3-CNF Satisfiability

- A formula is 3-CNF provided the formula is CNF and each clause has exactly 3 distinct literals. For example,

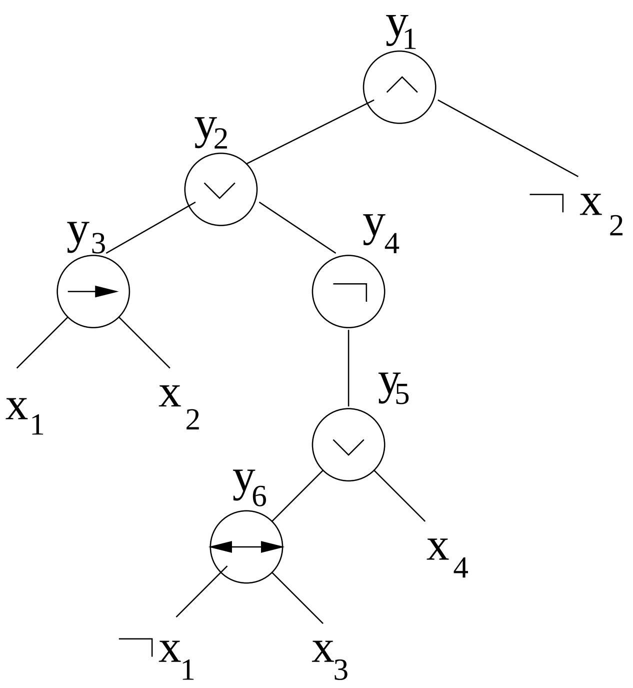
  $$(x_1 \vee \neg x_1 \vee \neg x_2) \wedge (x_3 \vee x_2 \vee x_4) \wedge (\neg x_1 \vee \neg x_1 \neg x_2)$$

- 3-SAT problem is to find a truth assignment for satisfiability of a formula in 3-CNF form.
- 3-SAT is in $\mathcal{NP}$. Just guess the truth assignments for the variables.
- In order to prove that 3-CNF-SAT is $\mathcal{NP}$ complete find a polynomial transformation of SAT to 3-CNF-SAT.
- SAT is in $\mathcal{NP}$.

3-SAT

- Consider the parse tree for:

  $\Phi = ((x_1 \rightarrow x_2) \vee \neg((\neg x_1 \leftrightarrow x_3) \vee x_4)) \wedge \neg x_2$



- If the input formula contains a clause such as OR of several literals then use associativity to parenthesize the expression and let every interal node of parse tree have 1 or 2 children.

3-SAT

- Let y_i be output for an internal node.
- Rewrite the original formula as AND of root variable and conjunction of clauses describing the operation at each node.

$$\begin{aligned}\Phi' = y_1 &\wedge (y_1 \leftrightarrow (y_2 \wedge \neg x_2))\\ &\wedge (y_2 \leftrightarrow (y_3 \wedge y_4))\\ &\wedge (y_3 \leftrightarrow (x_1 \rightarrow x_2))\\ &\wedge (y_4 \leftrightarrow \neg y_5)\\ &\wedge (y_5 \leftrightarrow (y_6 \vee x_4))\\ &\wedge (y_6 \leftrightarrow (\neg x_1 \leftrightarrow x_3))\end{aligned}$$

- Each clause has at most 3 literals.
- Now try to convert the clauses into CNF.

3-SAT

- First covert into DNF by truth table for each clause.
- The reason for the two-step conversion of clauses into DNF and then to CNF is simplicity.
- Consider the truth table for the first clause $C_1^{\Phi'}$.

y_1	y_2	x_2	$(y_1 \leftrightarrow (y_2 \wedge \neg x_2))$
1	1	1	0
1	1	0	1
1	0	1	0
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	1
0	0	0	1

