Extensions of CDCL Branching Heuristics by Exploration during Conflict Depression

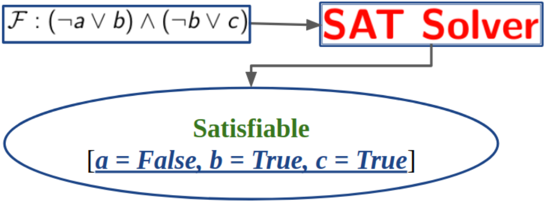
Md Solimul Chowdhury Martin Müller Jia-Huai You

Department of Computing Science, University of Alberta.

February 21, 2022

Introduction

- Boolean Satisfiability (SAT)
  - Given a SAT formula, determine assignments of the variables to satisfy that formula, if one exists. Otherwise, unsatisfiable ....
    - A toy example:



- Easy? Not so easy ...

Introduction

- SAT solving is **NP-Complete**
  - *SAT is hard!*
  - Intractable, in general.
- However, Modern SAT solvers are very efficient.
  - Conflict Directed Clause Learning (CDCL) has many real world applications.
    - Applications in many important domains:
      - Hardware design verification
      - Software testing
      - Encryption
      - Planning
      - Theorem proving
    - CDCL is highly scalable.
      - able to solve large formulas consisting of millions of variables and clauses.
      - **Hypothesis:** CDCL Exploits problem structures

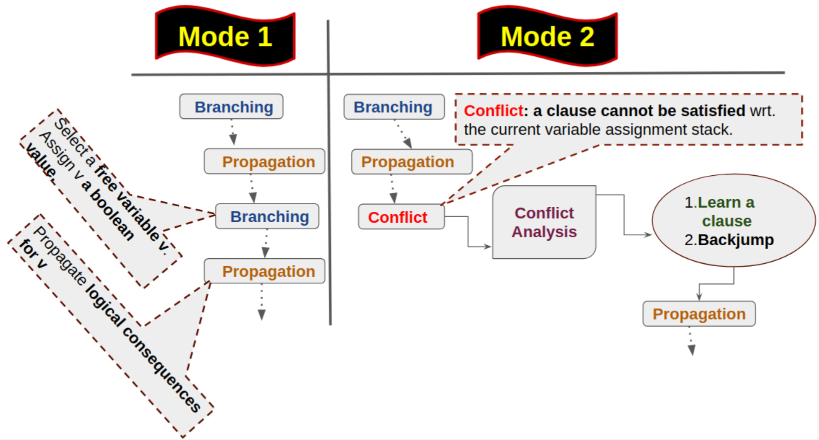
Contributions

Major contributions:

- **Contribution 1:** An empirical investigation of the CDCL SAT solving process to obtain insights on its conflict generation pattern.
  - Identifies a pathological phase in CDCL search.
- **Contribution 2:** A CDCL algorithmic extension, based on the obtained insights.
  - Employs random exploration in a novel way.
- **Contribution 3:** An extensive evaluation.
- **Contribution 4:** Analysis of the results to reveal insights.

Background: How does a CDCL SAT solver work?

- Performs a backjumping tree-search to determine satisfiabilty.



- **Restarts frequently:** abandons the current partial assignment and starts the search from the scratch.

Background: Importance of Fast Conflict Generation

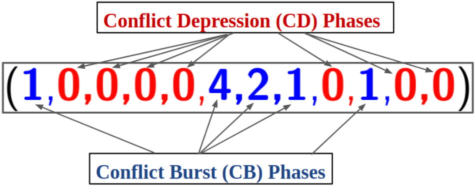
- Conflict Generation at a **fast rate is crucial** for CDCL SAT solvers.
- **Conflict** $\longrightarrow$ **Clause learning** $\longrightarrow$ **Pruning** $\longrightarrow$ **Faster Solving.**
- CDCL branching heuristics are **conflict-greedy**.
  - Examples:
    Variable State Independent Decaying Sum (VSIDS) (Moskewicz 2001 et. al.)
    Learning Rate Based (LRB) (Liang 2016 et. al.)
    - based on **look-back principle**
    - selection priority is based on **recent conflict involvements**.
    - **intuition**: such selection will generate more conflicts.
- CDCL branching heuristics **generate conflicts at a fast rate**.
  - On average, 1 conflict in 2 decisions. (Liang 2017 et. al.)

Background: Monte-Carlo Random-Walks (MRW)

- Designed for random sampling of non-terminal states.
- First proposed by (Nakhost & Müller, 2009) in the context of deterministic planning.
- At a given state s of a plan search, MRW performs
  - a fixed number of random walks in the local neighborhood of s.
  - a walk consists of a fixed number of steps.
  - Goal: to find a state s^* which is the best among the explored states near s.
- Exploration in our work is fundamentally inspired by MRW.

