The Effect of VSIDS on SAT Solver Performance

Jaeheon Yi
University of California, Santa Cruz
E-mail jaeheon@cs.ucsc.edu.

December 11, 2007

Abstract

VSIDS, a popular decision heuristic introduced in CHAFF, is compared against several simple heuristics to evaluate its effectiveness on a given set of benchmarks.

Key Words

VSIDS, CHAFF, MINISAT, branching heuristics, propositional satisfiability

1 Introduction

We assume that the reader is familiar with the problem of propositional satisfiability and the DPLL family of search algorithms. CHAFF [1] is a well-known satisfiability solver that employs several heuristics to achieve high performance in a wide variety of benchmarks. However, the paper does not discuss the *individual* contributions of each heuristic toward achieving this high performance. In this paper, we perform controlled experiments using MINISAT [2] to assess the effectiveness of the VSIDS branching heuristic on performance.

2 Minisat

MINISAT is a popular high-performance satisfiability solver that implements many well-known performance heuristics in a concise manner. It has won several prizes in the SAT 2005 [3] competition, and has the advantage of being open-source. MINISAT uses a version of the VSIDS branching heuristic, described in Section 3. We refer the reader to [4] for implementation details.

3 Branching Heuristics

In a DPLL-style algorithm, the search for a model proceeds by choosing a variable upon which to branch. The choice of such a variable is made by a *branching heuristic*. We describe some of the branching heuristics used in our experiment.

3.1 VSIDS

CHAFF introduced the VSIDS (Variable State Independent, Decaying Sum) heuristic in [1], and is defined as follows:

1. Each literal has a counter, initialized to 0
2. When a clause is added to the clause database, the counter associated with each literal in the clause is incremented. The counter keeps track of how often the literal is used.
3. The unassigned literal with highest counter is chosen at each decision point.
4. Ties are broken by random choice.
5. All counters are divided by a constant, periodically.

The heuristic is independent of variable state, because the counter value for each literal is independent of the counter value for other literals. It is believed that the VSIDS heuristic contributes to the high performance of the CHAFF solver partially through the fifth item [1]; namely that literals in "old" clauses drop in value over time, thus ensuring that recent clauses are resolved first.

MINISAT's version of VSIDS makes some simple changes to the original formula, as follows:

1. Each *variable* has a counter.
2. Instead of a decay factor, variable counters are "bumped" with larger and larger values in a floating point representation. When very large values are encountered, all counters are scaled down.
3. 2% of the time, a random decision is made instead. This factor is set at run-time.

In addition, MINISAT's implementation of VSIDS ("REF") always keeps the variables in order by placing them in a array-based priority queue.

3.2 RAND

RAND is a simple heuristic that has been shown to be relatively effective [5] at a low cost, both in time and space.

In our implementation of RAND in MINISAT, we make no attempt to optimize the data structures for using RAND. Instead we reuse the bookkeeping mechanism from VSIDS, and simply boost a random variable's counter to have the greatest value in the priority queue. Although this incurs overhead that might not have necessarily occurred in a fully optimized implementation of RAND, in Section 4.2 we note that time is not the only consideration when comparing heuristics.

3.3 LINSCAN and REVLINSCAN

We also include two implementationally simple branching heuristics to make things interesting. MINISAT occasionally chooses a variable at random from the variable assignment array, at random times. If the chosen variable has already been assigned some value, then it defaults to using VSIDS.

LINSCAN makes use of the variable assignment array by scanning forward from the 0th index to find an unassigned variable. Although this can be very inefficient, we cannot simply make it more efficient by recording the previously scanned position and proceed forward from there, because MINISAT performs backtracking and unassigns previously assigned variables.

REVLINSCAN scans backward from the last index in the variable assignment array to choose a branching variable.

In both of these implementations, we again make no attempt to optimize MINISAT's other data structures to support these heuristics. In this case, the machinery involving the update of variable counters is kept intact.

4 Methodology

4.1 Measurements

Timing measurements were performed on a 4-core Linux compute server with 8 GB of memory. Each benchmark was measured with an arbitrary timeout limit of 1200 seconds.