3-SAT

- DNF can be obtained just by ORing of the truth values of literals corresponding to the rows with 0 in the last column.
- However, CNF can not be found directly from truth table.
- But DNF can be converted to CNF by simple application of DeMorgan's law.
- The DNF formula equivalent to $\neg C_1^{\Phi'}$ is

$$(y_1 \wedge y_2 \wedge x_2) \vee (y_1 \wedge \neg y_2 \wedge x_2) \vee$$
$$(y_1 \wedge \neg y_2 \wedge \neg x_2) \vee (\neg y_1 \wedge y_2 \wedge \neg x_2)$$

- So the CNF equivalent of $C_1^{\Phi'}$ is

$$(\neg y_1 \vee \neg y_2 \vee \neg x_2) \wedge (\neg y_1 \vee y_2 \vee \neg x_2) \wedge$$
$$(\neg y_1 \vee y_2 \vee x_2) \wedge (y_1 \vee \neg y_2 \vee x_2)$$

- So Φ' is converted into a conjuction of clauses $C_i^{\Phi'}$ each of which is CNF and has at most 3 literals.

3-SAT

- Let us call this transformed formula as Φ''.
- Final step is to ensure that Φ'' can be transformed to a formula where each clause has exactly 3 distinct literals.
- We use two auxiliary variables p and q.
- For each clause C_i in transformed formula Φ'', we do as follows.
  - If C_i has three literals, then let it be a clause of Φ''.
  - If C_i has 2 literals, $C = (l_1 \vee l_2)$, then replace it by
    $$(l_1 \vee l_2 \vee p) \wedge (l_1 \vee l_2 \vee \neg p)$$
    to fulfill the syntactic requirement of 3 clauses.
  - If C_i has 1 distinct literal l, then replace it by 4 clauses
    $$(l \vee p \vee q) \wedge (l \vee p \vee \neg q) \wedge (l \vee \neg p \vee q) \wedge (l \vee \neg p \vee \neg q)$$

3-SAT

- We can at once see that Φ is satisfied if and only if Φ'' is.
- Now we have to prove one more things: the reduction is computed in p-time.
- Construction of Φ' from Φ introduces at most 1 variable and 1 clause per connective in Φ.
- Construction of Φ'' from Φ' can introduces at most 8 clauses per each clause, this follows since a clause in Φ' can have at most three literals, so the truth table can have at most 8 rows.
- For last transformation involving Φ' and Φ'' we can expect a maximum of 4 clauses replacing a clause.
- So the size of Φ'' is a polynomial of the size of Φ.
- Clearly each construction can be done in p-time.

3-D Matching

- Generalization of classical `marriage problem`.

  Given n unmarried men and n unmarried women, along with a list of all male-female pairs who would be willing to marry one another so that polygamy is avoided and everyone receives an acceptable spouse.

- In 3-D matching we have three sets W, X, and Y corresponding `three` different sexes, and each triple in M corresponds to a 3-way marriage that is acceptable to all three participants.

