Davis & Putnam Procedure (DP)

- dates back to the 50ies:

  original version is *resolution based* (less successful)

- **idea:** case analysis (try $x = 0, 1$ in turn and recurse)

- most successful SAT solvers (autumn 2003)

  works for very large instances

- recent ($\leq$ 10 years) optimizations:

  backjumping, learning, UIPs, dynamic splitting heuristics, fast data structures

  (we will have a look at each of them)

Resolution

- basis for first (less successful) resolution based DP

- can be extended to first order logic

- helps to explain learning

Resolution Rule

$$\frac{C \cup \{v\} \qquad D \cup \{\neg v\}}{C \cup D} \qquad \{v, \neg v\} \cap C = \{v, \neg v\} \cap D = \emptyset$$

Read: resolving the clause $C \cup \{v\}$ with the clause $D \cup \{\neg v\}$, both above the line, on the variable v, results in the clause $D \cup C$ below the line.

Correctness of Resolution Rule

Usage of such rules: if you can derive what is above the line (premise) then you are allowed to deduce what is below the line (conclusion).

Theorem. (premise satisfiable $\Rightarrow$ conclusion satisfiable)

$$\sigma(C \cup \{v\}) = \sigma(D \cup \{\neg v\}) = 1 \quad \Rightarrow \quad \sigma(C \cup D) = 1$$

Proof.

let $c \in C$, $d \in D$ with $\quad (\sigma(c) = 1$ or $\sigma(v) = 1)$ and $\quad (\sigma(d) = 1$ or $\sigma(\neg v) = 1)$

if $\quad \sigma(c) = 1$ or $\sigma(d) = 1 \quad$ conclusion follows immediately

otherwise $\quad \sigma(v) = \sigma(\neg v) = 1 \quad \Rightarrow$ contradiction

q.e.d.

Completeness of Resolution Rule

Theorem. (conclusion satisfiable $\Rightarrow$ premise satisfiable)

$$\sigma(C \cup D) = 1 \quad \Rightarrow \quad \exists \sigma' \quad \text{with} \quad \sigma'(C \cup \{v\}) = \sigma'(D \cup \{\neg v\}) = 1$$

Proof.

with out loss of generality pick $c \in C$ with $\sigma(c) = 1$

define $\quad \sigma'(x) = \begin{cases} 0 & \text{if } x = v \\ \sigma(x) & \text{else} \end{cases}$

since v and $\neg v$ do not occur in C, we still have $\sigma'(C) = 1$ and thus $\sigma'(C \cup \{v\}) = 1$

by definition $\sigma'(\neg v) = 1$ and thus $\sigma'(D \cup \{\neg v\}) = 1$

q.e.d.

Resolution Based DP

Idea: use resolution to *existentially* quantify out variables

1. if empty clause found then terminate with result **unsatisfiable**

2. find variables which only occur in one phase (only positive or negative)

3. remove all clauses in which these variables occur

4. if no clause left then terminate with result **satisfiable**

5. choose x as one of the remaining variables with occurrences in both phases

6. add results of all possible resolutions on this variable

7. remove all trivial clauses and all clauses in which x occurs

8. continue with **1.**

Example for Resolution DP

check whether XOR is weaker than OR, i.e. validity of:

$$a \vee b \rightarrow (a \oplus b)$$

which is equivalent to unsatisfiability of the negation:

$$(a \vee b) \wedge \neg(a \oplus b)$$

since negation of XOR is XNOR (equivalence):

$$(a \vee b) \wedge (a \leftrightarrow b)$$

we end up checking the following CNF for satisfiability:

$$(a \vee b) \wedge (\neg a \vee b) \wedge (a \vee \neg b)$$

Example for Resolution DP cont.

