CS264A Project Report

Hao Chen (904547539)

June 6, 2015

1 Structure

In this part, we introduce the structure of Var, Literal, Clause, and SatState.

1.1 Var

```
typedef struct var {
  c2dSize index;  // index, starts from 1

  c2dSize num_cnf_clauses;  // number of cnf_clauses mentioning the var

  c2dSize num_clauses; // number of clauses mentioning the var
  c2dSize dyn_cap;     // capacity of the clauses list
  Clause** clauses;    // clauses mentioning the variable, index starts from 0.

  Lit* p_literal;      // positive literal corresponding to the var
  Lit* n_literal;      // negative literal corresponding to the var

  BOOLEAN mark; //THIS FIELD MUST STAY AS IS
} Var;
```

Figure 1: structure of var

Clause** clauses is a dynamic array in which dyn_cap is the capacity and num_clauses is the current size. Once the capacity is not enough, we double the capacity. We use the same method to handle dynamic array in other places.

1.2 Literal

```
struct literal {
  c2dLiteral index;   // pos from 1 to n; neg from -n to -1

  c2dLiteral decision_level;
  Clause* decision_clause;

  Lit* op_lit;
  Var* var;

  c2dSize num_clauses; // number of clauses mentioning the literal
  c2dSize dyn_cap;
  Clause** clauses; // clauses mentioning the literal, index starts from 0. It's dynamic.
};
```

Figure 2: structure of Literal

If decision_level > 0, then decision_clause denotes how a literal is set, NULL means that the literal is set by a decision, otherwise means that the literal set by an implication through clause decision_clause.

1.3 Clause

```
struct clause {
  c2dSize index;

  Lit** literals; // index starts from 0
  c2dSize size;   // size of literals

  c2dSize decision_level; // the maximum decision level over all its literals
  c2dSize num_false;      // the number of literals cannot be satisfied

  c2dLiteral assertion_level;

  BOOLEAN mark; //THIS FIELD MUST STAY AS IS
};
```

Figure 3: structure of Clause

We introduce the decision_level for clause. It's the maximum decision level over all its literals which are set. It shows which decision level the clause is subsumed.

1.4 SatState

```
typedef struct sat_state_t {
  c2dSize num_vars;

  Var** variables;  // variables variabels, start from 1
  Lit** p_literals; // positive literals, start from 1
  Lit** n_literals; // negtive literals, start from 1

  c2dSize num_cnf_clauses;
  Clause** cnf_clauses;     // starts from 1

  c2dSize num_learned_clauses;
  c2dSize dyn_cap;
  Clause** learned_clauses; // starts from 0. it's dynamic

  c2dSize cur_level;

  Lit** decided_literals;
  c2dSize num_decided_literals;
  Lit** implied_literals;
  c2dSize num_implied_literals;

  Clause* asserted_clause;

  c2dSize unit_resolution_s;  // Type of unit_resolution
} SatState;
```

Figure 4: structure of SatState

2 Unit Resolution and Clause Learning

We keep detecting whether there has new unit clause. However, new unit clause appears only if new literal implied or decided. Therefore, we can maintain a list of literals which are newly implied or decided. Every time we implied or decided a new literal, we go through all the clause containing the literal. If a clause becomes unit clauses and the unit literal hasn't been set, we push the literal into the list.

Once the list is empty, it finishes the unit resolution.

How to initial the list? As we can see, there are three possible places that we perform perform unit resolution. (1) after deciding on a new literal. (2) after adding an asserting clause (3) initially perform (happen only once and before the other two types)

For the first case, the initial list only contain the new literal. For the second case, we just check whether the asserting clause is unit clause currently. For the third case, we need to go through all the clause and push the literal which is in the unit clause into list.

Once a contradiction happened at a clause (node). We form the conflict set as following:

$$C(n) = \begin{cases} n & \text{if n is root;} \\ ePa(n) \cup \cup_{m \in Pa(n)} C(m) & \text{otherwise} \end{cases} \tag{1}$$

where $Pa(n)$ are the parents of node n set at earlier levels, $ePa(n)$ are the parents of node n set at earlier levels.

In literal structure, we store how a literal is set in decision_clause. Then we can easily obtain the conflict set as above by BFS.

Also, the conflict clauses which are obtained by above methods is asserting.

3 Experiment

filename	compile time	model counting	node	edge
2bitcomp_5.cnf	0.559	9840070722846720	117843	235338
2bitmax_6.cnf	202.265	$20682964643570488029 * 10^{10}$	6491021	12981274
4blocksb.cnf	48.768	4	1878	2824
C163_FW.cnf	134.783	$29676906523136185728 * 10^{121}$	12725621	25447238
C169_FW.cnf	0.001	3216989843619840	3041	3246
C171_FR.cnf	1.759	$14051294365287595402 * 10^{97}$	387832	771974
C210_FVF.cnf	61.182	$30161428963068191203 * 10^{151}$	2844058	5684320
C211_FS.cnf	0.227	$13762971811969386785 * 10^{48}$	74510	145638
C215_FC.cnf	71.281	$18559191073670376139 * 10^{131}$	9023441	18043032
C230_FR.cnf	38.931	$32608326043677328952 * 10^{116}$	4659647	9315378
C250_FW.cnf	0.002	$12047082580849261441 * 10^{18}$	3609	4220
C638_FKA.cnf	175.288	$22079134950953694386 * 10^{109}$	4677174	9349900
C638_FVK.cnf	0.097	$88271038388920812991 * 10^{102}$	17694	31106
ais10.cnf	X	X	X	X
bw_large.a.cnf	0.233	1	917	916
bw_large.b.cnf	X	X	X	X
cnt06.shuffled.cnf	4.320	1	1523	1522
huge.cnf	0.119	1	917	916

filename	compile time	model counting	nodes	edges
log-1.cnf	0.028	$56415355251141796875 * 10^{1}$	10914	19382
log-2.cnf	X	X	X	X
log-3.cnf	X	X	X	X
medium.cnf	0.000	2	323	376
par16-1-c.cnf	8.872	1	633	632
par16-2-c.cnf	15.676	1	697	696
par16-2.cnf	43.490	1	2029	2028
par16-3.cnf	6.817	1	2029	2028
par16-5-c.cnf	34.174	1	681	680
par16-5.cnf	12.252	1	2029	2028
prob004-log-a.cnf	218.941	26134280339375952	12186912	24368028
qg1-07.cnf	X	X	X	X
qg2-07.cnf	X	X	X	X
qg3-08.cnf	0.866	18	8204	14894
qg6-09.cnf	0.125	4	3806	5706
qg7-09.cnf	0.030	4	3814	5722
ra.cnf	7.141	$18739277038847939886 * 10^{267}$	491768	978592
ssa7552-038.cnf	0.385	$28432833270798238107 * 10^{21}$	42203	78870
tire-2.cnf	0.080	738969640920	20631	39808
tire-3.cnf	0.067	222560409176	32022	62488
tire-4.cnf	0.523	103191650628000	103000	203840
uf250-017.cnf	X	X	X	X
uf250-026.cnf	X	X	X	X

'X' marks time limited exceed.