Closing the Performance Gap between Doubles and Rationals for Octagons

Aziem Chawdhary and Andy King

University of Kent, Canterbury, CT2 7NF, UK

Abstract. Octagons have enduring appeal because their domain operations are simple, readily mapping to for-loops which apply max, min and sum to the entries of a Difference Bound Matrix (DBM). In the quest for efficiency, arithmetic is often realised with double-precision floating-point, albeit at the cost of the certainty provided by arbitrary-precision rationals. In this paper we show how Compact DBMs (CoDBMs), which has recently been proposed as a memory refinement for DBMs, enable arithmetic calculation to be short-circuited in various domain operations. We also show how comparisons can be avoided by changing the tables which underpin CoDBMs. From the perspective of implementation, the optimisations are attractive because they too are conceptually simple, following the ethos of Octagons. Yet they halve the running time on rationals, putting CoDBMs on rationals on a par with DBMs on doubles.

1 Introduction

The dominating arithmetic operations for Difference Bound Matrices (DBMs) are addition and comparison. The speed of these operations for double-precision floating-point arithmetic is comparable with that of long integer arithmetic for modern 64-bit desktop processors, hence the trend to work with floating-point rather than idealised arithmetic, even though the latter is arguably more attractive for verification. The problem is not just one of speed: arbitrary-precision rational numbers, as supported by the GNU multiple precision (GMP) library, require at least 24 bytes to store each entry of a DBM, whereas an IEEE 754-1983 double occupies exactly 8 bytes.

Recent progress has been made on reducing space requirements by observing the DBM entries are frequently repeated [7]. This leads to a factored representation for a DBM [7] in which the entries in the matrix are identifiers for the rationals rather than the rationals themselves. The idea is to interpret matrix entries using a table which maps each identifier to its corresponding rational; a second table is used for searching for the (unique) identifier for a given rational. The first table is used for reading a matrix and the second is used for writing to a matrix; both tables are shared across all matrices. Since the number of distinct rationals occurring as DBM entries is small, typically thousands over the lifetime of a long-running static analysis, the identifiers can be represented as 16-bit integers. This reduces the space consumption of a matrix, even with the overhead of the two additional tables. The resulting alternative representation for a DBM

been dubbed a Compact DBM (CoDBM) [7]. The net reduction in space over DBMs, which derives from each rational now being represented exactly once, improves cache behaviour. It also saves repeatedly initialising memory for storing the rationals, an auxiliary operation which matches the frequency of the addition and comparison. For long-running analyses, CoDBMs reduce memory consumption by approximately 30% and improve running-time by approximately 40%, by virtue of the reduction in memory initialisation and improved locality [7].

This paper focuses on the computational, rather than the space-saving aspects of CoDBMs. Our first contribution is in optimising a write to a CoDBM. A CoDBM employs an ordered table (the second table) which maps each rational encountered thus far during analysis to its unique identifier. Whenever an entry is to be written, the table is searched (using the binary search) for a rational and its corresponding identifier. We show how hashing and linear probing, can avoid the repeated comparisons made by binary search and avoid the need to maintain an ordered table. We report that the number of resulting comparisons is indeed small and report a compensate speedup and improved cache behaviour.

Our second contribution relates to join, which is one of the domain operations that occurs with high frequency. Join is computed pairwise on the entries of two DBMs, and likewise for CoDBMs, by comparing each entry point-wise and taking the maximum. Point-wise join can be simplified by checking if the two identifiers align, or if one matches the special identifier which is reserved for infinity. Both operations can be implemented in a lightweight manner using CoDBMs, thus avoiding expensive number comparison operations. These refinements constitute our second contribution.

Our third contribution exploits the infinity identifier in another domain operation: closure. Closure reduces to a sequence of addition and maximum calculations, the results of which will be infinity if either of their arguments are infinity. Thus, if an entry of the CoDBM feeds an addition, and that entry is the infinity identifier, the result of the addition is infinity, irrespective of its other argument. Likewise for maximum. A lightweight check can be introduced to detect when the inner loop of the closure calculation can be by-passed. An analogous refinement carries over to incremental closure [9,21]. These refinements make up the third contribution.

Cumulatively, these refinements close the performance gap between doubles and rationals for octagons, from which we conclude that the role of rationals needs to be reevaluated. The paper feeds into the growing body of work [2,3,7,14,23,25,26] on how best to realise octagons on stock architectures.

2 Background

An octagonal constraint [1,20,21] is a two-variable inequality of the syntactic form $x_i - x_j \leqslant c$, $x_i + x_j \leqslant c$ or $-x_i - x_j \leqslant c$ where c is a constant, and x_i and x_j are drawn from a finite set of variables $\{x_0, \ldots, x_{n-1}\}$. This class includes unary inequalities $x_i + x_i \leqslant c$ and $-x_i - x_i \leqslant c$ which express interval constraints.

$$
\begin{array}{r}
x_0 \leqslant 3 \\
x_1 \leqslant 2 \\
x_0 + x_1 \leqslant 6 \\
-x_0 - x_1 \leqslant 5 \\
-x_0 \leqslant 3
\end{array}
\qquad
\begin{array}{l}
x'_0 - x'_1 \leqslant 6 \\
x'_2 - x'_3 \leqslant 4 \\
x'_0 - x'_3 \leqslant 6 \\
x'_2 - x'_1 \leqslant 6 \\
x'_1 - x'_2 \leqslant 5 \\
x'_3 - x'_0 \leqslant 5 \\
x'_1 - x'_0 \leqslant 6
\end{array}
\qquad
\begin{array}{c}
\begin{array}{cccc} x'_0 & x'_1 & x'_2 & x'_3 \end{array} \\
\begin{array}{c} x'_0 \\ x'_1 \\ x'_2 \\ x'_3 \end{array}
\left[\begin{array}{cccc}
\infty & 6 & \infty & 6 \\
6 & \infty & 5 & \infty \\
\infty & 6 & \infty & 4 \\
5 & \infty & \infty & \infty
\end{array}\right]
\end{array}
$$

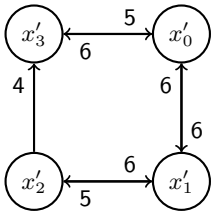

Fig. 1: Example of an octagonal system and its DBM representation

An octagon is a set of points satisfying a system of octagonal constraints. The octagon domain is the set of all octagons defined over a given set of variables.

2.1 DBMs

Implementations of the octagon domain reuse machinery developed for solving difference constraints of the form $x_i - x_j \leqslant c$. An octagonal constraint over $\{x_0, \ldots, x_{n-1}\}$ can be translated [21] to a difference constraint over an augmented set of variables $\{x'_0, \ldots, x'_{2n-1}\}$, which are interpreted by $x'_{2i} = x_i$ and $x'_{2i+1} = -x_i$. The translation proceeds as follows:

$$
\begin{array}{rl}
x_i - x_j \leqslant c \rightsquigarrow & x'_{2i} - x'_{2j} \leqslant c \ \wedge\ x'_{2j+1} - x'_{2i+1} \leqslant c \\
x_i + x_j \leqslant c \rightsquigarrow & x'_{2i} - x'_{2j+1} \leqslant c \ \wedge\ x'_{2j} - x'_{2i+1} \leqslant c \\
-x_i - x_j \leqslant c \rightsquigarrow & x'_{2i+1} - x'_{2j} \leqslant c \ \wedge\ x'_{2j+1} - x'_{2i} \leqslant c \\
x_i \leqslant c \rightsquigarrow & x'_{2i} - x'_{2i+1} \leqslant 2c \\
-x_i \leqslant c \rightsquigarrow & x'_{2i+1} - x'_{2i} \leqslant 2c
\end{array}
$$

A difference bound matrix (DBM) [10,19] (denoted $\mathbf{m}$), which is a square matrix of dimension $n \times n$, is commonly used to represent a systems of n^2 (syntactically irredundant [18]) difference constraints over n variables. The entry $\mathbf{m}_{i,j}$ represents the constant c of the inequality $x_i - x_j \leqslant c$ where $i, j \in [0, n)$. Since an octagonal constraint system over n variables translates to a difference constraint system over $2n$ variables, a DBM representing an octagon has dimension $2n \times 2n$. Figure 1 illustrates how an octagon translates to a system of differences. The entries of the DBM correspond to the constants in the difference constraints. Note how differences which are (syntactically) absent from the system lead to entries which take a symbolic value of ∞. Observe too how the DBM can be viewed as an adjacency matrix for the illustrated graph.

2.2 Closure

Closure properties define canonical representations of DBMs, and can decide satisfiability and support operations such as join and projection. Bellman [4] showed that the satisfiability of a difference system can be decided using shortest path algorithms on a graph representing the differences. If the graph contains

```
1: function CLOSE(m)
2:     for k ∈ {0, ..., 2n − 1} do
3:         for i ∈ {0, ..., 2n − 1} do
4:             for j ∈ {0, ..., 2n − 1} do
5:                 m_{i,j} ← min(m_{i,j}, m_{i,k} + m_{k,j})
6:             end for
7:         end for
8:     end for
9:     return m
10: end function
```

```
1: function STR(m)
2:     for i ∈ {0, ..., 2n − 1} do
3:         for j ∈ {0, ..., 2n − 1} do
4:             m_{i,j} ← min(m_{i,j}, (m_{i,ī} + m_{j̄,j})/2)
5:         end for
6:     end for
7:     return m
8: end function
```

Fig. 2: Non-incremental closure and strengthening

a negative cycle (a cycle whose edge weights sum to a negative value) then the difference system is unsatisfiable. The same applies for DBMs representing octagons. Closure propagates all the implicit (entailed) constraints in a system, leaving each entry in the DBM with the sharpest possible constraint entailed between the variables. A DBM $\mathbf{m}$ of dimension $n \times n$ is said to be closed iff $\forall i.\mathbf{m}_{i,i} = 0$ for all $i \in [0, n)$ and $\mathbf{m}_{i,j} \leqslant \mathbf{m}_{i,k} + \mathbf{m}_{k,j}$ for all $i, j, k \in [0, n)$. The DBM is said to be strongly closed iff additionally $\forall i, j.\mathbf{m}_{i,j} \leqslant \mathbf{m}_{i,\bar{i}}/2 + \mathbf{m}_{\bar{j},j}/2$ for all $i, j \in [0, n)$, where $\bar{i}$ is $i + 1$ if i is even, and $i - 1$ otherwise. The first property is enjoyed by octagons which are satisfiable, the second corresponds to all binary octagon constraints being propagated, and the third corresponds to all unary constraints being merged into binary constraints. Figure 2 gives a cubic implementation which tightens a DBM to ensure condition 2 and a quadratic pass which enforces condition 3. Condition 1 is checked by merely inspecting the diagonal of the tightened DBM.

2.3 Incremental Closure

Minè introduced incremental closure [21] which reestablishes closure once a small number of constraints are added to a closed DBM. This algorithm was subsequently refined [8,9] to give the quadratic algorithm listed in figure 3, presented both with and without loop-invariant code hoisting. The idea is to determine how each DBM entry $\mathbf{m}_{i,j}$ is effected by the addition of a new constraint $x'_a - x'_b \leqslant d$, independent of every other DBM entry.

The force of (strong) closure, whether incremental or not, is that it gives a canonical representation for DBMs; it also reduces join to the pointwise max of two closed DBMs, to give the quadratic join operation illustrated in Figure 4.

2.4 Apron

Apron is a widely-used Octagon domain library [15] which is implemented in C, with bindings for C++, Java and OCaml. It supports various number systems. Numbers are represented by a type `bound_t`, which, depending on compile-time options, will select a specific header file with a specific concrete implementation

```
1: function IncClose(m, x'_a − x'_b ⩽ d)
2:     for i ∈ {0, ..., 2n − 1} do
3:         for j ∈ {0, ..., 2n − 1} do
4:             m'_{i,j} ← min( m_{i,j},
                               m_{i,a} + d + m_{b,j},
                               m_{i,b̄} + d + m_{ā,j},
                               m_{i,b̄} + d + m_{ā,a} + d + m_{b,j},
                               m_{i,a} + d + m_{b,b̄} + d + m_{ā,j} )
5:         end for
6:     end for
7: end function
```

```
1: function IncCloseHoist(m, x'_a − x'_b ⩽ d)
2:     t1 ← d + m_{ā,a} + d;
3:     t2 ← d + m_{b,b̄} + d;
4:     for i ∈ {0, ..., 2n − 1} do
5:         t3 ← min(m_{i,a} + d, m_{i,b̄} + t1);
6:         t4 ← min(m_{i,b̄} + d, m_{i,a} + t2);
7:         for j ∈ {0, ..., 2n − 1} do
8:             m_{i,j} ← min(m_{i,j}, t3 + m_{b,j}, t4 + m_{ā,j})
9:         end for
10:    end for
11:    return m
12: end function
```

Fig. 3: Incremental Closure (without and with code hoisting)

```
1: function Join(m^1, m^2)
2:     for i ∈ {0, ..., 2n − 1} do
3:         for j ∈ {0, ..., 2n − 1} do
4:             m_{i,j} ← max(m^1_{i,j}, m^2_{i,j})
5:         end for
6:     end for
7:     return m
8: end function
```

Fig. 4: Join of two closed DBMs

of numbers extended with symbolic values of $-\infty$ and $+\infty$. Every `bound_t` object is initialised via a call to `bound_init`, which in the case of GMP rationals will call a `malloc` function. Numbers of type `bound_t` cannot be assigned directly, but instead are assigned via function calls such as `bound_set`.

DBMs are stored by taking advantage of coherence [21], which can be assumed without loss of generality. A DBM $\mathbf{m}$ is said to be coherent if $\mathbf{m}_{i,j} = \mathbf{m}_{\bar{j},\bar{\imath}}$ for all $i, j \in [0, n)$. Coherence allows a half-matrix to be represented which, in turn, can then be packed into a (linear) array of `bound_t` objects as follows: If $i \geqslant j$ or $i = \bar{j}$ then the entry at (i, j) in the DBM is stored at index $j + \lfloor i^2/2 \rfloor$ in the array. Otherwise (i, j) is stored at the index location reserved for entry $(\bar{j}, \bar{\imath})$. A DBM of dimension $n \times n$ then requires an array of size $2n(n + 1)$.

