- Defined the following terms: variable, literal, positive literal, negative literal, truth assignment, truth value of a literal, and boolean formula.
- A boolean formula $\mathcal{B}$ is satisfiable whenever there exists a truth assignment from its variables to $\{T, F\}$ so that substituting corresponding truth values to literals in $\mathcal{B}$ yields T. Otherwise, $\mathcal{B}$ is not satisfiable.
- The disjunction of a set of literals is a clause. A boolean formula which is a conjuction of collection of clauses is said to be in conjunctive normal form (CNF). Further, if each clause in the boolean formula in CNF has exactly two literals, then that boolean formula is in 2-CNF.

- example 2-CNF formula: $(\neg x \vee y) \wedge (\neg y \vee z) \wedge (x \vee \neg z) \wedge (z \vee y)$

- 2-SAT problem: Given a 2-CNF formula ϕ, decide whether it is satisfiable. If it is satisfiable, output a truth assignment that satisfies ϕ.
- Constructing ϕ' from ϕ: Noting that for $\alpha \vee \beta$ to be true, if α is F then β must be T and if β is F then α must be T, corresponding to each clause $(\alpha \vee \beta)$ in ϕ, introduce two implications $(\neg\alpha \rightarrow \beta)$ and $(\neg\beta \rightarrow \alpha)$ into a boolean formula ϕ'.
- Observation: ϕ is satisfiable iff ϕ' is satisfiable. Significantly, a satisfiable truth assignment for ϕ satisfies ϕ', and vice versa.
- Constructing $G(\phi')$ from ϕ': For each variable α in ϕ', introduce one vertex for α and one vertex for $\neg\alpha$ in $G(\phi)$. Further, introduce an edge (α, β) in $G(\phi')$ iff the implication $\alpha \rightarrow \beta$ is in ϕ'.
- Observation: (α, β) is an edge in $G(\phi)$ iff $(\neg\beta, \neg\alpha)$ is an edge in $G(\phi)$. —— (1)
- If $G(\phi)$ contains a path from α to β, it also contains a path from $\neg\beta$ to $\neg\alpha$. —— (2)
- A 2-CNF formula ϕ is not satisfiable iff there exists a variable x such that there is a path from x to $\neg x$ and there is a path from $\neg x$ to x in $G(\phi)$.

  "$\Leftarrow$" (proof by contradiction)

  for $T(x) = T$[1] (and $T(\neg x) = F$), there exists two neighboring nodes α, β along path P' from x to $\neg x$ such that $T(\alpha) = T$, $T(\beta) = F$, and $\alpha \rightarrow \beta$; and, the truth value of clause $\neg\alpha \vee \beta$ corresponding to $\alpha \rightarrow \beta$ is F

  for $T(x) = F$ (and $T(\neg x) = T$), there exists two neighboring nodes α, β along P'' from $\neg x$ to x such that $T(\alpha) = T$ and $T(\beta) = F$; and, the truth value of clause $\neg\alpha \vee \beta$ corresponding to $\alpha \rightarrow \beta$ is F

  "$\Rightarrow$" (proving the contrapositive)

  – suppose that in $G(\phi)$ there is no variable such that there is a path from x and $\neg x$ and there is a path from $\neg x$ to x

[1] $T()$ is a truth assignment from the set of variables to $\{T, F\}$

- repeatedly do the following to obtain a truth assignment $T()$:

  pick a node α (not necessiarly a positive literal) whose truth value has not yet been defined and there is no path from α to $\neg\alpha$ —— (3)

  for every node β reachable from α (including α), assign $T(\beta) = T$ and $T(\neg\beta) = F$ (the latter correspond to all those nodes from which $\neg\alpha$ is reachable)

- proof showing $T()$ is a satisfiable truth assignment for ϕ:

* whenever a node is assigned T, all the nodes that are reachable from it are also assigned T, and hence there can be no arc along any of these paths which goes from a node assigned to T to a node assigned to F

  analogously, whenever a node is assigned F, all the nodes from which it is reachable are also assigned F, and hence there can be no edge from T to F: if β is reachable from α then $\neg\alpha$ is reachable from $\neg\beta$; considering (3), $\neg\beta$ is not reachable from α; hence, by assigning $T(\neg\beta) = T(\neg\alpha) = F$, no implication along $\neg\beta$ to $\neg\alpha$ is false

* further, since it is not possible for both a path from x to $\neg x$ 'as well as' a path from $\neg x$ to x to exist, every literal α is assigned a truth value that is consistent to $\neg\alpha$ and vice versa

- Algorithm 1:

  1. construct $G(\phi)$ from input formula ϕ
  2. compute SCCs of $G(\phi)$
  3. find a topological order of nodes of SCC-graph of these SCCs
  4. associate ids to SCCs in increasing order of their occurance along that topological order

     for every SCC CC, associate id i to each node $v \in CC$ whereas i is the SCC id of CC
  5. for every literal α whose truth value is not yet set
     (i) if the SCC ids of both α and $\neg\alpha$ are equal, then print "ϕ is not satisfiable" and return
     (ii) if $\neg\alpha$ is not reachable from α, then for every node β reachable from α (this includes α), assign $T(\beta) = T$; for every node β from which we can reach $\neg\alpha$ (this includes $\neg\alpha$), set $T(\beta) = F$
     (iii) else (i.e., α is not reachable from $\neg\alpha$), for every node β reachable from $\neg\alpha$ (including $\neg\alpha$), assign $T(\beta) = T$; for every node β from which we can reach α (including α), set $T(\beta) = F$ (here, β is not necessarily a positive literal)
  6. print "ϕ is satisfiable" and output the truth assignment due to $T()$

  Proof of correctness: Immediate from the proof of necessary and sufficient conditions for satisfiability.

