Example 4.12 (Loss of information in coverability tree)
This example illustrates the loss of information regarding specific reachable states when a coverability tree is constructed for an unbounded Petri net. In Fig. 4.15 (a) we show a simple Petri net and its coverability tree. Observe that the first time transition $t_2$ fires the new state is $[1, 0, 1]$. The next time transition $t_2$ fires the new state becomes $[1, 0, 2]$, and so on. Thus, the reachable markings for place $p_3$ are $x(p_3) = 1, 2, \ldots$. All those are aggregated under the symbol $\omega$.

In Fig. 4.15 (b), we show a similar - but different - Petri net along with its coverability tree. Note that in the Petri net of Fig. 4.15 (b), the first time transition $t_2$ fires the new state is $[1, 0, 2]$. The next time transition $t_2$ fires the new state becomes $[1, 0, 4]$, and so on. The reachable markings for $p_3$ in this case are $x(p_3) = 2, 4, \ldots$, and not all positive integers as in the Petri net of Fig. 4.15 (a). Despite this difference, the coverability trees in Figs. 4.15 (a) and (b) are identical. The reason is the inherent vagueness of $\omega$.

![Diagram](image.png)

**Figure 4.15:** Two unbounded Petri nets and their coverability trees (Example 4.12). In (a), the symbol $\omega$ for place $p_3$ represents the set $\{1, 2, \ldots\}$, while in (b), $\omega$ for the same place represents the set $\{2, 4, \ldots\}$.

As Example 4.12 illustrates, state reachability issues cannot be dealt with, in general, using a coverability tree approach. These limitations of the cover-
**Analysis Problems**

**Boundedness:** A place is **bounded** if there is an upper bound on # of tokens.

- K-bounded if \( \nu(p_i) \leq k \)
- Safe if \( \nu(p_i) \leq 1 \)

**Petri Net** is \( \{ \text{bounded} \} \) if all its places are \( \{ \text{k-bounded} \} \)

- K-bounded
- Safe

**PN is conservative** if total # of tokens always same w.r.t. \( \gamma \), \( \gamma \geq 0 \):

\[ \sum_i \gamma_i \nu(p_i) = \text{constant} \]

**Transition is dead** if it cannot fire. (EVER)

**Transition is live** if it may always be enabled.

**L1-live:** There is a sample path s.t. \( t_j \) can fire once from \( \alpha_0 \)

**Live:** \( t_j \) is L1-live for every reachable state
FLAVORS OF LIVENESS

Definition. (Liveness)
Petri net $N$ with initial state $x_0$ is said to be live if there always exists some sample path such that any transition can eventually fire from any state reached from $x_0$.

Clearly, this is a very stringent condition on the behavior of the system. Moreover, checking for liveness as defined above is an extremely tedious process, often practically infeasible for many systems of interest. This has motivated a further classification of liveness into four levels. Thus, given an initial state $x_0$, a transition in a Petri net may be:

- **Dead or L0-live**, if the transition can never fire from this state;

- **L1-live**, if there is some firing sequence from $x_0$ such that the transition can fire at least once;

- **L2-live**, if the transition can fire at least $k$ times for some given positive integer $k$;

- **L3-live**, if there exists some infinite firing sequence in which the transition appears infinitely often;

- **Live or L4-live**, if the transition is L1-live for every possible state reached from $x_0$.

![Figure 4.10: Petri net for Example 4.7. Transition $t_2$ is dead, because it can never fire. Transition $t_1$ is L1-live, since it can fire once. Transition $t_3$ is L3-live, because it can fire an infinite number of times, but it is not L4-live, since it becomes dead if $t_1$ fires.](image-url)
**Petri Net Supervisors**

Des \( G \) with \( \mathcal{L}(G) = a^* b^* \)

Admissible language \( \mathcal{L}_a = \{ a^n b^n : n = m = 0 \} \)

**Note:** \( \mathcal{L}_a \) not regular

\( \mathcal{L}(G) \) is regular

Suppose \( \mathcal{E}_c = \{ a \} \)

**Petri Net Supervisor**

\[ \begin{array}{c}
P_1 \\
\downarrow t_1 \\
\downarrow q \\
\downarrow t_2 \\
P_2 \\
\downarrow t_3 \\
P_3
\end{array} \]

For \( s \in \mathcal{L}(G) \) and \( x = f(x_0, s) \)

\[ s(s) = \begin{cases} 
\{ a, b \}, & x(p_3) > 0 \\
\{ a \}, & x(p_3) = 0 
\end{cases} \]

**Control of Petri Nets:** Idea is specifications

Add conditions \( \Rightarrow \) places

Goal:

\[ x(p_2) + x(p_3) \leq 2 \]