Improved Beacon-Stuffing and Quantification of Criticisms in Analytic Hierarchy Process for Infrastructure WLANs

THESIS

Submitted in partial fulfillment
of the requirements for the degree of
DOCTOR OF PHILOSOPHY

by

VISHAL GUPTA

Under the Supervision of
Prof. Mukesh Kumar Rohil

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

2013

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

CERTIFICATE

This is to certify that the thesis entitled **"Improved Beacon-Stuffing and Quantification of Criticisms in Analytic Hierarchy Process for Infrastructure WLANs"** and submitted by **Vishal Gupta** ID No **2008PHXF032P** for award of Ph.D. degree of the Institute embodies original work done by him under my supervision.

Signature of the Supervisor

PROF. MUKESH KUMAR ROHIL
Associate Professor
Department of Computer Science and Information Systems
BITS Pilani, Pilani Campus
Pilani - 333 031 (Rajasthan) INDIA

Date:

ACKNOWLEDGEMENTS

First and foremost, I express my sincere appreciation to my supervisor, Prof. Mukesh Kumar Rohil, who has guided me all the way to this Ph.D. thesis. Being an excellent mentor, he had patiently led me to this research. I thank him for constantly motivating me to work consistently towards achieving the research goals. His continuous, timely, and genuine support is indispensable for me to finish the Ph.D. study.

Special thanks are due to Prof. B. N. Jain, honorable Vice Chancellor; Prof. G. Raghurama, Director; Prof. R. N. Saha, Deputy Director; Prof. G. Sundar: Deputy Director; Prof. S K Verma, Dean, Academic Research Division, Birla Institute of Technology and Science, Pilani (BITS Pilani), (Raj.), for giving me an opportunity to pursue my PhD program.

Furthermore, I would like to thank Prof. Rahul Banerjee, Professor, and Dr. K. Hari Babu, Assistant Professor, Department of Computer Science and Information Systems, BITS Pilani, for being the Doctoral Advisory Committee (DAC) members for this research work. The time they have devoted in examining the draft thesis and giving insightful comments and invaluable suggestions have helped in improving this work.

Special thanks to Prof. Navneet Goel, Head of Department, Department of Computer Science and Information Systems, BITS Pilani for encouraging research related activities in the department and cultivating an open and free research environment. I would also like to extend my sincere thanks to Prof Shan Balasubramaniam, Dean Academic and Resource Planning, BITS Pilani for his support and advice. Thanks to Prof J P Misra, chief Information Processing Center unit, BITS Pilani for providing computing facilities for this research.

I wish to thank all those who helped me directly or indirectly during this entire work.

Vishal Gupta

ABSTRACT

With the widespread acceptability of 3G services, increasing demand of wireless multimedia and other high data rate services, and widely available mobile devices which can connect to 3G and WLAN simultaneously, the integration of 3G network and WLANs is highly significant. Moreover, to make this multi-access solution effective, the integrated solution should provide seamless mobility and session continuity between access technologies. 3G-WLAN interworking aims to combine the ubiquitous coverage of 3G network and the high speed data service offered by WLANs into a seamless wireless network. Within the scope of network-assisted downward vertical handover, the corresponding proposed decision models mainly differ with respect to the decision parameters. To make these models practically useful requires making the values of these parameters available to the mobile device. Another constraint towards making a seamless downward vertical handover is to make these values available without enforcing the mobile device to connect to the network. Now, with this constraint, since beacon frame is the only communication frame from the AP to the mobile device, stuffing additional informative contents in it is a viable (and in fact *essential* for push based model unless another special frame is designed for the purpose) option. The research work in this doctoral thesis proposes an improved technique to stuff additional information in three fields of an IEEE 802.11-2012 standard compatible beacon frame: a) Basic Service Set Identification (BSSID), b) Length field of Information Elements, and c) Vendor Specific Information Element. Moreover, the proposed technique does not limit the size of information to be stuffed up to a single beacon frame only. Rather, motivated by its advertising perspective, which makes it suitable for Location Based Advertising, the proposed technique in this research work allows the stuffing of large information contents in successive beacon frames.

Also, Analytic Hierarchy Process (AHP) is a Multi Criteria Decision Making (MCDM) technique used in many diverse fields for selecting a best alternative. Many authors have used it for deciding upon the best alternative network during downward vertical handover decision. Although AHP is a popular technique, it is also criticized in the literature for allowing the consideration of "contradictory judgement matrices". The research work in this thesis also focuses on measuring these criticisms with respect to the percentage of contradictory matrices which are allowed to be considered by AHP. Furthermore, the quality of priority vector for contradictory and non-contradictory matrices is compared on the basis of qualitative metric, i.e. Rank Reversal for Scale Inversion, and 13 different quantitative metrics given under the common framework of "aggregated deviation". Based upon the results of this comparison, a feedback based technique for getting the values in pair-wise comparison matrix is proposed. Using it, the decision maker can correct the contradictory decisions as soon as two decision elements are compared; and do not have to wait until all the entries in the pair-wise comparison matrix are filled. To show the usefulness of the results, a hypothesis is proposed using which the user-preferences for network selection during downward vertical handover can be captured. This model uses AHP (both *with* and *without* feedback based mechanism) for capturing the user preferences and the parameters for it are mainly derived from the 9^{th} amendment to the IEEE 802.11-2007 compatible beacon frame (i.e. IEEE 802.11u).

With respect to stuffing additional non-standard information and assuming that first 30 frame body Information Elements are present in the beacon frame, results show that the resulting bandwidth will be in the range of 294 Kbps to 1775 Kbps; as against 229 Kbps resulting from the technique proposed earlier. With respect to measuring contradictory judgement matrices, results show that as the order of judgement matrices increases from 3x3 to 9x9, the percentage of contradictory judgement matrices also increases from 0.7% to 72.87% respectively. Using the proposed feedback based technique to enter the values in pair-wise comparison judgement matrix, theoretically the contradictory judgement matrices should be reduced by 100%. But since it is proposed to avoid (and not eliminate) the contradictory matrices, practically while capturing user-preferences these are reduced by 60.27% for order 4x4 and 65.74% for order 5x5.

TABLE OF CONTENTS

Content	Page

Content **Page**

Content **Page**

Content	Page

Content	Page

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Expanded Form
2G	Second Generation wireless telephone technology
3G	Third Generation telephone technology
3GPP	3rd Generation Partnership Project
AAA	Authentication, Authorization, and Accounting
AES	Advances Encryption Standard
AHP	Analytic Hierarchy Process
AP	Access Point
ANQP	Access Network Query Protocol
ASRA	Additional Step required for Access
BSS	Basic Service Set
BSSID	Basis Service Set Identification
CI	Consistency Index
CR	Consistency Ratio
CRC	Cyclic Redundancy Check
CSMA/CA	Carrier Sense Multiple Access with Collision Avoidance
DHCP	Dynamic Host Configuration Protocol
DS	Distribution System
EAS	Emergency Alert System
EID	Element ID
ESR	Emergency Services Reachable
ESS	Extended Service Set
FCS	Frame Check Sequence
FCoS	Saaty's Fundamental Comparison Scale
GHz	Giga Hertz
IBSS	Independent Basic Service Set
IEEE	Institute of Electrical and Electronics Engineers
Kbps	Kilo bits per second

Abbreviation	Expanded Form
LAE	Least Absolute Error
LBS	Location Based Service
LLAE	Logarithmic Least Absolute Error
LLS	Logarithmic Least Square
LLWAE	Logarithmic Least Worst Absolute Error
LLWS	Logarithmic Least Worst Square
LSM	Least Square Method
LWAE	Least Worst Absolute Error
LWS	Least Worst Square
MAC	Medium Access Control
MADM	Multi-Attribute Decision Making
Mbps	Mega bits per second
MIMO	Multiple Input Multiple Output
MCDM	Multi-Criteria Decision Making
MDO	Mobile Data Offloading
OI	Organizational Identifier
OUI	Organizational Unit Identifier
PCF	Point Coordination Function
PHY	Physical Layer
PWLAE	Preference Weighted Least Absolute Error
PWLS	Preference Weighted Least Square
PWLWAE	Preference Weighted Least Worst absolute Error
PWLWS	Preference Weighted Least Worst Square
QoS	Quality of Service
RADIUS	Remote Authentication Dial In User service
RR-SI	Rank Reversal Problem of Scale Inversion
RSS	Received Signal Strength
RTT	Round Trip Time
SMS	Short Messaging Service

Abbreviation	Expanded Form
SSID	Subscription Service Identifier
SSP	Subscription Service Providers
SSPN	Subscription Service Provider Network
STA	Mobile Station
TOPSIS	Technique for Order of Preference by Similarity to Ideal Solution
UESA	Unauthenticated Emergency Service Accessible
Wi-Fi	Wireless Fidelity
WLAN	Wireless Local Area Network
WPM	Weighted Product Model
WSM	Weighted Sum Model

List of Symbols & Notations

Abbreviation	Meaning
$\sum$	Summation
π	Product
λ_{max}	Maximal Eigen Value
$[P_i/P_j]$	Error free judgement matrix corresponding to Priority Vector P_i and P_j
a_{ij}	Performance of alternative a_i with respect to decision criteria c_j
$A_{K\,j}$	Weight of j^{th} alternative for K^{th} criterion
D_i	i^{th} diagonal above the main diagonal in a judgement matrix
ln (x)	Natural Logarithm of x
max(Values)	maximum value among the set of Values
N x N	Order of judgement Matrix
N_{ut}	the number of elements in the upper triangle of NxN matrix
N_{ex}	Total number of possible matrices with FCoS for NxN judgement matrix
P	Principal Eigen Vector
p_i	Weight of i^{th} priority vector element
R_i	i^{th} row in a judgement matrix, where $1 \leq i \leq (N-1)$.
P_i^G	Global priority of i^{th} alternative
V_p	Preference Vector
W	Judgement Matrix
w_{ij}	weight in i^{th} row and j^{th} column of the judgement matrix

CHAPTER 1

INTRODUCTION

Wireless cellular network now covers most of the populated-part of the world. It is predicted that mobile subscribers worldwide will reach 6.9 billion by the end of 2013 and 8 billion by the end of 2016 [*portioresearch.com, 2013*]. In fact, in developed countries, there is at least one cell phone subscription per person [*mobithinking.com, 2013*]. Statistics had shown that it started reaching saturation levels in 2010 itself; with an average 116 subscriptions per 100 inhabitants at the end of 2010, just a marginal growth of 1.6% from 2009-2010 [*The world in 2010, 2010*].

For many years in the past, the cutting edge services provided by cellular network were voice and SMS. But mobile phones – especially smart phones and iphones - are now poised to take over the traditional information access devices as the dominant platform for accessing the information. The nature of data transformed from ranges of conventional and plain text to emails, multimedia and chats. Subscribers today have an easy access to streaming media for video and audio. So, now apart from voice services, it is the data services which govern the telecom industry. As a result, many countries have already started offering 3G services and people are moving rapidly from 2G to 3G platforms in both developed and developing countries. In fact, in 2010, there were 143 countries offering 3G services commercially, compared to 95 in 2007 [*The world in 2010, 2010*].

Though switching to 3G network seems to be a promising solution, the predicted statistics about the surge in data usage show that the expected growth of data services cannot be sustained *only* by 3G [*mobithinking.com, 2013*]. More specifically, following are some issues which cannot be overlooked:

- Increasing Network Traffic: 3G offers rich data services and the bandwidth consumption for these services are not expected to slowdown. Mobile data traffic is expected to grow at a Compound Annual Growth Rate (CAGR) of 74 percent from 2012 to 2017, which is a 16 fold growth as against the projected CAGR of

66% during the same time period (i.e. 2012 to 2017) [*Cisco Visual Networking index, 2013*].

- Increasing usage of Smart Phones: Accelerated adoption of Smart phone by mobile phone subscribers is the major cause for the unexpected data surge. Operators are seeing increasing data traffic driven by the growth of Smart phones and other connected devices that offer ubiquitous Internet access [*mobithinking.com, 2013*].
- Spectrum is costly and scarce: As the world prefers increasingly wireless connections, allocation of the available spectrum to each technology becomes increasingly contentious. This requires that the means of allocation of radio communications resources to satisfy our future need for increasingly dense, fast, flexible mobile communications networks should be done judiciously. Because of this scarcity, the organizations allotting the spectrum to vendors charge them heavily.

To tackle the above issues with 3G networks, three immediate possibilities are:

a) Scaling the network: It refers to building more cell towers and/or increasing the backhaul capacity to tackle increasing network usage.
b) Optimization: It refers to optimizing the radio and backhaul usage.
c) Mobile Data Offloading (MDO): It is the use of other (preferably “complementary”) network technologies for delivering data originally targeted for cellular users with comparatively much better performance capability. These networks can function with the macro-cellular network as an adjunct network either operating independently or as an overlay network. For the end users MDO contributes in higher bandwidth availability and reduced data services cost due to reduction in the congestion of the cellular networks.

Out of the above three possibilities, MDO has emerged as a most promising solution [*A Ghosal, 2010*]. The two promising candidate technologies for Mobile Data Offloading are *Fem-to-Cell* and *Wi-Fi*. Comparing these two shows that a typical customer who don't want to pay for a dual mode smart phone or iphone, is a little technical savvy, want to get good availability of network at indoor, Fem-to-cell is a good solution. Rather, considering the continuous increase in usage of smart phones and iphones [*Cisco Visual Networking index, 2013*], projected increase in the data requirements of mobile subscribers in future, Wi-Fi is the clear winner [*A Ghosal, 2010*].

To leverage upon the alternate path of delivering data, as required by MDO, the need of interworking between 3G and 802.11 networks was recognized by research community and 3G service providers [*Buddhikot et al, 2003*]. Independent research articles proposed the design and evaluation of 3G-WLAN interworking architectures [*Tsao & Lin, 2002a; Mohanty, 2006; Jaseemuddin, 2003; Song & Lee, 2007; Pinto, Bernardo, & Sobral, 2006;Tsao & Lin, 2002b; Liu & Zhou, 2005a; Liu & Zhou, 2005b; Xiao, Leung, Pan, & Du, 2005; Ruggeri, Iera, & Polito, 2005; Lehr & McKnight, 2003; Buddhikot et al, 2003*]. Also, the organizations like 3GPP and IEEE started the development of standards based on interworking between 3G and 802.11 networks [*3GPP TR 22.934; 3GPP TR 22.935; 3GPP TR 22.937; 3GPP TS 22.234; 3GPP TS 23.234; 3GPP TS 23.237; IEEE standard 802.11u, 2011*].

1.1 Vertical Handover in 3G-WLAN interworking environment

In 3G-WLAN interworking environment the concept of Always Best Connected (ABC) has become prevalent, which is to be connected anytime and anywhere to the best possible network. One of the major issues to be addressed for ABC is vertical handover, a handover management scheme between different access networks.

Several proposals [*Kassar, Kervella, & Pujolle, 2008; Kassar, Kervella, & Pujolle, 2008a; Stevens-Navarro, Lin, & Wong, 2008*] split the Vertical handover process into following three phases:

- **Phase 1**: *Handover Information Gathering phase*: This phase corresponds to gathering the necessary information about the network, required during handover decision phase.
- **Phase 2**: *Handover decision*: Its purpose is to decide whether to change the access network or not; and if yes then out of all the discovered networks to *which* and *when* to perform the handover.

- **Phase 3**: *Handover execution*: Handover execution is the transfer of routing task to the newly decided network in Phase 2 for the proper forwarding of packets. Moreover, handover can be either *network controlled* or *mobile controlled*. In the former case, the handover is initiated and fully controlled by the network and is typically used for load balancing and traffic management [*Kim, Yun, Shim, Cho, & Choi, 2010*]. On the contrary, for mobile controlled, the handover is initiated and controlled by the mobile device itself. This type of handover management is most prevalent and most commonly used.

1.2 Handover Information Gathering

For the handover decision algorithm to work correctly and efficiently and to perform the "*Always Best Connected*" handover, a full set of information is to be gathered. For mobile controlled vertical handover, this information is to be made available to the mobile device efficiently and in real time. This information is not just limited to a particular layer, but spans to all the layers of the protocol stack as shown in Table 1.1 [*Barja et al, 2011*].

Table 1.1: Information at each layer of protocol stack for vertical handover

Layer	**Information for vertical handover**
Application Layer	• QoS Parameters (Jitter, bandwidth, etc.) • User Preferences (or choice of users) • Context information (type of service, speed, etc.)
Transport Layer	• Network Load (Available bandwidth, number of users connected, etc.)
Network Layer	• Network pre-authentication • Network topology • Network configuration
Data Link Layer	• Link Status • Radio access network conditions
Physical Layer	• Available access media

Typically, the information under the three heads is required to be collected for being processed during phase 2 (i.e. *handover decision phase*): a) network information, b) mobile device information, and c) user preferences.

1.2.1 Network information

To collect the available information from different networks in the vicinity, following are the possible two ways:

- <u>Active Scanning</u>: All the networks in the vicinity are surveyed by the mobile device in order to discover the values of the parameters. This require a unified way of interaction between a wide set of wireless technologies.
- <u>Passive Scanning</u>: As a complement to active scanning, passive scanning relies on the advertisement messages being sent out by dedicated network entities. This way the network presence is announced and the network information is also broadcast. For example, in Wi-Fi network, an Access Point (AP) periodically broadcast the beacon frame. If the mobile device receives this beacon frame, the corresponding network is discovered and the information embedded in it is used to decide whether to attempt "Association".

1.2.2 Mobile Device Information

This information is typically about the state of the mobile device, i.e. speed, battery status, features, type of service, mobility pattern etc.

1.2.3 User Preferences in Network Selection

User preference is one of the important criteria when performing vertical handover decision. This is because when access network changes, there are many aspects changing: cost, security, speed, latency, power consumption etc. Changing the access network without considering the preferences of the user would mean "hacking" the private space of the subscriber. For example, the user might prefer to connect to a privately owned Wi-Fi network rather than the one with which the corresponding 3G service provider has a roaming partnership with. So, it is a must to capture the preferences of the user and to consider it while selecting a network. These preferences are normally captured in the form of a questionnaire which the user fills in.

1.3 Analytic Hierarchy Process

A Multi Criteria Decision Making (MCDM) technique is often utilized to assist in vertical handover decision algorithm and to capture the choices and preferences of decision makers. While vertical handover decision with multiple attribute is a complex problem, AHP seems to be the most popular method to decompose it into a hierarchy of simpler and more manageable subproblems [*Kassar et al, 2008*]. It is one of most popular MCDM technique for organizing and analyzing complex decisions which involves (possibly) multiple criteria. It was developed by Thomas L. Saaty [*Saaty, 1980; Saaty, 2000*]. Since the inception of AHP, it has been extensively studied and refined for making it more effective and accurate. It provides a framework for structuring a decision problem hierarchically, for representing and quantifying the choice of decision maker on each decision element, for relating these choices with respect to the overall goal, and for ranking all the available alternatives with respect to all the decision criteria. It's been used extensively by many independent authors in their respective vertical handover decision algorithms [*Kassar et al, 2008a; Song et al, 2005a; Wang et al, 2012; Zhang et al, 2010; Song et al, 2005; Alkhawlani et al, 2008; Taheri et al, 2011*].

1.4 Objectives

The main objectives of this research are the following:

1) To propose the suitable fields of IEEE 802.11 beacon frame for stuffing additional information and to devise the stuffing technique for the same. The scope is not just limited to selecting an infrastructure WLAN, where additional network information required to be stuffed is not large. Rather, using the concept of fragmentation, allowing the beacon frame to carry general large advertisements.
2) To measure the magnitude of criticisms in Analytic Hierarchy Process (AHP) with respect to contradictory judgement matrices and to compare the priority vector of contradictory judgement matrices with non-contradictory ones on a qualitative and various quantitative metrics.

3) To propose a feedback based technique for avoiding contradictory judgement matrices in AHP.

1.5 Thesis Outline

The rest of the chapters in this thesis are arranged as follows:

Chapter 2, *Literature Review and Problem Statement*, shows the literature review closely related to:

a) Beacon fields and technique already proposed for beacon stuffing along with its applications in various diverse fields,
b) Vertical handover decision parameters and User Preferences in 3G-WLAN interworking environment, and
c) Analytic Hierarchy Process.

It then lists the identified research gaps and problem definition.

Chapter 3, *Related Theory*, presents the theory related to the research. More specifically, it explains the IEEE 802.11 network with the focus on the beacon frame format and the amendments proposed in the beacon frame by IEEE 802.11u. The intricacies of AHP technique are also included in this chapter.

Chapter 4, *A Novel Approach for Beacon Stuffing with Non-Standard Custom Information*, explains the proposed novel contribution in terms of enhanced beacon stuffing technique and its evaluation.

Chapter 5, *Measuring the Effectiveness of Contradictory Judgement Matrices in Analytic Hierarchy Process*, shows the results of measuring the magnitude of criticisms in AHP. More specifically, the magnitude of contradictory judgement matrices is quantified and these are compared with non-contradictory judgement matrices on a qualitative metric and thirteen different quantitative metrics. It finally proposes a feedback based technique for avoiding contradictory judgement matrices.

Chapter 6, *Modeling User Preferences for Vertical Handover in 3G-WLAN Interworking Environment*, shows the usefulness of beacon stuffing and the practical usefulness of feedback based technique proposed in Chapter 5 while capturing user preferences for network selection on top of IEEE 802.11u and using AHP.

Finally, Chapter 7, *Conclusions, Limitations and Scope for Further Research*, gives various conclusions drawn from the research, the limitations of the work, and provides some directions for future research.

CHAPTER 2

LITERATURE REVIEW & PROBLEM STATEMENT

This chapter explains various studies and models already developed and different domains where these models have been found useful. Section 2.1 reviews the concept of *Beacon Stuffing* and its applications. Section 2.2 shows various vertical handover decision parameters. It is followed by Section 2.3 which gives a review of Multi-Criteria Decision Making (MCDM) technique, called *AHP*, its criticisms and applications. Section 2.4 explains how user preferences for network selection have been captured in the literature. Section 2.5 gives a brief overview of the amendments proposed to 802.11-2007 standard. Section 2.6 lists the motivations to carry out this research. Finally, Section 2.7 defines the research gaps and Section 2.8 gives the problem definition.

2.1 Beacon Stuffing

IEEE 802.11 attempts to model WLAN's as a replacement for wired networks. A wireless client has to first "Associate" with an AP before it can communicate with it. Also, once a wireless client associates itself to an AP, it can no longer communicate with other AP's in the vicinity without using sophisticated software [*Chandra, Padhye, Ravindranath, & Wolman, 2007*]. There are some inherent limitations of such kind of communication model with modern devices (for e.g. smart phones) which can connect to multiple interfaces.

2.1.1 Beacon stuffing: Wi-Fi without Association

Proposed by R Chandra et al [*Chandra et al, 2007*], Beacon-Stuffing is a low bandwidth communication protocol for IEEE 802.11 networks. It enable AP's to communicate to wireless clients without association, which enables these clients to receive any custom information from the AP's in the vicinity even when they are associated to another AP or any other network. This protocol is based on following two key observations:

- Wireless clients always receive the beacon frames from all the APs in the vicinity, irrespective of association, and
- Beacon frame can be overloaded with more data.

Based on the above two key observations, Chandra et al (2007) proposed to stuff additional information in the IEEE 802.11 beacons. As a result, an 802.11 wireless client receives this additional information from all the nearby AP's. This push model is in contrast to the pull model wherein a wireless client first establishes an internet connection, then transmits information about its location (which can be obtained in a variety of ways), and finally "pulls" information relevant to that location [*Chandra et al, 2007*].

The proposed protocol embeds the additional information in the beacon frame using its three fields, namely:

a) Subscription Service Identifier (SSID),
b) Basic Service Set Identifier (BSSID), and
c) Vendor Specific Information Element

The proposed scheme is not just limited to small pieces of information, but can also deliver large piece of information (for example a short audio jingle) by provisioning the AP to split the message into smaller fragments, and transmit each fragment in a separate, contiguous beacon frame. Each fragment has the following format:

Unique-ID: Sequence-Number: More-Flag: Info-Chunk

Unique-ID is the unique identification number of the message, Sequence-Number is the fragment number, and More-Flag is an indication to the client that whether the received fragment is the last fragment of the message or not. Info-Chunk is the information embedded.

