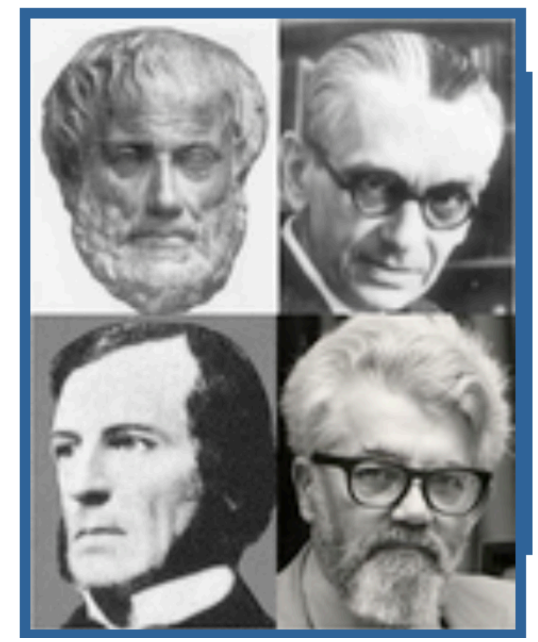
Propositional Logic

Pete Manolios
Northeastern

Logic and Computation, 1/28/2019

Short History

- Aristotle (384-322 B.C.) invented syllogistic logic
  - His “Organon” books were the foundations of logic for 2000 years.
  - Kant: *Logic ... since Aristotle … has been unable to advance a step, and thus to all appearance has reached its completion.*
  - Karl von Prantl: logicians who say anything new about logic are *“confused, stupid or perverse.”*
- George Boole (1815-1864) introduced Boolean algebra
  - algebraic notation for reasoning about sets
  - based on a symbolic language
- Godel’s incompleteness theorem (1931)
  - Considered by many the deepest result of the 20th century
- McCarthy (1961)
  - Instead of debugging a program, one should prove that it meets its specifications, and this proof should be checked by a computer program.



Propositional Logic

- Also called Boolean Logic
- What is the simplest, non-trivial domain (set of objects)?
  - Empty set, set with one object: trivial
  - Set with two objects {T, F} or {0, 1} or {true, false} or {t, nil} or …
  - Functions over such set are non-trivial
- Amazingly rich domain
  - Logic, AI, SAT, scheduling, circuit design, game theory, verification, reliability theory, security, etc.
- Example from safety analysis highlighting: modeling, formula simplification, probability analysis
  - See https://github.com/pmanolios/safety-analysis
  - Look at Webpage
  - run make examples/777-….out; emacs examples/777-….out
  - Show output regarding importance metrics

Expressions

- The expressions of propositional logic include:
  - The constant expressions true and false: they always evaluate to T and F
  - The propositional atoms: e.g., p, q, and r, which range over values T and F
  - Expressions are also called formulas
- Propositional expressions can be combined with propositional operators
  - $\neg$: negation, e.g., $\neg p$
  - $\wedge$: conjunction, e.g., $p \wedge q$
  - $\vee$: disjunction, e.g., $p \vee q$
  - $\Rightarrow$: implication, e.g., $p \Rightarrow q$
  - $\equiv$: equivalence, e.g., $p \equiv q$
  - $\oplus$: exclusive or (xor), e.g, $p \oplus q$

Truth Tables

- The meaning of operators is defined using truth tables
  - Show what the operator returns for every possible input
  - Feasible to do this because there are a finite number of inputs
  - Each row corresponds to an assignment

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

English Usage

- In English usage, “or” often means p or q, but not both
  - You can have ice cream or a cookie
  - In logic “or” (v) always means at least one (truth table!)
  - If you want to say exactly one, use “xor” ($\oplus$)
- True or false?
  - If Trump is 95 years old, then there are only dragons in this room
  - If Trump is president, then there are only dragons in this room
  - Logically, only the first is true, but many English speakers will say that an implication is false if there is no connection between the antecedent and consequent
- Logical (or material) implication: $p \Rightarrow q$: p is the antecedent and q the consequent
  - True or False? If x is a natural number, x >= 0
  - True. What if x is -1, isn't that a counterexample?
  - Think of this way $p \Rightarrow q$ is: if p then q else T
  - Important: the only way to falsify $p \Rightarrow q$ is to set p to T and q to F (truth table!)