4.2 Benchmarking Methodologies

This paper poses the question, "Is the VSIDS heuristic any good?" A review of the relevant literature indicates that the answer may be, "It Depends on Whom You Ask." In fact, the state of the art is such that there is no standard way to compare different heuristics or implementations of satisfiability solvers. One can select among various metrics, and choose among various benchmark tests to show "improvement" over previous work [6].

In [7], the authors point out that very often, little is known about when or why a heuristic works well when it does. They provide an example of a factorized experimental design and provide statistical evidence that one heuristic is better than another, in 10 different problem classes.

In [8], the authors use CPU execution time and the number of branches to compare heuristics, over parameterized problem classes of random and structured CNF formulas. In addition, they advocate the use of several statistics to provide a balanced view of an algorithm's behavior, and provide guidelines on how to avoid common statistical mistakes.

In [5], Marques-Silva compares different heuristics and implementations by measuring their CPU execution times, the number of aborted instances after reaching a timeout limit, and the number of branches taken. When comparing branching heuristics, he also compares the number of backtracks performed.

In [1], the authors use execution time as the only criterion for comparison, and claim that using the VSIDS heuristic improved performance by an order of magnitude on problems found to be difficult with CHAFF.

In [9], robustness is defined as being able to solve more benchmark problems than CHAFF within a given period

of time. The authors claim that BERKMIN is more robust than CHAFF, on the grounds that a difficult benchmark suite was completely solved using BERKMIN, while CHAFF could not solve some of the benchmarks within the given period of time.

In [10], Zarpas observes discrepancies between experimental results found in the literature and experimental results performed by his research group. Because modern solvers are highly dependent upon heuristics, he attempts to define guidelines for a more consistent benchmarking methodology. Zarpas advocates using benchmarks that are relevant to the domain of interest, using very long timeouts, and using several metrics for comparison.

4.3 Benchmarks

We ran our four heuristics on 47 benchmarks, measuring the average time it took to complete a benchmark, the number of conflicts reported, and the number of decisions made.

The composition of the benchmarks is as follows: 12 from Pigeonhole, 20 from CertBench, 8 from SAT 2007, 7 from SAT 2005.

5 Results

5.1 Metric Correlation

In Figures 1 to 4, we plot number of conflicts versus time and number of decisions versus time to see if they have any correlation. If a strong correlation exists for some metric and time, then that metric can be considered a predictor of performance.

However we have found that both number of conflicts and number of decisions are not effective at predicting execution time. On the other hand, this conclusion might not hold for a specific subset of benchmarks with similar characteristics.

5.2 Comparison of Speedup

We assess the relative speedup of various heuristics with respect to VSIDS (Figure 5). The data points were obtained by taking the logarithm of the normalized time against VSIDS. *Positive* values indicate that the heuristic performed *slower* than VSIDS on a particular benchmark; negative values indicate a speedup.

