

Analysis of Carrier Sense Multiple Access Protocols using High Level Petri Nets

A Dissertation Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Engineering
(Computer Technology and Application)

by

Kailash Gupta
(University Roll No.10188)

Under the Esteemed Guidance of

Mr. Manoj Sethi
Delhi College of Engineering, Delhi



Department of Computer Engineering
Delhi College of Engineering, Delhi -110042
(University of Delhi)

2007-2008

CERTIFICATE



Delhi College of Engineering
(Govt. of National Capital Territory of Delhi)
Bawana Road, Delhi - 110042

Date.....

This is certified that a dissertation on “**Analysis of Carrier Sense Multiple Access Protocols using High Level Petri Nets**” by Kailash Gupta (University Roll no. 10188) as the requirement in the partial fulfillment for the award of degree of Master of Engineering in Computer Technology & Application at Delhi College of Engineering is a bona-fide record work done by him under my supervision and guidance in the academic year 2007-2008.

Mr. Manoj Sethi
(Project Guide)

Department of Computer Engineering
Delhi College of Engineering, Delhi-110042

ACKNOWLEDGEMENT

It is a great pleasure to have the opportunity to extend my heartiest felt gratitude to everybody who helped me throughout the course of this project.

It is distinct pleasure to express my deep sense of gratitude and indebtedness to my learned supervisor Mr. Manoj Sethi for his invaluable guidance, encouragement and patient reviews. His continuous inspiration only has made me complete this dissertation. He is the one who kept on boosting me time and again for putting an extra ounce of effort to realize this work.

I would also like to take this opportunity to present my sincere regards to my teachers Prof. D. R. Choudhary, Dr. Daya Gupta, Mrs. Rajni Jindal, Dr. S.K.Saxena and Mr. Rajeev Kumar for their support and encouragement.

I am grateful to my parents and fiancée Neha Singla for their moral support all the time; they have been always around to cheer me up in the odd times of this work. I am also thankful to my friends Ankur Gupta, Sanjay Kumar and others whose names I haven't been able to mention, for their unconditional support and motivation during this work.

I would like to hear from any readers with comment, suggestions, or bugs reports at kailashgupta1012@yahoo.co.in . Please forward me a copy or patch of changes you made to this version for reviews.

Kailash Gupta
College Roll No. 09/CTA/06
University Roll No. 10188

**Analysis of Carrier Sense Multiple
Access Protocols Using High Level
Petri Nets**

ABSTRACT

Several methodologies have been proposed for modeling but most of them have had limited success in comprehensively incorporating the various aspects of the system. The various modeling techniques differ significantly in the extent to which they provide the ability to model different system perspectives.

Petri net, a combination of the simulation and the mathematical numerical methods can be effectively used for describing and studying information processing systems that are characterized as being parallel, distributed, asynchronous, concurrent, deterministic and/or stochastic.

This dissertation proposes to explore the capabilities of Petri nets, a novel mean of modeling and simulation analysis of a system. As there is hardly any professional, particularly in engineering, that has not been the user of computer communication networks, a good modeling technique will help in better describing and analyzing these systems. The dissertation is an attempt to present models for Open Traffic Information System (OTIS), Aloha protocols with both pure and slotted versions and finally carrier sense multiple access protocols (CSMA) using high level Petri nets.

Based on queueing theory systems and Markov Modulated Poisson arrival processes, Timed Petri nets have been used to analyze and develop Petri net models for various communication protocols. Slotted versions of persistent and non-persistent CSMA and special case of Markov modulated Poisson process also have been discussed. To validate the models and to analyze the behavior of communication protocols a simulation tool HPSim has been used. The results show that communication protocols that typically comprise several generally distributed delays and stochastic processes can be efficiently emulated using Petri net modeling techniques as they have proper constructs to effectively represent all these characteristics.

Table of Contents

ABSTRACT	i
TABLE OF CONTENTS	ii
LIST OF TABLES	iv
LIST OF FIGURES	v
CHAPTER 1 INTRODUCTION	1
1.1 MOTIVATION	2
1.2 OBJECTIVE.....	3
1.3 ORGANIZATION.....	4
CHAPTER 2 PETRI NETS	5
2.1 BASIC DEFINITIONS AND NOTATIONS	5
2.2 TRANSITION RULES.....	7
2.3 INHIBITOR ARC PETRI NETS (EXTENDED PNs)	8
2.4 FIRING SEQUENCE.....	8
2.5 INCIDENCE MATRIX	9
CHAPTER 3 QUEUEING THEORY AND PETRI NET SIMULATION	11
3.1 QUEUEING MODELS.....	11
3.2 KENDALL'S NOTATION FOR CLASSIFICATION OF QUEUE TYPES.....	12
3.3 PERFORMANCE MEASURES	13
3.4 PROBABILITY DISTRIBUTIONS.....	13
3.4.1 Poisson Distribution.....	13
3.4.2 Exponential Distribution.....	14
3.5 LAW OF LARGE NUMBERS	15
3.6 MARKOV CHAIN	15
3.7 MARKOV-MODULATED POISSON PROCESSES	16
3.8 PETRI NET SIMULATION USING HPSIM.....	17
3.8.1 Simulation Stop Condition.....	18
3.8.2 Simulation Output.....	18

CHAPTER 4 HIGH LEVEL PETRI NETS.....	21
4.1 TIMED PETRI NETS.....	21
4.1.1 Policies Associated with Transition Firings	23
4.1.2 Conflicts among Transition Firings	24
4.1.3 Memory Policies	24
4.1.4 Multiple Transition Enabling.....	24
4.1.5 Transition States	25
4.2 TYPES OF NETWORK TRAFFIC.....	26
4.3 SIMULATION OF TIMED PETRI NETS.....	28
4.4 EXAMPLE: MODELING OF OPEN TRAFFIC INFORMATION SYSTEM.....	30
4.4.1 Open Traffic Information System (OTIS)	30
4.4.2 OTIS Model	31
4.4.3 Analysis and Results.....	34
CHAPTER 5 MODELING AND ANALYSIS OF CARRIER SENSE MULTIPLE ACCESS PROTOCOLS.....	35
5.1 TRAFFIC MODEL: ASSUMPTIONS AND NOTATION	36
5.2 MODELING AND ANALYSIS OF COMMUNICATION PROTOCOLS.....	38
5.3 ALOHA PROTOCOLS	39
5.3.1 Pure Aloha	39
5.3.2 Slotted Aloha	46
5.4 CARRIER SENSE MULTIPLE ACCESS PROTOCOL.....	52
5.4.1 Non-Persistent CSMA	55
5.4.2 1-Persistent CSMA	64
5.4.3 p-Persistent CSMA	71
CHAPTER 6 CONCLUSION AND FUTURE WORK.....	78
CHAPTER 7 PUBLICATIONS.....	80
REFERENCES.....	81
GLOSSARY OF NOTATION	83

List of Tables

Table 4.1:	Explanation of Petri Net Model of OTIS.....	33
Table 5.1:	Description of Various Notations used in the Petri Net Simulation.....	38
Table 5.2:	Explanation of the Petri Net Model of Pure Aloha.....	41
Table 5.3:	Description of the Transitions in the Petri Net Model of Pure Aloha	41
Table 5.4:	Explanation of the Petri Net Model of Slotted Aloha.....	48
Table 5.5:	Transition Description in the Petri Net Model of Slotted Aloha	48
Table 5.6:	Explanation of the Petri Net Model of CSMA Protocol.....	54
Table 5.7:	Description of the Transitions in the Petri Net Model of CSMA.....	55
Table 5.8:	Explanation of the Petri Net Model of NP-CSMA.....	58
Table 5.9:	Description of the Transitions in Petri Net Model of NP-CSMA	58
Table 5.10:	Explanation of the Petri Net Model of Persistent CSMA.....	67
Table 5.11:	Description of the Transitions in the Petri Net Model of 1P-CSMA	69
Table 5.12:	Transition Description of the Petri Net Model of p Persistent CSMA.....	73
Table 5.13:	Explanation of the Petri Net Model of p-Persistent CSMA	73

List of Figures

Figure 2.1: Petri Net Model of a Critical Section	7
Figure 2.2: Transformation of a Self-loop to a loop.....	8
Figure 3.1: Basic Queueing Model.....	11
Figure 3.2: Two State MMPP	17
Figure 3.3: Simulator HPSim	17
Figure 3.4: Setting Simulation Properties in HPSim.....	18
Figure 4.1: Transition State Diagram	25
Figure 4.2: Poisson Arrival Processes in Petri Nets	26
Figure 4.3: Two state MMPP Arrival.....	27
Figure 4.4: Two State MMPP Retrial Arrival	27
Figure 4.5: Data Flow Diagram of OTIS.....	31
Figure 4.6: Petri Net Model of OTIS.....	32
Figure 5.1: Pure Aloha: Packet Timing.....	40
Figure 5.2: Petri Net Model of Pure Aloha.....	40
Figure 5.3: Petri Net Model of Pure Aloha: the channel is free and packet transmission will not result a collision.....	42
Figure 5.4: Petri Net Model of Pure Aloha: a new packet arrive before completion of transmission of previous packet, which results in a collision.....	43
Figure 5.5: Petri Net Model of Pure Aloha: a new packet request for transmission before the collision is resolved.....	44
Figure 5.6: Petri Net Model of Pure Aloha: finite population.....	45
Figure 5.7: Petri Net Model of Pure Aloha (Finite Population): packet transmitted by second node (T12) will collide with the packet sent by first node (T1) ...	45
Figure 5.8: Slotted Aloha: Packet Timing	46
Figure 5.9: Throughput(S) - Offered load (G) of Pure and Slotted Aloha	47
Figure 5.10: Petri Net Model of Slotted Aloha Protocol	47
Figure 5.11: Petri Net Model of Slotted Aloha: beginning of the new slot, packet can be transmitted.....	50

Figure 5.12: Petri Net Model of Slotted Aloha: more than one packet scheduled for transmission in previous slot, both will be transmitted and will result in collision.....	51
Figure 5.13: Petri Net Model of Slotted Aloha: finite population.....	52
Figure 5.14: Petri Net Model of CSMA Protocol.....	53
Figure 5.15: Non-persistent CSMA: Packet Timing	56
Figure 5.16: Petri Net Model of NP-CSMA	57
Figure 5.17: Petri Net Model of NP-CSMA: Channel is in free state, packet can be transmitted.....	60
Figure 5.18: Petri Net Model of NP-CSMA: If a new packet arrives in the vulnerable period it will sense channel in free state and will interfere with the current packet in transmission.....	60
Figure 5.19: Petri Net Model of NP-CSMA: channel is sensed busy, reschedule the packet	61
Figure 5.20: Petri Net Model of NP-CSMA with Rescheduling Mechanism	62
Figure 5.21: Petri Net Model of NP-CSMA: if the channel is sensed busy Packet will not be transmitted.....	62
Figure 5.22: Petri Net Model of NP-CSMA: packet is rescheduled after a random period of time when channel is sensed busy.	63
Figure 5.23: Petri Net Model of Slotted NP-CSMA.....	64
Figure 5.24: Persistent CSMA: Packet Timing	65
Figure 5.25: State Transitions of 1P CSMA.....	66
Figure 5.26: Petri Net Model of Persistent CSMA.....	67
Figure 5.27: Petri Net Model of 1P- CSMA: channel is in free state, new transmission can take place	69
Figure 5.28: Petri Net Model of Persistent CSMA: start of the vulnerable period, if a new packet arrives it will sense channel free and will result in collision .	70
Figure 5.29: Petri Net Model of Persistent CSMA: two or more packets, waiting for transmission, will result in collision.	70
Figure 5.30: Petri Net Model of p-Persistent CSMA	72
Figure 5.31: Petri Net Model of Persistent CSMA Modeled as Markov Modulated Poisson Process Arrivals.....	74

Figure 5.32: Petri Net Model of Persistent CSMA: start of idle period, no new request can arrive until there is an unsent packet in output buffer.	74
Figure 5.33: Petri Net Model of Persistent CSMA: finite population	75
Figure 5.34: Petri Net model of Persistent CSMA: Slotted version	76
Figure 5.35: Petri Net Model of Persistent CSMA(slotted version): beginning of the new slot, packet can be transmitted	77

Chapter 1

Introduction

System design and modeling always has been a point of attention in scientific, engineering and industrial applications. System modeling techniques, using diagrams and pictorial representations, enable improved understanding and communication of the information to both the end users and the developers. The effectiveness and efficiency of the modeling technique greatly determines the quality of the implementation of a system.

Petri nets offer a mathematically defined technique for the specification, design, analysis, verification and performance evaluation of a system with varied characteristics. In spite of being able to represent large system complexities, Petri nets use a limited number of symbols that make them fairly straightforward to employ. Petri nets have powerful abilities for representation of system dynamics like - entity arrival dynamics, resource availability, resource interdependency, start and termination of activities, queueing time, queue length, event firing conditions and other control mechanisms.

It allows simulation of the system to measure performance characteristics and test results. Petri net tools for formal analysis and simulation of system are widely available. Such an advantage of closely tying the system model with its simulation is a major benefit that Petri nets have over other modeling techniques.

Due to the generality and permissiveness inherent in Petri nets, they have been proposed for a very wide variety of applications. One of the application areas where Petri nets can be successfully used is computer communication

networks. Today, there is hardly any professional, particularly in engineering, that has not been the user of such a network. This proliferation requires the thorough understanding of the behavior of networks by those who are responsible for their operation as well as by those whose task is to design such networks. This is probably the reason for the large number of books, monographs, and articles treating relevant issues, problems, and solutions.

Three major components characterize computer communication networks: switches, channels and protocols. The switches (or nodes) are the hardware entities that house the data communication functions; the protocols are the sets of rules and agreements among the communicating parties that dictate the behavior of the switches, and the channel is the physical medium over which signals, representing data, travel from one switch to another.

To make a transmission successful interference must be avoided or at least controlled. The channel then becomes the shared resource whose allocation is critical for proper operation of the network. *Multiple Access Protocols* are nothing but channel allocation schemes that possess desirable performance characteristics.

Petri nets that are combination of simulation and mathematical numerical methods can provide a greater environment to analyze these communication protocols as they capture all the elements important for process dynamics and system behavior presentation. The great advantage of simulation of Petri net models lies in transparency, availability and ease of making changes in configuration.

1.1 Motivation

For the analysis of communication networks, Kleinrock [1] mentions several directions: mathematical analysis, which yields explicit performance expressions or which yields a numerical evaluation procedure, simulation or measure the performance by building the system. Simulation which is a combination of programming and modeling is highly used for analysis purposes.

The basic Idea behind simulation is to acquire knowledge [2] and to reach some informed decisions regarding a real world system. But the system is not easy to study directly, so one has to proceed indirectly by creating and studying another entity (the simulation model), which is sufficiently similar to the real world system and ensure that the facts learned about the model will also be true for the system.

Several methodologies have been proposed for modeling [2], [3] but most of them have had limited success in comprehensively incorporating the various aspects of the system. For example Data Flow Diagrams [3] have long been popular for design representation of a system. But the modeling remains incomplete due to the absence of any control flows. Most of the modeling techniques are unable to cope with stochastic elements that are so frequent to real world problems.

However Petri net, a combination of the simulation and the mathematical numerical methods [4], [5] can be effectively used for describing and studying systems that are characterized as being parallel, distributed, asynchronous, concurrent, deterministic and/or stochastic. Petri nets are one of the most widely used methods in modeling because of their characteristics: simplicity, representation power comprising concurrency, synchronization and resource sharing.

Petri nets have been used due to their advantage in quick construction and numerical analysis. They make possible to describe precisely and unambiguously detailed view of large and complex systems. Also Petri net models are flexible, scalable and easy to analyze.

1.2 Objective

Based on the above, it has been seen that several modeling methodologies have been proposed but most of them have had limited success in comprehensively incorporating the various aspects of a system. However Petri nets are effective to emulate a system with varied characteristics. This dissertation proposes to explore the capabilities of Petri nets, a novel mean of

modeling and simulation analysis of a system. As information processing systems and communication networks have a great role in scientific, engineering and industrial applications, a good modeling technique will help in better describing and analyzing these systems. The dissertation is an attempt to present models for Open Traffic Information System (OTIS), Aloha protocols with both pure and slotted versions and finally carrier sense multiple access protocols (CSMA) using high level Petri nets. Based on queueing systems and Markov modulated Poisson arrival processes, the dissertation proposes to develop timed Petri net models for persistent and non-persistent CSMA, their slotted versions, and also persistent CSMA as a special case of Markov modulated Poisson process by using primitives provided by Petri nets.

1.3 Organization

The rest of the dissertation is organized as follows: Chapter 2 provides a brief introduction to Petri nets and gives a basis for Petri nets abbreviation necessary for analysis of Petri net models presented in consequent Chapters. Chapter 3 discusses important fundamental result from queueing theory and statistics, Markovian arrival processes required for the modeling and analysis purpose. Chapter 4 discusses High Level Petri Nets and their application in information processing systems and explains the Petri net model of Open Traffic Information System (OTIS). Chapter 5 develops Petri net models of Aloha and carrier sense multiple access protocols and analyze them using simulator HPSim. Chapter 6 concludes the dissertation and gives some research directions. Chapter 7 lists the paper that has been published during the preparation of the dissertation. In last, the list of references is given which we have gone through during the project.

Chapter 2

Petri Nets

Petri net, introduced by C.A.Petri in 1962 [4], [5] is a method which enables graphical modeling of system behavior while simultaneously enabling introduction of mathematical formal rules for system behavior definition.

Petri nets are one of the most widely used methods for modeling of concurrent, asynchronous, distributed, parallel, nondeterministic and/or stochastic systems. They have been proposed for a very wide variety of applications. This is due to the generality and permissiveness inherent in Petri nets.

This chapter discusses basic definitions, special kinds and behavior of Petri nets, introductory modeling examples to give a basis for Petri nets abbreviation presented in consequent chapters.

2.1 Basic Definitions and Notations

A Petri net (PN) is a particular kind of directed graph [4], [6], together with an initial state called the *initial marking* M_0 . The underlying graph of a Petri net is a directed, weighted, bipartite graph consisting of two kinds of nodes, called *places* and *transitions*, where arcs are either from a place to a transition or from a transition to a place.

In graphical representation, places are drawn as circles, transitions as bars or rectangles. Arcs are labeled with their weights (positive integers), where a k -weighted arc can be interpreted as the set of k parallel arcs. Labels for unit weight are usually omitted.

A marking (state) assigns to each place a non negative integer. If a marking assigns to place P a non negative integer k , then P is said to be marked with k *tokens*. Pictorially, k black dots (tokens) are placed in place P . A marking is denoted by M , a $|\mathbf{P}|$ -vector, where $|\mathbf{P}|$ is the total number of places. The i th part of M , denoted by $M(P_i)$ represents the number of tokens in place P_i .

A transition (an event) has a certain number of input and output places representing the pre-conditions and post conditions of the event, respectively.

Definition 2.1: Formal Definition of a Petri Net [4]

A Petri net is a 5-tuple, $PN = (\mathbf{P}, \mathbf{T}, \mathbf{F}, \mathbf{W}, \mathbf{M}_0)$ where:

$\mathbf{P} = \{P_1, P_2, \dots, P_n\}$ is a finite set of places,

$\mathbf{T} = \{T_1, T_2, \dots, T_m\}$ is a finite set of transitions,

$\mathbf{F} \subseteq (\mathbf{P} \times \mathbf{T}) \cup (\mathbf{T} \times \mathbf{P})$ is a set of arcs (flow relation),

$\mathbf{W}: \mathbf{F} \rightarrow \{1, 2, 3, \dots\}$ is a weight function,

$\mathbf{M}_0: \mathbf{P} \rightarrow \{0, 1, 2, 3, \dots\}$ is the initial marking,

A Petri net structure $PN = (\mathbf{P}, \mathbf{T}, \mathbf{F}, \mathbf{W})$ without any specific initial marking is denoted by N . A Petri net with the given initial marking M_0 is denoted by (N, M_0) .

The following symbols are used for input and output sets of a place P and a transition T (\mathbf{F} is the set of arcs):

$\bullet T = \{P \mid (P, T) \in \mathbf{F}\}$ = the set of input places of T

$T^\bullet = \{P \mid (T, P) \in \mathbf{F}\}$ = the set of output places of T

$\bullet P = \{T \mid (T, P) \in \mathbf{F}\}$ = the set of input transitions of P

$P^\bullet = \{T \mid (P, T) \in \mathbf{F}\}$ = the set of output transitions of P

This notation can be extended to a subset. For example, let $S_1 \subseteq \mathbf{P}$, then $\bullet S_1$ is the union of all $\bullet P$ such that $P \in S_1$.

In the next section the rule for transition enabling and firing are given. Although this rule appears very simple, its implication in Petri-net theory is very deep and complex.

2.2 Transition Rules

In order to simulate the dynamic behavior of a system, a state or marking in a Petri net is changed according to the following transition (firing) rules [6]:

1. A transition T is said to be enabled if each input place P of T is marked with at least $w(P,T)$ tokens, where $w(P,T)$ is the weight of the arc from P to T .
2. An enabled transition may or may not fire (depending on whether or not the event actually takes place).
3. A firing of an enabled transition T removes $w(P,T)$ tokens from each input place P of T , and adds $w(T, P)$ tokens to each output place P of T , where $w(T, P)$ is the weight of the arc from T to P .
4. The marking of the other places which are neither input nor output of T remains unchanged.

A transition without any input place is called a source transition, and one without any output place is called a sink transition. Note that a source transition is unconditionally enabled, and that the firing of a sink transition consumes tokens, but does not produce any.

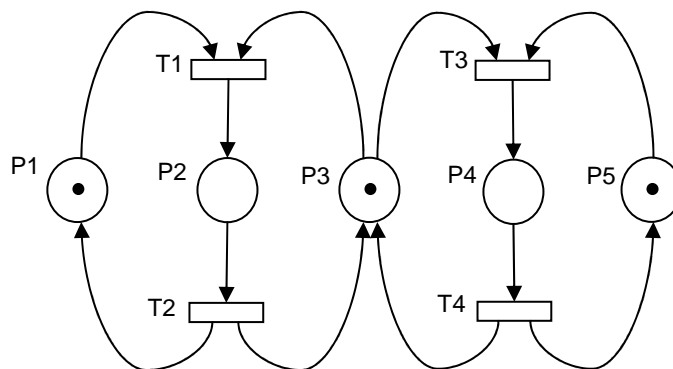


Figure 2.1: Petri Net Model of a Critical Section

A Petri Net model of two processors and critical section is described in figure 2.1. Transitions $T1$, and $T3$ represent entering the critical section and place $P3$ represents semaphore. Transitions $T1$ and $T3$ are in conflict, because the transition $T3$ will be disabled if the left processor enters the critical section and

vice versa. Places P2 & P4 represent autonomous processing of each processor. Transitions T2 & T4 represent leaving the critical section.

Two events e_1 and e_2 are in *conflict* if either e_1 or e_2 can occur but not both and they are *concurrent* if both events can occur in any order without conflicts. A situation where conflict and concurrency are mixed is called *confusion*.

A pair of a place P and a transition T is called a *self-loop* if P is both an input and output place of T. A Petri net is said to be *pure* if it has no self-loops. A Petri net is said to be *ordinary* if all of its arc weights are 1's. Note that a self-loop can be refined or transformed into a loop by introducing a dummy pair of a transition and a place, as is illustrated in FIGure 2.2.

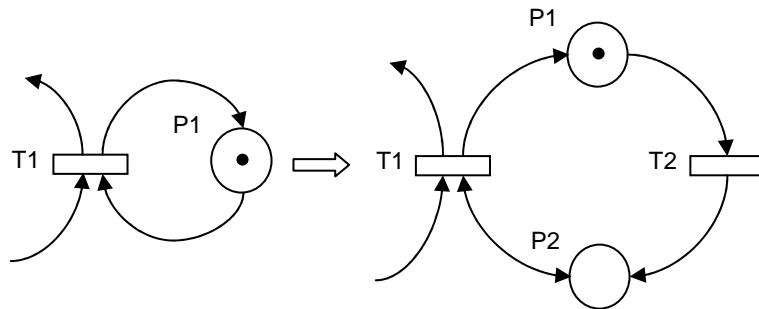


Figure 2.2: Transformation of a Self-loop to a loop.

