Review

- Complexity classes: **P**, **NP**, **PSPACE**, **EXP**
- Example problems: 3-COLORING, EULERIAN, HAMILTONIAN, EC, VC, CLIQUE, IS, ILP
- Nondeterministic WHILE+ program: "guess x_i"

Boolean formula – Syntax

F is...	A formula is...
• 0 or 1	• FALSE or TRUE
• x_i	• a variable
• $\neg F$	• negation (NOT) of another formula
• $(F \wedge F)$	• conjunction (AND) of two formulas
• $(F \vee F)$	• disjunction (OR) of two formulas

Length of formula: number of times to apply these

Boolean formula – Syntax

Formula or not?

- 10
- $\neg(x_1 \wedge \neg 1)$
- $\big((x_1 \vee x_2) \vee \neg x_1\big)$
- $a \vee b \vee \neg a$

Things that are okay:

- Rename variables
- Remove excess parentheses
  - Outer-most parentheses
  - $\big({\color{red}(}F_1 \vee F_2{\color{red})} \vee F_3\big)$ and similar

Boolean formula – Semantics

- Formula without variables is either FALSE or TRUE
- $0 = \text{FALSE}, 1 = \text{TRUE}$
- $\neg F$: the other value from F
- $F_1 \wedge F_2$: TRUE iff both F_1, F_2 are TRUE
- $F_1 \vee F_2$: FALSE iff both F_1, F_2 are FALSE

- Formula with variables: need to replace (substitute) variables first

Boolean formula – Examples

- $(1 \wedge \neg 0) \vee (1 \wedge \neg 1)$
- $(0 \wedge \neg 0) \vee (1 \wedge \neg 1)$
- $(1 \wedge \neg 0) \vee (0 \wedge \neg 0)$
- $(a \wedge \neg 0) \vee (1 \wedge \neg 1)$
- $(1 \wedge \neg 0) \vee (b \wedge \neg b)$
- $(1 \vee 0) \wedge 0$
- $1 \vee (0 \wedge 0)$

Boolean formula – Equivalent formulas

- Formulas that give the same evaluation regardless of assignments
- $(a \land \neg 0) \lor (1 \land \neg 1) \equiv a$
- $(1 \land \neg 0) \lor (b \land \neg b) \equiv 1$
- $(a \land \neg 0) \lor (b \land \neg b) \equiv a$

- Truth table
- Laws

Boolean formula – Laws

- Associativity
  - $(a \vee b) \vee c \equiv a \vee (b \vee c)$
  - $(a \wedge b) \wedge c \equiv a \wedge (b \wedge c)$
- Commutativity
  - $a \vee b \equiv b \vee a$
  - $a \wedge b \equiv b \wedge a$
- Distributivity
  - $(a \vee b) \wedge c \equiv (a \wedge c) \vee (b \wedge c)$
  - $(a \wedge b) \vee c \equiv (a \vee c) \wedge (b \vee c)$
- Idempotence
  - $a \vee a \equiv a$
  - $a \wedge a \equiv a$
- Absorption
  - $(a \vee b) \wedge a \equiv a$
  - $(a \wedge b) \vee a \equiv a$
- Identity
  - $a \vee 0 \equiv a$
  - $a \wedge 1 \equiv a$
- Annihilator
  - $a \vee 1 \equiv 1$
  - $a \wedge 0 \equiv 0$
- Negation
  - $\neg 0 \equiv 1$
  - $\neg 1 \equiv 0$
- Complementation
  - $a \vee \neg a \equiv 1$
  - $a \wedge \neg a \equiv 0$
- Duality
  - $\neg\neg a \equiv a$
- De Morgan's laws
  - $\neg(a \vee b) \equiv \neg a \wedge \neg b$
  - $\neg(a \wedge b) \equiv \neg a \vee \neg b$

EVAL

- Given Boolean formula and assignments to variables, is the result true?
- e.g. "$(a \land \neg 0) \lor (b \land \neg b)$ with $a = \text{TRUE}$, $b = \text{FALSE}$"
- In **P**
    - Reproduce the derivation tree
    - Evaluate recursively