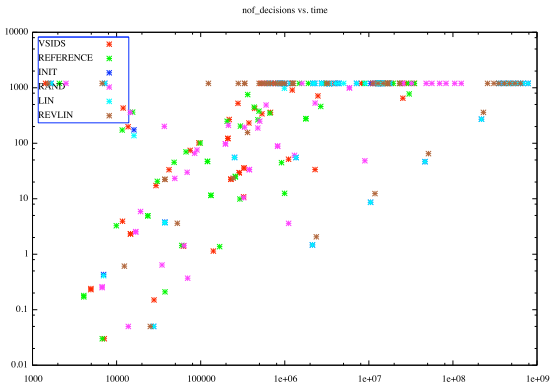

Figure 1: # of decisions vs. time, all heuristics

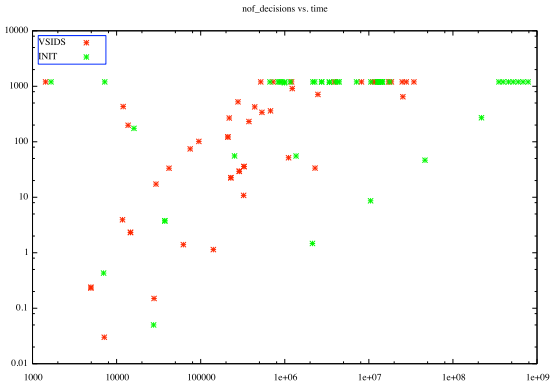

Figure 2: # of decisions vs. time, VSIDS and INIT

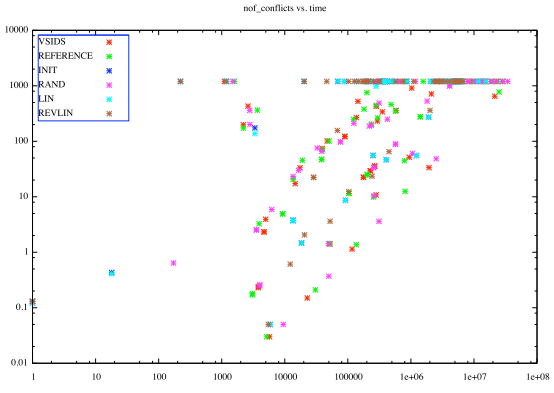

Figure 3: # of conflicts vs. time, all heuristics

We have found that VSIDS is comparable to RAND in terms of speedup; that is, it provides essentially the same performance for a given benchmark. We can also observe that RAND is slightly slower than VSIDS, but not by much. The other heuristics, namely INIT, LIN, and REVLIN, are either faster or much slower than VSIDS.

5.3 Comparison of Metrics

We compare various metrics for two heuristics on scatterplots, in Figures 6 to 8. With VSIDS and INIT, we see that there is little correspondence in running time between the two heuristics, for a given benchmark. On the other hand, for VSIDS and RAND, we see that they have roughly the equivalent amount of running time.

When we compare number of conflicts for VSIDS and INIT, we see that there is a loose correlation for a given benchmark. Again, however, the number of conflicts is more similar between VSIDS and RAND.

We reach the same conclusion for number of decisions: RAND behaves more like VSIDS than INIT does.

6 Conclusion

Although it is difficult to draw any kind of general conclusion from the experiments that we have run, we can say several things. First, number of decisions and number of conflicts are poor predictors of performance, as measured in execution time. Second, VSIDS has roughly equivalent performance and similar characteristics to randomized variable selection.

References

[1] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an Efficient SAT Solver. In *Proceedings of the 38th Design Automation Conference (DAC'01)*, 2001.

[2] Niklas En and Niklas Srensson. An extensible sat-solver. In Enrico Giunchiglia and Armando Tacchella, editors, *SAT*, volume 2919 of *Lecture Notes in Computer Science*, pages 502–518. Springer, 2003.

[3] http://www.satcompetition.org.

[4] http://minisat.se.

