A Petri Net Model of a Hospital OPD

Thesis submitted
in Partial Fulfillment of the Requirement for the
Degree of

MASTER OF SCIENCE
IN
APPLIED MATHEMATICS

by

Ishpreet Kaur
(23/MSCMAT/77)

Under the supervision of
Prof. Sangita Kansal
Department of Applied Mathematics
Delhi Technological University

DEPARTMENT OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Shahbad Daulatpur , Main Bawana Road, Delhi – 110042 , India

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)
Shahbad Daulatpur , Main Bawana Road, Delhi – 110042

CANDIDATE'S DECLARATION

I, **Ishprret Kaur**, Roll No. 23/MSCMAT/77, hereby certify that the work which is being presented entitled *"A Petri Net Model of a Hospital OPD"*,in partial fulfillment of the requirements for the degree of Master of Science, submitted in Department of Applied Mathematics, Delhi Technological University is an authentic record of my own work carried out during the period from August 2024 to May 2025 under the supervision of Prof. Sangita Kansal.

The matter presented in the thesis has not been submitted by me for the award of any other degrees of this or any other Institute.

Candidate's Signature

This is to certify that the student have incorporated all the correction suggested by the examiners in the thesis and the statement made by the candidate to the best of our knowledge.

Signature of Supervisior **Signature of External Examiner**

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur , Main Bawana Road, Delhi – 110042

CERTIFICATE BY THE SUPERVISOR

Certified that **Ishpreet kaur**, Roll No. 23/MSCMAT/77 have carried her research work presented in this thesis entitled **"A Petri Net Model of a Hospital OPD"** for the award of **Master of Science** from Department of Applied Mathematics, Delhi Technological University, Delhi, under my supervision. The thesis embodies results of original work, and studies are carried out by the student herself and the contents of the thesis do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University.

Signature

(Sangita Kansal)

(Department of Applied Mathematics)

Date:

Abstract

In Hospitals, managing patient flow in OPD is crucial to minimize waiting time, ensuring optimal resource utilization, and improving patient satisfaction. In this paper, a Petri net model for hospital OPD has been constructed and used to solve the problem of queuing for too long. Petri net allows for detailed visualization due to its graphical representation.

Petri nets are applied to model the movement of patients through various stages of OPD, such as registration, consultation, diagnostic tests, and discharge. Furthermore, its boundedness, liveness, and deadlock are discussed in detail. To reduce waiting time, online registration through an app/website to avoid standing in a queue for a long time has been introduced.

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Shahbad Daulatpur , Main Bawana Road, Delhi – 110042

ACKNOWLEDGMENT

I want to express my appreciation to Prof. Sangita Kansal, Department of Applied Mathematics, Delhi Technological University, New Delhi, for her careful and knowledgeable guidance, constructive criticism, patient hearing, and kind demeanour throughout our ordeal of the present report. I will always be appreciative of her kind, helpful and insightful advice, which served as a catalyst for the effective completion of our dissertation report.

I am grateful to our Department of Mathematics for their continuous motivation and involvement in this project work. I am also thankful to all those who, in any way, have helped me in this journey. Finally, I am thankful to the efforts of my parents and family members for supporting me with this project.

Contents

List of Tables

List of Figures

Chapter 1

Introduction and Literature Review

1.1 Introduction

Background

In recent years, healthcare systems around the world have been under increasing pressure due to rising patient loads, limited medical resources, and the growing complexity of treatment processes. Among the various units of a hospital, the Outpatient Department (OPD) plays a crucial role as it is the primary point of contact for non-emergency patients. The OPD involves several key activities such as patient registration, waiting, consultation with doctors, diagnostic tests, and prescription services.

Efficient management of these processes is essential to reduce patient waiting times, avoid congestion, and ensure smooth operations. However, due to the inherent complexity, concurrency, and uncertainty in patient flow and resource availability, managing an OPD effectively remains a challenge.

Mathematical modeling and simulation of OPD workflows can help healthcare administrators understand and improve system performance. In this context, Petri nets offer a formal and visual tool that can represent the dynamic and concurrent nature of OPD processes, enabling both qualitative and quantitative analysis.

Petri nets, being a graph-based mathematical modeling tool, are capable of representing such complexities in a clear and analyzable manner. Their ability to model concurrency, synchronization, and resource sharing makes them highly suitable for modeling hospital OPD workflows.Moreover, Petri nets support formal analysis techniques (e.g.,

reachability, liveness, deadlock detection), allowing system designers to evaluate the correctness and efficiency of the workflow before implementing any changes.

Objective

The primary objective of this thesis is to model the workflow of a typical hospital OPD using Petri nets. The specific goals include:

- To represent key components of an OPD (registration, consultation, diagnostics, etc.) using a Petri net framework.
- To represent the workflow of patients and resources.
- To analyze potential bottlenecks and system performance.
- To analyze system properties like resource utilization, delays, and possible improvements.

Scope of the Study

The scope of this thesis is limited to modeling a general OPD in a hospital. The model is based on typical procedures found in government or large public hospitals, focusing on:

- Patient registration and appointment handling.
- Doctor consultation process.
- Diagnostic testing and result handling.
- Prescription and exit procedures.

Exclusions:

- Emergency services , surgical units are not considered in this model.
- The study does not involve real patient data but simulates general workflow based on standard OPD structures.

1.2 Literature Review

Efficient healthcare service delivery, especially in outpatient departments (OPDs), remains a challenge due to increasing patient loads, limited resources, and the need for timely diagnosis and treatment. Modeling such systems can help administrators understand process flow, identify bottlenecks, and evaluate performance. This chapter reviews existing modeling techniques applied to healthcare systems, with a focus on Petri nets and their application in OPD settings.

Petri Nets in Healthcare Modeling

Petri nets (PNs) are a mathematical modeling tool ideal for representing concurrent, and distributed systems. They consist of places, transitions, and tokens that visually and formally describe the flow of information or resources.

Applications in Healthcare

- **Emergency Department Modeling:** Petri Nets have been used to model the complex patient journey through emergency units, capturing concurrent activities like testing, and treatment.
- **Surgical Scheduling:** Petri nets have been applied to model operating room scheduling, accounting for resource availability and time constraints.
- **Clinical Pathways and Diagnostic Labs** They effectively represent workflows with dependencies, such as blood tests or imaging procedures.

Advantages of Petri nets include:

- The graphical representation allows for easy visualization and understanding of complex system dynamics.
- Formal verification through reachability, boundedness, and liveness analysis
- Extension capabilities (e.g., timed, colored, or stochastic Petri nets)

Existing OPD Models Using Petri Nets

While Petri nets have been applied in various hospital departments, their use specifically in Outpatient Department modeling is limited. Some relevant works include:

- Models representing patient registration and consultation processes.
- Studies addressing resource allocation and waiting time reduction.
- Applications focused on workflow optimization for specific hospital services.

However, most models are either:

- Narrow in scope (focusing on a single service like diagnostics)
- Lack formal Petri net representation,
- Does not have work on Tokens

This shows a gap in the literature for a comprehensive, formally defined, and analyzable Petri net model for general OPD workflows.

Summary

This chapter highlighted key modeling methods used in healthcare systems, emphasizing their benefits and limitations. Petri nets emerge as a strong candidate for OPD modeling due to their ability to represent concurrency, resource usage, and process dynamics in a formal and visual way.