2.3 Inhibitor Arc Petri Nets (Extended PNs)

The *inhibitor arc* was introduced to Petri nets to allow testing of zero. The inhibitor arc disables the transition when input place has tokens and enables it when the input place has no token [4], [5]. Graphically, an inhibitor arc is a dashed line, which ends with an empty circle.

2.4 Firing Sequence

A *firing sequence* is an ordered set of transition firings, which has associated a *characteristic vector* \vec{S} (also the *firing count vector* {Murata [4 pp. 552]}). i^{th} part of \vec{S} indicates the number of times the transition T_i was fired in the sequence. From marking in figure 2.1 one can have, for example, firing sequence A=T1 T2, B=T3 T4 T3 T4, or C=T1 T2 T1 T2 T3 T4 T1 T2. The firing vectors

associated to the firing sequences are: $\vec{S}_A = (1,1,0,0)$; $\vec{S}_B = (0,0,2,2)$; $\vec{S}_C = (3,3,1,1)$.

A characteristic vector may correspond to several firing sequences. For example $(1,1,1,1)$ corresponds both to $T_1 T_2 T_3 T_4$, and $T_4 T_1 T_3 T_2$. But not all the characteristic vectors \vec{S} whose parts are positive or zero integers are possible: For example there is no firing sequence with characteristic vector $\vec{S} = (0,1,1,1)$ reachable from the initial marking because the transition T_2 can not be fired before firing of T_1 .

2.5 Incidence Matrix

Let a Petri net N with $P = P_1, \dots, P_m$ and $T = T_1, \dots, T_n$ be given. A Matrix $A = (a_{ij})$ where $(1 \leq i \leq m, 1 \leq j \leq n)$ is called the *incidence matrix* of N if and only if:

$$(2.1) \quad \mathbf{A} = \mathbf{A}^+ - \mathbf{A}^-$$

where
$$a_{ij} = a_{ij}^+ - a_{ij}^-$$

Where a_{ij}^- is the weight of the arc from place P_i to transition T_j and a_{ij}^+ is the weight of the arc from transition T_j to place P_i . \mathbf{A}^- is Pre-condition matrix of size $[|P|, |T|]$ having a_{ij}^- as (i,j) th element, and \mathbf{A}^+ is Post-condition matrix of size $[|P|, |T|]$ having a_{ij}^+ as (i,j) th element. E.g. following are the pre-condition, post-condition and incidence matrix for the Petri net of figure 2.1:

$$\mathbf{A}^- = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \mathbf{A}^+ = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{A} = \begin{bmatrix} -1 & 1 & -1 & 0 \\ 1 & -1 & 0 & 0 \\ -1 & 1 & -1 & 1 \\ 0 & 0 & 1 & -1 \\ 0 & 0 & -1 & 1 \end{bmatrix}$$

It is easy to see from the transition rules described before that Pre-condition, Post-condition and incidence matrix respectively represent the number of tokens removed, added and changed in any place.

Sometimes incidence matrix is also represented as difference of pre-condition matrix and post condition matrix (i.e. $\mathbf{A} = \mathbf{A}^- - \mathbf{A}^+$). In that case incidence matrix represents the difference of number of tokens consumed and tokens fired by a transition.

Chapter 3

Queueing Theory and Petri Net Simulation

In this chapter some important fundamental results from queueing theory, Probability distributions and Markovian Processes which forms the basis for modeling the communication networks using Petri nets have been discussed. Also, the simulation properties of simulator HPSim used for analysis of communication protocols have been explained.

3.1 Queueing models

The basic queueing model is shown in figure 3.1. It can be used to model, e.g., machines or operators processing orders or communication equipment processing information. Depending on the specific network model or device operation, the queueing system has an arrival process, a departure process and a recycle process, a queue size and a service discipline [7].

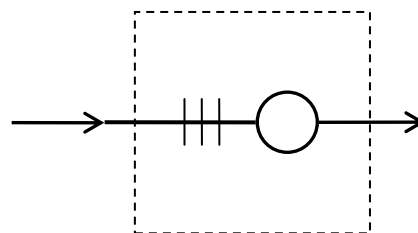


Figure 3.1: Basic Queueing Model

In the basic model, customers arrive one by one and they are always allowed to enter the system, there is always room, there are no priority rules and customers are served in order of arrival.

Following are some factors using which a queueing model is characterised:

- The arrival process of customers
- The behavior of customers
- The service times
- The service discipline
- The service capacity
- The waiting room

There is a shorthand notation introduced by Kendall to characterize a range of these queueing models described in the next section. .

3.2 Kendall's Notation for Classification of Queue Types

There is a standard notation for classifying queueing systems into different types. This was proposed by D.G.Kendall [7]. Systems are described by five part code A / B / C / D / E

where:

- A : Distribution of inter arrival times of customers
- B : Distribution of service times
- C : Number of servers
- D : Maximum total number of customers which can be accommodated in the system
- E : Calling population size

A and B can take any of following distribution types:

- M : Exponential Distribution (Memory Less)
- E_k : Erlang Distribution (k = shape parameter)
- G : General Distribution (arbitrary distribution)

Some examples are M/M/1, M/M/c, M/G/1, G/M/1 and M/D/1. E.g., M/M/m/K/N - would describe a queueing system with an exponential distribution for the inter arrival times of customers and the service times of customers, m servers, a

maximum of K customers in the queueing system at once, and N potential customers in the calling population.

3.3 Performance Measures

The whole idea behind performance analysis of a queueing system is to quantify the network behavior in terms of some meaningful numerical measures. Relevant performance measures in the analysis of queueing models are:

- The distribution of the waiting time and the sojourn time of a customer. The sojourn time is the waiting time plus the service time.
- The distribution of the number of customers in the system (including or excluding the one or those in service).
- The distribution of the busy period of the server. This is a period of time during which the server is working continuously.
- The distribution of the amount of work in the system. That is the sum of service times of the waiting customers and the residual service time of the customer in service.

3.4 Probability Distributions

This section discusses various probability distribution required for the statistical analysis of Petri net models with stochastic elements and to understand modeling of various arrival processes (network traffic) in communication protocols.

3.4.1 Poisson Distribution

The Poisson distribution arises in many situations. It is one of the most important discrete probability distributions. Consider a random variable X , which counts the number of occurrences happening randomly in a given time interval. Let occurrence rate is given by λ and is assumed to be independent in two non-

overlapping time intervals. Then the distribution of random variable X is given by equation 3.1 [8]

$$(3.1) \quad P(X = k) = \frac{\lambda^k}{k!} e^{-\lambda}, \quad k = 0, 1, 2, \dots$$

The above distribution is the Poisson distribution for which it holds that

$$(3.2) \quad E(X) = \sigma^2(X) = \lambda$$

where $E(X)$ and $\sigma^2(X)$ are expected value and variance of the random variable X respectively.

For example consider the occurrences of incoming telephone calls to a police station in a large city. Now to have a hope of computing the probabilities of events such as more than 10 phone calls occurring in a 5-minute time interval, it must be assumed that the average rate i.e. the average number of occurrences per minute is a constant. This is the rate which is denoted by λ above (Thus, in a given 5-minute time interval, there are about 5λ occurrences).

Presumably, in this example, there would be more incoming calls between 6 to 7 P.M. than between 4 to 5 A.M., and this fact would certainly affect the above probability. So now two different rates for the two time periods given above would be used thereby obtaining two different probabilities for the given event.

3.4.2 Exponential Distribution

The density of an exponential distributed random variable X with parameter λ [8] is given by

$$(3.3) \quad f(x) = \begin{cases} \lambda e^{-\lambda x}, & 0 \leq x < \infty \\ 0, & \text{otherwise} \end{cases}$$

Here λ is any positive constant, depending on the experiment. The exponential density is often used to describe experiments involving a question of the form: How long until something happens? For example, the exponential density is often used to study the time between emissions of particles from a radioactive source.

If $x \geq 0$, then the distribution function equals

$$(3.4) \quad F(x) = P[X \leq x] = 1 - e^{-\lambda x}$$

Also expected mean and variance are given by

$$(3.5) \quad E(X) = 1/\lambda, \quad \sigma^2(X) = 1/\lambda^2$$

There is a very important relationship between the exponential density and the Poisson distribution [9]. Suppose X is a Poisson random variable, with parameter λ , so that the expected number of occurrences per unit time is equal to λ . Then the random variable T representing the time between successive occurrences follows an exponential distribution with parameter λ .

3.5 Law of Large Numbers

Let X_1, X_2, \dots, X_n be an independent trials process with finite expected value $\mu = E(X_j)$ and finite variance $\sigma^2 = V(X_j)$. Let $S_n = X_1 + X_2 + \dots + X_n$

Then for any $\varepsilon > 0$,

$$(3.6) \quad P\left(\left|\frac{S_n}{n} - \mu\right| \geq \varepsilon\right) \rightarrow 0 \quad \text{as } n \rightarrow \infty$$

Equivalently

$$(3.7) \quad P\left(\left|\frac{S_n}{n} - \mu\right| < \varepsilon\right) \rightarrow 1 \quad \text{as } n \rightarrow \infty$$

For example consider the special case of tossing coin n times with S_n the number of heads that turn up. Then the random variable S_n/n represents the fraction of times heads turns up and will have values between 0 and 1. The Law of Large Numbers predicts that the outcomes for this random variable will, for large n , be near 1/2.

3.6 Markov Chain

When a sequence of chance experiments forms an independent trials process, the possible outcomes for each experiment are the same and occur with the

same probability. Further, knowledge of the outcomes of the previous experiments does not influence our predictions for the outcomes of the next experiment.

In 1907, A. A. Markov began the study of an important new type of chance process [9]. In this process, the outcome of a given experiment can affect the outcome of the next experiment. For example, this should be the case in predicting a student's grades on a sequence of exams in a course. This type of process is called a Markov chain.

A Markov chain is described as follows: $S = \{s_1, s_2, \dots, s_r\}$ is a set of *states*. The process starts in one of these states and moves successively from one state to another. Each move is called a *step*. If the chain is currently in state s_i , then it moves to state s_j at the next step with a probability denoted by p_{ij} and this probability does not depend upon which states the chain was in before the current state. The probabilities p_{ij} are called *transition probabilities*.

Markov chain can be described as a frog jumping on a set of lily pads. The frog starts on one of the pads and then jumps from lily pad to lily pad with the appropriate transition probabilities.

3.7 Markov-Modulated Poisson Processes

A *Markov Modulated Poisson Process* (MMPP) of order m consists of an m *state* Markov Chain in which each state s_i ($i = 1, 2, \dots, m$) is associated with a Poisson process of rate (λ_i) . In other words, MMPP is a time varying Poisson process, the rate of which is changed (modulated) according to the Markov chain. The Markov chain can therefore be said to modulate the Poisson process, hence the name. This modulation introduces correlations between successive inter arrival times in the process. The MMPP can be identified as a special case of the *Markovian arrival process* (MAP).

An example of MMPP is a process where requests arrive with different rates in different time periods. An MMPP can be classified by the number of states the modulating Markov chain contains, e.g. a Markov chain with two states and

two (different) intensities is denoted as MMPP-2, or sometimes Switched Poisson Process (SPP).

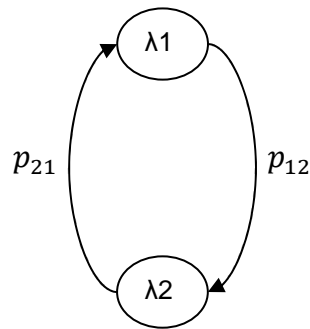


Figure 3.2: Two State MMPP

If the two intensities are equal the model becomes an ordinary Poisson process. The special case when one intensity is zero is called Interrupted Poisson process (IPP).

3.8 Petri Net Simulation Using HPSim

HPSim supports the design and simulation of high level Petri nets, both in a graphical and intuitive manner [10].

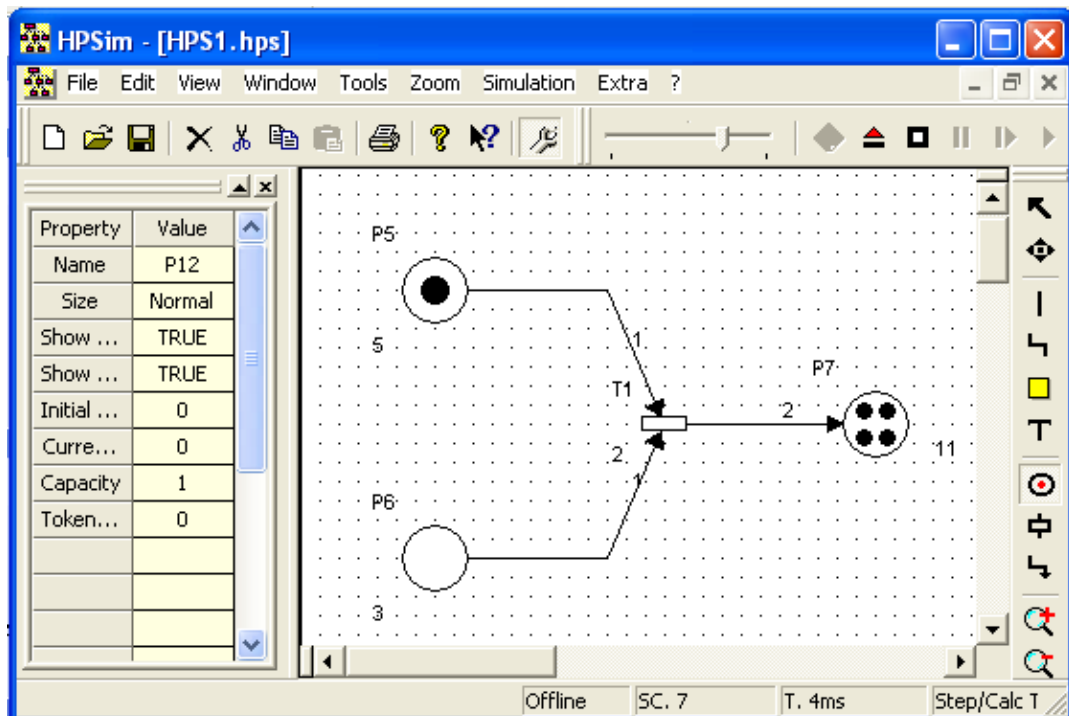


Figure 3.3: Simulator HPSim

The actual simulation is visualized as a Token Game Animation. This can be executed in single step or continuous mode. Besides that, a fast forward mode is also available, in which the graphical representation of the token is not synchronized with the simulated actual position.

3.8.1 Simulation Stop Condition

An important issue in simulations is the selection of stop condition, because correctly selected stop condition saves the computational power. The simulation has to be long enough to ensure the confidence of simulated results, but the simulation must stop in reasonable time. The simulation stop condition should be exact, but its evaluation should be fast and simple. To ensure that all the cases of transition firings & their conflicts, cases of dead locks have exploited, the simulation time should be carefully chosen.

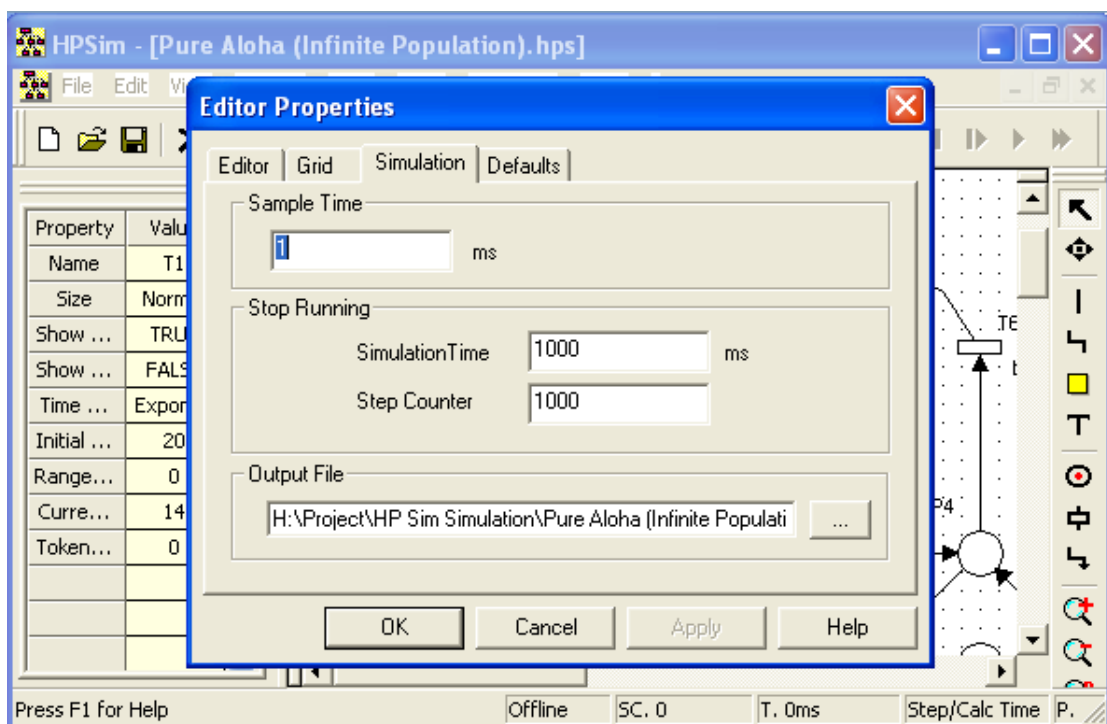


Figure 3.4: Setting Simulation Properties in HPSim

3.8.2 Simulation Output

The occurrence of an event in the simulation system relates to the firing of a transition in the Petri net model. The state of the system is represented by the

current marking which is being changed by the processing of an event i.e. the firing of a transition [11]. In terms of the Petri net model, the changes that occur when a transition fires are:

1. The input places of the transition lose tokens.
2. The output places of the transition gain tokens.
3. The clock in the simulated system advances by the firing time of the transition.
4. The total number of times the transition has fired gets incremented by 1.

In the simulation, each time a transition fires, the above changes are monitored and stored in suitably chosen data structures. At the end of the simulation, performance measures are computed by looking at the values contained in these data structures. Below are some data structures used in simulation of the Petri net models:

- *Firings*: 'Firings' is a vector with one element for every transition. At any given point in simulation, it gives the total number of times the transition T has fired from the start of the simulation.
- *Token Loss*: 'Token loss' is a vector with one element for every place. For place P, token loss [P] gives the total number of times P has lost a token during the simulation so far.
- *Transition Name Vector*: It represents the set of all transitions in the net and provides the name information about them.
- *Position Name Vector*: It represents the set of all places in the net and provides their names description.
- *Arc Type Matrix*: it is two dimensional array whose rows corresponds to position name vector and columns corresponds to transition name vector and gives the information about the type of arc used based on whether arc is ordinary or inhibitor.
- *Marking Vector*: it gives the number of tokens in any place P at given instance of time.

- *Incidence Matrix*: it is a two dimensional array whose rows and columns corresponds to position name vector and transition time vector respectively and elements gives the difference between the total number of tokens consumed and the total number of tokens fired by a transition.

Chapter 4

High Level Petri Nets

Petri nets can be applied informally to any area or system that can be described graphically like flow charts and that needs some means of representing parallel or concurrent activities. But basic Petri nets are not concise and convenient enough to be useful in modeling high level, complex processes. To counter this, a number of Petri nets extensions have been developed to add special modifications or restrictions suited to particular applications. These include the notion of 'color', 'time' and 'hierarchy'. High Level Petri Nets collectively refers to these extensions.

This chapter discusses Petri Nets with time and their application to information processing systems. First Stochastic Petri nets (SPNs) and some extensions of SPNs have been discussed and then discrete event simulation of Petri nets has been described. To expose the features of Petri nets over other modeling techniques, a Petri net model of Open Traffic Information System (OTIS) has been presented and analyzed.

4.1 Timed Petri Nets

Many ways of incorporating time have been introduced in Petri nets to evaluate system performance and to integrate stochastic elements that are so frequent in real world problems. Time can be associated with transitions, places, tokens and/or arcs.

In most timed Petri nets, transitions have associated time which determines the delay between the moment the transition becomes enabled and the moment

the transition becomes fireable [12], [13]. Some examples are *Stochastic Petri Nets* (SPNs), *Stochastic Timed Petri Nets* (STPNs), *Deterministic and Stochastic Petri Nets* (DSPNs), *Generalized Stochastic Petri nets* (GSPNs), *Extended Stochastic Petri Nets* (ESPNs) and others.

A stochastic Petri net (SPN) is a Petri net where each transition is associated with an exponentially distributed random variable that expresses the delay from the enabling to the firing of the transition [4]. In a case where several transitions are simultaneously enabled, the transition that has the shortest delay will fire first.

Suppose the delay d_i associated with transition T_i is a non negative continuous random variable X with the exponential distribution function

$$(4.1) \quad F_x(X) = \Pr[X \leq x] = 1 - e^{-\lambda_i x}$$

(or the probability density function, $f_x(x) = \lambda_i e^{-\lambda_i x}$).

Then, the average delay is given by

$$(4.2) \quad \bar{d}_i = \int_0^{\infty} [1 - F_x(X)] dx = \int_0^{\infty} e^{-\lambda_i x} dx = \frac{1}{\lambda_i}$$

Where λ_i is the firing rate of transition T_i .

SPNs have been extended to a class of *Generalized Stochastic Petri nets* (GSPN) [4]. A GSPN has two types of transitions (timed and immediate). A timed transition has an exponentially distributed firing rate, and an immediate transition has no firing delay. It is used to represent a logical control or an activity whose delay is negligible compared with those associated with timed transitions.

There are many more classes of Petri nets, *Deterministic and Stochastic Petri nets* (DSPN)] where transitions can be fired either in zero time or after a constant (deterministic time transitions) or exponentially distributed time delay (stochastic time transitions), *Extended Stochastic Petri nets* (ESPN) have generally distributed stochastic transitions delay (the distribution of transitions

time is general e.g. the superposition of two distributions), *Stochastic Timed Petri Nets* (STPN) where transitions have zero, deterministic or stochastic time interval. Stochastic time interval is generally distributed.

Note that in “non time” Petri nets the enabled transitions are always fireable transitions. From here onwards, the only non-zero time objects in Petri nets will be transitions.

4.1.1 Policies Associated with Transition Firings

Different policies may be assumed for transition firings. Following are two basic policies differing in treatment of tokens in incoming places [12], [13]:

1. *Three-phase firing*: in this policy tokens are reserved during the transition delay. More precisely,
 - a) tokens are consumed from input places when the transition is enabled,
 - b) the delay elapses,
 - c) tokens are generated in output places.

A conflict cannot arise after the reservation of tokens, because they are invisible for the rest of the model.

2. *Atomic firing*: tokens remain in input places for the transition delay; they are consumed from input places and generated in output places when the transition fires. A transition T is fireable if and only if it has been continuously enabled during its associated time. The transition is called a time-waiting transition during this time.

Each firing policy has its own field of scope. Three phase firings are more appropriate for modeling of resource sharing and atomic firings suit better for example for modeling and analysis of communication protocols.

The main contrast between the three phase firing and atomic firing policies is that in atomic firing, the tokens may be consumed by another transition during the time delay.

4.1.2 Conflicts among Transition Firings

When more than one transition with atomic firing is enabled, the behaviour is similar, but a problem arises: Which one of the enabled transitions is going to fire?

A conflict (when a firing of one transition disables the other) between two immediate transitions is resolved by some pre defined rule e.g. priority, probability, inhibitor arc etc. A conflict between a time waiting transition and a fireable transition, the fireable is fired and time waiting transition becomes disabled. A conflict among fireable transitions or fireable transitions and immediate transitions is resolved in the same way as a conflict among immediate transitions. To resolve conflicts between two simultaneously enabled timed transition there are two alternative selection rules [12]:

1. *Preselection*: The enabled transition that will fire is chosen when the marking is entered, according to some metric (priority, probability, ...)
2. *Race Condition*: The enabled transition that will fire is the one whose firing delay is minimum. During the delay, the transition has to be continuously enabled.

4.1.3 Memory Policies

When a time-waiting timed transition becomes disabled and enables again, there are two possibilities what happens with the “state of its time”. After becoming enabled again, the transition may continue, where it stopped or it always starts from the beginning. Although both of the definitions have its importance, only the second one will be used for the analysis purposes because it better suits for modeling time constants like packet transmission time or channel idle period in communication protocols.