Though technique proposed by Chandra et al (2007) successfully overloads the three fields of the beacon frame with additional data, it has certain limitations as listed below:

a) Approach of embedding information in SSID field has an advantage of being simple and it does not require any kernel modification on client devices. Despite its advantages, this approach has a significant limitation. According to 802.11, SSID Information Element indicates the identity of an ESS. Many client devices (for example smart phones, iPhones, laptops etc) display each unique SSID in the received beacons of all the 802.11 networks within the range to facilitate the end user in network selection. If one AP is sending fragments of a large message chunk embedded in SSID field of subsequent beacons and the AP beacon interval is assumed to be 10 milliseconds [*Chandra et al, 2007*], each client device in the region will display 100 "bogus" SSIDs per second. And if multiple APs within the range are using SSID field for stuffing bits, the situation will become even worse for the client device. In fact, the legitimate SSIDs will virtually be lost and it will be extremely difficult for the client to decide on the network to which to attempt Association to.

b) Other than a) above, turning off the broadcast SSID is a security method that is normally implemented for securing 802.11 wireless networks [*Summers & Dejoie, 2004*].

c) The limitation with BSSID is that it is not always free, for e.g., if the Source Address field of MAC header contains a group address, all the receiving stations (or wireless clients) also validate the BSSID. If it is free also then the information content it can carry is limited to 6 octets only. Moreover, stuffing data in BSSID and Vendor Specific fields requires significant changes in the corresponding WLAN drivers at the AP as well as the mobile device.

d) The fragmentation technique proposed uses explicit sequence numbers. But those are redundant because beacon frame already has 12-bit sequence number field in the sequence control field structure of MAC header, and it can be used to the same effect. Beacons not received in order will automatically generate error in the driver code.

e) SSID concatenation, BSSID concatenation and Vendor Specific Elements could send only 32, 6, 252 octets respectively. The maximum size of beacon frame body is 2320 octets [*IEEE Standard 802.11, 2007*], and a lot more data could be embedded to increase the transmission rate.

f) The technique uses only one Vendor Specific Information Element to carry the data. But, according to IEEE 802.11 standard, multiple Vendor Specific elements can be added as long as the maximum size of the beacon is not exceeded.

g) Though three fields have been listed; they are used independently of each other. Combining multiple fields/approaches will lead to a greater transmission rate.

2.1.2 Applications of Beacon stuffing

Other than serving the purpose as specified in the standard, the beacon frame has often been looked upon from a different perspective: a carrier which is *advertising* the information to the mobile devices in the vicinity. Also, the beacon frames can be received only within a limited physical range in which the wireless network is available, thus inherently facilitating Location Based Services (LBS). Often these perspectives of the beacon frame are exploited in numerous ways by many authors in the literature, as shown in the rest of this section.

[*Chandra et al, 2007*] proposes beacon stuffing as a compelling technique to deliver location-sensitive advertisements. Also, it is proposed to be used to broadcast coupons from participating businesses.

Nicholson et al [*Nicholson, Wolchok, & Noble, 2010*] advocates' additional network interfaces to support parallelism in network flows, to improve handoff times, and to provide side band communication with nearby peers. Because of many issues with an additional physical interface, they proposed a link layer implementation of a virtual 802.11 networking layer. For the same, beacon stuffing is proposed to be used to increase the efficiency of the mobile station with reduced overhead of Dynamic Host Configuration Protocol (DHCP) configuration process when mobile device is migrating to a new AP.

Mhatre et al [*Mhatre, Lundgren, Baccelli & Diot, 2007*] had proposed MaLB (MAC-aware and load balanced routing algorithm), a greedy, tractable, and distributed mesh routing algorithm. In it, they propose to use the beacon stuffing approach to embed information about all the active links of a node.

Grunenburger et al [*Grunenberger & Rousseau, 2010*] had proposed the concept of virtual AP's to manage mobile stations in the infrastructure networks wherein the complexity of managing the mobility of stations is pushed back inside the networks, thus optimizing network resources and providing a better quality of service. They propose to use beacon stuffing for piggybacking the signal level, thus keeping the neighbors informed about the signal quality.

Chandra et al [*Chandra, Padhye, & Ravindranath, 2008*] used it in their scheme, called Neighborcast, which is a group communication paradigm for physically nearby Wi-Fi clients to communicate with each other.

Banerjee et al [*Banerjee et al, 2010*] proposes a scheme, called virtual compass, to automatically determine the social content by mobile social applications. It uses Wi-Fi to detect nearby mobile devices and uses beacon stuffing in there technique to enable every device to periodically broadcast its ID and the ID's and distance of its peers to each of its peers.

Shere [*Shere, 2009*] presents the design of an application architecture that addresses the needs of the opportunistic networking applications. There application uses beacon stuffing to embed device information in MAC layer frames, thus facilitating the gathering of application level information fast.

Champion et al [*Champion, Li, Zhai, Teng, & Xuan, 2012*] uses it in their proposed scheme, called Enclave, which helps people with smart phones to communicate unobtrusively.

2.2 Vertical Handover Decision Parameters

The vertical handover decision phase is based on the algorithms that takes the available information related to network, device, and user preferences as input; and evaluates and assigns weights to the available network choices based on the decision parameters and decision criteria. These algorithms are referred to as vertical handover decision algorithms.

Many vertical handover decision algorithms have been proposed in the literature based on diverse parameters. [*Nay & Zhou, 2009; Lassoued, Bonnin, Hamouda, & Belghith, 2008; Stevens-Navarro & Wong, 2006; Stevens-Navarro, Lin, & Wong, 2008*] considers end to end latency (L_e), which is defined as the time taken to deliver a packet from the source to the destination (or from packet generation to packet reception), in their proposed algorithm. Received Signal Strength (RSS) is one of the most referenced parameter, both as a input one and the one which triggers the vertical handover process [*Buburuzan & Nyamen, 2008; Stevens-Navarro et al, 2008; Sur & Sicker, 2005; Kassar, Kervella, & Pujolle, 2007; Li, Chen, & Xie, 2007; Stevens-Navarro et al, 2006; Chang & Chen, 2008; Mehbodniya & Chitizadeh, 2005; Xia, Ling-ge, Chen, & Hong-wei, 2008; Horrich, Ben, & Godlewski, 2007; Joe & Hong, 2007; Goyal & Saxena, 2008; Bernaschi, Cacace, & Iannello, 2004; Mani & Crespi, 2006*]. Network delay, which is a time taken to deliver a packet has been considered as an input parameter by many individual authors [*Kassar et al, 2007; Stevens-Navarro et al, 2006; Ying, Jun, Yun, Gen, & Ping, 2008; Kassar, Kervella, & Pujolle, 2008b; Fall, & Varadhan, 2000; Wright, 2007; Siddiqui & Zeadally, 2006;*]. Network availability, which is a measure of coverage area of the network, has been used by [*Dutta et al, 2007; Kassar et al, 2007; Bernaschi et al, 2004; Sen & Ukil, 2010*]. Bit Error Rate (BER), which is the number of bit errors divided by the total number of transferred bits during a studied time interval, is considered as input parameter by [*Lassoued et al, 2008; Kassar et al, 2008; Sur et al, 2005; Niyato, 2009; Corvaja, 2006; Mani et al, 2006; Kassar et al, 2008b; Wright, 2007*]. Signal to noise and Interference ratio (SINR) is another important parameter about the network and is considered by [*Corvaja et al, 2006; Mani et al, 2006; Inzerilli, Vegni, Neri, & Cusani, 2008; Sen et al, 2010*]. Round Trip Time (RTT), which refers to the time it takes for a signal to be sent plus the time taken for an acknowledgment to be received, has been considered by [*Sur et al, 2005; Stevens-Navarro et al, 2008; Stevens-Navarro et al, 2006; Chang et al, 2008; Fall et al, 2009; Sen et al, 2008*]. Security about the network is again an important parameter and the level of security deployed in the network is considered as input by [*Lassoued et al, 2008; Stevens-Navarro et al, 2006; Hasswa, Nasser, & Hossanein, 2005*]. Jitter introduced by the network is considered by [*Attaullah, Iqbal, & Javed, 2008; Kassar et al, 2008; Stevans-Navarro et al, 2006; Hasswa et al, 2005; Ying et al, 2008; Kassar et al, 2008b; Fall et al, 2009; Wright et al, 2007; Sen et al, 2010*]. Packet loss, which refers to number of percentage of unsuccessful packets delivered, is given as input by [*Kassar et al, 2007; Stevans-Navarro, 2006; Ying et al, 2008; Fall et al, 2009; Siddiqui et al, 2006; Sen et al, 2010*]. Available bandwidth is again an important parameter for the decision algorithm and [*Stevans-Navarro et al, 2008; Lassoued et al, 2008; Nay et al, 2009; Xia et al, 2008; Liao, Tie, & Du, 2006; Latvakoski et al, 2008; Kassar et al, 2008b; Attaullah et al, 2008; Singhrova & Prakash, 2007; Siddiqui et al, 2006; Sen et al, 2010*]. Since, 802.11 network operates on the unlicenced spectrum, the number of users connected to the network becomes an important parameter and is considered by [*Liu & Zhou, 2005; Nay et al, 2009*]. Number of vertical handover events performed by the network has been used as the evaluation metric by [*Stevans-Navarro et al, 2008; Joe et al, 2007; Lassoued et al, 2008; Mehbodniya et al, 2005; Xia et al, 2008; Corvaja et al, 2006; Joe et al, 2007; Inzerrilli et al, 2008; Goyal et al, 2008; Bernaschi et al, 2004; Mani et al, 2006; Liau et al, 2006; Kim, Han, & Han, 2010*] and vertical handover success rate is used by [Buburuzan et al, 2008]. Vertical handover packet loss, i.e. percentage of undelivered packets during the vertical handover process, has been considered by [*Dutta et al, 2007; Li et al, 2007; Bazzi, Pasolini, & Gambetti, 2006*]. The price for using the network is considered during handover decision phase by [*Lassoued et al, 2008; Kassar et al, 2007; Stevans-Navarro et al, 2006; Hasswa et al, 2005; Xia et al, 2008; Corvaja et al, 2006; Liau et al, 2006; Ying et al, 2008; Goyal et al, 2008; Kassar et al, 2008a; Singhrova et al, 2007; Niyato & Hossain 2009; Siddiqui et al, 2006*].

2.3 Analytic hierarchy Process (*Saaty, 1977*)

Analytic Hierarchy Process (AHP) is a Multi-criteria Decision Making (MCDM) technique and is useful especially in presence of multiple conflicting and subjective criteria. It is typically used to help the decision maker in selecting the most appropriate alternative.

Since its inception, AHP is used in many diverse fields. Many of these fields are specified by Ishizaka et al [*Ishizaka & Labib, 2011*] in their survey paper. These include evaluating the performance of website (*Liu & Chen, 2009*), manufacturing systems [*İç & Yurdakul, 2009; Li & Huang, 2009; Yang, Chuang, & Huang, 2009*], selection of supplier [*Chamodrakas, Batis, & Martakos, 2010; Labib, 2011; Wang, Che, & Wu, 2010; Wang & Yang, 2009*], evaluation of software [*cibeci, 2009; Chang, Wu, & Lin, 2009*], weapon selection [*Dagdeviren, Yavuz, & KilInç, 2009*], software design [*Hsu, Kao, & Wu, 2009*], performance evaluation of organization [*Tseng & Lee, 2009*], recruitment of staff [*celik, kandakoglu & Er, 2009; Khosla , goonesekra & chu, 2009*], planning of route [*niaraki & kim, 2009*], rating of customer requirements [*Li, tang & luo 2010; Lin, Chen, & TZeng, 2010*], university selection and evaluation [*Lee, 2010*] and many others.

Apart from above fields, AHP is also been used by many individual authors for vertical handover in 3G WLAN interworking environment [*Kassar et al, 2008a; Song et al, 2005a; Wang et al, 2012; Zhang et al, 2010; Song et al, 2005; Alkhawlani et al, 2008; Taheri et al, 2011*].

AHP is based on following five steps (as explained from Section 2.3.1 to 2.3.5):

2.3.1 Modeling the problem into hierarchy of decision elements

Structuring a problem hierarchically is beneficial as it facilitates to concentrate at a sub problem at a time and permits a clear understanding of the affect of priority changes of higher level elements on the priorities of lower level elements. One of the advantages with AHP is that it permits the hierarchical structure of the criteria. Brugha [*Brugha, 2004*] has provided a complete guideline on how to structure a problem hierarchically.

Saaty & Forman [*Saaty & Forman, 1992*] facilitates a deep insight by compiling hierarchies in many different applications.

2.3.2 Pair-wise Comparisons

Psychologists argue that for a decision maker it is comparatively easier and more accurate to express the opinion on only two decision elements (or criteria) at a time rather than all the decision elements simultaneously. Such pair-wise comparisons have been used long time before by psychologists [*Thurstone 1927; Yokoyama 1921*], as specified by Ishizaka et al [*Ishizaka et al, 2011*]. Another advantage of using pair-wise comparisons is that it allows cross checking the consistency of the decision maker.

For recording the intensity of comparisons, AHP uses a ratio scale [*Saaty, 1977*] which does not require units in the comparison. This is contrary to the methods using interval scales [*Kainulainen et al 2009*]. Here the judgement is a relative value. Moreover, it is not required for the decision maker to always provide a numerical judgement. Rather, relative verbal judgement with which we are more familiar in our daily lives is sufficient.

2.3.3 Priority Derivation

Once the ratio-scale matrix is ready, the next task is to derive the priorities (and thus ranking) of criteria which were compared among each other in the corresponding judgement matrix. The objective is to find the priorities p_1, p_2 ... p_n such that $p_i/p_j \approx a_{ij}$ (i, j = 1, 2 ... N), where a_{ij} is the corresponding entry in i^{th} row and j^{th} column of the ratio scale matrix. If the pair wise comparisons are consistent (a_{ij}. a_{jk} = a_{ik}), i.e. follows both ordinal a well as cardinal transitivity, the ordering is obvious, and simple "Mean of the rows of matrix with normalized columns" [*Ishizaka & Labib, 2011; Golany & Kress, 1993*] can find the priorities. If however the matrix is not consistent, then an appropriate method is required for obtaining these weights. These inconsistent ratio-scale matrices are often a result of judgements as it is the human beings who are responsible for filling up the entries in the matrix. If the inconsistency is small while recording the pair-wise comparisons, small distortions are introduced. Based on this, Saaty used perturbation theory to advocate (and in fact "justified") the principal Eigen vector of the judgement matrix, P, as the desired priority vector.

After the publication of AHP, Johnson et al [*Charles R Johnson et al, 1979*] pointed out that the left Eigen vector is equally justified as the priority vector if the order of judgements in matrix A is reversed. They also proved with the help of few examples out that resulting prioritized vectors may disagree even when the comparisons are nearly consistent. This problem was called as rank reversal problem for scale inversion (RR-SI). Moreover, this inconsistency with right and left Eigen vector arises only for inconsistent matrices with order higher than three [*Saaty & Vargas, 1984*].

To avoid this problem of left and right inconsistency, Gordon Crawford et al (*Gordon & Williams, 1985*) proposed another approach to derive a comparable estimate; the Geometric mean vector. Using this method the priorities are given by taking the geometric mean of the corresponding rows of the rank-order matrix and normalizing the values. It can also be applied to hierarchical problems in exactly the same way as Eigen vector method.

Comparing the two methods, i.e. principal Eigen vector and geometric mean vector, the mathematical evidences support the geometric mean over Eigen vector approach [*Ishizaka et al, 2011*]. But when compared under the simulated environment, no there is no clear difference between the two methods [*Choo & Wedley, 2004; Herman & Koczkodaj, 1996; Ishizaka & Lusti, 2006; Jones & Mardle, 2004*]. Because of the lack of this practical evidence [*Choo & Wedley, 2004; Herman & Koczkodaj, 1996; Ishizaka & Lusti, 2006; Jones & Mardle, 2004*], the Eigen vector method (as proposed by Saaty) is largely supported and used [*Saaty & Forman, 2003; Saaty & Hu 1998*].

Other than the Eigen vector method and geometric mean method, E.U. Choo [*Choo & Wedley, 2004*] enumerated 18 different methods for finding the priority vector which later on were proved to be effectively 15 only [*Lin, 2007*].

2.3.4 Consistency

AHP allows certain degree of inconsistency present in the judgement matrix. More specifically, the amount of inconsistency considered to be acceptable corresponds to Consistency Ratio (CR) < 10% [*Saaty, 1977*]. If CR > 10% the decision maker is asked to reconsider the decisions (or comparisons).

But, this consistency index is again been criticized in the literature for mainly two following aspects:

a) Karapetrovic et al [*Karapetrovic & Rosenbloom, 1999*] shown that there are situations where the decision maker has been reasonable, logical, and non-random in doing comparisons and yet the corresponding judgement matrix can fail the consistency test.

a) It was shown that there exists a subset of inconsistent matrices (with CR < 10%), called contradictory matrices [*Kwiesielewicz & Uden, 2004; Carlos, Costa, & Vansnick A., 2008*]. The biggest issue with these contradictory matrices is that there exists no ordering of corresponding decision elements which satisfies all the judgements expressed in the judgement matrix. This fact was proved using combinatorics [*Kwiesielewicz & Uden, 2004*].

2.3.5 Aggregation

This step determines the global priorities of alternatives by synthesizing all the local priorities. AHP along with this aggregation process is called Distributive mode AHP [*Ishizaka et al, 2011*]. This distributive mode AHP is subject to Rank Reversal, a phenomenon for which AHP is much debated in the literature and often criticized [*Dyer, 1990a, 1990b; Holder, 1990, 1991; Stam & Duarte Silva, 2003*]. Originally, Belton & Gear (*1983*) discovered this phenomenon where if a copy of an alternative is added, the global priorities might change. It is worth mentioning here that this Rank Reversal is different from the local rank reversal, or RR-SI.

The revised AHP (called ideal mode AHP) was suggested by Belton and Gear [*Belton and Gear, 1985, 1983*]. It was argued that rank reversal happens in distributive mode because the relative performance measures of all alternatives in terms of each criterion summed to one. So, in ideal mode AHP, instead of having the relative performance values sum up to one, dividing each performance relative value by the maximal value in the corresponding priority vector was suggested [*Belton and Gear, 1985, 1983*]. Later, Saaty and Vargas (*1984b*) provided an example to show that ideal mode AHP is also subject to rank reversal.

2.4 User Preferences for Vertical Handover Decision

Many research articles in the past have considered the user preference as a criterion for vertical handover decision in one form or the other. These preferences are based on many diverse parameters.

Nay et al [*Nay & Zhou, 2009*] propose the vertical handover decision algorithm for UMTS and Low earth Orbit (LEO) satellite network. It takes into consideration two leading factors, namely the performance of QoS (PQoS) and the cost of vertical handover (Ch). The proposed algorithm measures and quantifies numerically these two factors and compare and choose the best PQoS and lowest Ch to perform the handover. Out of five factors considered for evaluating Ch, one of the factors is "choice of user". This factor only captures the choice of user to activate or deactivate the handover on his/her mobile device; and therefore its scope is very much limited and fixed.

Kassar [*Kassar et al, 2007*] propose a vertical handover decision algorithm based on the concept of context awareness. The proposed algorithm groups the context information under Network context and Terminal context. Terminal context is further classified as static and dynamic context; and user preferences is the static criteria on possibly all the criteria considered. It then uses Criteria Scoring, which maps the decision criteria into scores according to user preferences. Here user gives preference for many criteria and also it is not clearly mentioned how the preferences of the user are to be captured.

Pawar [*Pawar et al, 2008*] shows the context sources on the mobile device, and the context description provided by them. One of the context source is "User Preferences Context Source" and the context information provided by it is the ranked list by user of all the mobile network providers, network names, and network technologies a user is subscribed to, a ranked list of all the device services according to their importance to the user, and the power preference (yes/no) indicating whether the power usage by interfaces should be considered or not. Here again user has many objectives to rank.

Siddiqui [*Siddiqui et al, 2006*] considers user preferences to cater special requests for one type of system over another. It lists various reasons that why user preferences should be captured and considered while deciding upon the network to connect to.

Sen et al [*Sen et al, 2010*] proposes an QoS and context aware algorithm which utilizes the end to end network QoS context (like available bandwidth, RTT, and jitter) to arrive at QoS parameters like minimum throughput, maximum delay, maximum cost, security, privacy requirement etc. The selection of the most suitable target segment depends mostly on the user profile containing information such as the minimum and maximum cost and the list of segments with the highest and lowest priority [*Sen et al, 2010*].

Chen [*Chen el al, 2004*] proposes a smart decision model which uses a well defined score function to handover to the best network interface at the best moment according to the properties of the available network interfaces, system information, and user preferences. Basically here the decision module provides flexibility in controlling the desired network interface to the user.

Kassar et al [*Kassar et al, 2008b*] presents a handover management scheme which is based on autonomic-oriented architecture. It is able adapt dynamically with changes in the environment. The scheme contains three main sources of information: the "Piloting", the "Control", and the "Knowledge". The information considered under "Knowledge" is the context environment, for example, movement of mobile device, parameters related to QoS, Preferences of user, availability of access network etc.

2.5 History of Amendments To 802.11

IEEE 802.11–2007 is the base protocol published by IEEE LAN/MAN standards committee (IEEE 802) in the year 2007. Since then it is amended ten times in the form of ten amendments published since 2008 to 2011. Every amendment has either added new functionalities or has extended the scope of the existing ones. In March 2012, the committee combined all the ten amendments and 802.11-2007 base standard; and published IEEE802.11-2012 standard.

Following is the brief description of the amendments to 802.11-2007 standards:

- Amendment 1 [*802.11k-2008*]: This amendment specifies the extensions to IEEE standard 802.11 for Wireless LANs providing mechanisms for Radio Resource Measurement. The proposed Radio Resource Measurement approach is to add measurements that extend the capability, reliability, and maintainability of WLAN through measurements and provide that information to upper layers in the communications stack [*802.11k-2008*].

- Amendment 2 [*802.11r-2008*]: This amendment describes mechanisms that minimize the amount of time for data connectivity lost between the mobile station and the distribution system (DS) during a BSS transition. It permits continuous connectivity aboard wireless devices in motion, with fast and secure handoffs from one base station to another managed in a seamless manner.

- Amendment 3 [*802.11y-2008*]: It defines enhancements to the IEEE 802.11 physical layer (PHY) and medium access control (MAC) to support operation in the 3650–3700 MHz band in the United States.

- Amendment 4 [*802.11w-2009*]: IEEE 802.11-2007 addresses security of data frames but systems are still vulnerable to malicious attack because management frames are unprotected. The purpose of this amendment is to reduce the susceptibility of systems to such attack.

- Amendment 5 [*802.11n-2009*]: This amendment defines modifications to both the 802.11 physical layers (PHY) and the 802.11 Medium Access Control Layer (MAC) so that modes of operation can be enabled that are capable of much higher throughputs.

- Amendment 6 [*802.11p-2010*]: IEEE 802.11p is an approved amendment to the IEEE 802.11 standard to add wireless access in vehicular environments. The purpose of this standard is to provide the minimum set of specifications required to ensure interoperability between wireless devices attempting to communicate in potentially rapidly changing communications environments [*802.11p-2010*].

- Amendment 7 [*802.11z-2010*]: 802.11z amendment defines mechanisms that allow setting up a direct link between client devices while also remaining associated with the AP.

- Amendment 8 [*802.11v-2011*]: This amendment defines enhancements to the IEEE 802.11 physical layer (PHY) and medium access control (MAC) to support managing IEEE 802.11 devices in wireless networks.

- Amendment 9 [*802.11u-2011*]: IEEE 802.11u-2011 is an amendment to add features that improve interworking with external networks. The interworking service aids network discovery and selection, which in turn enables information transfer from external networks and enables emergency services. It provides information to the stations (STAs) about the networks prior to association.

- Amendment 10 [*802.11s-2011*]: This amendment specifies enhancements to support mesh networking. The networks described in this amendment make use of layer-2 mesh path selection and forwarding (that is, a wireless mesh network that performs routing at the link layer).

2.5.1 IEEE 802.11 - 2012 standard

802.11-2012 gives users, in one document, the IEEE 802.11 standard for wireless local area networks (WLAN's) with all the amendments that have been published till 2011. Consolidating the 10 amendments to the 2007 standard, 802.11-2012 is expected to provide 600Mbps throughput without using MIMO technology. For providing this impressive speed, the physical layer and the software layer components are reworked. These changes allow for new additions like "mesh" networking, direct-link setup, security changes, broadcast/multicast/unicast data delivery and additional network management features [*IEEE Standard 802.11, 2012*].