Notions and Definitions

- We formulate **two novel notions:**
  - **Conflict Depression**: Sequence of one or more consecutive decisions with no conflict.
  - **Conflict Burst**: Sequence of one or more consecutive decisions with at least one conflict.



- Some Measures:
  - **Decision Rate**: number of decisions per restart.
  - **CD phase Rate**: number of CD phases per restarts.
  - **Average CD phase Length**

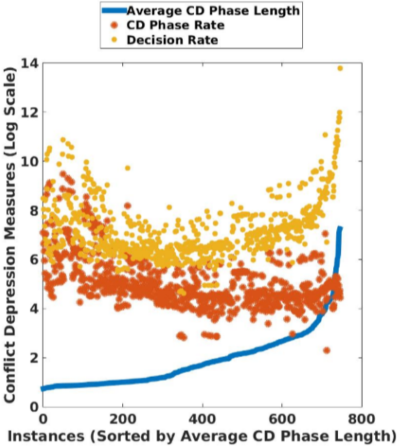
Contribution 1: A series of empirical insights

- We studied CD (and briefly, CB) phases with VSIDS and LRB.
  - CDCL solvers:
    (i) Glucose (uses VSIDS exclusively).
    (ii) MAPLECOMSPS_PURE_LRB (uses LRB exclusively)
  - 750 maintrack instances from SAT-2017, 2018.
  - Time-out per run: 5000 seconds.
- We have a series of interesting observations ...

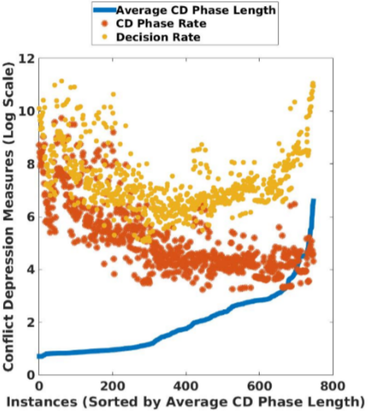
CD Phase Rate with VSIDS and LRB

- Observations:

  #1: CD phases occur frequently with VSIDS (left) and LRB (right).



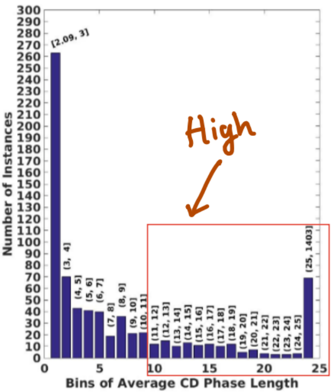
CD phase Rate with VSIDS



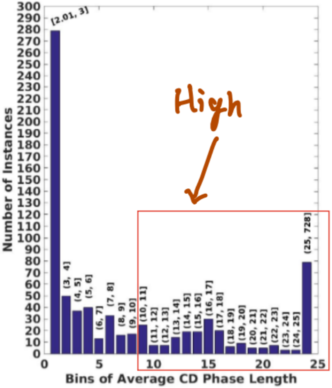
CD phase Rate with LRB

Average CD Phase Length with VSIDS and LRB

- Distributed 750 instances into 25 bins
    - By average CD phase lengths of these instances



CD phase length with VSIDS



CD phase length with LRB

#2: For many instances, avg. CD phase length are high with VSIDS (left) and LRB (right).

Propagation Depression during a CD phase

- During a CD phase, VSIDS/LRB decisions are ineffective to create conflicts and **only create truth value propagation**.
  - But, how much propagation ?

#3:

  - On average, for both VSIDS and LRB Propagations in a CD phase is **10 times lower** than Propagations in a CB phase.
    $\longrightarrow$ During a CD phase, VSIDS/LRB branching decisions **go through a propagation depression** as well !

Bursts of Conflict Generation

- **#4:**
  For both of the heuristics, on average,
  - Only 25% of the decisions produce at least one conflict.
    - Some of which produces **more than one conflicts.** How many of them?
  - Of all decisions which produce any conflict,
    61% produce more than one.
- **Conflict Burst** phases are **short**, but **conflict intense**.
  - Many conflicts within a short span of consecutive decisions.

