Decision procedures and verification - CDCL heuristic

Jaroslav Šafář

September 22, 2020

1 Experimental evaluation

The CDCL solver uses restarts. Restarts occur when the number of conflicts reaches some limit. The initial limit was set to 100. This limit increases geometrically by a factor of 1.1 every time the restart occurs.

During restarts, the CDCL solver also deletes some learned clauses with LBD greater than a specified limit. The initial limit was set to 3. This limit also increases geometrically by a factor of 1.1 every time the restart occurs.

In this case, we wanted to test and compare the performance of two nontrivial heuristics for the branching literal and compare them with the random choice of the branching literal.

- The first heuristic (cdcl0) finds the unassigned literal which occurs in the largest number of not satisfied clauses.
- The second heuristic (cdcl1) finds the unassigned literal based on VSIDS heuristic, i.e. the literal which is present the most in the learned clauses.
- The third method (cdcl2) finds the unassigned literal at random.

Some smaller Uniform Random-3-SAT examples from SATLIB Benchmark Problems were used for experimental evaluation of CDCL algorithm.

1.1 SAT examples

In the case of SAT problems of sizes 20, 50, 75, 100, 125, the first few examples (usually between 5 and 10) were selected for the experimental evaluation. Their resulting CPU times, number of decided variables, propagated variables and restarts were averaged, and the results can be found in the following figures [1], [2], [3] and [4].

size	cdcl0	cdcl1	cdcl2
20	0.005416	0.006089	0.007279
50	0.078524	0.042270	0.058268
75	0.370158	0.225985	0.500884
100	2.543120	1.184686	11.657246
125	5.813448	3.215847	61.646794

Figure 1: Average time results of SAT examples

size	cdcl0	cdcl1	cdcl2
20	8.0	12.1	15.7
50	44.5	44.8	63.5
75	138.9	180.8	436.9
100	594.2	741.4	3345.3
125	1097.6	1397.5	8156.0

Figure 2: Average number of decisions of SAT examples

size	cdcl0	cdcl1	cdcl2
20	21.0	76.40	90.7
50	349.10	488.50	655.9
75	1659.84	2839.8	5765.5
100	9191.6	13829.2	53020.6
125	18561.00	29937.1	141874.4

Figure 3: Average number of unit propagations of SAT examples

size	cdcl0	cdcl1	cdcl2
20	0.0	0.0	0.0
50	0.0	0.0	0.1
75	0.5	0.8	2.0
100	2.5	3.6	10.1
125	4.3	6.3	15.9

Figure 4: Average number of restarts of SAT examples

1.2 UNSAT examples

In the case of UNSAT problems of sizes 50, 75, 100, the first few examples (usually between 5 and 10) were again selected for the experimental evaluation. Their resulting CPU times, number of decided variables, propagated variables and restarts were averaged, and the results

can be found in the following figures [5], [6], [7] and [8].

size	cdcl0	cdcl1	cdcl2
50	0.163991	0.096180	0.231239
75	1.236552	0.614967	4.368183
100	6.696350	3.335396	361.101432

Figure 5: Average time results of UNSAT examples

size	cdcl0	cdcl1	cdcl2
50	58.8	81.6	222.8
75	320.2	424.2	1993.0
100	1293.5	1614.0	18817.5

Figure 6: Average number of decisions of UNSAT examples

size	cdcl0	cdcl1	cdcl2
50	692.8	1090.0	2512.8
75	4352.8	6630.6	27228.2
100	21351.0	30643.5	296726.0

Figure 7: Average number of unit propagations of UNSAT examples

size	cdcl0	cdcl1	cdcl2
50	0.0	0.1	1.1
75	1.6	2.6	8.6
100	6.0	7.5	27.5

Figure 8: Average number of restarts of UNSAT examples

1.3 Conclusion

As we can see from the results, the random choice of the decision literal is significantly slower and leads to more computation, as expected.

The interesting part is that the VSIDS heuristic (cdcl1) is slightly faster than cdcl0 but uses more decisions, unit propagations, and restarts than cdcl0. This effect is caused mainly by the fact that cdcl0 needs to go through the watched lists of all unassigned variables to determine which variable occurs in the largest number of clauses.

This computation is only an approximation. If we wanted to know the actual number of occurrences of the given variable, we would need to store adjacency lists of clauses for each variable and iterate through them, which would be even more computationally expensive.