4.1.4 Multiple Transition Enabling

The enabling degree of a transition is the number of times the transition could fire in the given marking before becoming disabled. When the enabling degree of a transition is greater than one, attention must be paid to the firing semantics.

Following are three definitions differentiating by overlapping or non-overlapping of time delay [13].

1. *Single server semantics*: A firing delay is set when the transition is first enabled and new delays are generated upon transition firing if the transition is still enabled in the new marking. The transition cannot be enabled again during its time delay.
2. *Infinite server semantics*: If the transition becomes enabled again during its time waiting period, a new independent firing process can be initiated which will overlap with previous firing processes. There is no limit of simultaneous firings of the same transition. If a transition is enabled several times it may start, several independent firings or time waiting period at the same moment.
3. *Multiple server semantics or n-limited semantics*: Enabling sets of tokens are processed as soon as they form in the input places of the transition up to a maximum degree of parallelism (say n) which allows the transition to be in n states simultaneously.

4.1.5 Transition States

A non-zero time transition immediately moves to time-waiting state after enabling. Here it may be disabled or it may become a fireable. A fireable may be fired or may be disabled (removing the tokens from its precondition places). After firing, a transition becomes either disabled or enabled.

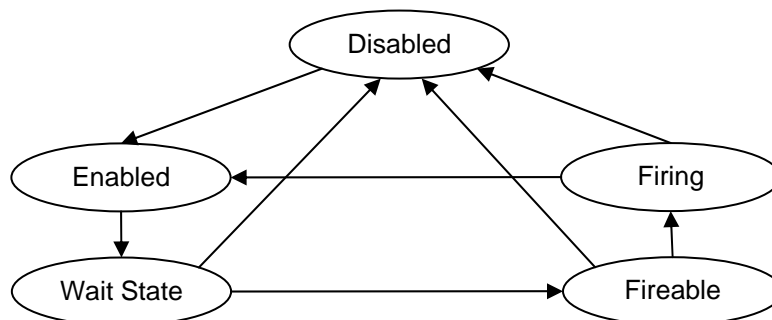


Figure 4.1: Transition State Diagram

4.2 Types of Network Traffic

In this section various types of network traffic have been considered and shown that how to model these packet arrivals by timed Petri nets.

Deterministic Packet Arrival Process

Deterministic process of packet arrivals applies for traffic generated by an automatic process e.g. information packets in some routing protocols. Such type of packet traffic can be modeled by a deterministic time source transition. Time associated to the transition represents the time between two consecutive packet arrivals.

Poisson Packet Arrival Process

In the Poisson packet arrival process number of packets generated per one time unit, is given by the Poisson distribution. It is modeled by a transition with exponential distribution. The mean time associated to the transition represents the mean value of time between two consecutive packet arrivals.

In queueing systems, the users requesting the service are either satisfied without any delay or they are placed in a queue or in retrial queueing systems, they retry the service again after some period.

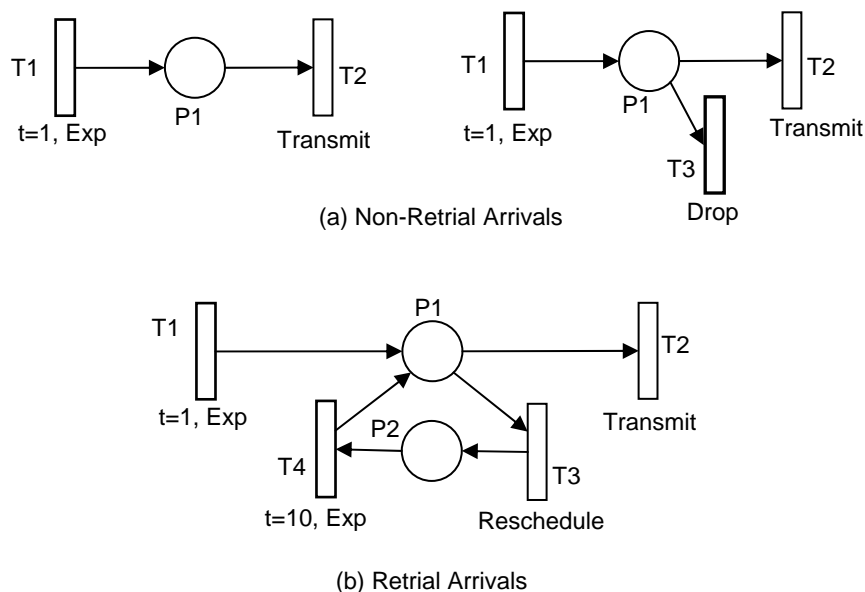


Figure 4.2: Poisson Arrival Processes in Petri Nets

Markov Modulated Poisson Packet Arrivals Process

An example of MMPP is a process, which switches between silent periods and periods in which the requests arrive with rate λ . Such example applies in modeling of finite population systems (like real computer networks) when the nodes have output buffers of capacity one. In such case, new packet cannot arrive when there is a packet in output buffer of transmitter.

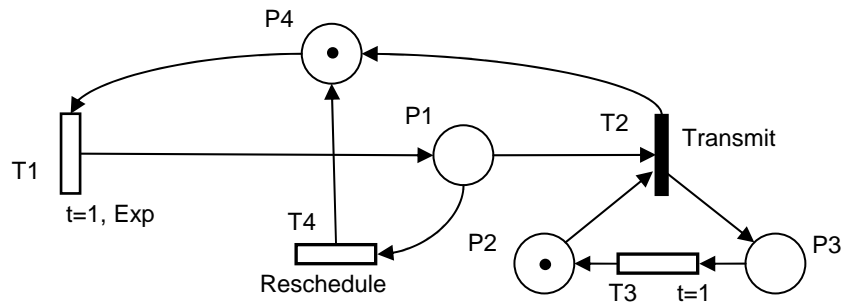


Figure 4.3: Two state MMPP Arrival

In figure 4.3, a new packet may arrive during transmission of previous one but will not be transmitted. The sending buffer is full until transmission of packet is finished (the capacity of both output queue and sending buffer equals to one). Arrival process is disabled until the request is either transmitted or rescheduled. Immediate transition T2 shown by black box represents starting of packet transmission and time of deterministic time transition T3 corresponds to completion time of the transmission. An example of a queueing system with such an arrival process is a G/D/1 system.

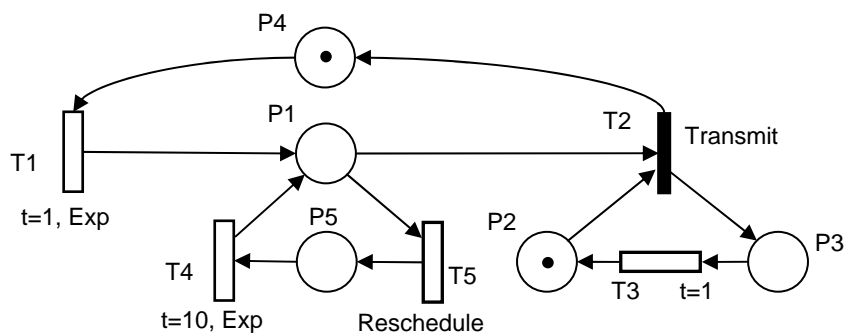


Figure 4.4: Two State MMPP Retrial Arrival

Only difference in figure 4.4 from the previous one is that when a packet arrive during transmission of another packet it is rescheduled for retransmission after a random period given by exponentially distributed time of transition T_4 until it is successfully transmitted. A new packet can not arrive during retrials of previously unsent packets.

4.3 Simulation of Timed Petri Nets

Developing and analyzing stochastic models of complex computer, manufacturing, telecommunication, workflow, or transportation systems is almost always a challenging task. Such systems are often viewed as being discrete-event stochastic systems, i.e., as making stochastic state transitions at a strictly increasing sequence of random times. Real-world systems are often so complicated that simulation is the only available analysis tool. Simulation, however, cannot be applied blindly, or misleading results will ensue. Key issues include:

1. How to model the system of interest,
2. How to determine whether the system is stable, so that long-run performance measures of interest are well defined,
3. How to efficiently generate sample paths of the system process on a computer, and
4. How to analyze the output of a simulation in order to draw statistically meaningful conclusions.

Stochastic Petri nets (SPNs) with general firing times are an ideal theoretical and practical framework for addressing these challenges. Since the use of SPNs as an analytical tool is based on the generation of the entire state space, the technique becomes intractable for large systems. In such cases, discrete event simulation (DES) is the preferred tool for analysis.

Discrete Event Simulation of Petri Net Models

When executed DES sequentially and repeatedly processes the occurrence of events in simulated time maintaining a time ordered event list. There is a natural

correspondence between events and transition firings. The occurrence of an event in the simulation system relates to the firing of a transition in the Petri net model [11].

The event list hence carries transitions and the time instant at which they will fire, given that firing does not become obsolete in the mean time. The state of the system is represented by the current marking which is being changed by the processing of an event i.e. the firing of a transition: the transition with the smallest time stamp is withdrawn from the event list; tokens are removed from its input places and deposited in its output places.

However, the new marking can enable new transitions or disable enabled transitions such that event list has to be corrected accordingly: new enabled are scheduled with their firing time to occur in the future by inserting them into the event list while disabled transitions are removed. Finally the simulated time is set to the timestamp of the transition just simulated. It is obvious that always the transition with smallest timestamp is simulated first; otherwise the future of the simulation could have an impact onto the past which could yield causality errors.

There may be situations where several transitions have identical smallest time stamps. This happens when there are two or more events (potentially) can occur or (actually) do occur simultaneously. Then conflicts are resolved by priorities, probabilistic functions, pre-selection rules or race conditions.

The checking, after firing any transition, of transitions which are affected i.e. which one is now enabled and which one is disabled can be simplified as compared to checking for all transitions.

This is based on the idea that firing of a transition T_i : $T_i \in \mathbf{T}$ may influence those transitions, which are in *structural conflict* with T_i $\{(\bullet T_i)^\bullet\}$, or *causally connected transitions* of T_i $\{(T_i)^\bullet\}$ After firing of T_i , the conflict transitions may become disabled and the causally connected transitions may become enabled. Thus, after firing of T_i , one has to check $|(\bullet T_i)^\bullet| + |(T_i)^\bullet|$ transitions whether they are enabled or not.

The checking causally dependent and conflict transitions may be simplified for some Petri net models. For example, all conflict transitions of T_i become disabled after firing of transition T_i in *safe* Petri nets (in *safe* Petri nets each place at any time instant can have at most one token). In pure *safe* Petri nets also T_i becomes disabled. All causally connected transitions become enabled in *state machines*. All transitions T_j : $|\bullet T_j| = 1$, $T_j \in (T_i^\bullet)^\bullet$ become enabled after firing of T_i .

4.4 Example: Modeling of Open Traffic Information System

Petri nets have received considerable attention for describing and studying information processing systems. Next, an example, Open Traffic Information System (OTIS) [14] is modeled using Petri net. The reason why OTIS is chosen is due to its dynamic requirements and concurrent, asynchronous and stochastic properties that would efficiently expose the potential of Petri nets. It has shown that systems with such dynamic properties cannot be efficiently emulated using modeling techniques DFD while Petri nets have proper constructs to effectively represent all these characteristics.

4.4.1 Open Traffic Information System (OTIS)

The OTIS model [14] has been proposed as a means to generate and disseminate traffic information to mobile users. It allows any mobile user to provide traffic information through Short Message Service (SMS). The information gathered is in the form of an update SMS containing a <keyword> <location> pair so that each location in the underlying database of the system can be appropriately updated with the traffic status. The keywords signify the level of traffic congestion. For instance, 1- smooth, 2- slow, 3- jam etc. The second type of message that the user may send to the system is a request SMS to get traffic information about a particular location. On receipt of such a message, the system accesses the information database and sends back the current traffic information against the requested location.

This system is concurrent and asynchronous as any number of messages can be received or sent by the system simultaneously. Moreover, there are real

time constraints on the system as it has to be ensured that only up-to-date traffic information is sent to the user.

4.4.2 OTIS Model

This section discusses both the approaches to model the information system OTIS: *Data Flow Diagram (DFD) approach* and *Petri Net (PN) approach*.

Data Flow Diagram Approach

A data flow diagram (DFD) is a modeling technique that models the flow of information in the system by tracking various processes and entities that input information to the system and gain output from the system. Figure 4.5 [14] depicts the DFD of the information system OTIS.

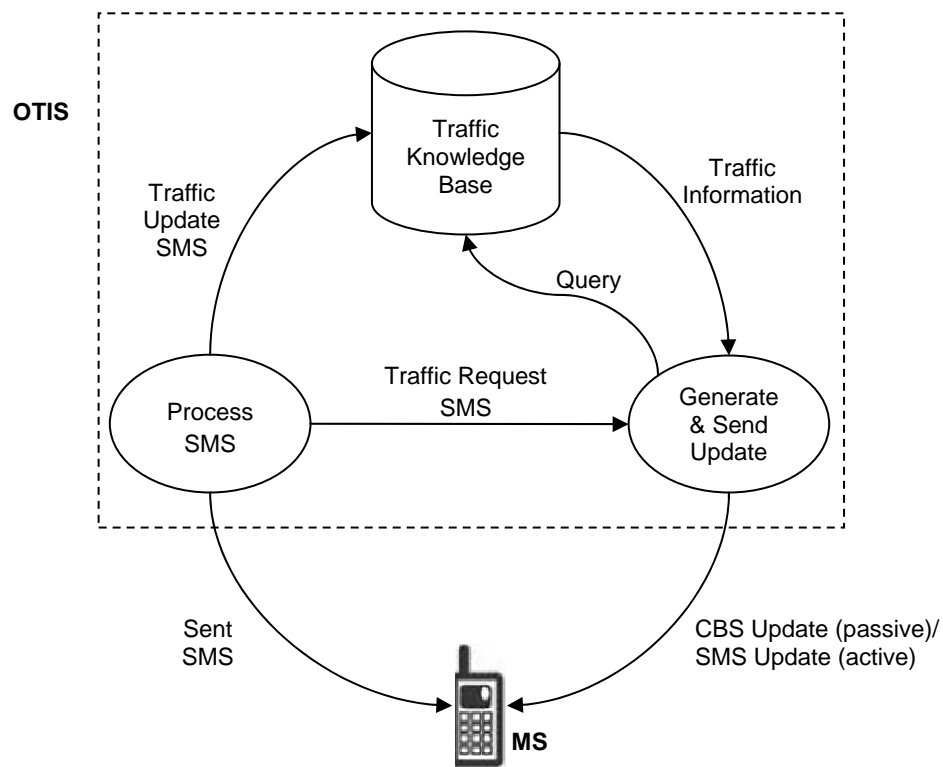


Figure 4.5: Data Flow Diagram of OTIS

The mobile station is the external entity that supplies information to the system as well as consumes from it. The figure also shows the data store that contains the database of locations and traffic information and is queried while sending traffic updates to the users.

Petri Net Approach

In Petri net model of OTIS, as depicted in figure 4.6, transitions follow infinite server semantics and atomic firings. There are no capacity restrictions on places. To represent databases used in the system, doubled line circles have been used.

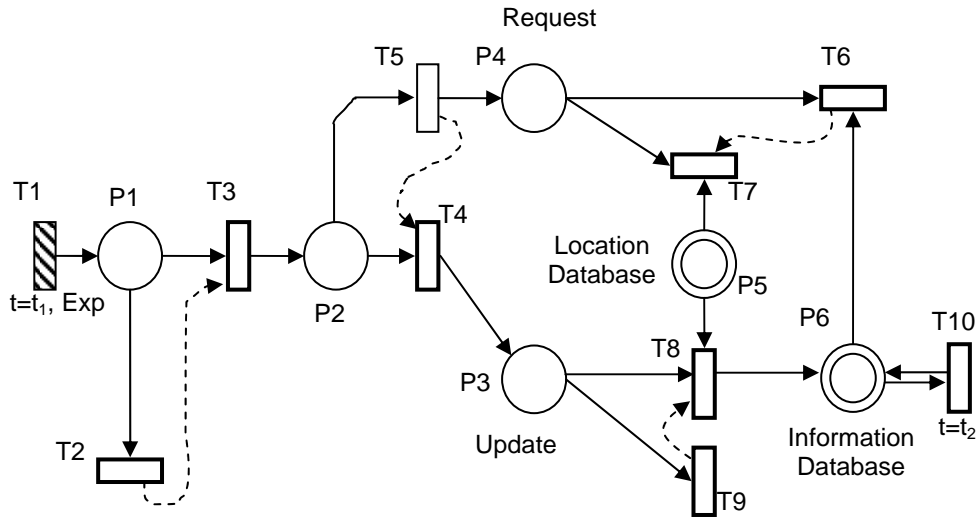


Figure 4.6: Petri Net Model of OTIS

To resolve conflicts a binary priority relation on conflicting transitions represented by a dashed line going from the less priority transition to the transition with higher priority has been used. The relation is assumed to be non-reflexive and anti-symmetric. An enabled transition can fire only if no other transition with higher priority fires.

In the model, all transitions are deterministic transitions with non-zero time delay except transitions T1 and T5. Transition T1 is a stochastic transition while T5 is a zero-time delay deterministic transition. The various places and transitions depicted in figure 4.6 are defined in the Table 4.1. Following is a step-by-step description of the OTIS functioning as depicted in the figure 4.6:

1. T1 transition signals the random arrival of messages as multiple users may send messages to the system in any arbitrary order.
2. Next, the incoming message needs to be processed for validation. If the message was found to be valid, transition T3 signals a valid message and forwards the message for further processing. Otherwise

T2 sends back a reply to the user indicating that the message sent was in an invalid format.

3. A valid message is further processed by transition T4 to determine its type. If it is an update message T4 fires otherwise transition T5 processes the message.
4. If location mentioned in request and update messages does not match with those in the location database, Transitions T7 and T9 respectively send "location not found" messages to the user.
5. If location is matched then transition T8 performs a traffic update in the information database while transition T6 will query the information database to get the traffic status and reply appropriately to the user.
6. To prevent the database from giving back old information to the users, the transition T10, a deterministic transition with some non-zero time delay, keeps updating the database periodically, say every 10 minutes.

Table 4.1: Explanation of Petri Net Model of OTIS

Pi	Explanation	Ti	Explanation
P1	Message received	T1	Shows event of message arrival
P2	Received message is valid	T2	Sends " Invalid Format" message to user
P3	Received message is an update message	T3	Valid message is received
P4	Received message is a request message	T4	Received message is update message
P5	Location Database	T5	Received message is request message
P6	Information Database	T6	Sends traffic information status to user
		T7	Sends "location not found" message to user
		T8	Message has valid location, information database updated
		T9	Sends "location not found" message to user
		T10	Update database for each time slot

4.4.3 Analysis and Results

The transition T10 in figure 4.6 can simultaneously fire for any number of times. This represents the real time constraints of the system which was not easily deducible from the earlier DFD approach. Transition T1 has exponentially distributed time association which depicts the random arrival of user messages. Here more than one transition in place of single transition can also depict concurrent arrival of messages.

The source transition T1 and sink transitions like T2, T7 and T9 represent respectively the start and end points of the main process which wasn't visible in the earlier approach. The model also depicts system processes of deterministic durations. For instance, transition T8 updates the database and transition T4 checks the nature of incoming message, both are examples of deterministic transitions.

The transition pairs like (T4 and T5) and (T6 and T7), examples of decision sequences, are clearly modeled using priority relation. Although both transitions are enabled simultaneously, only one of them may fire. Such control flow and decision constructs that are inherent to OTIS are unambiguously depicted by the Petri net model.

For a system like OTIS that needs to deal with aspects like communication, concurrency, synchronization and resource sharing is especially well modeled by Petri nets as they have powerful abilities for representation of system dynamics like - entity arrival dynamics, resource availability, resource interdependency, unique representation of causal relations, start and termination of activities, event firing conditions and other control mechanisms. Moreover, Petri nets are highly extensible and any further modifications that may come up in the system can be easily incorporated in the model.

Chapter 5

Modeling and Analysis of Carrier Sense Multiple Access Protocols

Computer communication networks have come of age. Today, there is hardly any professional, particularly in engineering, that has not been the user of such a network. This proliferation requires the thorough understanding of the behavior of networks by those who are responsible for their operation as well as by those whose task is to design such networks.

The Aloha family of protocols is probably the richest family of multiple access protocols. Its popularity is first of all due to seniority, as it is the first random access technique introduced. Second, many of these protocols are so simple that their implementation is straightforward. But the Aloha schemes exhibited fairly poor performance which can be attributed to the “impolite” behavior of the users namely, whenever one has a packet to transmit he does so without consideration of others.

Consider a behavior that is generically characterize as “listen before talk”, that is, every user before attempting any transmission listens whether somebody else is already using the channel. If this is the case the user will refrain from transmission to the benefit of all; his packet will clearly not be successful if transmitted and, further, disturbing another user will cause the currently transmitted packet to be retransmitted, possibly disturbing yet another packet.

Carrier sensing does not, however, relieve us from collisions. Suppose the channel has been idle for a while and two users concurrently generate a packet.

Each will sense the channel, discover it is idle, and transmit the packet to result in collision. “Concurrently” here does not really mean at the very same time; if one user starts transmitting it takes some time for the signal to propagate and arrive at the other user. Hence concurrently actually means within a time window of duration equal to signal propagation time. This latter quantity becomes therefore a crucial parameter in the performance of these protocols.

5.1 Traffic Model: Assumptions and Notation

The channel, through which data is to be transferred from its source to its destination, is *errorless without capture*: whenever a transmission of a packet does not interfere with any other packet transmission, the transmitted packet is received correctly while if two or more packet transmissions overlap in time, a collision is caused and none of the colliding packets is received correctly and in most protocols have to be retransmitted.

The traffic source consists of an infinite number of users who collectively form an independent Poisson source with an aggregate mean packet generation rate of λ packets/sec. This is an approximation to a large but finite population in which each user generates packets infrequently and each packet can be successfully transmitted in a time interval much less than the average time between successive packets generated by a given user. Each user in the infinite population is assumed to have at most one packet requiring transmission at any time (including any previously blocked packet).

A node may, at any one time, either be transmitting or receiving (but not both simultaneously). However the delay incurred to switch from one mode to the other is negligible. The traffic is characterized as follows: each packet is of constant length requiring T seconds for transmission. Let $S (= \lambda T)$ is the average number of packets generated per transmission time, i.e., it is the input rate normalized with respect to T . Under steady-state conditions, S can also be referred to as the channel throughput rate.

If there is no overlap or gaps between the packets, one can achieve a maximum throughput equal to 1; therefore S sometimes also referred as the

channel utilization. But because of the interference problem inherent in the random nature of the access modes, the achievable throughput will always be less than 1.

Since conflicts can occur, some acknowledgment scheme is necessary to inform the transmitter of its success or failure. An acknowledgment packet is created only when a packet is correctly received (a positive acknowledgment scheme) is followed by the network. The channel for acknowledgment is assumed to be separate from the channel under study i.e. acknowledgements arrive reliably and at no cost. If within some specified delay (an appropriate time-out period) after the transmission of a packet, a user does not receive an acknowledgment, he knows he has conflicted. If he now retransmits immediately, and if all users behave likewise, then he will definitely be interfered with again. Consequently, each user delays the transmission of a previously collided packet by some random time whose mean is \hat{X} .

The traffic offered to the channel from the collection of users consists not only of new packets but also of previously collided packets: this increases the mean *offered traffic rate* which is denoted by G (packets per transmission time T) where $G \geq S$. Following are the four further assumptions considered in the models:

1. The average retransmission delay \hat{X} is large compared to T .
2. The interarrival times of the requests for packet transmission defined by the start times of all the packets plus retransmissions are random, independent and exponentially distributed.
3. The acknowledgment packets are always correctly received with probability one.
4. The processing time needed to perform the sumcheck and to generate the acknowledgment packet is negligible.

Without loss of generality, T can be chosen equal to 1. This is equivalent to expressing time in units of T . 'a', the propagation delay in normalized time units, is identical for all source-destination pairs and has very small value as compared to the packet transmission time.

5.2 Modeling and Analysis of Communication Protocols








The modeling is based on queueing theory and probabilistic arguments requiring independence of random variables provided by assumption 2. The packet arrival from whole population is approximated by a single packet Poisson arrival process with corresponding mean value (law of large numbers).

