A Novel Approach to Combine a SLS- and a DPLL-Solver for the Satisfiability Problem

Reporter: Chi Zhiming

November 8, 2022

目录

目录

Contribution

- They have presented a novel and simple approach to create an incomplete hybrid SAT solver *hybridGM*, utilizing *gNovelty+* as the SLS component and *March_ks* as the DPLL component
- They first define the term of a search space partition (SSP) and explain its construction and use in order to develop the idea behind our approach.

Background

SLS(Stochastic Local Search):

- SLS solvers scale very well on random instances and use comparatively little memory.
- On the other hand they can not disclose the unsatisfiability of a problem.

DPLL:

- They are good at solving industrial and structured problems and they can ascertain if a problem is satisfiable or unsatisfiable.
- But they have difficulties solving random instances and use a larger amount of memory than SLS solvers.

Background

Hybrid SAT solver: Combining both approaches seems promising.

- use a SLS solver to support a DPLL solver.
- use information gathered by DPLL solvers on a certain formula to support the search of a SLS solver.
- SLS and DPLL solvers are supposed to benefit equally from each other.

Background

- *gNovelty+*:In its core, gNovelty+ utilizes a gradient-based variable score update scheme to calculate candidate variables for the next flip;winner of the random category of the SAT 2007 Competition
- *March_ks*:a double look-ahead DPLL solver;the winner of random **UNSAT** category of the SAT 2007 Competition

目录

Preliminary Study

Observation:

- The runtime of a SLS solver on formulae of the same size can vary greatly.

Assumption:

- The search space structure of the hard to solve formulae contained many attractive local minima that were visited by the SLS solver very often.

Verify:

- Tried to cluster all points from *gNovelty+*'s search trajectory → too large
- used a bloom filter to save all local minima, then checked how many assignments, that *gNovelty+* visited, fell in the neighborhood of the saved local minima →

Preliminary Study

Analysis:

- Whether the local minima is solution or not.
- Method:search the complete neighborhood of a local minimum within a certain Hamming distance
- But it's too large to be computed in foreseeable time
- The Hamming distance between a good local minimum and the nearest solution is correlated with the quality of that local minimum.

Some definition

Example 1. For

$$F = (x_1 \vee \overline{x_2} \vee x_5) \wedge (\overline{x_1} \vee x_3 \vee \overline{x_5}) \wedge (\overline{x_3} \vee x_4 \vee x_5) \wedge (x_2 \vee x_3 \vee \overline{x_4}) \wedge (\overline{x_1} \vee \overline{x_2} \vee \overline{x_5})$$

an example for α and β could be $\alpha = (1,0,1,1,0)$ and $\beta = (?,1,1,?,?)$. The application of α and β on F is: $F(\alpha) = 1$ and $F' = F(\beta) = (x_1 \vee x_5) \wedge (x_4 \vee x_5) \wedge (\overline{x_1} \vee \overline{x_5})$.

Some definition

Definition 1 (flip trajectory of a SLS solver). *Given a SLS solver S with input formula F and a complete starting assignment α_s we define the flip trajectory of $S(F, \alpha_s)$ as $T_S(F, \alpha_s) = (t_1, \ldots, t_w)$ where $t_i \in \{x_1, \ldots, x_n\}$ denote the variables being flipped by the SLS-algorithm S, and w is the total number of flips made, starting with the formula F and the initial assignment α_s.*

Example 2. Given the formula from example 1 and a starting assignment $\alpha_s = (0, 1, 0, 0, 0)$, a possible flip trajectory that would lead to a satisfying assignment could be $T_S(F, \alpha_s) = (x_1, x_5, x_3, x_2)$