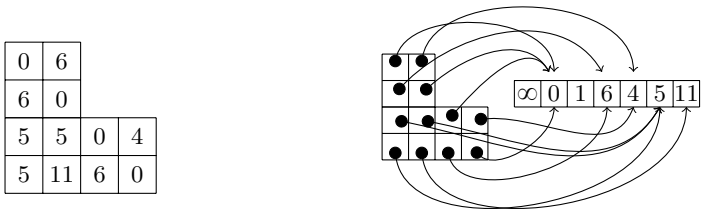

Fig. 5: Example illustrating the difference between DBMs and CoDBMs

```
1: function SEARCH(v)
2:     if v ∈ dom(search) then
3:         return search(v)
4:     else
5:         values ← values ∪ {ℓ ↦ v} where ℓ ∉ dom(values)
6:         search ← search ∪ {v ↦ ℓ}
7:         return ℓ
8:     end if
9: end function
```

Fig. 6: Searching and extending search and values

2.5 CoDBMs

Compact DBMs (CoDBMs) [7] redistribute the cost of memory allocation and initialisation, and do so in a way that is sensitive to the relative frequency of DBM reads to DBM writes (the latter being less frequent than the former). CoDBMs are matrices where the entries are identifiers (short integers), rather than numeric values (rationals), and each identifier references a number in a shared number pool, as illustrated in figure 5. The number pool is abstracted by two functions: $\mathsf{values} : \mathbb{N} \to \mathbb{Q}$ and $\mathsf{search} : \mathbb{Q} \to \mathbb{N}$, which are mutual inverses.

The change from DBMs to CoDBMs requires a new API for reading and writing an entry $\mathbf{c}_{i,j}$ of a CoDBM $\mathbf{c}$. Reading $\mathbf{c}_{i,j}$ amounts to interpreting the index stored in $\mathbf{c}_{i,j}$ using values to obtain a value $v = \mathsf{values}(\mathbf{c}_{i,j})$. Writing a value v to $\mathbf{c}_{i,j}$ involves applying a function $\ell = \textsc{search}(v)$ to retrieve the identifier ℓ for v and then assigning $\mathbf{c}_{i,j}$ to ℓ. The function SEARCH, which is listed in figure 6, manufactures a unique identifier if v is fresh and extends values and search accordingly. Previous work [7] realised values as an array of rationals and search as an ordered array of rationals, the index of a particular rational defining the identifier. The identifier was found using Bisection search [29]. CoDBMs achieve speedups because they store identifiers which are more compact than rationals (improving locality) and each distinct rational is stored once in the number pool (saving initialisation).

3 Hashing

The GMP manual [12] alludes to the fact that comparisons on rationals are expensive since $p/q \leqslant r/s$ reduces to $sp \leqslant qr$ if the denominators p and s are positive. Comparison thus involves two multiplications. Moreover, SEARCH is invoked on every write to the CoDBM and each invocation will compute $\lceil log_2(n) \rceil$ comparisons in the worst case where n is the number of rationals in number pool. Thus, even if the pool contains just 256 rationals, a write can induce 16 multi-precision multiplications. Moreover, to insert a new rational into an ordered table it is necessary to shuffle along other elements. These costs motivate hashing.

A rational r is hashed by converting it to a double-precision floating point number f, an operation which is supported by GMP. If s is the size of the hash table then a multiplicative hash [17] $h(f)$ is computed by calculating $\ell = \lceil fs(1 + \sqrt{5})/2 \rceil \mod s$ with floating-point arithmetic and defining $h(f) = \ell$ if $\ell \geqslant 0$ and $h(f) = s - \ell$ otherwise. Hashing with the Golden Ratio helps ensure that the hashes are scattered evenly, reducing the chance of collisions [17, Chapter 6.4]. If a rational does not exist at entry $h(f)$ then r is inserted at this entry and $h(f)$ is returned by SEARCH. If the entry $h(f)$ is already occupied by a rational r', then an equality check $r = r'$ is performed on rationals (which is constant-time by virtue of a canonical representation). If $r = r'$ succeeds then $h(f)$ is returned by SEARCH. If $r = r'$ fails then linear probing is applied to find the next consecutive (modulo s) identifier ℓ' whose entry is empty in the table. The rational r is then inserted at ℓ' and ℓ' is returned by SEARCH.

Although multiplicative hashes are not renowned for avoiding collisions, it turns out that collisions are incredibly rare simply because the number of distinct rationals is small and thus the occupancy of the table is low.

4 Optimising Join

DBMs typically contain many symbolic infinity values: a property has sparked an interest in using sparse representations for difference constraints [11]. However, sparse representations complicate the join of octagons [16], the simplicity of which we want to preserve. Nevertheless, the identifiers employed by CoDBMs enable join to by-pass vacuous DBM entries, without adding any conceptual complexity to join itself. The idea is to merely fix the identifier for symbolic infinity up-front so that infinity can be intercepted with a lightweight check without inspecting the symbolic value itself.

To reflect on the cost of join, consider the implementation of a max operation (SETDBMMAX) used in join, which assigns $\mathbf{c}_{i,j}$ to the maximum of $\mathbf{c^1}_{i,j}$ and $\mathbf{c^2}_{i,j}$ shown in figure 7. Quite apart from the two multi-precision multiplications used in the comparison which underpins BOUND_MAX in line 3, line 2 allocates and initialises memory (which we make explicit to highlight a hidden cost).

Yet if either $\mathbf{c^1}_{i,j}$ or $\mathbf{c^2}_{i,j}$ is the identifier for infinity, denoted ℓ_∞, then there is no need to perform any comparison between rationals: the entry $\mathbf{c}_{i,j}$ can simple be assigned the identifier ℓ_∞. Moreover, if the identifiers $\mathbf{c^1}_{i,j}$ and $\mathbf{c^2}_{i,j}$ align,

```
1: function SETDBMMAX(c, (i, j), c¹, c²)
2:     BOUND_INIT(TMP)
3:     TMP ← BOUND_MAX(values(c¹_{i,j}), values(c²_{i,j}))
4:     c_{i,j} ← SEARCH(TMP)
5: end function
```

```
1: function SETDBMMAX_OPT(c, (i, j), c¹, c²)
2:     if (c¹_{i,j} = ℓ∞ ∨ c²_{i,j} = ℓ∞ then
3:         c_{i,j} ← ℓ∞
4:     else if (c¹_{i,j} = c²_{i,j}) then
5:         c_{i,j} ← c¹_{i,j}
6:     else
7:         if values(c¹_{i,j}) ⩾ values(c²_{i,j}) then
8:             c_{i,j} ← c¹_{i,j}
9:         else
10:            c_{i,j} ← c²_{i,j}
11:        end if
12:    end if
13: end function
```

Fig. 7: DBM max operation used in join and widening, and its optimised version

then again a rational comparison is not needed. In fact, only in exceptional cases do the rationals need to be looked-up at all, which reduces memory pressure. This optimisation can be rolled out for widening which also uses SETDBMMAX. An analogous optimisation applies for meet, using min instead of max (though meet arises relatively infrequently during analysis).

5 Optimising Closure

Figure 8 shows how the identifier ℓ_∞ can be likewise trapped to speed up non-incremental and incremental closure. The observation is that if $\mathbf{c}_{i,k} = \ell_\infty$ then then sum $\mathsf{values}(\mathbf{c}_{i,k}) + \mathsf{values}(\mathbf{c}_{k,j})$ will be infinity irrespective of the identifier stored in $\mathbf{c}_{k,j}$. Moreover, the check $\mathbf{c}_{i,k} = \ell_\infty$ is performed on identifiers (integers) rather than rationals, so has neglible overhead, yet it potentially enables the entire inner loop of closure to be short-circuited (see CLOSEOPT of figure 8). Incremental closure algorithm can also be optimised (see INCCLOSEHOISTOPT of figure 8) where both the outer and inner loop can be skipped if certain indices match the fixed identifier ℓ_∞. These optimisations will only really benefit closure calculations on large CoDBMs so it is important that the checks are sufficiently lightweight to not overburden closures operating on small CoDBMs.

6 Experiments

This section compares the performance of CoDBMs over rationals against DBMs over doubles using three abstract interpreters [6,13,27], reporting execution times augmented with memory statistics for the longest running analyses. All statistics were gathered on a Linux machine equipped with 128GB of RAM and dual 2.0GHz Intel Xeon E5-2650 processors. Timings were averaged over five runs using multitime (`http://tratt.net/laurie/src/multitime/`) and include the time required to perform a complete analysis from parsing source to output.

```
1: function CloseOpt(c)
2:     for k ∈ {0, ..., 2n − 1} do
3:         for i ∈ {0, ..., 2n − 1} do
4:             if c_{i,k} ≠ ℓ_∞ then
5:                 for j ∈ {0, ..., 2n − 1} do
6:                     c_{i,j} ← Search(min(values(c_{i,j}), values(c_{i,k}) + values(c_{k,j})))
7:                 end for
8:             end if
9:         end for
10:    end for
11: end function
12:
13: function IncCloseHoistOpt(c, x'_a − x'_b ⩽ d)
14:    t_1 ← d + values(c_{ā,a}) + d;
15:    t_2 ← d + values(c_{b,b̄}) + d;
16:    for i ∈ {0, ..., 2n − 1} do
17:        if c_{i,a} ≠ ℓ_∞ ∧ c_{i,b̄} ≠ ℓ_∞ then
18:            t_3 ← min(values(c_{i,a}) + d, values(c_{i,b̄}) + t_1);
19:            t_4 ← min(values(c_{i,b̄}) + d, values(c_{i,a}) + t_2);
20:            for j ∈ {0, ..., 2n − 1} do
21:                if c_{b,j} ≠ ℓ_∞ ∧ c_{ā,j} ≠ ℓ_∞ then
22:                    c_{i,j} ← Search(min(values(c_{i,j}), t_3 + values(c_{b,j}), t_4 + values(c_{ā,j})))
23:                end if
24:            end for
25:        end if
26:    end for
27: end function
```

Fig. 8: Optimised versions of closure and incremental closure

6.1 FuncTion: Timings

Figure 9[1] presents the running times of the FuncTion termination analyser on all the 58 benchmarks from its repository (`https://github.com/caterinaurban/function`). FuncTion [27] applies abstract interpretation to infer piece-wise ranking functions for verifying termination. It is implemented in OCaml and can analyse simple programs in a C-like language. FuncTion has options for intervals, arbitrary polyhedra and octagons. For octagons, the default setting is Apron DBMs instantiated with rationals, reflecting a focus on verification. Doubles and CoDBMs were supported by changing the build system.

The cross marks of the scatter plot compare the running time for Apron DBMs over rationals against the execution time for Apron DBMs over doubles, so as quantify the overhead induced by rationals. The dotted-line has the gradient of one. The triangles illustrate the execution time of CoDBMs over rationals, equipped with the complete set of optimisations, again relative to doubles on DBMs. The top graph illustrates these timing for all benchmarks, whereas the bottom graph zooms in on the cluster of benchmarks around the origin. The proximity of the triangles near to the dotted line suggests that CoDBMs over rationals, when optimised, compare favourably to DBMs over doubles, at least for the benchmarks under test. Figure 10 compares CoDBMs to DBMs using a

[1] The two graphs of Figure 9 summarise the timings detailed in the long table of section A.1 which will be removed in the final version of the paper.

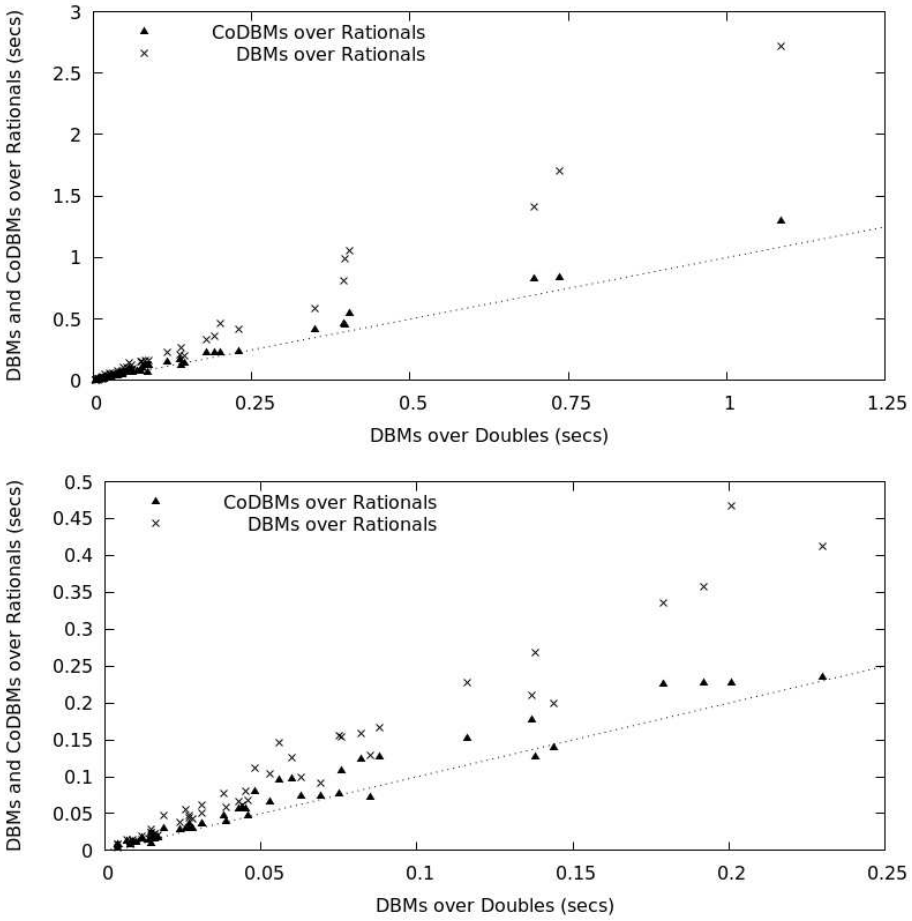

Fig. 9: CoDBMs and DBMs for rationals against DBMs for doubles

different perspective: it also compares CoDBMs over rationals to CoDBMs over doubles, showing how CoDBMs, when optimised, reduce the overhead of moving from doubles to rationals relative to the same move on DBMs.

6.2 Crab-LLVM: Timings

To compare rationals against doubles with a state-of-the-art [13] interprocedural analysis, Crab-LLVM (`https://github.com/seahorn/crab-llvm`) was built against Apron and CoDBMs and then applied to the 596 benchmarks of product-line SV-COMP series to infer octagonal invariants. This setup also exercised the domain operations from C rather than through OCaml bindings so as to check whether the bindings impacted performance.

