CSE491/596 Lecture Wed. Nov. 3: Graph Coloring and DomSet Reductions

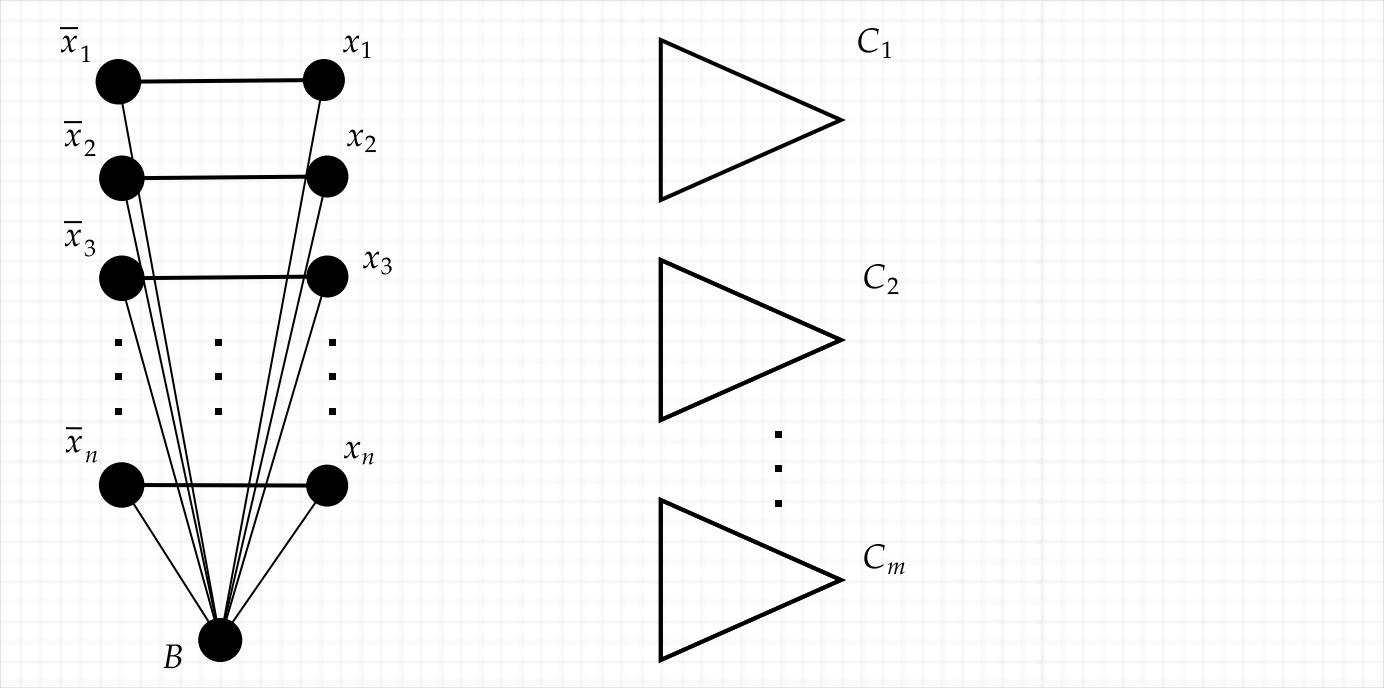
Since $IND\ SET \leq_m^p CLIQUE$ and $IND\ SET \leq_m^p VERTEX\ COVER$ these problems (which we showed to be in NP) are also NP-complete. Now we consider the Graph 3-Coloring Problem.

G3C

Instance: Just an undirected graph $G = (V, E)$ (no k).
Question: Is there a map $\chi : V \to \{R, G, B\}$ such that for all $(u, v) \in E, \chi(u) \neq \chi(v)$?

The Greek chi for "chromo-" meaning "color". The language of 3-colorable graphs is clearly in NP: we just guess the coloring, which is a string in $\{R, G, B\}^n$, and verify the coloring on each of $m \leq \binom{n}{2} = O(n^2)$ edges. To show it is NP-complete, we use the same basic rungs-and-gadgets layout, but with one or two twists.

