Construction

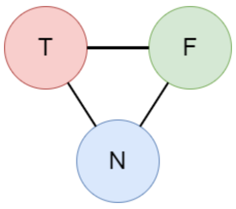
3-satisfiability problems entails an boolean formula in CNF. The objective is to determine if, and what, truth assignments to the literals lets the entire boolean formula to output True, while ensuring consistency of the literals.

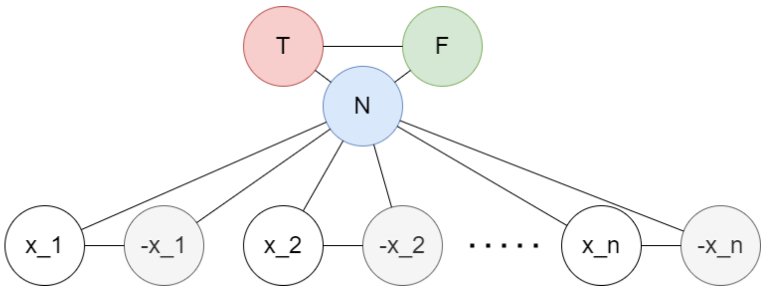
$$(x_1 \vee x_2 \vee x_3) \wedge (x_4 \vee x_5 \vee x_6) \wedge \cdots \wedge (x_r \vee x_s \vee x_t)$$

We can reduce any clause $(x_a \vee x_b \vee x_c)$ in the CNF to a graph 3-colorability problem by using 3-colorability as logic gates. WLOG, we assign Red to True, Green to False, and Blue to Neutral. Neutral is a value with no boolean significance, but an intermediary to allow 3-colorability of a particular structure to behave like logic gates.

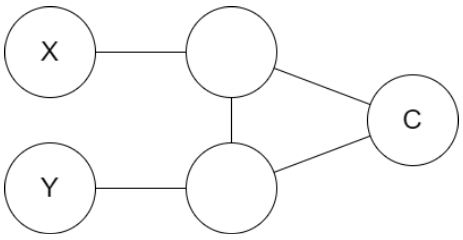
First, we build the control gadget (left), which ensures that the Blue node will indeed be assigned Neutral by enforcing that no adjacent vertices have the same color.

After that, we can build a control structure (right) which will make sure any x_n is assigned either Red (True) or Green (False), and $\neg x_n$ is its logical complement. This construct forces a truth assignment (in colors) for boolean literals in the formula. This step is WLOG because we 1-to-1 mapped the truth value of all available literals to some color.



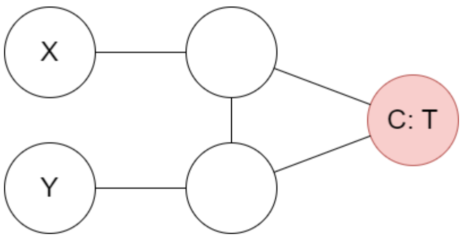


Next, by taking advantage of the constraint that adjacent colors must be disjunct, we can build the following logic gadget that allows logic gate behavior.

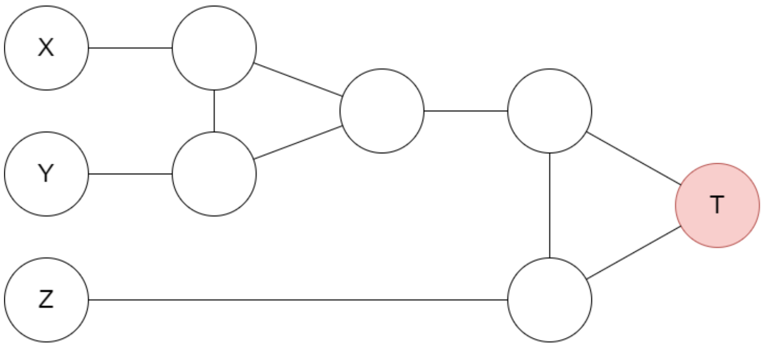


WLOG, say that a clause is made of arbitrary values x, y, z, so we build a 3-colorable graph that sets the clause vertex to Red (True). Since x, y, z are boolean literals (inputs), they must be assigned either Red (True) or Green (False), and they can't be Blue (Neutral).

If we set the the clause vertex C to be Red (True), then the 3-colorability of the entire logic gadget behaves like a logic OR gate (**Proof 1**). Therefore, we refer to the following variant as the OR gagdet. This substructure is only not 3-colorable when both x, y are both Green (False)