2.6 Motivation for Research

The following three aspects can be credited for motivating this research:

a) *Increasing number of Wi-Fi hotspots*: Wi-Fi operates in more than 220,000 public hotspots and in tens of millions of homes and corporate and university campuses worldwide [*Hotspot usage to reach 120 billion connects by 2015, 2011*]. Also, AT&T's Q3 2009 hotspot connection numbers were 25.4 million sessions, up from 15 million the quarter before. Of these connections, 60% were from “integrated devices”, meaning Smart phones. It is estimated that in 2015 wireless hotspots will account for nearly 120 billion connect sessions [*Hotspot usage to reach 120 billion connects by 2015, 2011*].

   Also, by 2017, almost 21 Exabyte of mobile data traffic will be offloaded to the fixed network by means of Wi-Fi devices and fem-to-cells each month [*Cisco Visual Networking index, 2013*].

b) *Broadcasting perspective of IEEE 802.11 Beacon frame*: The mobile devices which had activated (or turned ON) their respective 802.11 network interfaces inherently receives all the beacon frames of the corresponding networks in the vicinity. So, other than serving the purpose as specified in the standard, the beacon frame can be looked upon from a different perspective: a carrier which is advertising the information to the mobile devices in the vicinity. Here the meaning of the word advertisement is not restricted to only commercial products advertising, rather it means inherent broadcast of *any* information. Often this perspective of the beacon frame is exploited in numerous ways. Two important ones are:

- Embedding and passing the values of network parameters to the mobile device, thus making these values available to all the mobile devices in the vicinity, those connected to the network and those not connected to the network. This facilitates designing the downward, mobile assisted, vertical handover decision algorithms which do not necessarily rely on the standard information available in the beacon frame.
- Sending the location specific service advertisements to the mobile device because the location of the AP is generally fixed and its coverage area is limited.

c) *Publish of 9th amendment to IEEE 802.11-2007 standard*: Also called IEEE 802.11u, this amendment is expected to further boost the popularity of Wi-Fi networks and enhancing the user experience in terms of network selection, thus further facilitating the terminology of ABC.

2.7 Research Gaps

Based on the literature survey, following are the research gaps identified which serves as guiding factors for the research work in this thesis:

1) Section 2.2 explains many vertical handover decision parameters considered by individual authors in there corresponding decision algorithms. Many of these parameters are not the part of IEEE 802.11 compatible beacon frame. Now with this limitation, in push based mobile assisted vertical handover environment the proposed decision algorithms are confined to theoretical proposals only.
2) Existing location-sensitive advertisement delivery mechanisms are constrained with two main limitations: a) the mobile device should have a reasonable quality connection to the internet, and b) there must be an automatic mechanism for indicating the location of the user. Many existing systems rely on Global Positioning System (GPS) for location indication, which many a times doesn't

work indoors. Moreover, always connected to internet while moving can be a costly option and is difficult with respect to maintainability of reasonable QoS.

3) Section 2.4 explains various research articles in which user preferences have been considered for network selection during handover execution. These proposals vary widely with respect to the parameters for capturing user preferences.
4) Though many independent authors have considered user preferences as one important criterion during network selection, the mechanism for capturing the same is still to be proposed.
5) Stuffing additional information in the beacon frame is proposed by R Chandra [*Chandra et al, 2007*]. Section 2.1.1 explains its various limitations.
6) AHP is used extensively in many areas, including vertical handover. Though it is proved that AHP technique allows the consideration of contradictory judgement matrices, the magnitude of this criticism is yet to be measured.
7) The principal Eigen vector as a priority vector in AHP is also criticized in the literature for RR-SI. Measuring its magnitude for contradictory and non-contradictory matrices is again required to be checked.

2.8 Problem Definition

Based upon the research gaps identified and listed in Section 2.7, following are the problems which are attempted to be resolved via this research study:

a) With respect to research gaps numbered (1), (2), and (5) listed in Section 2.7, how the beacon stuffing technique can be improved to overcome the limitations of the existing technique (as listed in Section 2.1.1)?
b) With respect to research gap numbered (6) in Section 2.7; though AHP is often criticized for allowing a subset of contradictory judgement matrices, what is the magnitude of this problem? This quantification will help in taking much informed decisions about avoiding/eliminating/ignoring such kind of judgement matrices.
c) With respect to research gap numbered (7) in Section 2.7; what is the difference between the quality of priority vector for contradictory judgement matrices and

non-contradictory judgement matrices with respect to various qualitative and quantitative metrics?

d) Based on the results of point (b) and (c) above, whether to avoid/eliminate/ignore the contradictory judgement matrices, and how?

CHAPTER 3

RELATED THEORY

3.1 Introduction

This chapter introduces the relevant theory required to clearly understand the purpose of this research. The concepts covered and their respective explanation is limited to the scope of this research work.

Section 3.2 briefs about the wireless networks and the success factors of Wi-Fi networks. In Section 3.3, the underlying protocol, i.e. IEEE 802.11, is explained. It briefs about the network architecture, steps to be followed by a mobile device to start communication in a Wi-Fi network, and the 802.11 compatible beacon frame structure in detail. Section 3.4 explains the amendments proposed by the ninth amendment of IEEE 802.11 (i.e. IEEE 802.11u) in the beacon frame. Section 3.5 explains about the taxonomy of vertical handover. Section 3.6 briefs about the Multi Criteria Decision Making (MCDM); by emphasizing in detail about the Analytic Hierarchy Process (AHP). Finally, Section 3.7 explains about the four UMTS QoS classes.

3.2 Wireless Networks

To satisfy the data needs of end user via wireless medium, many different types of wireless networks are available. These include cellular network, Wi-Fi network, ad-hoc network and WIMAX network. Basically these networks differ on a) Coverage area, b) Communication / Data transfer speed, c) Spectrum usage and its regulation (i.e. whether the spectrum is licensed spectrum or, unlicensed spectrum) and, d) Infrastructure/Infrastructure-less network.

3.2.1 WI-FI Success factors

Over the years Wi-Fi has emerged as one of the most promising technology for satisfying the data needs of the mobile users, thus making it ideal for interworking with 3G network. Following six factors can be credited for the huge success of Wi-Fi.

1) Vast unlicensed spectrum: Wi-Fi operates in unlicensed ISM 2.4 GHz and 5 GHz frequency bands. Moreover the regulatory approval is not required for individual deployments, and Wi-Fi has a large “free” spectrum available for network deployment.

2) High data rates and user experience: It is expected that IMT-2000 provides the minimum transmission data rates of 2 Mbit/s for stationary or walking users, and 384 Kbit/s in a moving vehicle [*Patil, Karhe, & Aher, 2012*]. Compared to this, IEEE 802.11n, which operates on both the 2.4 GHz and 5GHz band, can provide the sufficiently high data rates, thus making a much better network solution for consumers.

3) Total ownership cost: Wi-Fi offers huge capital expenditure and operational expense benefits for operators. Over the last decade, since the launch of Wi-Fi technology, it has evolved and matured enough, thus bringing down the equipment cost significantly. Also, without requiring large investment in channel planning and site surveys, the Wi-Fi networks can be easily and cost-effectively scaled.

4) Advanced Security and QoS: With the most common wireless encryption standard, Wired Equivalent Privacy (WEP), been shown to be easily breakable [*Asthana N. C., Nirmal A., 2009*], the Wi-Fi protected access (WPA and WPA2) encryption aimed to solve the problem. WPA2 is based on IEEE 802.11i and it provides 128-bit AES-based encryption using Pre-Shared Key (PSK) or 802.1x RADIUS authentication, which is ideal for operators to provide Authentication, Authorization and Accounting (AAA) services.

5) Increasing number of Wi-Fi hotspots: As already said in chapter-2, the usage of Wi-Fi hotspots to satisfy the user requirements are constantly increasing with the projection that by 2017, almost 21 Exabyte of mobile data traffic will be offloaded to the fixed network by means of Wi-Fi devices and fem-to-cells each month [*Cisco Visual Networking index: Global Mobile Data Traffic Forecast Update, Feb 6, 2013*].

6) Wi-Fi Complement to 3G: Over the years Wi-Fi has proved to be complementary technology to 3G and there are several important ways in which Wi-Fi and 3G approach of offering wireless access services are substantially different. *First*, the corresponding network deployment and business models are different. The basic business model of

3G is the telecommunication service model in which service providers own and manage the infrastructure and sell service on it. In contrast, Wi-Fi favors data communication industry (LANs). The basic business model is the equipment makers selling equipments to customers and services provided by the equipment are free to its customers. *Second*, 3G mobile technology use licensed spectrum, while Wi-Fi uses unlicensed shared spectrum. Thus there cost of service and quality of service are different. *Third*, the standardizing bodies of two are different. 3G has been standardized by 3GPP and is a relatively small family of internationally sanctioned standards. In contrast, Wi-Fi is one of the families of continuously evolving 802.11x wireless Ethernet standards. *Finally*, 3G offers communication in much broader geographical area with ubiquitous services, but at comparatively less speed. In contrast, Wi-Fi offers communication in smaller geographical area, but at very high speed.

3.3 IEEE 802.11 Standard

IEEE 802.11 refers to a family of specifications developed by IEEE for implementing Wireless Local Area Network (WLAN) computer communication in 5, 3.6, and 2.4 GHz frequency bands. It specifies an over-the-air interface between a wireless client and a base station or between two wireless clients. It is somewhat similar to a cellular architecture where the system is subdivided into cells, where each cell (called Basic Service Set or BSS in 802.11 nomenclature) is controlled by an Access Point (AP). It provides the basis for wireless network products using the Wi-Fi brand.

3.3.1 802.11 network architecture

Figure 3.1 shows the architectural components of 802.11 network. These components of the architecture interact with each other to provide a WLAN that supports mobility of wireless client (or mobile station) transparently to upper layers. The basic building block of the architecture is a Basic Service Set (BSS) which can be thought of as a coverage area of one AP. The member mobile stations of the BSS can communicate with the corresponding AP or to each other. Every BSS is identified by its BSSID, which is the MAC address of the AP serving the BSS. There are two types of BSS:

- Independent BSS (IBSS): IBSS is basically an ad-hoc network without the presence of any AP station. Any two or more stations can form an IBSS network with a common protocol. In such architecture, there is no central station to co-ordinate the transfer of information in the wireless medium and the stations co-ordinate this work among themselves.

- Infrastructure BSS: Infrastructure BSS, on the other hand, has a single central station called the AP which coordinates the medium access among all stations. The AP also has access to a wired network.

A wireless client's membership with the BSS is dynamic, i.e. wireless client come within the range of AP, go out of range of AP, turn the power ON and OFF. For a wireless client to become a member of the BSS and to access all the services, it must first "Associate" with the BSS. This Association process involves the Distribution System (DS).

DS is another important architectural component that is used to interconnect multiple BSS's, thus extending the possible range of the network. In 802.11 terminology, this extended form of the network is called Extended Service Set (ESS).

AP is another architectural entity which facilitates the movement of data between wireless client and the DS. In the infrastructure mode, the wireless client in a BSS always sends/receives the data packets to/from the associated AP. In other words, the wireless client always communicates with all the other stations via the AP. It is worth mentioning here that AP's are also addressable wireless stations and the addresses used for communication on the wireless medium and on the distribution system medium are not necessarily the same.

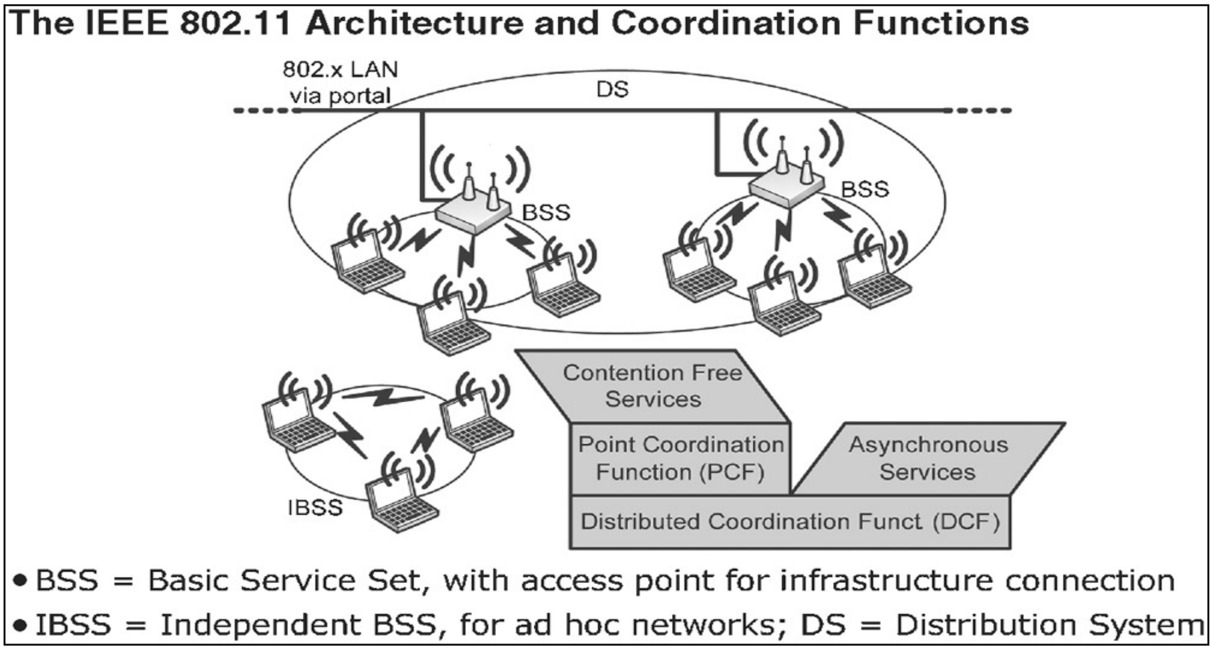


Figure 3.1: The IEEE 802.11 architecture [adopted from *Walke et al, 2006*].

3.3.2 Communication using 802.11 network

When a mobile station enters into a BSS area (or it powers up) and it requires to access the services via an existing BSS, it needs to execute series of steps. It first requires getting network parameters and synchronization information from the AP. Depending upon the power consumption and performance trade off, the decision to get this information is taken via one of the following two means:

1. Passive Scanning: The station waits to receive a beacon frame. This beacon frame is periodically broadcast by the AP and basically it contains the network specific parameters and synchronization information. Beacon frames shall be generated for transmission by the AP once every dot11BeaconPeriod Time Units [*IEEE standard 802.11, 2007*].

2. Active Scanning: Instead of waiting for a beacon frame to arrive, 802.11 facilitate the mobile station to find an AP by sending a probe request frame. In response to probe request frame, AP sends a probe response frame. Moreover, active scanning is prohibited in some frequency bands and regulatory domains.

With passive scanning, mobile stations do the following to start communication:

1) There can be more than one AP's present in the same geographical region, thus multiple 802.11 networks may be available within that region. The mobile station scans all the supported channels by listening to the beacon frames. After the scan is done, the mobile station decides on the BSS (or AP) which it has to join.

2) The mobile station starts an authentication procedure by sending an Authentication frame to the BSS. Authentication is the process of exchange of information between AP and mobile station to ensure that the station is authorized to access the network.

3) If the authentication succeeds, the mobile station must be "Associated" with the AP. This is how the network determines where to send data that is intended for that station. It routes it through the AP with which the node is associated to. This is why a node may only be associated to a single AP.

4) On successful association, the AP and mobile station can communicate using the basic medium access mechanism, called Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA). The corresponding Medium Access Control (MAC) protocol depends upon the coordination function, i.e. Distributed Coordination Function (DCF) for traffic without Quality of Service (QoS), the Point Coordination Function (PCF) for traffic with QoS requirements, or the Hybrid Coordination Function (HCF).

3.3.3 IEEE 802.11BEACON FRAME STRUCTURE

Beacon frame is a type of management frame as per 802.11 standard and gets broadcast periodically by the AP. This period between the two beacon frames is configurable and can vary from network to network. If a mobile device needs to communicate using the 802.11 network in passive mode, it first listens to the beacon frame. Using the information embedded in the beacon, it decides whether to connect to it or not. If yes, it attempts Association with the corresponding AP.

Mostly beacon frame carries the information related to the network with which the corresponding associated AP broadcast it. The syntax, semantics and arrangement of this information in the beacon is standardized by 802.11 standard. This is to facilitate the communication between any two 802.11 compatible devices, especially when they are manufactured by different vendors.

The IEEE 802.11 beacon frame format is shown in Figure - 3.2. The 28 octets long frame header (or MAC header) consists of the following fields, in order:

1. Frame Control: This 2 octet field is further divided into eleven sub fields. The basic purpose of this field is to specify the version of the protocol used, the type of frame (out of management, control, or data frame and its further subtypes), whether operating in power management mode etc.

2. Duration: The contents of this 2-octet field depends upon

   a) Frame type and subtype

   b) Whether the frame is transmitted during the Contention Free Period

   c) Whether the frame is transmitted with the QoS capabilities of the sending station.

3. Address 1: The Address 1 field of the management frame is the Receiver Address (=Destination Address) and determines the destination address of the frame.

4. Source Address (or Address 2): The Address 2 field of the management frame is the Transmitter Address (=Source Address) and determines the address of the mobile station transmitting the frame.

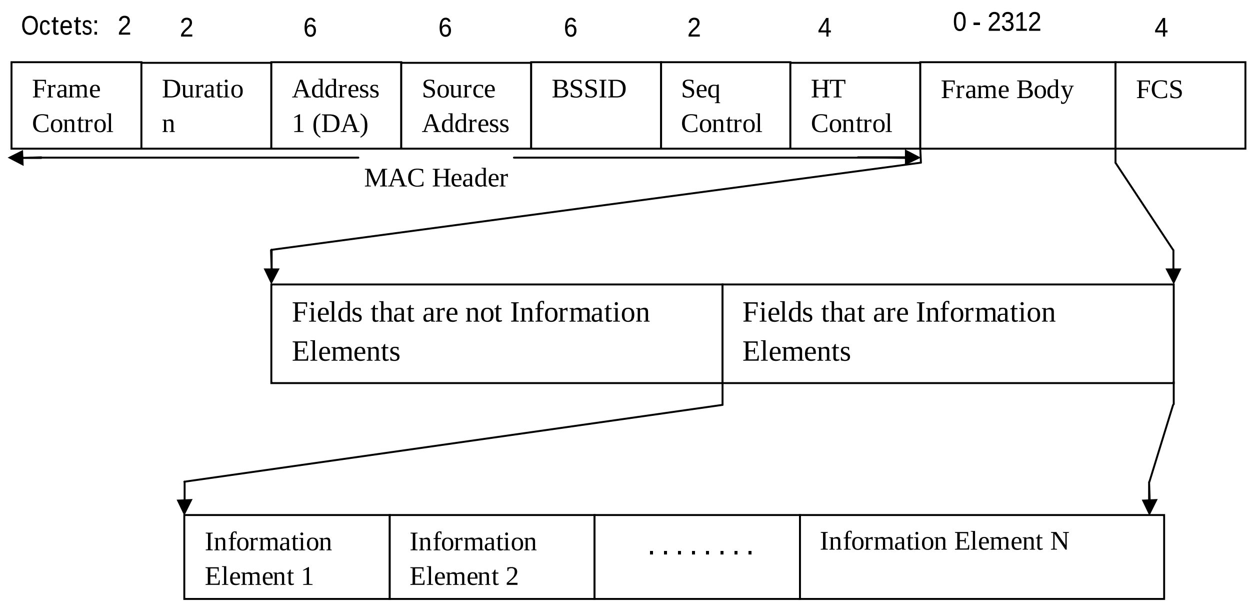


Figure 3.2: Beacon Frame Format of 802.11

5. BSSID (or Address 3): BSSID is the MAC address of the mobile station currently in use by the AP. So address 3 field represents the BSSID of the mobile station.

6. Sequence Control: This two octet field is further subdivided into two sub fields: sequence number and fragment number as shown in Figure-3.3.

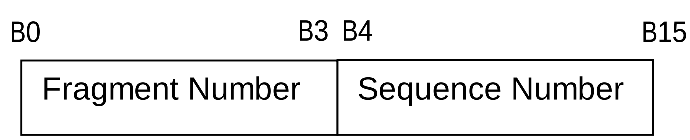


Figure 3.3: Sequence Control Field.

Sequence Number indicates the sequence number of each frame. The sequence number is same for each frame sent for a fragmented frame; otherwise, the number is incremented by one [*IEEE Standard 802.11, 2007*]. **Fragment Number** indicates the number of each frame sent of a fragmented frame. The initial value is set to 0 and then incremented by one for each subsequent fragmented frame sent [*IEEE Standard 802.11, 2007*].

7. HT Control: 802.11n amendment adds a new field to the 802.11 MAC header, called the HT Control field. It is a four octet's long field and follows the QoS control field in the 802.11 MAC header.

The MAC header is followed by the variable length Frame Body, which can be up to 2320 octets long. Frame Check Sequence (FCS) is a 4-octet long field used to perform 32-bit Cyclic Redundancy Code (CRC) for validating the received frames.

The frame body is a variable length field having information which is specific to frame type and subtype. It consists of a series of fields that are classified as fields that are not Information Elements followed by fields that are Information Elements. Information Elements appear in a fixed relative order and are identified by respective unique Element ID. The general format of all the Information Elements is shown in Figure 3.4.

Octets:	1	1	Length
	Element ID	Length	Information

Figure 3.4: General Format of Information Element

Each Information Element has following three fields:

1) 1-octet long Element ID (EID) field to uniquely identify the Information Element. Hence, 256 unique Information Elements are possible. 802.11-2012 specifies 119 EID's out of which up to 53 EID's can be the part of beacon frame. The undefined EID's are reserved by IEEE for future use.

2) 1-octet long Length field the value of which specifies the number of octets in the Information field.

3) Variable length, element specific Information (Data) field: Since its exact length is specified in Length field, the maximum length of it can be 255 octets.

3.4 IEEE 802.11u: Interworking with external networks

IEEE 802.11u [*IEEE Standard 802.11u, 2011*] is the ninth amendment to the IEEE 802.11-2007 standard and specifies enhancements to IEEE 802.11 Medium Access Control (MAC) that supports WLAN interworking with external networks. This amendment defines mechanisms that facilitate IEEE 802.11 technology to interwork with external networks. The external network can be cellular core network, fixed line networks, etc. IEEE 802.11u defines enhancements to the MAC layer to allow mobile communication devices, such as laptop computers or multi-mode mobile phones, to join a widely used 802.11 network. These enhancements cover the areas of device enrolment, selection of network, support for emergency services, user traffic segmentation, and service advertisement. The MAC changes allow a device to discover an enhanced and rich set of information about the network, prior to the commencement of a user session. This considerably improves the user experience of Wireless LAN connection and setup.

3.4.1 IEEE 802.11u: Purpose and anticipated effects

It enables 802.11 devices to interwork with external networks, irrespective of whether the service is subscription based or free. Interworking service facilitates:

a) Network discovery and selection: One of the key features of 802.11u is the ability to advertise pre-association information to clients. It facilitates the AP to send this pre-association information the beacon frame. More specifically, it allows the embedding of information in the beacon frame using which the AP can describe the type of network offered from a predefined list. The AP can also advertise the roaming consortium, and venue information.

b) Emergency services: It provides users to access emergency services by providing methods to access emergency services via 802.11 infrastructures.

c) Subscription Service Provider Network (SSPN) interface service: It supports service provisioning and transfer of user permissions from the SSPN to the AP.

3.4.2 Amendments proposed in the Beacon Frame

Here the discussion is limited to the enhancements proposed by 802.11u in the beacon frame only. Table 3.1 shows the four Information Elements are allowed to be the part of the frame bode of the beacon:

Table 3.1: Information Elements as a part of Beacon in 802.11u

Order	Element ID	Information Element	Notes
45	107	Interworking	The Interworking element is present if dot11InterworkingServiceActivated is true.
46	108	Advertisement Protocol	Advertisement Protocol element is present if dot11InterworkingServiceActivated is true and at least one dot11GASAdvertisementID MIB attribute exists.
47	111	Roaming Consortium	The Roaming Consortium element is present if dot11InterworkingServiceActivated is true and the dot11RoamingConsortiumTable has at least one entry.
48	112	Emergency Alert Identifier	One or more Emergency Alert Identifier elements are present if dot11EASActivated is true and there are one or more EAS message(s) active in the network.

