CMSC 451: SAT, Coloring, Hamiltonian Cycle, TSP

Slides By: Carl Kingsford

Department of Computer Science
University of Maryland, College Park

Based on Sects. 8.2, 8.7, 8.5 of *Algorithm Design* by Kleinberg & Tardos.

Boolean Formulas

Boolean Formulas:

Variables: x_1, x_2, x_3 (can be either **true** or **false**)

Terms: $t_1, t_2, \ldots, t_\ell$: t_j is either x_i or $\bar{x}_i$
(meaning either x_i or **not** x_i).

Clauses: $t_1 \vee t_2 \vee \cdots \vee t_\ell$ ($\vee$ stands for "OR")
A clause is **true** if any term in it is **true**.

Example 1: $(x_1 \vee \bar{x}_2), (\bar{x}_1 \vee \bar{x}_3), (x_2 \vee \bar{v}_3)$

Example 2: $(x_1 \vee x_2 \vee \bar{x}_3), (\bar{x}_2 \vee x_1)$

Boolean Formulas

Def. A truth assignment is a choice of **true** or **false** for each variable, ie, a function $v : X \longrightarrow \{\mathbf{true}, \mathbf{false}\}$.

Def. A CNF formula is a conjunction of clauses:

$$C_1 \wedge C_2, \wedge \cdots \wedge C_k$$

Example: $(x_1 \vee \bar{x_2}) \wedge (\bar{x_1} \vee \bar{x_3}) \wedge (x_2 \vee \bar{v_3})$

Def. A truth assignment is a satisfying assignment for such a formula if it makes every clause **true**.

SAT and 3-SAT

Satisfiability (SAT)

Given a set of clauses $C_1, \ldots, C_k$ over variables $X = \{x_1, \ldots, x_n\}$ is there a satisfying assignment?

Satisfiability (3-SAT)

Given a set of clauses $C_1, \ldots, C_k$, each of length 3, over variables $X = \{x_1, \ldots, x_n\}$ is there a satisfying assignment?

Graph Coloring

Graph Coloring Problem

Graph Coloring Problem

Given a graph G, can you color the nodes with $\leq k$ colors such that the endpoints of every edge are colored differently?

Notation: A k-coloring is a function $f : V \rightarrow \{1, \ldots, k\}$ such that for every edge $\{u, v\}$ we have $f(u) \neq f(v)$.

If such a function exists for a given graph G, then G is k-colorable.

Graph Coloring is NP-complete

3-Coloring $\in$ **NP**: A valid coloring gives a certificate.

We will show that:

$$\text{3-SAT} \leq_P \text{3-Coloring}$$

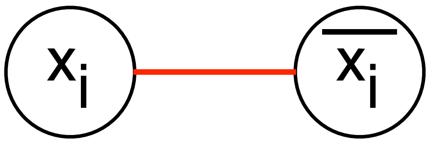
Let $x_1, \ldots, x_n, C_1, \ldots, C_k$ be an instance of 3-SAT.

We show how to use 3-Coloring to solve it.

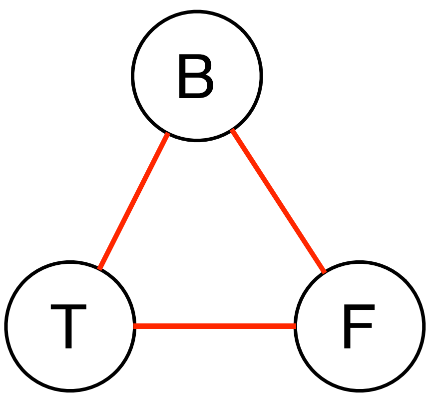
Reduction from 3-SAT

We construct a graph G that will be 3-colorable iff the 3-SAT instance is satisfiable.