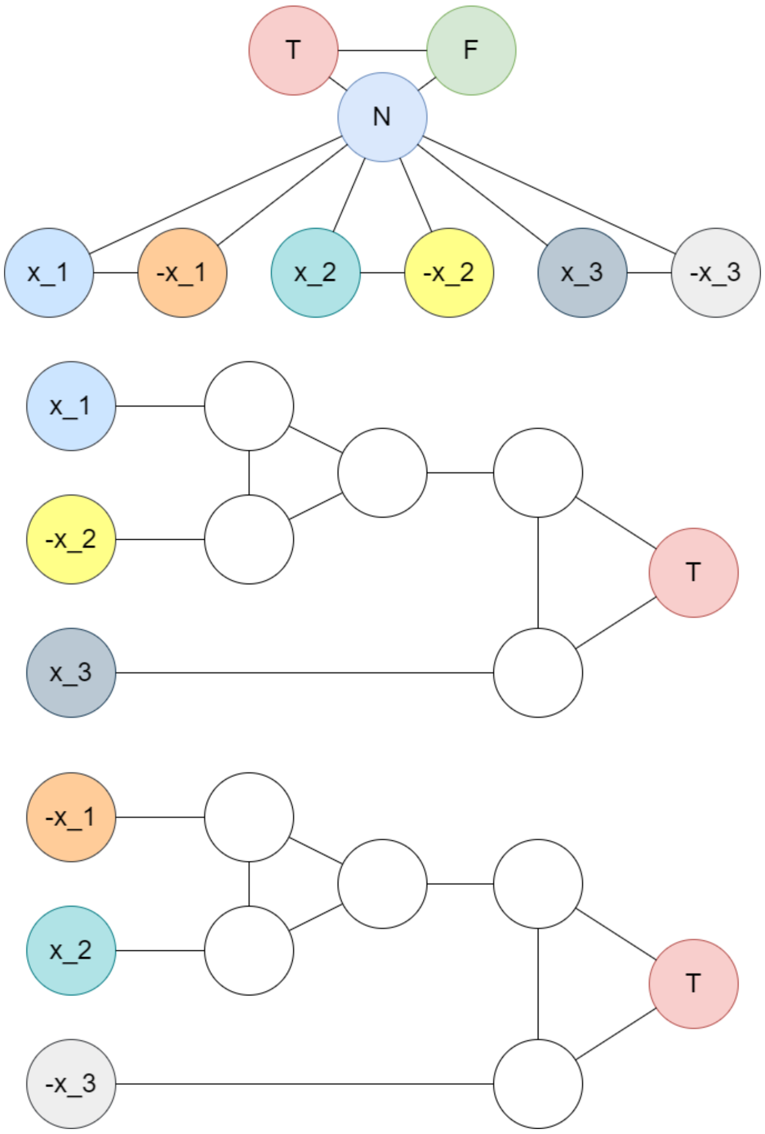


Since that 3-satisfiability have three literals in each clause, $(x_1 \lor x_2 \lor x_3) = ((x_1 \lor x_2) \lor x_3)$, we can now build a clause gadget, which determines the truth value of a clause by its 3-colorability. (**Proof 2**)



To construct the final graph G, we use the control structure we build earlier to supply the value for boolean literals. Then, we'll go through each clause in the original boolean formula and connect them to one of X, Y, Z on the clause gadget.

For example, this is the G for formula $(x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2 \lor \neg x_3)$. Same colored and named nodes are the same nodes.



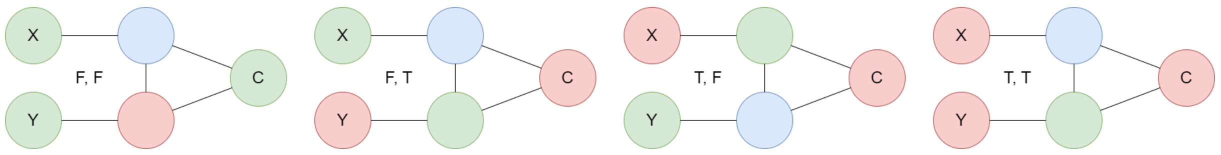
Proof

1. By exhaustion, we can show that the OR gadget is indeed only 3-colorable when either or both of X, Y is Red (True). The process is trivial by plugging in $(F, F), (T, F), (F, T), (T, T)$ for (X, Y). This is a lot to type up for the straightforwardness of this process, so I omit it here at submission.

2. To show that the transformation from 3-satisfiability is correct, we will show that the $(x_1 \lor x_2 \lor x_3) =$ colorable(G), where colorable(G) represents whether the graph G is 3-colorable. Notice that the final structure is a logic gadget nested in the OR variant.

   The original clause means that colorable(G) is False *if and only if* $x_1, x_2, x_3 = F$.

   First, we can show that the generic logic gagdet is always 3-colorable when the vertex C is unspecified for any combination of X, Y.

   As we can observe, the clause vertex C must be equal to X, Y if $X = Y$

   In **Proof 1**, we found that the OR gadget is always 3-colorable when at least one of the input is True. Substituting value of vertex C of the sub-gadget into the OR gadget, we infer that Z must be True for the entire clause gadget to be 3-colorable if $X, Y = F$. Otherwise, when either or both of X, Y is True, the entire clause gadget is 3-colorable irrespective of whether Z is True because the smaller logic gadget is 3-colorable as shown, with the vertex $C = T$.

   At this point, we've shown that the graph G is 3-colorable *if and only if* the clause is true, that is, when any of X, Y, Z is True. By construction and exhaustively explaining why they would have the same truth table.