Despite several successful applications in healthcare, a unified and formal Petri net model of a hospital OPD remains underexplored, thus motivating the development of the model presented in this thesis.

Chapter 2

Theoretical Background

2.1 Background

This chapter introduces the fundamental concepts and mathematical structure of Petri nets, which serve as the modeling framework for this study. It covers the basic components, formal definitions, behavioral properties and analyzing complex systems like hospital workflows.

2.2 Basic Concept of Petri Net

Petri net is invented by Carl Adam Petri in 1962, as part of his Phd. It is used to model the functionality/Behaviour of the system. It is a formal/graphical language for modelling systems with concurrency. It is a bipartite graph which includes

- **Places (Circle)** - represent conditions or resources (e.g., waiting patients).
- **Transition (Rectangle or bars)** - represent event/activity (e.g., Registering Patients)
- **Tokens (Dots)** - represent the presence of a condition or availability of a resource.

Places and transitions are connected by directed arcs (no place connects directly to another place, and no transition connects directly to another transition).

2.3 Formal Definitions

Petri Net

A Petri Net is a 5-tuple (P, T, I, O, M) where:

- P and T are non-empty finite sets of places and transitions, respectively.
- $I: \ P \times T \rightarrow \mathbb{N}$ denotes the input function, with $\mathbb{N}$ being the set of non-negative integers.
- $O: \ T \times P \rightarrow \mathbb{N}$ denotes the output function.
- M is called marking, indicating the number of tokens in each place.

Places

Places represent conditions or states within the system. They are graphically represented as empty circles. They can hold tokens, which indicate the current state of the system.

Transitions

Transitions represent events or activities that occur in the system. They are graphically represented as rectangles or bars. A transition fires if all its input places have the required number of tokens.

Tokens

Tokens represent resources, objects, or information within the system. They are represented as dots inside places. Their movement between places through transitions models the dynamics of the system.

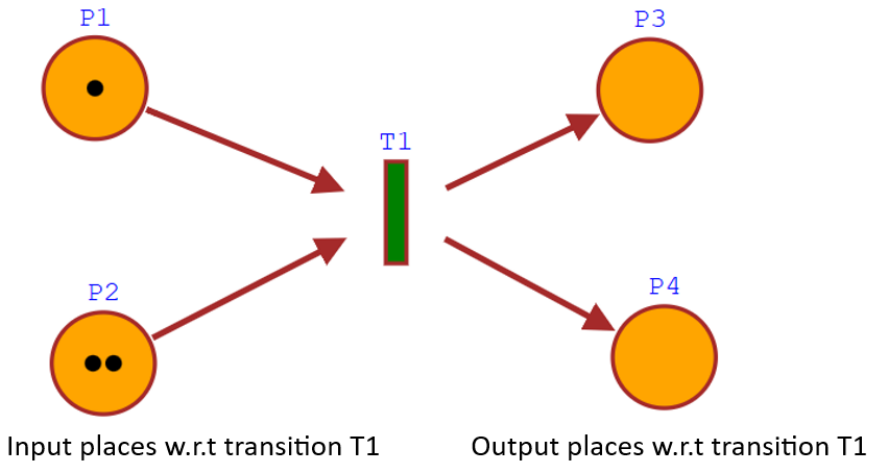


Figure 2.1: Pictorial representation showing input places, transitions, and output places of Petri net

Marking

Marking in Petri Net refers to the allocation of the tokens across the places at a given time.It represents the current state of the system. If a PN has places P_1, P_2, P_3, a marking M can be written as:

$$M = (m_1, m_2, m_3) \tag{2.1}$$

where m_1, m_2, m_3 represent the number of tokens in P_1, P_2, P_3 respectively.

M_0 represents initial marking (initial token distribution).

Arc Weight

The arc weight represents the number of tokens that an arc consumes from an input place or produces in an output place when the associated transition fires. It is usually a positive integer (default weight is 1 if not explicitly mentioned).

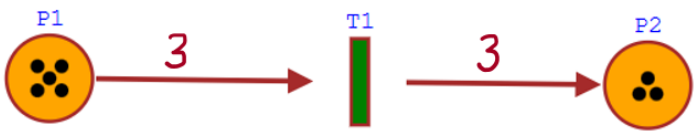


Figure 2.2: Representation of Arc weight

Here 3 written on the arc represents the arc weight.

- Input Arcs(Places $\rightarrow$ Transition): The transition requires number of tokens equal to the arc weight from the input place to fire. When the transition fires, it deducts quantity of tokens according to the arc weight from an input place.
- Output Arcs(Transition $\rightarrow$ Places): When the transition fires, it adds number of tokens equal to the arc weight to the output place.

The Arc weight is labelled on the arc in a Petri Net diagram as shown in above figure.

2.4 Firing rule in a Petri Net

2.4.1 Conditions for firing a transition:

A transition T is enabled when all of its input places has at least as many tokens as required by the input arc weights which means, if a place P has an arc to transition T with weight w, then P must have at least w tokens for T to fire:

$$I^{-}(P,T) \leq M(P) \tag{2.2}$$

2.4.2 Effects of firing a Transition:

When a transition T fires, it removes tokens from all of its input places according to the input arc weight and adds it to all of its output places according to the output arc weights.

Tokens can be consumed by the transition as well.

Example : Suppose transition t_1 represents "Doctor Consultation". It is connected to an input place "Waiting Room" and an output place "Consulted Patients". When a patient (token) is in the "Waiting Room", t_1 is enabled. Firing it will move the token from "Waiting Room" to "Consulted Patients".

2.4.3 Updated Marking:

The updated marking is as follows-

$$M'(P) \ = M(P) - I^{-}(P,T) + I^{+}(P,T) \tag{2.3}$$

where $M'(P)$ be the new marking of place P after transition T fires, $I^{-}(P,T)$ be the tokens removed from place P by transition T, and $I^{+}(P,T)$ be the tokens added to place P by transition T. $M(P)$ represents the current marking of place P.

Figure 2.3: Movement of tokens after firing

Here 2 tokens are removed from p_1 but only 1 token is given to p_2. This means that one token is consumed by the transition t_1. **So its not always necessary that number of tokens consumed from input places would be equal to the number of tokens produced in output places**.

$M_0 = (3, 1) \rightarrow$ Initial marking

New Marking after $T1$ fires:

$M' = (1, 2) \rightarrow$ Updated marking

2.5 Structural Properties

1. **P-invariant :** A P-invariant (Place Invariant) is a vector that, when multiplied with any marking vector over time, gives a constant result. It is useful in proving conservation properties.

2. **T-invariant :** A T-invariant (Transition Invariant) is a multiset of transitions that, when fired in sequence, returns the net to its original marking. It helps identify cyclic behaviors.

3. **Incidence Matrix:** Let a Petri Net be defined as

$PN = (P, T, F, W)$, where:

- $P = \{p_1, p_2, \ldots, p_m\}$ is the set of places,
- $T = \{t_1, t_2, \ldots, t_n\}$ is the set of transitions,
- $F \subseteq (P \times T) \cup (T \times P)$ is the flow relation,
- $W : F \to \mathbb{N}$ assigns weights to arcs.