All the parameters considered are normalized with respect to the *packet transmission time* T which is assumed to be constant. In practice, T is given by the length of the packet and by the channel transmission rate. Inter-arrival times between consecutive packets form an exponentially distributed random variable with mean $1/G$.

Three types of transitions have been used in Petri net models: *immediate transitions*, *deterministic transitions* and *stochastic transitions* which are associated with exponentially distributed random variable. Transitions follow infinite server semantics and atomic firing policy i.e. tokens are not reserved during transition delay. A transition which is enabled or fireable may be disabled by another transition. Whenever a disabled timed transition enables again, its delay period starts from beginning.

To run simulation, simulator HPSim is used where simulation can be visualized as a token game animation. In Table 5.1, description of various notations that are used during simulation of Petri net models of communication protocols is given.

Table 5.1: Description of Various Notations used in the Petri Net Simulation

Notation	Description	Notation	Description
	Place with no token		An immediate transition
	Place with a single token		A fireable transition
	Place with n number of tokens		An enable transition
			A timed transition

Two more additional data structures (section 3.8.2) used in simulation of timed Petri nets are:

1. *Global Clock*: This is a global real variable with initial value zero. It gets incremented each time a timed transition fires, by an amount equal to the firing time of the transition. When an immediate transition fires, the global clock is not incremented.
2. *Transition Time Model Vector*: This gives the information about the type of transition used in the model.

5.3 Aloha Protocols

Aloha refers to a simple communications scheme in which each source (transmitter) in a network sends data whenever there is a packet to send. If the packet successfully reaches the destination (receiver), the next packet is sent. If the packet fails to be received at the destination, it is sent again.

There are two versions of Aloha: *pure* and *slotted*. They differ with respect to whether time is divided into discrete slots into which all packets must fit.

5.3.1 Pure Aloha

The *pure* Aloha protocol considers a single hop system with an infinite population generating packets of equal length T according to a Poisson process with rate λ packets/sec. The users can transmit at any time they need. If the users fail to hear their successful transmission after a propagation delay, they know that a collision occurred and retransmit the packets according to retransmission delay function. A given packet will overlap with another packet if there exists at least one start of transmission within T seconds before or after the start time of the given packet. The channel throughput of pure Aloha is [16]

$$(5.1) \quad S = G e^{-2G}$$

Successful transmission of a packet means that there was exactly one transmitting user on the channel during the interval $2T$. Pure Aloha achieves a maximal channel throughput of $1/(2e) = 0.1839$ at $G = 1/2$. Pure Aloha packet

timing is shown in figure 5.1 [17] (arrow below time axis denotes arrivals of packets).

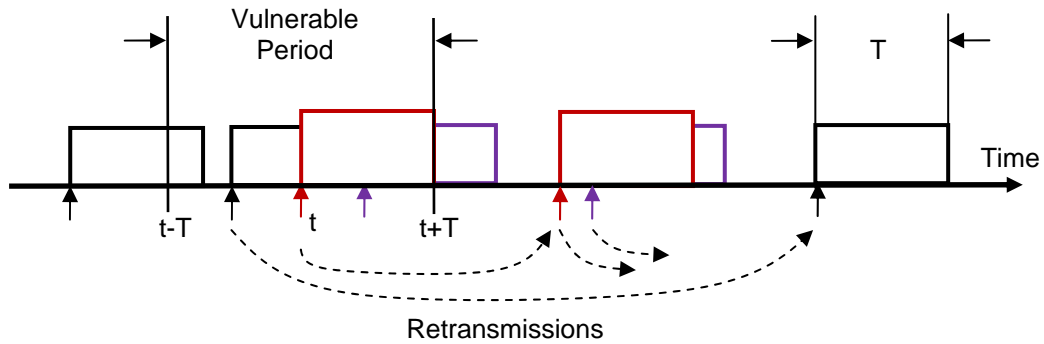


Figure 5.1: Pure Aloha: Packet Timing

Petri Net Model of Pure Aloha

Vulnerable interval equals to the packet transmission time under condition that all nodes, never starts their transmission when their receiver part is receiving a packet. Otherwise vulnerable interval is a sum of propagation delay a (given by physical properties of the channel) and packet transmission time.

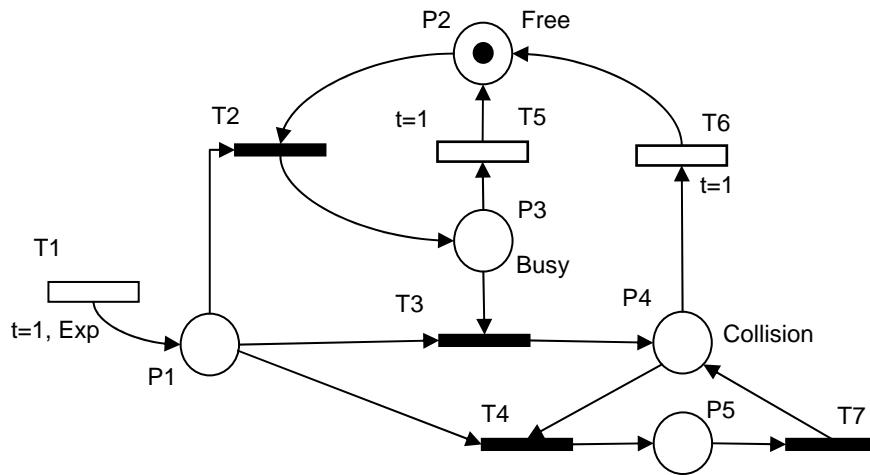


Figure 5.2: Petri Net Model of Pure Aloha

The stochastic time transition T1 represents the packet arrival process in Petri net model of pure Aloha because the transition is associated with exponentially distributed random variable. T1 is a source transition. Transition T5 represents

packet transmission delay Transition and T6 represents time elapsed between the arrival time of the last packet participated in collision and the time when collision resolved. Both transitions are deterministic time transitions with time delay $t = 1$.

Table 5.2: Explanation of the Petri Net Model of Pure Aloha

Pi	Explanation	Ti	Explanation
P1	A request for packet transmission	T1	Represent packet arrivals process
P2	Channel is free	T2	Transmission of the packet started
P3	Packet is being transmitted	T3	A collision occurred
P4	A collision occurred	T4	A packet arrived during collision
P5	A packet arrived during collision	T5	Transmission of the packet completed
		T6	Channel is free from collision

Table 5.3: Description of the Transitions in the Petri Net Model of Pure Aloha

Transition	Transition Type	Associated Delay
T1	Exponential	Variable depending on number of nodes or requests for transmission per unit transmission time
T2, T3, T4, T7	Immediate	0
T5, T6	Deterministic	T (the packet transmission time)

Table 5.2 gives the explanation of various places and transitions while table 5.3 describes about the type and associated delay of these transitions of the Petri net model of pure Aloha in figure 5.2. Following is the net description of the model in terms of various data structures:

```
// Transition Name Vector:
(T1 ;T2 ;T3 ;T4 ;T5 ;T6 ;T7 ;)
// Position Name Vector:
(P1;P2;P3;P4;P5;)
```

```

// Marking Vector:
(0 1 0 0 0 )
// Incidence Matrix:
{
(-1 1 1 1 0 0 0 )
( 0 1 0 0 -1 -1 0 )
( 0 -1 1 0 1 0 0 )
( 0 0 -1 1 0 1 -1 )
( 0 0 0 -1 0 0 1 )
}
// Transition Time Model Vector:
// Code:1 = Immediate; 2= Delay;3 = Exponential
(3 ;1 ;1 ;1 ;2 ;2 ;1 ;)

```

Simulation of the Petri Net Model of Pure Aloha

When a request arrives for transmission (a token in place P1) and channel is free (presence of token in place P2) packet transmission will not result in collision (figure 5.3).

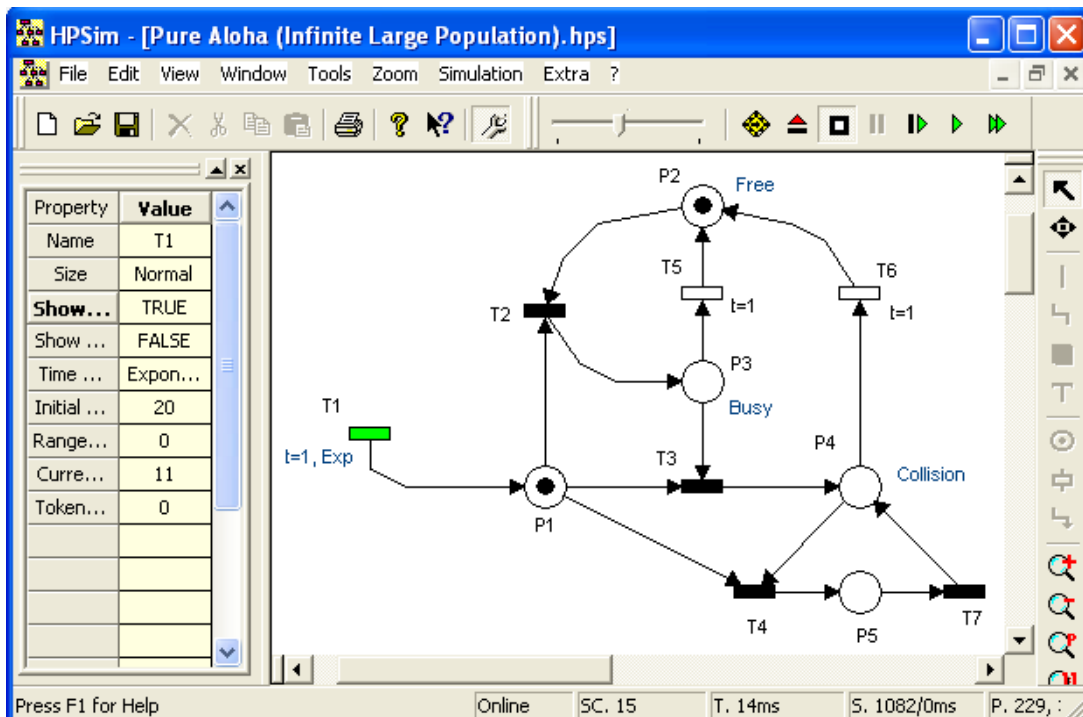


Figure 5.3: Petri Net Model of Pure Aloha: the channel is free and packet transmission will not result a collision

Place P5 and transition T7 avoid self-loop between P4 and T4. This has been done just to avoid any inconvenience due to presence of self-loops during simulation. The token resides in P3 until the deterministic time transition T5 is fired or until a new packet arrives. In latter case, the transition T3 is fired and the channel changes its state to collision (figure 5.4).

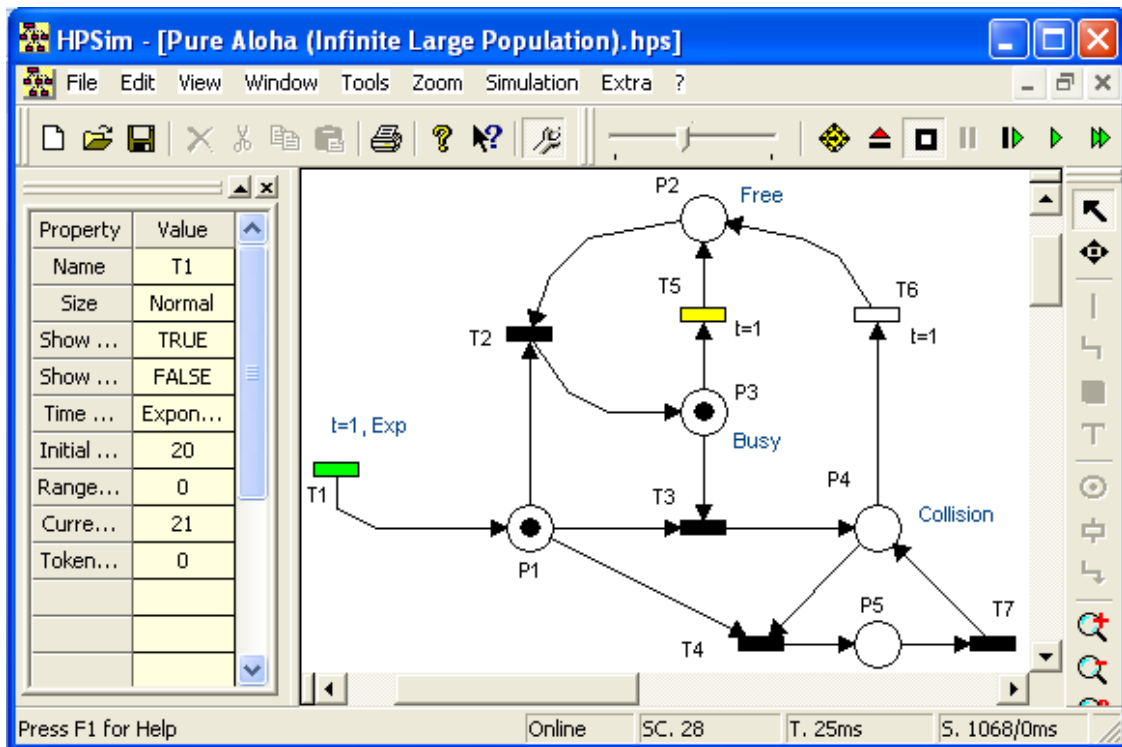


Figure 5.4: Petri Net Model of Pure Aloha: a new packet arrive before completion of transmission of previous packet, which results in a collision

If the channel is in collision state at time t and a new packet is offered to channel, the collision persists until time $t + 1$. This is modeled as follow: if no other packet arrives during the time of the collision, the channel returns to the idle state (the transition T6 is fired), otherwise the transition T4 is fired (figure 5.5) and the token in place P4 is moved to P5 and transition T6 is disabled. From place P5 token is again moved to place P4 by transition T7 which enables transition T6 again so that transmission delay starts from beginning.

Firing frequencies of transitions have the following meaning: Firing count of transition T1 means the offered traffic G . Firing frequency of transition T2 represents the frequency of initiations of the transmission on the previously

inactive channel. Firing frequency of transition T3 represents the number of collisions arising on the channel being previously in active but non-collision state and the firing frequency of transition T4 (same as T7) represents the number of collisions arising on the channel being previously in collision state. The number of successfully transmitted packets equals to the firing count of the transition T5. Thus, firing count of T6 represents the number of collision resolutions.

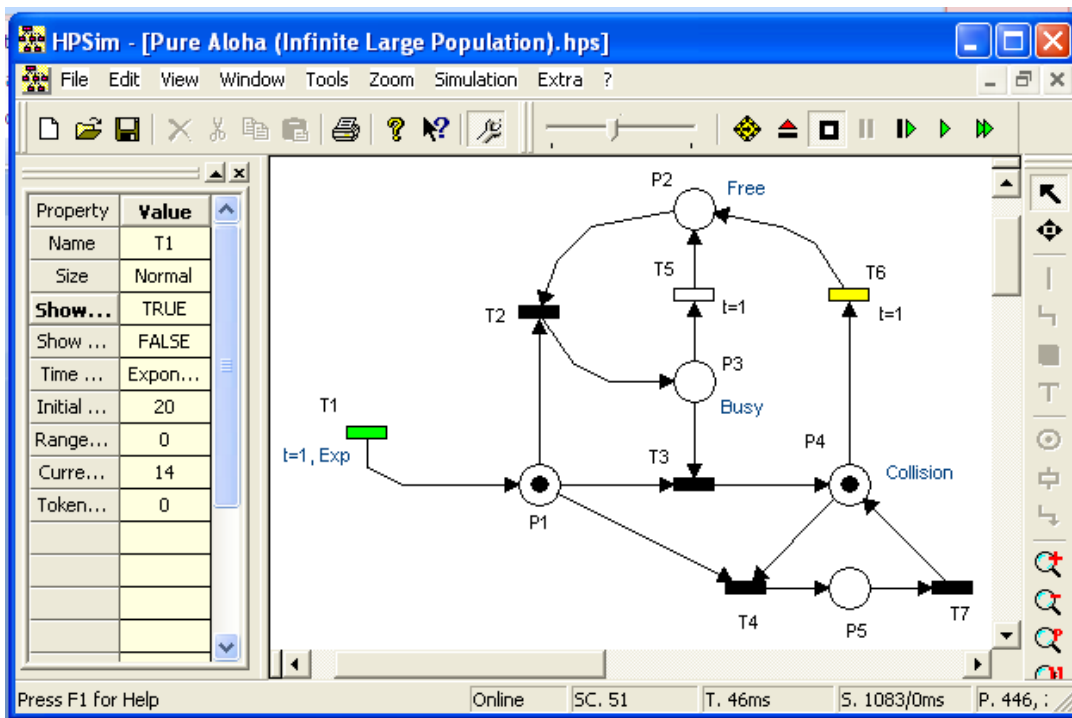


Figure 5.5: Petri Net Model of Pure Aloha: a new packet request for transmission before the collision is resolved

Petri Net Model of Pure Aloha with Finite Number of Users

To make the previous model more realistic, the case of an Aloha system with a finite number of users has been analyzed here. The model consisting of finite number of users (say k) can be viewed as finite population queueing system with a single packet output buffer. Such a model with two users is shown in figure 5.6. All packets are of the same size, requiring T seconds for transmission.

Arrival process generated by whole population is sum of Poisson processes; each of them is packet arrival process to one of k nodes. It is assumed that

there is no correlation between random variables generated by firings of transitions. Mean value of offered traffic is given by the sum of reciprocals of mean times of transitions representing packet arrival processes.

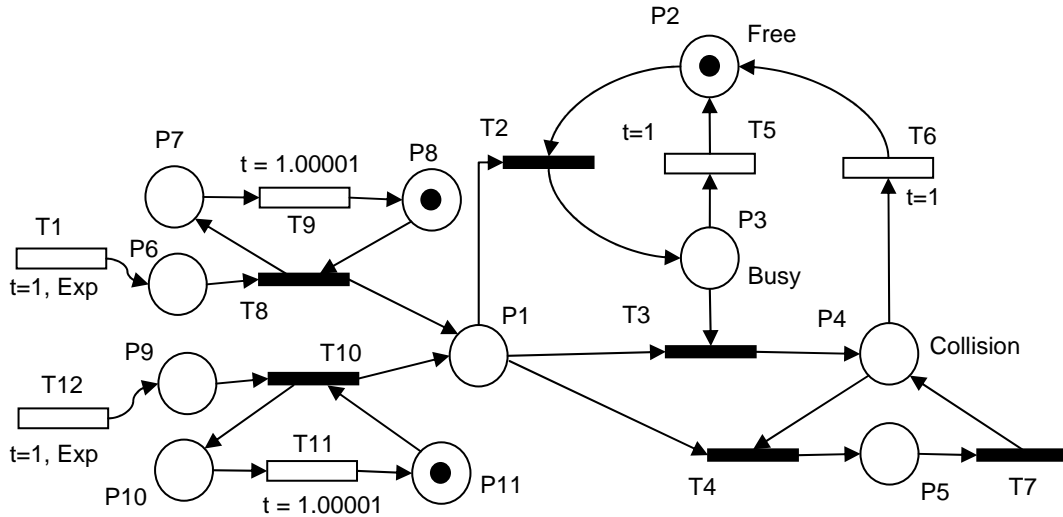


Figure 5.6: Petri Net Model of Pure Aloha: finite population

Transitions T8, T9, places P7 and P8 (and similarly transition T10, T11, & places P10, P11) ensure, that a node does not transmit during its own transmission. Transition T9 and T11 are deterministic time transitions with $t = 1$.

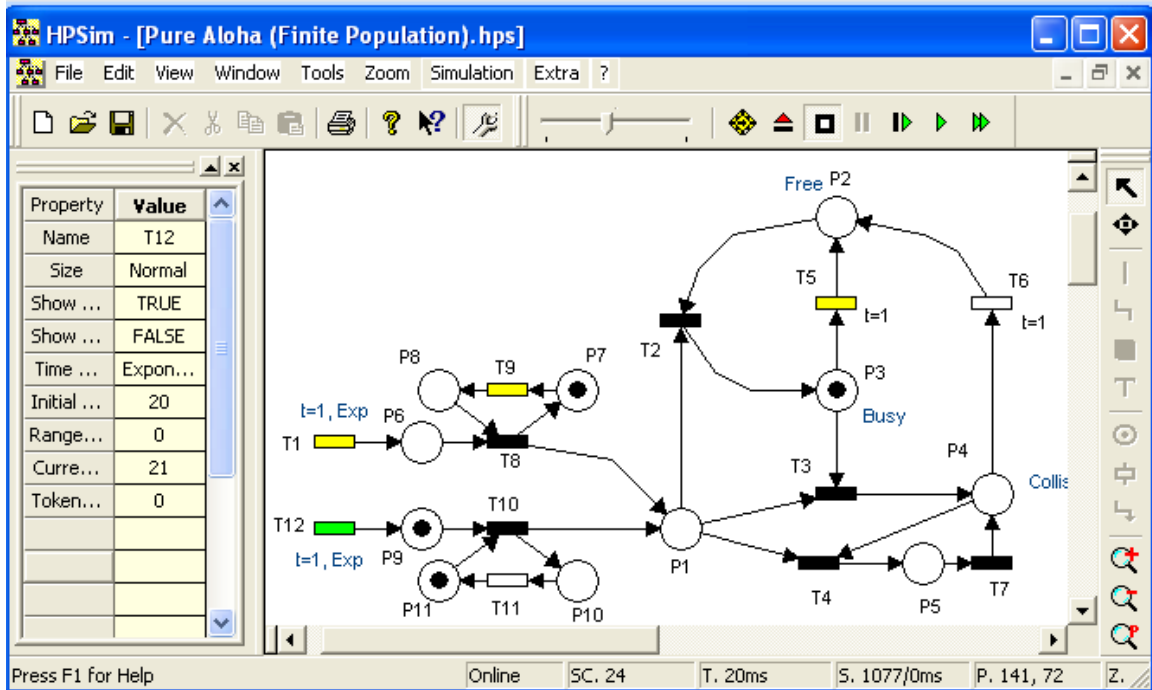


Figure 5.7: Petri Net Model of Pure Aloha (Finite Population): packet transmitted by second node (T12) will collide with the packet sent by first node (T1)

Transition T9 and T11 are deterministic time transitions with value just larger than the packet transmission time, which has been done to avoid the collision of the transmitted packet with the last bit of the previously transmitted packet.

5.3.2 Slotted Aloha

In *Slotted Aloha* [16] the time is slotted into segments, whose duration exactly equals to the packet transmission time T . The packet transmission always starts at the beginning of a time slot.

A slot will be successful if and only if exactly one packet was scheduled for transmission sometime during the previous slot. The throughput equation is given by

$$(5.2) \quad S = G e^{-G}$$

In slotted Aloha, the maximum channel throughput is equal to $1/e = 0.368$ at $G = 1$.

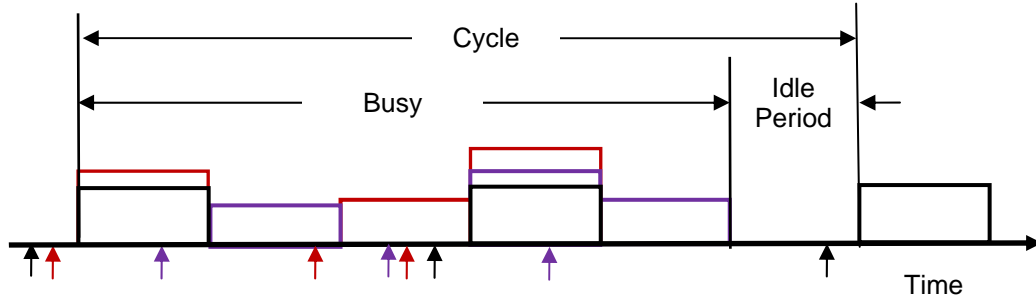


Figure 5.8: Slotted Aloha: Packet Timing

Note that, in contrast to pure Aloha, if two packets conflict, they will overlap completely rather than partially providing an increase in channel efficiency. Slotted Aloha achieves larger maximal throughput than pure Aloha (see figure 5.9) by reducing of time wasted by collisions. While in pure Aloha the time wasted by a collision is up to two time units long (two packet transmission times), a collision in slotted Aloha wastes exactly one time unit.