1) Interworking information element: It contains information about the interworking service capabilities of a WLAN network. Figure 3.5 shows the format of interworking information element and is further subdivided into following sub fields:

a) Element ID: It is fixed to be 107.

b) Length: Its value is equal to 1 plus sum of lengths of each optional field. Accordingly, its value can be 1 or 3 or 7 or 9.

c) Access Network Type: This one octet long field specifies the type of corresponding access network. As shown in Figure 3.5, it is further subdivided into the following fields:

i) Most significant four bits (B0 to B3): These represent the type of access network from one of the following eight possibilities:

1) Private network: (value 0000) Non-authorized users are not permitted on this type of network. Examples of this type of network are home networks and enterprise networks. These may employ user accounts.
2) Private network with guest access (B0 B1 B2 B3 = 0001): These types of access network are the same as that of Private Networks, but guest accounts are available.
3) Chargeable public network (B0 B1 B2 B3 = 0010): This type of network can be accessed by anyone. However, the access is not free and requires payment. Example of this type of access network is the hotspot in a coffee shop or a hotel room, where the network is accessible to anyone but will be charged.
4) Free public network (B0 B1 B2 B3 = 0011): Examples of this type of access network is hotspot at a public place such as airport etc where the network is accessible to anyone and there is no charge for the usage of the network.
5) Personal device network (B0 B1 B2 B3 = 0100): It is a network of personal devices, for example camera attached to a printer, thus a network formed for the purpose of printing pictures.
6) Emergency services only network (B0 B1 B2 B3 = 0101): This type of network is dedicated and is limited to the access of emergency services only.
7) Test or experimental network (B0 B1 B2 B3 = 1110): This type of network is used for test or experimental purposes only.
8) Wildcard network (B0 B1 B2 B3 = 1111): Wildcard access network type.

ii) Bit B4: Using this bit the AP specifies about whether the network provides connectivity to the internet or not. B4 = 1 means the network provides connectivity to the Internet. Rather B4 = 0 means that it is not specified whether the network provides connectivity to the Internet.

iii) Bit B5 (ASRA: Additional Step required for Access): AP set this bit to 1 to indicate that the network requires further step for allowing its access.

iv) Bit B6 (ESR: Emergency Services Reachable): AP set this bit to 1 to indicate that emergency services are reachable through the AP. Rather setting this bit to 0 means that it is not specified about the reachability of emergency services.

v) Bit B7 (UESA: Unauthenticated Emergency Service Accessible): AP set this bit to 1 to indicate that higher layer unauthenticated emergency services are reachable via this AP. Rather, setting this bit to 0 indicates that no unauthenticated emergency services are reachable through this AP.

d) Venue Info: This optional field advertises the information about the venue where the access point is present. The values it can contain are drawn from the International Building Code's Use and Occupancy Classifications [*IEEE standard 802.11u, 2011*].

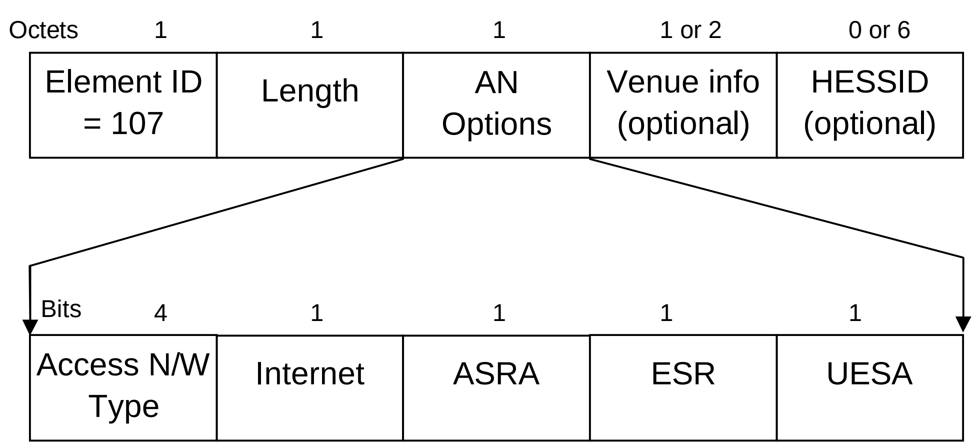


Figure 3.5: Interworking Information Element Format [*IEEE 802.11u, 2011*]

2) Roaming Consortium Information Element: Roaming Consortium is defined as a group of subscription service providers (SSP's) with inter SSP roaming agreement. The security credentials of this Roaming Consortium can be used to authenticate with the access point transmitting this element.

Octets	1	1	1	1	Variable	Variable	Variable
	Element ID	Length	No. of ANQP OI's	OI_1 & OI_2 Length	OI_1	OI_2 (optional)	OI_3 (optional)

Figure 3.6: Roaming Consortium Information Element Format.

Figure-3.6 shows the Roaming Consortium information element format. Here OI stands for Organizational Identifier, which identifies a Roaming Consortium or a single subscription service provider. Roaming Consortium information element can advertise *up to* three OI's in the beacon frame for facilitating a mobile station to choose the WLAN network on the known Roaming Consortium or SSP. The OI's present here are the values of the first three OI's in the dot11RoamingConsortiumTable. The rest of OI's can be queried by the mobile station using Access Network Query Protocol (ANQP). The explanation of fields is as follows:

a) LENGTH: The value of one octet Length field is equal to 2 plus the number of octets in each OI field present.

b) Number of ANQP OI's: The format of this field is a one-octet unsigned integer whose value indicates the additional roaming consortium organization identifiers (OIs) which can be obtained via ANQP. A value of 0 indicates that no additional OIs are listed in Roaming Consortium List and nothing will be returned in response to an ANQP query. On the other extreme, a value of 255 means that greater than or equal to 255 additional OIs are obtainable via ANQP.

c) OI_1 and OI_2 Lengths: Its format is as shown in Figure 3.7.

Bit B0 to B3: These indicate the length in octets of the OI_1 field.

Bit B4 to B7: These indicate the length in octets of the OI_2 field. If the OI_2 field is not present, B4 B5 B6 B7 = 0000.

B0 B3	B4 B7
OI_1 Length	OI_1 Length

Figure 3.7: OI_1 & OI_2 Length field format

d) OI_1, OI_2, and OI_3: These indicate the actual Roaming Consortium Organizational Identifiers.

3) Advertisement Protocol Information Element: It contains information to identify an advertisement protocol which can be used.

4) Emergency Alert Identifier Information Element: It provides a hash to identify instances of the active EAS (Emergency Alert System) messages that are currently available from the network.

3.5 Vertical Handover Taxonomy

Figure 3.8 shows the taxonomy of vertical handover. It shows the three phases of vertical handover: information gathering phase, decision phase, and execution phase. During information gathering phase, the information about network, mobile device and user preferences is required to be captured. This collected information is given to the decision phase and is processed by the corresponding decision algorithm and an alternative network is chosen to which the access is to be transferred. Finally, the execution phase performs this transfer of network.

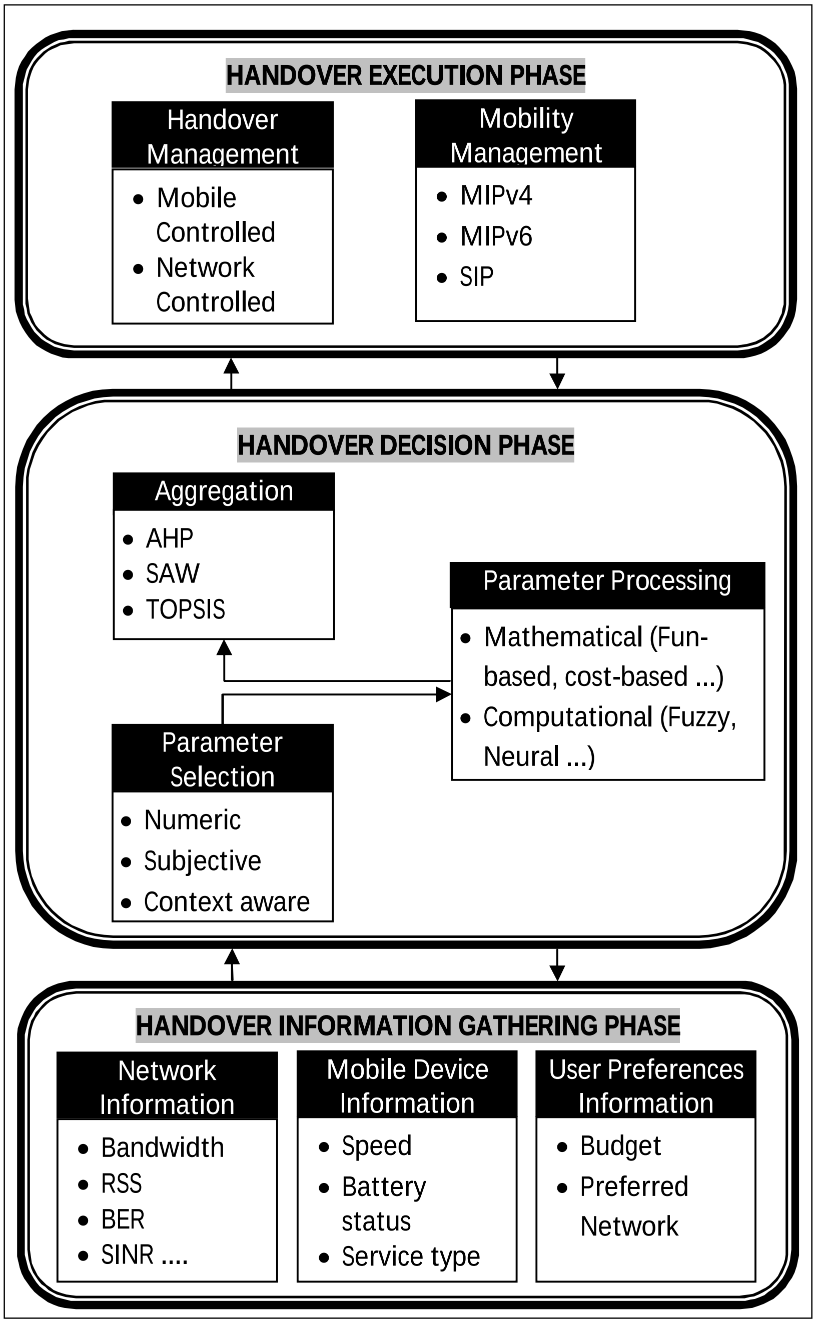


Figure 3.8: Vertical Handover Taxonomy

3.6 Multi-Criteria Decision Making (MCDM)

Multi-Criteria Decision Making is a branch of decision making techniques and refers to making decisions when generally more than one, usually conflicting, criteria are present.

Though MCDM application areas are widely diverse, they share some common characteristics. Some of these are:

a) In the presence of multiple criteria, it is generally rare that all criteria have the same unit of measurement. Rather, each criterion has its own unit of measurement which, as a result, does not allow the direct comparison among them. For example, while selecting a best laptop, cost is indicated in Rupees, processor speed in measured in gigahertz (GHz), battery life is indicated in minutes, and memory in Megabytes (MB)/Gigabytes (GB).

b) There may be conflict among criteria, as a result of which one criterion can be compromised because of other. For example, while designing a laptop the objective of production with low cost may result in sacrificing part of the performance.

c) The goal of decision making can be either to design the optimal alternative or selecting the best alternative from the existing/predefined ones.

In fact, the last characteristic provides a way to classify the MCDM problems into two following categories [*Yoon & Hwang, 1995*]:

1) Multiple Attribute Decision Making (MADM)

The multi criteria problems which come under the banner of MADM category, criteria emerge as a form of attributes. Attributes are defined as factors, or performance parameters, that affect our choice. For example, a customer decides to purchase a car by comparing it with its peers of different attributes, or criteria, like mileage, cost, and color. These are basically selection problems from a predetermined finite number of alternatives. It specifies how attribute information is to be processed to arrive at a choice.

2) Multiple Objective Decision Making (MODM)

In MODM problems, instead of attributes, objectives define the criteria. An objective is a goal for designers to attain, or something which is to be pursued. In the example of designing a car, the objectives (or criteria) for engineers is minimizing the cost of production, or maximizing the car mileage. These are basically the design problems where decision variable values have either an infinitive of a large number of choices. The best of these choices should satisfy the decision maker's constraints and preference priorities. For example, while purchasing the car a customer may have only limited choices since the number of cars kept for sale at any time is finite. On the contrary, while designing a car the number of designs which engineers may have exercised is infinite, or at least a very large number. Also, MODM methods have a set of well defined constraints.

3.6.1 MADM Classification

There are many MADM methods available in the literature [*Yoon & Hwang, 1995; Köksalan, Wallenius, & Zionts, 2011*], with each method having its own characteristics. Basically all the MADM methods can be classified in following two ways:

1) The first way of classifying the existing MADM methods is according to the type of the data they use. Thus, there can be fuzzy, stochastic, or deterministic MADM methods.
2) Another way of classifying MADM methods is according to the number of decision makers involved in the process of decision making, thus having single decision maker MADM methods and group decision making MADM methods.

Since the theory presented in this chapter is only relevant to the research study, here the discussion is limited to: (a) deterministic and (b) single decision making MADM methods. In this scope the most popular MADM methods in practice today are Weighted Sum Model (WSM), Weighted Product Model (WPM), Analytic Hierarchy Process (AHP), revised AHP, and TOPSIS [*Yoon & Hwang, 1995; Köksalan, 2011*].

Before describing one of the most used methods, i.e. AHP, few definitions are required to be explained:

1) Alternatives: These represent the different choices which as a result of action are available to the decision maker. Usually, these set of alternatives are assumed to be

finite. The corresponding MADM method is suppose to screen, prioritize and eventually rank the alternatives.

2) Incommensurable Units: Different criteria may be associated with different units of measure. This consideration of different units makes MADM problems to be intrinsically hard to solve.

3) Criteria: Each MADM problem is associated with multiple criteria. These represent different dimensions from which the alternatives can be perceived. Some MADM methods require arranging the criteria in a hierarchical manner, i.e. associating major criteria with several sub-criteria, sub criteria with several sub-sub-criteria, and so on.

4) Decision Weights: Most of MADM methods require assigning weights to the criteria. These weights represent the importance of one criterion over the other. Usually, these weights are normalized to add to one. The method of assigning these weights depends upon the MADM method in use.

5) Decision Matrix: Let

$A = \{A_i$, for $i = 1, 2........ M\}$ be the finite set of alternatives

$C = \{C_j$, for $j = 1, 2.........N\}$ be the finite set of criteria according to which the desirability of an action is judged.

$W = \{W_j$, for $j = 1, 2........N\}$ be the weights of relative performance of decision criteria. Assuming that these weights are already determined by the decision maker, the MADM problem can be expressed in the form of a decision matrix as shown in Figure 3.9. Here a_{ij} ($1 \leq i \leq M$ and $1 \leq j \leq N$) indicates the performance of alternative A_i in terms of decision criteria C_j.

Alter-natives	**CRITERIA**				
	$\mathbf{C_1}$	$\mathbf{C_2}$	$\mathbf{C_3}$	**....**	$\mathbf{C_N}$
	W_1	W_2	W_3	....	W_N
$\mathbf{A_1}$	a_{11}	a_{12}	a_{13}	....	a_{1N}
$\mathbf{A_2}$	a_{21}	a_{22}	a_{23}	....	a_{2N}
$\mathbf{A_3}$	a_{31}	a_{32}	a_{33}	....	a_{3N}
.	.	.	.	.	.
.	.	.	.	.	.
$\mathbf{A_M}$	a_{M1}	a_{M2}	a_{M3}	....	a_{MN}

Figure 3.9: Decision Matrix

Now, given the decision matrix, the problem is to find the optimal alternative A_k ($1 <= k <= M$) which has the highest degree of desirability with respect to all the criteria.

In general following are the three steps required to be executed:

Step 1: Determine the relevant criteria and alternatives for the decision problem.

Step 2: Find and attach numerical measures to the relative importance of the selected criteria. Also, attach numerical measures to the impacts of the alternatives on these criteria.

Step 3: Process these numerical values to determine the ranking of all the alternatives.

3.6.2 ANALYTIC HIERARCHY PROCESS (AHP)

When we think, we not only identify objects or ideas but also relations among them. Once an object or idea is identified, we tend to *decompose* its complexity up to a manageable and understandable granularity. Analogously, while discovering the relations, we *synthesize* the constituent objects or abstract entities into a single unified entity. So, the fundamental process of underlying perception contains *decomposition* and *synthesis* [*Saaty, 1980*]

Here it is worth noticing that even though the decomposition of reality may differ from person to person, the operational level evaluation tends to be close to each other. This is particularly true when supported with successful experience. Therefore, the reality may be modeled differently by different people, but we very well manage the communication of sense of judgement which involves common understanding. With the purpose of developing a theory behind this and providing methodology for modeling unstructured problems in the social, economic, and management sciences, Thomas L. Saaty proposed a technique called as Analytic Hierarchy Process (AHP).

Based on mathematics and psychology, AHP is a popular MCDM technique for organizing and analyzing complex decisions. Structuring complex problems well and explicitly considering multiple criteria leads to more informed and better decisions.

3.6.2.1 AHP Process

AHP views each MCDM problem as a composition of several alternatives available and various criteria on which these alternatives can be judged.

The main advantage AHP provides is that it permits a hierarchical structure of criteria, thus allowing users to concentrate on individual criteria and sub-criteria when assigning weights. AHP's strength also lies in the fact that the qualitative as well as the quantitative criteria can be evaluated on the same preference scale.

The various steps involved in solving any MCDM problem using AHP are:

Step 1: Problem structuring / developing a hierarchy

For practical purposes a system is often regarded in terms of its Structure and Function [*Saaty, 1980*]:

a) Structure: systems are characterized according to the biological, social, physical, or psychological arrangements of its parts and according to the flow of material and people which define the relations and dynamics of the structure.

b) Function: Systems are characterized according to the functions of the components of the system.

Though structure and function are two different terms, with respect to a system they cannot be separated. We experience them as reality. When we look at them simultaneously, the structure serves as a vehicle for analyzing a function. Hierarchy is an abstraction of the structure of a system which facilitates to study the functional interactions of the components. Representing a system in the hierarchical fashion facilitates in clearly understanding the affect of priority changes of higher level elements on the priorities of lower level elements. In AHP, a problem is structured as a hierarchy followed by the process of prioritization.

Two immediate questions arise for structuring the systems hierarchically:

a) How to hierarchically structure the functions of a system, and

b) How to measure the impacts of an element in the hierarchy

In practice, there is no set procedure to answer the first question. Modeling the problem is perhaps the most creative part of decision making because it can affect the outcome significantly. With respect to AHP, to structure the problem the decision maker(s) and the facilitator sit together. It is a crucial step and should be done with great care because different structure may lead to different final ranking. A useful way to proceed for hierarchically structuring a decision is to come down from the goal as far as feasible, and in fact as far as one can, by decomposing it into most general and most easily controlled factors. Some suggestions for an elaborative design of a hierarchy as listed by Saaty [*Saaty, 1994*] are:

- Identify what is been tried to accomplish. In other words, try to answer the question "What is the overall goal".
- Identify the sub goals of the overall goal.
- Identify criteria that should be satisfied to fulfill the sub goals.
- Identify sub criteria under each criterion.

- Identify the alternatives which are to be prioritized and put them at the last level of the hierarchy.

While choosing criteria or sub criteria it is to be noted that these may be specified in terms of ranges of values or that of verbal intensities such as low, medium, high.

Once the problem is structured, we are ready to apply AHP technique to prioritize the alternatives.

Step 2: Pair-wise comparison and pair-wise comparison matrix

Instead of directly allocating weights to the various criteria involved, AHP requires comparing two criteria at a time. For it, pair wise comparison matrix is used, which depicts the relative importance of the criteria.

Pair-wise Comparison: Suppose that there are N elements at one level of a hierarchy and one element, say ELE, of the next higher level. Pair-wise comparison refers to comparing the N elements pair-wise in their strengths of influence on ELE. It is to be noted here that at a time only two elements (from N elements) are compared with respect to the element at next higher level (i.e. ELE).

Rank Order Matrix/Pair-wise comparison matrix/Judgement Matrix: For N elements it is a square matrix of order NxN where k^{th} row and k^{th} column ($1 \le k \le N$) represents k^{th} element. This matrix records the preference intensities resulting from pair-wise comparison among elements.

Eq 3.1 shows a typical pair wise comparison matrix for N decision elements.

$$W = \begin{pmatrix} 1 & w_{12} & w_{13} & w_{14} & ..... & w_{1N} \\ w_{21} & 1 & w_{23} & w_{24} & ..... & w_{2N} \\ w_{31} & w_{32} & 1 & w_{34} & ..... & w_{3N} \\ ..... & ..... & ..... & ..... & ..... & ..... \\ w_{N1} & w_{N2} & w_{N3} & w_{N4} & ..... & 1 \end{pmatrix} \qquad \text{(Eq 3.1)}$$

As shown in Eq 3.1 above, each decision element is assigned one row and one column. So, for N decision elements the judgement matrix is of order NxN. It is filled by the decision maker by placing the corresponding positive number NUM at the intersection of the j^{th} row (x_j being the corresponding decision element) with the k^{th} column (x_k being the corresponding decision element) such that:

$$NUM = \begin{cases} w_{jk} & \text{if } x_j \text{ dominates } x_k \\ 1/\,w_{jk} & \text{if } x_k \text{ dominates } x_j \\ 1 & \text{if neither } x_j \text{ dominates } x_k \text{ , nor } x_k \text{ dominates } x_j \end{cases} \qquad \text{Eq 3.2}$$

It can be noted that the principal diagonal of the reciprocal matrix will always be 1. Also the lower triangle is the mirror image of the upper triangle, i.e. $w_{kj} = 1/w_{jk}$ for all j, k: 1 to N. So the decision maker has to give the preference values only for the upper triangle (or only lower triangle) decision elements.

Saaty suggested a 9-point scale to perform pair-wise comparison between the alternatives or criteria [*Saaty, 1994*]. This scale is [1/9, 1/8, 1/7... 1, 2... 7, 8, 9] and the entries in the judgement matrix are chosen from it depending upon Table 3.2.

Step 3: Estimating consistency and synthesizing judgements

Since mostly human beings are responsible for filling up the judgement matrix, AHP does not build on "perfect rationality" of judgements, but allows some degree of inconsistency instead. If all the judgements of the decision maker in the judgement matrix are consistent, it must always follow the transitivity rule. Here it is important to highlight the importance and meaning of transitivity. There are two types of transitivity we are talking about:

a) Ordinal Transitivity: It means that the order of preference among decision elements should always be maintained. For example, if X, Y, and Z are three decision elements and X is preferred to Y, and Y is preferred to Z, then X should always be preferred to Z.

Table 3.2: The fundamental scale [*Saaty, 1980*]

The Fundamental Scale:		
Intensity of importance on an absolute scale	**Definition**	**Explanation**
1	Equal Importance	Two activities contribute equally to the objective.
3	Moderate Importance of one over another	Experience and judgement strongly favor one activity over another
5	Essential or strong importance	Experience and judgement strongly favor one activity over another
7	Very strong importance	An activity is strongly favored and its dominance demonstrated in practice.
9	Extreme importance	The evidence favoring one activity over another is of the highest possible order of affirmation.
2, 4, 6, 8	Intermediate values between the two adjacent judgements	When compromise is needed.
Reciprocal	If activity i has one of the above numbers assigned to it when compared with activity j, then j has the reciprocal value when compared to i.	
Rationales	Ratios arising from the scale	If consistency were to be forced by obtaining n numerical values to span the matrix.

b) Cardinal Transitivity: It means that intensity or cardinal preference among decision elements should always be maintained. For example, if X, Y, and Z are three decision elements and X is p times preferred to Y, and Y is q times preferred to Z, then number of times X should be preferred to Z is p*q.

A consistent matrix is the one which is cardinally transitive, and hence ordinally transitive. Rather, an inconsistent matrix need not be either. As priorities make sense only if derived from consistent or near consistent matrices, a consistency check must be applied. Saaty (1977) has proposed a Consistency Index (CI), which is related to the eigenvalue method. CI can be described as Eq 3.3:

$$CI = \frac{\lambda max - N}{N - 1} \qquad \text{(Eq 3.3)}$$

Where N is the order of the judgement matrix and λ_{max} is the maximal Eigen value. Using CI, Consistency Ratio (CR) can be calculated as Eq 3.4

$$CR = \frac{CI}{RI} \qquad \text{(Eq 3.4)}$$

Where RI is the Random Index, and is defined as the average CI of 500 randomly filled matrices. Saaty calculated the random indices as shown in Table 3.3:

Table 3.3: Random Index

N	3	4	5	6	7	8	9
RI	0.58	0.9	1.12	1.24	1.32	1.41	1.45

Saaty proposed that if the judgement matrix has CR < 10%, then it can be considered as having an acceptable consistency. Therefore, if the consistency of the judgement matrix is acceptable, then we are ready to synthesize the judgements; otherwise the decision maker is asked to reconsider his/her judgements.