The *Incidence Matrix* C is an $m \times n$ matrix where each entry $C(i, j)$ is given by:

$$C(i, j) = W(t_j, p_i) - W(p_i, t_j)$$

- $C(i, j) > 0$ indicates that tokens are added to place p_i when transition t_j fires,
- $C(i, j) < 0$ indicates that tokens are removed from place p_i,
- $C(i, j) = 0$ implies no direct arc exists between p_i and t_j.

Incidence Matrix represents the net change in tokens in each place when a transition fires. This matrix is often used for mathematical analysis of Petri nets (especially in algebraic approaches).

2.6 Behavioural Properties

Components of Analysis of Petri Net

Analyzing a petri net helps in understanding the behaviour of a system. The key components of petri net analysis include:

1. **Reachability Analysis:** It shows whether a certain marking can be attained from an initial marking M_0. A marking M is reachable from M_0 if there exists a sequence of transition firing that lead to M.
    - Direct Marking- If a marking directly reached from initial marking(Previous marking) then it is called Direct marking.
    - Indirect marking - If a marking is achieved after sequence of markings then it is called as Indirect marking.

A reachability tree is a rooted tree representing all reachable states (or markings) from an initial state within a system. It is constructed by starting with the initial state and recursively exploring all possible next states reachable through transitions or edges, creating a tree where each node represents a state and edges represent transitions.

2. **Boundedness:** If every place P has less than or equal to k number of tokens for every reachable marking, then the petri net is k bounded. In other words if a petri net if k bounded then tokens in each place will not exceed from k. Boundedness ensures resources don't overflow.

3. **Safeness:** A special case of the more general boundedness property is safeness. A place in a Petri Net is safe if the number of tokens in that place never exceeds one. A Petri Net is safe if all the places on the net are safe.

4. **Conservation:** It guarantees that there are always the same number of tokens across all the reachable markings. A Petri Net with initial marking M_0 is strictly conservative if for all markings M':

$$\sum_{P_i \in P} M'(P_i) = \sum_{P_i \in P} M_0(P_i) \tag{2.4}$$

A conservative Petri Net is always bounded but a bounded Petri Net may not be conservative.

5. **Liveness:** A transition is live if it can eventually fire in some reachable marking.If every transition in every reachable marking is live, then the Petri net is said to be live.

   Liveness Levels

   - **L0-liveness**: At least one transition can fire.
   - **L1-liveness**: Each transition can fire at least once.
   - **L2-liveness**: Each transition can fire infinitely often.
   - **L3-liveness**: Each transition will eventually fire.

   Liveness analysis helps detect deadlocks.

6. **Deadlock Analysis:** When no transition is enabled, the system is stuck in a state, which is known as a deadlock.Reachability and liveness analysis help detect and prevent deadlocks.

7. **Coverability:** If a transition sequence can reach a marking with at least as many tokens as M in each place, then the marking M is coverable. Coverability analysis is useful for verifying system safety and checking infinite token accumulation.

8. **Reversibility** A Petri net is reversible if from any reachable marking, it is possible to return to the initial marking. This can be important in systems that require resets or loops.

2.7 Additional Definitions in Petri Net Theory

Pre-set and Post-set

- Pre-set of a transition ($\bullet t$): The set of places from which arcs go into transition t . The pre-set of transition t is defined as:

$$\bullet t \ = \{ \, p \in P \mid (p, t) \ \in F \, \}$$

- Post-set of a transition ($t\bullet$): The set of places to which arcs go from transition t . The post-set of transition t is defined as:

$$t \bullet \ = \{ \, p \in P \mid (t, p) \ \in F \, \}$$

These concepts help define which places are inputs and outputs of a transition.

Firing Sequence

A firing sequence is a series of transitions that fire in a specific order starting from the initial marking M_0. It is denoted by a sequence like:

$$\sigma = t_1 t_2 t_3 ...$$

If this sequence leads to a marking M, then M is said to be reachable via σ

Coverability Tree

A coverability tree is a graphical tool used to analyze:

- Reachability
- Boundedness
- Liveness

In places where tokens can grow indefinitely, the tree uses the symbol ω to indicate an unbounded number of tokens.

Conflict and Concurrency

- **Conflict:** Two or more transitions are in conflict if they share the same input place and their simultaneous firing is not possible.
- **Concurrency:** Two transitions are concurrent if they can be enabled and fired independently at the same time.

This distinction is very important in healthcare systems where, for example, patients may compete for a single doctor (conflict) or multiple diagnostic labs may process patients simultaneously (concurrency).

Sink and Source Places

- **Source place:** A place with only outgoing arcs and no incoming arcs. It represents a starting point in the system (e.g., new patient arrival).
- **Sink place:** A place with only incoming arcs and no outgoing arcs. It typically represents the end of a process (e.g., patient exits after treatment).

2.8 Extensions of Petri Nets

To enhance modeling capabilities, several extensions of classical Petri nets are used in practical applications:

1. **Timed Petri Nets**
    - Incorporate time delays associated with transitions or places.
    - Useful for modeling waiting times, processing times, and delays in OPDs.
2. **Colored Petri Nets (CPNs)**
    - Tokens carry additional information (”colors”) such as patient ID, type, or priority.
    - Enable modeling of more complex and data-dependent systems.
3. **Stochastic Petri Nets (SPNs)**
    - Transitions fire based on probabilistic distributions, suitable for performance evaluation and queuing behavior.

These extensions provide greater modeling power and are particularly valuable in healthcare settings where processes vary with time, priority, and probability.

2.9 Advantages of Using Petri Nets

- **Formalism and Structure:** Petri nets are both graphical and mathematical, allowing for visualization and analysis.
- **Concurrency Handling:** Naturally model simultaneous processes—ideal for hospital scenarios.
- **Model Verification:** Facilitate verification of properties like deadlock, resource usage, and workflow correctness.
- **Simulation Support:** Petri nets can be simulated to evaluate performance and test what-if scenarios.

2.10 Summary

This chapter provided an in-depth understanding of the theoretical framework of Petri nets. It covered the structure, mathematical definitions, transition firing rules, and behavioral properties critical for modeling dynamic systems. The extensions of basic Petri

nets allow for even richer representations of real-world systems like hospital OPDs. In the next chapter, these concepts will be applied to construct a Petri net model for a hospital Outpatient Department.

Chapter 3

Design of the Hospital OPD Model using Petri Nets

3.1 Introduction

In this chapter, we present the Petri net-based model developed to represent the functioning of a hospital Outpatient Department (OPD). The model captures the flow of patients, the interaction between services, and the resources involved. It highlights the concurrency, synchronization, and possible bottlenecks within the OPD system. Based on the theoretical foundation from Chapter 3, this design provides a formal and visual representation of the OPD workflow using places, transitions, and tokens.After that Boundedness, safeness, liveness of the model would be discussed.

3.2 Overview of the OPD process

A typical OPD process involves the following stages:

1. Patient Arrival and Registration
2. Allotment of Doctor
3. Doctor Consultation
4. Diagnostic Tests (Pathology, Radiology, etc.)
5. Pharmacy / Prescription

6. Patient Discharge or Follow-up Scheduling

Each of these stages may involve waiting, resource sharing, and concurrent operations, which make them well-suited for Petri net modeling.

