CSE 20
DISCRETE MATH

New HW deadline: Saturday 11pm

Fall 2017

http://cseweb.ucsd.edu/classes/fa17/cse20-ab/

Today's learning goals

- Prove propositional equivalences using truth tables
- Prove propositional equivalences using other known equivalences, e.g.
  - DeMorgan’s laws
  - Double negation laws
  - Distributive laws, etc.
- Compute the CNF and DNF of a given compound proposition.

(Some) Useful equivalences

output 1 | output 2

F F T T

Rosen p. 26-28

$$\neg(p \wedge q) \equiv \neg p \vee \neg q \qquad \neg(p \vee q) \equiv \neg p \wedge \neg q$$ – De Morgan's

$$p \vee q \equiv q \vee p \qquad p \wedge q \equiv q \wedge p$$ – commutative

$$p \wedge F \equiv F \qquad p \vee T \equiv T$$

$$p \wedge T \equiv p \qquad p \vee F \equiv p$$

$$p \to q \equiv \neg p \vee q \qquad p \to q \equiv \neg q \to \neg p$$ – Contrapositive

$$\neg(p \to q) \equiv \neg(\neg p \vee q) \equiv p \wedge \neg q$$

…. 32 equivalences listed in book!

Can replace *p* and *q* with any (compound) proposition

(Some) Useful equivalences

Rosen p. 26-28

For constructing (minimal) circuits with specified gates

- **only NOTs?**
- **only ANDs?**

$p \wedge T \equiv p$

$p \vee F \equiv p$

$p \wedge \neg p \equiv F$

…. 32 equivalences listed in book!

(Some) Useful equivalences

Rosen p. 26-28

For simplying and evaluating complicated compound propositions

- **Remove parentheses?**
- **Reduce subexpressions to simpler ones**

…. 32 equivalences listed in book!

(Some) Useful equivalences

Rosen p. 26-28

For devising proofs of statements

- **Translate using existing logical structure.**
- **Try to apply known proof strategy.**
- **Rewrite in equivalent way to apply additional proof strategies.**

(more on this later)

HYP CONC HYP CONC

$p \to (q \to r) \equiv p \to (\neg q \vee r) \equiv \neg p \vee \neg q \vee r$

Sample equivalence proof

Prove that $(p \wedge q) \to r$ is logically equivalent to $p \to (q \to r)$

Choice (1) Write out truth table

output 1	output 2

Choice (2) Apply known logical equivalences

Equiv in book DM

$(p \wedge q) \to r \equiv \neg (p \wedge q) \vee r \equiv \neg p \vee \neg q \vee r$

Are these compound propositions logically equivalent to $(p \to q) \to r$?

BONUS

No: Pf give example eg. $P = F$ $Q = ?$ $R = F$

Other laws of equivalence

Rosen p. 29-31

Any compound proposition can be translated to one using …

A. only ANDs.
B. only ORs.
C. only IFs.
D. only NOTs.
E. None of the above

Other laws of equivalence

Rosen p. 35 #42-53

Any compound proposition can be translated to one using …

A. only ANDs.
B. only ORs.
C. only IFs.
D. only NOTs.
E. None of the above

Functionally complete collection of connectives.

output

F

Functionally complete set of connectives *Rosen p. 35 #42-53*

Claim: The connectives AND, NOT are functionally complete.

Any compound proposition can be rewritten as a logically equivalent one that only has the operators AND, NOT

Functionally complete set of connectives *Rosen p. 35 #42-53*

Claim: The connectives AND, NOT are functionally complete.

Any compound proposition can be rewritten as a logically equivalent one that only has the operators AND, NOT

Example: $p \wedge (q \wedge \neg r)$

Circuits?

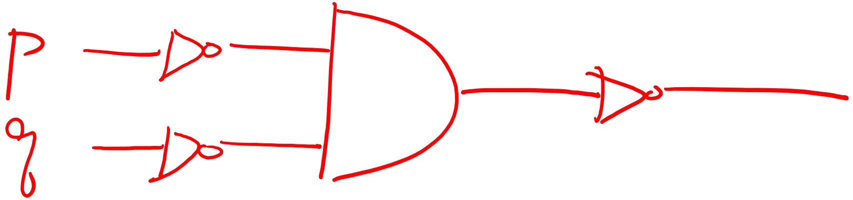
Functionally complete set of connectives *Rosen p. 35 #42-53*

Claim: The connectives AND, NOT are functionally complete.

Any compound proposition can be rewritten as a logically equivalent one that only has the operators AND, NOT

Example: $p \vee q$ $\neg(\neg(p \vee q)) \equiv \neg(\neg p \wedge \neg q)$

Circuits?