Step 4: Overall Priority Ranking

Once a consistent (or nearly consistent) judgement matrix is ready, the corresponding decision elements can be ranked. There are various methods of getting this ranking, but

Saaty proved mathematically that Principal Eigen Vector of the judgement matrix gives the best ranking of the decision elements.

Step 5: Aggregation

Performing/executing step-1 to step 5 above, results in a decision matrix as shown in Figure-3.9. Now the last step is to synthesize the local priorities across all criteria in order to determine the global priority. This step determines the global priorities of alternatives by synthesizing all the local priorities. Originally, it was achieved using additive aggregation (Eq 3.5), much similar to SAW.

$$P_i^G = \sum_j w_j . l_{ij} \qquad \text{(Eq 3.5)}$$

Where: P_i^G = global priority of i^{th} alternative

l_{ij} = Local priority of i^{th} alternative w.r.t. j^{th} criteria

w_j = weight of j^{th} criteria

3.6.2.2 Advantages of AHP over other MCDM techniques

Following are the advantages of AHP over other MCDM techniques because of which the quantification of its shortcomings have been considered in this doctoral thesis:

1) Psychologists argue that for a decision maker it is easier and more accurate to express the opinion on only two alternatives at a time, and AHP requires the same.

2) It uses ratio scale, so no units are required for comparison.

3) Allows qualitative (i.e. subjective, which is more familiar in our daily lives) as well as numerical (i.e. objective) judgements.

4) The most popular vertical handover parameter aggregation algorithms are [*Barja et al, 2011*]: (a) AHP, (b) Grey Relational Analysis (GRA), (c) TOPSIS, (d) Simple Additive Weighting (SAW), and (e) Multiplicative Weighting Exponent (MWE). These methods require assigning weights to criteria and alternatives with respect to each criterion. Since AHP allows consistency check and pair wise comparisons, many authors have

used AHP process to assign these weights [*Chen et al, 2009; Sgora et al, 2010*]. Moreover,

5) In continuation to point (4) above, Stevens-Navarro et al [*Stevens-Navarro et al, 2006*] compared the performance of MCDM techniques on different attributes (delay, rate of packet loss, bandwidth, cost) for vertical handoff decision. The results conclude that mostly (i.e. 72.34% to 99.84%) all algorithms resulted the same network selected [*Stevens-Navarro et al, 2006*]. Thus performance depends heavily on the correct weights assigned to the decision criteria.

6) Salomon & Montevechi [*Salomon & Montevechi, 2001*] compared AHP process with other MCDM techniques and concludes that AHP is not an inferior method with any other MCDM methods. It further concludes that "*several advantages of the application of the AHP has been observed in all the cases*" [*Salomon & Montevechi, 2001*].

3.7 UMTS QoS Classes

ETSI TS 123 107 [*3GPP TS 23.107, 2012*] defines four different traffic service classes. These are:

a) Conversational Class: Here the required characteristics are strictly governed by human perception and mainly consists of real time services. This traffic class is the most delay sensitive and the time relation between information entities of the stream should be preserved. Some example services which come under this traffic class are telephony speech, voice over IP, and video conferencing.

b) Streaming Class: This scheme of real time stream applies when the user is looking at real time video and/or listening to real time audio. As with conversational class, here also the time relation between information entities of the stream should be preserved, but does not have any stringent requirements on low transfer delay. The services which come under this traffic class are characterized by one way transport.

c) Interactive class: This consists of services where an end user requests data online from remote equipment. The end user can be a human being or a machine. Here round trip delay time is a key attribute. Some example services which come under this traffic class are web browsing, database retrieval etc.

d) Background class: This traffic class consists of services where the end user sends and receives data files at the background. Here, the destination does not expect the data within a certain time, but the payload should be preserved. Some example services which come under this traffic class are downloading of databases, files, background delivery of emails etc.

CHAPTER 4

A Novel Approach for Beacon Stuffing with Non-Standard Custom Information

This chapter explains the improved technique for beacon stuffing. Section 4.2 explains the three fields proposed for stuffing additional information. Section 4.3, 4.4, and 4.5 explains the concepts of fragmentation, control information, and data integrity check. Section 4.6 explains the proposed algorithm and section 4.7 shows the results of implementing the technique in ns-3.

4.1 Introduction

In an infrastructure Basic Service Set (BSS) beacon frames are a type of management frames and are broadcast periodically by the Access Point (AP). Allowing the configurable time period between the two successive beacon frames, it serves two basic purposes:

- To announce the presence of a wireless network in the vicinity, and
- To broadcast the necessary and basic information about the network.

If a wireless client needs to communicate using 802.11 network and it is in passive mode, it first listens to the beacon frame. Receiving the beacon frame inherently means that the corresponding AP, and thus network, is reachable. Using the information embedded in it, the device decides whether to communicate with a particular AP (as there can be multiple beacons from multiple APs in the vicinity) or not. If yes, it attempts "Association". This decision to attempt Association with a particular AP is heavily based on the parameters standardized by the protocol and is present in various fields of the beacon. Once Association is successful then only the communication can start.

The syntax, semantics and arrangement of the information in the beacon is standardized by 802.11 standard. This is to facilitate the communication between any two 802.11 compatible devices, especially when they are manufactured by different vendors.

Even though the mobile device is associated to a particular 802.11 network, it continuously receives all the beacon frames of the corresponding networks in the vicinity. So, other than serving the two purposes as specified above, the beacon frame is often looked upon from a different perspective: a carrier which is advertising the information to the mobile devices in the vicinity. Often this perspective of the beacon frame is exploited in numerous ways. Section 2.1.2 lists few of them. Here the meaning of the word advertisement is not restricted to only commercial products advertising, rather it means inherent broadcast of any information.

This advertising perspective of the beacon frame often requires embedding additional non-standard information in it without changing the meaning of the beacon as per the standard. R. Chandra et al [*Chandra et al, 2007*] had proposed to use SSID, BSSID, and Vendor Specific fields of 802.11 beacon for carrying such information. Section 2.1.1 lists the various limitations of their proposed work. In this chapter, other than the BSSID and Vendor Specific fields, as a novel contribution it is additionally proposed to use the Length fields of all the Information Elements present as a part of its frame body. More specifically, in addition to BSSID and Vendor Specific fields, it is shown that considering IEEE 802.11-2007 standard and all its ten amendments (and thus IEEE 802.11 - 2012 standard), up to 24 octets of data, and not information, always gets transmitted in Information Element fields of every beacon frame. It is proposed to exploit these Information Elements to embed any additional information in them. Also, 802.11 standard specifies the maximum size of the beacon frame. Using it, the amount of additional information to be embedded in the three proposed fields is maximized. It is also shown that how to do fragmentation of the large chunk of data such that it can be embedded in multiple successive beacon frames.

The scope of the work is limited to the following:

a) The ideas and results presented are valid up to the standard IEEE 802.11-2012 published in March 2012 [*IEEE standard 802.11, 2012*]. This standard incorporates Amendments 1 to 10 [*IEEE standard 802.11k, 2008; IEEE standard 802.11r, 2008; IEEE standard 802.11y, 2008; IEEE standard 802.11w, 2009; IEEE standard 802.11n, 2009; IEEE standard 802.11p, 2010; IEEE standard 802.11z, 2010; IEEE standard 802.11v, 2011; IEEE standard 802.11u, 2011; IEEE standard 802.11s, 2011*] of base IEEE 802.11-2007 standard [*IEEE standard 802.11, 2007*], published since 2008 to 2011.

b) Many operation modes can be supported by WLAN devices. These are infrastructure, ad hoc, virtual access point, wireless distribution system, mesh, virtual interface and monitor. Everything presented here is related to infrastructure mode only. Of course, further research can be done for other modes of operation also.

4.2 Fields for embedding information

Depending upon the length of the information to be embedded, the proposed technique uses three fields for embedding additional non-standard information. As proposed, it is not mandatory to use all the three fields, but any combination of these can be used. These fields are explained in section 4.2.1, 4.2.2, and 4.2.3.

4.2.1 BSSID field of beacon frame header

It is always present as a part of Medium Access Control (MAC) header in the beacon frame and is a 6-octet field. It uniquely identifies each BSS. More specifically, it indicates the MAC address currently in use by the station contained in an AP.

If it is free then it should be preferred over Length and Vendor Specific Information Element. This is because, whether free or not, BSSID will always be present as a MAC header and thus always gets transmitted. If INFOLEN is the length of the information to be embedded, the first six octets of INFOLEN can be stuffed in it.

4.2.2 Length field of Information Elements

Figure-3.4 shows the general format of an Information Element which can be present as a frame body of the beacon frame. EIDs are numbered from 0 to 255. Out of these, the unspecified EIDs are reserved by IEEE. Corresponding to each EID the Length field specifies the exact length of the Information field.

Though the Information field is a variable length field, its minimum and maximum length is fixed by 802.11 standard. In other words, the minimum and maximum value which can be stored in the Length field corresponding to each EID is known and fixed. Since one octet has been assigned to Length field and there are many Information fields whose maximum length is always less than or equal to 255, it is these Length fields of Information Elements which makes it suitable for carrying additional information. For example, BSS Average Access Delay Information Element (with EID 63) can have maximum length of Information field to be of one octet only. So, the Length field will always contain the value 1, thus leaving 7 most significant bits with value 0 always.

To get the total number of free bits, the data of all the Information Elements, which can be the part of beacon frame up to 802.11-2012 standard, was compiled. Since all the WLAN's does not necessarily implement the functionality of all the ten amendments, the compiled data reflects the availability of free bits with each successive amendment. This is shown in Table A.1 of Appendix-A. Figure-4.1 shows the graph depicting total number of free bits available with each successive amendment to IEEE 802.11-2007. From Table A.1 it can be concluded that if all the Information Elements are part of the beacon, 191 bits of additional information can be overloaded in the Length field.

Although the technique gives a choice for using the Length field, if it is used then it should be preferred over the Vendor Specific field. This is because these bits always gets transmitted, thus embedding information does not increase the size of the beacon. The number of free octets (let's say FLEN) in all the Length fields is calculated first because the number of Information Elements that can be the part of beacon is not fixed and depends upon the functionality implemented at the AP. If the first six octets of Information are embedded in BSSID, then the FLEN octets of information starting from the seventh octet of INFOLEN can be stuffed into it.

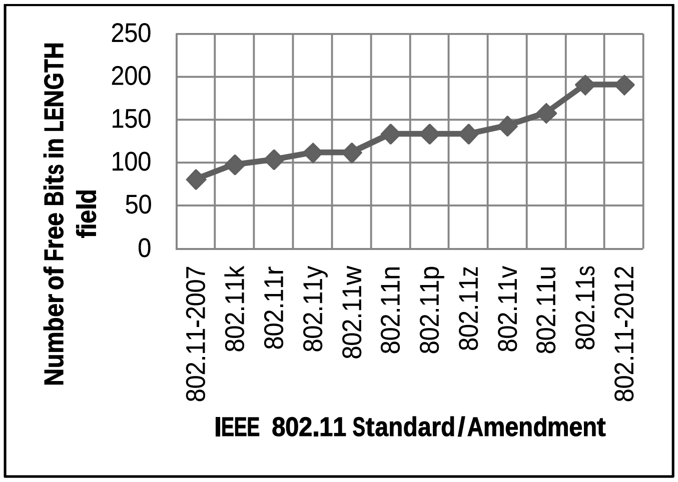


Figure 4.1: Free Bits with each successive amendment to IEEE 802.11-2007.

Also, for embedding information in the Length fields of the Information Elements, an array data structure is used. It is defined as integer FREE_BITS[255], where FREE_BITS [i] is the maximum number of available bits in the Length field corresponding to EID [i], 0 ≤ i ≤ 255. Since 802.11-2007 standard (and all its successive amendments) has fixed the maximum length of Information field corresponding to each EID, the values in this array are fixed up to a particular amendment. Only when the functionality of a new amendment is to be implemented, this array values need to be changed.

4.2.2.1 Advantages of Embedding the Information in the Length field

Following are the advantages of beacon overloading in the proposed Length field:

a) To implement the information embedding in the Length field, the WLAN drivers are required to be modified at the AP and the mobile device. Though it appears to be a limitation, it is not because using Vendor Specific field also requires changes in the drivers.

b) The Information embedding algorithm, as explained in section 4.6, is scalable with respect to embedding Information in the Length fields of Information Elements. This is because when the functionality of an existing WLAN si to be modified / extended to include the functionality of any new amendment, the change required is minimal. In fact, only the FREE_BITS data structure is required to be modified. The rest of the embedding function will remain as it is.

c) Embedding Information in the Length field results in better utilization of channel resources. This because, without any Information in the Length fields the channel resources are used for only transmitting "zeroes". Whereas with Information embedding the same channel resources are used for transmitting Information contents.

d) If the information can be embedded in all the Length fields only, there are no extra network resources required for the transmission of information. Of course, the computational resources at the two end points (i.e. AP and wireless client) are required for embedding and extracting the information.

4.2.3 Vendor Specific Information Element

To allow flexibility to the vendors for implementing the optional functions and proprietary features, 802.11 standard has a provision to carry Vendor Specific non-standard information in the beacon frame. Vendor Specific Information Element (with Element ID 221) is provisioned to be the last Information Element in the frame body of the beacon frame and can carry information contents up to 252 octets [*IEEE Standard 802.11, 2007*].

A beacon can have multiple Vendor Specific Information Elements as long as the size of the beacon is less than the maximum size allowed (i.e. 2320 octets [*IEEE 802.11-2012, 2012*]). Possibly, using Vendor Specific Information Element for transmitting Information should be at the last priority. If FLEN+6 octets of INFOLEN is already stuffed in Length fields and BSSID field correspondingly, the next (up to) [2320-(INFOLEN - (FLEN+6))] octets of INFOLEN, starting from the (FLEN+7)th octet of INFOLEN can be stuffed into it.

4.3 Fragmentation

If the size of INFOLEN is large such that the complete information cannot be stuffed in a single beacon, the information can be fragmented such that multiple successive beacons can be stuffed with the fragmented parts. To indicate fragmentation, only 1 bit is sufficient. If the bit reads 0, that would imply that it is the last fragment or trivially that it is the only fragment. If the bit reads 1, it would imply that more fragments would follow.

Since the Length field of SSID Information Element has most significant two bits free and is always present in the frame body of the beacon, the bit to indicate fragmentation is added to it. The mobile device reassembles the information by using this fragmentation bit and the 12-bit sequence number of the sequence control field structure of MAC header.

4.4 Control Information

To signify to the client device about the fields (out of the three proposed) in which the additional information is stuffed, control information is required to be passed. For it, the Length field of another Information Element (called Supported Rates) is utilized. It is also always the part of frame body of the beacon frame and has four unused bits in the Length field. These are used to carry control information as specified in Table 4.1.

4.5 Data Integrity Check

Each beacon frame has 32-bit Frame Check Sequence (FCS) field containing a 32-bit CRC. It is calculated over all the fields of the MAC header and the frame body fields. This covers the embedded information as well.

At the receiving end, depending upon the control information, the embedded Information from the corresponding fields is extracted and stored in the buffer. While extracting the Information, using FREE_BITS array data structure the most significant bits of the Length fields carrying custom information is reset 0, i.e. bring it into the format as required by other driver functions. Finally, if the received beacon contains the last fragment of custom Information, the content of the buffer is readied to be delivered to the upper layers.

Table 4.1: Control bits and the associated meaning

Control Bits	Associated Meaning
000	No Information is carried.
001	Only BSSID have additional Information.
010	Only Length fields of Information Elements have additional Information.
011	BSSID and Length fields together have additional Information.
100	Only Vendor Specific Information Elements have additional Information.
101	BSSID and Vendor Specific Information Elements together have additional Information.
110	Length fields and Vendor Specific Information Elements together have additional Information.
111	All three fields have additional Information.

4.6 Algorithm

Here the algorithms for embedding the Information in the beacon frame on the AP side and for extracting the Information on the mobile station side are presented. The exact implementation is shown in Appendix B.

4.6.1 Access Point side

The algorithm for AP side is as shown in Figure 4.2. The time complexity of the algorithm is O(N), where N is the number of bits to be embedded in a beacon frame. Before the beacon transmission begins, all the variables involved are initialized to sensible defaults. During initialization, the information to be embedded over beacon transmissions is added to the **`info`** array.

`info_read` is a counter that maintains the index of octets that have been read from **`info`** and already transmitted. In case that the **`info`** length exceeds what can be embedded into a single beacon, the data will be transmitted in successive beacons. In the next beacon, the info will be read from **`info_read`** onwards. When there is no more data to be read, **`info_read`** is set to zero such that the next beacon carries data from the start.

Depending upon the **`control_rates`**, whose value is one of those as shown in Table-4.1, if the BSSID field is to be used for embedding data, the first 6 octets from **`info`**, starting from **`info_read`** is copied on to the BSSID field.

If there is more data and **`control_rates`** indicate that Length field can be used, proceed to calculate the number of octets than can be embedded to the free bits of Length fields of Information Elements. Information is added only in multiples of 8 bits to the free bits of Length fields. This means that as many as 7 free bits may go unused. Then, copy those many octets from **`info`** to **`fragment_info`** (used as a temporary buffer) and increment **`info_read`** by those many octets.

```
control_rates = get the control data to be added to Supported Rates IE
while beacon is transmitted:
   ResetAP()
   isBssidChanged(control_rates):
      Set BSSID from data from info, starting from info_read
      info_read += 6
   if more data and isLengthChanged(control_rates):
      length_bits = get number of bits from length fields that will be used
      copy (length_bits / 8) octets to fragment_info from info starting from
        info_read
      info_read += (length_bits / 8)
  if more data and isVendorChanged(control_rates):
      vendor_octets_used = get number of vendor octets that will be used
      copy vendor_octets_used octets to fragment_info from info starting
      from info_read
      info_read += vendor_octets_used

  if no more data:
      control_ssid = 0
```

```
        info_read = 0
    else:
        control_ssid = 1
    embed control_ssid in length field of ssid and add ssid to beacon
    embed control_rates in length field of supported rates and add it to beacon
    while more Information Elements:
        beacon.write(element_id)
        free_bits = free_bits_available(element_id)
        isLengthChanged(control_rates):
        mask = GetBits(free_bits)
        new_length = old_length | mask
        beacon.write(new_length)
        beacon.write(element_id_data)
  while vendor_octets_used > 0 and isVendorChanged(control_rates):
        beacon.write(vendor_specific_element_id)
        if vendor_octets_used >= 252:
            length = 255
        else:
            length = vendor_octets_used + 3
        beacon.write(length)
        beacon.write(custom_OUI)
        data = copy length octets from fragment_info starting from   current_index
        beacon.write(data)
        current_index += length - 3
        vendor_octets_used -= length – 3
```

Figure 4.2: Algorithm for beacon stuffing at AP

If there is more data, and **`control_rates`** specify that Vendor Specific Information Elements can be used, then proceed to embedded in Vendor Specific fields. Calculate the number of vendor octets that will be used to carry custom data and copy those many octets from **`info`** to **`fragment_info`** and increment **`info_read`**.

Now **`fragment_info`** contains the data that is yet to be embedded to length and vendor specific fields. **`current_index`** stores the index of octets read from **`fragment_info`**.

control_ssid stores the control data that will be added to the free bits of length field of SSID Information Element. If there is no more data i.e. this is the last beacon carrying the data fragment, **control_ssid** is 0. Otherwise, it would be 1. The functions isBssidChanged(), isLengthChanged() and isVendorChanged() access the **control_rates** and return true/false appropriately.

Now, add the information from temporary buffer, i.e. **fragement_info**, to length fields and vendor specific information elements. For every information element (before Vendor Specific), the following steps occur:

- The element ID of the Information Element is written to the beacon.
- Calculate number of free bits available in the length of that Information Element, store in **free_bits** and pass **free_bits** as the parameter to the GetBits() function.
- The GetBits() function returns the next **free_bits** number of data bits from **fragment_info**. It increments the **current_index** variable appropriately.
- The returned data is masked to the Length field and write it to the beacon.

If there were vendor octets used, first write the 3-octet long Custom OUI to the beacon, followed by the data in Vendor Specific information field. Then, increment **current_index** and decrement **vendor_octets_used** by the number of information octets written to this Vendor Specific element. Lastly, the beacon is transmitted and carries the custom data.

4.6.2 Mobile Station side

The algorithm for Mobile Station side is as shown in Figure 4.3. The time complexity of the algorithm is O (N), where N is the number of bits of Information embedded in a beacon frame. Before the beacon reception begins, initialize all the variables involved to sensible defaults. The information read from the beacon will be added to **sta_info** buffer. **sta_current_index** is a counter that maintains the index of octets that have been read from the beacon.

```
InitializeVariables()
while beacon is received:
    ResetSTA()
    bssid_buffer = getBSSID()
    sta_control_ssid = get control data from SSID of beacon
    sta_control_rates = get control data from Supported Rates of
                              beacon
    if isBssidChanged(sta_control_rates):
         copy bssid_buffer to sta_info
         sta_current_index += 6
    while more info elements (not vendor):
          element_id = beacon.read()
          length = beacon.read()
          free_bits = free_bits_available(element_id)
          data = get bits from the most significant free_bits number of bits
                    from length
          set free_bits MSB of length to 0
          if isLengthChanged(sta_control_rates):
          sta_info = PutBits(data)
          beacon.next(length)
    while more vendor elements:
         element_id = beacon.read()
         length = beacon.read()
         oui = beacon.read(3)
         data = beacon.read(length - 3)
         if oui == customOUI && ifVendorChanged(sta_control_rates):
                copy data to sta_info starting from sta_current_index
                sta_current_index += length – 3
```

Figure 4.3: Algorithm for beacon stuffing at mobile station

BSSID from the beacon is read into **`bssid_buffer`**. The control data from SSID and Supported Rates Information Element is read into **`sta_control_ssid`** and

sta_control_rates, respectively. If there is no more data, i.e. this is the last beacon carrying the data fragment, **sta_control_ssid** is 0. Otherwise, it would be 1.

The functions isBssidChanged(), isLengthChanged() and isVendorChanged() work the same way as in Section 4.6.1, but now the parameter passed is **sta_control_rates**.

If isBssidChanged() return 1, it means BSSID contained custom information and **bssid_buffer** is copied to **sta_info** and **sta_current_index** is incremented.

Now, proceed to process all the Information Elements before Vendor Specific ones. For each Information Element, execute the following steps:

- Read the Information Element EID and Length.
- Calculate number of free bits available in the Length of that Information Element and store in **free_bits**
- Retrieve the data from the most significant **free_bits** number of bits of the Length field to the variable **data** and set those bits in **length** variable to 0.
- If the Information Elements carry custom information, call PutBits() with parameter as the data from **free_bits**.
- PutBits() adds the data to **sta_info** and increases **sta_current_index** appropriately.

Now, proceed to process all the Vendor Specific Information Elements. For each Vendor Specific Information Element, execute the following steps:

- Read the Information Element's EID and Length
- Read the OUI and the data following it.
- If the OUI is custom OUI and isVendorChanged()is true, the data in Vendor Specific Information field is non-standard and we copy it to **sta_info**
- Increment **sta_current_index**.

- Now, the beacon is received and the custom data from this beacon is read into **`sta_info`**.

4.7 Implementation and Results

The above scheme of embedding additional non-standard data to the beacon frame was implemented successfully in ns-3 simulator. Ns-3 is a (discrete-event) network simulator for internet systems, targeted primarily for research and educational use. Ns-3 is free software, licensed under the GNU GPLv2 license [*www.nsnam.org*].

Corresponding to Figure 4.2, i.e. algorithm at AP side, Figure 4.4 shows the functional flow of ns-3 relevant to this work. The functions are:

main - Driver function, where variables are initialized. This driver file sets up one AP and two STAs between which the communication happens.

ApWifiMac::SendOneBeacon - is the function called every time a beacon has to be sent. It sets up and generates the beacon and adds the beacon to the transmission queue.

MgtProbeResponseHeader::GetSerializedSize - Calculates the size of the beacon so that a buffer with that space could be allocated.