3.3 Assumptions for modelling

To simplify and structure the model, the following assumptions are made:

- One OPD room is modeled.
- A limited number of doctors, diagnostic resources, and registration desks are considered.
- Each patient follows the full cycle:

  Registration $\rightarrow$ Consultation $\rightarrow$ Diagnostics $\rightarrow$ Pharmacy $\rightarrow$ Discharge.
- Hospital can handle only 15 patients at a time.
- Delays are not explicitly modeled here (for untimed PN); however, time-based analysis can be added in future work using Timed Petri Nets.

3.4 Petri Net Model Elements

Petri Net Component	Meaning in OPD Context
Place (circle)	A state (e.g., Waiting for registration)
Transition (bar)	An action or event (e.g., Consultation)
Token (dot)	Represents a patient

Table 3.1: Petri Net Components and their Interpretation in OPD Context

3.5 Components of the Model

Let:

- $P = \{p_1, p_2, \ldots, p_{19}\}$ be the set of Places.
- $T = \{t_1, t_2, \ldots, t_{11}\}$ be the set of Transitions.

Places	**Description**
p_1	Waiting for registration
p_2	Registered patients
p_3	Cardio department
p_4	Gastro department
p_5	ENT department
p_6	Super Senior doctor
p_7	Senior doctor
p_8	Junior doctor
p_9	Treatment complete
p_{10}	Waiting for medical test
p_{11}	Waiting for medical test
p_{12}	Blood test
p_{13}	Urine test
p_{14}	Waiting for report
p_{15}	Result +ve (OK report)
p_{16}	Result -ve (NOT OK)
p_{17}	Waiting for doctor
p_{18}	Patient under treatment
p_{19}	Patient discharge

Table 3.2: Hospital workflow with places

Transitions	Description
t_1	Register Patient
t_2	Assign department
t_3	Assign doctor
t_4	Administer treatment
t_5	Register for medical test
t_6	Assign lab
t_7	Test complete
t_8	Give report
t_9	Deposit report
t_{10}	Assign doctor
t_{11}	Given medicine

Table 3.3: Hospital workflow with transitions

3.6 Model Diagram

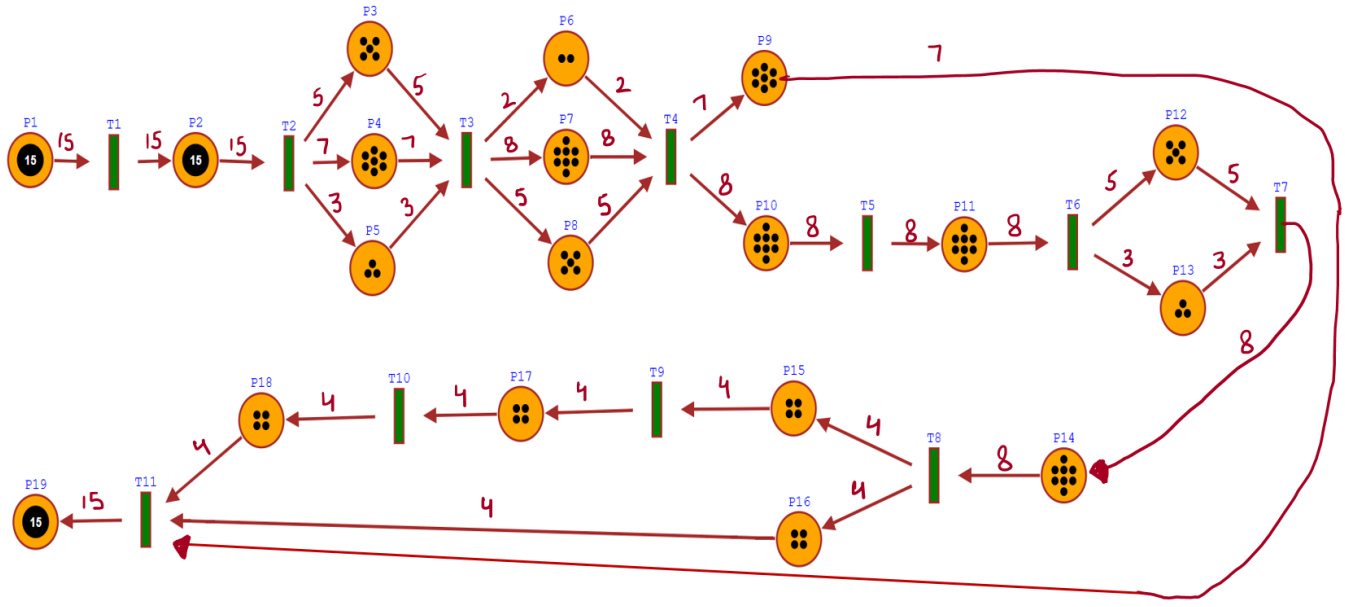


Figure 3.1: Petri Net Diagram

Firing Rule

Let us check the firing rule for this Petri Net.

(a) For Place p_1

$$I^-(p_1, t_1) \leq \mu(p_1)$$

$$15 \leq 15 \quad \text{(True)}$$

$\Rightarrow$ Transition t_1 is enabled for firing.

The marking after firing of t_1 (Tokens in p_2 after firing of t_1):

$$\mu'(p_2) = \mu(p_2) - I^-(p_2, t_1) + I^+(p_2, t_1)$$

$$= 0 - 0 + 15$$

$$= 15$$

(b) For Place p_2

$$I^-(p_2, t_2) \leq \mu(p_2)$$

$$15 \leq 15 \quad \text{(True)}$$

$\Rightarrow$ Transition t_2 is enabled for firing.

Tokens in p_3, p_4, p_5 after firing of t_2:

$$\mu'(p_3) = \mu(p_3) - I^-(p_3, t_2) + I^+(p_3, t_2)$$

$$= 0 - 0 + 5$$

$$= 5$$

$$\mu'(p_4) = \mu(p_4) - I^-(p_4, t_2) + I^+(p_4, t_2)$$
$$= 0 - 0 + 7$$
$$= 7$$

$$\mu'(p_5) = \mu(p_5) - I^-(p_5, t_2) + I^+(p_5, t_2)$$
$$= 0 - 0 + 3$$
$$= 3$$

Similarly, the firing rule can be verified for all transitions, and tokens move according to the firing rule.

For the entire Petri Net, the firing rule has already been verified, and tokens are accordingly distributed.

Description of Model

The proposed model provides a detailed representation of both the structural and dynamic aspects of a general hospital's OPD, offering insights into process efficiency and potential areas for optimization.

In traditional hospital settings, patients are required to wait in long queues for registration, which is both time-consuming and contributes to clogging within hospital premises. To address this issue, an online registration system has been introduced, enabling patients to register a day in advance before visiting the hospital. This system eliminates the need for patients to wait in queue for registration, allowing them to proceed directly to their respective doctors upon arrival. As a result, the implementation of online registration significantly reduces patient waiting times and enhances hospital efficiency. However, in cases where patients either forget to register online or require immediate medical attention without prior planning, they are still permitted to complete their registration physically at the hospital counter. The designated hours for physical registration are from 8:00 AM to 1:00 PM. Timings for treatment is 9:00 AM to 4:00 PM. All activities in the hospital are conducted within the designated timeframe of 8:00 AM to 4:00 PM including registration,and treatment.