$$(a \vee b) \wedge (\neg a \vee b) \wedge (a \vee \neg b)$$

initially we can skip **1.** - **4.** of the algorithm and choose $x = b$ in **5.**

in **6.** we resolve $(\neg a \vee b)$ with $(a \vee \neg b)$ and $(a \vee b)$ with $(a \vee \neg b)$ both on b

and add the results $(a \vee \neg a)$ and $(a \vee a)$:

$$(a \vee b) \wedge (\neg a \vee b) \wedge (a \vee \neg b) \wedge (a \vee \neg a) \wedge (a \vee a)$$

the trivial clause $(a \vee \neg a)$ and clauses with ocurrences of b are removed:

$$(a \vee a)$$

in **2.** we find a to occur only positive and in **3.** the remaining clause is removed

the test in **4.** succeeds and the CNF turns out to be **satisfiable**

(thus the original formula is invalid – not a tautology)

Correctness of Resolution Based DP

Proof. in three steps:

(A) show that termination criteria are correct

(B) each transformation preserves satisfiability

(C) each transformation preserves unsatisfiability

Ad (A):

an empty clause is an empty disjunction, which is unsatisfiable

if literals occur only in one phase assign those to $1 \Rightarrow$ all clauses satisfied

Correctness of Resolution Based DP Part (B)

CNF transformations preserve satisfiability:

removing a clause does not change satisfiability

thus only adding clauses could potentially not preserve satisfiability

the only clauses added are the results of resolution

correctness of resolution rule shows:
if the original CNF is satisfiable, then the added clause are satisfiable
(even with the same satisfying assignment)

Correctness of Resolution Based DP Part (C) cont.

If we interpret $\cup$ as disjunction and clauses as formulae, then

$$(C_1 \vee x) \wedge \ldots \wedge (C_k \vee x) \quad \wedge \quad (D_1 \vee \neg x) \wedge \ldots \wedge (D_l \vee \neg x)$$

is, via distributivity law, equivalent to

$$(\underbrace{(C_1 \wedge \ldots \wedge C_k)}_{C} \vee x) \quad \wedge \quad (\underbrace{(D_1 \wedge \ldots \wedge D_l)}_{D} \vee \neg x)$$

and the same proof applies as for the completeness of the resolution rule.

Note: just using the completeness of the resolution rule alone does not work, since those σ' derived for multiple resolutions are formally allowed to assign different values for the resolution variable.

Correctness of Resolution Based DP Part (C)

CNF transformations preserve unsatisfiability:

adding a clause does not change unsatisfiability

thus only removing clauses could potentially not preserve unsatisfiability

trivial clauses $(v \vee \neg v \vee \ldots)$ are always valid and can be removed

let f be the CNF after removing all trivial clauses (in step **7.**)

let g be the CNF after removing all clauses in which x occurs (after step **7.**)

we need to show (f unsat $\Rightarrow$ g unsat), or equivalently (g sat $\Rightarrow$ f sat)

the latter can be proven as the completeness proof for the resolution rule
(see next slide)

Problems with Resolution Based DP

- if variables have many occurences, then many resolutions are necessary
- in the worst x and $\neg x$ occur in half of the clauses ...
- ... then the number of clauses increases quadratically
- clauses become longer and longer
- unfortunately in real world examples the CNF explodes
  (we will latter see how BDDs can be used to overcome some of these problems)
- How to obtain the satisfying assignment efficiently (counter example)?

Second version of DP

- resolution based version often called DP, second version DPLL
  (DP after [DavisPutnam60] and DPLL after [DavisLogemannLoveland62])
- it eliminates variables through case analysis: time vs space
- only *unit resolution* used (also called *boolean constraint propagation*)
- case analysis is on-the-fly:

  cases are not elaborated in a predefined fixed order, but ...

  ... only remaining crucial cases have to be considered
- allows sophisticated optimizations

Unit-Resolution

a *unit clause* is a clause with a single literal

in CNF a unit clause forces its literal to be assigned to 1

unit resolution is an application of resolution, where one clause is a unit clause

also called *boolean constraint propagation*

Unit-Resolution Rule

$$\frac{C \cup \{\neg l\} \qquad \{l\}}{C} \qquad \{l, \neg l\} \cap C = \emptyset$$

here we identify $\neg\neg v$ with v for all variables v.