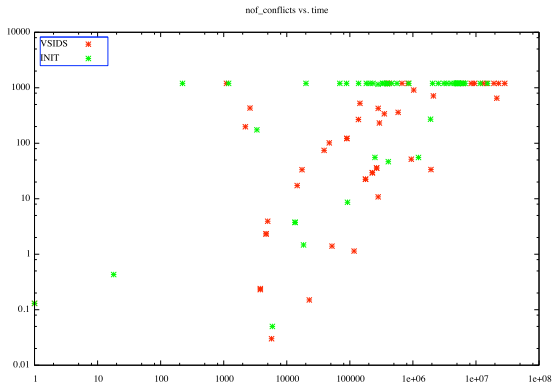

Figure 4: # of conflicts vs. time, VSIDS and INIT

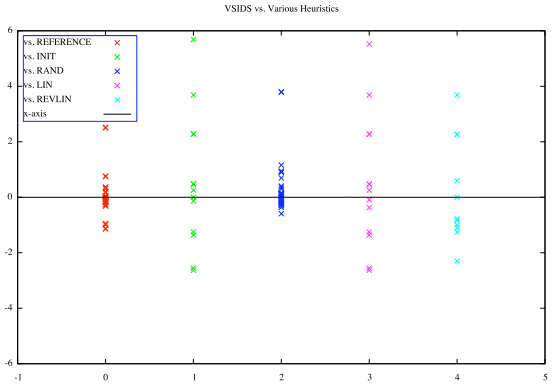

Figure 5: Speedup Comparison

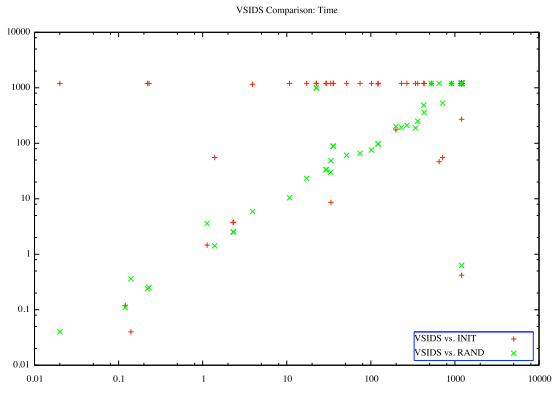

Figure 6: VSIDS vs. INIT and RAND, for time

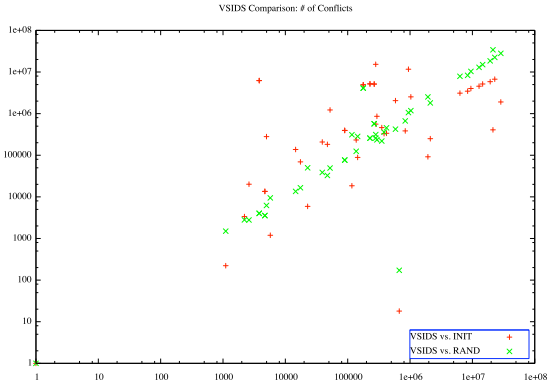

Figure 7: VSIDS vs. INIT and RAND, for # of conflicts

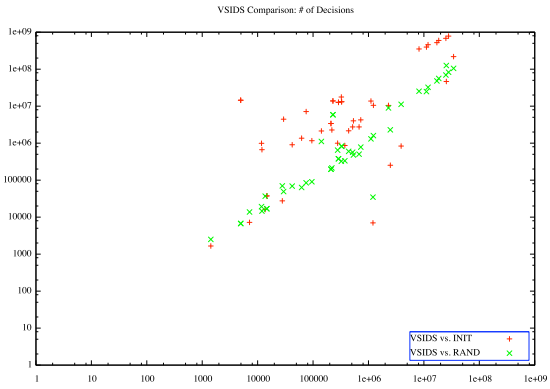

Figure 8: VSIDS vs. INIT and RAND, for # of decisions

[5] J. Marques-Silva. The Impact of Branching Heuristics in Propositional Satisfiability Algorithms. *Progress in Artificial Intelligence: 9th Portuguese Conference on Artificial Intelligence, EPIA'99, Évora, Portugal, September 21-24, 1999: Proceedings*, 1999.

[6] J.N. Hooker. Testing heuristics: We have it all wrong. *Journal of Heuristics*, 1(1):33–42, 1995.

[7] JN Hooker and V. Vinay. Branching rules for satisfiability. *Journal of Automated Reasoning*, 15(3):359–383, 1995.

[8] A. Van Gelder, Y. Tsuji, Santa Cruz University of California, and Computer Research Laboratory. *Satisfiability Testing with More Reasoning and Less Guessing*. Computer Research Laboratory, University of California, Santa Cruz, 1995.

[9] E. Goldberg and Y. Novikov. BerkMin: A fast and robust Sat-solver. *Discrete Applied Mathematics*, 155(12):1549–1561, 2007.

[10] E. Zarpas. Benchmarking SAT Solvers for Bounded Model Checking. *Theory And Applications of Satisfiability Testing: 8th International Conference, SAT 2005, StAndrews, UK, June 19-23, 2005: Proceedings*, 2005.

A Data - Execution Time

We have gathered the following data. The first table is execution time in seconds. The timeout limit is 1200 seconds.