Since a node has at most one packet in its queue, for infinite population, the average delay between the moment of arrival of a packet and the moment when its transmission begins equals to $1/2$ time slot.

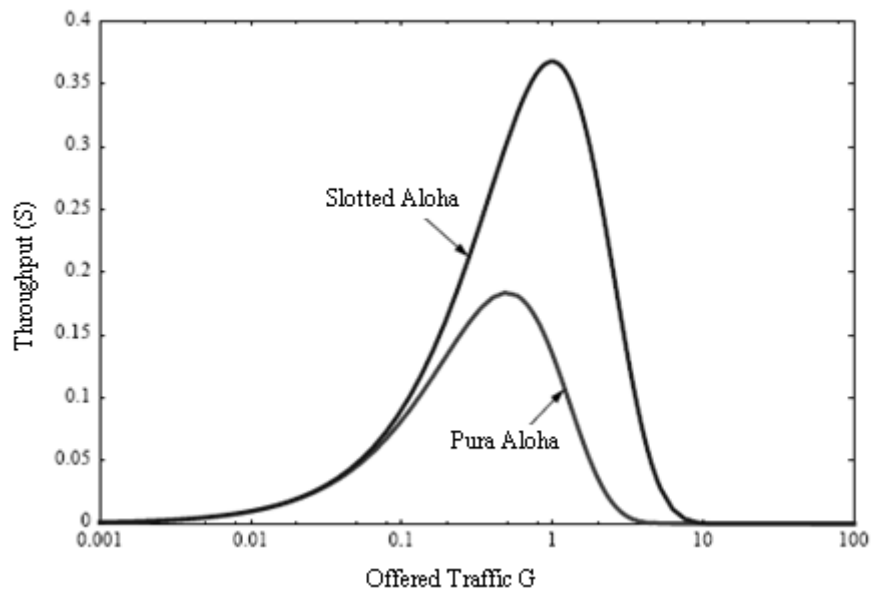


Figure 5.9: Throughput(S) - Offered load (G) of Pure and Slotted Aloha

Petri Net Model of Slotted Aloha

The model in figure 5.10 is almost similar as model of pure Aloha. To ensure that every transmission starts in slots transitions T8, T10, T11, T12, and places P6, P9, P10, P11 have been used. Transitions T11 and T10 are deterministic time transitions.

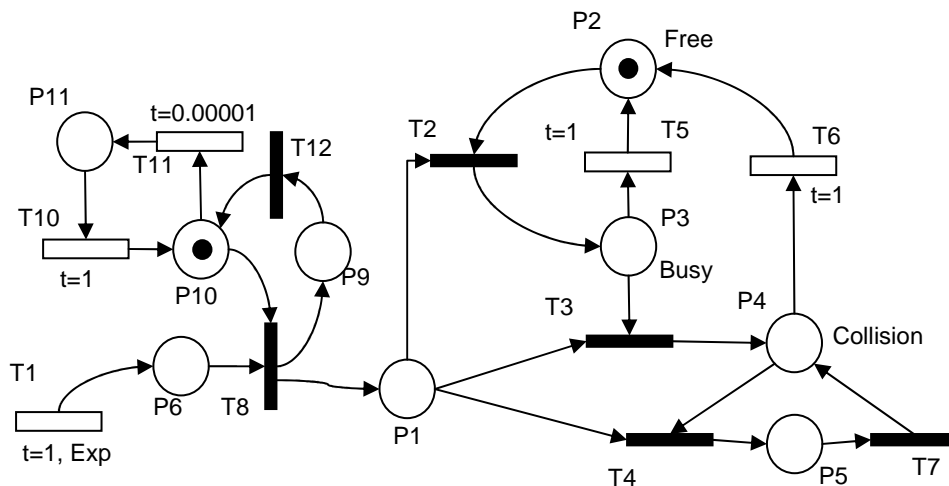


Figure 5.10: Petri Net Model of Slotted Aloha Protocol

Transition T1 is associated with exponentially distributed random variable to ensure that the number of tokens produced is a random variable with Poisson distribution. Transition T12 and place P9 are used to avoid self loop.

Table 5.4: Explanation of the Petri Net Model of Slotted Aloha

Pi	Explanation	Ti	Explanation
P1	A request for packet transmission	T1	Represent packets arrival process
P2	Channel is free	T2	Transmission of the packet started
P3	Packet is being transmitted	T3	A collision occurred
P4	A collision occurred	T4	A packet arrived during collision
P5	A packet arrived during collision	T5	Transmission of the packet completed
P6	Represent output buffer	T6	Channel is free from collision
P10	Transmission of packet can begin.	T8	Sends packet to the output buffer at the start of a new slot
		T10	Represents starting of the new slot
		T11	Avoids transmission of new packet in the middle of a slot

Table 5.4 gives the explanation of various places and transitions while table 5.4 describes about the type and associated delay of these transitions of the Petri net model of slotted Aloha in figure5.10.

Table 5.5: Transition Description in the Petri Net Model of Slotted Aloha

Transition	Transition Type	Associated Delay
T1	Exponential	Variable depending on number of nodes or requests for transmission per unit transmission time
T2, T3, T4, T7, T8, T12	Immediate	0
T5, T6, T10	Deterministic	T (the packet transmission time)
T11	Deterministic	Infinitesimal small so that transition T8 has the priority over T11

Time of transition T11 is equal to 0.00001, which is the little delay ensuring that a conflict does not occur exactly at the end of the transmission. The place P6 represents a transmission buffer from which the data is directly transmitted to the channel.

Following is the net description of the Petri net model of slotted Aloha in terms of various data structures:

```
// Transition Name Vector:
(T1 ;T2 ;T3 ;T4 ;T5 ;T6 ;T7 ;T12 ;T10 ;T11 ;T8 ;)

// Position Name Vector:
(P1;P2;P3;P4;P5;P6;P10;P11;P9;)

// Incidence Matrix:
{
( 0  1  1  1  0  0  0  0  0  0 -1 )
( 0  1  0  0 -1 -1  0  0  0  0  0 )
( 0 -1  1  0  1  0  0  0  0  0  0 )
( 0  0 -1  1  0  1 -1  0  0  0  0 )
( 0  0  0 -1  0  0  1  0  0  0  0 )
(-1  0  0  0  0  0  0  0  0  0  0  1 )
( 0  0  0  0  0  0  0 -1 -1  1  1 )
( 0  0  0  0  0  0  0  0  1 -1  0 )
( 0  0  0  0  0  0  0  1  0  0 -1 )
}

// Marking Vector:
(0 1 0 0 0 0 1 0 0 )

// Arc Type Matrix:
// Code:0 = None; 1 = Normal; 2 = Inhibitor; 3 = Test
{
(0 1 1 1 0 0 0 0 0 0 1 )
(0 1 0 0 1 1 0 0 0 0 0 )
(0 1 1 0 1 0 0 0 0 0 0 )
(0 0 1 1 0 1 1 0 0 0 0 )
(0 0 0 1 0 0 1 0 0 0 0 )
(1 0 0 0 0 0 0 0 0 0 1 )
(0 0 0 0 0 0 0 1 1 1 1 )
(0 0 0 0 0 0 0 0 1 1 0 )
(0 0 0 0 0 0 0 1 0 0 1 )
}

```

```
// Transition Time Model Vector:
// Code:1 = Immediate; 2= Delay;3 = Exponential;
(3 ;1 ;1 ;1 ;2 ;2 ;1 ;1 ;2 ;2 ;1 ;
```

Simulation of the Petri net model of Slotted Aloha

Transition T5 and T6 are deterministic transitions with time equal to 1. The mean time, associated to the transition T1 represents the reciprocal of offered traffic G.

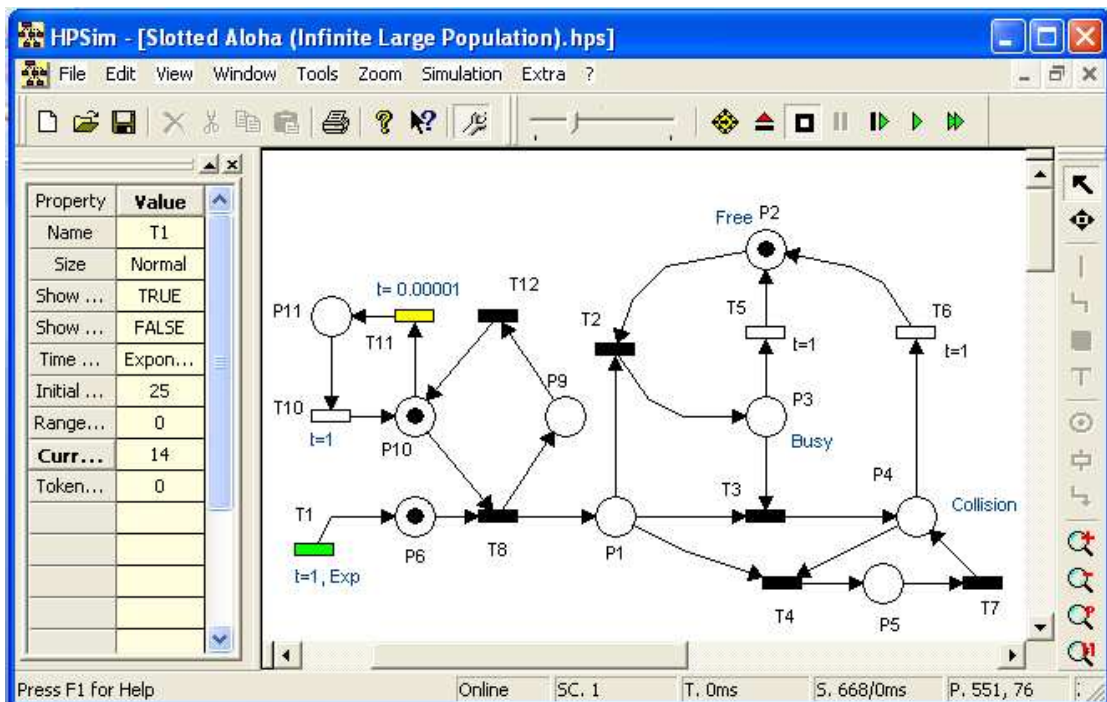


Figure 5.11: Petri Net Model of Slotted Aloha: beginning of the new slot, packet can be transmitted

Firing frequency of transition T2 represents the number of time slots in which channel is not idle. The number of packets which caused a collision and the number of packets which find channel in collision state are represented by transitions T3 and T4 respectively. The sum of firing frequencies of transitions T3 and T4 represent the channel collision ratio, the number of collided packets versus number of offered packets, which is low for small values of G.

Transition T8 is deterministic time transition with time equal to the packet transmission time T. Note that the time associated to transition T11 acts also as a priority function which makes transition T8 a higher priority transition over

T11. This ensures that all the packets waiting for transmission in output buffer (represented by the tokens in place P6) are sent at the beginning of the time slot (see figure 5.12)

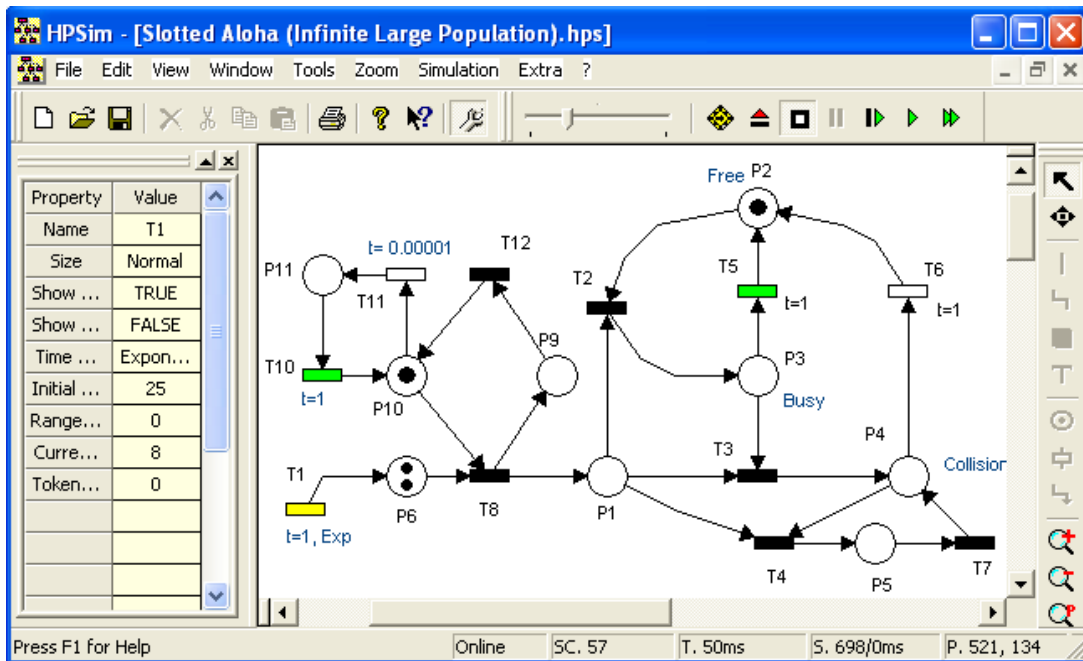


Figure 5.12: Petri Net Model of Slotted Aloha: more than one packet scheduled for transmission in previous slot, both will be transmitted and will result in collision

Petri Net Model of Slotted Aloha with Finite Number of Users

The Petri net model of slotted Aloha considered as $M/D/1/\infty/k$ is similar to model of pure Aloha and is represented in figure 5.13 for $k=2$. Transitions T8, T9, places P7 and P8 (and transition T14, T16 & places P13, P14) ensure that a node does not transmit during its own transmission. Transitions T9 and T14 are deterministic time transitions with delay $t = 1$.

Note that transitions T10, T11 and place P11 are shared by all the users. Time delay of transition T11 which is very small ensures that two packets sent by the same node do not collide.

In slotted Aloha, as the collided packets overlap completely, the same number of collisions wastes less channel capacity than in pure Aloha. In slotted Aloha, a packet is always stored in the outgoing buffer (place P6, P12) until the beginning of next time slot.

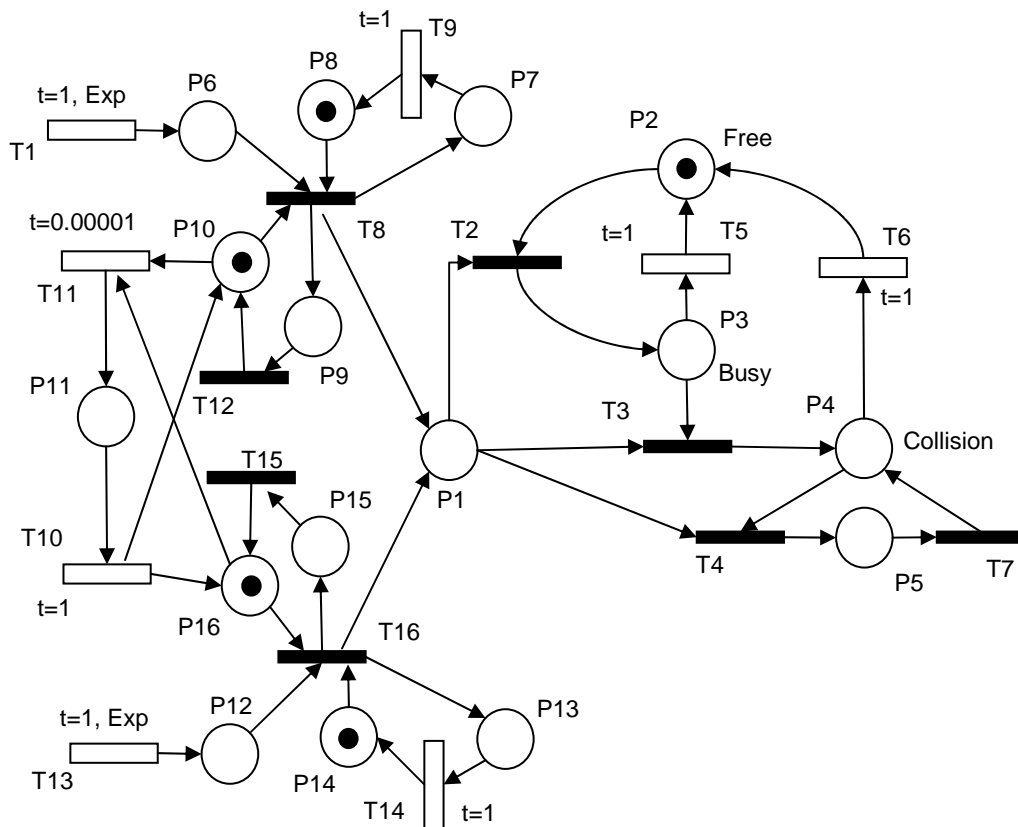


Figure 5.13: Petri Net Model of Slotted Aloha: finite population

5.4 Carrier Sense Multiple Access Protocol

All the carrier sense multiple access (CSMA) protocols share the same philosophy: when a user generates a new packet the channel is sensed and if found idle the packet is transmitted without further ado. When a collision takes place every transmitting user reschedules a retransmission of the collided packet to some other time in the future (chosen with some randomization) at which time the same operation is repeated. The variations on the CSMA scheme are due to the behavior of users that wish to transmit and find (by sensing) the channel busy.

A terminal may, at any one time, either be transmitting or receiving (but not both simultaneously). However, the delay incurred to switch from one mode to the other is negligible. Furthermore, the time required to detect the carrier due to packet transmissions is negligible (that is a zero detection time is assumed).

All packets are of constant length and are transmitted over an assumed noiseless channel (i.e., the errors in packet reception caused by random noise are not considered to be a serious problem and are neglected in comparison with errors caused by overlap interference). The system assumes noncapture (i.e., the overlap of any fraction of two packets results in destructive interference and both packets must be retransmitted). The propagation delay is identical for all source destination pairs and is very small as compared to the packet transmission time.

There are two basic types of CSMA methods [16]: The *persistent CSMA* (also called 1-persistent CSMA), which enters the transmission of packet immediately when the channel becomes free and the *non-persistent CSMA*, which defers the packet transmission by random delay beginning immediately after the channel becomes free.

Petri Net Model for CSMA Protocols

The origin of collisions in the CSMA methods is the propagation delay 'a' which is assumed to be constant for all nodes. The model used for CSMA protocols (figure 5.14) is much similar to previous models of Aloha protocols. Places P1, P2, P3, P4, P5, and transitions T2, T3, T4, T5 T6 have the same meaning as they had before.

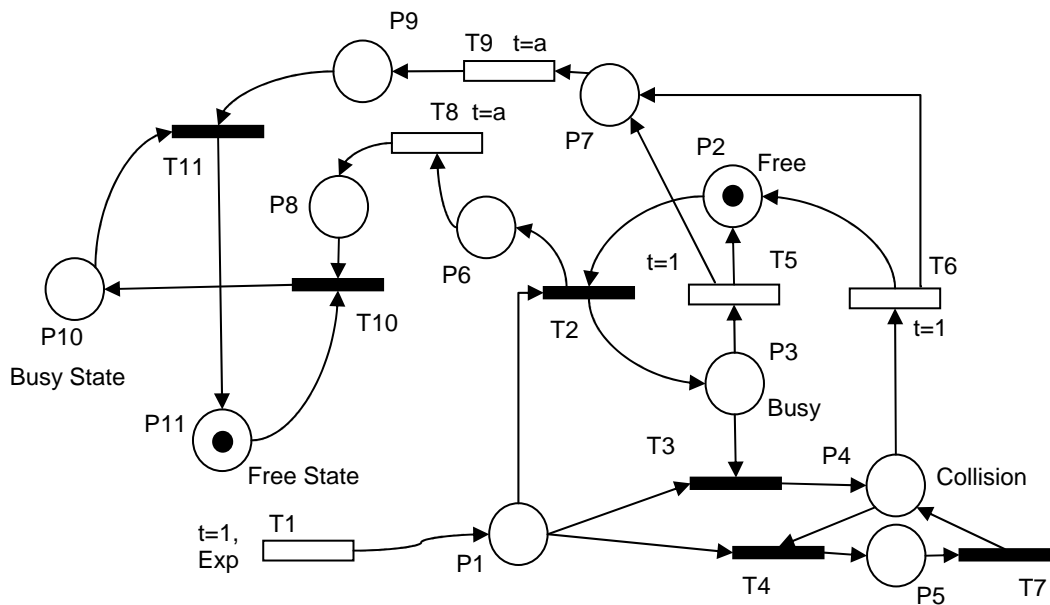


Figure 5.14: Petri Net Model of CSMA Protocol

The model displays its state by two places P10 and P11. A token in place P11 represents a *free state* and a token in place P10 represents *busy state*. The channel becomes *busy* a seconds after an arbitrary node enters the transmission (represented by firing of the transition T2). The channel becomes again *free*, a seconds after the currently transmitting node finishes its transmission (firing of T5).

The transitions are fired according to infinite server semantics. Tokens are not reserved during transition delays i.e. atomic firings. Timed transition which is enabled again does not remember its past status.

Table 5.6: Explanation of the Petri Net Model of CSMA Protocol

Pi	Explanation	Ti	Explanation
P1	Represent output buffer	T2	Transmission of the packet started
P2	Channel is free	T3	A collision occurred
P3	Packet is being transmitted	T4	A packet arrived during collision
P4	A collision occurred	T5	Transmission of the packet completed
P5	A packet arrived during collision	T6	Channel is free from collision and ready for new transmission
P6	Vulnerable period started	T8	End of the vulnerable period
P7	Packet is being propagated	T9	End of the busy period
P8	Vulnerable period ended	T10	Channel enters in busy state
P10	Channel is busy	T11	Channel enters in free state
P11	Channel is in free state		

When the channel is in collision, it will be free 'a' seconds after the completion of transmission of last packet participated in collision shown by firing of transition T6 and then firing of transition T9. Thus, a token in place P10 appears 'a' seconds after the beginning of the transmission and a token in place P11 appears 'a' seconds after completion of the transmission. Transitions T10 and T11 ensure, that the time state of the transitions T8 and T9 is not reset each time a node reads the state of channel.

Table 5.7: Description of the Transitions in the Petri Net Model of CSMA

Transition	Transition Type	Associated Delay
T1	Exponential	Depends on number of nodes or requests for transmission per unit transmission time
T2, T3, T4, T7, T10, T11	Immediate	0
T5, T6	Deterministic	T (the packet transmission time)
T8, T9	Deterministic	a (the propagation delay)

When reading the state, a token in place P10 or P11 is removed and returned back in zero-time. In a steady state, it holds, that firing counts of transition T2 equals to sum of firing counts of transitions T5 and T6. And firing count of transition T6 equals to firing count of transition T3. When two or more nodes seem to start their transmission at the same time i.e. there are more than one token in place P1, the channel changes its state from free state to state of collision.

5.4.1 Non-Persistent CSMA

In the *non-persistent* versions of CSMA (NP-CSMA) a user that generated a packet and found the channel to be busy and operates as follows [17], [18]:

1. If the channel is sensed to be idle, the node transmits the packet immediately.
2. When the node senses the channel being busy, it schedules the retransmission of the packet according to the retransmission delay distribution, and the procedure repeats at this new point in time.

There is an infinite population of users aggregately generating packets according to a Poisson process with parameter λ . All packets are of the same length and require T seconds for transmission. When observing the channel, packets (new and retransmitted) arrive according to a Poisson process with parameter G packets/sec. 'a' is the propagation delay between every pair of users.