MgtProbeResponseHeader::Serialize - In ns3, all the data including BSSID and Information Elements is represented in terms of classes. This function, along with its helper functions, serializes the relevant data from the classes and adds it to the beacon frame.

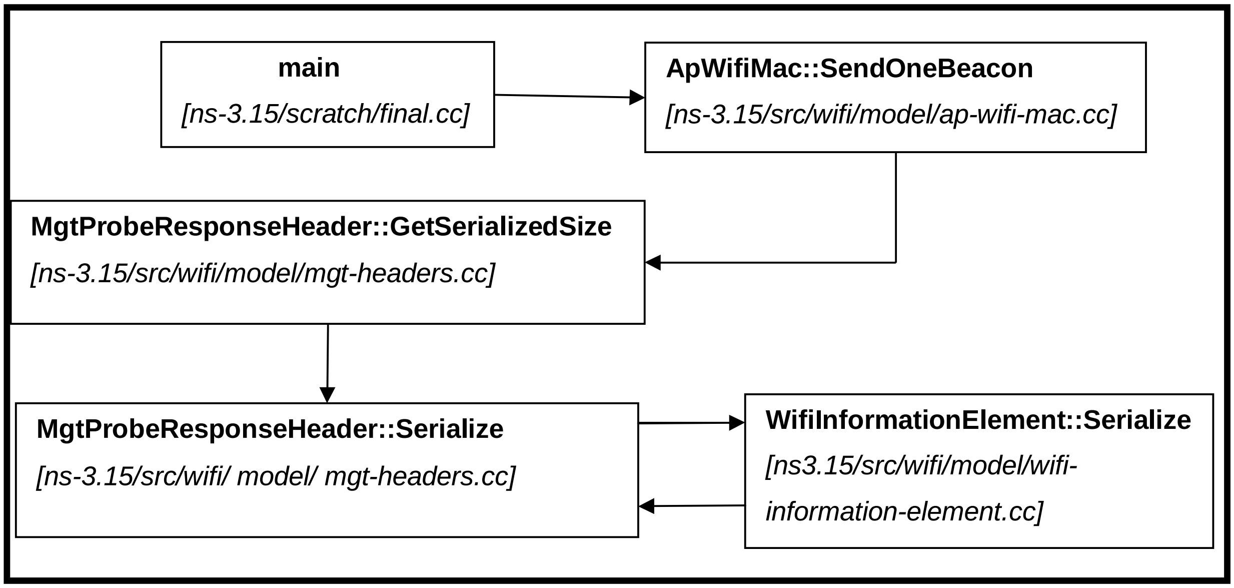


Figure 4.4: Functional flow of ns-3 w.r.t. implementation at AP side.

Also, corresponding to Figure 4.3, i.e. algorithm at Station side, Figure 4.5 shows the functional flow of ns-3 relevant to this work.

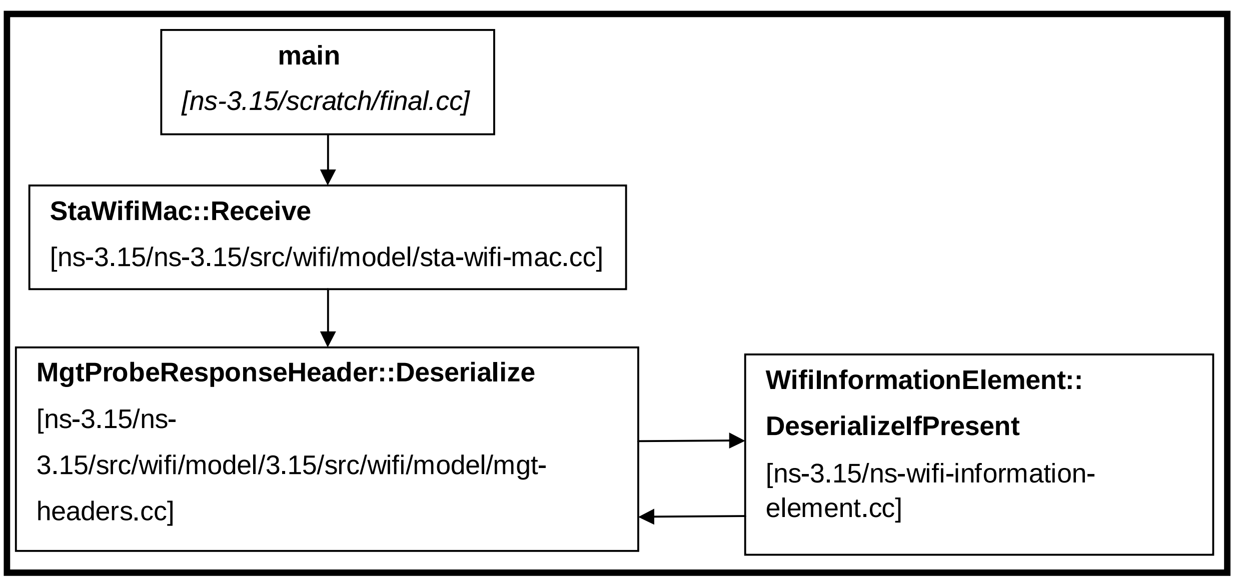


Figure 4.5: Functional flow of ns-3 w.r.t. implementation at Station side.

The amount of custom information that can be embedded in to a beacon frame has been maximized and is shown in the Figure 4.6. The EID's shown on the X-axis corresponds to the ones in increasing order as per the standard [*IEEE 802.11-2007,* *2007]*. Adding another Information Element after the one with ID 66, would exceed the size of frame body of the beacon and hence, is not listed on the graph. The amount of information is represented on a logarithmic scale.

The three lines represent the amount of custom Information which can be stuffed in the three proposed fields. The graph clearly shows that as the number of Information Elements increases:

a) The amount of Information which can be embedded in Vendor Specific elements decreases.
b) The amount of Information which can be embedded in Length field increases.
c) Since the size of BSSID is constant, the amount of Information which can be embedded in BSSID field remains constant.

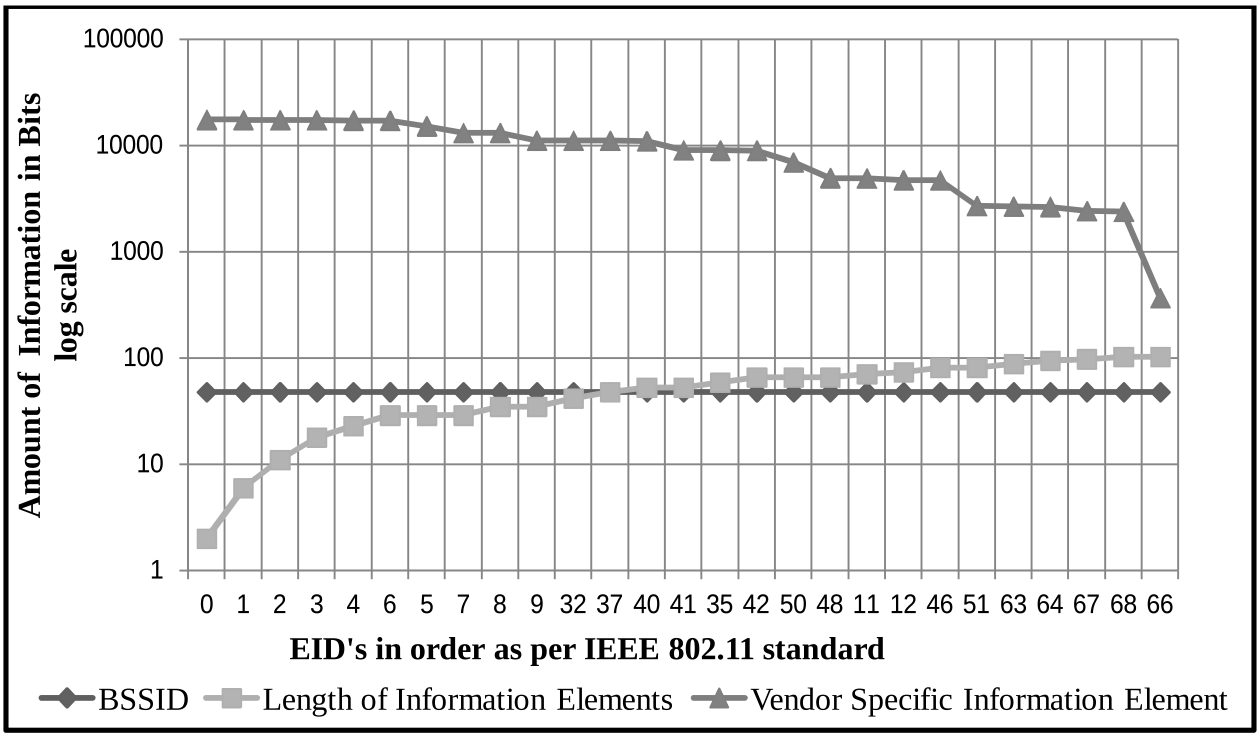


Figure 4.6: Amount of information that can be added versus each additional information element.

Figure 4.7 shows the comparison of the proposed beacon stuffing technique with that of R Chandra's work. Assuming the beacon interval to be 10 msec [*Chandra et al, 2007*], it shows the achieved bandwidth with each successive Information Element added to the frame body of the beacon frame. The EID's on the X-axis starts from 2 as the first two EID's are used for carrying control information and the fragmentation information in the proposed technique. Moreover, the EID's shown on the X-axis corresponds to the ones in order as per the standard [*IEEE 802.11-2007, 2007]*. By successively increasing the Information Elements (constrained to first 30 frame body elements), the graph clearly shows that:

a) Using R Chandra's beacon stuffing technique (with all three fields proposed), a constant bandwidth of 229 Kbps can be achieved.
b) Using the proposed scheme and with successive addition of each Information Element, the achieved bandwidth is in the range of 1775 Kbps to 294 Kbps.
c) When the size of frame body of the beacon frame reaches its maximum limit (i.e. 2320 octets), no additional information can be stuffed as per R Chandra's proposed scheme. But with the improved stuffing technique, some additional information can still be stuffed in the Length field of Information Elements.

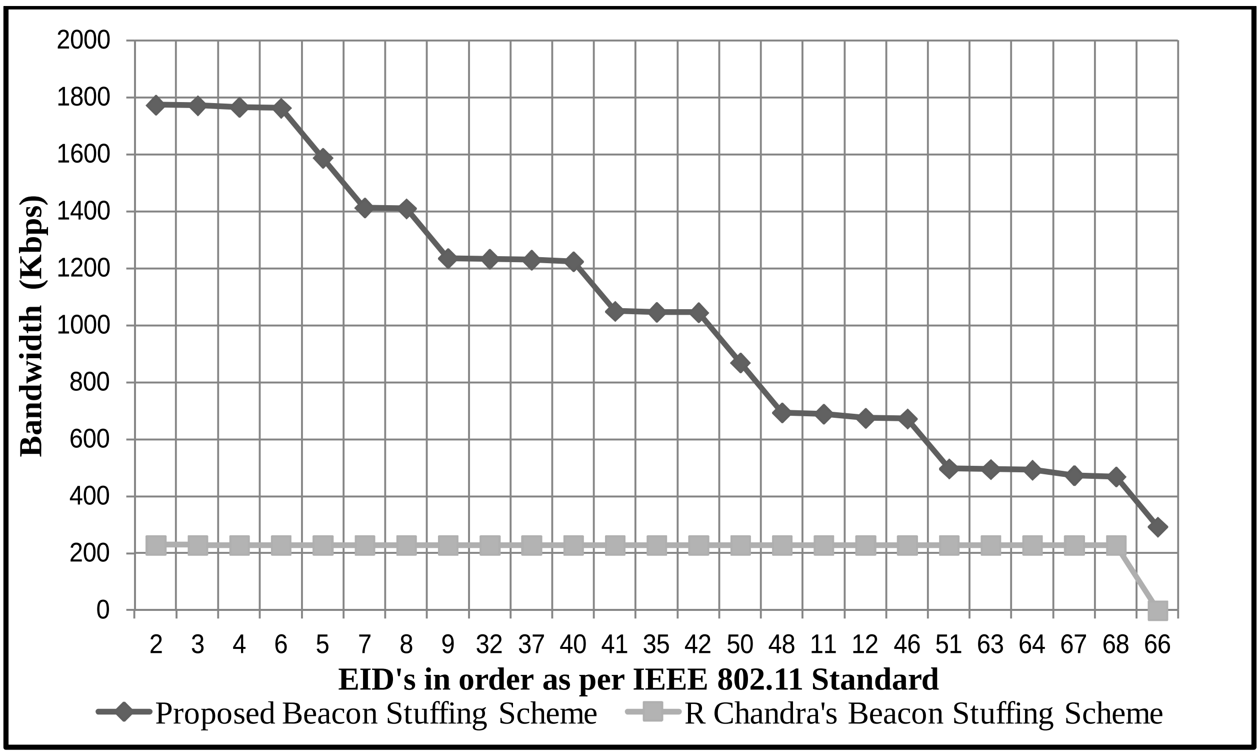


Figure 4.7: Comparison of proposed beacon stuffing technique with that of R Chandra's technique.

CHAPTER 5

Measuring the Effectiveness of Contradictory Judgement Matrices in AHP

5.1 Introduction

This chapter shows the results of comparing contradictory judgement matrices with non-contradictory judgement matrices on the basis of qualitative metric (i.e. local rank reversals), twelve quantitative distance metrics, and a metric called as Minimum Violation. Section 5.2 explains about what type of judgement matrices in AHP qualify to be classified under the banner of "contradictory judgement matrices". Section 5.3 explains about the experimental set up to generate the judgement matrices (both contradictory and non-contradictory) and the results of the simulation to measure this "criticism" in AHP. It also explains about various qualitative and quantitative metrics and the results of comparing contradictory matrices with non-contradictory matrices on these metrics. Section 5.4 proposes a feedback based technique based on the enhanced notion of contradictory entry. Finally, Section 5.5 lists various observation and their interpretations.

5.2 Contradictory Judgement Matrices

AHP uses CI to check the consistency of the judgements specified in pair wise comparison matrix. Depending upon the CI value the judgement matrix is classified as either consistent matrix or in-consistent matrix. Since judgements are provided by human beings, it is quite possible in real world to get slightly perturbed judgement matrix. To quantify this abstract parameter "slightly perturbed", the standard procedure of AHP allows accepting a set of judgements when CI is less than one-tenth of the mean consistency index of randomly generated matrices. This mean consistency of matrices of order 3x3 to 9x9 has been given by Saaty & Vargas [*Saaty et al, 1984a*]. Also, correspondingly CR for such matrix will always be less than 10% [*Saaty, 1980*].

Raising the issue on passing the judgement matrix on only CI value, Kwiesielewicz et al [*Kwiesielewicz et al, 2004*] introduced a class of judgement matrices, called *contradictory matrices*. The biggest issue with contradictory matrix R_c is that it is not possible to rank the corresponding decision elements which satisfies all the judgements given in R_c. Using combinatorics this fact is already been proved. [*Kwiesielewicz et al, 2004*].

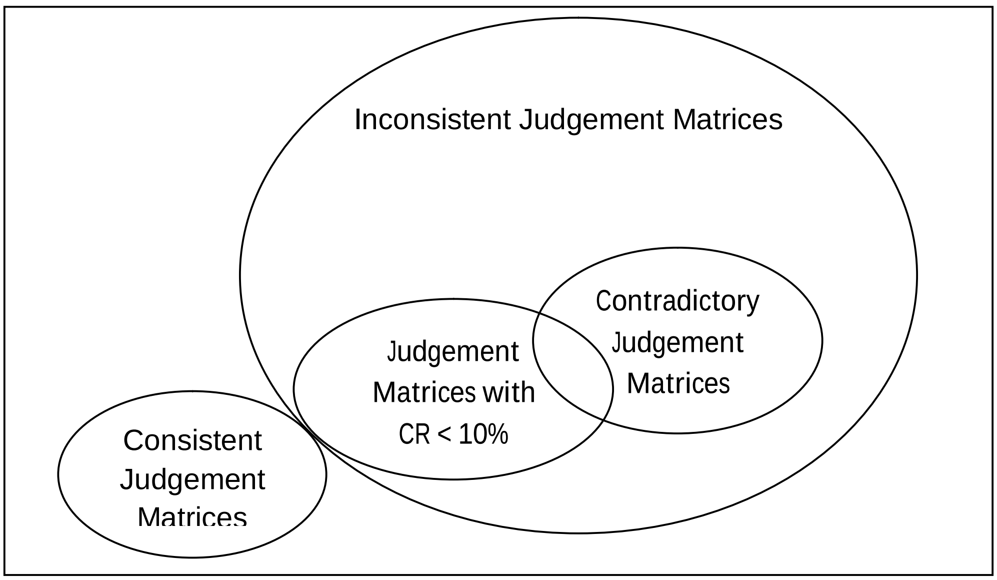


Figure 5.1: Set Structure of Judgement Matrices

The different types of judgement matrices are shown in Figure-5.1. It shows that the set of judgement matrices, called "Inconsistent Judgement Matrices", can be classified into three parts: a) judgement matrices with CR < 10% and allowed to be processed by AHP, b) judgement matrices with CR ≥ 10% and are not allowed to be processed by AHP, and c) contradictory judgement matrices which may or may not be allowed to be processed by AHP.

Formally, the judgement matrix R_c of order NxN is contradictory if there exists a, b, c: 1, 2, ..., N such that any of (Eq 5.1) to (Eq 5.6) holds [*Kwiesielewicz et al, 2004*]:

$$w_{ac} < 1 \text{ AND } w_{ab} > 1 \text{ AND } w_{bc} > 1 \qquad \text{(Eq 5.1)}$$

$$w_{ac} > 1 \text{ AND } w_{ab} < 1 \text{ AND } w_{bc} < 1 \qquad \text{(Eq 5.2)}$$

$w_{ac} > 1$ AND $w_{ab} = 1$ AND $w_{bc} < 1$ (Eq 5.3)

$w_{ac} < 1$ AND $w_{ab} = 1$ AND $w_{bc} > 1$ (Eq 5.4)

$w_{ac} = 1$ AND $w_{ab} = 1$ AND $w_{bc} < 1$ (Eq 5.5)

$w_{ac} = 1$ AND $w_{ab} = 1$ AND $w_{bc} > 1$ (Eq 5.6)

Where w_{ij} is the weight in the i^{th} row and j^{th} column of the judgement matrix; as that depicted in Eq 5.9.

5.3 Simulation, Experimental Set Up, and Results

Contradictory judgement matrices which pass the consistency test have been treated as a setback for AHP as no ranking of decision elements exist which satisfy all the judgements. In order to quantify this problem, thus enabling the comparative study, an experimental test was conducted on matrices of order 3x3 to 9x9. The simulation was done in C language, using a gcc compiler, on a 3.3 GHz Intel core i5 machine.

As explained earlier in section 3.6.2, to fill a judgement matrix the decision maker is required to give the opinion for only upper (or lower) triangle elements. For a matrix of order NxN, the number of elements in the upper triangle is:

$$N_{ut} = \frac{(N * N) - N}{2} \qquad \text{(Eq 5.7)}$$

Considering Saaty's Fundamental Comparison Scale (FCoS) [*Saaty, 1980*], each position in the upper triangle can take 17 different values. Thus for exhaustive analysis the number of matrices to be checked is:

$$N_{ex} = (17)^{N_{ut}} \qquad \text{(Eq 5.8)}$$

This is an exponential number and as N increases, N_{ex} increases exponentially. Considering the limitations of the machine on which simulation was performed, for matrices of order 3x3 and 4x4 it is possible to generate all the possible judgement matrices, thus enabling exhaustive testing. But for higher order matrices it is computationally costly to do exhaustive testing. Therefore, for matrices of order 3x3 and 4x4 an exhaustive analysis was done. But for matrices of higher order, i.e. from 5x5 to 9x9, different method was followed.

Budescu et al. [*Budescu et al, 1986*] and Triantaphyllou et al [*Triantaphyllou, Pardalos, & Mann, 1990*] generated matrices randomly, checked for their consistency, and if corresponding CR is less than 10% then assigned them to corresponding consistency group. There are two issues with this process:

- It does not reflect the process used by decision makers while filling up the judgement matrices [*Ishizaka et al, 2006*].
- There will be very high number of totally inconsistent matrices (CR > 10%).

In fact to prove this second point, initially the same process was followed to get the inconsistent matrices. For a matrix of order NxN (where N varies from 5 to 9) one billion matrices were generated by selecting the upper triangle decision values randomly from the FCoS. Each matrix was checked for CR, and if it is less than 10% it was assigned to a proper Bin. 10 Bins were considered for classifying the matrices with CR < 10%, where Bin[i], $0 \leq i < 10$ contains all the matrices with $i \leq CR < i+1$ and $CR \neq 0$. Though it appears impressive, this method did not result in large number of matrices with CR < 10%. In fact, for matrices of order 5x5 this number was 0.00094% (of 1 billion), for order 6x6 it was 0.000019%, and for 7x7 it was only 0.00000014%. For matrices of order 8x8 and 9x9 this method did not generated any matrix with CR < 10%.

So, instead of generating matrices randomly, there are methods reported in the literature using which the perfectly consistent matrices (with CR = 0) can be generated and then some of the values are perturbed. Also, this process is more close to how decision makers fill the judgement matrices. Moreover, for generating perfectly consistent matrix only the elements in the first row [*Golany et al, 1993*] or first diagonal above the main diagonal [*Ishizaka et al, 2006*] are required to be generated and the other values can be derived using the property of transitivity and reciprocity. Ishizaka et al [*Ishizaka et al, 2006*] introduced the concept of adding impurities by considering errors to be additive.

The simulation was based on following steps (Step 1 to Step 4) where in the consistent and nearly consistent matrices are generated, impurities are added in it, checked if CR< 10%, and if yes it is assigned to proper Bin.

$$W = \begin{pmatrix} 1 & w_{12} & w_{13} & w_{14} & ..... & w_{1N} \\ w_{21} & 1 & w_{23} & w_{24} & ..... & w_{2N} \\ w_{31} & w_{32} & 1 & w_{34} & ..... & w_{3N} \\ ..... & ..... & ..... & ..... & ..... & ..... \\ w_{N1} & w_{N2} & w_{N3} & w_{N4} & ..... & 1 \end{pmatrix} \qquad \text{(Eq 5.9)}$$

Step 1: Randomly choose each element of the first diagonal, above main diagonal, from the Saaty's FCoS values. For example, with respect to Eq 5.9 choose w_{12}, w_{23}, ... w_{NN-1} randomly from the FCoS.

Step 2: Using transitivity rule deduce rest of the values of upper triangle. For example, $w_{24} = w_{23} * w_{34}$. The value deduced here can go outside the range of FCoS or a value may be generated which is not in the FCoS at all. For example, if $w_{23} = 6$ and $w_{34} = 4$, then $w_{24} = 24$. Also, if $w_{23} = 1/4$ and $w_{34} = 7$, then $w_{24} = 7/4$. To consider this aspect, following three different cases are considered here. These cases satisfy the theory that whenever decision maker wishes to enter a value outside the scale; the extreme value in the scale will be chosen. Also, all the values given by decision maker in the matrix will be from the scale only.

- Case 1: If value is outside FCoS then take the boundary value. For example, take $w_{24} = 9$ instead of 24 in the above example. Similar rule follows on the other side of the scale. Also, if the value is not in the fundamental scale then always take the value just prior to it in the scale. For example, take $w_{24} = 1$ instead of 7/4.

- Case 2: If value is outside the fundamental scale then take the boundary value. Instead if the value is not in the fundamental scale then always take the value just after it in the scale. For example, take w_{24} = 2 instead of 7/4.

- Case 3: If value is outside the fundamental scale then take the boundary value. Instead if the value is not in the fundamental scale then always take the nearest value from the scale. For example, take w_{24} = 2 instead of 7/4.

Here it is to be noted that the above three cases does not guarantee to generate perfectly consistent matrices. Since the overall purpose is to generate sufficiently large number of inconsistent matrices, the matrix generated are checked for consistency check and if CR < 10% they are assigned to corresponding Bin. The same matrix is also considered for perturbation in Step-3 below.

Step-3: Now add the impurities in the matrix as suggested by A. Ishizaka et al [*Ishizaka et al, 2006*]. The number of impurities to be added in the upper triangle is randomly selected from the interval [0 , 1 , 2 .... $(N^2\text{-}N)/2$]. Each impurity is equivalent to an additive error term. The number of comparisons (decision values) to be perturbed is also choosen randomly. This value is then replaced with the randomly selected value at position of ±4 of the original position and again checked for consistency. If CR < 10%, the matrix is assigned to proper Bin.