- The marriage problem is solvable in polynomial time but not 3-D matching

3-D Matching

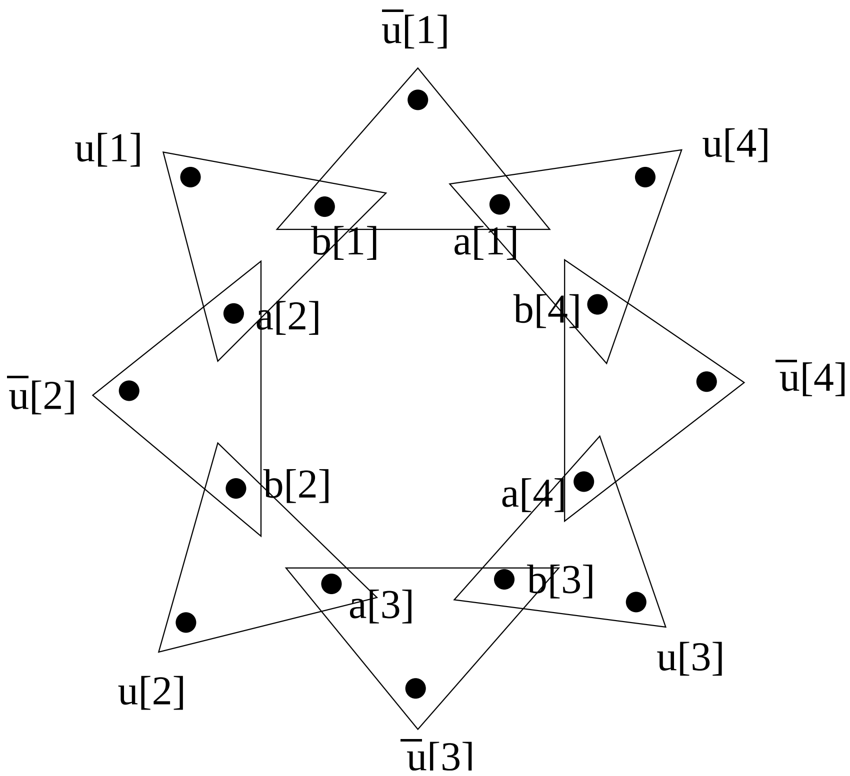
- To show 3-D matching belongs to $\mathcal{NP}$, just get a guess of subset $q = |W| = |X| = |Y|$ of triples from set M.
- The checking stage is to find whether the subset represents an valid marriage: that is if no two of the guessed triples agree in one co-ordinate.
- To show it is also $\mathcal{NP}$ hard try convert 3-SAT to 3-D matching.

3-D Matching

- Let $U = \{u_1, u_2, \ldots, u_n\}$ be the set of variables and $C = \{c_1, c_2, \ldots, c_m\}$ be the set of clauses in an arbitrary instance of 3-SAT.
- Then construct disjoint sets W, X, and Y with same cardinality, and a set $M \subseteq W \times X \times Y$ such that M contains a matching if and only if C is satisfiable.
- The set M of ordered triples will be partitioned into three classes, grouped according to their intended function:
  - `truth setting and fanout`,
  - `satisfication testing`, and
  - `garbage collection`
- Each truth setting and fanout component corresponds to one variable $u \in U$.

3-D Matching

- Its structure depends on the total number m of clauses in C.
- For $m = 4$ the structure is as follows.



3-D Matching

- The truth setting and fanout component will involve two internal elements $a_i[j] \in X$ and $b_i[j] \in Y$, $1 \leq j \leq M$, which will not occur in other triples.
- It will also involve two external elements $u[_i[j], \bar{u}_i[j] \in W$ which may occur in other triples.
- The triples making up this component can be divided into two sets:

$$
\begin{aligned}
T_i^t &= \{(\bar{u}_i[j], a_i[j], b_i[j])|1 \leq j \leq m\} \\
T_i^f &= \{(u_i[j], a_i[j+1], b_i[j])|1 \leq j < m\} \cup \\
&\quad \{u[m], a_i[1], b_i[m])\}
\end{aligned}
$$

- Let $T_i = T_i^f \cup T_i^t$.
- None of the internal elements $\{a_i[j], b_i[j]|1 \leq j \leq m\}$ appear in the triples outside the set of triples T_i.

3-D Matching

- Any matching will have to include exactly m triples from T_i.
- Note that we can not have triples from both T_i^f and T_i^t for a matching.
- So it means the matching M' will include either all triples from T_i^t or from T_i^f.
- This means either all $u_i[j]$ or $\bar{u}_i[j]$ are left uncovered in the matching.
- We can consider this choice as forcing a truth assignment either `true` or `false` to the variable u_i.
- More precisely if $M' \cap T_i = T_i^t$ then it means u_i is to set `true`.

Satisfaction Testing Comp

- Each satisfaction testing component in M corresponds to a single clause $c_j \in C$.
- The set of triples making up this component is defined as

$$C_j = \{(u_i[j], s_1[j], s_2[j]) | u_i \in c_j\} \cup \{(\bar{u}_i[j], s_1[j], s_2[j]) | \bar{u}_i \in c_j\}$$

- Thus any matching $M' \subseteq M$ will have to include only one triple from C_j.
- But the triple can be chosen provided some $u_i[j]$ or $\bar{u}_i[j]$ does not occur in the set of triples $T_i \cap M'$.
- If u_i (or $\bar{u}_i$) was left uncovered among the triples chosen from T_i then due to construction of `truth setting` and `fanout component` it forces u_i to be set `true` (or `false`).
- This means truth setting determined by M' would satisfy the clause c_j.