SAT

- Given Boolean formula, is it satisfiable (has assignment to make it true)?
- e.g. "$(a \wedge \neg 0) \vee (b \wedge \neg b)$" and "$(a \wedge 0) \vee (b \wedge \neg b)$"
- In **NP**
  - Witness: assignment
    - First one: $a = \text{TRUE}$, $b = \text{FALSE}$
    - Second one: not satisfiable
  - Because EVAL in **P**

Conjunctive normal form

- $c_1 \wedge c_2 \wedge \cdots \wedge c_m$ where c_i is a CNF clause
- CNF clause: $(p_1 \vee p_2 \vee \cdots \vee p_k)$ where $p_i = 0, 1, x_j, \neg x_j$ called literal
- Length: sum of k's

- $(a \vee \neg b) \wedge (b \vee c \vee 0) \wedge (a \vee \neg c)$
- $a \vee b$
- $a \wedge b$
- $(a \wedge b) \vee c$

Converting to equivalent CNF

- $(a \land b) \lor c$
- $(a \lor c) \land (b \lor c)$

- Laws: distributivity, double negation, De Morgan

- Push all $\neg$ in
  - De Morgan, double negation
  - Negation to resolve constants
- Structural induction
  - Constant, variable: leave as is
  - $F \land F$: leave as is
  - $F \lor F$: distributivity
  - $\neg F$: only happens with variable, so leave as is

Disjunctive normal form

- $c_1 \vee c_2 \vee \cdots \vee c_m$ where c_i is a DNF clause
- DNF clause: $(p_1 \wedge p_2 \wedge \cdots \wedge p_k)$ where $p_i = 0, 1, x_j, \neg x_j$

- $(a \wedge \neg b) \vee (b \wedge c \wedge 0) \vee (a \wedge \neg c)$

SAT, revisited

- k-SAT: Formula is in CNF, each clause has at most k literals
- Notable cases: 2-SAT, 3-SAT

- If formula is in DNF, SAT is in **P**
  - Only need one clause true
  - Check if clause can be satisfied: make all literals true (only way to do it)
  - If can't (has $x \land \neg x$), continue to next clause

Why don't we just convert to DNF?

- CNF $(p_1 \lor q_1) \land (p_2 \lor q_2) \land \cdots \land (p_k \lor q_k)$ has length $2k$
- Equivalent DNF is
$$(p_1 \land p_2 \land \cdots \land p_k) \lor (p_1 \land p_2 \land \cdots \land q_k) \lor \cdots \lor (q_1 \land q_2 \land \cdots \land q_k)$$
- All 2^k possible ways to take one literal from each CNF clause
- Has length $k \cdot 2^k$

Recap

- EVAL (evaluation): given formula and assignments, is it true?
- SAT (satisfiability): given formula, can it be true?
- k-SAT: formula is in k-CNF

- 3-SAT is "the hardest" problem in **NP**: coming soon!
- 2-SAT is in **P**

2-SAT is in **P** – Krom (1967)

- Preprocessing: remove constants and duplicate clauses
- n variables, up to $4n^2$ clauses

- If there are $(a \vee b)$ and $(\neg b \vee c)$, we can generate $(a \vee c)$
- If there is $(a \vee a)$, then a must be true
- Idea: From φ, generate extra clauses over and over to get φ'
- Consistent: φ' doesn't have $(a \vee a)$ and $(\neg a \vee \neg a)$
- φ satisfiable if and only if φ' consistent

2-SAT is in **P** – Satisfiable ⇒ Consistent

- Assignment for φ works for φ'
- Induct on number of generations
- 0: $\varphi' = \varphi$
- k to $k+1$: only thing that matters is $(a \vee b) \wedge (\neg b \vee c) \rightarrow (a \vee c)$
- $(a \vee a) \wedge (\neg a \vee \neg a)$ not satisfiable
- Not consistent ⇒ not satisfiable

2-SAT is in **P** – Consistent $\Rightarrow$ Satisfiable

- Free variable a: no $(a \vee a)$ or $(\neg a \vee \neg a)$ in φ'
- Induct on number of free variables
- 0: all variables set; $(a \vee a)$ means a true, $(\neg a \vee \neg a)$ means a false
- k to $k + 1$: suppose a free, we claim adding $(a \vee a)$ is consistent