The first thing to think about is how to establish a correspondence between colorings and truth assignments to begin with, before thinking about "good" colorings (i.e., those that meet the "such that" property of having no monochrome edges) *vis-à-vis* satisfying assignments. The natural idea is to give each rung an edge so that each x_i and $\overline{x}_i$ pair must be given different colors so that one color stands for true and the other for false. Well, we have to limit that to two colors for each rung, so we do so by connecting *all* $2n$ rung nodes to a special node called B for the intent to color it blue. So on the ladder side, we have:

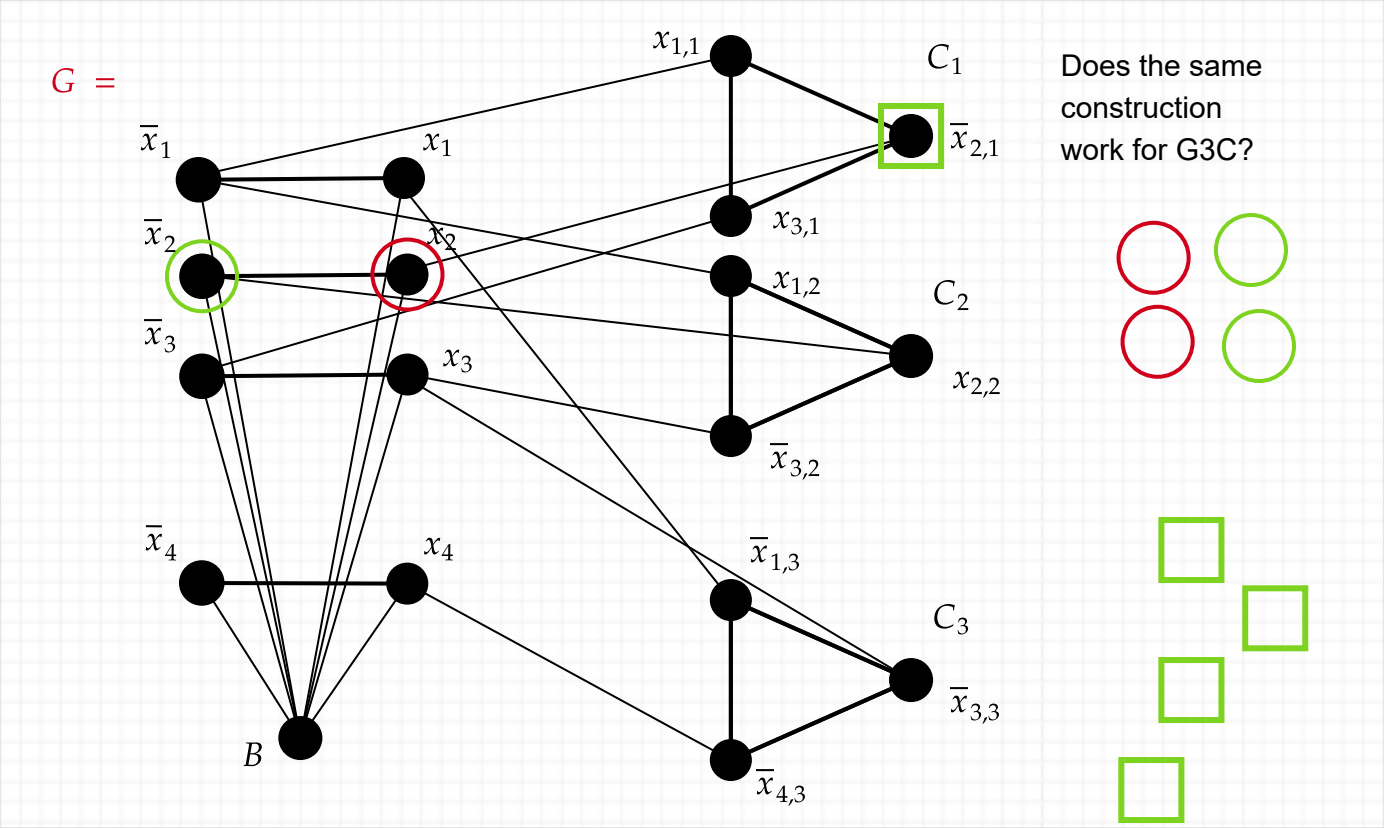


This forces each rung to use one R and one G. Now incidentally, $\phi(B) = B$ is not something the reduction is able to define---it is not part of G. But any good coloring remains good under any of the 6

permutations of the colors, so it is "wlog." that we presume $\phi(B) = B$. This leaves R and G for the rung nodes. We can make G stand for the literals that are made true, R for false, or vice-versa---and the presence of "vice-versa" is something to watch. The permutation that swaps R and G while keeping B fixed stays good, but if flipping an assignment a like 1010 to 0101 satisfies ϕ one way but not the other, there could be a mismatch on correctness requirements. Let's take R to mean true (as the ALR notes do, but a change from last year's notes).