	VSIDS	REF	INIT	RAND	LIN	REVLIN
PigeonHole						
ph09.cnf	0.14	0.20	0.04	0.36	0.04	0.04
ph10.cnf	1.13	1.36	1.46	3.59	1.46	2.05
ph11.cnf	33.53	12.51	8.62	48.36	8.61	12.36
ph12.cnf	648.23	774.67	46.56	-	46.62	64.97
ph13.cnf	-	-	271.09	-	270.91	360.26
ph14.cnf	-	-	-	-	-	-
ph15.cnf	-	-	-	-	-	-
ph16.cnf	-	-	-	-	-	-
ph17.cnf	-	-	-	-	-	-
ph18.cnf	-	-	-	-	-	-
ph19.cnf	-	-	-	-	-	-
ph20.cnf	-	-	-	-	-	-
CertBench						
clauses-2.cnf	2.33	4.91	3.77	2.50	3.71	22.31
clauses-4.cnf	121.31	47.04	1180.25	97.40	1180.28	-
clauses-6.cnf	-	378.69	-	-	-	-
IBM_FV_2004_rule_batch_30_SAT_dat.k15.cnf	425.42	447.38	-	488.61	-	-
eq.atree.braun.7.unsat.cnf	1.39	1.43	55.42	1.42	55.17	0.60
eq.atree.braun.8.unsat.cnf	10.80	9.88	-	10.43	-	3.60
eq.atree.braun.9.unsat.cnf	51.55	44.66	-	60.54	-	23.46
AProVE07-02.cnf	910.30	-	-	-	-	-
AProVE07-15.cnf	339.39	265.01	-	187.93	-	-
AProVE07-20.cnf	360.75	349.75	-	249.83	-	-
AProVE07-21.cnf	713.89	-	55.54	529.63	55.28	-
AProVE07-22.cnf	231.62	203.78	-	194.50	-	-
itox_vc965.cnf	0.12	0.12	0.12	0.11	0.11	0.12
dated-5-11-u.cnf	268.20	248.99	-	210.44	-	-
total-5-11-u.cnf	17.26	20.61	-	23.24	-	-
dated-5-15-u.cnf	-	-	-	-	-	-
total-5-13-u.cnf	74.44	70.03	-	65.87	-	-
dated-10-15-u.cnf	33.34	45.35	-	29.94	-	-
dspam_dump_vc973.cnf	-	-	0.42	0.63	0.40	-
manol-pipe-c10nidw_s.cnf	-	460.51	-	-	-	-
SAT 2007						
ran_1_SAT.cnf	22.40	278.53	-	997.20	-	-
ran_1_UNSAT.cnf	29.25	25.31	-	33.52	-	-
ran_2_SAT.cnf	35.67	11.46	-	89.36	-	-
ran_2_UNSAT.cnf	0.22	0.16	-	0.24	-	-
unif2p-p0.7-v3500-c9345-S1286605994-07-SAT.cnf	22.55	275.71	-	995.32	-	-
unif2p-p0.7-v3500-c9345-S1377128774-12-UNSAT.cnf	29.54	24.69	-	33.44	-	-
unif2p-p0.7-v3500-c9345-S1455331099-09-SAT.cnf	35.89	11.56	-	88.19	-	-
unif2p-p0.7-v3500-c9345-S977957999-02-UNSAT.cnf	0.23	0.17	-	0.25	-	-
SAT 2005						
clauses-2.renamed-as.sat05-1961.cnf	2.30	4.92	3.72	2.55	3.71	22.39
clauses-2.shuffled-as.sat05-1966.cnf	3.92	3.25	1155.10	5.88	979.22	156.37
clauses-4.renamed-as.sat05-1962.cnf	122.68	47.07	1197.88	97.56	1195.26	-
clauses-4.shuffled-as.sat05-1967.cnf	101.62	101.00	-	75.73	-	-
clauses-6.shuffled-as.sat05-1968.cnf	524.25	753.53	-	-	-	-
clauses-10.renamed-as.sat05-1960.cnf	198.40	173.00	174.40	201.53	137.15	-
clauses-10.shuffled-as.sat05-1965.cnf	431.11	364.41	-	360.03	-	-

B Data - Number of Conflicts

The second table is number of conflicts.

