Introduction to KRR

Computing Propositional Logic

Ira A. Fulton Schools of Engineering
Arizona State University

Objectives

Objective

Apply the algorithm for computing models of propositional logic

DPLL: General Algorithm to Find Models of Propositional Formula

SAT Solver

- The propositional satisfiability problem (SAT) is the problem of deciding whether a given finite set of propositional formulas is satisfiable
- A SAT solver is a software tool for solving SAT
- Many existing SAT solvers are based on the Davis-Putnam-Logemann-Loveland procedure (DPLL), invented in 1962
- Most SAT solvers accepts CNF as input

Conjunctive Normal Form (1 of 3)

- A **literal** is either an atom p or its negation $\neg p$
- A **clause** is of the form $L_1 \vee \cdots \vee L_n$ $(n \geq 1)$ where each L_i is a literal
- A formula is in **conjunctive normal form (CNF)** if it is of the form
  - $F_1 \wedge \cdots \wedge F_m$ $(m \geq 1)$ where each F_i is a clause
- Are these in CNF?
  - $(\neg q \vee p \vee r) \wedge (\neg p \vee r) \wedge q$
  - $(\neg(q \vee p) \vee r) \wedge (\neg p \vee r) \wedge q$
  - $(\neg q \vee p \vee r)$

Conjunctive Normal Form (2 of 3)

- Any formula can be transformed into CNF
- CLAUSIFY(F)

  eliminate from $\boldsymbol{F}$ all connectives other than $\neg$, $\wedge$ *and* $\vee$;

  distribute $\neg$ over $\wedge$ and $\vee$ until it applies to atoms only;

  distribute $\vee$ over $\wedge$ until it applies to literals only;

  return the set of conjunctive terms of the resulting formula

- **Example:** $(\boldsymbol{p} \vee \neg \boldsymbol{q}) \rightarrow \boldsymbol{r}$ $\boldsymbol{u} \leftrightarrow \boldsymbol{p} \wedge \boldsymbol{q}$

Conjunctive Normal Form (3 of 3)

Q: How many clauses will be generated by Clausify(F) if F is $(p_1 \wedge q_1) \vee \cdots \vee (p_n \wedge q_n)$

A. 2^n

B. n^3

C. n^2

D. n

How Do We Avoid Blow-up?

- **A propositional formula F can be transformed into a "small" set of clauses Γ so that**
  - Γ is satisfiable iff F is satisfiable
  - Given an interpretation satisfying Γ, its projection onto the signature of F is an interpretation satisfying F

- **Example:**

F

$(p \vee q) \wedge r$

$\xrightarrow{\text{Tr}}$

Γ

$u \leftrightarrow p \vee q$

$u \wedge r$

Clausify*

Clausify*(F, Γ)

- If F is a conjunction of clauses $C_1 \wedge \cdots \wedge C_k$
  Then exit with $\{C_1, \ldots, C_n\} \cup \Gamma$;
- G := a minimal non-literal subformula of F;
- u := a new atom;
- F := the result of replacing G in F by u;
- Clausify*(F, $\mathbf{\Gamma} \cup$ Clausify($u \leftrightarrow G$))

Q: Apply Clausify* to $(p_1 \wedge q_1) \vee \ldots \vee (p_n \wedge q_n)$

Unit Propagation (1 of 3)

- **Sometimes satisfiability can be easily checked by "unit propagation."**
  - Example: $p \wedge (\neg p \vee \neg q) \wedge (\neg q \vee r)$
- **If a CNF formula contains a unit clause (a clause consisting of single literal), the formula can be simplified.**

Unit Propagation (2 of 2)

- **For any CNF F and atom A, $F|_A$ is obtained from F by replacing all occurrences of A by $\top$ and simplifying the result by removing**
  - all clauses containing the disjunctive term $\top$, and
  - the disjunctive terms $\neg\top$ in all remaining clauses.
- **Ex:** $\left(p \ \wedge (\neg p \vee \neg q) \wedge (\neg q \vee r)\right)|_{\boldsymbol{p}}$ =

- **Similarly, $F|_{\neg A}$ is the result of replacing all occurrences of A by $\bot$ and simplifying the result by removing**
  - all clauses containing the disjunctive term $\neg\bot$, and
  - the disjunctive terms $\bot$ in all remaining clauses.
- **Ex:** $(p \vee q \vee \neg r) \wedge (\neg p \vee r)|_{\neg p}$ =
- $\left(p \ \wedge (\neg p \vee \neg q) \wedge (\neg q \vee r)\right)|_{\neg \boldsymbol{p}}$ =

Unit Propagation (3 of 3)

Unit-Propagate (F, U)

While F contains no empty clause but has a unit clause L

$F \leftarrow F|_L$;

$U \leftarrow U \cup \{L\}$

end

After every execution of the body of the loop, the conjunction of *F* with the literals *U* remains equivalent to F_0.

Ex: Unit-Propagate $(\boldsymbol{p} \wedge (\neg \boldsymbol{p} \vee \neg \boldsymbol{q}) \wedge (\neg \boldsymbol{q} \vee \boldsymbol{r}) \wedge (\boldsymbol{q} \vee \neg \boldsymbol{r}), \emptyset)$

Limitation of Unit Propagation

Q: What if there is no unit clause?

E.g., $(\neg p \vee q) \wedge (\neg p \vee r) \wedge (\neg q \vee \neg r)$

DPLL

DPLL(F, U)

$\quad$ Unit-propagate(F, U);

$\quad$ **if** F contains the empty clause **then** return;

$\quad$ **if** $F = \top$ **then** exit with a model of U;

$\quad$ $L \leftarrow$ a literal containing an atom from F;

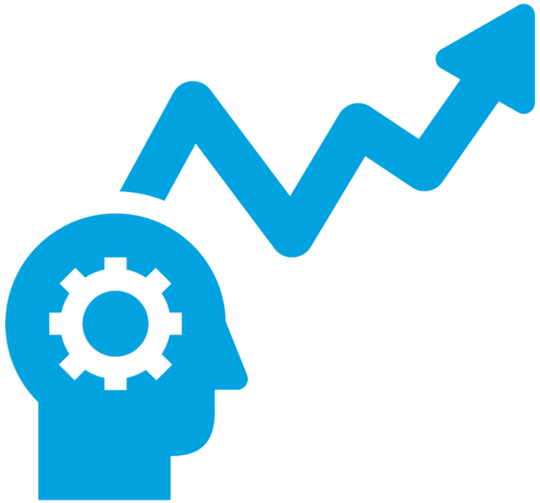
$\quad$ DPLL$(F|_L, U \cup \{L\})$;

$\quad$ DPLL$(F|_{\overline{L}}, U \cup \{\overline{L}\})$

To solve the satisfiability of *F*, call DPLL$(F, \emptyset)$

DPLL

Q: Apply DPLL to $(\neg p \lor q) \land (\neg p \lor r) \land (q \lor r) \land (\neg q \lor \neg r)$



Wrap-Up