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

Functionally complete set of connectives *Rosen p. 35 #42-53*

Claim: The connectives AND, NOT are functionally complete.

Any compound proposition can be rewritten as a logically equivalent one that only has the operators AND, NOT

Example: $p \to q$ $\equiv \neg p \vee q \equiv \neg(p \wedge \neg q)$

Circuit?

Functionally complete set of connectives *Rosen p. 35 #42-53*

Claim: The connectives AND, NOT are functionally complete.

Any compound proposition can be rewritten as a logically equivalent one that only has the operators AND, NOT

Example: $p \leftrightarrow q$

Circuit?

Functionally complete set of connectives *Rosen p. 35 #42-53*

Claim: The connectives AND, NOT are functionally complete.

Any compound proposition can be rewritten as a logically equivalent one that only has the operators AND, NOT

Example: $p \oplus q$

Functionally complete set of connectives *Rosen p. 35 #42-53*

Rewriting compound propositions using only NOT, AND

1. Work from the inside out …
2. For each connective, replace it with an equivalent form that uses only NOT, AND:
   - If the connective is NOT or AND, do nothing.
   - If the connective is OR: replace $p \vee q$ with $\neg(\neg p \wedge \neg q)$
   - If the connective is IF..THEN: replace $p \rightarrow q$ with … $\neg(p \wedge \neg q)$
   - If the connective is IFF: replace $p \leftrightarrow q$ with … $\neg(p \wedge \neg q) \wedge \neg(\neg p \wedge q)$
   - If the connective is XOR: replace $p \oplus q$ with … $\neg(\neg(p \wedge \neg q) \wedge \neg(\neg p \wedge q))$

Functionally complete set of connectives *Rosen p. 35 #42-53*

Example: express $A \rightarrow (B \vee C)$ as a logically equivalent compound proposition that only uses ANDs and NOTs.

Functionally complete set of connectives *Rosen p. 35 #42-53*

Example: express $A \to (B \vee C)$ as a logically equivalent compound proposition that only uses ANDs and NOTs.

Use $p \vee q \equiv \neg(\neg p \wedge \neg q)$ to rewrite intermediate step:

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

Functionally complete set of connectives *Rosen p. 35 #42-53*

Example: express $A \to (B \vee C)$ as a logically equivalent compound proposition that only uses ANDs and NOTs.

Use $p \vee q \equiv \neg(\neg p \wedge \neg q)$ to rewrite intermediate step:

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

Use $p \to q \equiv \neg(p \wedge \neg q)$ to rewrite:

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

Simplify double negation and use associativity: $\boxed{\neg(A \wedge \neg B \wedge \neg C)}$

Going backwards

Given compound proposition, use

- Truth tables
- Logical equivalences

to compute truth values.

Reverse?

Given truth table settings, want compound proposition with that output.

E.g. Think back to HW2 Q2

CNF and DNF

Rosen p. 35 #42-53

Conjunctive normal form: AND of ORs (of variables or their negations).

Disjunctive normal form: OR of ANDs (of variables or their negations).

Which of the following is in CNF?

A. $(p \vee q)$ CNF, DNF

B. ~~$\neg(p \vee q)$~~ NOT

C. $(\neg p \vee q) \wedge (p \vee \neg q)$

D. ~~$(p \wedge q) \vee (\neg p \wedge \neg q)$~~ DNF

E. More than one of the above.

Edge case: A can be interpreted as an

- AND (of itself), and as an
- OR (of itself)

Reverse-engineering

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

Reverse-engineering

Approach 1: classify rows based on one variable

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

$$p \wedge (r \rightarrow q)$$

$$\neg p \wedge \neg q \wedge r$$

Reverse-engineering

Approach 2: algorithmically convert to normal form

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

Reverse-engineering

Approach 2: algorithmically convert to normal form

DNF: when is output T?

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

LAND IN THESE ROWS!

Reverse-engineering

Approach 2: algorithmically convert to normal form

DNF: when is output T?

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

Reverse-engineering

Approach 2: algorithmically convert to normal form

DNF: when is output T?

p	q	r	?
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

is T is T is T

$p \wedge q \wedge r$

r is F

$p \wedge q \wedge \neg r$

$p \wedge \neg q \wedge \neg r$

$\neg p \wedge \neg q \wedge r$

Reverse-engineering

Approach 2: algorithmically convert to normal form

DNF: when is output T?

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

$p \wedge q \wedge r$

$p \wedge q \wedge \neg r$

$p \wedge \neg q \wedge \neg r$

$\neg p \wedge \neg q \wedge r$

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

Reverse-engineering

Approach 2: algorithmically convert to normal form

DNF: when is output T?