Step 4: The above steps (Step-1 to Step-3) were repeated for matrices of order 5x5 to 9x9. For each order and for each of the three cases all the possible matrices w.r.t. the first diagonal above the main diagonal were generated, the impurities were added, and the resulting matrices were checked for consistency. If CR < 10% then it was assigned to a proper Bin.

The above procedure of adding impurities and generating inconsistent matrices is also shown in the flow chart in Appendix F.

5.3.1 Magnitude of Contradictory Judgement Matrices

The overlapping region of "judgement matrices with CR<10%" and "contradictory judgement matrices" in Figure 5.1 represents contradictory judgement matrices being considered by AHP. As an attempt to quantify this region, all the matrices for each bin and for each order (where order varies from 3x3 to 9x9) were checked for being contradictory. Bin [i] indicates the judgement matrices with $i \leq CR < i+1$ and $CR \neq 0$. The percentage of such matrices was recorded. Table C.1 through C.7 in Appendix C shows the data.

Figure-5.2 shows the percentage of contradictory matrices, of all the possible judgement matrices, for order 3x3 and 4x4. For 3x3 the number of contradictory matrices is very small. It is only 0.7% of all the matrices with CR<10% and that too in Bin [5] only. For 4x4 this percentage is slightly higher, i.e. on average 11.77%. So, for matrices of order 3x3 and 4x4, the probability of a judgement matrix being contradictory is small.

Figure-5.3, Figure-5.4, and Figure-5.5 show the percentage of contradictory matrices for case-1, case-2, and case-3 respectively and each for order 5x5 to 9x9. One thing is clear from all the three cases that percentage of contradictory matrices increases with increase in the order of judgement matrix. Also, there is not much difference between the trends of all three cases. In fact, between case 2 and case 3, the difference between the percentages of matrices got was only marginally different thus showing a very much similar kind of graph.

Figure-5.6 shows the overall summarized results (i.e. average of all) of the percentage of contradictory matrices for order 3x3 to 9x9. The plot shows the results for case-1 and case-3 only as case-2 is very much similar to case-3 (Figure-5.5 and Figure-5.6). It clearly shows that as the order of judgement matrix increases, the percentage of contradictory ones also increases sharply. For order 3x3 it is as low as ≈ 0.7% and for order 9x9 it is as high as ≈ 72.87%. In other words, in AHP as the order of judgement matrix increases, the probability of this matrix qualifying as being contradictory also increases.

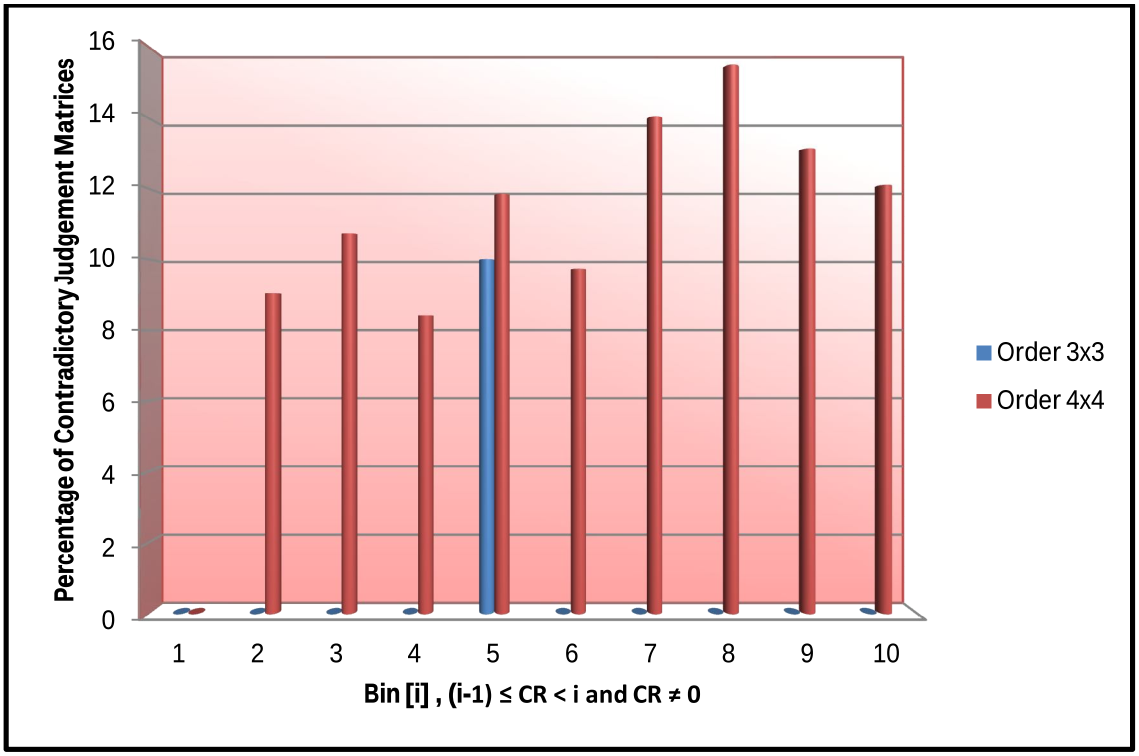


Figure 5.2: Contradictory Matrices for order 3x3 and 4x4.

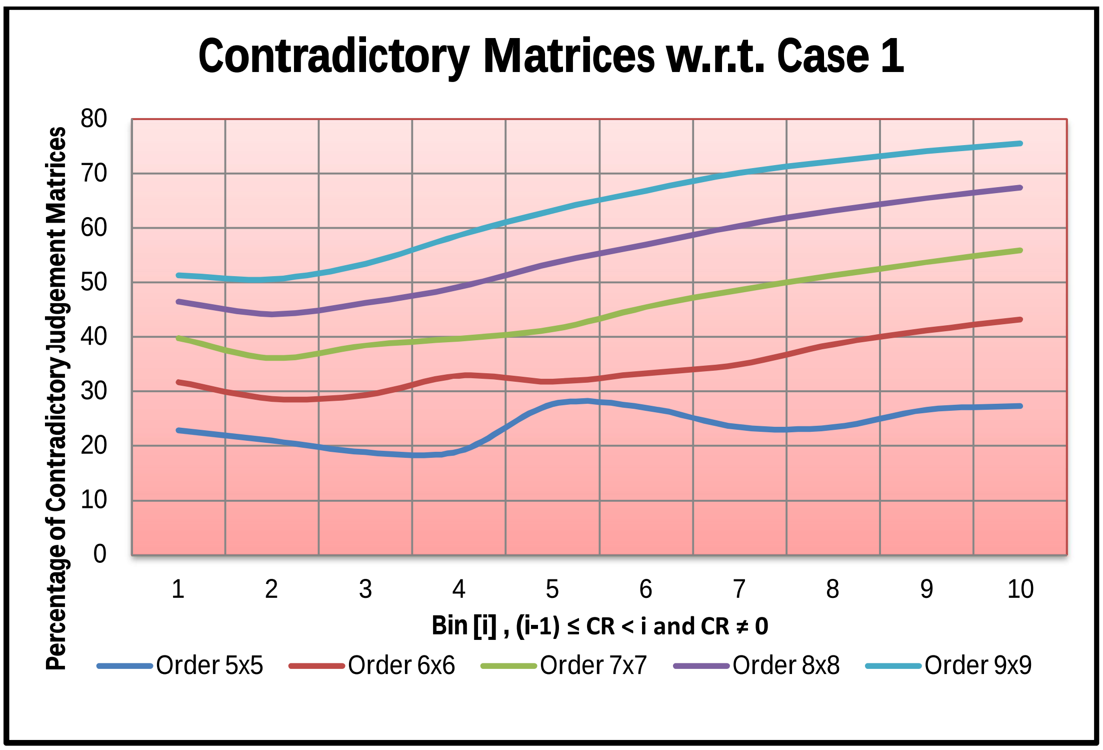


Figure 5.3: Percentage of Contradictory Matrices w.r.t. Case 1

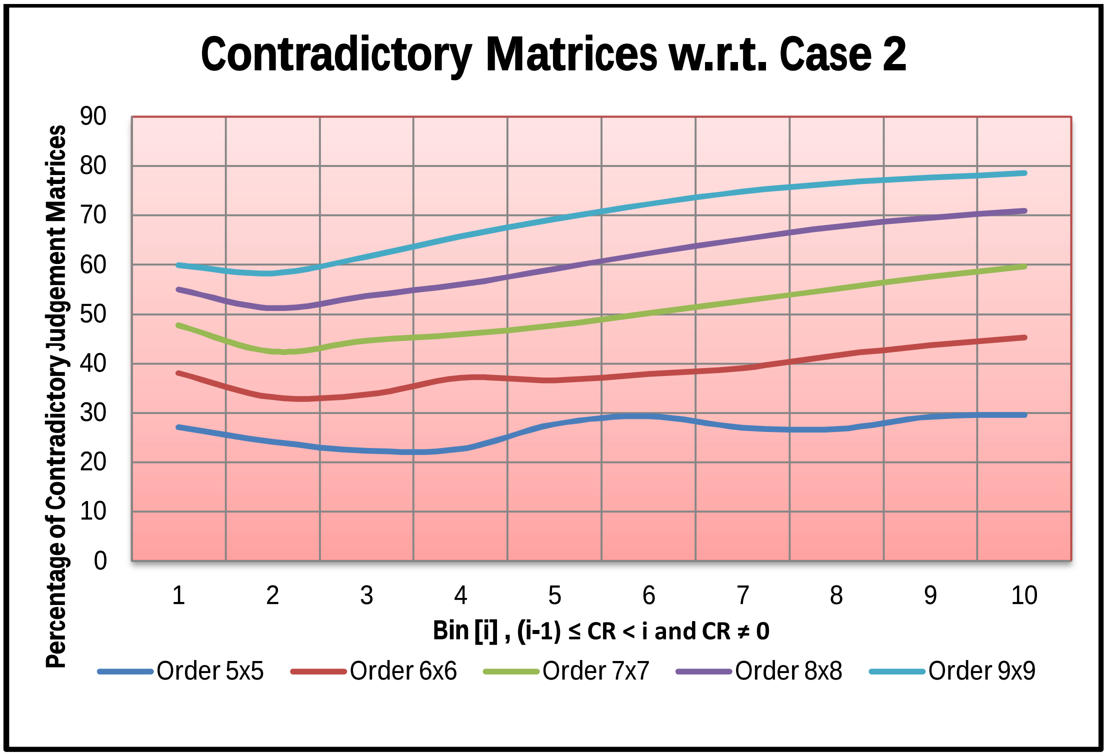


Figure 5.4: Percentage of Contradictory Matrices w.r.t. Case 2

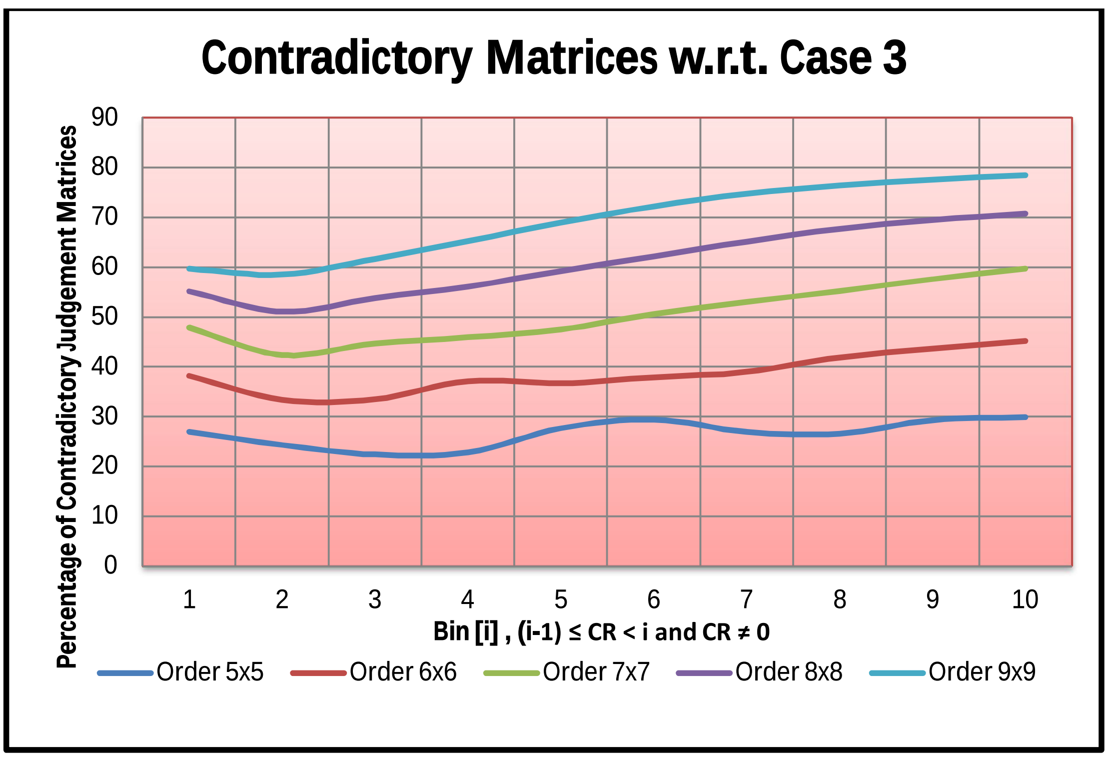


Figure 5.5: Percentage of Contradictory Matrices w.r.t. Case 3

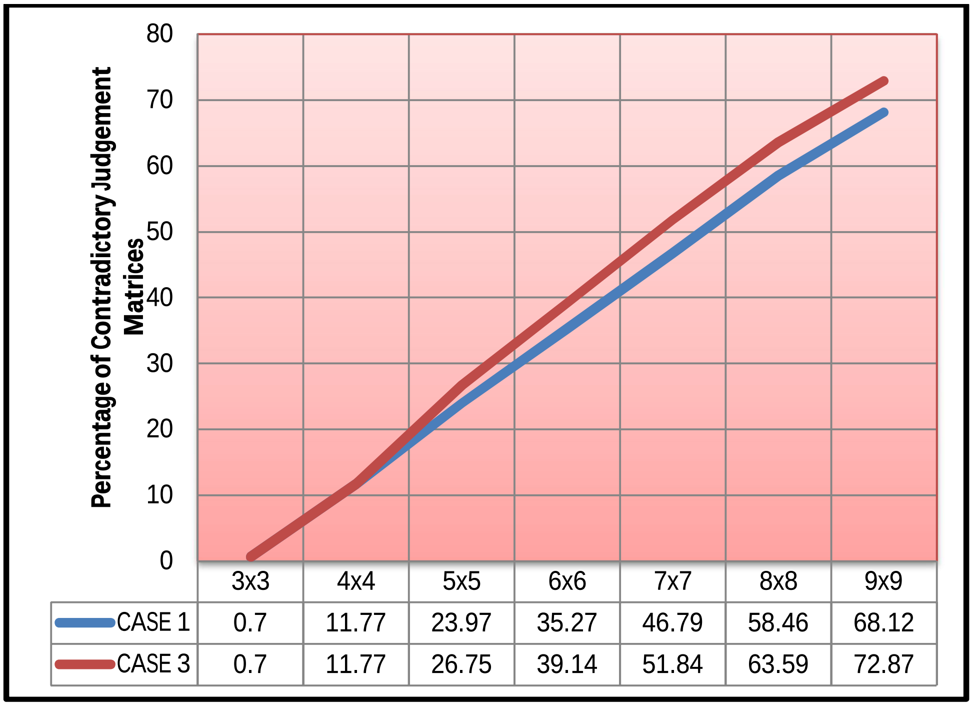


Figure 5.6: Summarized results showing the percentage of contradictory matrices for order 3x3 to 9x9.

5.3.2 Contradictory Judgement Matrices and Local Rank Reversals

Local Rank Reversal (also called rank reversal problem of scale inversion [*Johnson, Beine, & Theodore, 1979*]) is another important factor for which AHP is often criticized [*Ishizaka et al, 2011*]. To illustrate the concept of Local Rank Reversals, consider an example judgement matrix M shown in Figure 5.7 (a) [*Johnson et al, 1979*]:

$\begin{pmatrix} 1 & 3 & 1/3 & 1/2 \\ 1/3 & 1 & 1/6 & 2 \\ 3 & 6 & 1 & 1 \\ 2 & 1/2 & 1 & 1 \end{pmatrix}$	$\begin{pmatrix} 0.184 \\ 0.152 \\ 0.436 \\ 0.227 \end{pmatrix}$	$\begin{pmatrix} 0.248 \\ 0.338 \\ 0.105 \\ 0.259 \end{pmatrix}$
a) Matrix M	b) P_c^T	c) P_c^T

Figure 5.7: Example for Local Rank Reversal

The principal Eigen vector, which corresponds to the priority vector of matrix M of Figure 5.7 (a), is shown in Figure 5.7 (b). The priority vector shows that the ranking of decision elements is: $\text{Item}_3 > \text{Item}_4 > \text{Item}_1 > \text{Item}_2$.

Now reverse the decision weights in the decision matrix M, i.e. $M_{ij} = 1/M_{ij}$ for all i, j = 1 to 4. With this new reversed decision matrix, the priority vector is as shown in Figure 5.7 (c). Ideally the ranking of the decision elements should also be reversed with the reverse of decision weights. But, the new priority vector shows that the ranking of decision elements is: $\text{Item}_2 > \text{Item}_4 > \text{Item}_1 > \text{Item}_3$, i.e. it is not the reverse as was predicted. Thus local rank reversal can be considered as a qualitative metric with respect to which the contradictory judgement matrices can be compared with that of non-contradictory matrices. In other words, in order to compare the quality of contradictory matrices with that of non-contradictory ones, find the probability of local rank reversal (referred to as just *rank reversal* henceforth) in both cases.

Also, to further gain insight, here two types of rank reversals are quantified:

a) Rank Reversal in Best case: Suppose R is the judgement matrix of order N (with P_c as the priority vector) and R_c is the corresponding judgement matrix with decisions inverted (with P_c' as the priority vector). If D_1 is the first decision element in P_c and D_2 is the last decision element in P_c' and D_1 and D_2 are different, then rank reversal occurs in the best case.

b) Rank Reversal in Any case: Let the N decision elements of P_c be p^1, p^2......., p^{n-1}, p^n. If there is no rank reversal then the corresponding decision elements of P_c' should be p^n, p^{n-1}......, p^2, p^1. Rather, If the order of decision elements of p_c' is different from p_c, then rank reversal occurs in any case.

Step 1	N = 4
Step 2	For i = 0 to 9: CO-BE[i] = 0, NO-CO-BE[i] = 0, CO-AN[i] = 0, NO-CO-AN[i] = 0
Step 3	Generate inconsistent matrix M of order NxN with CR < 10%. Assign it to i^{th} bin, where 0 ≤ i ≤ 9. Bin i contains matrices with i ≤ CR < i+1 and CR ≠ 0.
Step 4	Check M for being a contradictory matrix. If it is contradictory goto step 5, else go to Step 7
Step 5	Check M for Rank Reversal in best case. If Rank Reversal occurs, CO-BE[i] = CO-BE[i] + 1.
Step 6	Check M for Rank Reversal in any case. If Rank Reversal occurs, NO-CO-BE[i] = NO-CO-BE[i] + 1 and goto Step 9
Step 7	Check M for Rank Reversal in best case. If Rank Reversal occurs, CO-AN[i] = CO-AN[i] + 1.
Step 8	Check M for Rank Reversal in any case. If Rank Reversal occurs, NO-CO-AN[i] = NO-CO-AN[i] + 1.
Step 9	If there are more matrices of order NxN, goto Step 3.
Step 10	j=0
Step 11	Find percentage of Contradictory matrices for j^{th} bin in which Rank Reversal occurs in best case and any case.
Step 12	Find percentage of Non-Contradictory matrices for j^{th} bin in which Rank Reversal occurs in best case and any case.
Step 13	j = j+1. If j < 10 goto Step 11.
Step 14	N = N+1. If N < 10, goto Step 2.

Figure 5.8: Algorithm for comparing contradictory and non-contradictory matrices on the basis of Local Rank Reversals.

The algorithm in Figure 5.8 explains the simulation process. It is to be noted here that the judgement matrices of order 3x3 are always free from the anomaly of rank reversals [*Saaty, 1984*], which is why the order of matrices (i.e. value of N) starts from 4.

Simulation was done on judgement matrices of order 4x4 to 9x9 and percentage of rank reversals in each bin for each order was recorded. Bin[i] indicates the judgement matrices with $i \leq CR < i+1$ and $CR \neq 0$. Table C.8 to Table C.13 shows the recorded data.

Figure 5.9 to 5.14 shows the graphs depicting the rank reversal (in best and any case) for contradictory and non-contradictory matrices. Figure 5.9 shows the percentage of rank reversals for judgement matrices of order 4x4. It clearly shows that for Bin 6 and Bin 7 the percentage of Rank Reversals in best case are marginally higher for contradictory matrices. Otherwise, the non-contradictory matrices are more subjected to rank reversals in best case. On the other hand, for rank reversals in any case, up to Bin 4 the non-contradictory matrices are more subjected to rank reversals, and from Bin 5 to Bin 9 the percentage is higher for contradictory judgement matrices.

For order 5x5 and 6x6, the Rank reversals in best case are always higher for contradictory matrices. Rather, for rank reversal in any case, except for Bin 0 and Bin 1 (where percentage is marginally higher for non-contradictory matrices), the contradictory matrices are more subjected to Rank Reversals.

For order 7x7 and 8x8, expect in Bin 0 (where again the percentage is marginally higher for non-contradictory matrices), the contradictory matrices are more subject to Rank reversal.

For matrices of order 9x9, figure 5.14 clearly shows that the contradictory matrices are more subjected to rank reversals.

The graphs also show that for any order, as the value of CR increases the probability of rank reversal also increases. For best case, the percentage is as low as 0% for best and any case for order 4x4. Rather, it is as high as ≈ 58.1% for order 9x9 contradictory matrices.