Precedence

- To avoid using too many parentheses, from now on we will follow the convention: ¬ binds tightest, followed by {∧, ∨}, followed by ⇒, followed by {⊕, ≡}.
- Instead of ((p ∨ (¬q)) ⇒ r) ⊕ ((¬r) ⇒ (q ∧ (¬p))) we can write
  - p ∨ ¬q ⇒ r ⊕ ¬r ⇒ q ∧ ¬p
- A ternary operator: *ite*
  - Note: equivalent to `if` on Booleans

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

Proofs by Truth Table

- A simple way to prove that an expression is valid (or true) is to use truth tables
- To prove $\neg p \vee q \equiv p \Rightarrow q$, identify atom and construct column for each subexpression using truth table for connectives
- Valid iff every entry in final column is T

p	q	$\neg p$	$\neg p \vee q$	$p \Rightarrow q$	$\neg p \vee q \equiv p \Rightarrow q$
T	T	F	T	T	T
T	F	F	F	F	T
F	T	T	T	T	T
F	F	T	T	T	T

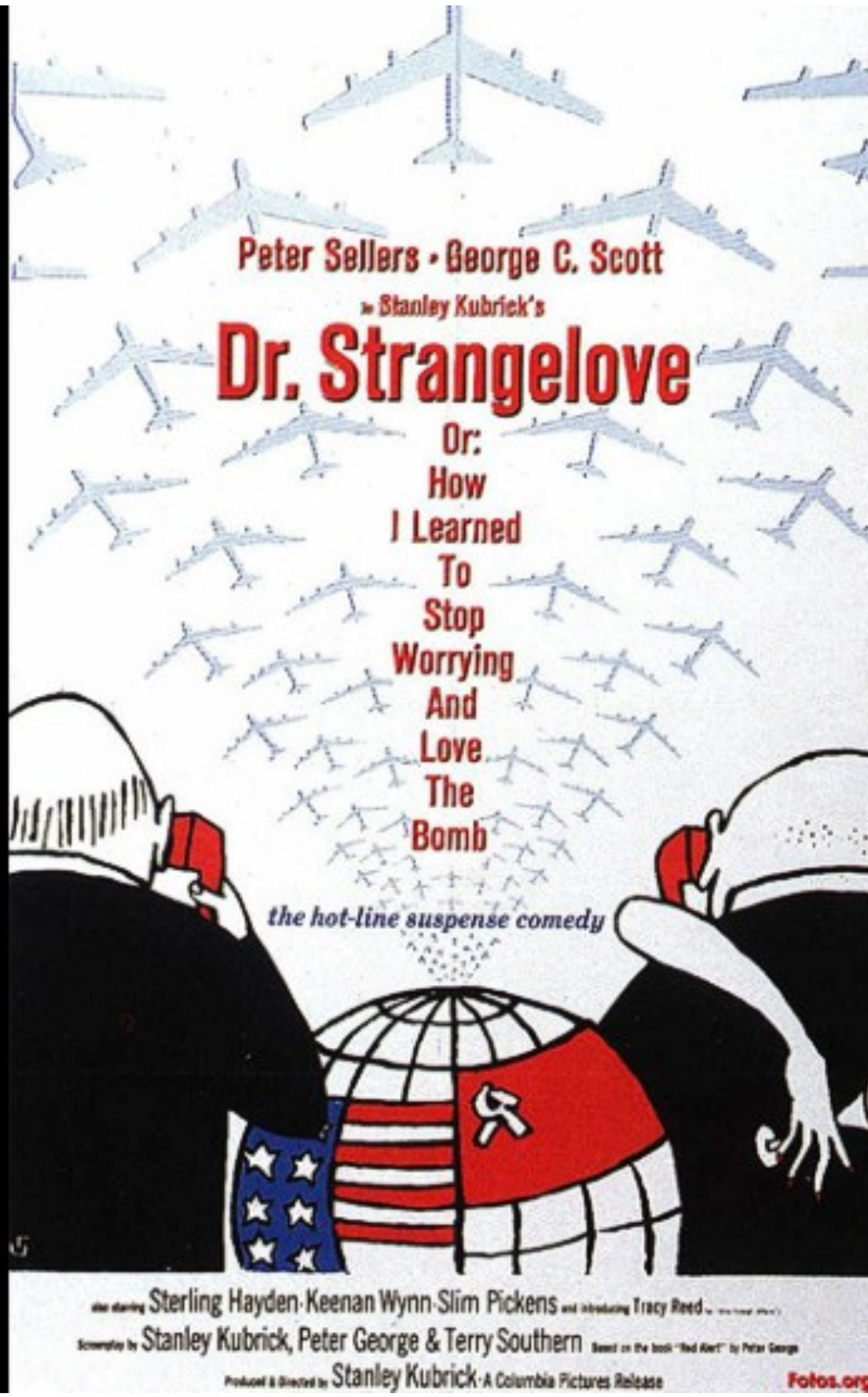