Unit-Resolution Example

check whether XNOR is weaker than AND, i.e. validity of:

$$a \wedge b \rightarrow (a \leftrightarrow b)$$

which is equivalent to unsatisfiability of the CNF (exercise)

$$a \wedge b \wedge (a \vee b) \wedge (\neg a \vee \neg b)$$

adding clause obtained from unit resolution on a results in

$$a \wedge b \wedge (a \vee b) \wedge (\neg a \vee \neg b) \wedge (\neg b)$$

removing clauses containing a or $\neg a$

$$b \wedge (\neg b)$$

unit resolution on b results in an empty clause and we conclude unsatisfiability.

Ad: Unit Resolution

- if unit resolution produces a unit, e.g. resolving $(a \vee \neg b)$ with b produces a, continue unit resolution with this new unit
- often this repeated application of unit resolution is also called unit resolution
- unit resolution + removal of subsumed clauses never increases size of CNF

  $$C \text{ subsumes } D \quad :\Leftrightarrow \quad C \subseteq D$$

  a unit(-clause) l subsumes all clauses in which l occurs in the same phase
- *boolean constraint propagation* (BCP): given a unit l, remove all clauses in which l occurs in the same phase, and remove all literals $\neg l$ in clauses, where it occurs in the opposite phase (the latter is unit resolution)

Basic DPLL Algorithm

1. apply repeated unit resolution and removal of all subsumed clauses (BCP)

2. if empty clause found then return **unsatisfiable**

3. find variables which only occur in one phase (only positive or negative)

4. remove all clauses in which these variables occur (pure literal rule)

5. if no clause left then return **satisfiable**

6. choose x as one of the remaining variables with occurrences in both phases

7. recursively call DPLL on current CNF with the unit clause $\{x\}$ added

8. recursively call DPLL on current CNF with the unit clause $\{\neg x\}$ added

9. if one of the recursive calls returns **satisfiable** return **satisfiable**

10. otherwise return **unsatisfiable**

DPLL Example

$$(\neg a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee \neg b)$$

Skip **1.** - **6.**, and choose $x = a$. First recursive call:

$$(\neg a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee \neg b) \wedge a$$

unit resolution on a and removal of subsumed clauses gives

$$b \wedge (\neg b)$$

BCP gives empty clause, return **unsatisfiable**. Second recursive call:

$$(\neg a \vee b) \wedge (a \vee \neg b) \wedge (\neg a \vee \neg b) \wedge \neg a$$

BCP gives $\neg b$, only positive recurrence of b left, return **satisfiable**

(satisfying assignment $\{a \mapsto 0, b \mapsto 0\}$)

Expansion Theorem of Shannon

Theorem.

$$f(x) \quad \equiv \quad x \wedge f(1) \vee \bar{x} \wedge f(0)$$

Proof.

Let σ be an arbitrary assignment to variables in f including x

case $\sigma(x) = 0$:

$$\sigma(f(x)) = \sigma(f(0)) = \sigma(0 \wedge f(1) \vee 1 \wedge f(0)) = \sigma(x \wedge f(1) \vee \bar{x} \wedge f(0))$$

case $\sigma(x) = 1$:

$$\sigma(f(x)) = \sigma(f(1)) = \sigma(1 \wedge f(1) \vee 0 \wedge f(0)) = \sigma(x \wedge f(1) \vee \bar{x} \wedge f(0))$$

Correctness of Basic DPLL Algorithm

first observe: $x \wedge f(x)$ is satisfiable iff $x \wedge f(1)$ is satisfiable

similarly, $\bar{x} \wedge f(x)$ is satisfiable iff $\bar{x} \wedge f(0)$ is satisfiable

then use expansion theorem of Shannon:

$f(x)$ satisfiable iff $\bar{x} \wedge f(0)$ or $x \wedge f(1)$ satisfiable iff $\bar{x} \wedge f(x)$ or $x \wedge f(x)$ satisfiable

rest follows along the lines of the the correctness proof for resolution based DP