Garbage Collection Comp

- The construction is completed by specifying a large garbage collection component G.
- It involves
  - internal elements $g_1[k] \in X$ and $g_2[k] \in Y$ for $1 \leq k \leq m(n-1)$, and
  - external elements of the form $u_i[j]$ and $\bar{u}_i[j]$ from W.
- The triples are

$$G = \{(u_i[j], g_1[k], g_2[k]), (\bar{u}_i[j], g_1[k], g_2[k]) | \\ 1 \leq k \leq m(n-1), 1 \leq i \leq n, 1 \leq j \leq m\}$$

Garbage Collection Component

- Each pair $g_1[k]$ and $g_2[k]$ must be matched with a unique $u_i[j]$ or $\bar{u}_i[j]$ that does not occur in any triples of $M' - G$.
- There are exactly $m(n-1)$ such uncovered external elements.
- The structure of G insures that they can always be covered by choosing $M' \cap G$ appropriately.
- Thus G merely guarantees that whenever a subset of $M - G$ satisfies all the constraints then the subset can be extended to a matching for M.

Summarizing 3-D Matching

- We set

$$\begin{aligned} W &= \{u_i[j], \bar{u}_i[j] \mid 1 \leq j \leq m\} \\ X &= A \cup S_1 \cup G_1 \\ Y &= B \cup S_2 \cup G_2 \end{aligned}$$

where

$$\begin{aligned} A &= \{a_i[j] \mid 1 \leq i \leq n, 1 \leq j \leq m\} \\ S_1 &= \{s_1[j] \mid 1 \leq j \leq m\} \\ G_1 &= \{g_1[k] \mid 1 \leq k \leq m(n-1)\} \end{aligned}$$

$$\begin{aligned} B &= \{b_i[j] \mid 1 \leq i \leq n, 1 \leq j \leq m\} \\ S_2 &= \{s_2[j] \mid 1 \leq j \leq m\} \\ G_2 &= \{g_2[k] \mid 1 \leq k \leq m(n-1)\} \end{aligned}$$

and

$$M = \left(\bigcup_{i=1}^{n} T_i\right) \cup \left(\bigcup_{j=1}^{m} C_j\right) \cup G$$

Summarizing 3-D Matching

- Every triple in M is an element of $W \times X \times Y$.
- M contains

$$2mn + 3m + 2m^2n(n-1)$$

triples.

1. Each T_i for $1 \leq i \leq n$ generates $2m$ triples giving $2mn$ triples.
2. Each C_j for $1 \leq j \leq m$ generates 3 triples giving $3m$ triples.
3. G generates $2m^2(n-1)$ for each literal u_i for $1 \leq i \leq n$ giving $2m^2n(n-1)$ triples.

Summarizing 3-D Matching

- Its definition in terms of 3-SAT is obvious.
- M can be constructed in polynomial time.
- M can not be 3-D matching unless C is satisfiable.
- To complete the proof: show that the existence of a truth assignment for satisfying C implies that M contains a matching.

Summarizing 3-D Matching

- Let $t : U \rightarrow \{T, F\}$ be any truth assignment satisfying C.
- Construct the matching $M' \subseteq M$ as follows.
- For each clause $c_j \in C$, let $z_j \in \{u_i, \bar{u}_i | 1 \leq i \leq n\} \cap c_j$ be a literal that is set true by t then we set

$$M' = \left(\bigcup_{t(u_i)=T} T_i^t \right) \cup \left(\bigcup_{t(u_i)=F} T_i^f \right) \cup \left(\bigcup_{j=1}^{m} \{(z_j[j], s_1[j], s_2[j])\} \right) \cup G'$$

where G' is an appropriately chosen subcollection of G, that includes all $g_1[k], g_2[k]$ and the remaining $u_i[j]$ and $\bar{u}_i[j]$.