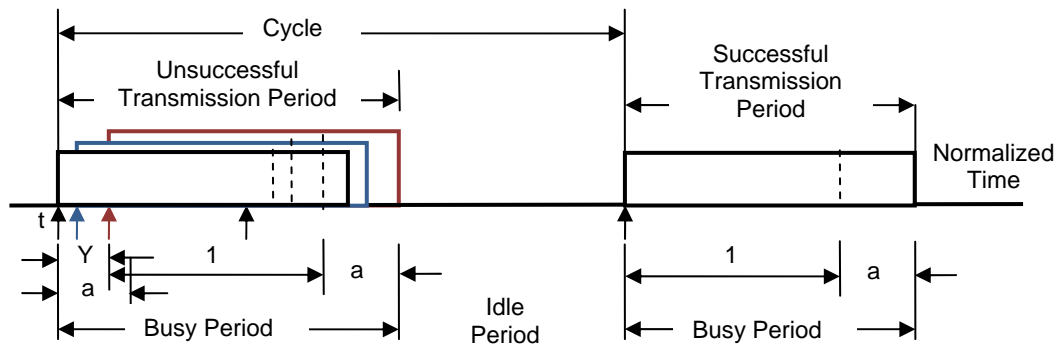


Figure 5.15: Non-persistent CSMA: Packet Timing

Consider an idle channel and a user scheduling a transmission at some time t (see figure 5.15) (a vertical arrow in the figure represents a terminal becoming ready). This user senses the channel, starts transmitting at time t and does so for 1 second (normalized time units); once he is done it will take 'a' additional seconds before the packet arrives at the destination. This transmission therefore causes the channel to be busy for a period of $1+a$ seconds. If at time $t' > t+a$, another user scheduled a packet for transmission, that user would sense the channel busy and refrain from transmission. If, however, some other user scheduled a packet for transmission during the period $[t, t+a]$, that user would sense the channel idle, transmits its packet, and causes a collision.

The initial period of the first 'a' seconds of transmission is called the *vulnerable period* since only within this period is a packet vulnerable to interference. Figure 5.15 depicts a situation in which a packet transmission starting at time t is interfered by other transmission that start in the interval $[t, t+a]$. In the case of a collision the channel will therefore be busy for some (random) duration between $1+a$ and $1+2a$. This period in which transmission takes place is referred to as the transmission period (TP). In the case of NP-CSMA the transmission period coincides with the busy period. Having completed a transmission period the channel will be idle for some time until the next packet transmission is scheduled.

Let $t + Y$ be the time of the occurrence of the last packet arriving in $(t, t + a)$ where a is the normalized propagation delay. The transmission of all the

packets arriving in $(t, t + Y)$ will be completed at time $t + Y + 1$. Only 'a' second later, the channel will be sensed free by all the nodes. Now, any node becoming ready between $t+a$ and $t+Y+1+a$ will sense the channel busy and hence it will reschedule its packet. The interval between t and $t+Y+1+a$ is the transmission period (TP) (in this case also the busy period). Note that there can be at most one successful transmission during a TP.

The equation for the throughput S is expressed in terms of a (the ratio of propagation delay to packet transmission time) and G (the offered traffic rate) can be given as [15], [16]

$$(5.3) \quad S = \frac{Ge^{-aG}}{G(1 + 2a) + e^{-aG}}$$

Petri Net Model of Non-Persistent CSMA

When a node senses the channel to be busy it reschedules its packet and tries again after a random period. Here the rescheduling delay of the packet is assumed to be 'large' compared to the packet transmission time and so to simplify the model the rescheduled packet can be treated as a new packet.

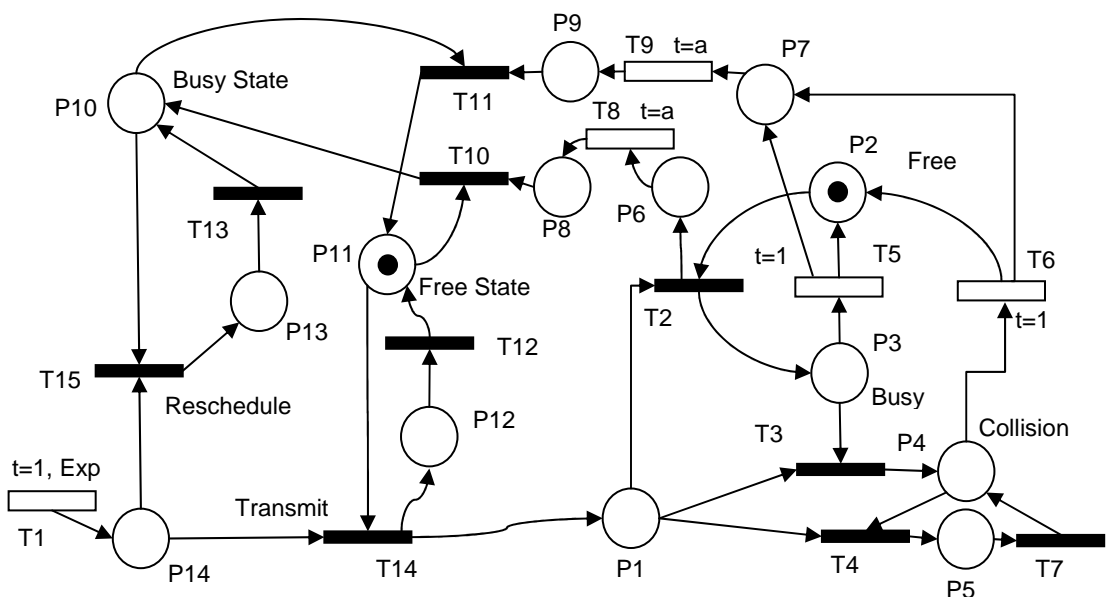


Figure 5.16: Petri Net Model of NP-CSMA

The transitions T12, T13 and the places P12, P13 are used only to avoid the self-loops. Note that the firing count of transition T1 equals to the firing count of transition T14 plus the firing count of transition T15.

Table 5.8: Explanation of the Petri Net Model of NP-CSMA

Pi	Explanation	Ti	Explanation
P7	Packet is being propagated	T9	End of the busy period
P6	Vulnerable period started	T8	End of the vulnerable period
P8	Vulnerable period ended	T10	Channel is busy: A packet is being transmitted
P10	Channel is in busy state	T11	Transmission of the current packet completed
P11	Channel is in free state	T14	Transmits the packets to the channel
P14	Packets waiting for transmission	T15	Reschedules the packet when channel is sensed busy

Table 5.9: Description of the Transitions in Petri Net Model of NP-CSMA

Transition	Transition Type	Associated Delay
T1	Exponential	Depends on number of nodes or requests for transmission per unit transmission time
T2, T3, T4, T7, T10, T11, T12, T13, T14, T15	Immediate	0
T5, T6	Deterministic	T (the packet transmission time)
T8, T9	Deterministic	a (the propagation delay)

Net description of the Petri net model of non-persistent CSMA in terms of data structures

```
// Transition Name Vector:
(T1 ;T2 ;T5 ;T6 ;T7 ;T9 ;T10 ;T11 ;T8 ;T14 ;T12 ;T4 ;T3
;T13 ;T15 ;)
// Position Name Vector:
(P1;P2;P3;P4;P5;P6;P7;P8;P9;P10;P11;P12;P14;P13;)
```

```

// Incidence Matrix:
{
( 0  1  0  0  0  0  0  0  0  0 -1  0  1  1  0  0 )
( 0  1 -1 -1  0  0  0  0  0  0  0  0  0  0  0  0 )
( 0 -1  1  0  0  0  0  0  0  0  0  0  0  1  0  0 )
( 0  0  0  1 -1  0  0  0  0  0  0  0  1 -1  0  0 )
( 0  0  0  0  1  0  0  0  0  0  0  0 -1  0  0  0 )
( 0 -1  0  0  0  0  0  0  0  1  0  0  0  0  0  0 )
( 0  0 -1 -1  0  1  0  0  0  0  0  0  0  0  0  0 )
( 0  0  0  0  0  0  1  0 -1  0  0  0  0  0  0  0 )
( 0  0  0  0  0 -1  0  1  0  0  0  0  0  0  0  0 )
( 0  0  0  0  0  0 -1  1  0  0  0  0  0  0 -1  1 )
( 0  0  0  0  0  0  1 -1  0  1 -1  0  0  0  0  0 )
( 0  0  0  0  0  0  0  0  0  0 -1  1  0  0  0  0 )
(-1  0  0  0  0  0  0  0  0  0  1  0  0  0  0  1 )
( 0  0  0  0  0  0  0  0  0  0  0  0  0  0  1 -1 )
}
// Marking Vector:
(0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 )
// Transition Time Model Vector:
// Code:1 = Immediate; 2= Delay;3 = Exponential;
(3 ;1 ;2 ;2 ;1 ;2 ;1 ;1 ;2 ;1 ;1 ;1 ;1 ;1 ;1 ;)

```

Simulation of Petri Net Model of NP-CSMA

When there is a token in place P11 indicating channel is in free state, all the packets in output buffer will be transmitted (figure 5.17).

If a packet arrives for transmission period in the vulnerable period (presence of a token in place P6) then channel will be sensed free and packet will be transmitted resulting in a collision (figure 5.18).

When a new packet arrives (tokens into place P14 increases by one) and the channel is in busy state the transmission is rescheduled (see figure 5.19).

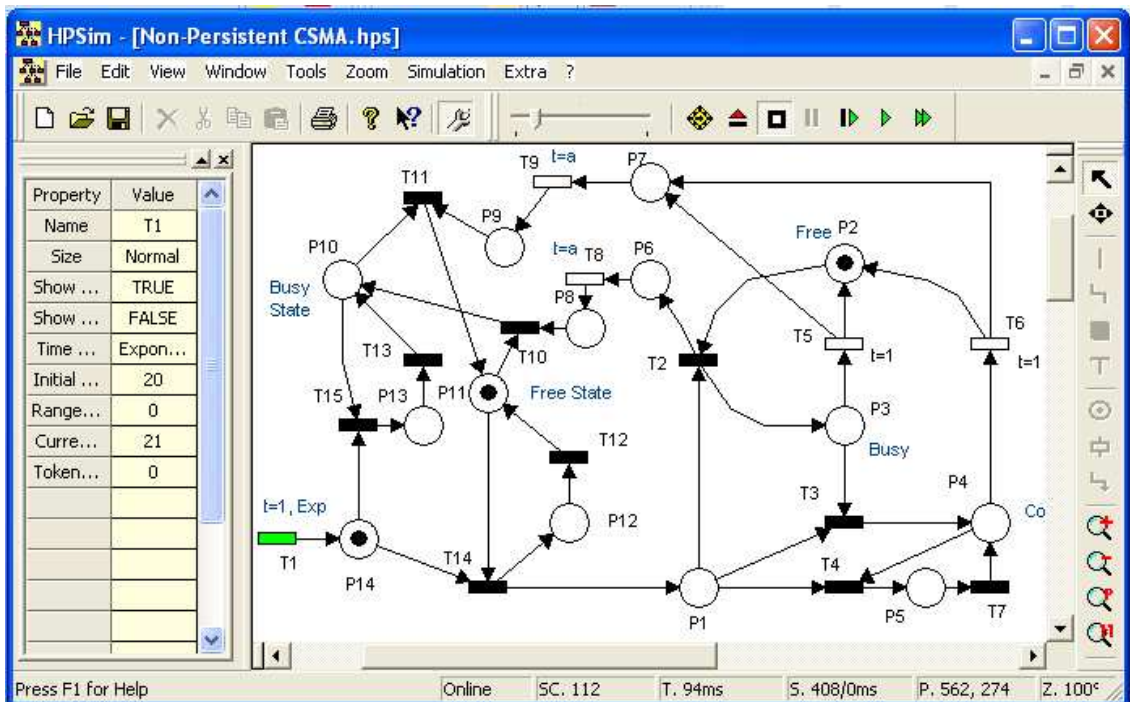


Figure 5.17: Petri Net Model of NP-CSMA: Channel is in free state, packet can be transmitted

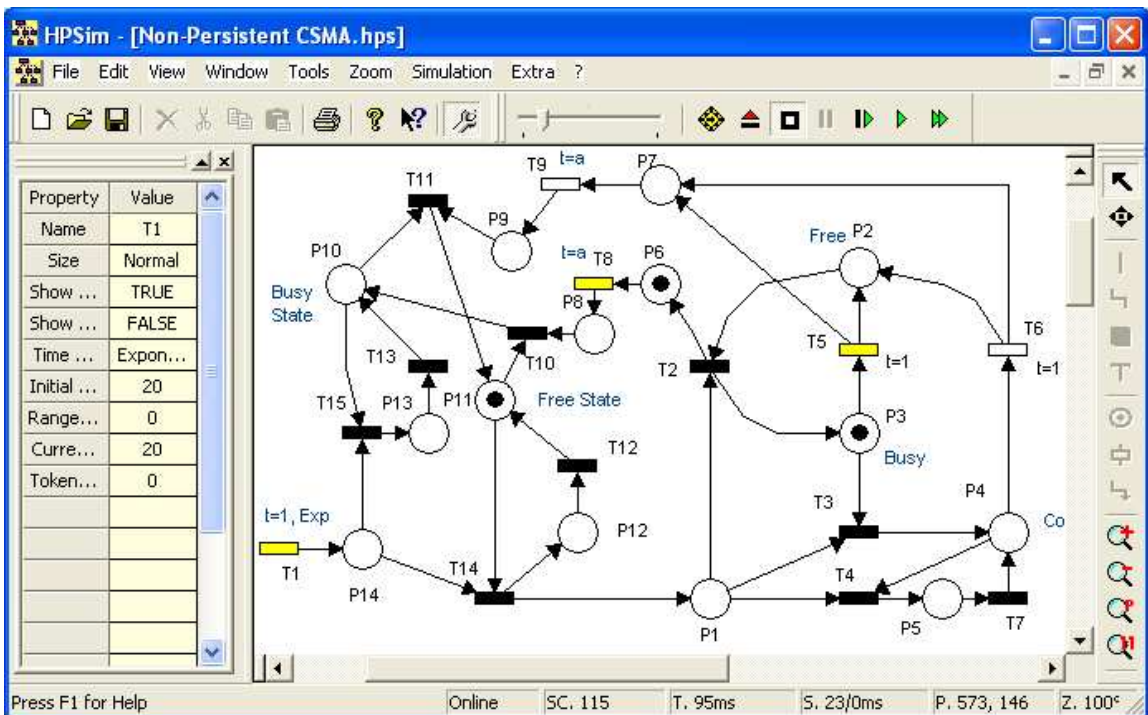


Figure 5.18: Petri Net Model of NP-CSMA: If a new packet arrives in the vulnerable period it will sense channel in free state and will interfere with the current packet in transmission

The transition T15 represents rescheduling of a packet and removes a token from place P14 when fires. The transition T15 fires if a new packet arrives for transmission and another packet is also being transmitted on the channel. Transition T14 fires if only there is a token in place P11 i. e. only if the channel is in free state.

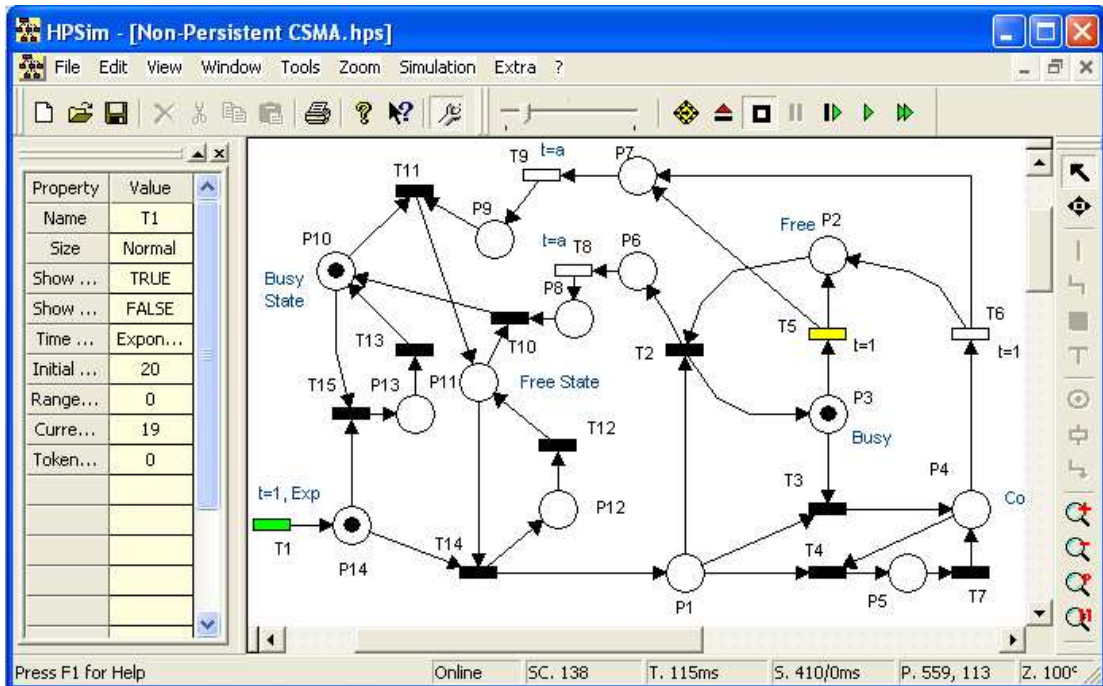


Figure 5.19: Petri Net Model of NP-CSMA: channel is sensed busy, reschedule the packet

Petri Net Model of NP-CSMA with Retrial Mechanism

The non-persistent CSMA, instead of assuming a rescheduled packet as a new packet, with rescheduling mechanism of unsent packets is modeled in figure 5.20. The model is similar to the non-retrial model in figure 5.16. Both models just differentiate in retrial mechanism. Note that for small values of the mean retrial delay, the throughput approaches to the throughput of persistent CSMA. For large values of the mean retrial time and large values of offered traffic G , the number of packets in retransmission buffer (place P18) counts to infinity.

Instead of being dropped as in the non-retrial system, an unsuccessful packet is returned back to the queue. T18 which is a stochastic transition

represent retransmission delay function. It reschedules the packet according to its probability distribution.

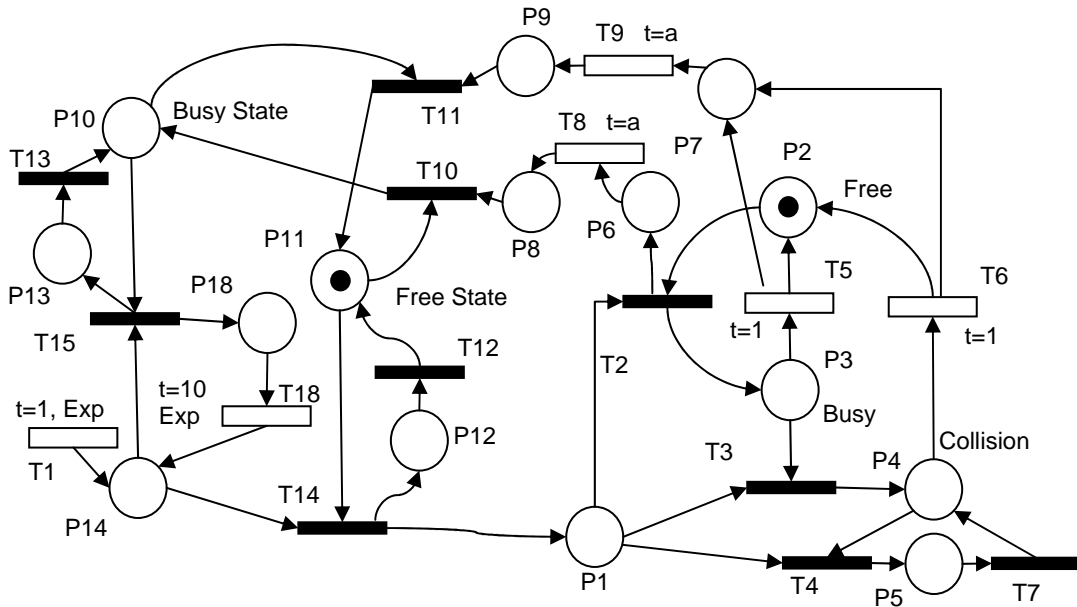


Figure 5.20: Petri Net Model of NP-CSMA with Rescheduling Mechanism

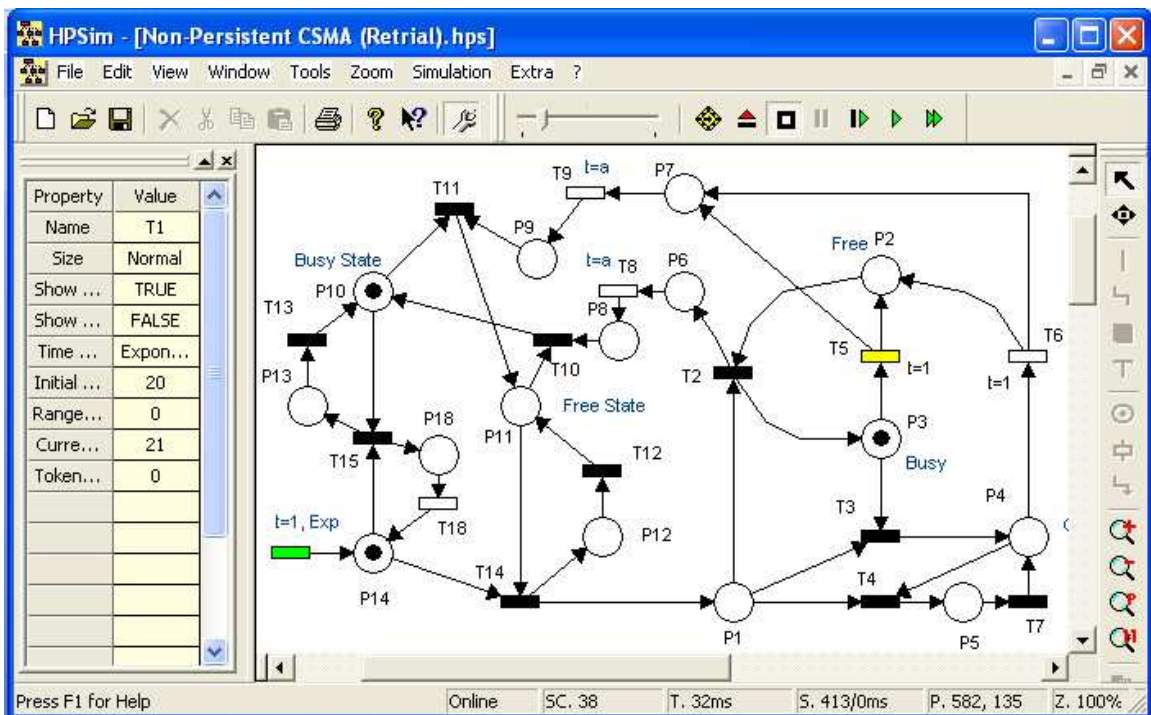


Figure 5.21: Petri Net Model of NP-CSMA: if the channel is sensed busy Packet will not be transmitted.

When there is a token in place P10 and a new packet arrives in output buffer (place P14), transition T15 is enabled (figure 5.21). After firing of transition T15 token will be returned in place P10 and one token will be added to place P18 (figure 5.22). The stochastic timed transition T18 retries the packets from the buffer and places them to output buffer P14. The packet is consequently transmitted or rescheduled again.

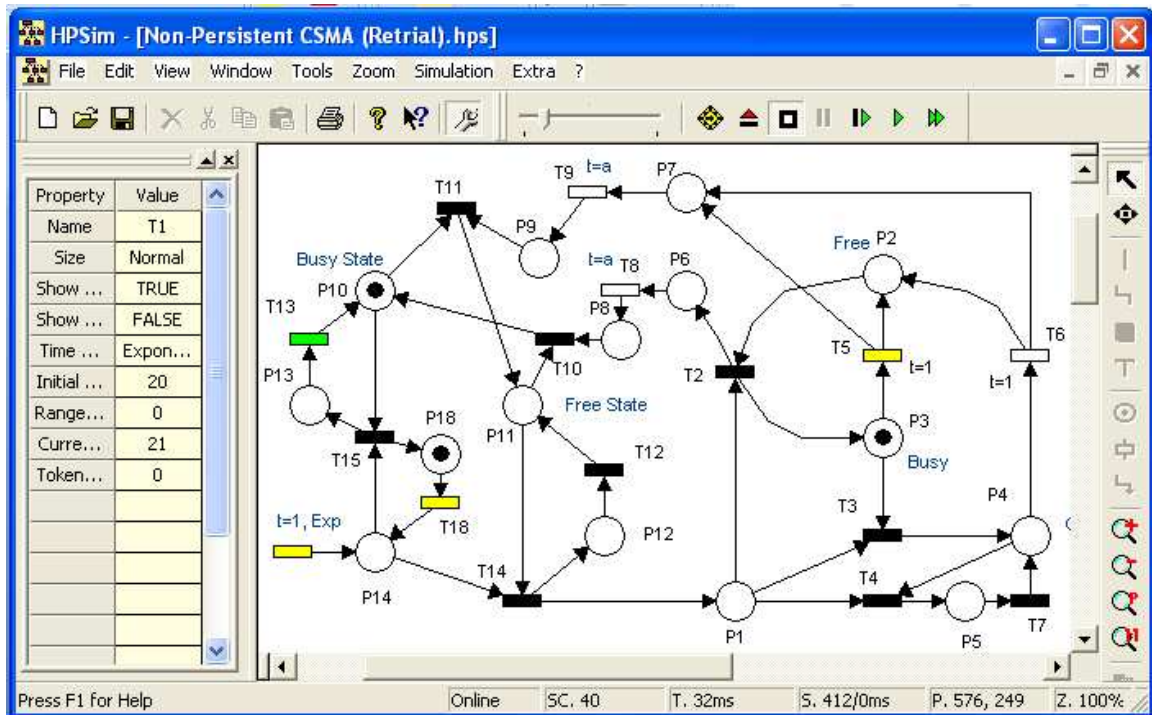


Figure 5.22: Petri Net Model of NP-CSMA: packet is rescheduled after a random period of time when channel is sensed busy.