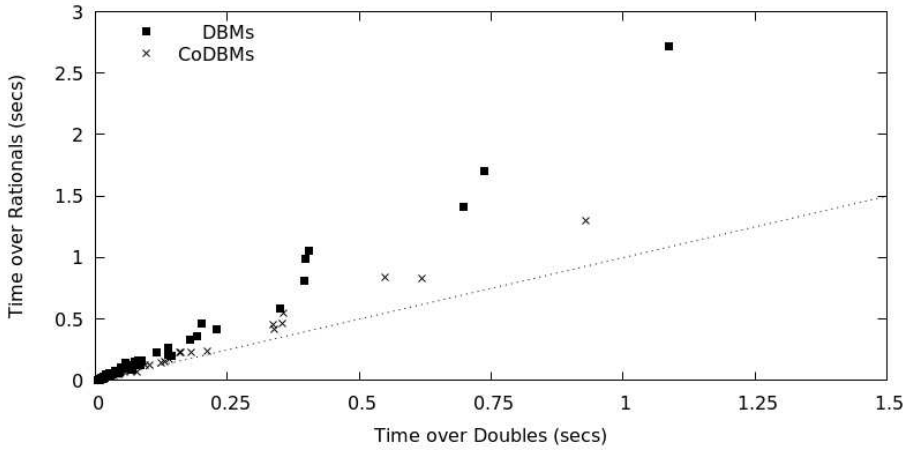

Fig. 10: DBMs for rationals against DBMs for doubles and likewise for CoDBMs

The large number of benchmarks make a scatter plot infeasible, hence Figure 11 plots[2] the cumulative running time for the first n benchmarks against n itself. These benchmarks divide into the elevator, email, and minepump sub-series of unreachability problems. Each benchmark in each sub-series has a broadly similar execution time, so the cumulative running time approximates a piece-wise linear function. The headline message is that, again, CoDBMs over rationals approach the performance of DBMS over doubles; moreover CoDBMs provide a modest gain on DBMs for doubles when all the optimisations are in place. Interestingly, Crab-LLVM defaults to the Elina library [26] which partitions a DBM on-the-fly into sub-DBMs that do not share variables. It also applies vectorisation. This invited a comparison. Elina did not perform as well as CoDBMs with the join and closure optimisations or even DBMs. Though unexpected, we include these results nevertheless. (It should be stressed that Elina was built exactly as specified, to the same level of optimisation as DBMs and CoDBMs, with the vector flag correctly set for an E5-2650 which supports vectorisation. Elina currently does not provide OCaml bindings otherwise we would have performed further comparisons using FuncTion and Frama-C.)

6.3 Frama-C: Timings

With an eye towards longer running analyses, EVA [6], the abstract interpretation plugin for Frama-C Sulfur, was used for comparing rationals against doubles for DBMs and CoDBMs. EVA is a prototype analyser for C99 which

[2] Again, the figure summarise the timings detailed in the long table in section A.2 which will be removed in the final version of the paper.

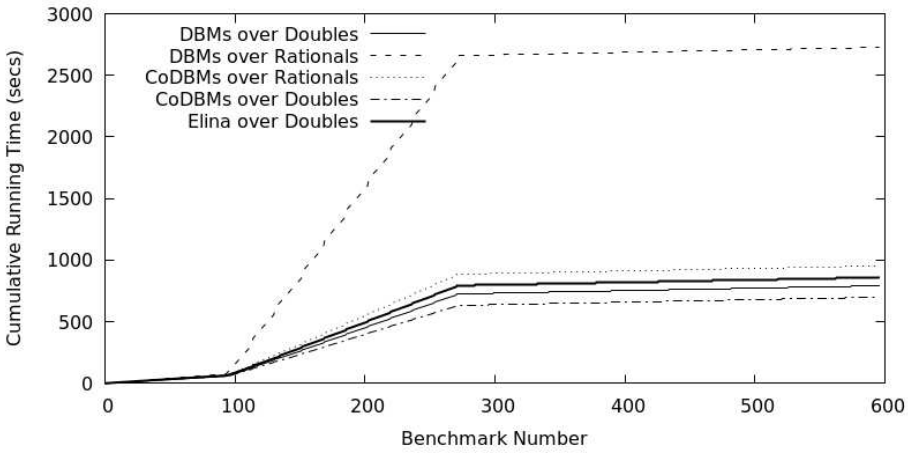

Fig. 11: Cumulative execution times over product-lines SV-COMP benchmarks

Abbrv	Benchmark	LOC	Description
lev	levenstein	187	Levenstein string distance library
sol	solitaire	334	card cipher
2048	2048	435	2048 game
kh	khash	652	hash code from klib C library
taes	Tiny-AES	813	portable AES-128 implementation
mod	libmodbus	7685	library to interact with Modbus protocol
mgmp	mini-gmp	11787	subset of GMP library
bzip	bzip-single-file	74017	bzip single file for static analysis benchmarking

Fig. 12: Benchmarks

	Apron DBM			CoDBM rationals				CoDBM doubles			
Abbrv	rationals	doubles	Ids	Bisect	Hash	Join	Close	Bisect	Hash	Join	Close
lev	22.78	4.71	900	12.16	10.66	8.15	7.41	5.87	4.78	4.74	4.30
sol	92.13	42.77	2161	80.22	71.00	51.48	50.61	52.09	44.03	42.89	42.52
2048	37.74	8.28	358	22.36	19.57	14.09	13.31	10.79	8.73	8.26	7.77
kh	3.087	1.92	196	2.44	2.447	2.167	2.131	1.871	2.014	1.928	1.843
taes	1883.50	153.43	140	740.47	663.37	411.72	386.67	261.32	164.85	143.63	129.67
mod	820.88	96.68	3627	558.27	494.12	321.72	293.68	192.63	111.92	102.78	91.70
mgmp	4.33	4.36	126	4.33	4.28	4.18	4.16	4.38	4.28	4.22	4.15
bzip	655.82	54.15	262	232.63	94.98	95.49	94.20	168.90	60.01	59.00	58.99

Fig. 13: Frama-C EVA plugin timings

supports Apron but does not provide state-of-the-art optimisations such as automatic variable clustering [14] or access-based localisation [3]. Nevertheless, figure 12 lists the programs used for benchmarking, which represent eight programs from the Frama-C case study repository (`https://github.com/Frama-C/open-source-case-studies`) that successfully terminate when the EVA plugin is instantiated with octagons.

		CoDBM			
rationals	Apron DBM	Bisect	Hash	Join	Close
lev	1,498,168	1,457,632 (24%)	1,452,868	106,072	106,620
sol	6,187,568	10,718,464 (35%)	10,717,924	246,008	247,044
2048	3,714,408	2,999,468	3,034,068	170,840	170,700
kh	163,840	142,168	142,264	69,696	69,440
taes	20,845,632	119,190,024 (8%)	119,153,636	591,228	590,700
mod	28,945,580	73,116,452 (38%)	73,111,892	911,572	901,788
mgmp	89,496	89,992 (46%)	88,844	89,308	88,508
bzip	9,499,460	1,636,276	1,634,444	551,664	551,516

		CoDBM			
doubles	Apron DBM	Bisect	Hash	Join	Close
lev	191,968	106,264	106,004	106,200	105,756
sol	690,476	284,556	284,068	283,812	283,952
2048	413,916	167,432	165,548	165,712	165,284
kh	73,608	68,780	68,720	68,376	68,696
taes	2,044,832	603,056	604,440	604,416	604,420
mod	2,871,612	921,780	939,056	919,808	922,108
mgmp	89,832	88,092	88,548	88,008	88,628
bzip	1,158,460	555,216	551,884	552,524	552,988

Fig. 14: Memory Usage in kb for Frama-C: rationals above and doubles below

Figure 13 details the overall execution for DBMs, both for doubles and rationals. Interestingly, teas, mod and bzip are ten-fold slower with rationals than doubles for DBMs. This stems from a high number of DBMs with high dimension so that, cumulatively, the total number of DBMs entries created during analysis for each of these three problems is between 40- and 400-fold the number of DBM entries created for any of the other five problems. The Ids column records the total number of identifiers (distinct DBM entries) used over the lifetime of each analysis. These counts are significantly smaller than the total number of DBM entries over the lifetime of each analysis by typically six orders of magnitude larger. (Shorter running analyses typically have smaller Ids counts.)

The Bisect column records the overall running time when bisection search is used to locate an identifier. Hash gives the runtime when hashing and linear probing is used instead. Join presents the time when hashing is augmented with join (and meet) optimisation. The Close column additions applies the optimisations on both closure and incremental closure. For the longer running problems, Hash significantly improves on Bisect and Join significantly improves on Bisect (which the notable exception of mgmp where it has little effect). Close makes a less significant improvement on Join, but is useful nevertheless.

As a control, the last four columns of the table repeat the experiments but with CoDBMs instantiated with doubles. Since arithmetic is faster on doubles and doubles are compact, it is surprising to see that CoDBMs sometimes outperform DBMs. We surmise that the speedup comes from reduced memory pressure.

6.4 Frama-C: Memory Usage

Figure 14 records total memory usage as harvested by GNU time, which returns the maximum resident set size of the process during its lifetime. The table contains some surprises. First, memory usage for CoDBMs over rationals gives a

Abbrv		Insts	Refs	Miss	Rate	Abbrv		Insts	Refs	Miss	Rate
2048	A	242190m	99227m	185m	0.187	lev	A	145649m	59428m	72m	0.121
2048	B	125304m	47092m	52m	0.112	lev	B	67952m	25346m	25m	0.101
2048	H	95078m	41082m	52m	0.128	lev	H	52800m	22565m	25m	0.113
2048	J	65529m	30433m	5m	0.016	lev	J	38634m	17472m	3m	0.017
2048	C	56981m	26670m	5m	0.018	lev	C	32925m	14975m	3m	0.020
kh	A	13870m	6545m	5.2m	0.08	bzip	A	52677557m	2166803m	19579m	0.904
kh	B	9228mm	4595m	2.7m	0.06	bzip	B	1855916m	410861m	76m	0.019
kh	H	8579mm	4465m	2.7m	0.06	bzip	H	533543m	229553m	76m	0.033
kh	J	7840m	4196m	1.3m	0.03	bzip	J	522984m	225737m	57m	0.026
kh	C	7690m	4130m	1.3m	0.03	bzip	C	512926m	221316m	57m	0.026

Fig. 15: Instruction count and cache statistics for Frama-C for rationals

net increase on DBMs over rationals for some problems. This increase occurs for the Sulfur version of Frama-C; the previous version gives a consistent reduction in memory usage (which is expressed as a percentage in the same column). The problem in recycling memory seems to relate to GNU multi-precision arithmetic since Sulfur gives a reduction when the same CoDBM code is instantiated with doubles. The second surprise is that Sulfur gives a dramatic overall decrease when CoDBMs are deployed with the join optimisation. This stems from the allocation and initialisation space for each maxima. This only occurs for rationals, hence Join offers little improvement for doubles.

Figure 15 records cache statistics for DBMs and CoDBMs which were harvested with Cachegrind [22] (for four representative benchmarks). Refs and Miss respectively denote the number of memory references and last-level cache misses for rationals for both DBMs and CoDBMs, where m denotes millions. A, B, H, J and C abbreviate the column names of Figure 14. Reading an element from a CoDBM incurs an extra layer of indirection compared to a DBM and writing to a CoDBM can incur multiple memory references, so one might expect additional memory references. Yet the number of references reduces uniformally between DBMs and CoDBMs for Bisect and then across the CoDBM optimisations. The number of cache misses reduces even faster, indicating that locality is improved too, hence the decreasing cache miss-rate percentage (Rate). Reassuringly, the number of misses does not increase between Bisect and Hash, even though hashing can map a number to any location in the hash table, whereas bisection will only search the portion of the second table which is actually populated. A single (non-local) read into the hash table (which is the norm as collisions are rare) seems to more than offset the multiple reads incurred by bisection, which become progressively more local as search proceeds. Join gives an order of magnitude reduction in the number of misses because it by-passes accessing numbers in the first table as well as avoiding initialising a temporary variable and then storing the maxima. Cachegrind also records the number of instructions executed, which is a reflected in the Insts column, and is a proxy for work. The reduction in instruction count stands independent of the timings which are ultimately dependent on system behaviour.

7 Related Work

The tension between the elegance of octagons and their scalability has motivated a number of imaginative techniques [2,3,5,7,14,23,25,26] for enhancing octagonal analysis. First, variable clustering was proposed [5,20,28] for grouping variables into sets which scope the relationships that are tracked. However, deciding variable groupings is an art, although there has been recent progress made in automating decomposition both before [14] and during [26] analysis.

Second, the domain operations themselves have been refined, notably showing how strengthening (the act of combining pairs of unary octagon constraints to improve binary octagon constraints) need not be applied repeatedly, but instead can be left to a single post-processing step [1]. This led to a significant performance improvement of approximately 20% [1].

Thirdly, and more recently, there has been a move to curb the size of DBMs using sparse analyses [24] and access-based localisation techniques [3]. Access-based localisation uses scoping heuristics to adjust the size of the DBM to those variables that can actually be updated [3]. Sparse analyses generalise access-based localisation techniques, using data dependencies to adjust the size of abstract states propagated to method calls: [24] defines a generic technique to apply sparse techniques to abstract interpretation and combines this with variable packing to scale an octagon-based abstract interpreter for C programs. Access-based localisation and sparse frameworks (and variable clustering too) are orthogonal to our work, and can take advantage of the techniques introduced in this paper. Sparse matrix representations have been proposed for octagons [16] and differences [11] as an alternative to DBMs, but these representations sit at odds with the simplicity of the original domains algorithms. The desirable property of strong closure [21] (the normal form for octagons) does not hold for a sparse representation, motivating the need to rework domain operations [16].

Fourthly, there has been a move to better exploit the underlying architecture, either to harness GPUs [2] or advanced vector extensions (AVX) [26] in closure and strengthening (the latter being the first work to comment on the impact of cache misses in domain engineering).

8 Conclusions

We buck the trend towards instantiating octagons with doubles by showing how CoDBMs, which save space over DBMs, can also save on computation if equipped with simple optimisations. These optimisations enable arithmetic to be short-circuited in join and closure, and also avoid repeated comparisons by changing the tables which underpin CoDBMs. The net effect is to put rationals on a par with doubles, so as to simultaneously achieve performance and soundness.

Acknowledgements We thank Colin King at Canonical and Laurie Tratt at Kings for their help with the performance analysis which underpinned this work.

