COMP30026 Models of Computation

Symbolic Deduction

Lecture Week 3 Part 1 (Zo

Semester 2, 2021

This Lecture is Being Recorded

Mechanising Deduction

We must not think that computation . . . has place only in numbers.

— *T. Hobbes (1655)*

Calculu

Proposi

G. Boole, A. De Morgan, E. Schröder (19th centu

Predicate logic: G. Frege (1879).

Universal computers:
C. Babbage (19th century), A. Turing (20th century).

Propositional Logic is Decidable

Generating a truth table is just a way of systematically exploring the truth value of a formula, for each possible truth assignment, that is, using bru

Formul ide if a formula is satisfiable, and whether it is valid.

Unfortunately truth tables may grow exponenti formulas.

Faster Satisfiability/Validity Checking?

Are there faster decision procedures for propositional logic?

It depend

What if we

Then satisfiability is NP-complete and .

These are terms from complexity theory y
problem for propositional logic, SAT, has b

SAT and Complexity Theory

A who's who of important problems in scheduling, network flow and routing, rcuit fault dete

tractabl

Most computer scientists conclude from this that i that there are decision procedures for SAT (and hence f er problems) that perform much better than

Normal Forms for Propositional Logic

A literal is P or $\neg P$ where P is a propositional letter.

A formula is in conjunctive normal form (CNF) if it is a conjunction of disjunctions of literals (a conjunction of "clauses").

It is in disjunctive normal form (DNF) i
conjunctions of literals.

$$(\neg A \wedge \neg B) \vee (\neg B \wedge C$$

Theorem: Every propositional formula can be expressed in CNF, as well as in DNF.

Converting a Formula to CNF or to DNF

1. Eliminate all occurrences of $\oplus$ using $A \oplus B \equiv (A \lor B) \land (\neg A \lor \neg B)$.
2. Eli
   A

3. Eli B.
4. Use De Morgan's Laws to push $\neg$

5. Eliminate double negations using
6. Use the distributive laws to get the required form.

Example Conversion to CNF

$$(\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S$$

Example Conversion to CNF

$$
\begin{aligned}
& (\neg P \land (\neg Q \Rightarrow R)) \Leftrightarrow S \\
\equiv \quad & ((\neg P \land (\neg Q \Rightarrow R)) \Rightarrow S) \land (S \Rightarrow (\neg P \land (\neg Q \Rightarrow R))) \qquad (2)
\end{aligned}
$$

$$
\begin{aligned}
& (\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S \\
\equiv \; & ((\neg P \wedge (\neg Q \Rightarrow R)) \Rightarrow S) \wedge (S \Rightarrow (\neg P \wedge (\neg Q \Rightarrow R))) \qquad (2) \\
\equiv \; & (\neg \qquad)))
\end{aligned}
$$

$$
\begin{aligned}
& (\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S \\
\equiv\ & ((\neg P \wedge (\neg Q \Rightarrow R)) \Rightarrow S) \wedge (S \Rightarrow (\neg P \wedge (\neg Q \Rightarrow R))) \quad (2) \\
\equiv\ & (\neg \qquad))) \\
\equiv\ & (\neg \qquad))) \quad (3)
\end{aligned}
$$

Example Conversion to CNF

$$
\begin{aligned}
& (\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S \\
\equiv\ & ((\neg P \wedge (\neg Q \Rightarrow R)) \Rightarrow S) \wedge (S \Rightarrow (\neg P \wedge (\neg Q \Rightarrow R))) && (2) \\
\equiv\ & (\neg \quad \ldots \quad))) \\
\equiv\ & (\neg \quad \ldots \quad))) && (3) \\
\equiv\ & ((\quad \ldots \quad \vee R))) && (4)
\end{aligned}
$$

Example Conversion to CNF

$$
\begin{aligned}
& (\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S \\
\equiv\ & ((\neg P \wedge (\neg Q \Rightarrow R)) \Rightarrow S) \wedge (S \Rightarrow (\neg P \wedge (\neg Q \Rightarrow R))) && (2) \\
\equiv\ & (\neg \quad))) \\
\equiv\ & (\neg \quad))) && (3) \\
\equiv\ & ((\quad \vee R))) && (4) \\
\equiv\ & ((P \vee (\neg Q \wedge \neg R)) \vee S) \wedge (\neg S \vee (
\end{aligned}
$$

