Logic and Computation – CS 2800
Fall 2019

Lecture 12
Propositional logic continued
CNF, DNF, complete Boolean bases

Stavros Tripakis

Northeastern University
Khoury College of Computer Sciences

Discuss homework 02 survey

- How many courses are you taking?
- How many hours per course would you like to be spending, ideally?
  - Including everything: attending lectures, reading, homeworks, projects, piazza, …
- How many of those hours on homework?

Outline

- Properties of Boolean operators: read and assimilate Section 3.3 of lecture notes!
- Normal forms, DNF and CNF
- Complete Boolean bases

Properties of Boolean operators

- Review lecture notes, section 3.3

Normal forms, CNF, DNF

Negation Normal Form (NNF)

- "Push" all negations all the way to the leaves of the syntax tree of the formula (literals)
- Eliminate double negations
- Examples:

$$\neg(p \wedge q) =$$

$$\neg(p \vee q) =$$

$$\neg(p \Rightarrow q) =$$

Negation Normal Form (NNF)

- "Push" all negations all the way to the leaves of the syntax tree of the formula (literals)
- Eliminate double negations
- Examples:

$$\neg(p \wedge q) = \neg p \vee \neg q$$

$$\neg(p \vee q) = \neg p \wedge \neg q$$

$$\neg(p \Rightarrow q) = \neg(\neg p \vee q) = \neg\neg p \wedge \neg q = p \wedge \neg q$$

Disjunctive Normal Form (DNF)

- A formula is in DNF if it is a disjunction of conjunctions of literals
- Literal = either a variable or a negated variable
- Examples: which formulas below are in DNF?

$p \lor q$ $\quad$ $p \land q$ $\quad$ $p \Rightarrow q$

$p \lor (q \land \neg r)$ $\quad$ $p \land (q \lor \neg r)$ $\quad$ $true$

$(p_1 \land q_1) \lor (p_2 \land q_2) \lor (p_3 \land q_3)$ $\quad$ $false$

Conjunctive Normal Form (CNF)

- A formula is in CNF if it is a conjunction of disjunctions of literals
- A disjunction of literals is also called a **clause**
- Examples: which formulas below are in CNF?

$$p \vee q \qquad p \wedge q \qquad p \Rightarrow q$$

$$p \vee (q \wedge \neg r) \qquad p \wedge (q \vee \neg r) \qquad true$$

$$(p_1 \vee q_1) \wedge (p_2 \vee q_2) \wedge (p_3 \vee q_3) \qquad false$$

Can we transform any Boolean formula into DNF?

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

- Yes
- Brute-force method:
  1. Build the truth table of the formula
  2. For each row that gives “T”, generate a conjunction of literals
  3. Take the disjunction of all conjunctions generated in step 2

Can we transform any Boolean formula into DNF?

p	q	p ⊕ q
T	T	F
T	F	T
F	T	T
F	F	F

- Yes
- Brute-force method:
  1. Build the truth table of the formula
  2. For each row that gives “T”, generate a conjunction of literals
  3. Take the disjunction of all conjunctions generated in step 2
- Our example: $(p \wedge \neg q) \vee (\neg p \wedge q)$
- Is this method efficient? No: it’s exponential!
- Does there exist a fundamentally better (polynomial) method?
- No: if there were, I could solve the SAT problem in polynomial time – how?

Sometimes there are better (shorter) DNFs

- Consider function f:
- What DNF does the brute-force method give?

$$(p \wedge q \wedge r) \vee$$
$$(p \wedge q \wedge \neg r) \vee$$
$$(\neg p \wedge q \wedge r) \vee$$
$$\cdots$$

p	q	r	f
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	F
F	T	T	T
F	T	F	T
F	F	T	T
F	F	F	T

- Is there a better DNF? $\neg p \vee q$

How long can the DNF get in the worst case?

- Consider this CNF formula (4 clauses):

$$(p_1 \vee q_1) \wedge (p_2 \vee q_2) \wedge (p_3 \vee q_3) \wedge (p_4 \vee q_4)$$

- Quiz: What's the minimal number of conjunctions needed for the corresponding DNF?
  - A: 1 D: 8
  - B: 3 E: 16
  - C: 6 F: 64

Can we transform any Boolean formula into CNF?

- Yes
- Brute-force method: the dual of the method for DNF
  1. Build the truth table of the formula
  2. For each row that gives "F", generate a conjunction of literals, and then negate it, obtaining a clause
  3. Take the conjunction of all clauses generated in step 2

- Is this method efficient? No: it's exponential!
- Does there exist a fundamentally better (polynomial) method?
  - **Yes! Tseytin transformation.**

Tseytin transformation by example

- It preserves satisfiability of the original formula, by adding extra variables $\Rightarrow$ resulting formula is **equisatisfiable**.