For every variable x_i, create 2 nodes in G, one for x_i and one for $\bar{x}_i$. Connect these nodes by an edge:

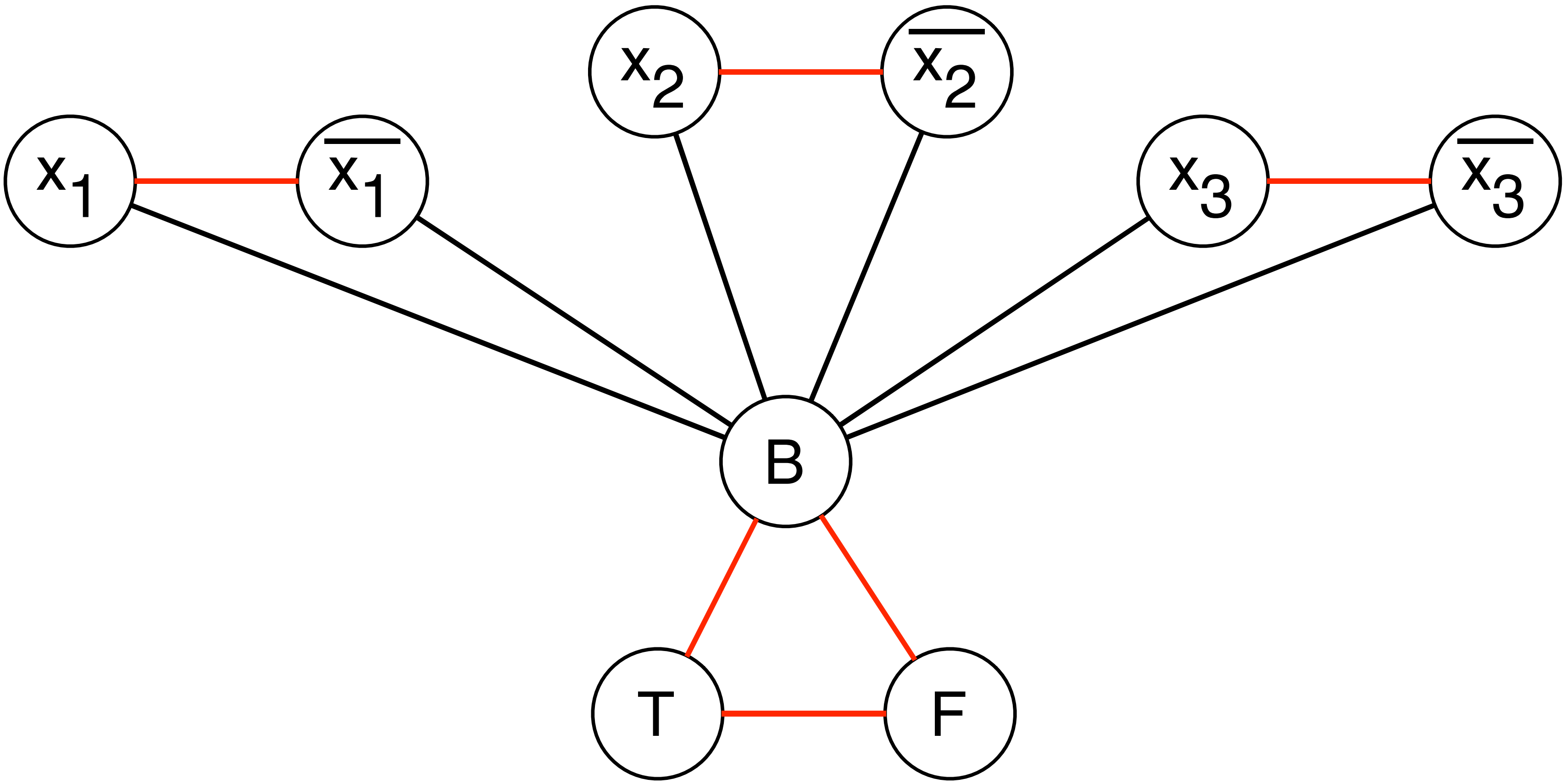


Create 3 *special nodes* T, F, and B, joined in a triangle:



Connecting them up

Connect every variable node to B:



Properties

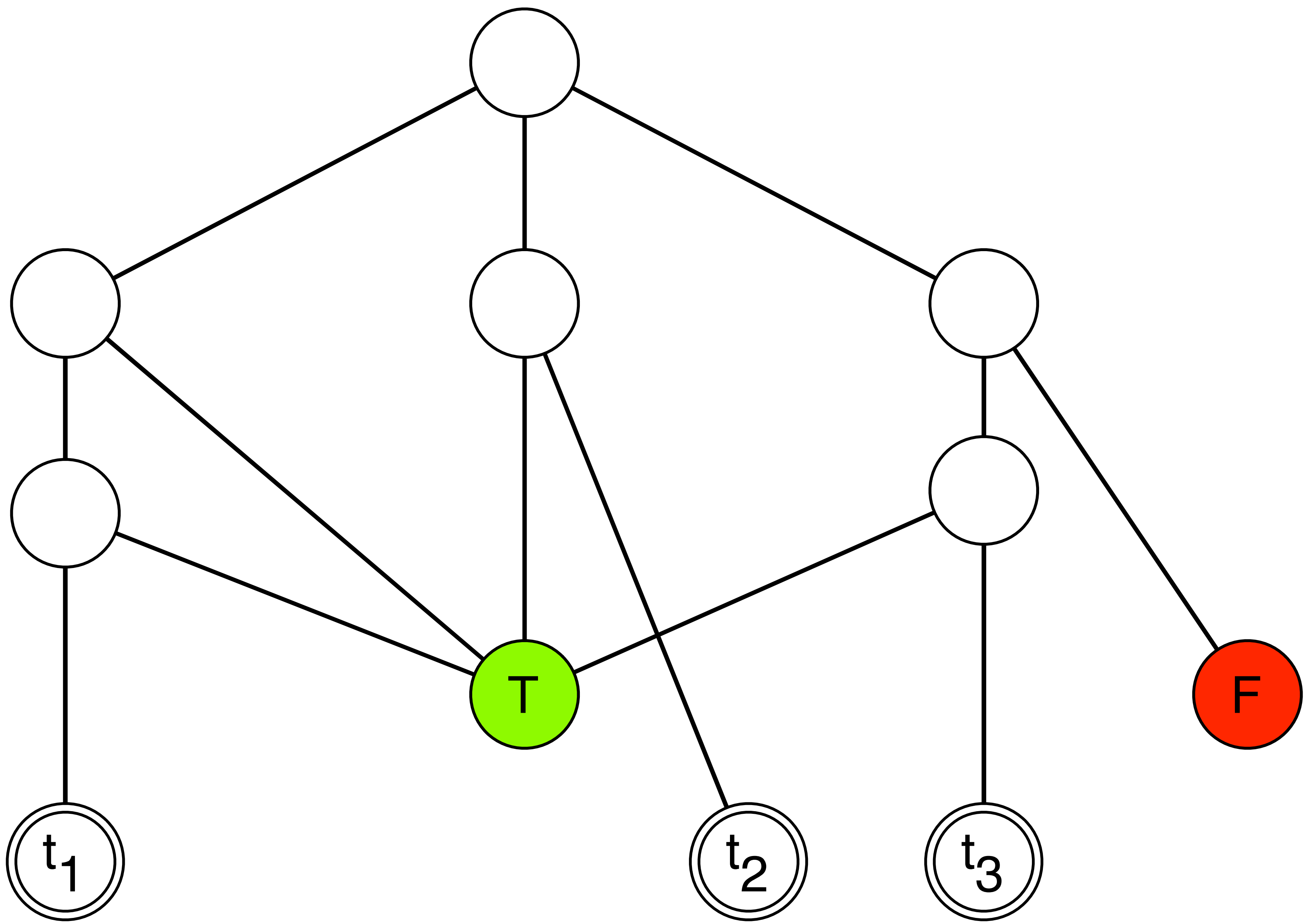
Properties:

- Each of x_i and $\bar{x}_i$ must get different colors
- Each must be different than the color of B.
- B, T, and F must get different colors.

Hence, any 3-coloring of this graph defines a valid truth assignment!

Still have to constrain the truth assignments to satisfy the given clauses, however.

