

Petri Nets — Part 2

System Behavior & Simple Analysis

BSc Course in Electronic & Computer Engineering

Quick Recap of Part 1

What We Learnt So Far

- Places (Circles) hold Tokens (Black Dots) to represent system state.
- Transitions (Rectangles) represent events or actions.
- Enabling: An event is ready if input places have enough tokens.
- Firing: Consumes input tokens and creates output tokens instantly.
- Now in Part 2: How do we verify that a system works correctly without bugs or crashes?

Important Questions Engineers Must Ask

System Safety Questions

- 1. Will storage buffers overflow? (Boundedness & Safety)
- 2. Can the system reach dangerous states? (Reachability)
- 3. Will the machine get completely stuck? (Deadlock Freedom)

System Health Questions

- 4. Can every action still happen in the future? (Liveness)
- 5. Are resources conserved over time? (P-Invariants)
- 6. Can the process reset and repeat cyclic operations? (T-Invariants)

Concept 1: Reachability $R(M_0)$

Where Can the System Go?

- A marking M_k is REACHABLE if there exists a valid sequence of firings starting from M_0 .
- The Reachability Set $R(M_0)$ contains ALL possible states the system can ever enter.
- If a state is NOT in $R(M_0)$, the system can NEVER reach it!

FORMULA & NOTATION: Reachability Set Notation

$$R(M_0) = \{ M \mid M_0 \rightarrow[*] M \}$$

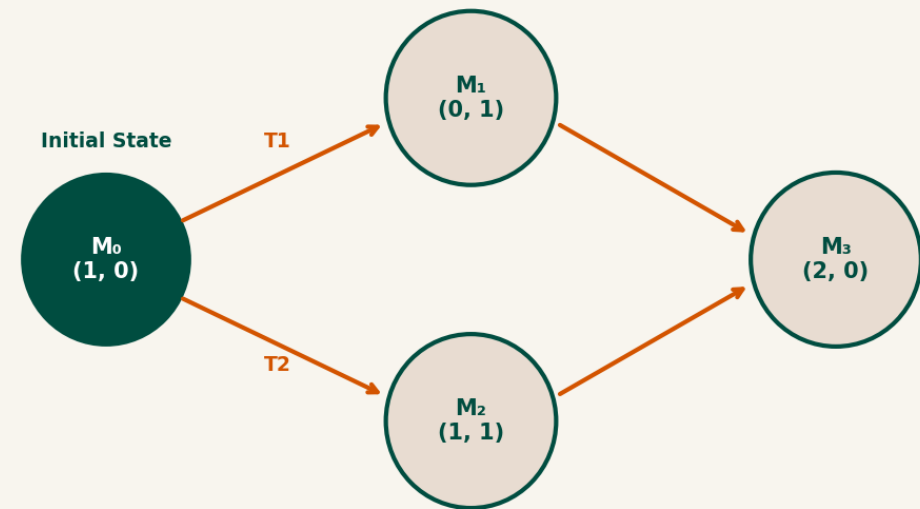
- $M_0 \rightarrow[*] M$ means state M is reached from M_0 after 0 or more firings.
- Essential for safety verification: prove that unsafe states $\notin R(M_0)$!

The Reachability Map (State Tree)

A GPS Roadmap of States

- We draw states (Markings) as nodes.
- We draw event firings as directed arrows.
- This forms the Reachability Tree, showing every path the system can take.

Reachability Map (GPS Roadmap of States)



Concept 2: Boundedness (Preventing Overflow)

What is Boundedness?

- A Petri Net is K-BOUNDED if no place ever holds more than K tokens in any reachable state.
- Guarantees that physical storage or memory buffers will NEVER overflow!
- K is the maximum token capacity bound.

FORMULA & NOTATION: K-Boundedness Formula

$$M(p) \leq K \quad \forall M \in R(M_0)$$

- For every place p and every reachable marking M, token count $\leq K$.
- If $K = 1$, the net is called SAFE!

Concept 3: Safe Nets (1-Bounded Nets)

Binary State Places

- A net is SAFE if every place holds at most 1 token ($K = 1$).
- Tokens act as binary flags: 1 = Condition TRUE, 0 = Condition FALSE.
- Extremely common in hardware logic and software flags.