Example Conversion to CNF

$$(\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S$$

$$\equiv ((\neg P \wedge (\neg Q \Rightarrow R)) \Rightarrow S) \wedge (S \Rightarrow (\neg P \wedge (\neg Q \Rightarrow R))) \quad (2)$$

$$\equiv (\neg \quad \ldots \quad)))$$

$$\equiv (\neg \quad \ldots \quad))) \quad (3)$$

$$\equiv ((\quad \ldots \quad \vee R))) \quad (4)$$

$$\equiv ((P \vee (\neg Q \wedge \neg R)) \vee S) \wedge (\neg S \vee (\quad \ldots$$

$$\equiv (((P \vee \neg Q) \wedge (P \vee \neg R)) \vee S)$$

$$\wedge ((\neg S \vee \neg P) \wedge (\neg S \vee (Q \vee \quad \ldots \quad (6)$$

Example Conversion to CNF

$$
\begin{aligned}
& (\neg P \wedge (\neg Q \Rightarrow R)) \Leftrightarrow S \\
\equiv\; & ((\neg P \wedge (\neg Q \Rightarrow R)) \Rightarrow S) \wedge (S \Rightarrow (\neg P \wedge (\neg Q \Rightarrow R))) && (2) \\
\equiv\; & (\neg \quad))) \\
\equiv\; & (\neg \quad))) && (3) \\
\equiv\; & ((\quad \vee R))) && (4) \\
\equiv\; & ((P \vee (\neg Q \wedge \neg R)) \vee S) \wedge (\neg S \vee (\\
\equiv\; & (((P \vee \neg Q) \wedge (P \vee \neg R)) \vee S) \\
& \quad \wedge ((\neg S \vee \neg P) \wedge (\neg S \vee (Q \vee && (6) \\
\equiv\; & (P \vee \neg Q \vee S) \wedge (P \vee \neg R \vee S) \\
& \quad \wedge (\neg S \vee \neg P) \wedge (\neg S \vee Q \vee R) && (6)
\end{aligned}
$$

The result is in conjunctive normal form.

Reduced CNF

There is one more transformation that it is easy and natural to do: simplifying clauses.

A CNF for [illegible] es,
no propo [illegible]

The trans [illegible]

$$(A \lor \neg B \lor \neg A) \land (\neg C \lor \neg B)$$

becomes

$$(\neg C \lor \neg B) \land (C \lor \neg A \lor B)$$

Normal Form Does Not Mean Unique Form

The formula

$$(P \land Q \land R) \lor (\neg P \land \neg Q \land \neg R) \lor (\neg P \land R) \lor (Q \land \neg R)$$

is in reduc [illegible]

$$\neg P \lor Q$$

So two DNF formulas may have different sizes and v [illegible] d yet be equivalent.

Similarly for (R)CNF.

Canonical Forms: Xor Normal Form

If a normal form leads to a unique representation for every Boolean function, we call it canonical.

One canonical form ("xor normal form") presents the function in a sum-of-

For exam

$$ABC \oplus AC \oplus$$

This form is unique, up to reordering of conjuncts and ds.

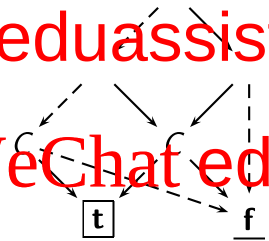
Or, representing the summands as sets:

$$\{\{A, B, C\}, \{A, C\}, \{B\}, \{C\}\}$$

Canonical Forms: ROBDDs

Binary decision diagrams (BDDs) give another canonical form.

Here $(A \Rightarrow B) \wedge (B \oplus C)$ is represented by the graph



Read: If A then [follow solid arc] else [follow dashed arc].

Validity and Satisfiability with ROBDDs

This graph representation becomes canonical when we enforce maximal sharing of subgraphs (and agree on an ordering of variables, like A, B, C).

The resul [illegible] DD

These have been very popular and useful for hardware verification etc.

Clearly a propositional formula is valid iff its ROBDD [illegible] single-leaf graph $\boxed{\mathbf{t}}$.

It is unsatisfiable iff its ROBDD is $\boxed{\mathbf{f}}$.

CNF and Clausal Form

Knowledge bases are often presented in CNF, as a set (conjunction) of clauses.

A claus

Abstrac rk of a formula i te

$$(P \vee \neg Q \vee S) \wedge (P \vee \neg R \vee S) \wedge (\quad R)$$

as

$$\{\{P, S, \neg Q\}, \{P, S, \neg R\}, \{\neg P, \neg S\}, \{Q, R, \neg S\}\}$$

We shall often make no distinction between these.

Clause $\{A, B\}$ represents $A \vee B$, and clause $\{A\}$ represents A.

How shal

The natu ment for

$\vee$—we could have written $A \vee B$ as $\mathbf{f} \vee$.