References

1. R. Bagnara, P. M. Hill, and E. Zaffanella. Weakly-relational Shapes for Numeric Abstractions: Improved Algorithms and Proofs of Correctness. *Formal Methods in System Design*, 35(3):279–323, 2009.
2. F. Banterle and R. Giacobazzi. A Fast Implementation of the Octagon Abstract Domain on Graphics Hardware. In *SAS*, volume 4634 of *LNCS*, pages 315–335. Springer, 2007.
3. E. Beckschulze, S. Kowalewski, and J. Brauer. Access-Based Localization for Octagons. *Electronic Notes in Theoretical Computer Science*, 287:29–40, 2012.
4. R. Bellman. On a Routing Problem. *Quarterly of Applied Mathematics*, 16:87–90, 1958.
5. B. Blanchet, P. Cousot, R. Cousot, J. Feret, L. Mauborgne, A. Miné, D. Monniaux, and X. Rival. A Static Analyzer for Large Safety-Critical Software. In *PLDI*, pages 196–207, 2003.
6. D. Bühler, P. Cuoq, B. Yakobowski, M. Lemerre, A. Maroneze, V. Perrelle, and V. Prevosto. *The EVA Plug-in*. CEA LIST, Software Reliability Laboratory, Saclay, France, F-91191, 2017. `https://frama-c.com/download/frama-c-value-analysis.pdf`.
7. A. Chawdhary and A. King. Compact Difference Bound Matrices. In *Asian Symposium on Programming Languages and Systems*, volume 10695 of *LNCS*, pages 471–490. Springer, 2017.
8. A. Chawdhary, E. Robbins, and A. King. Simple and Efficient Algorithms for Octagons. In *APLAS*, volume 8858 of *LNCS*, pages 296–313. Springer, 2014.
9. A. Chawdhary, E. Robbins, and A. King. Incrementally Closing Octagons. *Formal Methods in System Design*, pages 1–46, 2018. `https://doi.org/10.1007/s10703-017-0314-7`.
10. D. Dill. Timing Assumptions and Verification of Finite-State Concurrent Systems. In J. Sifakis, editor, *CAV*, volume 407 of *LNCS*, pages 187–212. Springer, 1989.
11. G. Gange, J. A. Navas, P. Schachte, H. Søndergaard, and P. Stuckey. Exploiting Sparsity in Difference-bound Matrices. In *SAS*, volume 9837 of *LNCS*, pages 189–211, 2016.
12. GNU. *GNU MP Manual*, 2016. `https://gmplib.org/manual/`.
13. A. Gurfinkel, T. Kahsai, A. Komuravelli, and J. Navas. The SeaHorn Verification Framework. In *CAV*, volume 9206 of *LNCS*, pages 343–361. Springer, 2015.
14. K. Heo, H. Oh, and H. Yang. Learning a Variable-Clustering Strategy for Octagon From Labeled Data Generated by a Static Analysis. In *SAS*, volume 9837 of *LNCS*, pages 237–256, 2016.
15. B. Jeannet and A. Miné. APRON: A library of numerical abstract domains for static analysis. In *CAV*, volume 5643 of *LNCS*, pages 661–667, 2009.
16. J.-H. Jourdan. *Verasco: a Formally Verified C Static Analyzer*. PhD thesis, Université Paris Diderot (Paris 7) Sorbonne Paris Cité, May 2016.
17. Knuth, D. *The Art of Computer Programming, Volume 3: (2nd Ed.) Sorting and Searching*. Addison Wesley, 1998.
18. J.-L. Lassez, T. Huynh, and K. McAloon. Simplication and Elimination of Redundant Linear Arithmetic Constraints. In *Constraint Logic Programming*, pages 73–87. MIT Press, 1993.
19. M. Measche and B. Berthomieu. Time Petri-nets for analyzing and verifying time dependent communication protocols. In H. Rudin and C. West, editors, *Protocol Specification, Testing and Verification III*, pages 161–172. North-Holland, 1983.

20. A. Miné. *Weakly Relational Numerical Abstract Domains*. PhD thesis, École Polytechnique En Informatique, 2004.
21. A. Miné. The Octagon Abstract Domain. *HOSC*, 19(1):31–100, 2006.
22. N. Nethercote. *Dynamic Binary Analysis and Instrumentation*. PhD thesis, Trinity College, University of Cambridge, 2004.
23. H. Oh, L. Brutschy, and K. Yi. Access Analysis-Based Tight Localization of Abstract Memories. In *VMCAI*, volume 6538 of *LNCS*, pages 356–370, 2011.
24. H. Oh, K. Heo, W. Lee, D. Park, J. Kang, and K. Yi. Global Sparse Analysis Framework. *ACM TOPLAS*, 36(3):8:1–8:44, 2014.
25. H. Oh, W. Lee, K. Heo, H. Yang, and K. Yi. Selective Context-Sensitivity Guided by Impact Pre-analysis. In *PLDI*, pages 475–484, 2014.
26. G. Singh, M. Püschel, and M. Vechev. Making Numerical Program Analysis Fast. In *PLDI*, pages 303–313. ACM Press, 2015.
27. C. Urban. FuncTion: An Abstract Domain Functor for Termination (Competition Contribution). In *TACAS*, volume 9035 of *LNCS*, pages 464–466. Springer, 2015.
28. A. Venet and G. Brat. Precise and Efficient Static Array Bound Checking for Large Embedded C Programs. In *PLDI*, pages 231–242, 2004.
29. L. F. Williams Jr. A Modification to the Half-Interval Search (Binary Search) Method. In *Proceedings of the 14th ACM Southeast Conference*, pages 95–101, 1976.

A Experimental Appendix

A.1 FuncTion

	Apron DBM		CoDBM rationals				CoDBM doubles			
file	rationals	doubles	Bisect	Hash	Join	Close	Bisect	Hash	Join	Close
boolean.c	0.045	0.027	0.038	0.039	0.037	0.031	0.024	0.023	0.028	0.024
cacm2009a.c	0.2	0.144	0.169	0.17	0.162	0.14	0.115	0.122	0.144	0.124
cacm2009b.c	0.129	0.085	0.113	0.1	0.107	0.073	0.078	0.075	0.079	0.078
cav2006.c	0.092	0.069	0.085	0.083	0.077	0.075	0.07	0.055	0.063	0.067
countdown.c	0.014	0.007	0.015	0.016	0.014	0.013	0.008	0.006	0.008	0.008
euclid.c	0.013	0.01	0.011	0.012	0.012	0.012	0.008	0.01	0.011	0.01
example0.c	0.038	0.024	0.035	0.03	0.032	0.029	0.026	0.024	0.024	0.023
example10.c	0.413	0.23	0.298	0.286	0.297	0.235	0.232	0.227	0.216	0.213
example1a.c	0.078	0.038	0.06	0.062	0.053	0.047	0.044	0.044	0.044	0.04
example1b.c	0.111	0.048	0.098	0.082	0.096	0.081	0.05	0.052	0.056	0.049
example1.c	0.015	0.012	0.015	0.016	0.015	0.016	0.012	0.012	0.012	0.012
example1c.c	0.126	0.06	0.096	0.11	0.108	0.097	0.068	0.062	0.068	0.057
example1d.c	0.154	0.076	0.123	0.124	0.124	0.108	0.08	0.058	0.075	0.072
example1e.c	0.159	0.082	0.146	0.14	0.142	0.125	0.083	0.092	0.092	0.082
example2a.c	0.048	0.027	0.039	0.037	0.037	0.036	0.031	0.028	0.026	0.023
example2b.c	0.467	0.201	0.276	0.28	0.256	0.228	0.208	0.17	0.2	0.163
example2.c	0.357	0.192	0.265	0.249	0.254	0.228	0.207	0.19	0.199	0.16
example2c.c	0.989	0.397	0.611	0.577	0.554	0.456	0.408	0.387	0.385	0.336
example2d.c	1.706	0.736	1.032	1.004	0.961	0.839	0.734	0.694	0.701	0.549
example2e.c	2.72	1.086	1.715	1.627	1.628	1.302	1.143	1.11	1.03	0.929
example5.c	0.021	0.017	0.019	0.013	0.02	0.018	0.016	0.016	0.016	0.016
example7.c	0.023	0.016	0.02	0.02	0.019	0.02	0.016	0.016	0.015	0.015
example8.c	0.068	0.046	0.056	0.054	0.052	0.048	0.037	0.043	0.045	0.044
example.c	0.056	0.026	0.04	0.033	0.033	0.031	0.025	0.026	0.023	0.023
issue8.c	0.156	0.075	0.106	0.093	0.1	0.078	0.08	0.073	0.071	0.065
mccarthy91.c	0.227	0.116	0.173	0.148	0.175	0.153	0.139	0.136	0.137	0.131
mnav.c	0.047	0.019	0.037	0.032	0.031	0.031	0.022	0.023	0.023	0.019
peterson.c	1.056	0.404	0.636	0.646	0.569	0.551	0.414	0.39	0.388	0.357
pingpong.c	0.051	0.031	0.038	0.034	0.038	0.037	0.028	0.025	0.03	0.03
postdecrement.c	0.023	0.016	0.02	0.019	0.019	0.016	0.013	0.014	0.015	0.016
postincrement.c	0.024	0.015	0.019	0.017	0.018	0.016	0.013	0.015	0.016	0.014
predecrement.c	0.024	0.015	0.019	0.018	0.018	0.01	0.015	0.014	0.016	0.015
preincrement.c	0.017	0.014	0.019	0.02	0.018	0.015	0.016	0.014	0.015	0.015
recursion.c	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004	0.004
sas2010.c	0.59	0.35	0.442	0.448	0.448	0.42	0.345	0.356	0.355	0.339
sas2014a.c	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.008	0.006	0.006
sas2014b.c	0.029	0.015	0.027	0.024	0.024	0.024	0.02	0.02	0.019	0.017
sas2014c.c	0.012	0.008	0.011	0.011	0.011	0.012	0.008	0.008	0.009	0.008
simple.c	0.008	0.004	0.008	0.008	0.008	0.008	0.004	0.004	0.005	0.005
sink.c	0.008	0.004	0.008	0.007	0.008	0.008	0.004	0.006	0.006	0.004

sorting4.c	0.268	0.138	0.157	0.158	0.152	0.128	0.117	0.13	0.127	0.103
tacas2013a.c	0.043	0.028	0.037	0.038	0.038	0.031	0.028	0.027	0.023	0.028
tacas2013b.c	0.059	0.039	0.045	0.043	0.041	0.039	0.039	0.039	0.028	0.035
tacas2013c.c	0.813	0.396	0.562	0.525	0.504	0.468	0.404	0.391	0.387	0.354
tacas2013d.c	1.416	0.697	0.903	0.972	0.908	0.83	0.699	0.689	0.651	0.618
tap2008a.c	0.066	0.043	0.068	0.06	0.066	0.057	0.047	0.043	0.042	0.042
tap2008b.c	0.081	0.045	0.067	0.06	0.06	0.057	0.045	0.045	0.043	0.043
tap2008c.c	0.335	0.179	0.258	0.236	0.244	0.226	0.19	0.182	0.192	0.181
tap2008d.c	0.146	0.056	0.111	0.111	0.103	0.096	0.081	0.069	0.068	0.061
tap2008e.c	0.211	0.137	0.183	0.186	0.172	0.177	0.15	0.147	0.135	0.14
tap2008f.c	0.167	0.088	0.142	0.137	0.13	0.128	0.093	0.088	0.104	0.095
vijay.c	0.062	0.031	0.042	0.043	0.036	0.037	0.036	0.032	0.03	0.029
vmcai2004a.c	0.019	0.012	0.017	0.02	0.018	0.018	0.015	0.016	0.016	0.013
vmcai2004b.c	0.02	0.016	0.02	0.018	0.019	0.02	0.016	0.016	0.016	0.014
widening1.c	0.099	0.063	0.084	0.076	0.074	0.074	0.065	0.066	0.058	0.048
widening2.c	0.061	0.044	0.055	0.058	0.057	0.059	0.043	0.045	0.044	0.045
widening3.c	0.014	0.009	0.012	0.012	0.011	0.012	0.01	0.011	0.01	0.01
zune.c	0.104	0.053	0.081	0.07	0.073	0.067	0.061	0.056	0.058	0.055

A.2 Crab-LLVM

	DBM		CoDBM		Elina
file	rationals	doubles	rationals	doubles	doubles
elevator_spec13_product21_true-unreach-call.cil.c	1.139	0.836	0.936	0.803	0.851
elevator_spec13_product22_true-unreach-call.cil.c	1.129	0.831	0.933	0.831	0.831
elevator_spec13_product23_true-unreach-call.cil.c	0.775	0.606	0.667	0.629	0.655
elevator_spec13_product24_true-unreach-call.cil.c	1.133	0.844	0.94	0.856	0.877
elevator_spec13_product29_true-unreach-call.cil.c	1.137	0.855	0.961	0.816	0.817
elevator_spec13_product30_true-unreach-call.cil.c	0.776	0.608	0.683	0.616	0.67
elevator_spec13_product31_true-unreach-call.cil.c	0.813	0.611	0.666	0.616	0.68
elevator_spec13_product32_true-unreach-call.cil.c	0.79	0.627	0.685	0.633	0.688
elevator_spec13_productSimulator_true-unreach-call.cil.c	0.845	0.635	0.687	0.635	0.73
elevator_spec14_product03_true-unreach-call.cil.c	0.796	0.584	0.656	0.595	0.65
elevator_spec14_product11_true-unreach-call.cil.c	0.77	0.605	0.634	0.614	0.648
elevator_spec14_product19_true-unreach-call.cil.c	0.762	0.608	0.667	0.596	0.653
elevator_spec14_product20_false-unreach-call.cil.c	0.781	0.614	0.684	0.604	0.66
elevator_spec14_product23_true-unreach-call.cil.c	0.779	0.589	0.694	0.58	0.695
elevator_spec14_product24_false-unreach-call.cil.c	0.794	0.594	0.658	0.599	0.667
elevator_spec14_product27_true-unreach-call.cil.c	0.788	0.574	0.668	0.595	0.619
elevator_spec14_product28_false-unreach-call.cil.c	0.79	0.588	0.67	0.613	0.651
elevator_spec14_product31_true-unreach-call.cil.c	0.805	0.588	0.667	0.627	0.686
elevator_spec14_product32_false-unreach-call.cil.c	0.805	0.618	0.711	0.637	0.676
elevator_spec14_productSimulator_false-unreach-call.cil.c	0.843	0.644	0.71	0.619	0.715

