Review of SAT, NP Completeness

- **kSAT**
  - Literals: variables or their negations
  - Clause: disjunction of literals
  - CNF formula (Conjunctive Normal Form): conjunction of clauses
  - kCNF: CNF formula w/ at most k literals per clause
  - kSAT: The set of satisfiable kCNF formulas

- Recall: SAT (= set of satisfiable CNF formulas) is NP-complete
  - NP: languages whose membership can be verified in P-time
  - NPC:
    - Hardest problems in NP
    - P-time algorithms for an NPC problem means P-time algorithm for every problem in NP

- 3SAT is NP-complete: Can reduce SAT to 3SAT (SAT ≤ₚ 3SAT)
  - Can define a P-time function f s.t. x ∈ SAT iff f(x) ∈ 3SAT
SAT Remarks

- Can use SAT to check validity
- How?
  - $\phi$ is valid iff $\neg\phi$ is not SAT
  - $\phi$ is SAT iff $\neg\phi$ is not valid
- So, does that prove that validity is NPC?
- Random SAT:
  - Phase transition phenomena, e.g., ~4.26 for 3SAT
  - Local search methods
  - Algorithms: WalkSAT, Survey propagation, ...
- Special cases: 2SAT, Horn SAT, Dual-horn SAT, MAX SAT
Algorithms for SAT

- Modern SAT solvers accept input in CNF
  - Dimacs format:
    - 1 -3 4 5 0
    - 2 -4 7 0
    - ...
- Davis & Putnam Procedure (DP)
  - Dates back to the 50’s
  - Based on resolution (modern algorithms are not)
  - Helps to explain learning
Resolution

Resolution rule:

\[
\begin{array}{c}
C, v \\
\hline
D, \neg v \\
\end{array}
\Rightarrow
\begin{array}{c}
\neg v, v \\
C, D \\
\neg v, v \notin C \cup D
\end{array}
\]

- Soundness of rule: above line implies below line
- Also below line is SAT, so is above line (w/ side conditions)
- DP SAT algorithm
  - Base case: empty clause: UNSAT
  - Base case: no clauses: SAT
  - Remove clauses containing pure literals
  - Choose var, perform all possible resolutions, remove trivial clauses and clauses containing x
- Problem: space blow-up
Boolean Constraint Propagation

Unit resolution rule:

\[
\begin{array}{c}
C, \neg \ell, \ell \\
C
\end{array}
\]

- **BCP**: given a set of clauses including \{\ell\}
  - remove all other clauses containing \ell (subsumption)
  - remove all occurrences of \neg \ell in clauses (unit resolution)
  - repeat until a fixpoint is reached
- Shannon expansion: \( f(x) \equiv [x \land f(1)] \lor [x \land f(0)] \)
DPLL SAT Algorithm

- BCP
- Base case: empty clause: UNSAT
- Remove clauses containing pure literals
- Base case: no clauses: SAT
- Choose some var, say x (has to appear in both phases)
  - Add \{x\} and recursively call DPLL
  - Add \{\neg x\} and recursively call DPLL
  - If one of the calls returns SAT, return SAT
  - Else return UNSAT
- Correctness follows from Shannon expansion
- In contrast to DP, space is not a problem
DPLL is DFS Search of SAT-Tree

$(x_1' + x_2')$
$(x_1' + x_2 + x_3')$
$(x_1' + x_3 + x_4')$
$(x_1 + x_4)$

Lintao Zhang
DPLL is DFS Search of SAT-Tree

\[(x_1' + x_2')\]
\[(x_1' + x_2 + x_3')\]
\[(x_1' + x_3 + x_4')\]
\[(x_1 + x_4)\]
DPLL is DFS Search of SAT-Tree

\((x_1' + x_2')\)
\((x_1' + x_2 + x_3')\)
\((x_1' + x_3 + x_4')\)
\((x_1 + x_4)\)

Lintao Zhang
DPLL is DFS Search of SAT-Tree

\((x_1' + x_2')\)
\((x_1' + x_2 + x_3')\)
\((x_1' + x_3 + x_4')\)
\((x_1 + x_4)\)

Lintao Zhang
Modern DPLL

- Decision heuristics
  - Many have been tried; we’ll look at VSIDS
- Efficient BCP
  - BCP is the workhorse of modern SAT solvers
  - 2-literal watching
- Non-chronological backtracking
  - Can make a huge difference
- Clause learning
  - Records non-trivial implications discovered during search
  - Avoids re-exploring similar parts of state space
  - A disciplined form of resolution, but can still lead to space blow-up
- Preprocessing: limited resolution, subsumption, etc
- Restarts: clause learning helps guide SAT solver to solution
Decision Heuristics