The next question is, can we re-use the clauses-as-triangles idea? With the same crossing edges? Let's try it for the same example formula:

$$\phi = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee \overline{x}_3 \vee \overline{x}_4)$$

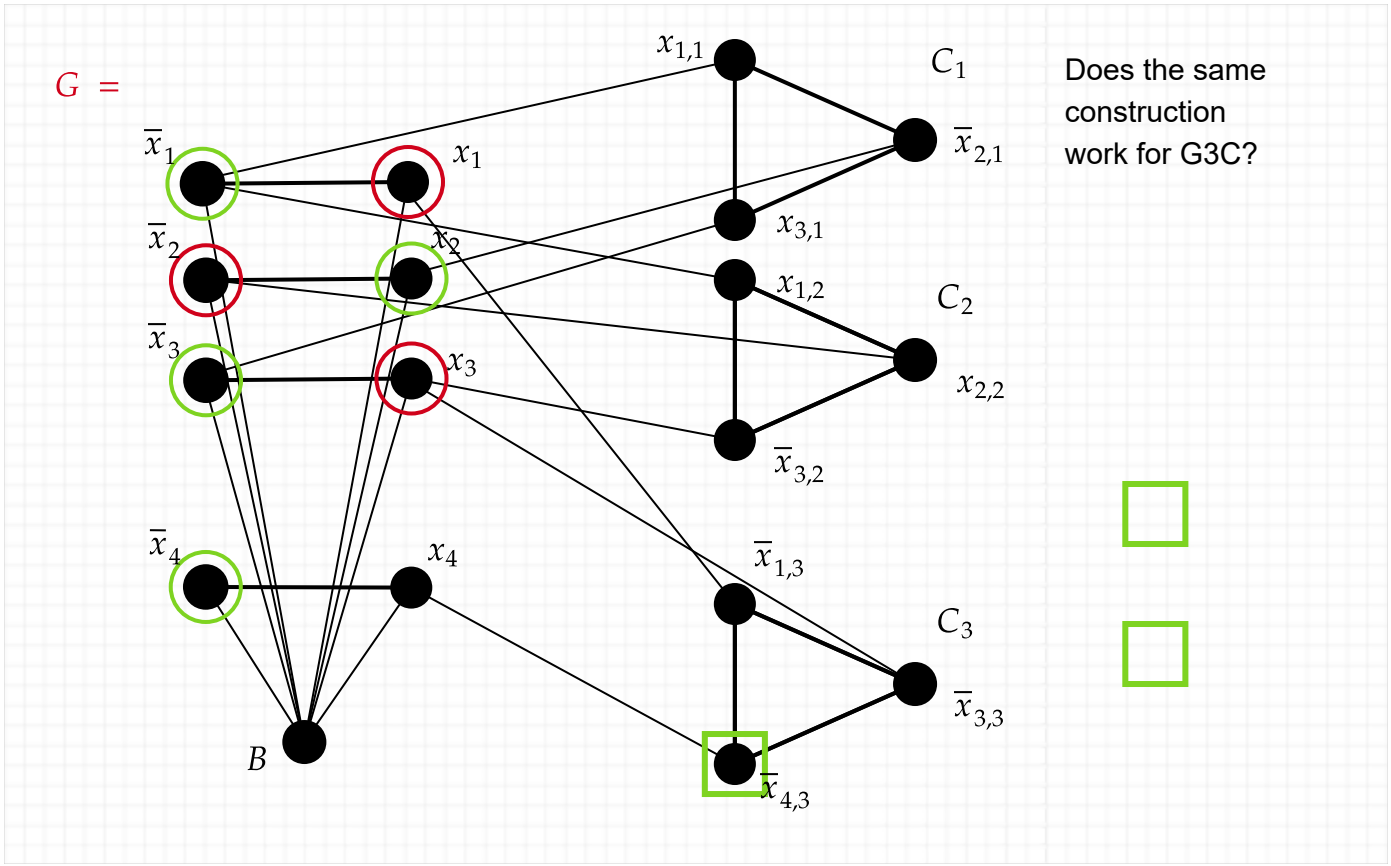


Here's the deal: If we have a 3-coloring χ, it has to use G once in each clause triangle and once in each rung. If x_{ij} is green in clause C_j then its crossing edge goes to $\overline{x}_i$ in rung i. This had to be green,
so x_i in the rung is red. This means x_i was set true, so C_j is satisfied. The reasoning for a negative literal $\overline{x}_{ij}$ in C_j being red is symmetrical: the crossing edge goes to x_i in the rung, which must be green, so x_i is set false, so $\overline{x}_i$ satisfies C_j. Therefore we get the ($\Longleftarrow$) direction that G being 3-colorable implies ϕ is satisfiable.