2-SAT is in **P** – Consistent ⇒ Satisfiable

Claim: Suppose a free, we claim adding $(a \vee a)$ is consistent

- What clauses got added?
  - $(a \vee a) \wedge (\neg a \vee b) \rightarrow (a \vee b)$
  - $(a \vee b) \wedge (\neg b \vee c) \rightarrow (a \vee c)$? Already made
  - $(b \vee a) \wedge (\neg a \vee c) \rightarrow (b \vee c)$
  - $(b \vee c) \wedge (\neg c \vee d) \rightarrow (b \vee d)$? Already made
- Only two kinds of new clauses
  - Let S be set of all x where $(\neg a \vee x)$ exists
  - New clauses: $(a \vee x)$ and $(x \vee y)$ for all $x, y \in S$
- In particular: $(x \vee x)$ for all $x \in S$

2-SAT is in $\mathbf{P}$ – Consistent $\Rightarrow$ Satisfiable

Claim: Suppose a free, we claim adding $(a \lor a)$ is consistent

- Suppose not consistent: has $(b \lor b) \land (\neg b \lor \neg b)$
- Claim: a wasn't free, contradicting assumption
- **Case 1:** $b = a$
  - $\neg a \in S$ so $(\neg a \lor x) = (\neg a \lor \neg a)$ already existed
- **Case 2:** $(b \lor b)$ and $(\neg b \lor \neg b)$ new
  - $b, \neg b \in S$ so $(\neg a \lor b) \land (\neg b \lor \neg a) \rightarrow (\neg a \lor \neg a)$ existed
- **Case 3:** $(b \lor b)$ new, $(\neg b \lor \neg b)$ not new
  - $b \in S$ so $(\neg a \lor b)$ existed
  - Then $(\neg a \lor \neg b)$ and $(\neg a \lor \neg a)$ existed

2-SAT is in **P** – Algorithm

- Proved: φ satisfiable if and only if φ' consistent
- Generate over and over
  - Find a clause to generate: $(4n^2)^2 = \mathcal{O}(n^4)$
  - At most $4n^2 = \mathcal{O}(n^2)$ new clauses
  - Check consistency: $(4n^2)^2 = \mathcal{O}(n^4)$
  - Total running time: $\mathcal{O}(n^4) \times \mathcal{O}(n^2) + \mathcal{O}(n^4) = \mathcal{O}(n^6)$ is polynomial time
- Can be improved to $\mathcal{O}(n^2)$
- Even, Itai, Shamir (1976): linear time by limited backtracking
- Aspvall, Plass, Tarjan (1979): linear time by strongly connected components of implication graph

More decision problems

- TAUTOLOGY: given formula, is it always true?
  - Complement in **NP**
  - If formula in CNF, in **P**
- EQUIV: given two formulas, are they equivalent?
  - Complement in **NP**
- SHORTER: given formula, is there an equivalent shorter formula?
  - Not clear! But in **PSPACE**
- LONGER: given formula, is there an equivalent longer formula?
  - In **P**

More satisfiability problems

- 1-IN-3-SAT: given 3-CNF, is there assignment so exactly one literal from each clause is true?
  - In **NP**
- ODD-3-SAT: given 3-CNF, is there assignment so an odd number of literals from each clause is true?
  - In **P**
- MAJ-SAT: given formula, is there a majority of the assignments that make it true?
  - Not clear! But in **PSPACE**
- #SAT: given formula, how many assignments make it true?
- MAX-SAT: given CNF, how many clauses at most can be satisfied?
  - Function problems, not decision problems
  - Hard, even for 2-CNF

HORN-SAT

- Horn clause: at most one positive literal (e.g. a, $\neg a$, $(a \vee \neg b)$, ...)
  - Implication: "if all negated variables are true, the positive is true"
- HORN-SAT: Given CNF where each clause is a Horn clause, is it satisfiable?
- In **P**
  - Clause of single literal l: set l true, remove all clauses with l, remove $\neg l$ from their clauses
  - If nothing else to remove: clauses have ≥ 2 literals, so has a negated variable; set all variables false
  - Unsatisfiable iff l and $\neg l$ happen