3.7 Explanation of Workflow

- p_1 to $p_2 \rightarrow t_1$ (Registration): A token in p_1 represents a new patient. Once registration t_1 occurs, the token moves to p_2, that means patient get registered.

- p_2 to p_3 , p_4 , $p_5 \rightarrow t_2$ (Assignment of Department): A token in p_2 represents registered patients. Once department assigned, the token move from p_2 to p_3 (Cardio),p_4 (Gastro) , p_5 (ENT) according to the need of the patients.

- p_3 , p_4 , p_5 to p_6 , p_7, $p_8 \rightarrow t_3$ (Assign Doctor): After assignment of department, now doctor will be assigned like super senior , senior , junior to patients according to their needs/priority.

- p_6 , p_7, p_8 to p_9 , $p_{10} \rightarrow t_4$ (Administer Treatment): After allotment of doctor, treatment will take place, so patients will undergo treatment here.

- p_9 to $p_{19} \rightarrow t_{11}$ (Given medicine): After treatment, some patients take their medicine and got discharged.

- p_{10} to $p_{11} \rightarrow t_5$ (Register for medical test): After consulting with doctor, some patients need medical test(Lab test) for further investigation. So they start with their registration for medical test.

- p_{11} to p_{12}, $p_{13} \rightarrow t_6$ (Assign Lab): After registration for medical test, patients would wait for either blood or urine test according to the needs of the patients.

- p_{12}, p_{13} to $p_{14} \rightarrow t_7$ (Test complete): Patients have given the blood / Urine sample for their test and now they have been waiting for their reports.

- p_{14} to p_{15} , $p_{16} \rightarrow t_8$(Given report): Now they have got their reports.

- p_{16} to $p_{19} \rightarrow t_{11}$(Given medicine): Some patients have got their OK report and they don't need to consult doctor again, so now they would take the medicine and got discharged from the hospital.

- p_{15} to $p_{17} \rightarrow t_9$(Deposit report): Some patients have NOT OK reports, they would visit doctor once again to show them reports and for further treatment.

- p_{17} to $p_{18} \rightarrow t_{10}$(Assigning doctor): They would again assigned doctor for further treatment.

- p_{18} to $p_{19} \rightarrow t_{11}$(Given medicine): After treatment, patients got their medicine and got discharged from hospital.

3.8 Concurrency and Synchronization in the Model

- Multiple patients can register simultaneously if multiple registration counters are modeled.
- Diagnostic tests and doctor consultations can occur concurrently for different patients.
- Synchronization is required when a single resource (like a doctor) is shared among patients.

3.9 Analysis and Insights from the Model

Using this model, one can analyze:

- **Throughput** : How many patients can be processed in a given time.
- **Bottlenecks** : Identify where delays or queues form.
- **Resource Utilization** : How efficiently the doctor, diagnostics, and pharmacy are used.
- **Reachability** : Can every patient eventually exit (reach p_{19}) from p_1
- **Deadlocks** : Are there any markings from which no further transitions are possible?

These insights are critical for hospital administrators to improve operational efficiency.

3.10 Summary

This chapter presented the complete design of the Petri net model for a hospital OPD, including assumptions, places, transitions, workflow explanation, and concurrency handling. This model lays the foundation for **simulation and performance evaluation**, which may be carried out in subsequent chapters using reachability analysis or software tools.

Chapter 4

Analysis of the Hospital OPD Petri Net Model

4.1 Introduction

After constructing the Petri Net model of the hospital OPD in Chapter 4, this chapter focuses on analyzing the model to evaluate its correctness and performance. Petri net theory provides a robust framework for verifying system properties like safety, liveness, boundedness, and reachability, all of which are crucial in assessing how effectively a hospital OPD can manage its workflow and resources.

This chapter applies qualitative (logical) and quantitative (numerical or structural) analysis methods to the model developed, with the goal of identifying potential bottlenecks, deadlocks, and resource conflicts. The analysis also enables recommendations for improvements in real-world OPD operations.

4.2 Structural Analysis

Structural analysis deals with the inherent properties of the net, independent of the initial marking.

4.2.1 P-Invariants (Place Invariants)

- A P-invariant is a linear combination of places that remains constant throughout the execution of the net. It typically reflects conserved resources.
- In our OPD model, a P-invariant may reflect that the total number of patients in the system (from arrival to exit) remains constant if there is no loss or duplication.Tokens represent patients, and no token is created or destroyed , it only moved from place to place via transitions.

  This invariant expresses that once a patient enters the system, they continue to exist in one of the states (registration, consultation, testing, etc.) until they reach the exit. No patient is lost or duplicated, assuming each transition behaves correctly.

$$M(p_1) + M(p_2) + \cdots + M(p_{19}) = 15$$

4.2.2 T-Invariants (Transition Invariants)

It represents a repeating cycle of activity—a complete process that can be repeated indefinitely.

- A T-invariant is a sequence of transitions that, when fired, leaves the marking of the net unchanged.
- It helps identify repetitive cycles in the system, such as a typical patient's journey through registration, consultation, diagnostics, and exit.
- A complete patient cycle looks like this:

  $t_1 \rightarrow t_2 \rightarrow \cdots \rightarrow t_{11}$

  If you fire all these transitions once in this order, the token (patient):

  1) Moves from arrival through the entire OPD process,

  2) Ends up in the exit place.

  Now, if a new patient arrives, this exact process can repeat, assuming the system resets. Thus: this model is T-invariant. It corresponds to a full, repeatable cycle of one patient entering the system, passing through each stage, and exiting.

4.2.3 Incidence Matrix

Places	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11
P1	-15	0	0	0	0	0	0	0	0	0	0
P2	15	-15	0	0	0	0	0	0	0	0	0
P3	0	-5	0	0	0	0	0	0	0	0	0
P4	0	7	-7	0	0	0	0	0	0	0	0
P5	0	-3	0	0	0	0	0	0	0	0	0
P6	0	0	-2	0	0	0	0	0	0	0	0
P7	0	0	8	-8	0	0	0	0	0	0	0
P8	0	0	-5	0	0	0	0	0	0	0	0
P9	0	0	0	7	-7	0	0	0	0	0	0
P10	0	0	0	8	-8	0	0	0	0	0	0
P11	0	0	0	0	8	-8	0	0	0	0	0
P12	0	0	0	0	0	5	-5	0	0	0	0
P13	0	0	0	0	0	3	-3	0	0	0	0
P14	0	0	0	0	0	0	8	-8	0	0	0
P15	0	0	0	0	0	0	0	4	-4	0	0
P16	0	0	0	0	0	0	0	4	0	-4	0
P17	0	0	0	0	0	0	0	0	4	-4	0
P18	0	0	0	0	0	0	0	0	0	4	-4
P19	0	0	0	0	0	0	0	0	0	0	15