The $\Longrightarrow$ direction hits a possible snag, however: Suppose ϕ is satisfiable, but only by assignments that make all three literals in some clause true. It's not just that we can't color all three nodes in the clause green, it's that their crossing edges go to red nodes in the rungs. Suppose this happens for

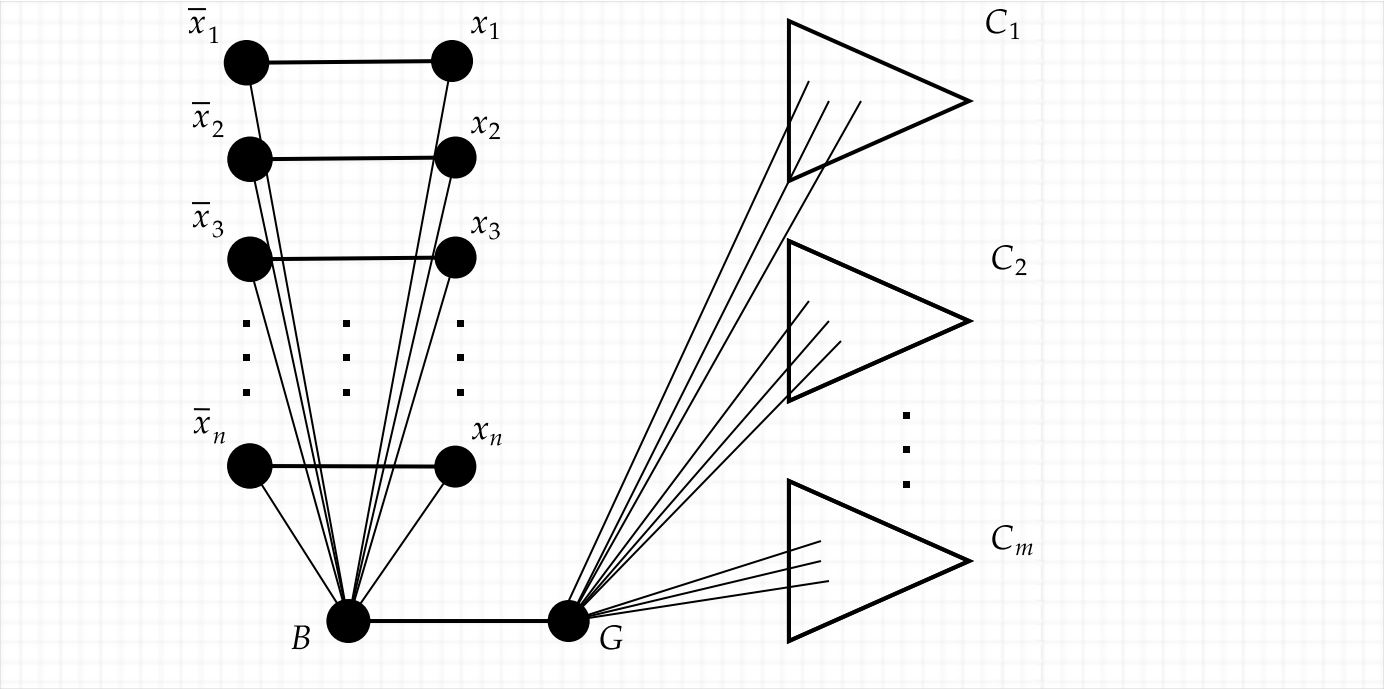
clause C_1 in our example:

$$\phi = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee \overline{x}_3 \vee \overline{x}_4)$$