Summary of Empirical Observations

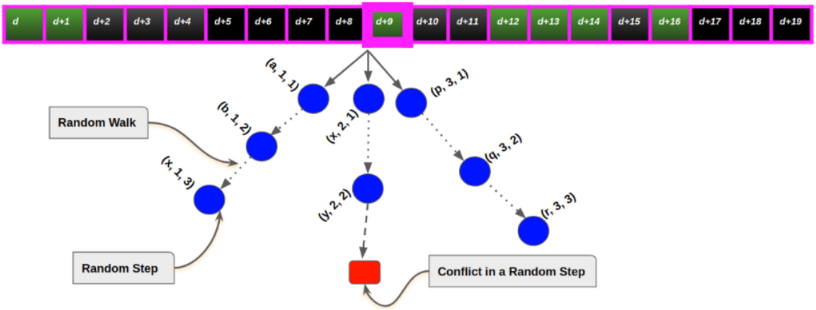
- The typical search behavior contains
  - shorter but conflict intense **Conflict Burst** phases
  - followed by longer **Conflict Depression** phases
    - where the search does not find any conflicts.

Contribution 2: Random Exploration amid CD phases

- Amid a CD phase, the variables that are deemed *best* by the CDCL heuristics are ineffective ...
- Can we do better amid a CD phase? variable re-ranking?
- Formulated an exploration based CDCL framework named *expSAT*.

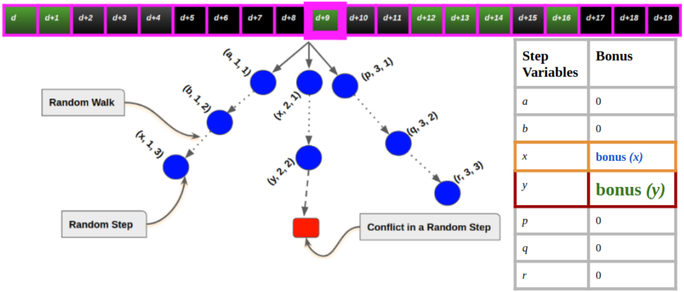
expSAT: an exploration guided CDCL SAT solver

- **Main Idea:** Amid a substantial CD phase, **with a non-zero probability**,
  - perform **exploration episodes** to identify conflict friendly variables.
  - Exploration Episode: a fixed number of **random walks**, with a fixed number of **steps per walk**.
- Goal: to find a conflict amid a CD phase.



expSAT: an exploration guided CDCL SAT solver

- Rewards variables in a conflict reaching walk by assigning **bonus**.
  - Only x and y in the 2nd walk receives **bonuses**.



- **bonus** are computed based on the **conflict quality** (i.e., LBD).
  - Largest bonus is given to the walk closest to conflict.
    - **bonus**(y) $>$ **bonus** (x)
  - Bonus decays exponentially with distance from the conflict.
  - Temporal Credit Assignment as studied in RL.
- *exploration score* $=$ **bonus**.

Branching in expSAT

- **expVSIDS**: VSIDS score + (scaled) exploration score for VSIDS.
- **expLRB**: LRB score + (scaled) exploration score for LRB.
  - Why scale?

Exploration scores are in range (0, 1]

VSIDS/LRB scores are unbounded decimal numbers.

Need to scale up the exploration score when combining.

currently active *base heuristic* $B \in \{\texttt{LRB},\texttt{VSIDS}\}$

$\exp(v) \leftarrow \text{compute\_exp\_score}(v)$;
$f \leftarrow \text{scale\_factor}(B)$;
$\texttt{expB}(v) \leftarrow \text{activity}(v) + f * \exp(v)$;

- The value of scale_factor f is different for VSDIS and LRB.
- **While branching**, expSAT selects the variable that maximizes the combined score.

$$v^* \leftarrow argmax_{v \in uVars(\mathcal{F})}\ \texttt{expB}(v);$$

Contribution 3: Empirical Evaluation

- Implemented expSAT on top of 5 CDCL solvers:
  - **glucose 4.2.1** (gLCM)
  - **MAPLECOMSPS_PURE_LRB** (MplLRB)
  - **MapleCOMSPS** (MplCOMSPS)
  - **Maple_CM** (MplCM)
  - **MapleLCMDist_ChronoBT** (MplCBT)
    - 11 expSAT solvers
      - (i) 4 employ expVSIDS exclusively
      - (ii) 4 employ expLRB exclusively
      - (iii) 3 employ a combination of expVSIDS and expLRB
- Exploration Parameter Settings:
  - exploration is expensive.
  - need to have a balance between overhead and gains.
    - 5 walks/exploration episode.
    - 5 steps/walk.
    - exploration trigger probability=0.02 .
  - same for all experiments.

Test Sets

- Two test sets:
  - **Competition Benchmarks**: 750 maintrack instances from SAT-2017 and 2018.
    - Time Out: 5,000 seconds
  - **SATCoin Benchmarks**: 52 hard SATCoin benchmark instances generated by an instance generator submitted for SAT Competition 2018.
    - Time Out: 36,000 secs

Experimental Results for 4 expVSIDS extensions

- Competition Benches Results: **Good-to-Strong** gains.