	VSIDS	REF	INIT	RAND	LIN	REVLIN
PigeonHole						
ph09.cnf	22775	30638	5908	49941	5908	5482
ph10.cnf	116858	136016	18462	311842	18462	20542
ph11.cnf	1933896	810957	91871	2519669	91871	103778
ph12.cnf	21394801	25095865	407593	34188195	407593	449521
ph13.cnf	28595101	28624333	1907150	28081837	1907150	2011898
ph14.cnf	22856144	23693223	6725283	22459881	6725390	5449922
ph15.cnf	19279990	18808698	5857521	18536993	5862295	4703047
ph16.cnf	14615414	15037343	5160896	15069483	5167299	4084742
ph17.cnf	12782483	12329759	4564404	12953303	4566718	3577353
ph18.cnf	9468736	9934925	4022582	10345056	4016676	3102412
ph19.cnf	8393297	8482332	3465960	8318805	3532084	2724341
ph20.cnf	6313685	6610605	3104890	7897765	3108684	2292214
CertBench						
clauses-2.cnf	4698	9263	13550	3556	13550	28674
clauses-4.cnf	89675	38424	394875	76540	394875	317724
clauses-6.cnf	383183	180830	330263	358553	364501	190124
IBM_FV_2004_rule_batch_30_SAT_dat.k15.cnf	281923	283294	559431	313845	559142	580773
eq.atree.braun.7.unsat.cnf	52089	51684	1225429	49254	1225429	12204
eq.atree.braun.8.unsat.cnf	281550	255387	15313292	259828	15412749	52320
eq.atree.braun.9.unsat.cnf	943138	788993	11671481	1060094	11755337	242432
AProVE07-02.cnf	1030003	1562761	2530448	1179753	2103307	2571391
AProVE07-15.cnf	352420	297183	462274	219454	463343	749928
AProVE07-20.cnf	584012	571387	2052306	423780	2043740	2678738
AProVE07-21.cnf	2112271	2607749	250148	1808401	250148	2987107
AProVE07-22.cnf	294396	237630	862679	234719	862044	1109617
itox_vc965.cnf	1	1	1	1	1	1
dated-5-11-u.cnf	136880	123115	232759	124556	232741	257379
total-5-11-u.cnf	14624	13616	137667	13553	137506	149098
dated-5-15-u.cnf	419626	503742	337449	455500	338343	208453
total-5-13-u.cnf	39119	39869	208286	38697	208245	172524
dated-10-15-u.cnf	17453	19037	69322	16554	69929	126726
dspam_dump_vc973.cnf	670979	798827	18	172	18	313647
manol-pipe-c10nidw_s.cnf	837452	486500	385037	667754	384433	213005
SAT 2007						
ran_1_SAT.cnf	177417	1410167	4934717	4107100	4959785	5025275
ran_1_UNSAT.cnf	227593	203419	5188425	257437	5165971	5676641
ran_2_SAT.cnf	266276	104062	5136680	571529	5151577	5047312
ran_2_UNSAT.cnf	3807	3065	6228416	4020	6219578	6632854
unif2p-p0.7-v3500-c9345-S1286605994-07-SAT.cnf	177417	1410167	4942952	4107100	4959625	5030584
unif2p-p0.7-v3500-c9345-S1377128774-12-UNSAT.cnf	227593	203419	5182662	257437	5178808	5699003
unif2p-p0.7-v3500-c9345-S1455331099-09-SAT.cnf	266276	104062	5142696	571529	5157623	5835662
unif2p-p0.7-v3500-c9345-S977957999-02-UNSAT.cnf	3807	3065	6213398	4020	6215950	6638914
SAT 2005						
clauses-2.renamed-as.sat05-1961.cnf	4698	9263	13550	3556	13550	28674
clauses-2.shuffled-as.sat05-1966.cnf	4984	3922	280430	6222	280430	68294
clauses-4.renamed-as.sat05-1962.cnf	89675	38424	394875	76540	394875	316748
clauses-4.shuffled-as.sat05-1967.cnf	47244	50271	182784	32726	183542	195427
clauses-6.shuffled-as.sat05-1968.cnf	144504	200891	88709	283524	88739	46320
clauses-10.renamed-as.sat05-1960.cnf	2206	2210	3359	2801	3359	103364
clauses-10.shuffled-as.sat05-1965.cnf	2613	3680	20158	2784	20170	20459

C Data - Number of Decisions

The third table is the number of decisions made.