Some definition

Definition 2 (search space partition). *We define a search space partition (SSP) by construction: Given a complete assignment* α_j*, which was visited by* S *in the* j*'th flip of the trajectory, we construct the SSP by starting with* $k = 0$ *and* $\beta = \alpha_j$*. Then we repeat setting* $\beta[t_{j+k}] = ?$ *and* $\beta[t_{j-k}] = ?$*, where* $t_{j\pm k} \in T_S(F, \alpha_s)$*, and increasing* k *by 1 until* $|\beta|_? \geq c \cdot n$ *where* c *is some constant* $c \in (0, 1)$ *(to be determined later).*

Some definition

Example 3. Let $\alpha_7 = (0,0,1,1,0,1,0,1,1,1)$ be a complete assignment for a formula F with 10 variables and let the surrounding flip trajectory be

$$T_S(F, \alpha_s) = (x_2, x_6, x_1, x_9, x_1, x_6, \underline{\mathbf{x_1}}, x_3, x_9, x_1, x_1, x_8, x_3, \ldots)$$

If we set $c = 0.5$ and start to construct a SSP from position $j = 7$ in $T_S(F, \alpha_s)$, then the first variable that is unassigned in β is x_1 $(k = 0)$. In the next step x_3 and x_6 get unassigned $(k = 1)$ according to $T_S(F, \alpha_s)$. This procedure is repeated until $|\beta|_? \geq 5$. After five steps the process will stop with $\beta = (?, 0, ?, 1, 0, ?, 0, ?, ?, 1)$.

Use of Search Space Partitions

- SSP can contain multiple minima
- Monitoring the flips made by the SLS solver around the discovered local minimum in the trajectory
- Then,unassigning these identified variables in the complete assignment of the local minimum, and calling DPLL
- If DPLL fail, continue to identify a new local minima

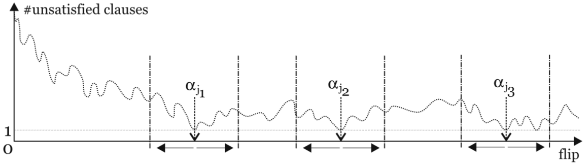

Fig. 1. A schematic visual of the objective function and the flips used for the construction of a SSP

Use of Search Space Partitions

All in all, a generic algorithm implementing the above idea would use a SLS solver to localize good local minima, build a SSP, apply the partial assignment of the SSP on the formula and try to find a solution for the simplified formula with a DPLL solver. This process would be repeated until a solution is found or until another stopping criterion is met. The algorithm can not prove the unsatisfiability of the problem but it could speed up the SLS solver by finding a solution sooner.

目录

Pseudocode for hybridGM

INPUT: formula F, cutoff. OUTPUT: model for F or UNKNOWN.

```
hybridGM(F, cutoff){
    α = α_s = randomly generated starting assignment;
    numFlips = 0; c = 0.5; barrier= 1; collectSSP = FALSE;
    while(numFlips < cutoff){
        var = pickVar();
        append(T_gNovelty+(F, α_s),var);
        α[var] = 1−α[var];
        numFlips++;
        if (α is model for F) return α;
        if (numUnsatClauses ≤ barrier){
            β = α;
            collectSSP = TRUE;
            j = numFlips; k = 0;
        }
        if (collectSSP == TRUE){
            β[ variableIndex( T_gNovelty+(F, α_s)[j + k] ) ] =?;
            β[ variableIndex( T_gNovelty+(F, α_s)[j − k] ) ] =?;
            k++;
        }
        if (|β|_? ≥ cn){
            μ = March_ks(F, β);
            if (μ is model for F) return μ;
            else if (unaryConflictOccurred() == TRUE) c = c + 0.05;
            collectSSP = FALSE;
        }
        updateParameters(); //noise, scores
    }
    return UNKNOWN;
}
```

目录

Conclusions

- We defined this new term SSP, explained how such SSPs are constructed and how they are used.
- We implemented our novel approach in the hybrid SAT solver *hybridGM*, utilizing *gNovelty+* as the SLS component and *March_ks* as the DPLL component.

Future Work

- On uniform random 5- and 7-SAT instances, March ks almost never finds a solution.
- Dynamically adapt the barrier while hybridGM performs a search.

Thank you