Petri Net Model of Slotted Version of NP-CSMA

A slotted version of the NP-CSMA [16] is modeled in figure 5.23 in which the time axis is slotted and the slot size is 'a' seconds (the propagation delay). All terminals are synchronized and are forced to start transmission only at the beginning of a slot. When a packet's arrival occurs during middle of a slot, the terminal senses the channel at the beginning of the next slot and operates according to the protocol described above.

The model in figure 5.23 is almost similar to the model of previous model of non-persistent CSMA. To ensure that every transmission starts in slots transitions T16, T17, T18 and places P15, P16, P17 have been used.

Transitions T16 and T17 are deterministic time transitions. Time of transition T17 is just greater than the time of the packet transmission time (say 1.00001) which ensure that a conflict does not occur exactly at the end of the transmission. Transition T14 is deterministic time transition with time equal to the packet transmission time T .

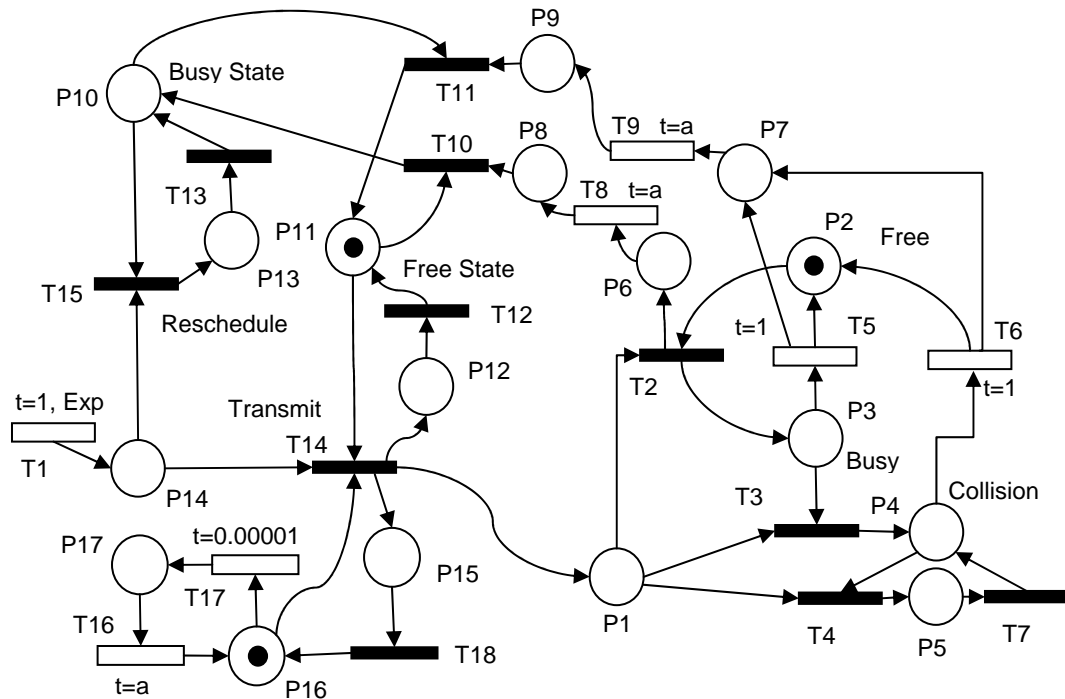


Figure 5.23: Petri Net Model of Slotted NP-CSMA

Note that the time of transition T17 acts also as a priority function which makes transition T14 a higher priority transition over T17. This ensures that all the packets, represented by the tokens in place P14 are sent at the beginning of the time slots. Transition T18 and place P15 are used to avoid self loop.

5.4.2 1-Persistent CSMA

With NP-CSMA there are situations in which the channel is idle although one or more users have packets to transmit. The *1-persistent CSMA* or *persistent CSMA* (1P-CSMA) is an alternative to NP-CSMA that avoids such situations.

A user that senses the channel and finds it busy persists to wait and transmits as soon as the channel becomes idle. Consequently, the channel is

always used if there is a user with a packet. More precisely, a node, which has data for transmission, operates as follows [18]:

1. If the channel is sensed idle, it transmits the packet with probability one.
2. When the channel is sensed busy, the node waits until the channel becomes idle, and then transmits the packet (with probability one—hence, the name of 1-persistent).

Let t be the time of arrival of a packet which senses the channel to be idle with no other packet in the process of transmission (figure 5.24). Any packet arriving in the time interval $[t+a, t+Y+1+a]$ will sense the channel busy and hence it waits until the channel is sensed idle (at time $t+Y+1+a$). At this time, the packet is transmitted. The number of packets accumulated at the end of transmission period (TP) is the number of the arrivals during $1+Y$ seconds. If this total equals to or is greater than two, then a conflict sure occurs in the next TP.

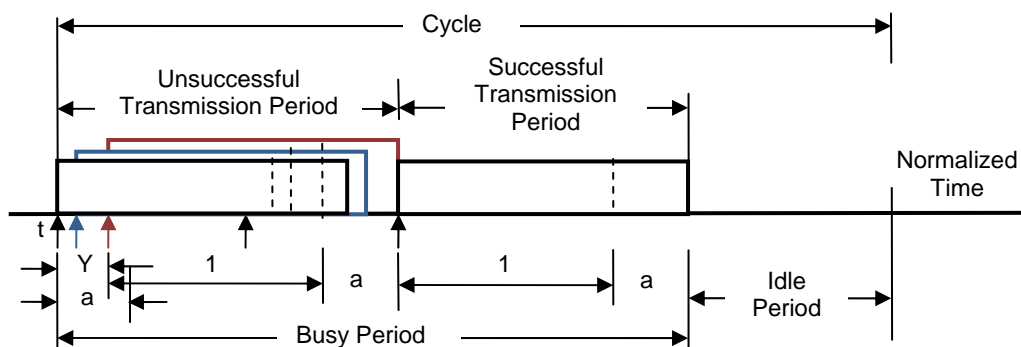


Figure 5.24: Persistent CSMA: Packet Timing

Notice that a transmission period starts either with the transmission of a single packet (call it type 1 transmission period), or with the transmission of at least two packets (call it type 2 transmission period).

The type of a transmission period that follows another transmission period depends on the number of those persistent users waiting for the current transmission to end. For consistency, an idle period is also viewed as a transmission period that starts with no transmitted packets (call it type 0 transmission period).

During a type 0 transmission period no packets are transmitted and during type 2 transmission periods two or more packets are transmitted and collide. Consequently, only type 1 transmission periods may result in a successful transmission. Yet, for a type 1 transmission period to be successful, it is necessary that no packets arrive during its first a seconds that constitute the vulnerable period.

Define the state of the system at the beginning of a transmission period to be the type of that transmission period. These states (0, 1 and 2) correspond to a three-state Markov chain embedded at the beginning of the transmission periods. The knowledge of the system state at the beginning of some transmission period (together with the scheduling points of packets during this transmission period) is sufficient to determine the system state at the beginning of the successive transmission period. The possible transitions among the three states of the embedded Markov chain are depicted in figure 5.25.

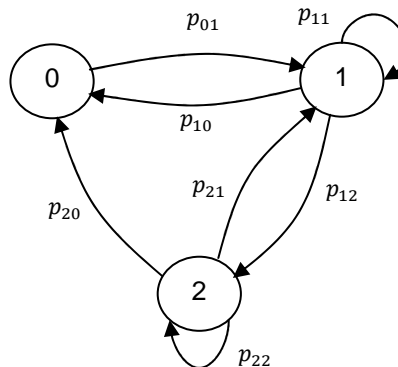


Figure 5.25: State Transitions of 1P CSMA

The channel throughput S achieved by 1-Persistent CSMA is given by

$$(5.4) \quad S = \frac{G[1 + G + aG(1 + G + aG/2)]e^{-G(1+2a)}}{G(1 + 2a) - (1 - e^{-aG}) + (1 + aG)e^{-G(1+a)}}$$

Petri Net Model of 1P-CSMA

Transitions are atomically fired i.e. transitions with time do not reserve the tokens in incoming places. All the transitions follow infinite server semantics. When a transition enables again it does not remember its past status.

Due to the stochastic nature of firing process of transition T1, the marking of place P14 is unbounded because each token in place P14 represents a node, which has a packet for transmission. The throughput of CSMA methods is dependent on the propagation delay a .

Net Description of the Petri net model of persistent CSMA in terms of various data structures

```
// Transition Name Vector:
(T1 ;T2 ;T5 ;T6 ;T7 ;T9 ;T10 ;T11 ;T8 ;T14 ;T12 ;T4 ;T3 ;)

// Position Name Vector:
(P1;P2;P3;P4;P5;P14;P6;P7;P8;P9;P10;P11;P12;)

// Incidence Matrix:
{
( 0  1  0  0  0  0  0  0  0  0 -1  0  1  1 )
( 0  1 -1 -1  0  0  0  0  0  0  0  0  0  0 )
( 0 -1  1  0  0  0  0  0  0  0  0  0  0  1 )
( 0  0  0  1 -1  0  0  0  0  0  0  0  1 -1 )
( 0  0  0  0  1  0  0  0  0  0  0  0 -1  0 )
(-1  0  0  0  0  0  0  0  0  0  1  0  0  0 )
( 0 -1  0  0  0  0  0  0  0  1  0  0  0  0 )
( 0  0 -1 -1  0  1  0  0  0  0  0  0  0  0 )
( 0  0  0  0  0  0  1  0 -1  0  0  0  0  0 )
( 0  0  0  0  0 -1  0  1  0  0  0  0  0  0 )
( 0  0  0  0  0  0 -1  1  0  0  0  0  0  0 )
( 0  0  0  0  0  0  1 -1  0  1 -1  0  0  0 )
( 0  0  0  0  0  0  0  0  0  0 -1  1  0  0 )
}

// Marking Vector:
(0 1 0 0 0 0 0 0 0 0 0 0 1 0 )
```

```
// Transition Time Model Vector:
// Code:1 = Immediate; 2= Delay;3 = Exponential;
(3 ;1 ;2 ;2 ;1 ;2 ;1 ;1 ;2 ;1 ;1 ;1 ;1 ;
```

Table 5.11: Description of the Transitions in the Petri Net Model of 1P-CSMA

Transition	Transition Type	Associated Delay
T1	Exponential	Depends on number of nodes or requests for transmission per unit transmission time
T2, T3, T4, T7, T10, T11, T12, T14	Immediate	0
T5, T6	Deterministic	T (the packet transmission time)
T8, T9	Deterministic	a (the propagation delay)

Simulation of the Petri Net Model of 1P-CSMA

The token in place P14 represents a new request by a node (figure 5.27). Now the node senses the channel and found channel free (presence of a token in place P11) and so it will transmit with probability 1.

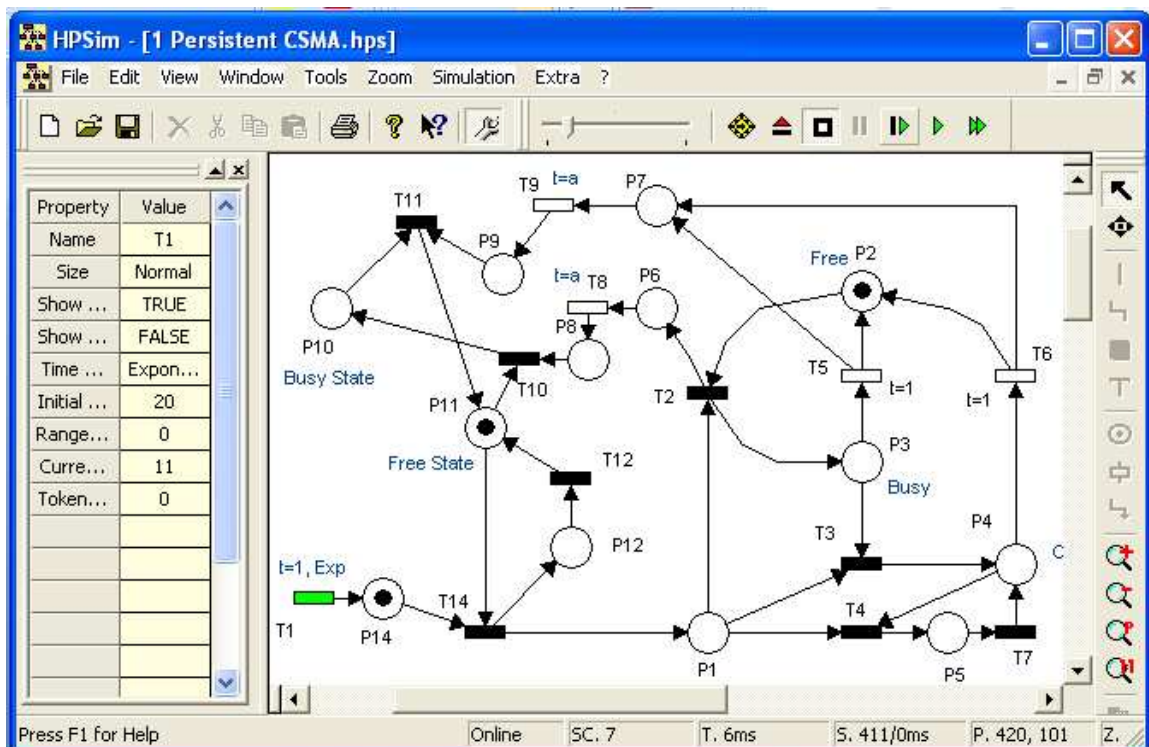


Figure 5.27: Petri Net Model of 1P- CSMA: channel is in free state, new transmission can take place

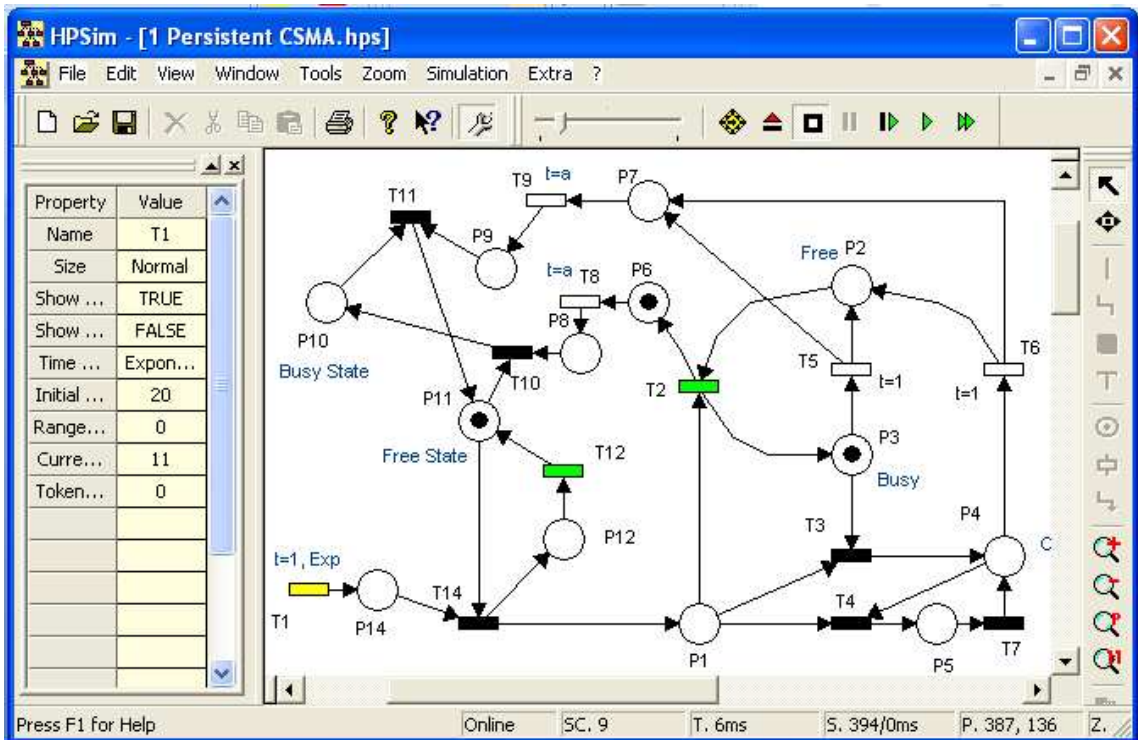


Figure 5.28: Petri Net Model of Persistent CSMA: start of the vulnerable period, if a new packet arrives it will sense channel free and will result in collision

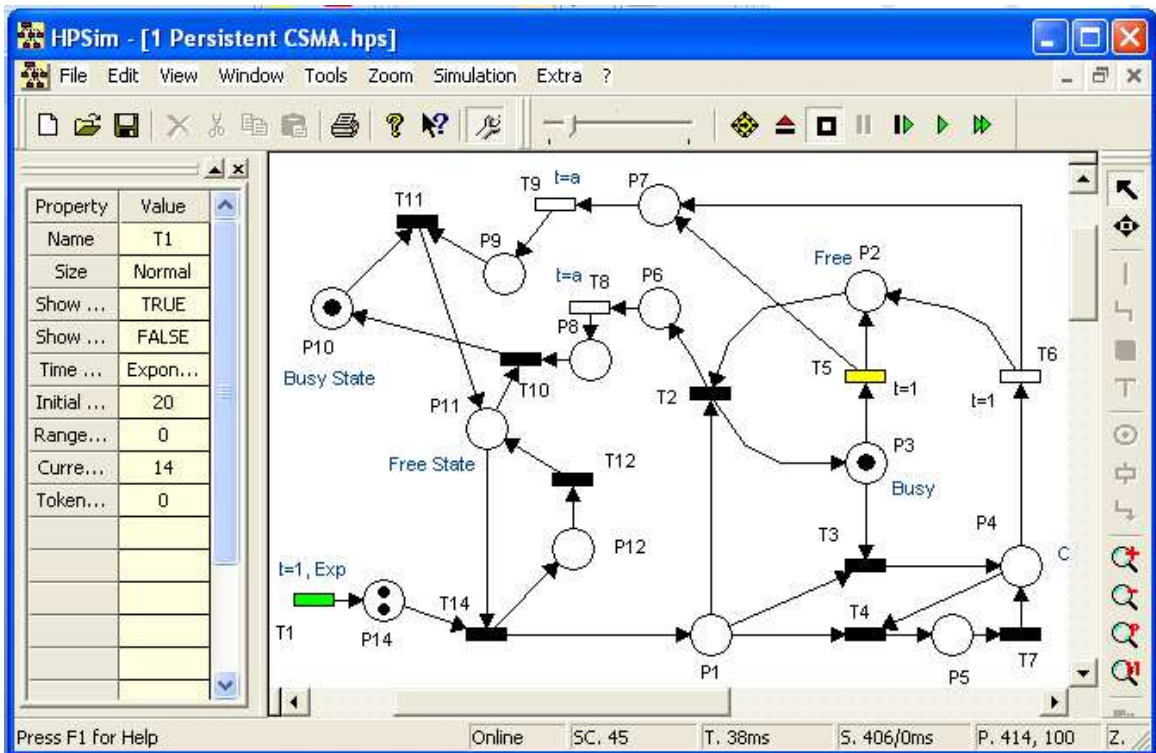


Figure 5.29: Petri Net Model of Persistent CSMA: two or more packets, waiting for transmission, will result in collision.

The vulnerable period begins after the firing of transition T2 which adds a token in place P6 (figure 5.28). Any ready terminal in this case will sense the channel free and will transmit resulting into a collision.

If there are two or more requests waiting for transmission in the busy period (figure 5.29) then both will be transmitted when the channel becomes free (a token will be added at place P11) and both will be collided.

5.4.3 p-Persistent CSMA

The 1-persistent and non-persistent protocols differ by the probability (one or zero) of not rescheduling a packet which upon arrival finds the channel busy. In the case of a 1-persistent CSMA, whenever two or more terminals become ready during a transmission period (TP), they wait for the channel to become idle (at the end of that transmission) and then they all transmit with probability one. A conflict will also occur with probability one. The idea of randomizing the starting time of transmission of packets accumulating at the end of a TP suggests itself for interference reduction and throughput improvement.

More precisely, the protocol consists of the following: the time axis is finely slotted where the (mini) slot size is 'a' seconds [16] For simplicity of analysis, the system is synchronized such that all packets begin their transmission at the beginning of a (mini) slot. A node, which has data for transmission operates as follows:

1. When the channel is sensed idle, it transmits the packet with a probability p . With the probability $(1 - p)$, the node delays the transmission of the packet by 'a' seconds (i.e. one slot size). At this new point in time, if the channel is still sensed idle, the same procedure is repeated otherwise the node schedules the retransmission of packet according to the retransmission delay distribution (as some packet must have started transmission).
2. When the channel is sensed busy, the node waits until the channel becomes idle, and then operates as above.

The parameter p will be chosen so as to reduce the level of interference while keeping the idle periods between any two consecutive non overlapped transmissions as small as possible.

Note that both persistent and non-persistent CSMA methods exist in slotted and un-slotted modifications. Obviously, only slotted version of p -persistent CSMA exists.

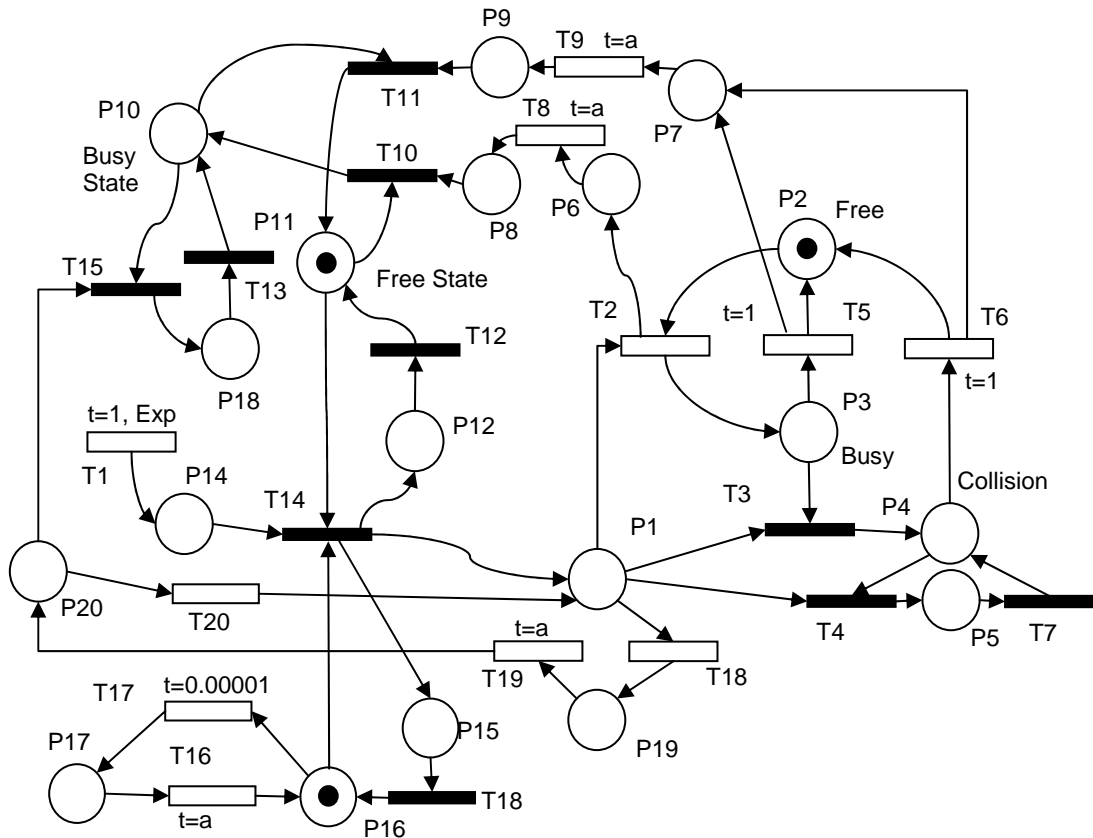


Figure 5.30: Petri Net Model of p -Persistent CSMA

When there is a new request (a token at place P14) and channel is in free state then packet is added to transmission buffer P1 from where it is either delayed with probability $1-p$ (firing of transition T18) or it is transmitted with probability p (firing of transition T2). Note that transition T2 and T20 have lower priority over transitions T18 and T15 respectively due to the infinitesimal time delay associated with them. If the packet is delayed it is added to place P20 after a seconds (firing of transition T19). If the channel is in busy state and there is a delayed packet (presence of a token in place P20), then transition T15 reschedules the packet according to retransmission delay function otherwise it

is added to Place P1. Here it is assumed that the retransmission delay is very large than the packet transmission delay. Transition T16, T17 and places P16, P17 restricts the transmissions only at the beginning of a slot. Transition T16 is a deterministic transition with delay 'a'.