	VSIDS	REF	INIT	RAND	LIN	REVLIN
PigeonHole						
ph09.cnf	27889	37647	27558	69957	27558	25169
ph10.cnf	141471	168095	2134096	1111345	2134096	2373989
ph11.cnf	2293625	995516	10485232	8968731	10485232	11825087
ph12.cnf	25380826	30206284	46499129	125527940	46499129	51143185
ph13.cnf	34349744	34863717	218419934	105042659	218419934	230447419
ph14.cnf	27865561	29452514	784348096	82794320	784361698	633692352
ph15.cnf	24922177	23635616	682659432	69545725	683229891	546242241
ph16.cnf	18534451	19448576	598111170	56145092	598798435	472994352
ph17.cnf	17159288	16479305	521900046	48078805	522164236	410110263
ph18.cnf	11884589	13218011	458771943	32351759	458064912	351165035
ph19.cnf	11117486	11180569	395843214	24777352	403591803	307586730
ph20.cnf	8167893	8896430	352053397	25313743	352505491	258319718
CertBench						
clauses-2.cnf	14623	23503	37425	16892	37425	37525
clauses-4.cnf	210540	120462	3394854	197184	3394854	334030
clauses-6.cnf	726329	493951	4262809	781854	5019975	280038
IBM_FV_2004_rule_batch_30_SAT_dat.k15.cnf	438891	433017	2157972	599995	2157610	998920
eq.atree.braun.7.unsat.cnf	62208	59491	1362417	63385	1362417	12372
eq.atree.braun.8.unsat.cnf	323933	292421	17678711	324102	17783496	52759
eq.atree.braun.9.unsat.cnf	1108791	914199	13764373	1304758	13862221	243798
AProVE07-02.cnf	1227743	1911117	10562245	1599987	9096619	14836152
AProVE07-15.cnf	535216	479417	4012051	477971	4022148	1261926
AProVE07-20.cnf	676840	664790	2748217	503937	2736448	3143368
AProVE07-21.cnf	2479710	2918458	252846	2296425	252846	3265931
AProVE07-22.cnf	374566	296876	865546	333317	864914	1241621
itox_vc965.cnf	0	0	0	0	0	0
dated-5-11-u.cnf	218704	206322	2271355	213617	2271259	780304
total-5-11-u.cnf	29406	30505	4441764	49178	4440455	629082
dated-5-15-u.cnf	517808	612804	2744899	568180	2751948	569930
total-5-13-u.cnf	75075	66904	7148624	84705	7147654	678101
dated-10-15-u.cnf	42008	48317	904955	69088	906610	945530
dspam_dump_vc973.cnf	1205228	1486931	7002	34687	7002	495591
manol-pipe-c10nidw_s.cnf	3880176	2679761	834774	11211961	833655	862948
SAT 2007						
ran_1_SAT.cnf	228504	1782081	13831228	5871023	13903408	23204672
ran_1_UNSAT.cnf	287450	258574	12700286	379145	12631001	16801301
ran_2_SAT.cnf	327437	132451	13130890	822081	13167861	11197455
ran_2_UNSAT.cnf	4948	4072	14488918	6730	14470650	16678865
unif2p-p0.7-v3500-c9345-S1286605994-07-SAT.cnf	228504	1782081	13855656	5871023	13903004	23226137
unif2p-p0.7-v3500-c9345-S1377128774-12-UNSAT.cnf	287450	258574	12682924	379145	12669803	16862788
unif2p-p0.7-v3500-c9345-S1455331099-09-SAT.cnf	327437	132451	13146316	822081	13184116	12856377
unif2p-p0.7-v3500-c9345-S977957999-02-UNSAT.cnf	4948	4072	14456686	6730	14462672	16692447
SAT 2005						
clauses-2.renamed-as.sat05-1961.cnf	14623	23503	37425	16892	37425	37525
clauses-2.shuffled-as.sat05-1966.cnf	11747	9924	991098	19264	991098	360699
clauses-4.renamed-as.sat05-1962.cnf	210540	120462	3394854	197184	3394854	332980
clauses-4.shuffled-as.sat05-1967.cnf	95117	98856	1164305	90400	1168339	1235907
clauses-6.shuffled-as.sat05-1968.cnf	277676	362410	997718	647572	998234	775814
clauses-10.renamed-as.sat05-1960.cnf	13701	11661	16092	36941	16092	123618
clauses-10.shuffled-as.sat05-1965.cnf	11969	15515	665826	14558	665939	536126