elevator_spec1_product01_true-unreach-call.cil.c	0.757	0.605	0.678	0.572	0.685
elevator_spec1_product03_true-unreach-call.cil.c	0.792	0.587	0.642	0.593	0.652
elevator_spec1_product09_true-unreach-call.cil.c	0.783	0.601	0.639	0.585	0.644
elevator_spec1_product11_true-unreach-call.cil.c	0.777	0.585	0.676	0.609	0.678
elevator_spec1_product17_true-unreach-call.cil.c	0.784	0.589	0.662	0.626	0.682
elevator_spec1_product18_false-unreach-call.cil.c	0.768	0.619	0.673	0.599	0.655
elevator_spec1_product19_true-unreach-call.cil.c	0.801	0.611	0.676	0.633	0.687
elevator_spec1_product20_false-unreach-call.cil.c	0.833	0.605	0.715	0.631	0.691
elevator_spec1_product21_true-unreach-call.cil.c	0.824	0.6	0.671	0.621	0.696
elevator_spec1_product22_false-unreach-call.cil.c	0.828	0.644	0.705	0.594	0.711
elevator_spec1_product23_true-unreach-call.cil.c	0.795	0.604	0.67	0.637	0.665
elevator_spec1_product24_false-unreach-call.cil.c	0.828	0.635	0.716	0.622	0.718
elevator_spec1_product25_true-unreach-call.cil.c	0.785	0.598	0.656	0.592	0.651
elevator_spec1_product26_false-unreach-call.cil.c	0.746	0.579	0.682	0.61	0.67
elevator_spec1_product27_true-unreach-call.cil.c	0.807	0.63	0.682	0.61	0.689
elevator_spec1_product28_false-unreach-call.cil.c	0.792	0.606	0.685	0.618	0.686
elevator_spec1_product29_true-unreach-call.cil.c	0.776	0.586	0.643	0.598	0.667
elevator_spec1_product30_false-unreach-call.cil.c	0.828	0.636	0.688	0.592	0.713
elevator_spec1_product31_true-unreach-call.cil.c	0.794	0.607	0.66	0.61	0.676
elevator_spec1_product32_false-unreach-call.cil.c	0.792	0.64	0.673	0.623	0.659
elevator_spec1_productSimulator_false-unreach-call.cil.c	0.838	0.625	0.704	0.638	0.723
elevator_spec2_product01_true-unreach-call.cil.c	0.773	0.601	0.668	0.592	0.653
elevator_spec2_product03_true-unreach-call.cil.c	0.797	0.607	0.695	0.598	0.692
elevator_spec2_product09_true-unreach-call.cil.c	0.735	0.604	0.656	0.615	0.661
elevator_spec2_product11_true-unreach-call.cil.c	0.797	0.641	0.667	0.615	0.697
elevator_spec2_product17_true-unreach-call.cil.c	0.796	0.629	0.685	0.608	0.694
elevator_spec2_product18_false-unreach-call.cil.c	0.803	0.631	0.696	0.619	0.665
elevator_spec2_product19_true-unreach-call.cil.c	0.73	0.581	0.681	0.608	0.676
elevator_spec2_product20_false-unreach-call.cil.c	0.756	0.605	0.636	0.599	0.641
elevator_spec2_product21_true-unreach-call.cil.c	0.796	0.61	0.677	0.599	0.651
elevator_spec2_product22_false-unreach-call.cil.c	0.827	0.643	0.664	0.523	0.695
elevator_spec2_product23_true-unreach-call.cil.c	0.791	0.621	0.695	0.628	0.674
elevator_spec2_product24_false-unreach-call.cil.c	0.815	0.64	0.706	0.613	0.725
elevator_spec2_product25_true-unreach-call.cil.c	0.787	0.602	0.685	0.542	0.657
elevator_spec2_product26_false-unreach-call.cil.c	0.781	0.625	0.7	0.621	0.722
elevator_spec2_product27_true-unreach-call.cil.c	0.806	0.616	0.662	0.607	0.663
elevator_spec2_product28_false-unreach-call.cil.c	0.821	0.614	0.694	0.636	0.704
elevator_spec2_product29_true-unreach-call.cil.c	0.814	0.646	0.704	0.638	0.715
elevator_spec2_product30_false-unreach-call.cil.c	0.784	0.609	0.666	0.611	0.675
elevator_spec2_product31_true-unreach-call.cil.c	0.805	0.587	0.646	0.616	0.676
elevator_spec2_product32_false-unreach-call.cil.c	0.816	0.609	0.692	0.618	0.696
elevator_spec2_productSimulator_false-unreach-call.cil.c	0.81	0.643	0.712	0.647	0.691
elevator_spec3_product01_true-unreach-call.cil.c	0.76	0.591	0.654	0.58	0.623
elevator_spec3_product03_false-unreach-call.cil.c	0.752	0.555	0.628	0.583	0.636
elevator_spec3_product09_true-unreach-call.cil.c	0.757	0.593	0.663	0.609	0.676

elevator_spec3_product11_false-unreach-call.cil.c	0.765	0.61	0.641	0.605	0.652
elevator_spec3_product17_true-unreach-call.cil.c	0.769	0.592	0.661	0.614	0.656
elevator_spec3_product18_true-unreach-call.cil.c	0.812	0.619	0.696	0.63	0.7
elevator_spec3_product19_false-unreach-call.cil.c	0.832	0.629	0.699	0.626	0.694
elevator_spec3_product20_false-unreach-call.cil.c	0.832	0.603	0.706	0.638	0.705
elevator_spec3_product21_true-unreach-call.cil.c	0.821	0.632	0.708	0.631	0.679
elevator_spec3_product22_true-unreach-call.cil.c	0.772	0.595	0.678	0.619	0.663
elevator_spec3_product23_false-unreach-call.cil.c	0.806	0.603	0.686	0.625	0.688
elevator_spec3_product24_false-unreach-call.cil.c	0.83	0.642	0.698	0.623	0.729
elevator_spec3_product25_true-unreach-call.cil.c	0.794	0.628	0.676	0.634	0.704
elevator_spec3_product26_true-unreach-call.cil.c	0.818	0.622	0.711	0.635	0.716
elevator_spec3_product27_false-unreach-call.cil.c	0.822	0.63	0.705	0.603	0.711
elevator_spec3_product28_false-unreach-call.cil.c	0.775	0.607	0.687	0.613	0.681
elevator_spec3_product29_true-unreach-call.cil.c	0.767	0.62	0.661	0.601	0.675
elevator_spec3_product30_true-unreach-call.cil.c	0.804	0.59	0.702	0.61	0.693
elevator_spec3_product31_false-unreach-call.cil.c	0.809	0.609	0.696	0.628	0.674
elevator_spec3_product32_false-unreach-call.cil.c	0.816	0.616	0.683	0.626	0.668
elevator_spec3_productSimulator_false-unreach-call.cil.c	0.869	0.659	0.742	0.665	0.763
elevator_spec9_product09_true-unreach-call.cil.c	0.781	0.625	0.676	0.632	0.691
elevator_spec9_product11_true-unreach-call.cil.c	0.788	0.598	0.668	0.611	0.66
elevator_spec9_product25_true-unreach-call.cil.c	0.781	0.608	0.657	0.586	0.667
elevator_spec9_product26_false-unreach-call.cil.c	0.791	0.601	0.681	0.595	0.655
elevator_spec9_product27_true-unreach-call.cil.c	0.814	0.634	0.692	0.636	0.684
elevator_spec9_product28_false-unreach-call.cil.c	0.825	0.622	0.662	0.638	0.699
elevator_spec9_product29_true-unreach-call.cil.c	0.798	0.605	0.68	0.594	0.671
elevator_spec9_product30_false-unreach-call.cil.c	0.802	0.606	0.68	0.62	0.67
elevator_spec9_product31_true-unreach-call.cil.c	0.814	0.621	0.681	0.591	0.681
elevator_spec9_product32_false-unreach-call.cil.c	0.809	0.623	0.672	0.628	0.697
elevator_spec9_productSimulator_false-unreach-call.cil.c	0.823	0.625	0.706	0.654	0.716
email_spec0_product05_true-unreach-call.cil.c	10.655	2.333	3.094	1.935	2.536
email_spec0_product09_true-unreach-call.cil.c	10.933	2.43	3.176	2.082	2.712
email_spec0_product10_true-unreach-call.cil.c	10.869	2.427	3.196	2.006	2.659
email_spec0_product11_true-unreach-call.cil.c	11.768	3.286	4.064	2.855	3.6
email_spec0_product16_false-unreach-call.cil.c	11.799	3.331	4.12	2.872	3.627
email_spec0_product19_true-unreach-call.cil.c	11.843	3.277	4.023	2.775	3.624
email_spec0_product21_false-unreach-call.cil.c	11.925	3.449	4.252	3.105	3.745
email_spec0_product22_false-unreach-call.cil.c	11.776	3.406	4.15	3.051	3.74
email_spec0_product24_true-unreach-call.cil.c	12.016	3.42	4.234	2.953	3.767
email_spec0_product25_true-unreach-call.cil.c	12.064	3.471	4.206	3.043	3.795
email_spec0_product26_false-unreach-call.cil.c	12.124	3.506	4.328	3.215	3.919
email_spec0_product27_true-unreach-call.cil.c	12.131	3.598	4.393	3.22	3.927
email_spec0_product31_false-unreach-call.cil.c	11.803	3.344	4.096	2.875	3.651
email_spec0_product33_false-unreach-call.cil.c	11.961	3.494	4.228	3.06	3.709
email_spec0_product34_false-unreach-call.cil.c	11.987	3.502	4.259	3.133	3.824
email_spec0_product35_false-unreach-call.cil.c	12.101	3.608	4.379	3.18	3.974

email_spec0_product36_true-unreach-call.cil.c	10.951	2.596	3.35	2.132	2.809
email_spec0_product37_true-unreach-call.cil.c	11.837	3.454	4.179	3.013	3.701
email_spec0_product38_true-unreach-call.cil.c	11.964	3.468	4.224	3.011	3.76
email_spec0_product40_true-unreach-call.cil.c	11.976	3.55	4.379	3.191	3.797
email_spec0_productSimulator_false-unreach-call.cil.c	58.2	10.355	12.641	7.711	11.23
email_spec11_product03_true-unreach-call.cil.c	10.73	2.26	3.039	1.94	2.547
email_spec11_product07_true-unreach-call.cil.c	10.899	2.465	3.236	2.066	2.692
email_spec11_product08_true-unreach-call.cil.c	11.84	3.32	4.112	2.892	3.455
email_spec11_product10_true-unreach-call.cil.c	10.888	2.433	3.199	2.064	2.63
email_spec11_product15_false-unreach-call.cil.c	11.837	3.223	4.101	2.936	3.614
email_spec11_product18_true-unreach-call.cil.c	11.842	3.308	4.07	2.945	3.628
email_spec11_product20_false-unreach-call.cil.c	11.997	3.501	4.179	3.028	3.801
email_spec11_product22_false-unreach-call.cil.c	11.993	3.381	4.159	3.064	3.775
email_spec11_product23_true-unreach-call.cil.c	11.989	3.485	4.217	3.065	3.771
email_spec11_product24_true-unreach-call.cil.c	11.933	3.4	4.188	2.972	3.726
email_spec11_product26_false-unreach-call.cil.c	12.114	3.54	4.354	3.217	3.919
email_spec11_product27_true-unreach-call.cil.c	12.187	3.617	4.374	3.17	3.904
email_spec11_product30_false-unreach-call.cil.c	11.702	3.325	4.081	2.924	3.635
email_spec11_product32_false-unreach-call.cil.c	12.036	3.51	4.272	3.097	3.82
email_spec11_product33_false-unreach-call.cil.c	11.874	3.496	4.228	3.063	3.797
email_spec11_product35_false-unreach-call.cil.c	12.098	3.622	4.39	3.248	3.944
email_spec11_product36_true-unreach-call.cil.c	10.993	2.621	3.379	2.203	2.863
email_spec11_product37_true-unreach-call.cil.c	11.867	3.38	4.181	2.995	3.693
email_spec11_product39_true-unreach-call.cil.c	11.946	3.435	4.214	3.107	3.804
email_spec11_product40_true-unreach-call.cil.c	12.094	3.571	4.359	3.168	3.967
email_spec11_productSimulator_false-unreach-call.cil.c	59.094	10.265	12.575	7.686	11.3
email_spec1_product12_true-unreach-call.cil.c	11.691	3.182	3.955	2.817	3.461
email_spec1_product14_false-unreach-call.cil.c	11.658	3.376	4.131	2.932	3.665
email_spec1_product15_false-unreach-call.cil.c	11.81	3.345	4.089	2.93	3.624
email_spec1_product16_false-unreach-call.cil.c	11.761	3.302	4.072	2.948	3.626
email_spec1_product20_false-unreach-call.cil.c	11.962	3.449	4.2	3.038	3.849
email_spec1_product21_false-unreach-call.cil.c	12.016	3.485	4.251	3.099	3.844
email_spec1_product22_false-unreach-call.cil.c	11.985	3.432	4.215	2.975	3.773
email_spec1_product26_false-unreach-call.cil.c	12.118	3.589	4.36	3.17	3.957
email_spec1_product28_true-unreach-call.cil.c	11.731	3.208	3.987	2.832	3.53
email_spec1_product29_false-unreach-call.cil.c	11.776	3.392	4.131	2.982	3.698
email_spec1_product30_false-unreach-call.cil.c	11.783	3.345	4.127	2.914	3.648
email_spec1_product31_false-unreach-call.cil.c	11.839	3.335	4.078	2.961	3.638
email_spec1_product32_false-unreach-call.cil.c	12.046	3.499	4.224	3.101	3.824
email_spec1_product33_false-unreach-call.cil.c	11.881	3.475	4.239	3.069	3.735
email_spec1_product34_false-unreach-call.cil.c	12.044	3.497	4.193	3.07	3.816
email_spec1_product35_false-unreach-call.cil.c	12.127	3.536	4.272	3.247	3.954
email_spec1_productSimulator_false-unreach-call.cil.c	58.627	10.228	12.672	7.636	11.36
email_spec27_product13_true-unreach-call.cil.c	11.772	3.176	3.975	2.859	3.448
email_spec27_product17_false-unreach-call.cil.c	11.923	3.374	3.994	2.946	3.67

email_spec27_product18_false-unreach-call.cil.c	11.83	3.34	4.083	2.922	3.548
email_spec27_product19_false-unreach-call.cil.c	11.877	3.265	4.087	2.852	3.612
email_spec27_product23_false-unreach-call.cil.c	12.047	3.478	4.269	2.994	3.8
email_spec27_product24_false-unreach-call.cil.c	11.929	3.483	4.186	3.056	3.686
email_spec27_product25_false-unreach-call.cil.c	11.962	3.499	4.277	3.08	3.815
email_spec27_product27_false-unreach-call.cil.c	12.141	3.64	4.374	3.213	3.955
email_spec27_product28_true-unreach-call.cil.c	11.863	3.204	3.904	2.816	3.514
email_spec27_product29_false-unreach-call.cil.c	11.803	3.386	4.122	2.954	3.681
email_spec27_product30_false-unreach-call.cil.c	11.8	3.313	4.089	2.966	3.635
email_spec27_product31_false-unreach-call.cil.c	11.821	3.317	4.083	2.974	3.705
email_spec27_product32_false-unreach-call.cil.c	11.974	3.468	4.23	3.133	3.799
email_spec27_product33_false-unreach-call.cil.c	12.001	3.396	4.271	2.998	3.778
email_spec27_product34_false-unreach-call.cil.c	12.152	3.491	4.298	3.066	3.762
email_spec27_product35_false-unreach-call.cil.c	12.106	3.501	4.39	3.201	3.912
email_spec27_productSimulator_false-unreach-call.cil.c	59	10.268	12.756	7.752	11.27
email_spec3_product13_false-unreach-call.cil.c	11.65	3.119	3.951	2.72	3.48
email_spec3_product17_false-unreach-call.cil.c	11.785	3.382	4.145	2.901	3.666
email_spec3_product18_false-unreach-call.cil.c	11.831	3.346	4.093	2.92	3.619
email_spec3_product19_false-unreach-call.cil.c	11.822	3.36	4.101	2.962	3.643
email_spec3_product23_false-unreach-call.cil.c	12.049	3.482	4.22	3.049	3.715
email_spec3_product24_false-unreach-call.cil.c	12.031	3.421	4.18	3.034	3.769
email_spec3_product25_false-unreach-call.cil.c	11.942	3.492	4.27	3.097	3.795
email_spec3_product27_false-unreach-call.cil.c	12.057	3.625	4.399	3.196	4.003
email_spec3_product28_false-unreach-call.cil.c	11.754	3.211	3.945	2.766	3.502
email_spec3_product29_false-unreach-call.cil.c	11.883	3.302	4.162	2.985	3.597
email_spec3_product30_false-unreach-call.cil.c	11.837	3.302	4.123	2.938	3.567
email_spec3_product31_false-unreach-call.cil.c	11.833	3.315	4.09	2.948	3.636
email_spec3_product32_false-unreach-call.cil.c	12.014	3.499	4.252	3.1	3.832
email_spec3_product33_false-unreach-call.cil.c	12.036	3.524	4.268	3.084	3.788
email_spec3_product34_false-unreach-call.cil.c	11.955	3.472	4.225	3.074	3.737
email_spec3_product35_false-unreach-call.cil.c	12.272	3.643	4.414	3.241	3.961
email_spec3_productSimulator_false-unreach-call.cil.c	58.839	10.334	12.694	7.688	11.38
email_spec4_product13_true-unreach-call.cil.c	11.698	3.195	3.968	2.8	3.417
email_spec4_product17_true-unreach-call.cil.c	11.972	3.342	4.143	2.93	3.663
email_spec4_product18_false-unreach-call.cil.c	11.819	3.286	4.067	2.944	3.537
email_spec4_product19_false-unreach-call.cil.c	11.843	3.316	4.084	2.938	3.596
email_spec4_product23_false-unreach-call.cil.c	11.951	3.4	4.199	3.073	3.801
email_spec4_product24_false-unreach-call.cil.c	12.001	3.49	4.184	3.065	3.841
email_spec4_product25_false-unreach-call.cil.c	11.923	3.464	4.171	3.071	3.821
email_spec4_product27_false-unreach-call.cil.c	12.172	3.602	4.395	3.238	3.913
email_spec4_product28_true-unreach-call.cil.c	11.675	3.169	4.008	2.772	3.437
email_spec4_product29_true-unreach-call.cil.c	11.875	3.285	4.109	2.99	3.685
email_spec4_product30_false-unreach-call.cil.c	11.799	3.346	4.108	2.925	3.513
email_spec4_product31_false-unreach-call.cil.c	11.85	3.376	4.087	2.867	3.646
email_spec4_product32_false-unreach-call.cil.c	12.094	3.503	4.278	3.079	3.817