Table 5.12: Transition Description of the Petri Net Model of p Persistent CSMA

Transition	Transition Type	Associated Delay
T1	Exponential	Depends on number of nodes or requests for transmission per unit transmission time
T3, T4, T7, T10, T11, T12, T13, T14, T15,	Immediate	0
T5, T6	Deterministic	T (the packet transmission time)
T8, T9, T16, T19	Deterministic	a (the propagation delay)
T2, T17, T20	Deterministic	Infinitesimal small value
T18	Stochastic	Depends on probability (1-p)

Table 5.13: Explanation of the Petri Net Model of p-Persistent CSMA

Pi	Explanation	Ti	Explanation
P6	Vulnerable period started	T9	End of the busy period
P7	Packet is being propagated	T10	Channel is busy: A packet is being transmitted
P8	Vulnerable period ended	T11	New packet now can be transmitted
P10	Channel is busy	T12	Avoids self loop in the model
P11	Channel is free: new packet can be transmitted	T14	Transmits the packets to the channel
P14	Packets waiting for transmission	T15	Denotes retransmission delay function
P20	Packet have been delayed for next slot	T16	Fires at the beginning of the slot
		T18	Reschedules the packets with probability 1- p

Petri Net Model of Persistent CSMA modeled as MMPP

In figure 5.31, the persistent CSMA is modeled when a node has outgoing buffer of capacity one. The packets arrive only when the outgoing buffer of the node is free.

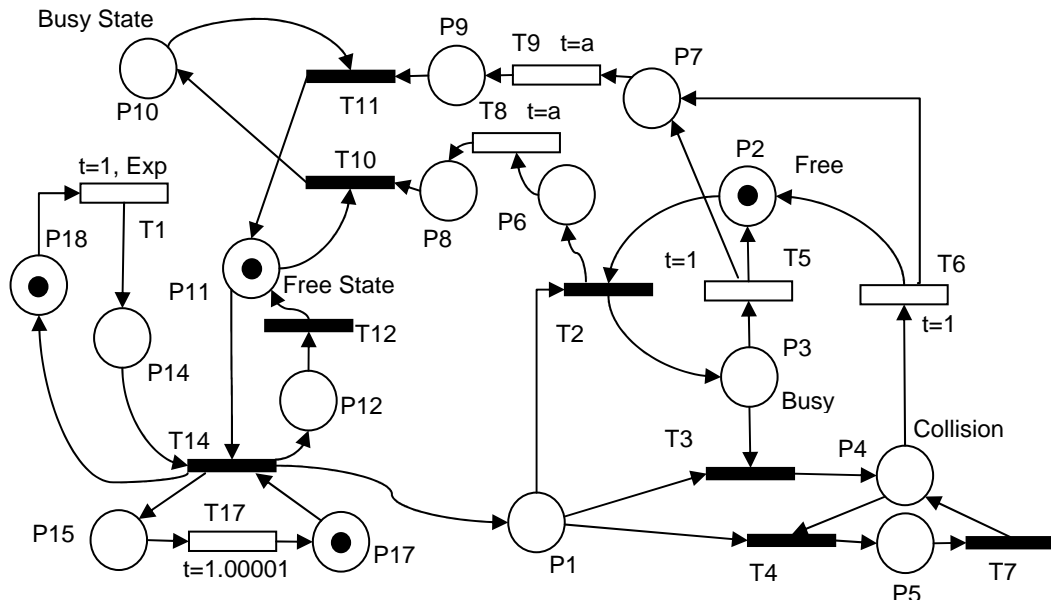


Figure 5.31: Petri Net Model of Persistent CSMA Modeled as Markov Modulated Poisson Process Arrivals

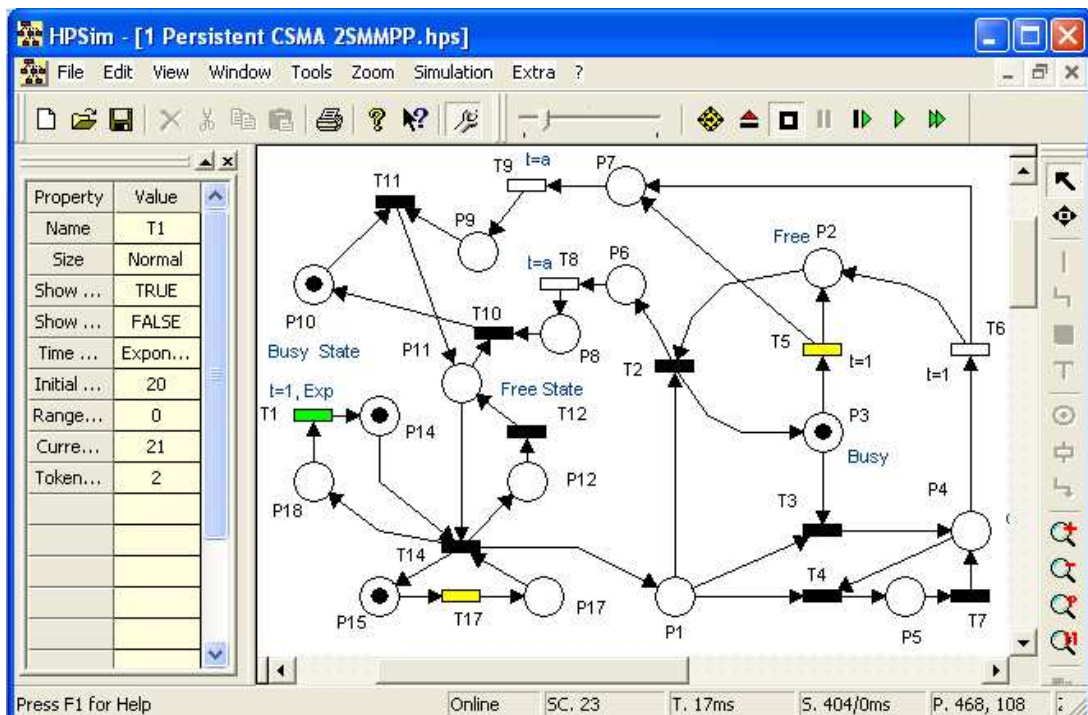


Figure 5.32: Petri Net Model of Persistent CSMA: start of idle period, no new request can arrive until there is an unsent packet in output buffer.

The packet arrival process is assumed to be Poisson when output buffer have no packet. The overall packet arrivals process, defining arrivals of packets to a node, is a special kind of the MMPP, known as two state MMPP which alternates between the idle periods when buffer is full and the regular Poisson arrival process when the buffer is empty.

When the initial marking of P18 is one, a token in the place represents empty outgoing buffer. When there is a token in place P14, no new request will be generated by transition T1 (figure 5.32).

The Petri net model of 1P-CSMA with finite number of user

The Petri net model of 1P-CSMA with finite number of user has been modeled using $M/D/1/\infty/k$ queueing system (see figure 5.33 for $k=2$). Note that the time of transition T17 has taken to be longer than the transmission time. Otherwise, the node could collide its own transmission by firing of the sequence of T17 and then T14.

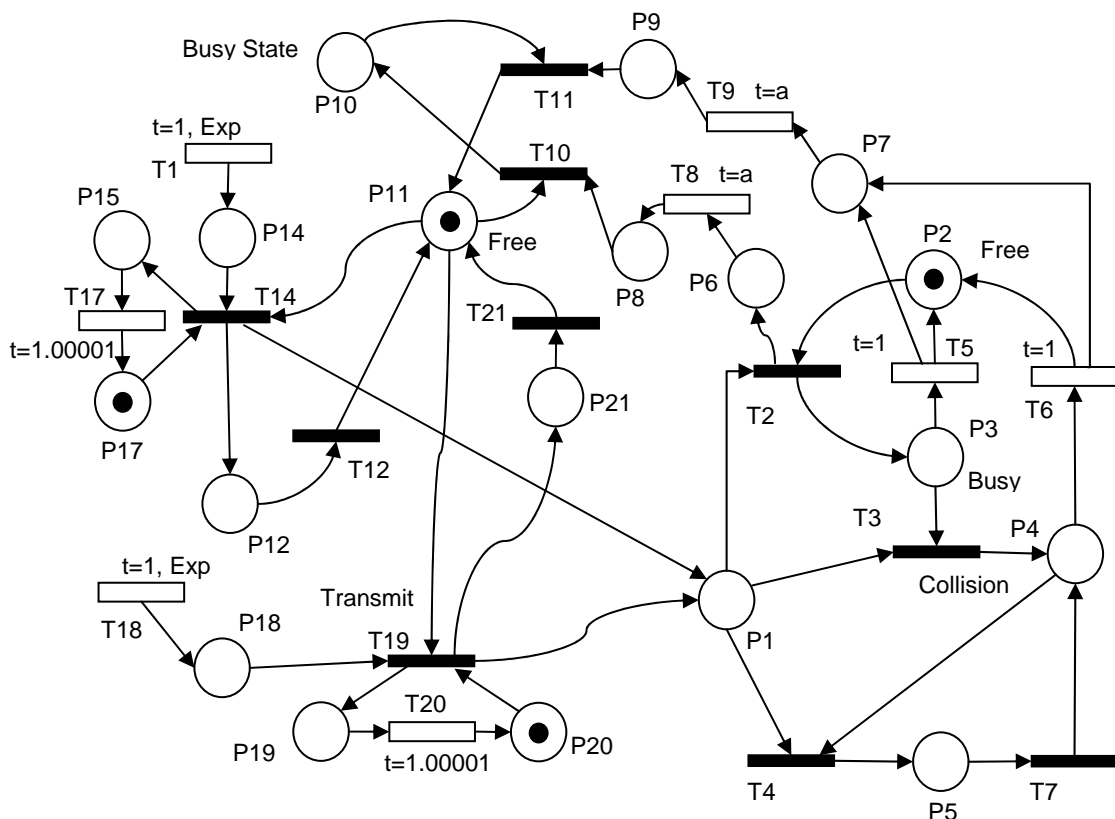


Figure 5.33: Petri Net Model of Persistent CSMA: finite population

Each node can thus send at most one packet per packet transmission time. The packets arrived to node during this time are buffered in place P14. Every node has output buffer of infinite capacity.

Petri Net Model of Slotted persistent CSMA

A slotted version of this 1-persistent CSMA can also be considered by slotting the time axis and synchronizing the transmission of packets in much the same way as for the slotted version of NP-CSMA [15].

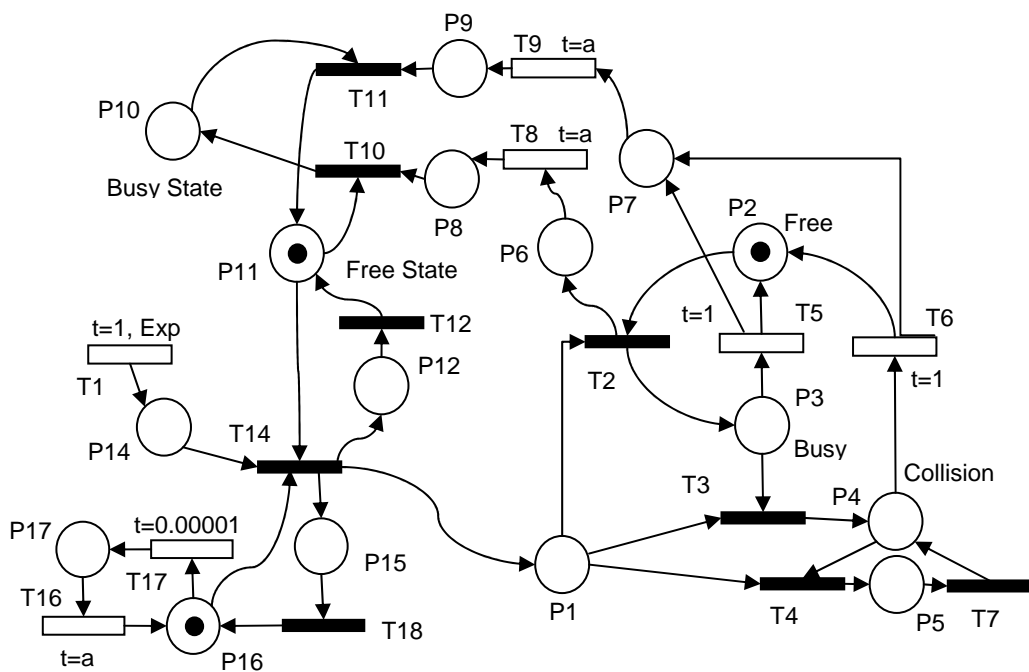


Figure 5.34: Petri Net model of Persistent CSMA: Slotted version

The slotted version of persistent CSMA is modeled in figure 5.34 which is almost similar to the model of persistent CSMA. To ensure that every transmission starts in slots transitions T16, T17, T18 and places P15, P16, P17 have been used.

Transitions T16 and T17 are deterministic time transitions. Delay associated to transition T16 is equal to the packet transmission time T . The time of transition T17 is very small so that transition T14 has higher priority. Also this ensure that two packets transmitted by the same node are not collided.

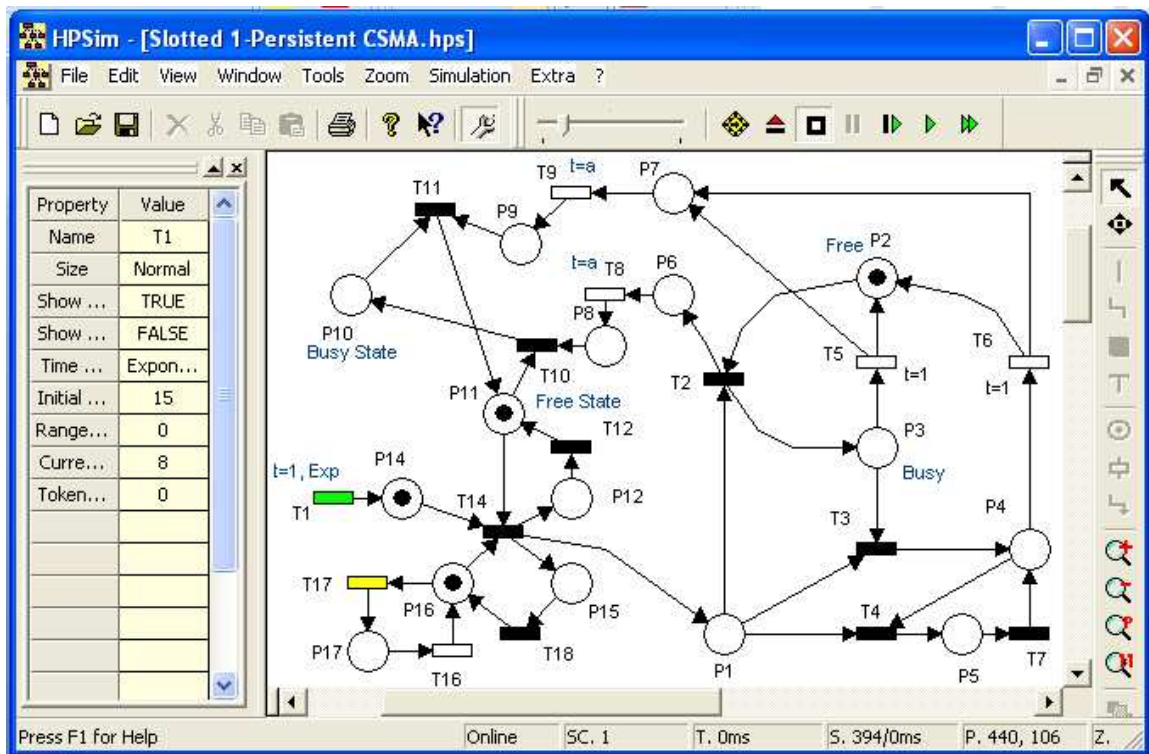


Figure 5.35: Petri Net Model of Persistent CSMA(slotted version): beginning of the new slot, packet can be transmitted

This also gives higher priority to transition T14 and so all the packets scheduled during the previous slot are transmitted simultaneously at the beginning of the time slot. Transition T18 and place P15 are used to avoid self loop.

Chapter 6

Conclusion and Future Work

Petri net, a combination of the simulation and the mathematical numerical methods represents a graphical modeling tool applicable to many systems in similar way as flow charts, block diagrams and graphs. In this dissertation, a novel approach to modeling and analysis of carrier sense multiple access protocols (CSMA) using high level Petri nets has been described.

In the dissertation, models for Open traffic information system (OTIS), Aloha protocols with both pure and slotted versions and finally CSMA protocols using timed Petri nets have been presented. Based on queueing systems and Markov modulated Poisson arrival processes, the dissertation proposed timed Petri net models for persistent and non-persistent CSMA, their slotted versions, and also persistent CSMA as a special case of Markov modulated Poisson process. To validate the models and to analyze the behavior of communication protocols a simulator HPSim have been used.

Throughout the discussion of communication protocols communication channels are considered errorless without capture. In case of infinite large population networks each user is assumed to have at most one packet requiring transmission at any time (including any previously blocked packet).

The results show that communication protocols that typically comprise several generally distributed delays or stochastic processes can be efficiently emulated using Petri net modeling techniques as they have proper constructs to effectively represent all these characteristics. The elementary phenomenon of these protocols has been modeled just by using primitives provided by Petri

nets. Moreover, Petri nets are highly extensible and flexible and any further changes or modifications that may come up in the system can be easily incorporated in the model. Petri nets are also very conducive for simulation which is a vital pre-requisite for analysis of communication networks that are known for their complex characteristics to allow analytical study for their performance analysis.

The Petri net models presented in the dissertation can be further used for performance evaluation of communication protocols. All the time delays associated with transitions in the models either were deterministic or exponentially distributed. This is not always true in a real system. The solution methods for Markov regenerative process can be used to analyze the non-exponentially distributed stochastic Petri nets. When a Petri net contains a large number of tokens, the number of reachable state explodes. In such cases continuous and hybrid Petri nets can be used.

Chapter 7

Publications

During the period of working over this project we interacted with several other people around the globe working on Petri nets. We collected the reviews and worked over the suggestions send by them. We communicated our approach for developing Petri net specifications to the International conference on Information Processing, 2008 at Bangalore, India and the same was accepted for publication. The paper presents the applications of timed Petri nets in the information processing systems.

Conference Details

Conference : International Conference on Information Processing, 2008
at Bangalore, India.

URL : <http://icjip.org/main/>

Paper Title : High Level Petri Nets - An Effective Modeling Tool for
Information Processing Systems

Authors : Manoj Sethi, Delhi College of Engineering, Delhi-42
: Kailash Gupta, Delhi College of Engineering, Delhi-42
: Ankur Gupta, Delhi College of Engineering, Delhi-42.

References

- [1] L. Kleinrock, "On the Modeling and Analysis of Computer Networks", Proceedings of the IEEE, Vol. 81, No. 8, pp 1179-1191, August 1993.
- [2] G.M., Giaglis, "A Taxonomy of Business Process Modeling and Information Systems Modeling Techniques", International Journal of Flexible Manufacturing Systems, 2001.
- [3] Joydeep Bhattacharyya, Adrish Ray Chaudhuri, Swapan Bhattacharya, "A Formal Approach towards Systems Modeling and Verification", Conference on Convergent Technologies for Asia-Pacific Region, Volume 1, pp 178-182, 2003.
- [4] T. Murata, "Petri Nets: Properties, Analysis and Applications", Proceedings of IEEE, Vol.77, No. 4, pp 541-580, 1989.
- [5] Peter J. Haas, "Stochastic Petri Nets for Modelling and Simulation", Proceedings of the 2004 Winter Simulation Conference, 2004.
- [6] Calin Costea, Damian Costea, Voicu Groza, Bogdan Groza, Diana Costea, "Petri Net Reduction", 0840-7789/07, IEEE, 2007.
- [7] Ivo Adan and Jacques Resing, "Queueing Theory", Online Book, 180 pages., 2001, www.win.tue.nl/~iadan/queueing.pdf
- [8] Charles M. Grinstead, J. Laurie Snell, "Introduction to Probability", American Mathematical Society, 1997, ISBN-10: 0821807498, ISBN-13: 978-0821807491.
- [9] Allen Arnold O'reilly, "Probability, Statistics, and Queueing Theory: With Computer Science Applications", 2/e Academic Press Inc. 1990, ISBN: 81-8147-971-8.

- [10] HpSim: A Petri Net Editor cum Simulator, <http://www.winpesim.de/>
- [11] G. Chiola, "Simulation framework for timed and stochastic Petri nets", International Journal on Computer Simulation 1(2): pp. 153-168, 1991
- [12] Gianfranco Balbo, "An Introduction to Generalized Stochastic Petri Nets", 21st International Conference on Application and Theory of Petri Nets, Aarhus, Denmark, June 26-30, 2000.
- [13] Reijers, Hajo, "Design and Control of Workflow Processes-Business Process Management for the Service Industry", Series: Lecture Notes in Computer Science, Vol. 2617 2003, XII, 320 p. Soft Cover ISBN: 978-3-540-01186-6, 2003.
- [14] Rajni Jindal, Ankur Gupta, "Open Traffic Information System", International Conference on Challenges and Development in IT-2008, 4th International Conference, India, 2008, (in press).
- [15] A. Abdelli, N. Badache, "Synchronized Transitions Preemptive Time Petri Nets: A New Model towards Specifying Multimedia Requirements", 1-4244-0212-3/06 IEEE, 2006.
- [16] L. Kleinrock, and F. A. Tobagi, "Packet Switching in radio Channels: Part-I Carrier Sense Multiple Access Modes and Their Throughput Delay Characteristic", IEEE Transactions on Communications, Corn-23, Dec. 1975, pp. 1400-1416.
- [17] Raphael Rom, Moshe Sidi, "Multiple Access Protocols-Performance and analysis", Springer-Verlag. New York, Pages 177, 1990, ISBN: 0-387-97253-6 .
- [18] A. S. Tanenbaum, "Computer Networks", Fourth Edition, Pearson Prentice Hall Inc., ISBN 81-7758-165-1, 2005

Glossary of Notation

PN	Petri net
N	A Petri net model
P, P _i	Place, i th place
T, T _j	Transition, j th transition
T	Set of transitions
P	Set of places
F	Set of arcs
M, M ₀	Marking of Petri net, Initial Marking
A	Incidence matrix
A ⁻ , A ⁺	Pre-condition matrix, Post condition Matrix
SPN	Stochastic Petri Net
MMPP	Markov modulated Poisson process
OTIS	Open Traffic Information System
CSMA	Carrier sense multiple Access
E(X)	Expected value of random variable X
σ ² (X)	Variance of random variable X
DES	Discrete Event Simulation
λ	Mean rate of arrival of queuing system
G	Offered traffic
S	Throughput of the channel
a	The propagation delay
T	The packet transmission time
TP	Transmission Period