Table 4.1: Incidence matrix of the Petri Net model for Hospital OPD

4.3 Behavioral Analysis

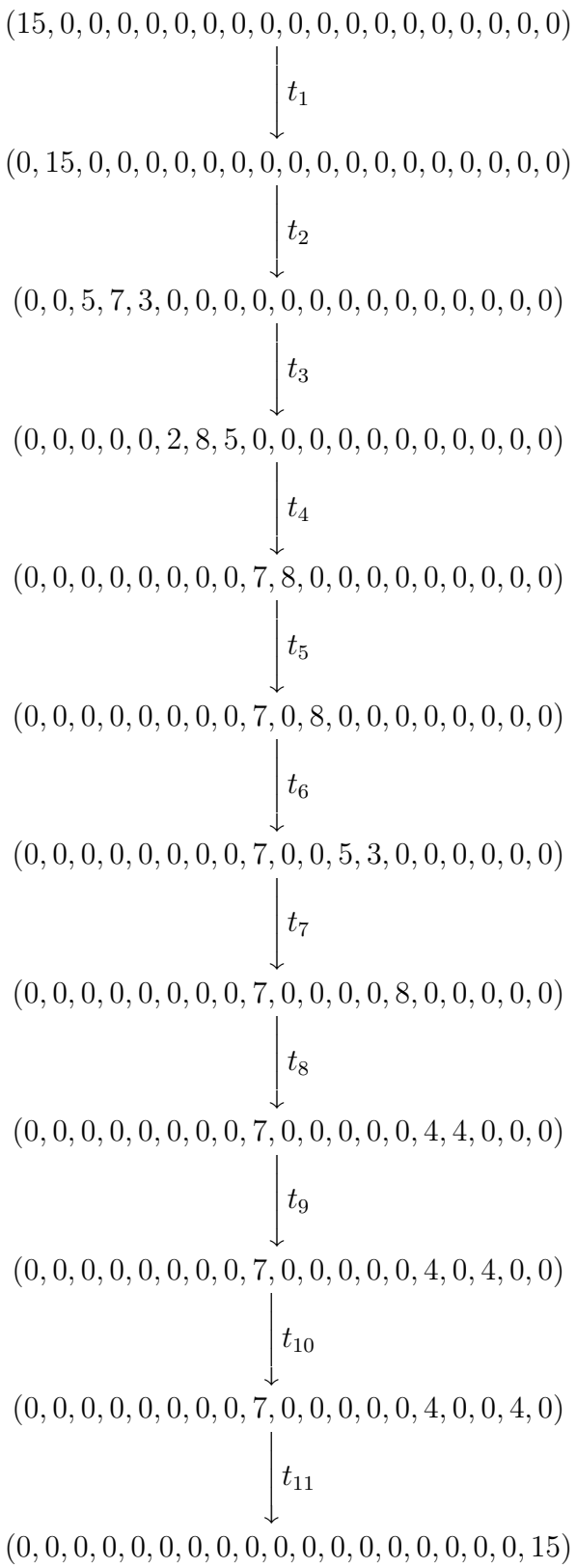
Behavioral analysis studies how the net behaves under various initial markings and firing sequences.

4.3.1 Reachability

A marking M is reachable from the initial marking M_0, if there exists a sequence of transitions that transforms M_0 into M. This helps confirm whether every patient eventually reaches the exit place p_{19}.

The reachability graph (or coverability tree) can be constructed to visualize all possible system states.

Let's make **Reachability Tree** for the Proposed model.


(15, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
t_1
(0, 15, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
t_2
(0, 0, 5, 7, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
t_3
(0, 0, 0, 0, 0, 2, 8, 5, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
t_4
(0, 0, 0, 0, 0, 0, 0, 0, 7, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)
t_5
(0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 8, 0, 0, 0, 0, 0, 0, 0, 0, 0)
t_6
(0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 5, 3, 0, 0, 0, 0, 0, 0, 0)
t_7
(0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, 0)
t_8
(0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 4, 4, 0, 0, 0, 0)
t_9
(0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 4, 0, 4, 0, 0, 0)
t_{10}
(0, 0, 0, 0, 0, 0, 0, 0, 7, 0, 0, 0, 0, 0, 4, 0, 0, 4, 0, 0)
t_{11}
(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 15)

4.3.2 Safeness

Safeness means 1-bounded. However, the Petri Net discussed here is not safe because there are always more than one patient in the hospital. So, there are always more than one token in this Petri Net. Therefore, the Petri Net is not safe.

4.3.3 Boundedness

- A place is k-bounded if the number of tokens in it never exceeds a constant k
- This Petri Net model is 15-bounded because the maximum capacity of the hospital is to handle only 15 patients at a time. However, this model can be extended to k-bounded, where k is a finite positive integer, depending on the availability of resources like doctors, nurses, staff, rooms, etc.

4.3.4 Liveness

- A transition is live if it can potentially fire at some point in the future, no matter the marking.
- A live net ensures that no part of the system becomes permanently idle.
- The hospital Petri Net model is live. Each transition can fire at least once each day. However, the Petri Net cannot fire infinitely because there is a restriction on the time during which patients can register and receive treatment.

4.3.5 Deadlock detection

- A deadlock is a marking from which no transitions are enabled.
- Deadlock scenarios in an OPD could represent patients stuck in waiting due to lack of doctors, broken machines, or unresponsive systems.
- Analysis can identify such markings and help prevent real-world inefficiencies.
- Deadlock does not occur in this Petri Net model because the Petri Net is live. There is always at least one transition that can fire in any of the markings. Hence, no deadlock arises in this Petri Net structure.

4.4 Interpretation and Practical Implications

From the analysis, we derive important conclusions:

- Doctor consultation may be a bottleneck due to single availability.
- Diagnostics can create delays if multiple patients wait for the same resource.
- System is live and safe under moderate patient loads but becomes unsafe or unbounded under peak load conditions unless resources scale accordingly.

These insights can guide administrators to:

- Increase staffing during peak hours
- Digitize registration to reduce delays
- Balance patient load across departments

4.5 Summary

This chapter presented a detailed structural and behavioral analysis of the hospital OPD Petri net model. By examining properties such as boundedness, liveness, reachability, and deadlocks, we assessed how well the model represents real-world performance. The analysis lays the foundation for future improvements or extensions, such as introducing timed or colored Petri nets for deeper insight.

Chapter 5

Simulation and Performance Analysis

5.1 Introduction

Simulation is a crucial part of validating a Petri Net model, especially for a complex and dynamic system like a hospital OPD. The goal of simulation is to test the behavior of the model under various scenarios and confirm that it accurately reflects real-world workflows. In this chapter, the Petri net model developed in previous chapters is simulated using numerical markings (token distributions), and the results are interpreted to understand system behavior, detect potential bottlenecks, and verify correctness.

To evaluate the effectiveness of the Petri Net model, we simulate patient flow through the OPD system under two scenarios: standard load (15 patients) and peak load (25 patients). This analysis demonstrates the model's ability to highlight bottlenecks, test resource allocation, and validate improvements.

5.2 Simulation Environment

- Time frame simulated: 8:00 AM – 4:00 PM (8 hours)
- Number of patients: 15 (standard), 25 (peak)
- Assumptions:
  - Registration takes 2–5 minutes
  - Consultation takes 10–25 minutes depending on doctor type
  - Medical tests take 10–20 minutes per patient

– Report processing takes 5–10 minutes

5.3 Scenario 1: Standard Load (15 Patients)

Initial Marking

The initial marking assumes 15 patients are waiting for registration:

$$M_0 = [15, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]$$

Here, only P_1 (waiting for registration) is marked with 15 tokens.

