1 P and NP

1.1 Intro

Language = problem
String = instance of the problem

Decidable = a problem is decidable, if there is an algorithm for it
Algorithm = a TM that halts on all inputs

Recursive languages: set of decidable problems
Time complexity $T(n)$ ("to have a running time of $T(n)$"): given a TM M, and an input w, M halts after making at most $T(n)$ moves
Problems solvable in Polynomial time (P):

- is $w \in L(G)$?
- Path from x to y in graph $G = (V, E)$
- ... (and many more)

Problems solvable in Nondeterministic Polynomial time (NP):

- Knapsack problem
- Graph coloring
- Traveling salesman problem
- ... (it's a long list!)

1.2 Boolean expressions

- Variables (0, 1)
- Operators ($\wedge, \vee, \neg$)

$x \wedge \neg(y \vee z)$
$x \wedge -(y + z)$
Satisfiability problem (SAT): give a truth assignment that satisfies a BE.
Cooke: SAT is NP-complete
CSAT: given a boolean expression in CNF, is it satisfyable?

1.3 Normal forms

- Literal: variable or its negation ($x, \neg y$)
- Clause: OR / AND of two or more literals
- Conjunctive normal form (CNF): AND of clauses with OR-ed literals
- $(x \vee \neg y) \wedge (\neg x \vee z) \rightarrow (x + \bar{y})(\bar{x} + z)$
- k-CNF: each clause has exactly k literals

k-CNF: A CNF, where each clause has exactly k distinct literals.
k-SAT: Satisfiyability problem of a k-CNF. A k-CNF is NP-complete when $k \geq 3$, but the 2-CNF is polynomially solvable.

1.4 Conversion from BE to CNF

There are two main ways to do it:

1. Use the reduction algorithm from SAT to CSAT.

   (a) Push $\neg$ below $\vee, \wedge$

      i. $\neg(E \wedge F) \rightarrow \neg E \vee \neg F$

      ii. $\neg(E \vee F) \rightarrow \neg E \wedge \neg F$

      iii. $\neg(\neg E) \rightarrow E$

   (b) Write the expression as a product of clauses by introducing new variables.

2. Use a truth table to find falsifying assignments.

1.5 Independent Set and Vertex Cover

Indepentend Set (IS): Let $G = (V, E)$ be and undirected graph. We say that $I \subset V$ is an independent set, if no two nodes of I are connected by any edge of E. An IS is *maximal*, if you cannot find a larger IS for the same graph.

Vertex Cover (VS) (alternatively: Node Cover): Let $G = (V, E)$ be and undirected graph. We say that $I \subset V$ is a vertex cover, if each edge $e \in E$ has at least one of its endpoints in I. A VC is *minimal*, if you cannot find a VC with fewer nodes for the same graph.