	-	-	-	63.7	0.4	-	-	-	-
rng021-6	-	-	8.8	-	-	-	-	-	8.5	-	-	-	-
rng025-4	-	-	-	-	-	-	-	-	13.6	-	-	-	-
rng025-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng026-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng027-8	-	-	-	-	-	-	-	-	-	-	-	-	-
rng028-5	-	-	-	-	-	-	-	-	-	-	-	-	-
rng028-8	-	-	-	-	-	-	-	-	-	-	-	-	-
rng029-5	-	-	-	-	-	-	-	-	-	-	-	-	-
rng029-6	-	-	-	-	-	-	-	-	-	-	-	-	-
rng035-7	-	-	-	-	-	-	-	-	-	-	-	-	-
rob005-1	64.4	58.3	129.0	-	-	-	-	-	77.0	-	-	-	-
total	6	14	12	4	5	0	0	3	4	2	3	1	1

Table 10: Unit equality division, extra argument transformation, time limit 5 minutes

	CASC	extra arg.				noise
prover\time limit	5	10	5	3	1	5
E-Setheo	54	53	46	41	38	30
Vampire	51	45	43	41	37	30
SPASS-CASC	50	30	29	28	17	30
E-CASC	47	45	42	39	33	26
Gandalf 98	39	39	33	32	27	36
Gandalf 99	36	38	38	37	26	38
Bliksem	24	23	23	21	19	25
Scott	18	25	20	19	3	18
Otter-MACE	16	18	17	12	9	21

Table 11: Mixed category provers

	CASC	extra arg.		
prover\time limit	5	30	15	5
Waldmeister 99	30	10	8	6
Waldmeister 98	19	18	15	14
Scott	13	14	13	12
E-CASC	10	10	8	5
SPASS-CASC	7	0	0	0
Fiesta	5	6	5	3
Otter-MACE	4	7	5	4
E-Setheo	3	4	4	2
Gandalf 99	3	3	3	3
Vampire	1	5	1	1
Bliksem	1	1	1	1

Table 12: Unit equality category provers

Index

Symbols

$\succ_{kb}$ — Knuth Bendix ordering 7
Σ 4
Σ^+ 6
$|\ t\ |$ — weight of t 7

A

arg 4
arity 4

C

$\mathcal{C}$ 4
c_f 4

E

E-SETHEO 9

F

$\mathcal{F}$ 4

K

Knuth-Bendix ordering 7

L

lexicographic path ordering 6

P

paramodulant 5
paramodulation 5
precedence relation 6

S

scheduling 10

T

T 4
time-slicing 10

W

w — weight function 7
weight 7
weight function 6