Departmental Distribution

- Cardiology: 5 patients
- Gastroenterology: 7 patients
- ENT: 3 patients

Average Waiting Times

Table 5.1: Average Waiting Time per Stage (Standard Load)

Stage	**Avg Waiting Time (min)**
Registration	4
Department Assignment	2
Consultation	12
Medical Testing	14
Report Processing	7
Final Treatment	10
Total	**49**

Doctor Utilization

Table 5.2: Doctor Utilization (%) - Standard Load

Doctor Type	Utilization (%)
Super Senior	85
Senior	72
Junior	68

5.4 Scenario 2: Peak Load (25 Patients)

Revised Departmental Distribution

- Cardiology: 9 patients
- Gastroenterology: 10 patients
- ENT: 6 patients

Increased Waiting Times

Table 5.3: Average Waiting Time per Stage (Peak Load)

Stage	Avg Waiting Time (min)
Registration	12
Department Assignment	5
Consultation	25
Medical Testing	25
Report Processing	14
Final Treatment	20
Total	**101**

Bottleneck Identification

- Registration desk gets saturated early

- Consultation queues form due to limited doctor availability
- Labs experience increased test processing delays

5.5 Graphical Representations (Placeholders)

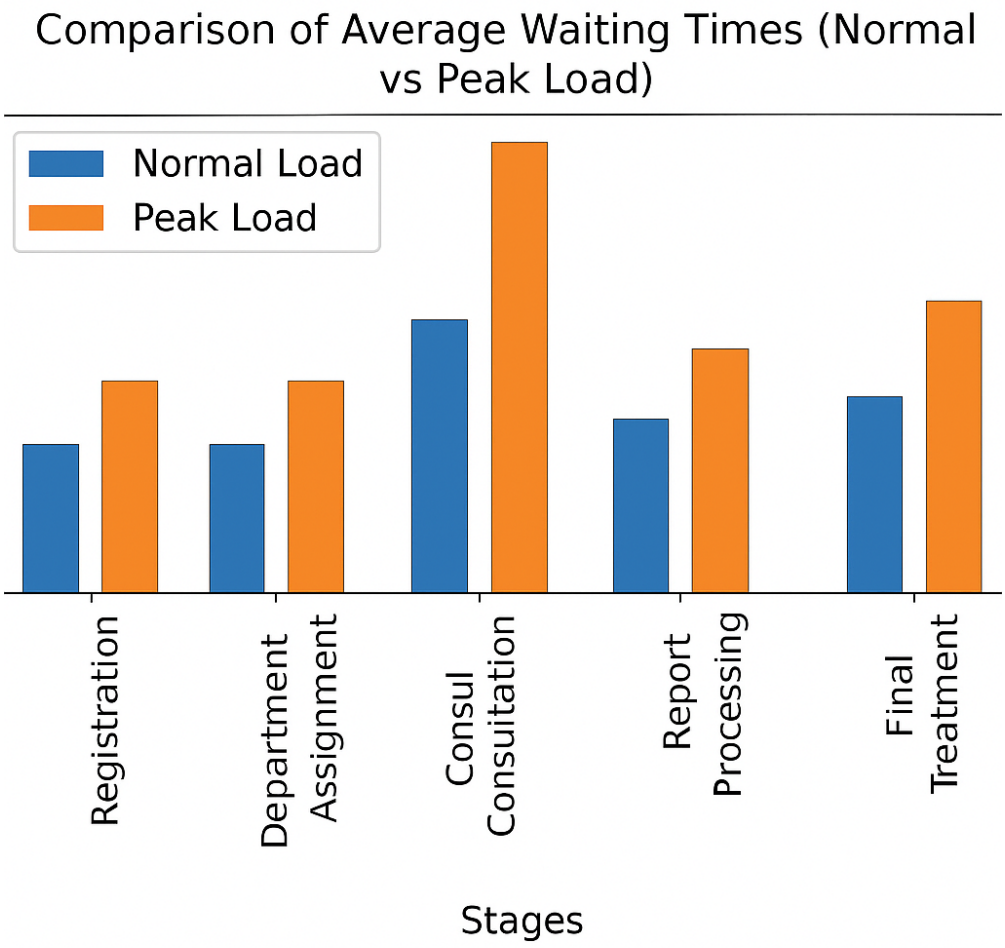


Figure 5.1: Comparison of Average Waiting Times (Normal vs Peak Load)

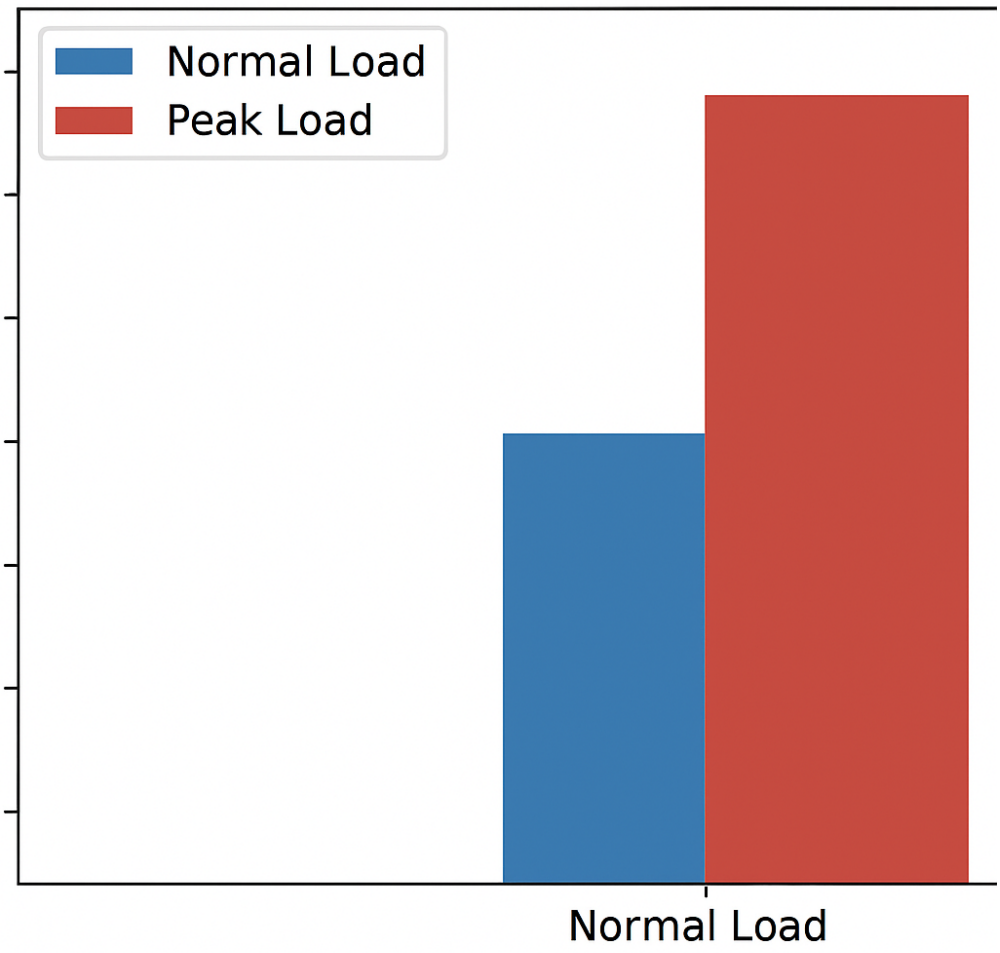


Figure 5.2: Doctor Utilization under Standard vs Peak Load

5.6 Discussion and Insights

The simulation results confirm the following:

- Online registration significantly reduces waiting times.
- Departmental load balancing can optimize consultation queues.
- Lab delays can be reduced by dynamic lab allocation.