email_spec4_product33_false-unreach-call.cil.c	11.949	3.46	4.228	3.095	3.774
email_spec4_product34_false-unreach-call.cil.c	11.971	3.47	4.232	3.057	3.8
email_spec4_product35_false-unreach-call.cil.c	12.199	3.599	4.401	3.209	3.969
email_spec4_productSimulator_false-unreach-call.cil.c	58.106	10.181	12.822	7.694	11.38
email_spec6_product12_false-unreach-call.cil.c	11.701	3.213	3.965	2.797	3.47
email_spec6_product14_false-unreach-call.cil.c	11.926	3.289	4.102	2.878	3.586
email_spec6_product15_false-unreach-call.cil.c	11.824	3.334	4.081	2.866	3.623
email_spec6_product16_false-unreach-call.cil.c	11.881	3.233	4.177	2.936	3.563
email_spec6_product20_false-unreach-call.cil.c	12.001	3.458	4.337	3.087	3.784
email_spec6_product21_false-unreach-call.cil.c	11.953	3.494	4.255	3.077	3.84
email_spec6_product22_false-unreach-call.cil.c	11.949	3.465	4.223	3.072	3.764
email_spec6_product26_false-unreach-call.cil.c	12.128	3.565	4.374	3.156	3.913
email_spec6_product28_false-unreach-call.cil.c	11.762	3.189	3.932	2.837	3.505
email_spec6_product29_false-unreach-call.cil.c	12.076	3.308	4.15	2.979	3.668
email_spec6_product30_false-unreach-call.cil.c	11.967	3.273	4.089	2.868	3.654
email_spec6_product31_false-unreach-call.cil.c	11.838	3.287	4.118	2.974	3.641
email_spec6_product32_false-unreach-call.cil.c	11.98	3.518	4.293	3.119	3.808
email_spec6_product33_false-unreach-call.cil.c	12.067	3.506	4.216	3.01	3.801
email_spec6_product34_false-unreach-call.cil.c	11.956	3.457	4.277	3.108	3.806
email_spec6_product35_false-unreach-call.cil.c	12.125	3.612	4.376	3.234	3.968
email_spec6_productSimulator_false-unreach-call.cil.c	59.126	10.321	12.736	7.652	11.38
email_spec7_product13_true-unreach-call.cil.c	11.684	3.195	3.904	2.766	3.485
email_spec7_product17_true-unreach-call.cil.c	11.877	3.322	4.161	2.857	3.661
email_spec7_product18_true-unreach-call.cil.c	11.81	3.333	4.028	2.958	3.541
email_spec7_product19_true-unreach-call.cil.c	11.824	3.26	3.961	2.852	3.627
email_spec7_product23_true-unreach-call.cil.c	12.06	3.418	4.204	3.094	3.788
email_spec7_product24_true-unreach-call.cil.c	11.981	3.47	4.226	2.954	3.777
email_spec7_product25_true-unreach-call.cil.c	12.065	3.473	4.305	3.033	3.763
email_spec7_product27_true-unreach-call.cil.c	12.226	3.638	4.352	3.201	3.939
email_spec7_product28_false-unreach-call.cil.c	11.65	3.207	3.975	2.837	3.512
email_spec7_product29_false-unreach-call.cil.c	11.857	3.365	4.15	2.758	3.692
email_spec7_product30_false-unreach-call.cil.c	11.937	3.306	4.131	2.913	3.628
email_spec7_product31_false-unreach-call.cil.c	11.911	3.343	4.124	2.948	3.663
email_spec7_product32_false-unreach-call.cil.c	12.015	3.506	4.24	2.982	3.837
email_spec7_product33_false-unreach-call.cil.c	12.018	3.425	4.206	3.072	3.778
email_spec7_product34_false-unreach-call.cil.c	11.989	3.501	4.266	3.1	3.826
email_spec7_product35_false-unreach-call.cil.c	12.162	3.647	4.379	3.235	3.937
email_spec7_productSimulator_false-unreach-call.cil.c	59.115	10.336	12.647	7.616	11.33
email_spec8_product12_true-unreach-call.cil.c	11.941	3.179	3.938	2.786	3.458
email_spec8_product14_true-unreach-call.cil.c	11.922	3.375	4.208	2.969	3.65
email_spec8_product15_false-unreach-call.cil.c	11.908	3.352	4.112	2.827	3.534
email_spec8_product16_false-unreach-call.cil.c	11.724	3.338	4.117	2.936	3.504
email_spec8_product20_false-unreach-call.cil.c	12.06	3.503	4.265	3.069	3.792
email_spec8_product21_false-unreach-call.cil.c	11.992	3.484	4.293	3.093	3.733
email_spec8_product22_false-unreach-call.cil.c	11.907	3.437	4.231	3.066	3.771

email_spec8_product26_false-unreach-call.cil.c	12.161	3.591	4.392	3.219	3.864
email_spec8_product28_true-unreach-call.cil.c	11.688	3.199	3.922	2.778	3.509
email_spec8_product29_true-unreach-call.cil.c	11.949	3.382	4.124	2.929	3.666
email_spec8_product30_false-unreach-call.cil.c	11.85	3.383	4.034	2.962	3.66
email_spec8_product31_false-unreach-call.cil.c	11.845	3.365	4.141	2.965	3.682
email_spec8_product32_false-unreach-call.cil.c	12.038	3.484	4.256	3.082	3.832
email_spec8_product33_false-unreach-call.cil.c	12.003	3.487	4.174	3.061	3.769
email_spec8_product34_false-unreach-call.cil.c	11.933	3.51	4.277	3.116	3.734
email_spec8_product35_false-unreach-call.cil.c	12.258	3.624	4.377	3.155	3.911
email_spec8_productSimulator_false-unreach-call.cil.c	59.3	10.283	12.648	7.701	11.37
email_spec9_product12_true-unreach-call.cil.c	11.756	3.186	3.828	2.802	3.492
email_spec9_product14_true-unreach-call.cil.c	11.878	3.249	4.098	2.962	3.661
email_spec9_product15_false-unreach-call.cil.c	11.946	3.34	4.111	2.908	3.616
email_spec9_product16_false-unreach-call.cil.c	11.861	3.311	4.118	2.921	3.61
email_spec9_product20_false-unreach-call.cil.c	11.993	3.485	4.199	3.108	3.792
email_spec9_product21_false-unreach-call.cil.c	12.029	3.501	4.262	3.095	3.8
email_spec9_product22_false-unreach-call.cil.c	11.951	3.473	4.225	3.069	3.786
email_spec9_product26_false-unreach-call.cil.c	12.078	3.626	4.374	3.207	3.951
email_spec9_product28_true-unreach-call.cil.c	11.737	3.228	3.927	2.742	3.466
email_spec9_product29_true-unreach-call.cil.c	11.907	3.253	4.127	2.971	3.712
email_spec9_product30_false-unreach-call.cil.c	11.811	3.3	4.11	2.909	3.688
email_spec9_product31_false-unreach-call.cil.c	11.804	3.37	4.086	3.014	3.681
email_spec9_product32_false-unreach-call.cil.c	11.958	3.544	4.195	3.089	3.809
email_spec9_product33_false-unreach-call.cil.c	11.966	3.414	4.263	3.089	3.76
email_spec9_product34_false-unreach-call.cil.c	11.944	3.484	4.285	3.119	3.797
email_spec9_product35_false-unreach-call.cil.c	12.176	3.538	4.394	3.205	3.947
email_spec9_productSimulator_false-unreach-call.cil.c	58.406	10.455	12.803	7.834	11.35
minepump_spec1_product01_true-unreach-call.cil.c	0.204	0.19	0.198	0.2	0.193
minepump_spec1_product02_true-unreach-call.cil.c	0.191	0.208	0.188	0.187	0.195
minepump_spec1_product03_true-unreach-call.cil.c	0.21	0.2	0.207	0.192	0.195
minepump_spec1_product04_true-unreach-call.cil.c	0.204	0.207	0.21	0.197	0.19
minepump_spec1_product05_true-unreach-call.cil.c	0.182	0.208	0.187	0.195	0.188
minepump_spec1_product06_true-unreach-call.cil.c	0.194	0.191	0.169	0.188	0.192
minepump_spec1_product07_true-unreach-call.cil.c	0.186	0.193	0.192	0.179	0.189
minepump_spec1_product08_true-unreach-call.cil.c	0.203	0.209	0.201	0.203	0.185
minepump_spec1_product09_true-unreach-call.cil.c	0.208	0.198	0.186	0.197	0.187
minepump_spec1_product10_true-unreach-call.cil.c	0.211	0.197	0.18	0.199	0.164
minepump_spec1_product11_true-unreach-call.cil.c	0.209	0.202	0.186	0.21	0.196
minepump_spec1_product12_true-unreach-call.cil.c	0.205	0.201	0.217	0.207	0.186
minepump_spec1_product13_true-unreach-call.cil.c	0.196	0.201	0.188	0.162	0.177
minepump_spec1_product14_true-unreach-call.cil.c	0.188	0.199	0.189	0.202	0.187
minepump_spec1_product15_true-unreach-call.cil.c	0.168	0.187	0.204	0.204	0.191
minepump_spec1_product16_true-unreach-call.cil.c	0.205	0.207	0.203	0.21	0.173
minepump_spec1_product17_true-unreach-call.cil.c	0.178	0.206	0.195	0.189	0.188
minepump_spec1_product18_true-unreach-call.cil.c	0.193	0.201	0.191	0.2	0.2

minepump_spec1_product19_true-unreach-call.cil.c	0.199	0.195	0.215	0.198	0.2
minepump_spec1_product20_true-unreach-call.cil.c	0.215	0.201	0.208	0.192	0.202
minepump_spec1_product21_true-unreach-call.cil.c	0.188	0.202	0.206	0.184	0.197
minepump_spec1_product22_true-unreach-call.cil.c	0.188	0.198	0.204	0.205	0.195
minepump_spec1_product23_true-unreach-call.cil.c	0.216	0.21	0.216	0.203	0.19
minepump_spec1_product24_true-unreach-call.cil.c	0.203	0.211	0.205	0.209	0.204
minepump_spec1_product25_true-unreach-call.cil.c	0.202	0.182	0.197	0.197	0.188
minepump_spec1_product26_true-unreach-call.cil.c	0.205	0.191	0.171	0.185	0.202
minepump_spec1_product27_true-unreach-call.cil.c	0.182	0.201	0.202	0.186	0.187
minepump_spec1_product28_true-unreach-call.cil.c	0.215	0.21	0.209	0.205	0.192
minepump_spec1_product29_true-unreach-call.cil.c	0.196	0.205	0.195	0.211	0.196
minepump_spec1_product30_true-unreach-call.cil.c	0.211	0.206	0.193	0.197	0.189
minepump_spec1_product31_true-unreach-call.cil.c	0.213	0.211	0.201	0.168	0.192
minepump_spec1_product32_true-unreach-call.cil.c	0.195	0.198	0.196	0.212	0.22
minepump_spec1_product33_false-unreach-call.cil.c	0.217	0.186	0.218	0.205	0.212
minepump_spec1_product34_false-unreach-call.cil.c	0.224	0.219	0.217	0.202	0.209
minepump_spec1_product35_false-unreach-call.cil.c	0.198	0.205	0.216	0.198	0.186
minepump_spec1_product36_false-unreach-call.cil.c	0.233	0.21	0.212	0.214	0.182
minepump_spec1_product37_false-unreach-call.cil.c	0.211	0.215	0.216	0.211	0.202
minepump_spec1_product38_false-unreach-call.cil.c	0.204	0.216	0.2	0.214	0.194
minepump_spec1_product39_false-unreach-call.cil.c	0.204	0.219	0.218	0.211	0.193
minepump_spec1_product40_false-unreach-call.cil.c	0.229	0.218	0.224	0.212	0.192
minepump_spec1_product41_false-unreach-call.cil.c	0.199	0.204	0.219	0.212	0.205
minepump_spec1_product42_false-unreach-call.cil.c	0.222	0.176	0.224	0.211	0.201
minepump_spec1_product43_false-unreach-call.cil.c	0.211	0.216	0.218	0.223	0.217
minepump_spec1_product44_false-unreach-call.cil.c	0.217	0.214	0.227	0.23	0.215
minepump_spec1_product45_true-unreach-call.cil.c	0.203	0.189	0.208	0.209	0.224
minepump_spec1_product46_true-unreach-call.cil.c	0.224	0.15	0.192	0.213	0.212
minepump_spec1_product47_true-unreach-call.cil.c	0.216	0.215	0.215	0.226	0.213
minepump_spec1_product48_true-unreach-call.cil.c	0.213	0.218	0.227	0.217	0.213
minepump_spec1_product49_false-unreach-call.cil.c	0.215	0.218	0.19	0.219	0.204
minepump_spec1_product50_false-unreach-call.cil.c	0.223	0.206	0.218	0.2	0.225
minepump_spec1_product51_false-unreach-call.cil.c	0.228	0.227	0.208	0.222	0.222
minepump_spec1_product52_false-unreach-call.cil.c	0.219	0.211	0.221	0.217	0.219
minepump_spec1_product53_false-unreach-call.cil.c	0.199	0.208	0.206	0.203	0.208
minepump_spec1_product54_false-unreach-call.cil.c	0.233	0.218	0.23	0.214	0.231
minepump_spec1_product55_false-unreach-call.cil.c	0.213	0.208	0.214	0.206	0.216
minepump_spec1_product56_false-unreach-call.cil.c	0.228	0.22	0.228	0.228	0.24
minepump_spec1_product57_true-unreach-call.cil.c	0.197	0.222	0.221	0.216	0.198
minepump_spec1_product58_true-unreach-call.cil.c	0.221	0.218	0.224	0.224	0.228
minepump_spec1_product59_true-unreach-call.cil.c	0.239	0.226	0.225	0.215	0.205
minepump_spec1_product60_true-unreach-call.cil.c	0.229	0.216	0.234	0.189	0.189
minepump_spec1_product61_true-unreach-call.cil.c	0.22	0.197	0.205	0.217	0.21
minepump_spec1_product62_true-unreach-call.cil.c	0.211	0.215	0.224	0.216	0.229
minepump_spec1_product63_true-unreach-call.cil.c	0.24	0.226	0.23	0.206	0.234