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

Reverse-engineering

Approach 2: algorithmically convert to normal form

CNF: when is output F?

p	*q*	*r*	*?*
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

p not T ∨ q not F
∨ r not T

AVOID THESE ROWS!

Reverse-engineering

Approach 2: algorithmically convert to normal form

CNF: when is output F?

p	q	r	?
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

AVOID THESE ROWS!

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

Reverse-engineering

Approach 2: algorithmically convert to normal form

CNF: when is output F?

p	q	r	?
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

$\neg p \vee q \vee \neg r$

$p \vee \neg q \vee \neg r$

$p \vee \neg q \vee r$

$p \vee q \vee r$

Reverse-engineering

Approach 2: algorithmically convert to normal form

CNF: when is output F?

p	q	r	?
T	T	T	T
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	T
F	F	F	F

$\neg p \vee q \vee \neg r$

$p \vee \neg q \vee \neg r$

$p \vee \neg q \vee r$

$p \vee q \vee r$

CNF

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

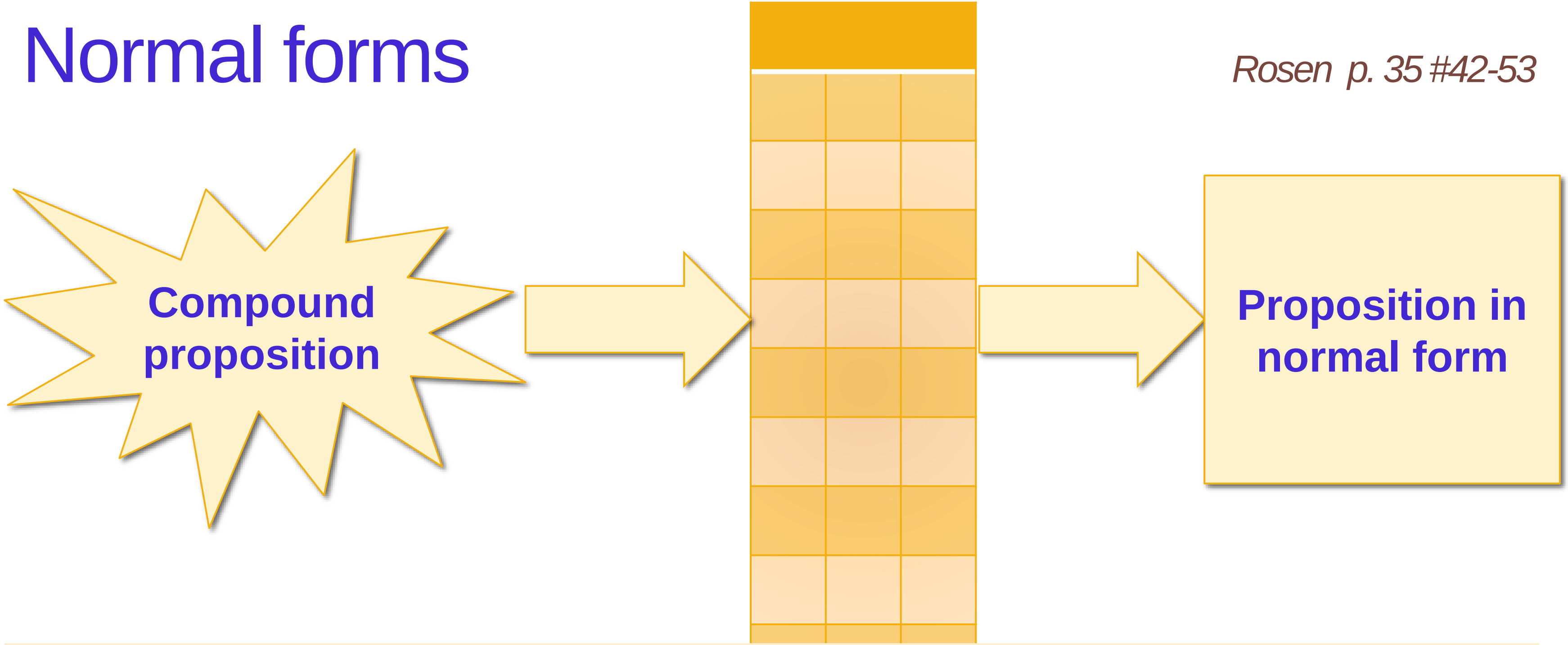
Payoff

- Any output column of a truth table (assignment of T/F to each combination of T/F input values) can be realized as a compound proposition.

- The collection $\vee \quad \wedge \quad \neg$ is functionally complete.

Normal forms

Rosen p. 35 #42-53



Added benefit: If want to reduce connectives further to prove a new collection of connectives is functionally complete, only need to consider those used in normal form.