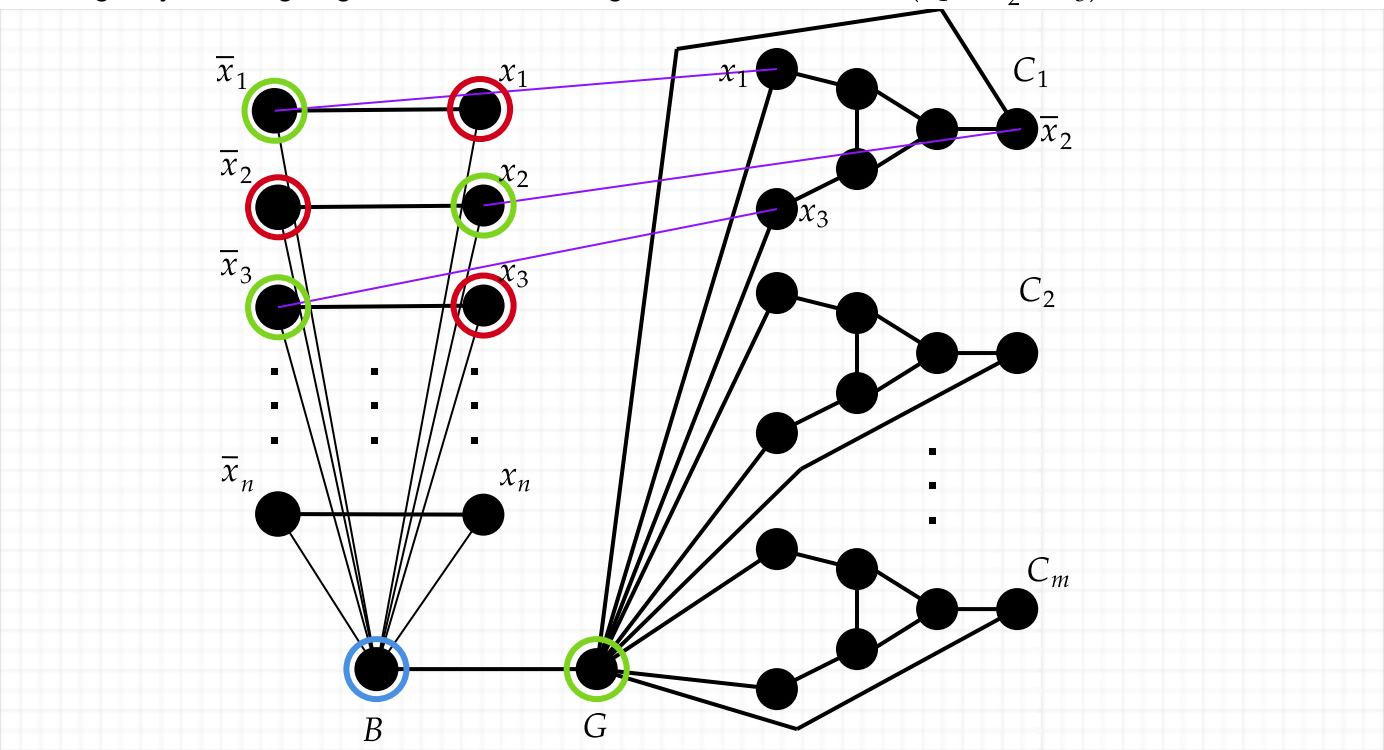


Now the clause C_1 is "greenlocked": we can't color any of its nodes green, so we cannot color it. [Note: The lecture moved things around to illustrate various properties. I think I've placed those things back in a state that matches the surrounding words, but I may have missed some.]

To fix this, we need to expand the clause gadgets. We add to the B node a second node G so that the colors used for those nodes wlog. count as "blue" and "green". Connections from the G node to the clause gadgets can fix the problem of symmetry betweed "red" and "green", which we need to do for reduction from **3SAT**.

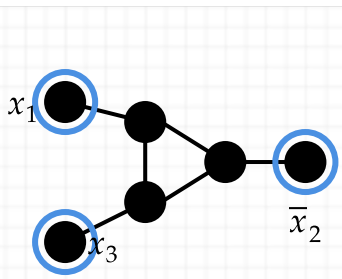


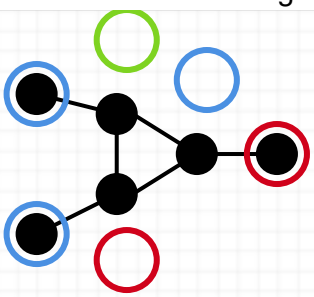
The second change is to include an outer layer of 3 nodes in the clause gadgets. These nodes will get an automatic "greenlock" from the G node. If they get a "redlock" from the rungs---which we want to mean all three literals being made false---then the 3 nodes are forced to be blue. This is without connecting the outer 3 nodes to each other. The resulting "bluelock", however, will prevent an inner triangle of each clause from being 3-colored. If, however, all three literals in the clause are made true, then the outer layer will see "greenlock" twice, and that is no problem. Here is the idea abstractly, showing only crossing edges between the rungs and the first clause $(x_1 \vee \overline{x}_2 \vee x_3)$:

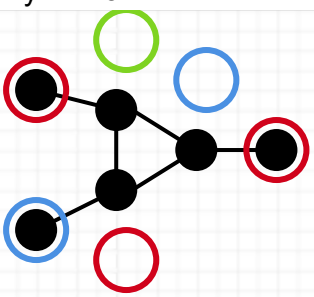


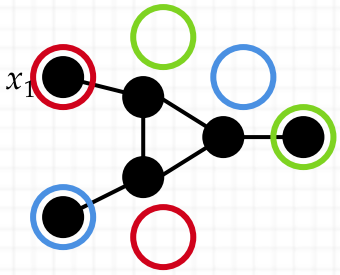
(Note: These are back to being the same as the connections from the ALR notes, connecting x_1 in a clause to $\overline{x}_1$ in the rung to stay consistent with the reduction to **IND. SET**.)

If x_1 and x_3 are made false and x_2 true, so that clause C_1 fails, then each of the outer nodes of the C_1 gadget "sees red" as well as green from node G. This forces each outer node to be blue, but then the inner triangle of the C_1 gadget cannot be 3-colored. Any other assignment, however, allows using two different colors for the outer nodes, and then the inner triangle can always be 3-colored:







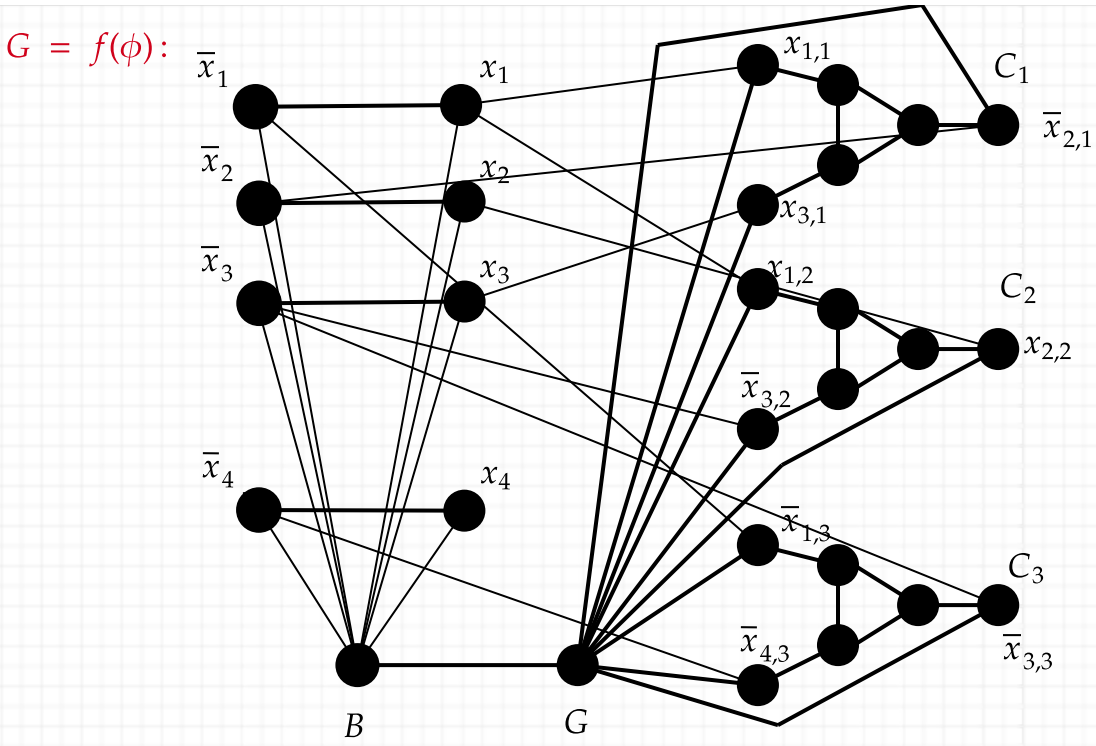


Not possible owing to the G node, but maybe still worth noting.