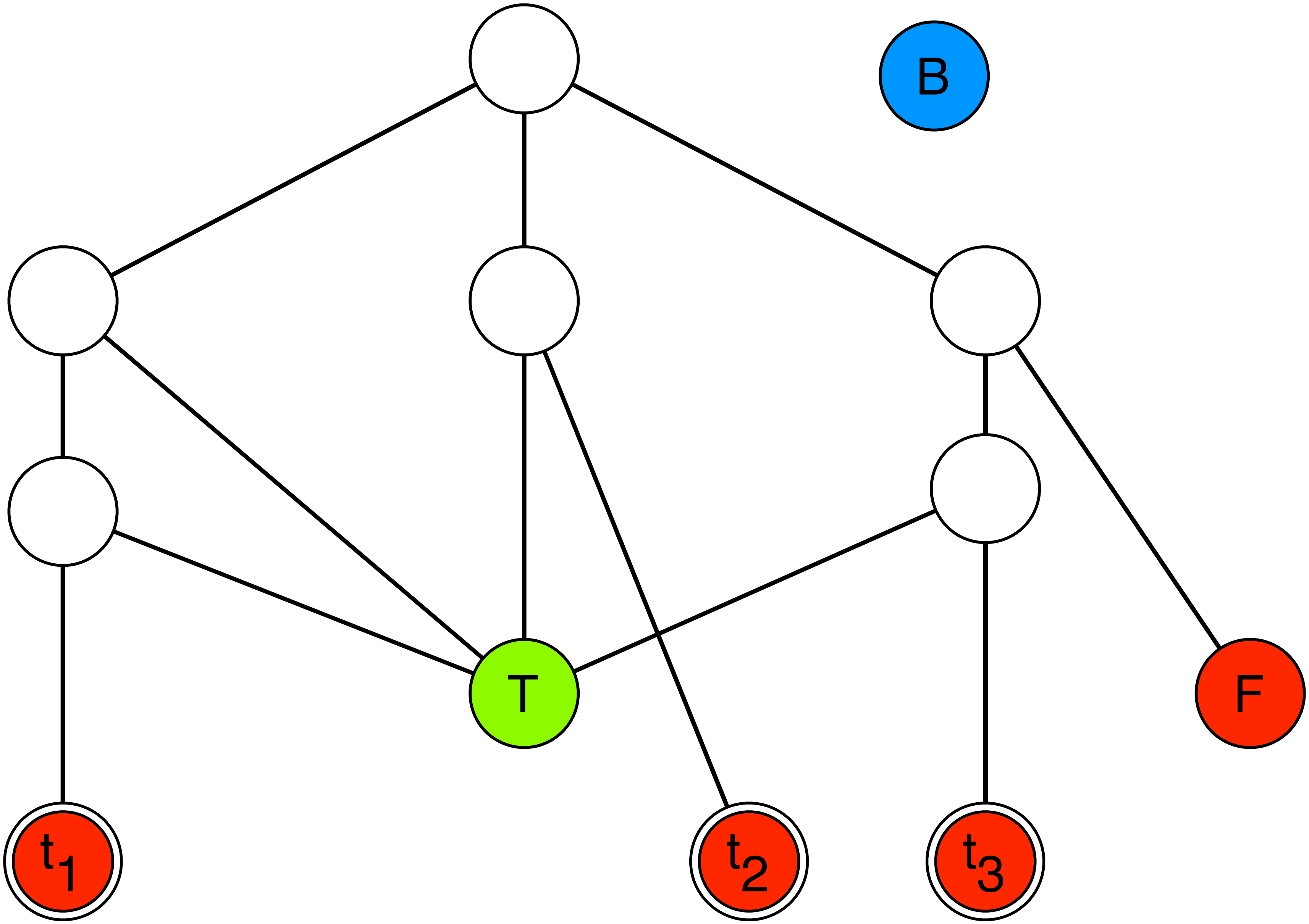
Connect Clause (t_1, t_2, t_3) up like this:



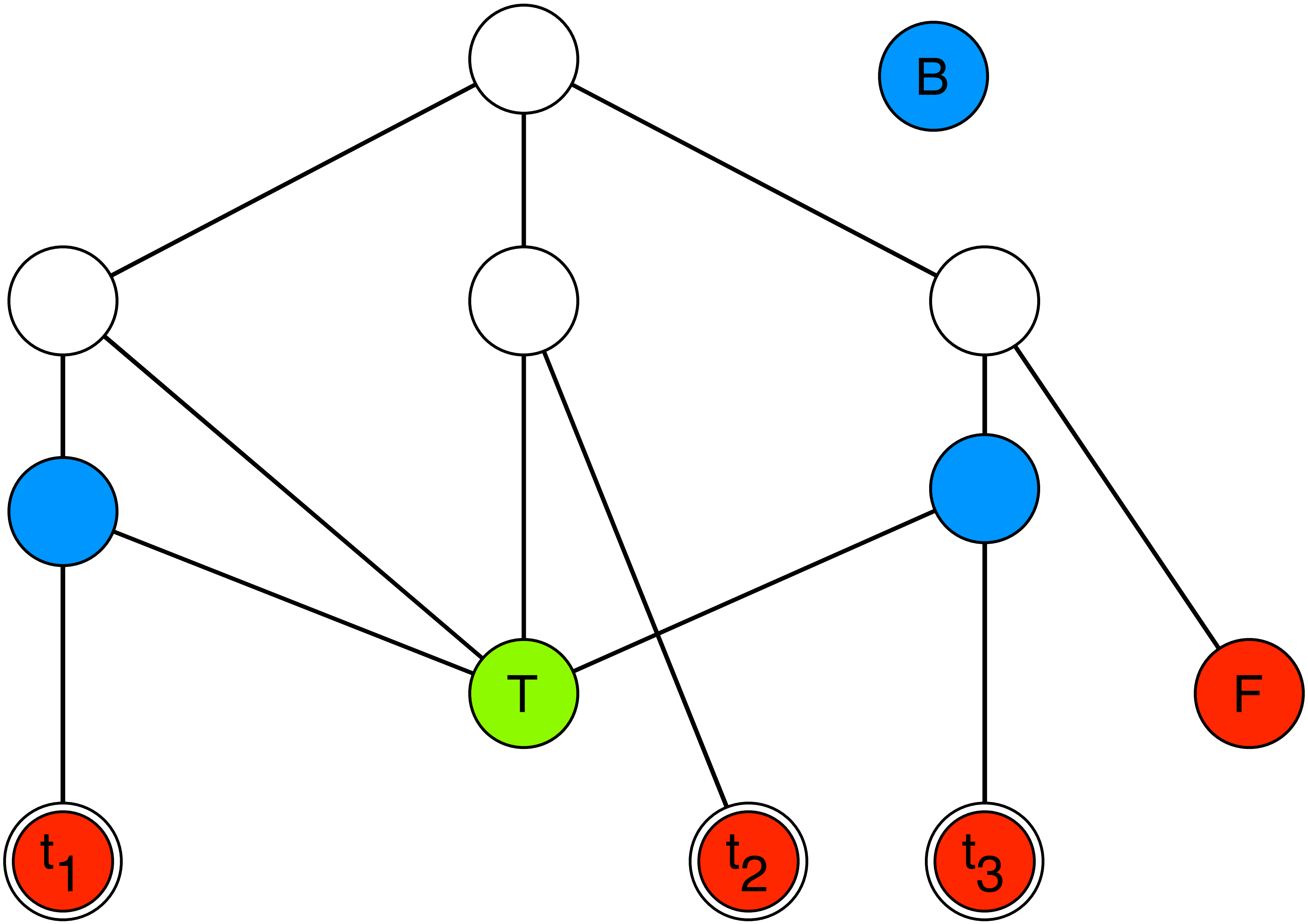
Suppose Every Term Was False

What if every term in the clause was assigned the **false** color?