Solver Name	**Solved by Baseline**	**Solved by expSAT Extension**	**PAR2 Score Decrement**
gLCM	372	**379 (+7)**	1.59%
MplCOMSPS	412	**428 (+16)**	7.52%
MplCM	442	**443 (+1)**	0.3%
MplCBT	442	**451 (+9)**	2.88%

- Results with SATCoin Benches: **Very Strong** gains

Solver Name	**Solved by Baseline**	**Solved by expSAT Extension**
gLCM	7	**12 (+5)**
MplCOMSPS	4	**13 (+9)**
MplCM	1	**10 (+9)**
MplCBT	41	**43 (+2)**

Experimental Results for 4 expLRB extensions

- Competition Benches Results: **Good-to-Strong** gains.

Solver Name	**Solved by Baseline**	**Solved by expSAT Extension**	**PAR2 Score Decrement**
MplLRB	378	**393 (+15)**	3.84%
MplCOMSPS	412	**423 (+11)**	2.37%
MplCM	442	**438 (-4)**	-0.7%
MplCBT	442	**449 (+7)**	2.46%

- Results with SATCoin Benches: **Very Strong-to-Fair** gains

Solver Name	**Solved by Baseline**	**Solved by expSAT Extension**
MplLRB	0	**46 (+46)**
MplCOMSPS	4	4 (+0)
MplCM	1	**3 (+2)**
MplCBT	41	**43 (+2)**

Experimental Results for 3 expVSIDS+expLRB extensions

- Competition Benches Results: **Good-to-Strong** gains.

Solver Name	**Solved by Baseline**	**Solved by expSAT Extension**	**PAR2 Score Decrement**
MplCOMSPS	412	**425 (+13)**	2.77%
MplCM	442	**436 (-6)**	-1.03%
MplCBT	442	**451 (+9)**	2.79%

- Results with SATCoin Benches: **Very Strong-to-Good** gains

Solver Name	**Solved by Baseline**	**Solved by expSAT Extension**
MplCOMSPS	4	**13 (+9)**
MplCM	1	**10 (+9)**
MplCBT	41	**44 (+3)**

Analysis of the Solving Efficiency

- Analyzed the experimental data for
  (i) **gLCM and expGLCM** and
  (ii) **MplLRB and expMplLRB**
- Observation for **conflict efficiency**: In general,
  - Better solver for a subset of the tested problems **is more conflict efficient**.
    - Produces conflict at a fast rate, from which high quality clauses are learned.
- Observation for **average CD phase Length** : In general,
  - Better solver for a subset of these problems **reduces average CD phase length**.
  - exploration helps a solver **to escape from CD phases**.

Pathological Perspective of CD

- Heuristics search often exhibit pathological phases ...
  - Plateaus in local search SAT
    - many consecutive search states without decrease in heuristic value (Frank, Cheeseman, Stutz, 1997)
  - Search in deterministic planning
    - heuristic values of states do not improve within a plateau region (Hoffmann, 2005)
- CDCL branching heuristic values for each variable amid a CD phase are different.
  - However, none of these selected variables generates any conflicts.
- Lack of progress of CDCL SAT search amid CD
  $\approx$ lack of progress of local search algorithms in plateau.

We characterize CD as a pathological phase for CDCL.

Conclusions

- **Conclusions**:
  - Defined concept of conflict depression (CD).
  - Showed that leading CDCL branching heuristics
    - frequently undergo the **pathological phase of CD**, in which branching decisions **are ineffective**.
  - To combat CD phases, **we proposed expSAT**
    - performs random exploration in SAT search space.
    - designed to overcome CD phase.
  - **Empirically showed the effectiveness** of the expSAT approach.
    - For some extensions, strong gains for competition benchmarks.
    - Impressive gains over **hard bitcoin mining** benchmarks.

Future Work

- **Future Work**:
  - Identify underlying reasons that causes CD phases ...
    - **Hypothesis:** Onset of CD phases corresponds to switch between communities (sub-problems)?
  - **Identify characteristics of SAT domains** which influence the effectiveness of exploration.
  - Can we develop Machine Learning models to
    - predict onset of a long CD phase?
    - predict better variable selection amid a CD phase?

Acknowledgements

- This work has been funded by
  - NSERC Post Graduate Scholarship - Doctoral
  - Alberta Innovates Graduate Students Scholarship
  - NSERC Discovery Grant (from my PhD supervisors)
    - Martin Müller and Jia You