Characterization of Formulas

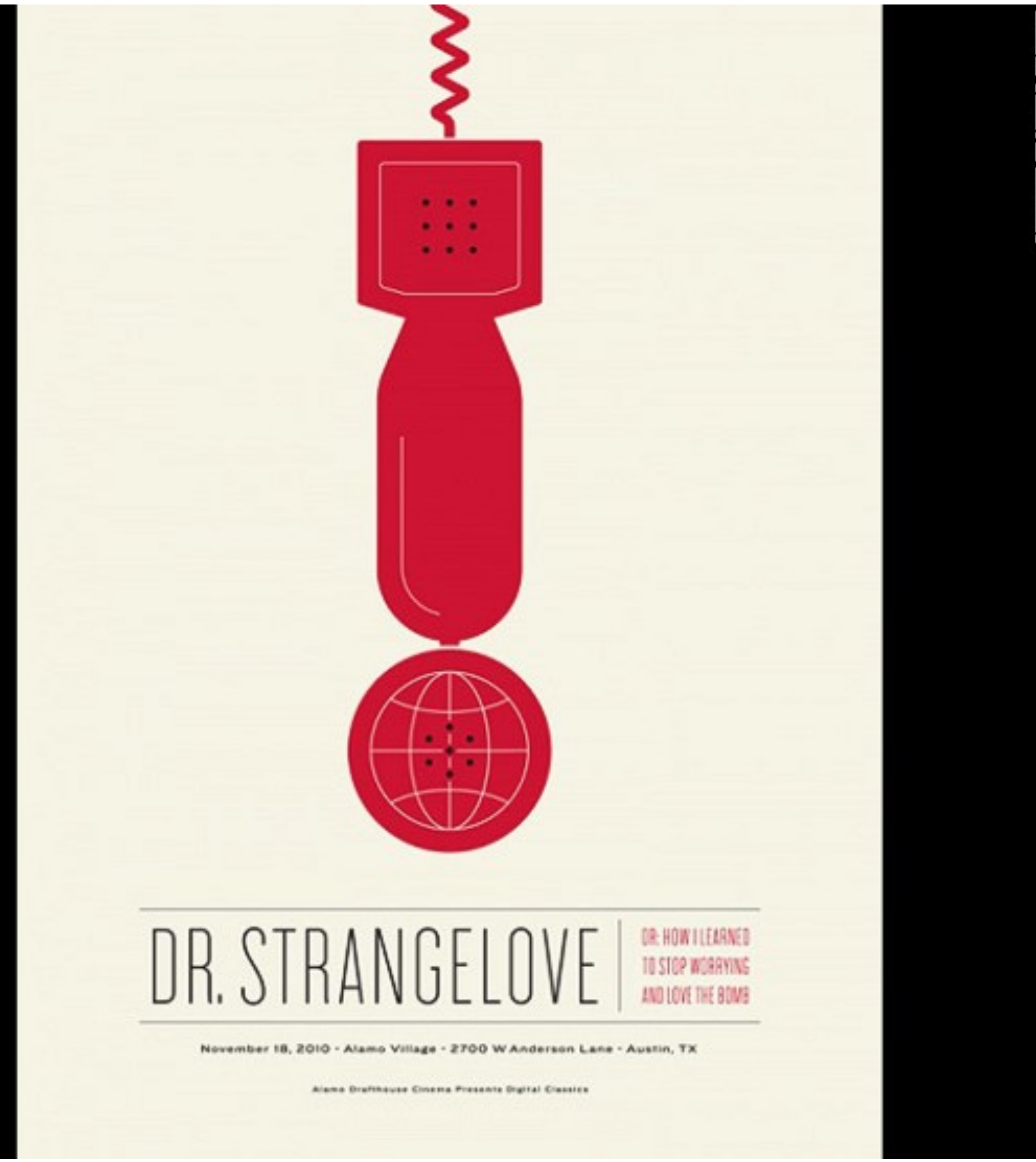
- Consider the truth table of the formula, then
- satisfiable if there is at least one T
  - e.g., true, p, $p \wedge q$, $p \vee q$, …
- unsatisfiable if it is not satisfiable, i.e., all entries are F
  - e.g., false, $p \wedge \neg p$
- falsifiable if there is at least one F
  - e.g., false, p, $p \wedge q$, $p \vee q$, …
- valid if it is not falsifiable, i.e., all entries are T
  - e.g., $p \vee \neg p$, $\neg p \vee q \equiv p \Rightarrow q$, …
- If a formula is valid then it is satisfiable
- If a formula is unsatisfiable then it is falsifiable
- Every formula satisfies exactly two of the above characterizations

The Power of Xor

- You have probably seen movies with the “red telephone” that connects the Pentagon with the Kremlin
- A classic is Dr. Strangelove