Real Examples of Safe Places

- Light Switch: 1 = ON, 0 = OFF.
- Single Robot Arm: 1 = Free, 0 = Busy.
- Traffic Light: 1 = Green, 0 = Red.

Unbounded Nets: Infinite Token Growth

When Tokens Keep Growing

- If an event produces more tokens than it consumes in a loop.
- Token count grows infinitely: 1, 2, 3, 4, ... ∞ .
- Represents memory leaks or buffer overflows in real engineering!

Unbounded System & Infinity Symbol ω



Tokens keep growing infinitely! Set to Symbol ω (Omega)

The Symbol ω (Omega) Made Simple

FORMULA & NOTATION: Infinity Symbol ω

ω (Omega = Infinitely Large Tokens)

- To keep state maps finite, we replace growing token counts with the symbol ω .
- ω represents an arbitrarily large number of tokens that keeps increasing.
- Arithmetic rules of ω : $\omega + 1 = \omega$ and $\omega - 1 = \omega$ and $\omega > n$ for any integer n .
- If ω appears in your Coverability Tree, that place is UNBOUNDED!

Concept 4: Liveness (Can Events Still Happen?)

What is Liveness?

- Liveness measures whether events remain reachable in the future.
- Guarantees that no operation becomes permanently disabled (dead).
- A LIVE system can continue working forever.

Levels of Liveness

- L0 (Dead): Event can NEVER fire.
- L1: Event can fire at least ONCE.
- L4 (Live): Event can ALWAYS be enabled again from any future state!

Concept 5: Deadlock (Completely Stuck!)

What is a Deadlock?

- A state where NO transition is enabled!
- Zero events can fire — execution halts permanently.
- The system is completely frozen.

FORMULA & NOTATION: Deadlock Condition

$$\text{Enabled}(M) = \emptyset$$

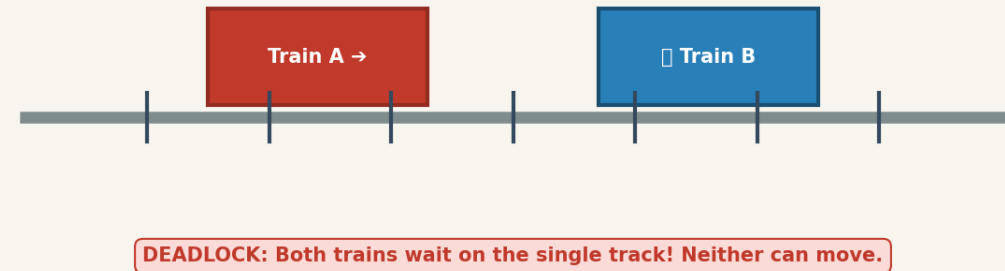
- No transition t satisfies the enabling condition at marking M .
- Every input place lacks tokens.
- Engineers MUST design systems to be Deadlock-Free!

Real Example: Two Trains on a Single Track

Circular Waiting

- Train A occupies Track 1 and waits for Track 2.
- Train B occupies Track 2 and waits for Track 1.
- Neither train can move forward — DEADLOCK!

Deadlock Real-World Analogy: Trains on Single Track



How Petri Nets Help Us Fix Deadlocks

Supervisor Control Places

- 1. We analyze the Petri Net to find 'Siphons' (subnets that lose tokens).
- 2. We add a Supervisor Control Place that acts as a traffic light.
- 3. The supervisor prevents Train B from entering Track 2 if Train A is already approaching!
- 4. Result: The system becomes provably DEADLOCK-FREE.

Linear Algebra Tool: Incidence Matrix A

FORMULA & NOTATION: Incidence Matrix A ($m \times n$)

$$A_{ij} = W(t_j, p_i) - W(p_i, t_j)$$

- Matrix A has m rows (Places) and n columns (Transitions).
- A_{ij} represents the NET CHANGE of tokens in place p_i when transition t_j fires.
- Positive entry (+1): Net token GAINED in place p_i .
- Negative entry (-1): Net token CONSUMED from place p_i .
- Zero entry (0): No connection or no net change.