minepump_spec1_product64_true-unreach-call.cil.c	0.206	0.216	0.23	0.219	0.221
minepump_spec1_productSimulator_false-unreach-call.cil.c	0.234	0.235	0.255	0.243	0.249
minepump_spec2_product01_true-unreach-call.cil.c	0.193	0.205	0.194	0.186	0.198
minepump_spec2_product02_true-unreach-call.cil.c	0.19	0.181	0.196	0.198	0.173
minepump_spec2_product03_true-unreach-call.cil.c	0.2	0.208	0.192	0.211	0.199
minepump_spec2_product04_true-unreach-call.cil.c	0.214	0.203	0.21	0.196	0.183
minepump_spec2_product05_true-unreach-call.cil.c	0.198	0.202	0.18	0.182	0.153
minepump_spec2_product06_true-unreach-call.cil.c	0.187	0.182	0.188	0.208	0.172
minepump_spec2_product07_true-unreach-call.cil.c	0.205	0.197	0.199	0.208	0.198
minepump_spec2_product08_true-unreach-call.cil.c	0.21	0.204	0.204	0.147	0.204
minepump_spec2_product09_true-unreach-call.cil.c	0.192	0.202	0.184	0.203	0.176
minepump_spec2_product10_true-unreach-call.cil.c	0.199	0.205	0.203	0.218	0.193
minepump_spec2_product11_true-unreach-call.cil.c	0.197	0.215	0.214	0.213	0.198
minepump_spec2_product12_true-unreach-call.cil.c	0.203	0.206	0.216	0.194	0.189
minepump_spec2_product13_true-unreach-call.cil.c	0.205	0.198	0.202	0.199	0.174
minepump_spec2_product14_true-unreach-call.cil.c	0.192	0.189	0.2	0.195	0.187
minepump_spec2_product15_true-unreach-call.cil.c	0.207	0.209	0.214	0.199	0.205
minepump_spec2_product16_true-unreach-call.cil.c	0.212	0.204	0.198	0.215	0.196
minepump_spec2_product17_true-unreach-call.cil.c	0.167	0.177	0.205	0.199	0.209
minepump_spec2_product18_true-unreach-call.cil.c	0.202	0.192	0.179	0.199	0.193
minepump_spec2_product19_true-unreach-call.cil.c	0.197	0.208	0.2	0.202	0.198
minepump_spec2_product20_true-unreach-call.cil.c	0.188	0.21	0.193	0.182	0.2
minepump_spec2_product21_true-unreach-call.cil.c	0.201	0.2	0.204	0.181	0.179
minepump_spec2_product22_true-unreach-call.cil.c	0.218	0.206	0.212	0.205	0.205
minepump_spec2_product23_true-unreach-call.cil.c	0.194	0.184	0.207	0.206	0.194
minepump_spec2_product24_true-unreach-call.cil.c	0.211	0.2	0.192	0.218	0.217
minepump_spec2_product25_true-unreach-call.cil.c	0.21	0.202	0.206	0.209	0.208
minepump_spec2_product26_true-unreach-call.cil.c	0.185	0.192	0.199	0.196	0.192
minepump_spec2_product27_true-unreach-call.cil.c	0.199	0.19	0.185	0.205	0.201
minepump_spec2_product28_true-unreach-call.cil.c	0.21	0.218	0.201	0.218	0.185
minepump_spec2_product29_true-unreach-call.cil.c	0.208	0.212	0.213	0.209	0.202
minepump_spec2_product30_true-unreach-call.cil.c	0.194	0.188	0.202	0.187	0.213
minepump_spec2_product31_true-unreach-call.cil.c	0.208	0.17	0.22	0.211	0.216
minepump_spec2_product32_true-unreach-call.cil.c	0.209	0.211	0.211	0.202	0.203
minepump_spec2_product33_false-unreach-call.cil.c	0.222	0.212	0.212	0.213	0.179
minepump_spec2_product34_false-unreach-call.cil.c	0.219	0.188	0.209	0.2	0.199
minepump_spec2_product35_false-unreach-call.cil.c	0.219	0.203	0.217	0.228	0.201
minepump_spec2_product36_false-unreach-call.cil.c	0.232	0.208	0.224	0.216	0.216
minepump_spec2_product37_true-unreach-call.cil.c	0.203	0.223	0.215	0.193	0.22
minepump_spec2_product38_true-unreach-call.cil.c	0.205	0.205	0.209	0.191	0.222
minepump_spec2_product39_true-unreach-call.cil.c	0.2	0.22	0.227	0.216	0.214
minepump_spec2_product40_true-unreach-call.cil.c	0.22	0.192	0.2	0.215	0.178
minepump_spec2_product41_false-unreach-call.cil.c	0.207	0.207	0.186	0.21	0.199
minepump_spec2_product42_false-unreach-call.cil.c	0.218	0.22	0.223	0.225	0.22
minepump_spec2_product43_false-unreach-call.cil.c	0.204	0.212	0.222	0.182	0.189

minepump_spec2_product44_false-unreach-call.cil.c	0.208	0.189	0.19	0.224	0.221
minepump_spec2_product45_true-unreach-call.cil.c	0.22	0.191	0.225	0.228	0.22
minepump_spec2_product46_true-unreach-call.cil.c	0.22	0.233	0.202	0.229	0.221
minepump_spec2_product47_true-unreach-call.cil.c	0.217	0.216	0.216	0.23	0.224
minepump_spec2_product48_true-unreach-call.cil.c	0.222	0.226	0.229	0.231	0.224
minepump_spec2_product49_true-unreach-call.cil.c	0.217	0.213	0.175	0.183	0.219
minepump_spec2_product50_true-unreach-call.cil.c	0.208	0.203	0.209	0.201	0.206
minepump_spec2_product51_true-unreach-call.cil.c	0.224	0.222	0.228	0.231	0.224
minepump_spec2_product52_true-unreach-call.cil.c	0.22	0.221	0.196	0.207	0.227
minepump_spec2_product53_true-unreach-call.cil.c	0.232	0.225	0.192	0.224	0.207
minepump_spec2_product54_true-unreach-call.cil.c	0.222	0.224	0.19	0.175	0.197
minepump_spec2_product55_true-unreach-call.cil.c	0.215	0.22	0.23	0.228	0.201
minepump_spec2_product56_true-unreach-call.cil.c	0.239	0.194	0.209	0.207	0.225
minepump_spec2_product57_true-unreach-call.cil.c	0.224	0.207	0.222	0.193	0.204
minepump_spec2_product58_true-unreach-call.cil.c	0.222	0.217	0.223	0.195	0.216
minepump_spec2_product59_true-unreach-call.cil.c	0.215	0.225	0.204	0.229	0.227
minepump_spec2_product60_true-unreach-call.cil.c	0.232	0.241	0.205	0.236	0.24
minepump_spec2_product61_true-unreach-call.cil.c	0.215	0.217	0.212	0.219	0.21
minepump_spec2_product62_true-unreach-call.cil.c	0.206	0.219	0.214	0.204	0.208
minepump_spec2_product63_true-unreach-call.cil.c	0.232	0.215	0.223	0.231	0.209
minepump_spec2_product64_true-unreach-call.cil.c	0.225	0.234	0.219	0.229	0.22
minepump_spec2_productSimulator_false-unreach-call.cil.c	0.226	0.211	0.242	0.228	0.254
minepump_spec3_product01_false-unreach-call.cil.c	0.166	0.182	0.21	0.187	0.185
minepump_spec3_product02_false-unreach-call.cil.c	0.183	0.204	0.198	0.173	0.199
minepump_spec3_product03_false-unreach-call.cil.c	0.212	0.205	0.195	0.183	0.213
minepump_spec3_product04_false-unreach-call.cil.c	0.197	0.179	0.206	0.193	0.2
minepump_spec3_product05_false-unreach-call.cil.c	0.211	0.195	0.204	0.196	0.179
minepump_spec3_product06_false-unreach-call.cil.c	0.208	0.188	0.209	0.201	0.218
minepump_spec3_product07_false-unreach-call.cil.c	0.187	0.203	0.2	0.204	0.187
minepump_spec3_product08_false-unreach-call.cil.c	0.207	0.198	0.196	0.191	0.196
minepump_spec3_product09_false-unreach-call.cil.c	0.201	0.182	0.184	0.197	0.19
minepump_spec3_product10_false-unreach-call.cil.c	0.188	0.213	0.21	0.195	0.207
minepump_spec3_product11_false-unreach-call.cil.c	0.207	0.189	0.189	0.189	0.181
minepump_spec3_product12_false-unreach-call.cil.c	0.193	0.186	0.197	0.199	0.185
minepump_spec3_product13_false-unreach-call.cil.c	0.203	0.186	0.214	0.188	0.203
minepump_spec3_product14_false-unreach-call.cil.c	0.199	0.198	0.191	0.191	0.203
minepump_spec3_product15_false-unreach-call.cil.c	0.217	0.213	0.214	0.182	0.209
minepump_spec3_product16_false-unreach-call.cil.c	0.187	0.197	0.219	0.204	0.202
minepump_spec3_product17_false-unreach-call.cil.c	0.201	0.197	0.204	0.188	0.193
minepump_spec3_product18_false-unreach-call.cil.c	0.191	0.191	0.208	0.208	0.176
minepump_spec3_product19_false-unreach-call.cil.c	0.179	0.195	0.192	0.198	0.187
minepump_spec3_product20_false-unreach-call.cil.c	0.206	0.202	0.202	0.198	0.199
minepump_spec3_product21_false-unreach-call.cil.c	0.177	0.186	0.21	0.176	0.203
minepump_spec3_product22_false-unreach-call.cil.c	0.206	0.214	0.204	0.192	0.201
minepump_spec3_product23_false-unreach-call.cil.c	0.205	0.203	0.206	0.207	0.183

minepump_spec3_product24_false-unreach-call.cil.c	0.205	0.21	0.204	0.206	0.19
minepump_spec3_product25_false-unreach-call.cil.c	0.201	0.203	0.188	0.184	0.196
minepump_spec3_product26_false-unreach-call.cil.c	0.207	0.205	0.213	0.194	0.19
minepump_spec3_product27_false-unreach-call.cil.c	0.188	0.195	0.202	0.186	0.186
minepump_spec3_product28_false-unreach-call.cil.c	0.169	0.189	0.211	0.196	0.198
minepump_spec3_product29_false-unreach-call.cil.c	0.19	0.202	0.204	0.186	0.185
minepump_spec3_product30_false-unreach-call.cil.c	0.218	0.195	0.192	0.208	0.202
minepump_spec3_product31_false-unreach-call.cil.c	0.194	0.216	0.178	0.196	0.205
minepump_spec3_product32_false-unreach-call.cil.c	0.222	0.216	0.216	0.223	0.214
minepump_spec3_product33_true-unreach-call.cil.c	0.198	0.198	0.22	0.222	0.207
minepump_spec3_product34_true-unreach-call.cil.c	0.185	0.171	0.194	0.222	0.202
minepump_spec3_product35_false-unreach-call.cil.c	0.214	0.182	0.214	0.228	0.195
minepump_spec3_product36_false-unreach-call.cil.c	0.195	0.213	0.226	0.211	0.201
minepump_spec3_product37_true-unreach-call.cil.c	0.218	0.196	0.224	0.218	0.198
minepump_spec3_product38_true-unreach-call.cil.c	0.225	0.212	0.213	0.225	0.204
minepump_spec3_product39_false-unreach-call.cil.c	0.225	0.219	0.225	0.204	0.222
minepump_spec3_product40_false-unreach-call.cil.c	0.217	0.212	0.219	0.204	0.219
minepump_spec3_product41_true-unreach-call.cil.c	0.221	0.215	0.212	0.217	0.213
minepump_spec3_product42_true-unreach-call.cil.c	0.205	0.226	0.214	0.209	0.214
minepump_spec3_product43_false-unreach-call.cil.c	0.216	0.215	0.21	0.214	0.213
minepump_spec3_product44_false-unreach-call.cil.c	0.225	0.168	0.217	0.216	0.212
minepump_spec3_product45_true-unreach-call.cil.c	0.214	0.216	0.209	0.227	0.22
minepump_spec3_product46_true-unreach-call.cil.c	0.214	0.225	0.197	0.208	0.22
minepump_spec3_product47_false-unreach-call.cil.c	0.211	0.202	0.216	0.22	0.202
minepump_spec3_product48_false-unreach-call.cil.c	0.23	0.185	0.216	0.202	0.221
minepump_spec3_product49_true-unreach-call.cil.c	0.215	0.175	0.225	0.207	0.221
minepump_spec3_product50_true-unreach-call.cil.c	0.203	0.229	0.21	0.183	0.182
minepump_spec3_product51_false-unreach-call.cil.c	0.211	0.222	0.205	0.213	0.21
minepump_spec3_product52_false-unreach-call.cil.c	0.212	0.215	0.229	0.206	0.204
minepump_spec3_product53_true-unreach-call.cil.c	0.211	0.209	0.23	0.221	0.215
minepump_spec3_product54_true-unreach-call.cil.c	0.222	0.215	0.205	0.219	0.214
minepump_spec3_product55_false-unreach-call.cil.c	0.23	0.228	0.217	0.232	0.241
minepump_spec3_product56_false-unreach-call.cil.c	0.241	0.223	0.218	0.223	0.221
minepump_spec3_product57_true-unreach-call.cil.c	0.22	0.191	0.216	0.205	0.213
minepump_spec3_product58_true-unreach-call.cil.c	0.21	0.203	0.215	0.214	0.215
minepump_spec3_product59_false-unreach-call.cil.c	0.215	0.226	0.223	0.218	0.213
minepump_spec3_product60_false-unreach-call.cil.c	0.229	0.217	0.196	0.228	0.216
minepump_spec3_product61_true-unreach-call.cil.c	0.22	0.195	0.215	0.211	0.213
minepump_spec3_product62_true-unreach-call.cil.c	0.23	0.192	0.229	0.219	0.222
minepump_spec3_product63_false-unreach-call.cil.c	0.24	0.212	0.229	0.23	0.211
minepump_spec3_product64_false-unreach-call.cil.c	0.216	0.219	0.231	0.218	0.215
minepump_spec3_productSimulator_false-unreach-call.cil.c	0.252	0.233	0.2	0.247	0.26
minepump_spec4_product01_true-unreach-call.cil.c	0.201	0.185	0.207	0.202	0.197
minepump_spec4_product02_true-unreach-call.cil.c	0.209	0.203	0.206	0.2	0.201
minepump_spec4_product03_true-unreach-call.cil.c	0.175	0.202	0.209	0.205	0.197