(In the actual 11/3/21 lecture, I did this diddling with colors on the previous diagram itself.)

Here is the whole reduction carried out for our example formula

$\phi = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (x_1 \vee x_2 \vee \overline{x}_3) \wedge (\overline{x}_1 \vee \overline{x}_3 \vee \overline{x}_4)$

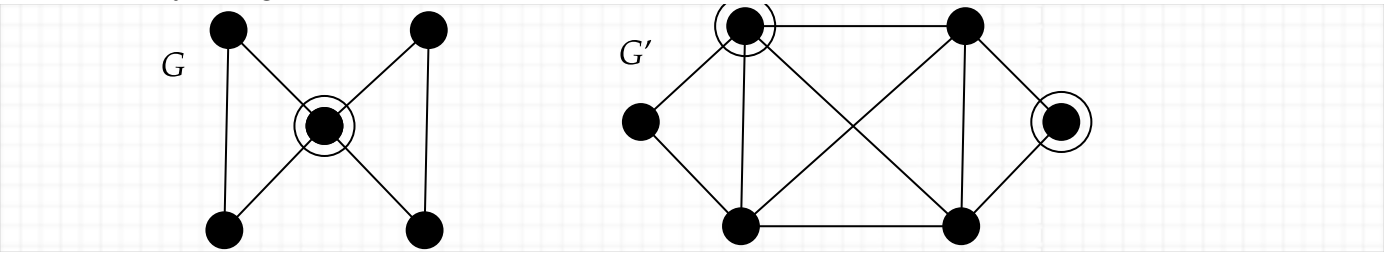


Again the reduction is linear-time computable in one sweep through ϕ. Correctness still needs to be re-checked in the other direction: If G has a good 3-coloring, then for every clause C_j, at least one of its 3 outer nodes x_{ij} or $\overline{x}_{ij}$ must be colored R. Since we are now sending crossing edges to the rung node with the *same* sign, this means the same-sign rung node must be colored G. In turn, this means the literal satisfies C_j. Thus any good coloring uniquely yields an assignment that satisfies all clauses. ☒

Now we will consider a different reduction where both the "rungs" and the "clause gadgets" get different treatment. The target problem is:

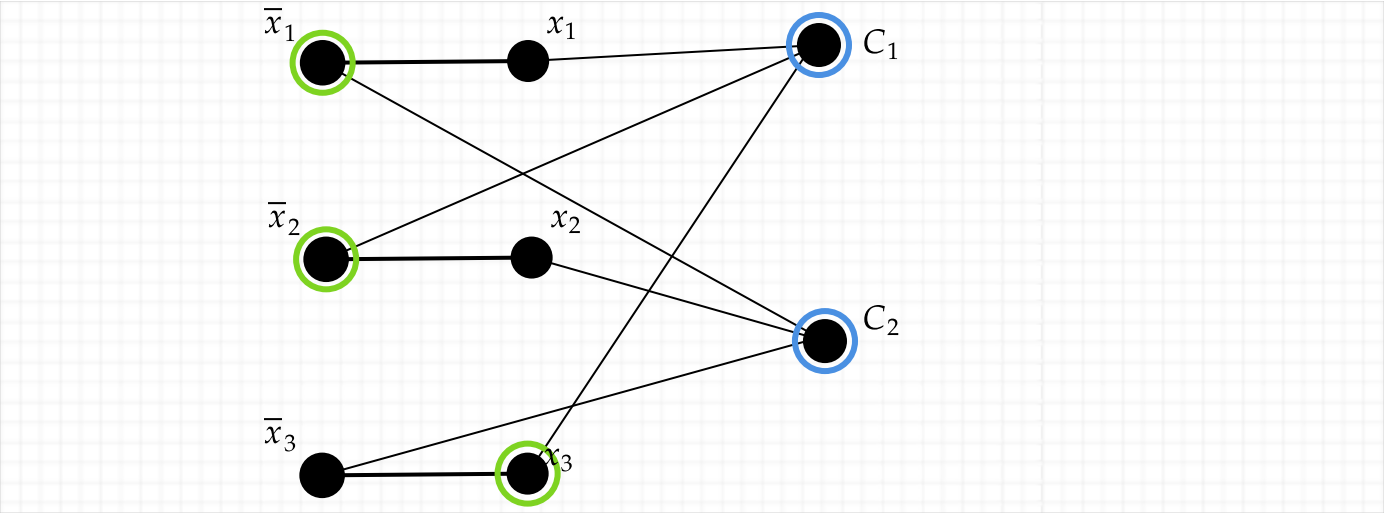
Dominating Set (**Dom Set**)
Instance: An undirected graph $G = (V, E)$ and an integer $k \geq 1$.
Question: Is there $S \subseteq V$, $|S| \leq k$, such that every node **not** in S is adjacent to a node in S?