The Linear State Equation $M_k = M_0 + A \cdot x$

FORMULA & NOTATION: Fundamental State Equation

$$M_k = M_0 + A \cdot x$$

- M_0 : Initial Marking Vector (Starting State).
- M_k : Final Marking Vector (Target State).
- A : Incidence Matrix.
- x : Firing Count Vector (records how many times each transition fired).
- If state M_k is reachable, the equation MUST have a non-negative integer solution $x \geq 0$!

Important Warning: Necessary vs Sufficient

Mathematical Caution

- If state M_k IS reachable $\rightarrow$ Equation $M_k = M_0 + A \cdot x$ ALWAYS has a solution $x \geq 0$ (Necessary Condition).
- BUT having a solution x does NOT 100% guarantee M_k is reachable!
- Why? Because vector x might require firing transitions in an impossible order (spurious solutions).
- It is a fast screening test used by computer verification algorithms.

Concept 6: Token Conservation (P-Invariants)

FORMULA & NOTATION: Place Invariant Condition

$$\mathbf{y}^T \cdot \mathbf{A} = \mathbf{0} \quad \rightarrow \quad \mathbf{y}^T \cdot \mathbf{M}_k = \mathbf{y}^T \cdot \mathbf{M}_0$$

- A P-Invariant is a weighted group of places y where total tokens NEVER change!
- $\mathbf{y}^T \cdot \mathbf{M}_k = \mathbf{y}^T \cdot \mathbf{M}_0$ for ALL reachable states $\mathbf{M}_k$.
- Proves that resource conservation laws hold physically.
- Guarantees that places in the invariant are STRUCTURALLY BOUNDED!

Real Example: Conserving 3 Robot Arms

Tracking Physical Equipment

- In a factory with 3 robot arms:
- Place P1: 'Idle Robots' + Place P2: 'Robots Working' + Place P3: 'Robots Repairing'.
- P-Invariant Equation: $M(P1) + M(P2) + M(P3) = 3$ Robots (Constant!).
- Robots move between states, but total robots in the plant is ALWAYS 3.

Concept 7: Cycles (T-Invariants)

FORMULA & NOTATION: Transition Invariant Condition

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{0} \rightarrow \mathbf{M}_k = \mathbf{M}_0$$

- A T-Invariant is a firing count vector x that returns the net back to its original state M_0 !
- Represents a complete cyclic work path.
- Essential for machines that repeat the same routine endlessly.

Real Example: Repeating Robot Work Loop

Cyclic Operation

- Step 1 (T1): Pick up part.
- Step 2 (T2): Weld part.
- Step 3 (T3): Drop off part.
- Firing sequence $x = [1, 1, 1]^T$ returns system state to M_0 , ready for the next part!

Engineering Checklist for Safe Systems

4 Steps to Verify a Design

- ✓ Step 1 (Boundedness): Check Coverability Tree for ω symbols → Ensure no buffer overflow.
- ✓ Step 2 (Liveness): Verify all transitions are L4 → Ensure no dead actions.
- ✓ Step 3 (Deadlock Freedom): Check $\text{Enabled}(M) \neq \emptyset$ for all M → Ensure no freezes.
- ✓ Step 4 (Conservation): Solve $y^T \cdot A = 0$ → Verify resource conservation.

Summary of Analysis Tools

Graphical Methods

- Token Game: Interactive simulation.
- Reachability Map: GPS roadmap of all reachable states.
- Coverability Tree: Uses ω symbol to handle infinite states.

Algebraic Methods

- Incidence Matrix A : Net token change per event.
- State Equation: $M_k = M_0 + A \cdot x$.
- P-Invariants ($y^T \cdot A = 0$): Token conservation.
- T-Invariants ($A \cdot x = 0$): Cyclic work loops.

Final Synthesis of Both Modules

Complete Overview

- Petri Nets combine simple visual diagrams with powerful linear algebra.
- They excel at modeling parallel actions, resource sharing, choices, and queues.
- Engineers use them to design deadlock-free factories, hardware circuits, and software workflows.

Review & Practice Questions

Self-Test Questions

- 1. What shape represents a Place? What shape represents a Transition?
- 2. When is a Transition enabled?
- 3. What happens when a Transition fires?
- 4. What does the symbol ω mean in a Coverability Tree?
- 5. What equation defines a P-Invariant?

Thank You!

You now possess the foundational tools to understand and analyze Petri Nets.