minepump_spec4_product04_true-unreach-call.cil.c	0.199	0.196	0.191	0.186	0.203
minepump_spec4_product05_true-unreach-call.cil.c	0.198	0.204	0.202	0.187	0.192
minepump_spec4_product06_true-unreach-call.cil.c	0.164	0.19	0.208	0.201	0.179
minepump_spec4_product07_true-unreach-call.cil.c	0.173	0.202	0.199	0.209	0.195
minepump_spec4_product08_true-unreach-call.cil.c	0.204	0.184	0.208	0.205	0.185
minepump_spec4_product09_true-unreach-call.cil.c	0.199	0.196	0.192	0.197	0.174
minepump_spec4_product10_true-unreach-call.cil.c	0.188	0.17	0.183	0.186	0.188
minepump_spec4_product11_true-unreach-call.cil.c	0.209	0.21	0.2	0.213	0.199
minepump_spec4_product12_true-unreach-call.cil.c	0.204	0.208	0.198	0.201	0.196
minepump_spec4_product13_true-unreach-call.cil.c	0.201	0.19	0.191	0.182	0.182
minepump_spec4_product14_true-unreach-call.cil.c	0.203	0.214	0.201	0.193	0.202
minepump_spec4_product15_true-unreach-call.cil.c	0.202	0.211	0.211	0.202	0.185
minepump_spec4_product16_true-unreach-call.cil.c	0.194	0.196	0.203	0.2	0.209
minepump_spec4_product17_true-unreach-call.cil.c	0.2	0.198	0.195	0.199	0.186
minepump_spec4_product18_true-unreach-call.cil.c	0.191	0.205	0.196	0.207	0.206
minepump_spec4_product19_true-unreach-call.cil.c	0.212	0.204	0.209	0.207	0.206
minepump_spec4_product20_true-unreach-call.cil.c	0.193	0.184	0.206	0.199	0.182
minepump_spec4_product21_true-unreach-call.cil.c	0.201	0.205	0.199	0.174	0.193
minepump_spec4_product22_true-unreach-call.cil.c	0.203	0.196	0.202	0.189	0.196
minepump_spec4_product23_true-unreach-call.cil.c	0.211	0.201	0.205	0.203	0.211
minepump_spec4_product24_true-unreach-call.cil.c	0.217	0.198	0.201	0.205	0.209
minepump_spec4_product25_true-unreach-call.cil.c	0.205	0.2	0.205	0.183	0.185
minepump_spec4_product26_true-unreach-call.cil.c	0.211	0.215	0.2	0.201	0.199
minepump_spec4_product27_true-unreach-call.cil.c	0.207	0.183	0.182	0.198	0.154
minepump_spec4_product28_true-unreach-call.cil.c	0.187	0.186	0.19	0.21	0.2
minepump_spec4_product29_true-unreach-call.cil.c	0.204	0.202	0.201	0.202	0.2
minepump_spec4_product30_true-unreach-call.cil.c	0.204	0.197	0.19	0.205	0.189
minepump_spec4_product31_true-unreach-call.cil.c	0.216	0.213	0.186	0.201	0.189
minepump_spec4_product32_true-unreach-call.cil.c	0.214	0.208	0.192	0.2	0.154
minepump_spec4_product33_false-unreach-call.cil.c	0.221	0.192	0.213	0.198	0.191
minepump_spec4_product34_false-unreach-call.cil.c	0.205	0.209	0.175	0.194	0.193
minepump_spec4_product35_false-unreach-call.cil.c	0.207	0.209	0.205	0.201	0.198
minepump_spec4_product36_false-unreach-call.cil.c	0.207	0.209	0.23	0.226	0.219
minepump_spec4_product37_false-unreach-call.cil.c	0.205	0.192	0.214	0.217	0.188
minepump_spec4_product38_false-unreach-call.cil.c	0.22	0.203	0.209	0.203	0.198
minepump_spec4_product39_false-unreach-call.cil.c	0.214	0.227	0.22	0.208	0.204
minepump_spec4_product40_false-unreach-call.cil.c	0.226	0.202	0.219	0.215	0.216
minepump_spec4_product41_false-unreach-call.cil.c	0.208	0.212	0.214	0.197	0.198
minepump_spec4_product42_false-unreach-call.cil.c	0.229	0.202	0.203	0.187	0.203
minepump_spec4_product43_false-unreach-call.cil.c	0.213	0.206	0.216	0.207	0.223
minepump_spec4_product44_false-unreach-call.cil.c	0.225	0.223	0.22	0.206	0.221
minepump_spec4_product45_false-unreach-call.cil.c	0.214	0.216	0.206	0.2	0.216
minepump_spec4_product46_false-unreach-call.cil.c	0.214	0.214	0.201	0.222	0.204
minepump_spec4_product47_false-unreach-call.cil.c	0.226	0.229	0.227	0.226	0.232
minepump_spec4_product48_false-unreach-call.cil.c	0.228	0.221	0.219	0.226	0.214

minepump_spec4_product49_true-unreach-call.cil.c	0.22	0.212	0.217	0.219	0.195
minepump_spec4_product50_true-unreach-call.cil.c	0.219	0.212	0.223	0.219	0.213
minepump_spec4_product51_true-unreach-call.cil.c	0.225	0.179	0.219	0.213	0.211
minepump_spec4_product52_true-unreach-call.cil.c	0.234	0.209	0.226	0.229	0.23
minepump_spec4_product53_true-unreach-call.cil.c	0.2	0.222	0.224	0.222	0.21
minepump_spec4_product54_true-unreach-call.cil.c	0.212	0.19	0.216	0.21	0.215
minepump_spec4_product55_true-unreach-call.cil.c	0.229	0.204	0.225	0.221	0.226
minepump_spec4_product56_true-unreach-call.cil.c	0.22	0.22	0.219	0.198	0.205
minepump_spec4_product57_true-unreach-call.cil.c	0.196	0.218	0.222	0.224	0.223
minepump_spec4_product58_true-unreach-call.cil.c	0.229	0.211	0.215	0.22	0.215
minepump_spec4_product59_true-unreach-call.cil.c	0.229	0.224	0.215	0.213	0.224
minepump_spec4_product60_true-unreach-call.cil.c	0.222	0.204	0.214	0.218	0.21
minepump_spec4_product61_true-unreach-call.cil.c	0.222	0.212	0.187	0.21	0.227
minepump_spec4_product62_true-unreach-call.cil.c	0.24	0.217	0.227	0.221	0.239
minepump_spec4_product63_true-unreach-call.cil.c	0.236	0.214	0.228	0.225	0.209
minepump_spec4_product64_true-unreach-call.cil.c	0.22	0.2	0.214	0.23	0.22
minepump_spec4_productSimulator_false-unreach-call.cil.c	0.239	0.226	0.235	0.241	0.236
minepump_spec5_product01_true-unreach-call.cil.c	0.193	0.181	0.173	0.193	0.175
minepump_spec5_product02_true-unreach-call.cil.c	0.197	0.207	0.187	0.18	0.199
minepump_spec5_product03_true-unreach-call.cil.c	0.192	0.194	0.214	0.211	0.195
minepump_spec5_product04_true-unreach-call.cil.c	0.2	0.191	0.185	0.208	0.203
minepump_spec5_product05_true-unreach-call.cil.c	0.195	0.197	0.195	0.201	0.205
minepump_spec5_product06_true-unreach-call.cil.c	0.203	0.207	0.184	0.191	0.181
minepump_spec5_product07_true-unreach-call.cil.c	0.2	0.186	0.197	0.201	0.181
minepump_spec5_product08_true-unreach-call.cil.c	0.212	0.203	0.215	0.221	0.187
minepump_spec5_product09_true-unreach-call.cil.c	0.204	0.2	0.176	0.2	0.204
minepump_spec5_product10_true-unreach-call.cil.c	0.207	0.2	0.174	0.196	0.173
minepump_spec5_product11_true-unreach-call.cil.c	0.201	0.195	0.214	0.209	0.194
minepump_spec5_product12_true-unreach-call.cil.c	0.195	0.2	0.193	0.201	0.183
minepump_spec5_product13_true-unreach-call.cil.c	0.161	0.198	0.193	0.145	0.189
minepump_spec5_product14_true-unreach-call.cil.c	0.202	0.197	0.204	0.191	0.188
minepump_spec5_product15_true-unreach-call.cil.c	0.195	0.196	0.214	0.183	0.19
minepump_spec5_product16_true-unreach-call.cil.c	0.212	0.193	0.207	0.2	0.19
minepump_spec5_product17_true-unreach-call.cil.c	0.204	0.191	0.188	0.174	0.184
minepump_spec5_product18_true-unreach-call.cil.c	0.209	0.207	0.212	0.208	0.198
minepump_spec5_product19_true-unreach-call.cil.c	0.202	0.194	0.212	0.178	0.205
minepump_spec5_product20_true-unreach-call.cil.c	0.204	0.199	0.21	0.211	0.174
minepump_spec5_product21_true-unreach-call.cil.c	0.199	0.188	0.207	0.175	0.184
minepump_spec5_product22_true-unreach-call.cil.c	0.21	0.215	0.196	0.191	0.204
minepump_spec5_product23_true-unreach-call.cil.c	0.221	0.217	0.198	0.186	0.205
minepump_spec5_product24_true-unreach-call.cil.c	0.216	0.21	0.213	0.195	0.215
minepump_spec5_product25_true-unreach-call.cil.c	0.209	0.209	0.181	0.184	0.201
minepump_spec5_product26_true-unreach-call.cil.c	0.185	0.211	0.178	0.218	0.204
minepump_spec5_product27_true-unreach-call.cil.c	0.202	0.204	0.182	0.182	0.202
minepump_spec5_product28_true-unreach-call.cil.c	0.207	0.213	0.205	0.2	0.191

minepump_spec5_product29_true-unreach-call.cil.c	0.21	0.21	0.194	0.212	0.19
minepump_spec5_product30_true-unreach-call.cil.c	0.2	0.182	0.216	0.193	0.196
minepump_spec5_product31_true-unreach-call.cil.c	0.19	0.196	0.212	0.198	0.19
minepump_spec5_product32_true-unreach-call.cil.c	0.199	0.21	0.214	0.18	0.206
minepump_spec5_product33_true-unreach-call.cil.c	0.222	0.208	0.215	0.197	0.212
minepump_spec5_product34_true-unreach-call.cil.c	0.212	0.212	0.21	0.203	0.224
minepump_spec5_product35_true-unreach-call.cil.c	0.236	0.21	0.22	0.192	0.222
minepump_spec5_product36_true-unreach-call.cil.c	0.226	0.208	0.231	0.188	0.171
minepump_spec5_product37_true-unreach-call.cil.c	0.22	0.202	0.214	0.213	0.185
minepump_spec5_product38_true-unreach-call.cil.c	0.201	0.218	0.215	0.197	0.204
minepump_spec5_product39_true-unreach-call.cil.c	0.228	0.227	0.21	0.225	0.224
minepump_spec5_product40_true-unreach-call.cil.c	0.222	0.228	0.22	0.216	0.218
minepump_spec5_product41_true-unreach-call.cil.c	0.223	0.211	0.211	0.202	0.202
minepump_spec5_product42_true-unreach-call.cil.c	0.223	0.207	0.222	0.211	0.207
minepump_spec5_product43_true-unreach-call.cil.c	0.204	0.207	0.212	0.218	0.217
minepump_spec5_product44_true-unreach-call.cil.c	0.214	0.219	0.232	0.215	0.227
minepump_spec5_product45_true-unreach-call.cil.c	0.212	0.197	0.227	0.216	0.204
minepump_spec5_product46_true-unreach-call.cil.c	0.198	0.214	0.208	0.204	0.211
minepump_spec5_product47_true-unreach-call.cil.c	0.219	0.229	0.206	0.22	0.236
minepump_spec5_product48_true-unreach-call.cil.c	0.22	0.221	0.212	0.221	0.225
minepump_spec5_product49_true-unreach-call.cil.c	0.226	0.225	0.208	0.188	0.198
minepump_spec5_product50_true-unreach-call.cil.c	0.221	0.209	0.229	0.207	0.198
minepump_spec5_product51_true-unreach-call.cil.c	0.222	0.19	0.234	0.233	0.199
minepump_spec5_product52_true-unreach-call.cil.c	0.209	0.204	0.219	0.232	0.185
minepump_spec5_product53_true-unreach-call.cil.c	0.204	0.221	0.211	0.223	0.208
minepump_spec5_product54_true-unreach-call.cil.c	0.218	0.22	0.21	0.2	0.194
minepump_spec5_product55_true-unreach-call.cil.c	0.217	0.227	0.24	0.227	0.218
minepump_spec5_product56_true-unreach-call.cil.c	0.242	0.202	0.235	0.217	0.209
minepump_spec5_product57_true-unreach-call.cil.c	0.214	0.212	0.212	0.214	0.225
minepump_spec5_product58_true-unreach-call.cil.c	0.215	0.215	0.216	0.217	0.216
minepump_spec5_product59_true-unreach-call.cil.c	0.235	0.222	0.227	0.205	0.216
minepump_spec5_product60_true-unreach-call.cil.c	0.225	0.235	0.235	0.216	0.233
minepump_spec5_product61_true-unreach-call.cil.c	0.212	0.224	0.231	0.223	0.213
minepump_spec5_product62_true-unreach-call.cil.c	0.232	0.222	0.227	0.232	0.232
minepump_spec5_product63_true-unreach-call.cil.c	0.233	0.209	0.209	0.217	0.22
minepump_spec5_product64_true-unreach-call.cil.c	0.232	0.235	0.227	0.204	0.225
minepump_spec5_productSimulator_true-unreach-call.cil.c	0.232	0.234	0.244	0.246	0.212