The difference between a dominating set and a vertex cover is that the nodes don't have to cover every edge. The bowtie graph has a dominating set of size just 1, while its line graph needs $k = 2$ but can do so even by taking the two non-central nodes:



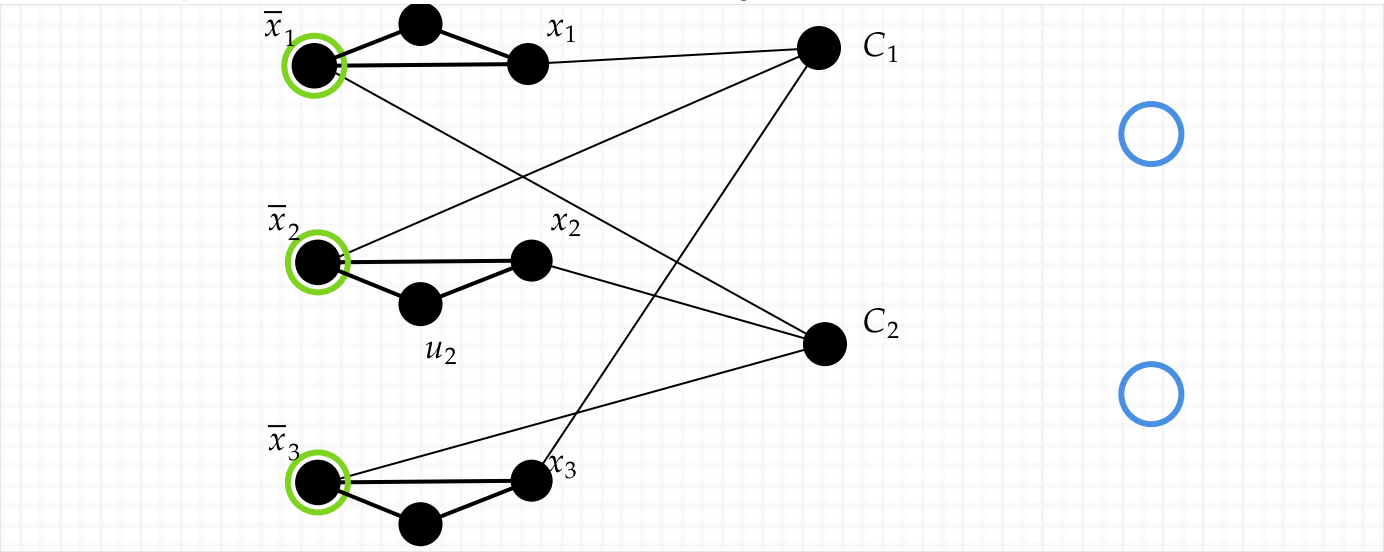
(Does the line-graph function give a reduction from **Edge Cover** to **Dom Set** or vice-versa? Hmmm... But we still want to reduce **3SAT** to **Dom Set** directly.)

The first key idea is the same: the rung nodes chosen in S correspond to those literals set true. The second key idea is simple: *make that true literal dominate every clause it satisfies.* This needs only one node per clause, and suggests taking $k = n$, irrespective of the number m of clauses. Here is how that looks for a simpler formula, $\psi = (x_1 \vee \overline{x}_2 \vee x_3) \wedge (\overline{x}_1 \vee x_2 \vee \overline{x}_3)$:



Setting all three variables false dominates the two clause nodes. So would setting them all true, whereas moving just x_2 to be true would fail to dominate (or satisfy) C_1. Is this all we need?

The flaw is that we have not enforced that in each rung, either x_i or $\overline{x}_i$ **must** be in S. This case allows a "surprise" domination by two nodes outside the rungs: use C_1 and C_2. To enforce the correspondence between possibly-good choices of S and truth assignments, and make sure $k = n$ is the minimum possible, we use a third node in each "rung":



Those extra nodes can only be dominated from the rung, and they do not help dominate each other, so n separate nodes are needed to dominate them. This fixes the problem. Defining the reduction formally in general and proving it correct is a self-study exercise.