The Red Phone





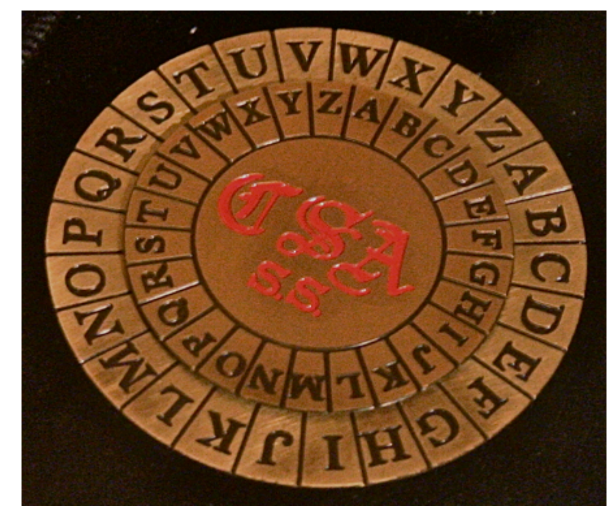
Slides by Pete Manolios for CS2800, Logic & Computation, NU 2019

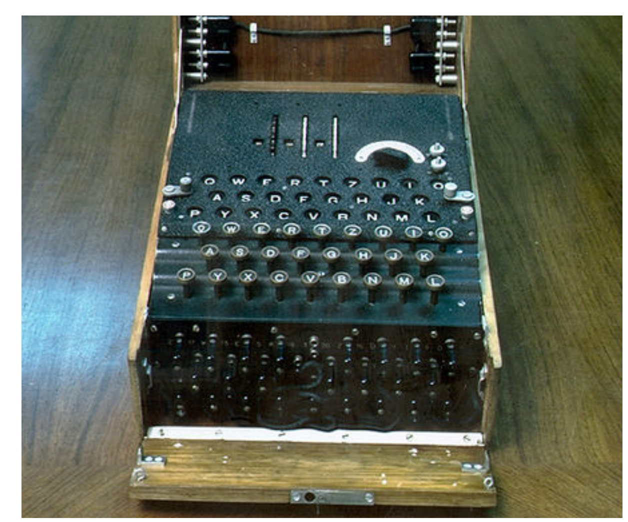
The Power of Xor

- You have probably seen movies with the “red telephone” that connects the Pentagon with the Kremlin
- A classic is Dr. Strangelove
- View https://www.youtube.com/watch?v=VEB-OoUrNuk to 1:24
- There was no red phone but there was a teletype-based encryption mechanism in place between the US and USSR that used the encryption method we will cover next

Cryptography

- Goal: secret communication
  - crypto, graphy are Greek for hidden, writing
- Date back to Egypt (1900 BCE)
- Used for commerce, war, love letters, religion, …
- Examples
  - Scytale: Archilochus 7th century BC
  - Caesar Shift Cipher: shift letters by some number
  - Confederate Cipher Disc: Civil War
  - Enigma: used by Germany in WWII
  - Breaking Enigma shortened the war (Turing et al)





Slides by Pete Manolios for CS2800, Logic & Computation, NU 2019

Exercise

- You got the following encrypted message. Decrypt it.
  - Uif tfdsfu pshbojabujpo nffut upojhiu
- Quiz: A. I got it! B: This is hard!
- Frequency analysis: the most common letters are e, t
  - u: 6
  - f: 5
- Hint: Caesar Shift Cipher: shift letters by some number
  - Shift by 16, 1
- Answer? The secret organization meets tonight

One-Time Pad

- Allow us to encrypt messages with "perfect" secrecy
  - If an adversary intercepts an encoded message, they gain no information, except for an upper bound on the message length
  - Compare: RSA can be broken, with enough computational resources
- A message is a sequence of bits, say 0's & 1's. Any ideas?
- Alice and Bob agree on a secret, a sequence of random 0's & 1's
- To send message m, Alice xor's m with s, the secret: $c = m \oplus s$
- When Bob gets c, he xor's it with s: $c \oplus s = m$
- Example: m=1001000100011…

  s=1101011010111…

  c=0100011110100…

  $c \oplus s$=1001000100011…

One-Time Pad

- Allow us to encrypt messages with "perfect" secrecy
  - If an adversary intercepts an encoded message, they gain no information, except for an upper bound on the message length
- To send message m, Alice xor's m with s, the secret: $c = m \oplus s$
- When Bob gets c, he xor's it with s: $c \oplus s = m$
- Why is it "perfect"?
  - If we have c, the encrypted msg, then for every, m, an arbitrary msg of the same length, there is some secret, s, that when used to decode c yields m
- Example: c=0100011110100… (the same c as before)

  m=0110111011100… (a different m)

  s=0010100101000… (the corresponding s)

  $c \oplus s$=0110111011100…

Next Time

- Propositional Logic
- See reading material