The Petri Net model successfully visualizes these scenarios, making it a valuable planning and diagnostic tool.

Chapter 6

Case Study: Urban Multispecialty Hospital OPD

This chapter demonstrates the application of the proposed Petri Net model to a fictional urban multispecialty hospital called ”HealthCareOne”. The hospital has high patient volume and a conventional OPD system handling general medicine, cardiology, ENT, and gastroenterology cases.

6.1 Hospital Background

- Location: New Delhi
- Average OPD Patients per Day: 150
- Departments: Cardiology, Gastroenterology, ENT, General Medicine
- Staff:
  - 3 Super Senior Doctors
  - 6 Senior Doctors
  - 10 Junior Doctors
  - 2 Labs (Blood and Urine)
- Hours of Operation: 8:00 AM to 4:00 PM

6.2 Observed Challenges (Pre-Modeling)

- Patients queue from 7:30 AM to register
- Long wait for doctor allocation, especially for specialists
- Diagnostic labs experience batch-based overload
- Patients often spend 2.5–3 hours for complete OPD visit

6.3 Petri Net Model Implementation

The hospital's OPD flow was modeled using our Petri Net framework. Each of the following stages was translated into places and transitions:

- Registration (online and counter-based)
- Departmental consultation
- Medical testing (blood/urine)
- Report handling and doctor follow-up
- Final treatment and discharge

Initial marking reflected 50 patients across entry points at 8:00 AM. Simulation was run with staggered arrival patterns throughout the morning.

6.4 Pre-Optimization Findings

- Peak congestion at counter registration from 8:00–9:00 AM
- Department assignment delayed due to doctor availability checks
- 25% of patients required tests; labs became a bottleneck from 10:00–12:00 PM
- Patients waited up to 45 minutes for final doctor review after test reports

6.5 Improvements Suggested via Simulation

- Shift to 60% online registration policy
- Pre-assign departments based on prior visit data
- Split lab processing time by adding third technician
- Digital report system for real-time delivery to doctors

6.6 Results After Implementation

Table 6.1: Comparison of Metrics (Before vs After Optimization)

Metric	**Before (min)**	**After (min)**
Registration Queue Time	25	8
Doctor Assignment Delay	15	5
Test Lab Wait Time	30	15
Final Consultation Wait	45	18
Total OPD Visit Duration	150	80

6.7 Conclusion

This case study validates that the Petri Net model can simulate hospital workflows accurately, identify operational inefficiencies, and test corrective strategies before implementation.

Hospitals with similar OPD layouts can adapt this model with minor structural changes and achieve tangible improvements in patient experience and resource utilization.

Chapter 7

Conclusion and Future Work

Conclusion

This thesis has focused on developing and analyzing a Petri net model for the workflow of a hospital's Outpatient Department (OPD). The main objective was to capture the sequential and concurrent processes that take place in a typical hospital setting—from patient registration to final discharge—using the formalism of Petri nets.

The model successfully represented various stages of the OPD such as:

- Patient registration
- Department assignment (Cardiology, Gastroenterology, ENT)
- Doctor allocation (Super Senior, Senior, Junior)
- Treatment and medical testing
- Report generation
- Final discharge

Each of these activities was translated into transitions, while the states (or conditions) such as "waiting for registration", "under treatment", and "waiting for report" were modeled as places.

The model was subjected to both structural and behavioral analyses:

- **Structural analysis** included the construction of the incidence matrix, and verification of P-invariants and T-invariants, which helped confirm the internal consistency and balance of the system. The P-invariants indicated that the total number of patients (tokens) was conserved throughout the transitions, ensuring that patients were not lost or duplicated in the system.
- **Behavioral analysis** was performed using simulation with initial token markings. This simulation validated that the transitions were logically fireable in a realistic hospital scenario and that the flow of tokens closely mirrored actual patient movements in an OPD.

The model was found to be:

- **Bounded**, nsuring that no place received an unmanageable number of tokens (patients), which is crucial for real-life resource constraints.
- **Live**, meaning that there were no deadlocks in the system—every transition could potentially fire under certain conditions, reflecting a continuously operating OPD.
- **Token-conserving**, as the total number of patients in the system remained constant, aligning with the conservation principles in a well-structured workflow.

Limitations

While the model is robust and informative, it includes some simplifications:

- Fixed patient entry (15 or 25) instead of stochastic arrivals
- No integration of emergency or in-patient interactions
- Fixed doctor availability across the time period
- Testing and report times were deterministic

Future Work

While the current model lays a solid foundation for modeling OPD workflows, there are several ways it can be further enhanced:

1. **Incorporating Time Elements**

   The current model does not account for time delays or durations of activities. By extending the model to a Timed Petri Net, one could simulate actual time-based performance metrics like waiting time, treatment duration, and system throughput.

2. **Stochastic Behavior**

   In a real-world hospital, the arrival of patients and the time taken for each activity is not deterministic. Introducing stochastic Petri nets could help model variability and randomness more accurately, making the system more reflective of real scenarios.

3. **Colored Petri Nets (CPNs)**

   To handle more complex patient attributes (e.g., priority levels, symptoms, age), Colored Petri Nets can be used. CPNs allow tokens to carry data (colors), enabling differentiated processing paths within the same model.

4. **Scalability to Full Hospital Systems**

   While this model focused on OPD, it can be scaled to model the entire hospital, including emergency wards, inpatient departments, operation theatres, and pharmacy services.

5. **Resource Allocation Optimization**

   The model can be enhanced to include constraints like doctor availability, lab equipment, or bed capacity. It can then be used to simulate and optimize resource allocation under different load scenarios.

References

[1] Desel J. and Reisig W, Place/Transition Petri Nets. Lectures on Petri Nets I ,Basic Models, Vol. 1491; pp. 122-173, 1998.

[2] Farooq Ahmad, Huang Hejiao and Wang Xiaolong. Analysis of Structure Properties of Petri nets using Transition Vectors, Information Technology Journal Vol. 7, No.2; pp 285-291, 2008.

[3] Peterson JL., Petri Nets, Computing Surveys, Vol. 9, No. 3; pp. 223–252, September 1977.

[4] Peterson JL., Petri Net Theory and Modeling of Systems, Prentice-Hall, 1981.

[5] T. Murata, Petri nets: properties, analysis, and applications, Proceeding of the IEEE, Vol. 77. No.4, April 1989.

Details of Candidate's Publication

The Paper titled "A Petri Net Model of a Hospital OPD" was accepted and presented in Scopus Indexed conference titled "3^{rd} International Conference on Recent Trends in Mathematical Sciences (ICRTMS-2025), held on 10^{th} - 11^{th} May, 2025