Hence we agree that the empty clause

Empty CNF Formulas

The formula $\{C_1, C_2\}$, with clauses C_1 and C_2, represents $C_1 \wedge C_2$.

The for

How shal

The natural reading is that it is true, bec or $\wedge$—we could have written $C_1 \wedge C_2$ as **t** $\wedge C$.

Hence we agree that the empty formul

Empty Clauses and Formulas

For clausal form representation (CNF) we then have:

- The set $\emptyset$ of clauses is valid.
- An

Don't be c

An empty set of clauses is valid, because it is trivial to sati l of the set's clauses—there is nothing to do

But a (non-empty) set that contains an
satisfied, because nothing satisfies that empty clause.

In particular, note that $\{\emptyset\} \neq \emptyset$.

Consider the two clauses $\neg P \vee A$ and $P \vee B$

- If P is true, they reduce to A and $\mathbf{t}$.
- If P is false, they reduce to $\mathbf{t}$ and B.

There are … d $(P \vee B)$ are both tr… the claus…

We call $A \vee B$ their resolvent.

both true $\quad \neg P \vee A \quad P$

this true $\quad A \vee B \quad \leftarrow$ the resolvent

Let C_1 and C_2 be clauses such that $P \in C_1$ and $\neg P \in C_2$.

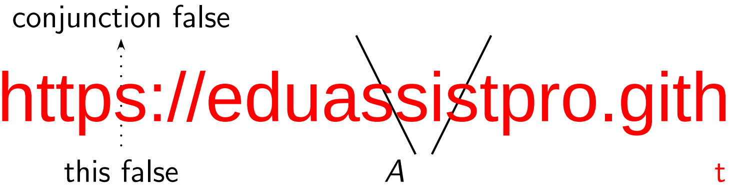
$(C_1 \setminus \{$ [illegible]

(Note: [illegible] t not in B.)

Theorem. If R is a resolvent of C_1 an [illegible] .

This generalises the well-known inference rule of [illegible]:
From A and $A \Rightarrow B$ deduce B.

Resolution suggests a way of verifying that a CNF formula is unsatisfiable.



If, through a number of resolution steps, we can derive the empty clause $\bot$, then the original set of clauses were unsatisfiable.

We talk about a refutation proof.

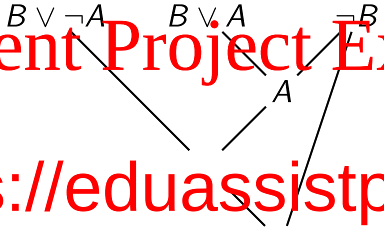
Deductions and Refutations

A resolution deduction of clause C from a set S of clauses is a finite sequence $C_1, C_2, \ldots, C_n$ of clauses such that $C_n = C$ and for each i, $1 \leq i \leq$

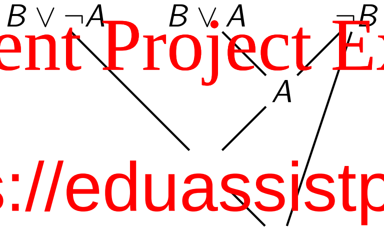
- a me
- a resolvent of C_j and C_k, for some

A resolution refutation of a set S of clau

of $\perp$ from S.



This shows that $(B \vee \neg A) \wedge (B \vee A) \wedge$



This shows that $(B \vee \neg A) \wedge (B \vee A) \wedge$

Exercise:

Find a simpler refutation to show the formula is a contradiction.

Refutation Exercise

Seek a resolution refutation of

$$(A \lor B) \land (\neg A \lor \neg B)$$

Refutation Exercise

Seek a resolution refutation of

$$(A \vee B) \wedge (\neg A \vee \neg B)$$

Not possible to derive $\perp$ here
—which is fortunate, because the original formul .

Note in particular that we cannot "cancel out" several literals in one go.

How to Use Refutations (1)

To show that F is valid, first put $\neg F$ in RCNF, yielding a set S of clauses.

Then ref

Conside ($(P \land Q))$

Negating yields: $\neg($ $Q)))$

Pushing negation then yields: $\neg Q)$

From this we can derive $\perp$ in a single resolution step.

How to Use Refutations (2)

Suppose we express a circuit design as a formula F in RCNF.

Suppose we wish to show that the design satisfies some property G, that is, sho

We can ex

Hence a strategy is:

1. Negate G and bring it into RCNF;
2. add those clauses to the set F; and
3. find a refutation of the resulting set of clauses.

After the Break

We move

Grok Wo ine of
22 August. Remember: No extensions are possible