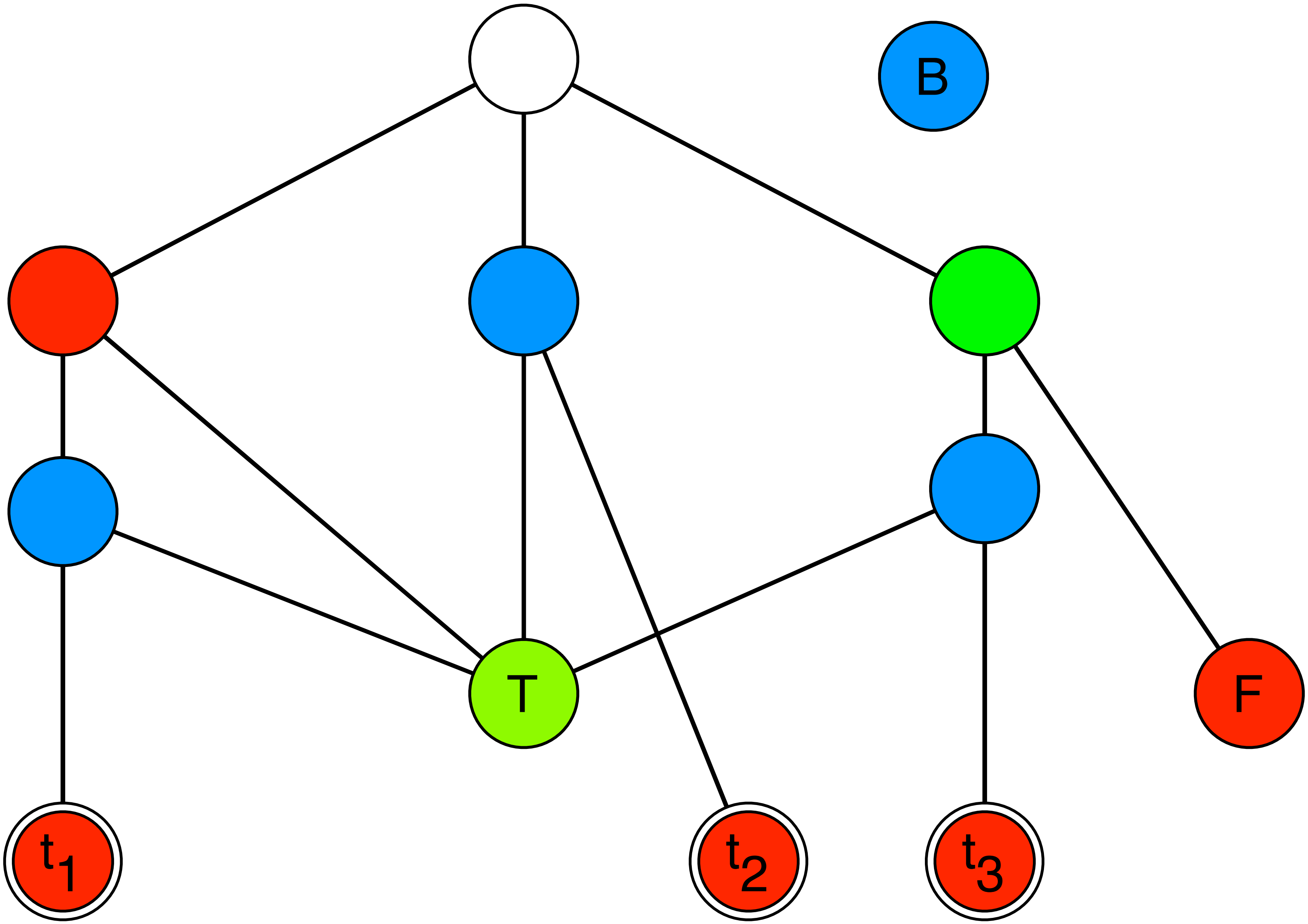
Connect Clause (t_1, t_2, t_3) up like this:



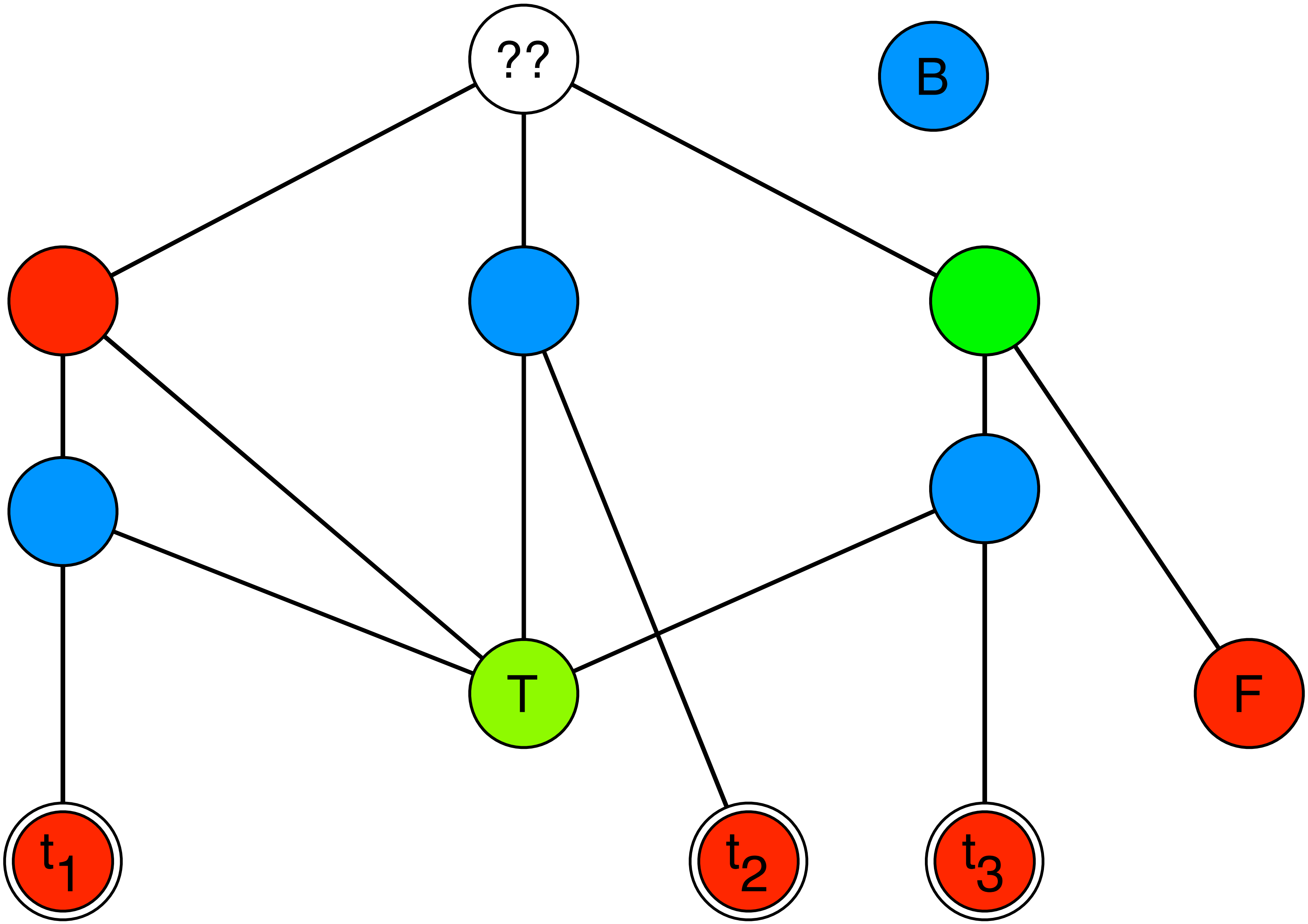
Connect Clause (t_1, t_2, t_3) up like this:



Connect Clause (t_1, t_2, t_3) up like this:



Connect Clause (t_1, t_2, t_3) up like this:



Suppose there is a 3-coloring

Top node is colorable iff one of its terms gets the **true** color.

Suppose there is a 3-coloring.

We get a satisfying assignment by:

- Setting $x_i =$ **true** iff v_i is colored the same as T

Let C be any clause in the formula. At least 1 of its terms must be true, because if they were all false, we couldn't complete the coloring (as shown above).

Suppose there is a satisfying assignment

Suppose there is a satisfying assignment.

We get a 3-coloring of G by:

- Coloring T, F, B arbitrarily with 3 different colors
- If $x_i = \textbf{true}$, color v_i with the same color as T and $\bar{v}_i$ with the color of F.
- If $x_i = \textbf{false}$, do the opposite.
- Extend this coloring into the clause gadgets.

Hence: the graph is 3-colorable iff the formula it is derived from is satisfiable.

General Proof Strategy

General Strategy for Proving Something is NP-complete:

1. Must show that $X \in \mathbf{NP}$. Do this by showing there is an certificate that can be efficiently checked.

2. Look at some problems that are known to be NP-complete (there are thousands), and choose one Y that seems "similar" to your problem in some way.

3. Show that $Y \leq_P X$.

Strategy for Showing $Y \leq_P X$

One strategy for showing that $Y \leq_P X$ often works:

1. Let I_Y be any instance of problem Y.
2. Show how to construct an instance I_X of problem X in polynomial time such that:
   - If $I_Y \in Y$, then $I_X \in X$
   - If $I_X \in X$, then $I_Y \in Y$