Example:

$$ab + cd \quad \rightsquigarrow \quad \underbrace{(a+c)(a+d)(b+c)(b+d)}_{\text{standard translation (equivalent CNF formula)}}$$

$$\begin{aligned} ab + cd &\rightsquigarrow (z_1 + z_2)(z_1 \leftrightarrow ab)(z_2 \leftrightarrow cd) \\ &\rightsquigarrow (z_1 + z_2)(z_1 \rightarrow ab)(ab \rightarrow z_1)(z_2 \rightarrow cd)(cd \rightarrow z_2) \end{aligned}$$

$$\rightsquigarrow \underbrace{(z_1 + z_2)(\overline{z_1} + a)(\overline{z_1} + b)(\overline{a} + \overline{b} + z_1)(\overline{z_2} + c)(\overline{z_2} + d)(\overline{c} + \overline{d} + z_2)}_{\text{translation adding new variables } z_1, z_2}$$

Idea: z_1 represents ab and z_2 represents cd.

Note: $a \cdot b = ab = a \wedge b$, $a + b = a \vee b$, $\bar{a} = \neg a$

Tseytin transformation by example

The new translation may appear to give longer formulas ...

$$ab + cd \quad \leadsto \quad \underbrace{(a+c)(a+d)(b+c)(b+d)}_{\text{standard translation (equivalent CNF formula)}}$$

$$ab + cd \quad \leadsto \quad \underbrace{(z_1 + z_2)(\overline{z_1} + a)(\overline{z_1} + b)(\overline{a} + \overline{b} + z_1)(\overline{z_2} + c)(\overline{z_2} + d)(\overline{c} + \overline{d} + z_2)}_{\text{translation adding new variables } z_1, z_2}$$

... until we consider the generalization:

$$a_1 b_1 + \cdots + a_n b_n \quad \leadsto \quad \underbrace{(a_1 + \cdots + a_n)(a_1 + \cdots + b_n) \cdots (b_1 + \cdots + b_n)}_{\text{standard translation size: } O(2^n)}$$

$$\underbrace{(z_1 + \cdots + z_n)(\overline{z_1} + a_1)(\overline{z_1} + b_1)(\overline{a_1} + \overline{b_1} + z_1) \cdots (\overline{z_n} + a_n)(\overline{z_n} + b_n)(\overline{a_n} + \overline{b_n} + z_n)}_{\text{new translation size: } O(3n+1) = O(n)}$$

Uses of CNF and DNF

- CNF: standard input format for SAT solvers (DIMACS format)
  - Avoids irrelevant details on parsing, simplification, etc
  - Efficient (polynomial) transformations to CNF exist (Tseytin transformation)

- Minimizing DNF has many applications:
  - Logic synthesis, reliability, …

- Several other "normal" ("canonical") forms: binary-decision diagrams (BDDs), …
  - E.g., see course CS-4830 on model-checking

Complete Boolean bases

Boolean functions

- Boolean function = truth table

p	$\neg p$
T	F
F	T

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

p	q	$p \Rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

p	q	$p \equiv q$
T	T	T
T	F	F
F	T	F
F	F	T

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

- How many Boolean functions of arity N are there?
  - Arity N => truth table has 2^N rows
  - Each row can have 2 possible values (T/F)
  - 2^{2^N} possible functions!

p	q	r	$ite(p,q,r)$
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	F
F	T	T	T
F	T	F	F
F	F	T	T
F	F	F	F

p	q	p ⊕ q
T	T	F
T	F	T
F	T	T
F	F	F

p	q	r	f
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	F
F	T	T	T
F	T	F	T
F	F	T	T
F	F	F	T

Boolean functions

- How many Boolean functions of arity 2 are there?
  - $2^{2^2} = 2^4 = 16$ possible functions
  - We have only seen some of those:

Can you think of a few more?

p	q	$p \wedge q$
T	T	T
T	F	F
F	T	F
F	F	F

p	q	$p \vee q$
T	T	T
T	F	T
F	T	T
F	F	F

p	q	$p \Rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

p	q	$p \equiv q$
T	T	T
T	F	F
F	T	F
F	F	T

p	q	$p \oplus q$
T	T	F
T	F	T
F	T	T
F	F	F

Boolean functions

- How many Boolean functions of arity 3 are there?
  - $2^{2^3} = 2^8 = 256$ possible functions
- How many Boolean functions of arity 4 are there?
  - $2^{2^4} = 2^{16} = 65536$ possible functions
- How many Boolean functions of arity 5 are there?
  - $2^{2^5} = 2^{32} = 4294967296$ possible functions

Can we represent all those functions
with the basic functions we have seen
(and, or, not, …) ?

Complete Boolean base

- Can we define all Boolean functions using some basic Boolean functions (and, or, not, …)?
- Yes: e.g., use DNF. DNF only uses $\wedge$, $\vee$, $\neg$
- We say that the set $\{\wedge, \vee, \neg\}$ is a **complete Boolean base**
- Is there a smaller complete Boolean base? (with fewer functions)
- Yes! $\{\wedge, \neg\}$ is one, and $\{\vee, \neg\}$ is another

Next time

- Formula simplification
- Equational proofs

- Read Chapter 3 until the end