- How do we decide what variable to split on?
- Variable State Independent Decaying Sum (VSIDS)
  - Keeps a score for each phase of a variable
  - Initially: the number of occurrences of a literal
  - Increases score by a constant whenever an added clause contains the variable
  - Periodically all the scores are divided by a constant
  - Choose free variable with the highest combined score
- VSIDS score is a literal occurrence count with higher weight on the more recently added clauses.
- VSIDS scores do not depend on the variable assignment
- Cheap to maintain (takes small percentage of the total run time)
Conflict Driven Learning and Non-chronological Backtracking

\[
x_1 + x_4 \\
x_1 + x_3' + x_8' \\
x_1 + x_8 + x_12 \\
x_2 + x_{11} \\
x_7' + x_3' + x_9 \\
x_7' + x_8 + x_9' \\
x_7 + x_8 + x_{10'} \\
x_7 + x_{10} + x_{12'}
\]
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10'} \]
\[ x_7 + x_{10} + x_{12'} \]

Lintao Zhang
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_12 \]
\[ x_2 + x_11 \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_10' \]
\[ x_7 + x_10 + x_12' \]

Lintao Zhang
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_{7'} + x_3' + x_9 \]
\[ x_{7'} + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10'} \]
\[ x_7 + x_{10} + x_{12'} \]

Lintao Zhang
Conflict Driven Learning and Non-chronological Backtracking

\[
x_1 + x_4 \\
x_1 + x_3' + x_8' \\
x_1 + x_8 + x_12 \\
x_2 + x_11 \\
x_7' + x_3' + x_9 \\
x_7' + x_8 + x_9' \\
x_7 + x_8 + x_10' \\
x_7 + x_10 + x_12' \\
\]

Lintao Zhang
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10'} \]
\[ x_7 + x_{10} + x_{12'} \]
Conflict Driven Learning and Non-chronological Backtracking

\[
\begin{align*}
&x_1 + x_4 \\
&x_1 + x_3' + x_8' \\
&x_1 + x_8 + x_{12} \\
&x_2 + x_{11} \\
&x_7' + x_3' + x_9 \\
&x_7' + x_8 + x_9' \\
&x_7 + x_8 + x_{10}' \\
&x_7 + x_{10} + x_{12}'
\end{align*}
\]
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_12 \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10'} \]
\[ x_7 + x_{10} + x_{12'} \]

\( x_1 = 0, x_4 = 1 \)
\( x_3 = 1, x_8 = 0, x_{12} = 1 \)
\( x_2 = 0, x_{11} = 1 \)
Conflict Driven Learning and Non-chronological Backtracking

- $x_1 + x_4$
- $x_1 + x_3' + x_8'$
- $x_1 + x_8 + x_{12}$
- $x_2 + x_{11}$
- $x_7' + x_3' + x_9$
- $x_7' + x_8 + x_9'$
- $x_7 + x_8 + x_{10}'$
- $x_7 + x_{10} + x_{12}'$

Diagram:

- $x_1 = 0, x_4 = 1$
- $x_3 = 1, x_8 = 0, x_{12} = 1$
- $x_2 = 0, x_{11} = 1$
- $x_7 = 1$
Conflict Driven Learning and Non-chronological Backtracking

x1 + x4
x1 + x3' + x8'
x1 + x8 + x12
x2 + x11
x7' + x3' + x9
x7' + x8 + x9'
x7 + x8 + x10'
x7 + x10 + x12'

x1 = 0, x4 = 1
x3 = 1, x8 = 0, x12 = 1
x2 = 0, x11 = 1
x7 = 1, x9 = 0, 1
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10}' \]
\[ x_7 + x_{10} + x_{12}' \]
Contra-proposition:

- If \( a \) implies \( b \), then \( b' \) implies \( a' \)

\[
x_3=1 \land x_7=1 \land x_8=0 \rightarrow \text{conflict}
\]
\[
\text{Not conflict} \rightarrow (x_3=1 \land x_7=1 \land x_8=0)'
\]
\[
\text{true} \rightarrow (x_3=1 \land x_7=1 \land x_8=0)'
\]
\[
(x_3=1 \land x_7=1 \land x_8=0)'
\]
\[
(x_3' + x_7' + x_8)
\]

Lintao Zhang
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10'} \]
\[ x_7 + x_{10} + x_{12'} \]
Conflict Driven Learning and Non-chronological Backtracking

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10'} \]
\[ x_7 + x_{10} + x_{12'} \]

\[ x_3=1, x_8=0, x_{12}=1 \]
\[ x_2=0, x_{11}=1 \]
\[ x_7=1, x_9=1 \]

\[ x_3=1 \land x_7=1 \land x_8=0 \rightarrow \text{conflict} \]

Add conflict clause: \( x_3'+x_7'+x_8 \)
Non-Chronological Backtracking and Learning

\[ x_1 + x_4 \]
\[ x_1 + x_3' + x_8' \]
\[ x_1 + x_8 + x_{12} \]
\[ x_2 + x_{11} \]
\[ x_7' + x_3' + x_9 \]
\[ x_7' + x_8 + x_9' \]
\[ x_7 + x_8 + x_{10}' \]
\[ x_7 + x_{10} + x_{12}' \]
\[ x_3' + x_8 + x_7' \]
DLL with Non-Chronological Backtracking and Learning

x1 + x4
x1 + x3' + x8'
x1 + x8 + x12
x2 + x11
x7' + x3' + x9
x7' + x8 + x9'
x7 + x8 + x10'
x7 + x10 + x12'
x3' + x8 + x7'

Lintao Zhang