  Time complexity: Homework.

- Algorithm 2 (simplified) -

  1. construct $G(\phi)$ from input formula ϕ

2. compute SCCs of $G(\phi)$
3. find a topological order of nodes of SCC-graph of these SCCs
4. associate ids to SCCs in increasing order of their occurance along that topological order
   for every SCC CC, associate id i to each node $v \in CC$ whereas i is the SCC id of CC
5. for every positive literal α
   (i) if the SCC ids of both α and $\neg\alpha$ are equal, then print "ϕ is not satisfiable" and return
   (ii) if the SCC id of α is larger than the SCC id of $\neg\alpha$ (i.e., $\neg\alpha$ is not reachable from α) then
   set, $T(\alpha) = T$ and $T(\neg\alpha) = F$
   (iii) else (i.e., if the SCC id of $\neg\alpha$ is larger than the SCC id of α)
   set, $T(\alpha) = F$ and $T(\neg\alpha) = T$
6. print "ϕ is satisfiable" and output the truth assignment due to $T()$

Proof of correctness - given Algorithm 1 is correct, prove Algorithm 2 is correct - Homework.

Time complexity: Homework.

— A Monte Carlo Algorithm for 2-SAT —

- Consider the following simple randomized algorithm:

  (1) initialize variables with an arbitrary truth assignment

  (2) repeat the following for t times

  (a) if all clauses are satisfied then return the truth assignment

  (b) pick an *arbitrary* clause C that is yet to be satisfied

  (c) choose a literal $\ell \in C$ *uniformly at random*
  change the truth value of ℓ

  (3) if all clauses are satisfied return the truth assignment else output the input formula is not satisfiable

  It is immediate that if ϕ is not satisfiable, this algorithm's ouptut is guaranteed to be correct. Also, if t is a polynomial in the number of clauses in ϕ, then this algorithm takes polynomial time. Assume ϕ is satisfiable. Fix on any one satisfying truth assignment S for variables in ϕ. Let X_i be the number of variables whose truth assignment agrees with S after i^{th}-iteration. It is clear that the algorithm terminates with a satisfying truth assignment when $X_i = n$ (in the worst case[2]).

  Suppose the clause C and the literal ℓ from C are chosen in $(i+1)^{st}$-iteration. Since C is a clause that is not satisfied, at least one of the literals in C is not agreeing with its truth value in S. If ℓ is not agreeing with its truth assignment in S, flipping the truth value of ℓ in this iteration leads to X_{i+1} being equal to $X_i + 1$; otherwise, flipping the truth value of ℓ leads to X_{i+1} being equal to $X_i - 1$. However, considering the possibility in which the truth values of both the variables do not agree with their respective truth values assigned via S, $pr(X_{i+1} = j+1 | X_i = j) \geq \frac{1}{2}$ and $pr(X_{i+1} = j-1 | X_i = j) \leq \frac{1}{2}$.

  Consider the following random walk of a particle P on the positive x-axis: for $X_i = j$, P is at $(j, 0)$; if $X_{i+1} = j - 1$, then P moves to $(j-1, 0)$; if X_{i+1}, then P moves to $(j+1, 0)$, etc. The objective is to determine the expected number of iterations for the particle P to reach $(n, 0)$, given the particle moves from $(j, 0)$ to $(j-1, 0)$ with probability $\leq \frac{1}{2}$, and from $(j, 0)$ to $(j+1, 0)$ with probability $\geq \frac{1}{2}$ for any j. Consider another particle Q, wherein Q moves from $(j, 0)$ to $(j-1, 0)$ with probability exactly $\frac{1}{2}$, and from $(j, 0)$ to $(j+1, 0)$ with probability exactly $\frac{1}{2}$. It is obvious the expected number of iterations for P to reach $(n, 0)$ from $(j, 0)$ is less than or equal to the expected number of iteration for Q to reach n from $(j, 0)$ for any j. Since we want to upper bound the expected number of iterations in reaching from $(j, 0)$ to $(n, 0)$, we focus on the expected number of iterations Q takes in moving from $(j, 0)$ to $(n, 0)$ for any j.

  Let Z_j be a random variable denoting the number of iterations to reach $(n, 0)$ from $(j, 0)$ for particle Q. Then, $E[Z_j] = E[\frac{1}{2}(1 + Z_{j-1}) + \frac{1}{2}(1 + Z_{j+1})]$. That is, by applying the linearity of expectations, $E[Z_j] = \frac{1}{2}E[Z_{j-1}] + \frac{1}{2}E[Z_{j+1}] + 1$. Solving this recurrence relation with guess and substitute, by guessing $E[Z_j] = n^2 - j^2$, yields $E[Z_0] = n^2$.

  Let Y be the number of iterations till the algorithm finds a solution. Then, from Markov's inequality, $pr(Y > 2n^2) \leq \frac{E[Y]}{2n^2} = \frac{n^2}{2n^2} = \frac{1}{2}$. That is, if the algorithm runs for $t = 2n^2$ iterations, the probability that it does not find a solution is $\leq \frac{1}{2}$. Hence, by setting $t = 2n^2 m$ in the algorithm, probability the algorithm fails to find a solution is at most $\frac{1}{2^m}$.

[2] since some other truth assignment S' may also be satisfying ϕ, and the algorithm terminates if the variables of ϕ gets assigned to truth values as in S', that is, before their respective truth values are as in S