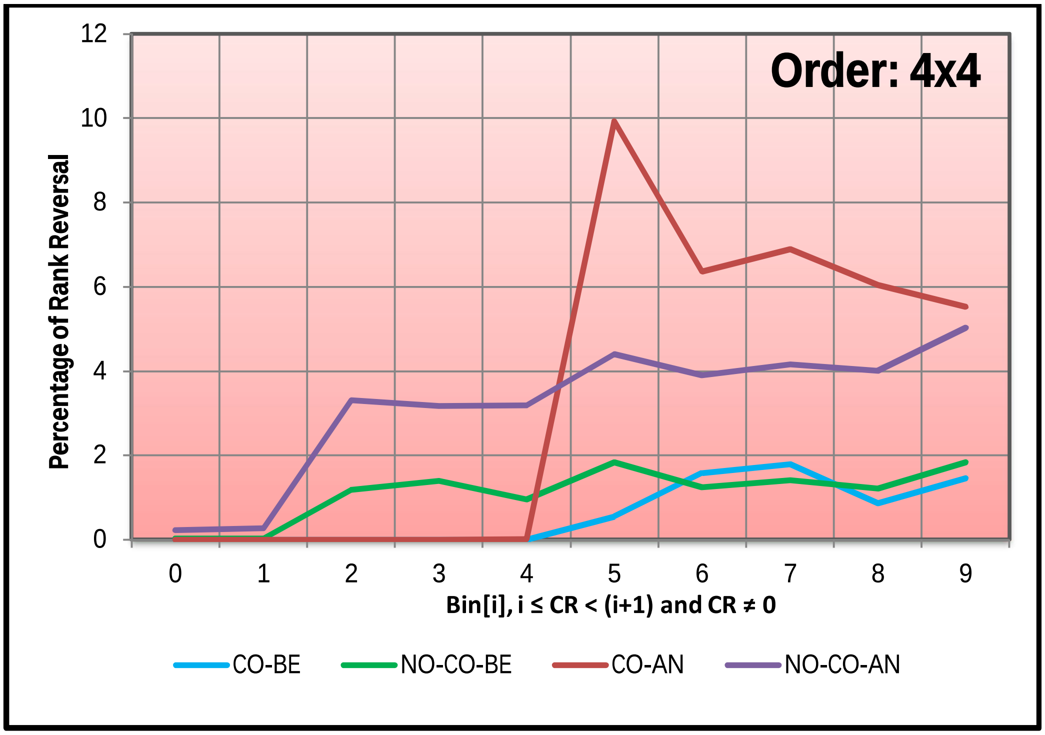


Figure 5.9: Percentage of Local Rank reversals in Judgement Matrices of Order 4x4

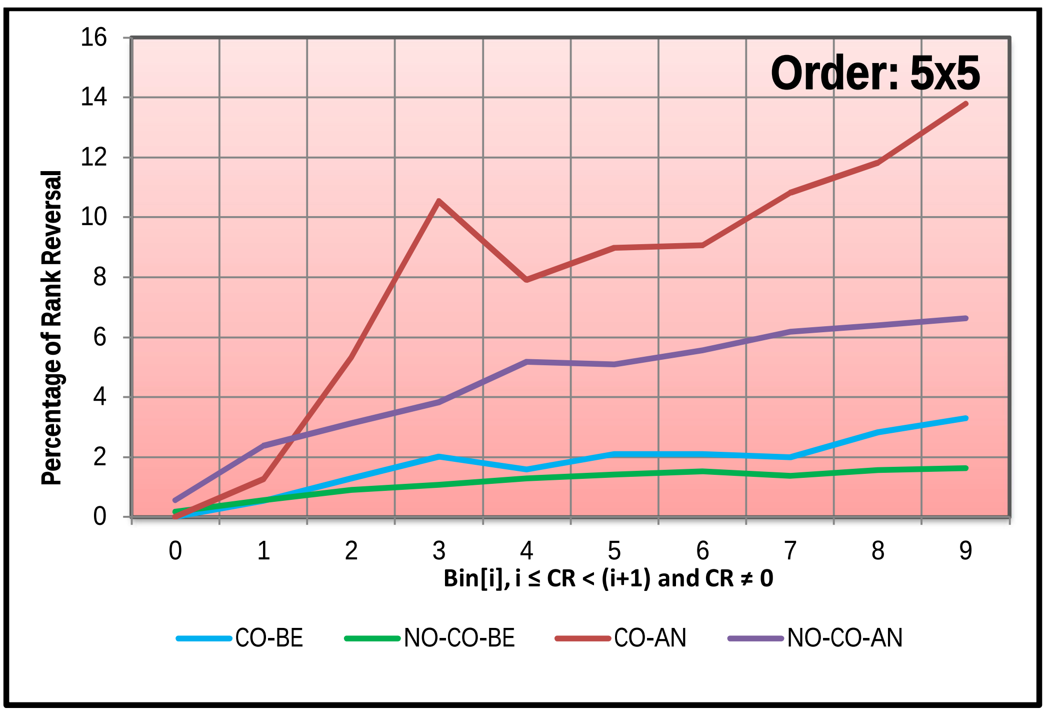


Figure 5.10: Percentage of Local Rank reversals in Judgement Matrices of Order 5x5

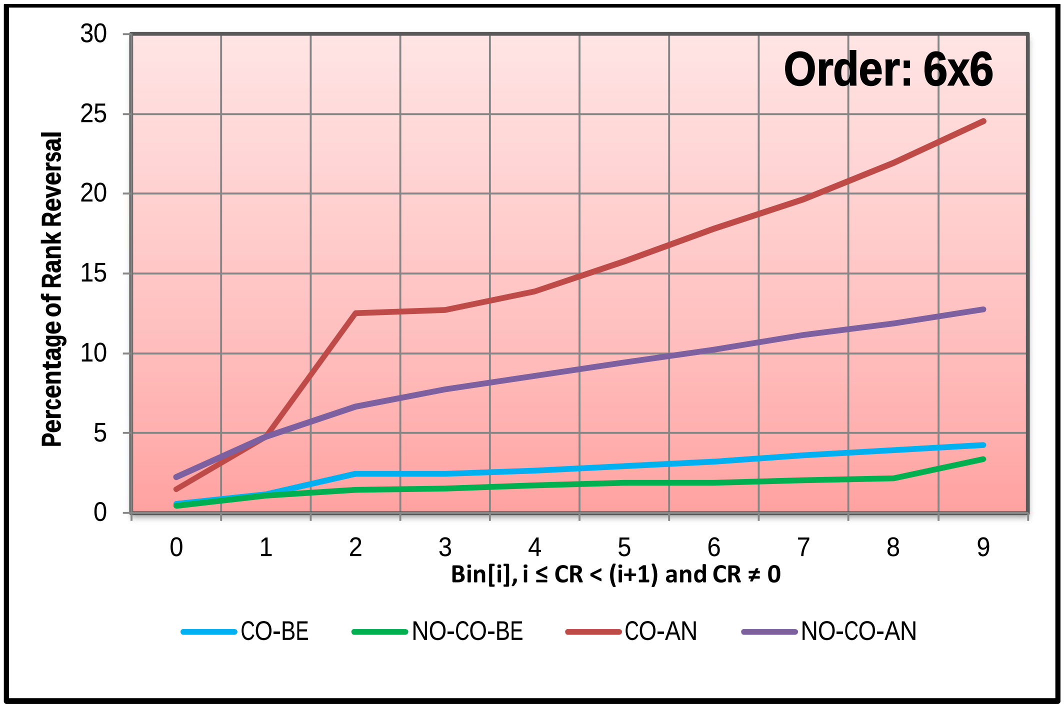


Figure 5.11: Percentage of Local Rank reversals in Judgement Matrices of Order 6x6

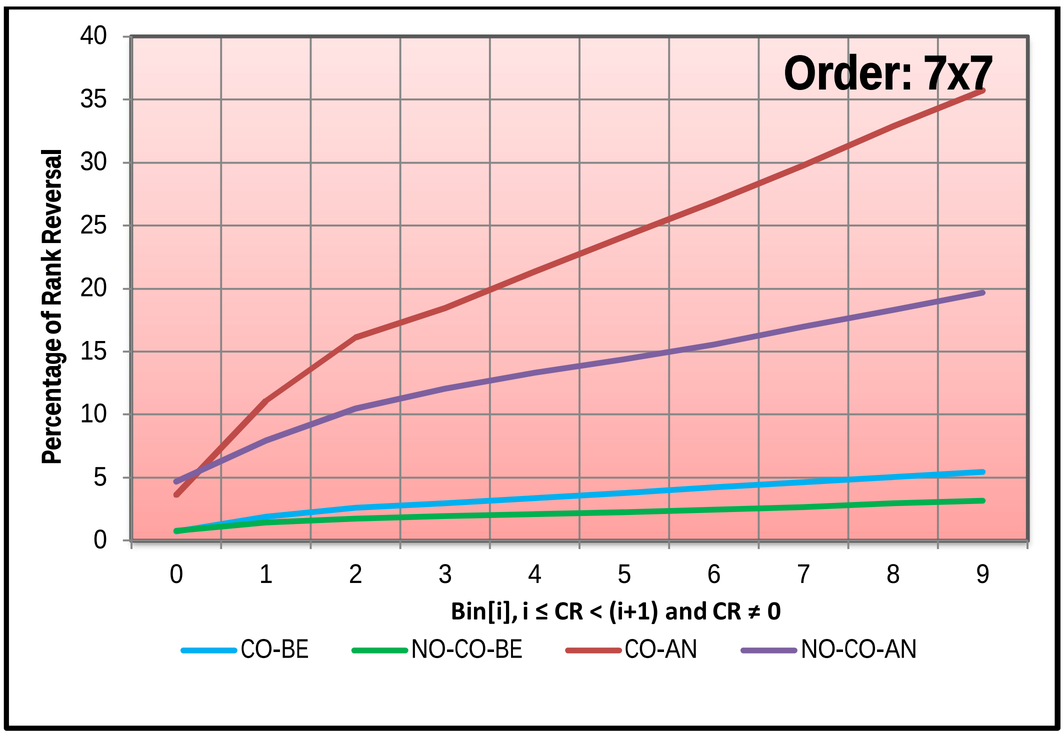


Figure 5.12: Percentage of Local Rank reversals in Judgement Matrices of Order 7x7

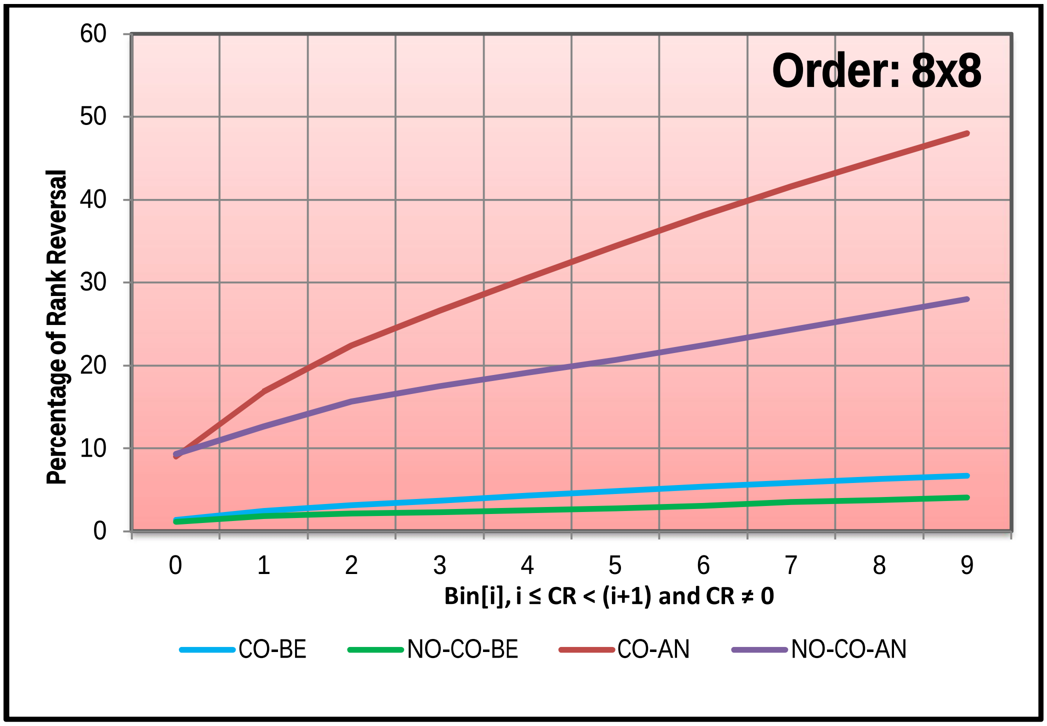


Figure 5.13: Percentage of Local Rank reversals in Judgement Matrices of Order 8x8

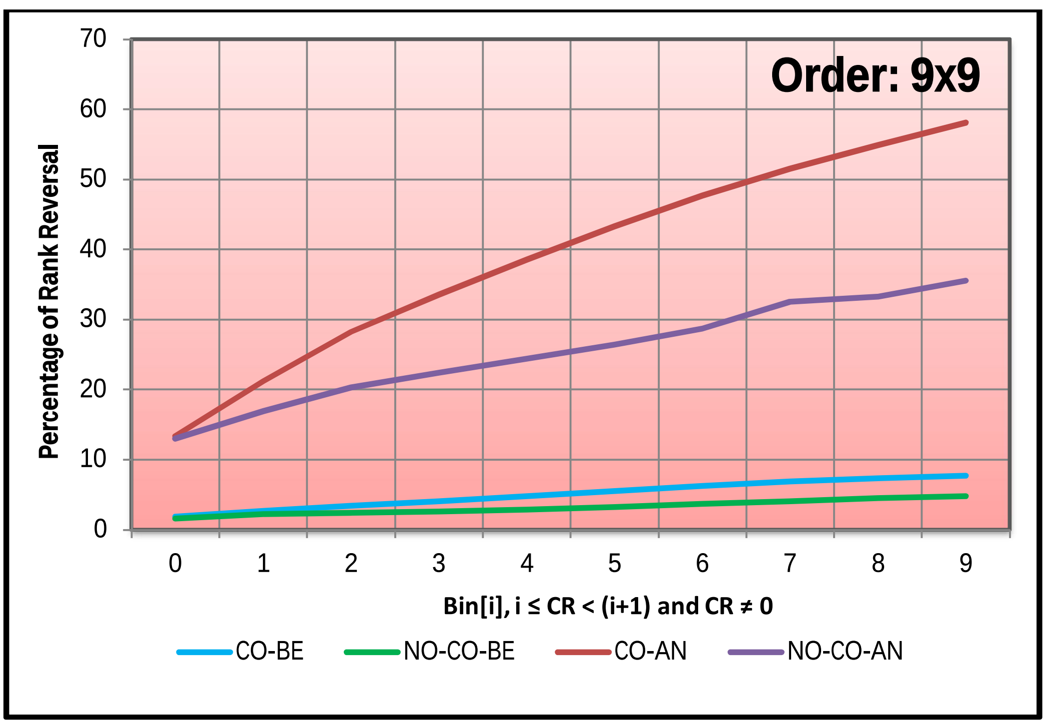


Figure 5.14: Percentage of Local Rank reversals in Judgement Matrices of Order 9x9

5.3.3 Measuring Contradictory Judgement Matrices w.r.t. Distance metric

As per AHP, Principal Eigen vector of an N x N judgement matrix represents the preference values of the decision elements. In this section the effectiveness of Principal Eigen vector for contradictory and non-contradictory matrices under the common framework of "aggregated deviation" is compared. It is based on various distance methods to find the aggregated distance between the ratio of derived preferences (p_i/p_j) and to the pair-wise comparison ratios (w_{ij}).

Here it is assumed that there exists preference values v_1, v_2....., v_n such that v_i represents the preference value of i^{th} decision element and decision maker has provided the values w_{ij} which represents the pair-wise comparison of i^{th} decision element with that of j^{th}.

The N x N judgement matrix W = [w_{ij}] is an approximation of preference vector $\mathbf{V_p}$ = [v_1, v_2... v_n]^T because w_{ij} is an approximation of v_i/v_j for all i, j = 1, 2....., n. If matrix W is an error free matrix (i.e. consistent matrix), then $w_{ij} = v_i/v_j$ for all i, j = 1, 2......, n. Also, in this case, the preference vector $\mathbf{V_p}$ is readily given by each of the columns of W, i.e. $C_j = (1/v_j)\mathbf{V_p}$ for j = 1, 2 ... , n [*Choo et al, 2004*].

But in general W is not error free and therefore there is a need to estimate the preference vector $\mathbf{V_p}$ from W. According to Saaty, the Principal Eigen vector is a good approximation of $\mathbf{V_p}$. Let P(p_1, p_2........, p_n) denote the Principal Eigen vector of matrix W, and correspondingly [p_i/p_j] be the N x N matrix for P such that [p_i/p_j] is error free for P. It follows from W that P should be close to v if W is close to [p_i/p_j].

To measure this abstract parameter "closeness", Choo et al [*Choo et al, 2004*] described the following 12 distance methods to measure the distance between two N x N matrices:

1) Least Square Method (LSM): (also called as Total Deviation [*Golany et al, 1993*]): We know that $w_{ij} \approx p_i/p_j$. This implies $(w_{ij} - p_i/p_j)^2 \approx 0$. Based on this idea, the aggregated distance or total deviation (as called by Golany et al), is given as:

$$D\ (W, [p_i/p_j]) \;=\; \sum_{i=1}^{N}\sum_{j=1}^{N}\left(w_{ij} - {}^{p_i}/_{p_j}\right)^2 \qquad \text{(Eq 5.9)}$$

2) Least Worst Square (LWS): Again based on $w_{ij} \approx p_i/p_j$, it is implied that $Max_{i\neq j}\ (w_{ij} - p_i/p_j)^2 \approx 0$. In other words, here the largest squared deviation is measured. It is given as:

$$D\ (W, [p_i/p_j]) \;=\; \max_{i\neq j}\left(w_{ij} - p_i/p_j\right)^2 \qquad \text{(Eq 5.10)}$$

3) Preference Weighted Least Square (PWLS): Here the deviations of each column are weighted by the preference values of the column referred to. It is given as:

$$D\ (W, [p_i/p_j]) \;=\; \sum_{i=1}^{N}\sum_{j=1}^{N}\left(w_{ij}p_j - p_i\right)^2 \qquad \text{(Eq 5.11)}$$

4) Preference Weighted Least Worst Square (PWLWS): Again, based on weighing the column deviations by the preference values of the column referred to, this distance function is given as:

$$D\ (W, [p_i/p_j]) \;=\; \max_{i\neq j}\left(w_{ij}\,p_j - p_i\right)^2 \qquad \text{(Eq 5.12)}$$

5) Least Absolute Error (LAE): Again the idea that $w_{ij} \approx p_i/p_j$ implies |wij - pi/pj| ≈0. Based on this concept, the sum of all the absolute deviations is given as:

$$D\ (W, [p_i/p_j]) \;=\; \sum_{i=1}^{N}\sum_{j=1}^{N}\left|w_{ij} - {}^{p_i}/_{p_j}\right| \qquad \text{(Eq 5.13)}$$

6) Least Worst Absolute Error (LWAE): It is a measure of maximum absolute error and is given as:

$$D\ (W, [p_i/p_j]) \;=\; \max_{i\neq j}\left|w_{ij} - {}^{p_i}/_{p_j}\right| \qquad \text{(Eq 5.14)}$$

7) Preference Weighted Least Absolute Error (PWLAE): Here the deviations are weighted by the preference values similar to the weighted Tchebycheff norm [*Choo et al, 2004; Steuer & Choo, 1983*]. It is given as:

$$D\,(W, [p_i/p_j]) \;=\; \sum_{i=1}^{N}\sum_{j=1}^{N}\left|w_{ij}p_j \,-\, p_i\right| \qquad \text{(Eq 5.15)}$$

8) Preference Weighted Least Worst absolute Error (PWLWAE): Based on weighing the deviations on preference values, this distance is given as:

$$D\,(W, [p_i/p_j]) \;=\; \max_{i \neq j} \left|w_{ij}p_j \,-\, p_i\right| \qquad \text{(Eq 5.16)}$$

9) Logarithmic Least Square (LLS): Based on the idea that squaring the difference of the log of the ratios gives greater emphasis to transformed deviations, this distance function is given as:

$$D\,(W, [p_i/p_j]) \;=\; \sum_{i=1}^{N}\sum_{j=1}^{N}\left[\ln(w_{ij}) - \left(\ln(p_i) - \ln(p_j)\right)\right]^2 \qquad \text{(Eq 5.17)}$$

10) Logarithmic Least Worst Square (LLWS): Here the largest deviation selection places total emphasis on the worst logarithmic deviation. The distance function is given as:

$$D\,(W, [p_i/p_j]) \;=\; \max_{i \neq j} \left[\ln(w_{ij}) - \left(\ln(p_i) - \ln(p_j)\right)\right]^2 \qquad \text{(Eq 5.18)}$$

11) Logarithmic Least Absolute Error (LLAE): It is given as:

$$D\,(W, [p_i/p_j]) \;=\; \sum_{i=1}^{N}\sum_{j=1}^{N}\left|\left[\ln(w_{ij}) - \left(\ln(p_i) - \ln(p_j)\right)\right]\right| \qquad \text{(Eq 5.19)}$$

12) Logarithmic Least Worst Absolute Error (LLWAE): It is given as:

$$D\,(W, [p_i/p_j]) \;=\; \max_{i \neq j} \left|\left[\ln(w_{ij}) - \left(\ln(p_i) - \ln(p_j)\right)\right]\right| \qquad \text{(Eq 5.20)}$$

Apart from above defined 12 methods, Golany et al (1993) defined another metric, called Minimum Violation. A violation is defined to occur if in ratio-scale matrix object j is preferred to object i but i get the larger weight in the final priority vector. This metric sum all the violations associated with the weight vector P. As defined by B. Golany [*Golany, et al, 1993*], the minimum violation is defined as:

$$\sum_{i=1}^{N}\sum_{j=1}^{N} I\ (i,j) \qquad \text{(Eq 5.21)}$$

Where,

$$I\ (i,j)) = \begin{cases} 1 \text{ if } p_i > p_j \text{ and } w_{ij} < 1 \\ 1/2 \text{ if } (p_i = p_j \text{ and } w_{ij} \neq 1) \text{ OR } (p_i \neq p_j \text{ and } w_{ij} = 1) \\ 0 \quad \text{otherwise.} \end{cases}$$

The algorithm in Figure 5.15 explains the simulation process. Simulation was done on judgement matrices of order 3x3 to 9x9 and average distance was recorded between the matrices for all the above defined twelve distance functions. Also, same simulation process was followed for the "Minimum Violation" metric. Table C.14 to Table C.20 shows the recorded data.

Figure 5.16 to Figure 5.28 shows the comparison of contradictory and non-contradictory matrices on the basis of each of the above defined quantitative criteria. These clearly show that:

- For PWLS, PWLWS, PWLAE, PWLWAE, LLS, LLWS, LLAE, and LLWAE distance methods, the distance between matrices w_{ij} and $[p_i/p_j]$ is always more for contradictory matrices than for non-contradictory matrices.
- The same trend is observed for the metric called Minimum Violation. More specifically, for contradictory matrices the Minimum Violation is always greater than for non-contradictory matrices.
- For distance method LAE the recorded distance is almost overlapping.
- On the contrary, for LSM, LWS, and LWAE the observation is reverse, i.e. the distance between w_{ij} and $[p_i/p_j]$ is always more for non-contradictory matrices than for contradictory matrices.

Step 1	N = 3
Step 2	for i = 0 to 9 Co-Mat[i][13] = 0, Non-Co-Mat[i][13] = 0.
Step 3	Generate inconsistent matrix M of order NxN with CR < 10%. Assign it to i^{th} bin, where 0 ≤ i ≤ 9. Bin i contains matrices with i ≤ CR < i+1 and i ≠ 0.
Step 4	Check M for being a contradictory matrix. If it is contradictory goto step 5, else go to Step 8
Step 5	Repeat Step 6 for j = 1 to 13
Step 6	Co-Mat[i][j] = distance between M and the corresponding matrix w.r.t. Eigen vector P for j^{th} quantitative metric (i.e. distance function).
Step 7	goto step 10
Step 8	Repeat Step 6 for j = 1 to 13
Step 9	Non-Co-Mat[i][j] = distance between M and the corresponding matrix w.r.t. Eigen vector P for j^{th} quantitative metric (i.e. distance function).
Step 10	If there are more matrices of order NxN, goto Step 3.
Step 11	Repeat Step 12 for j = 0 to 13.
Step 12	Find average distance for j^{th} quantitative metric.
Step 13	N = N+1. If N < 10, goto Step 2.

Figure 5.15: Algorithm for comparing contradictory and non-contradictory matrices with respect to the quantitative metric.

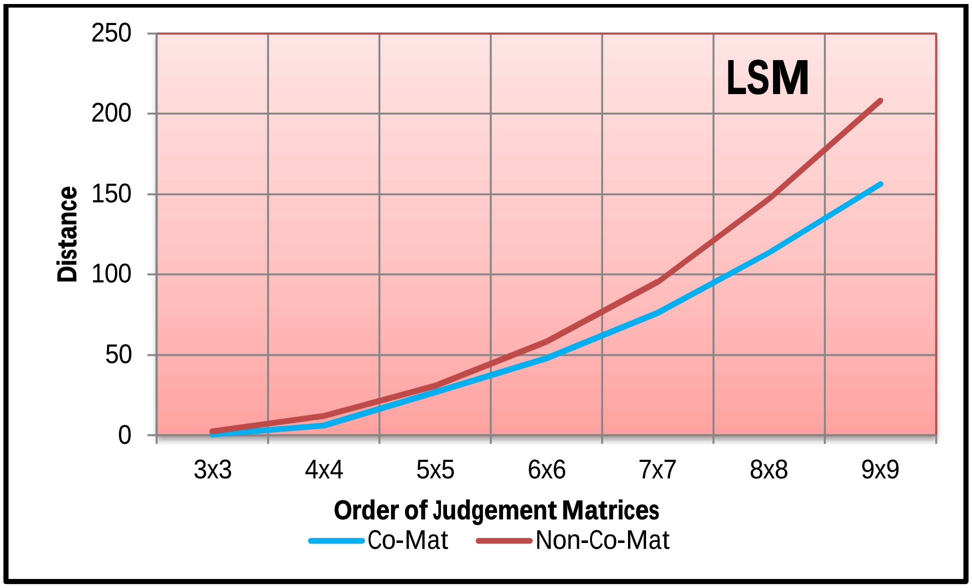


Figure 5.16: Comparing contradictory and non-contradictory matrices w.r.t. distance function LSM

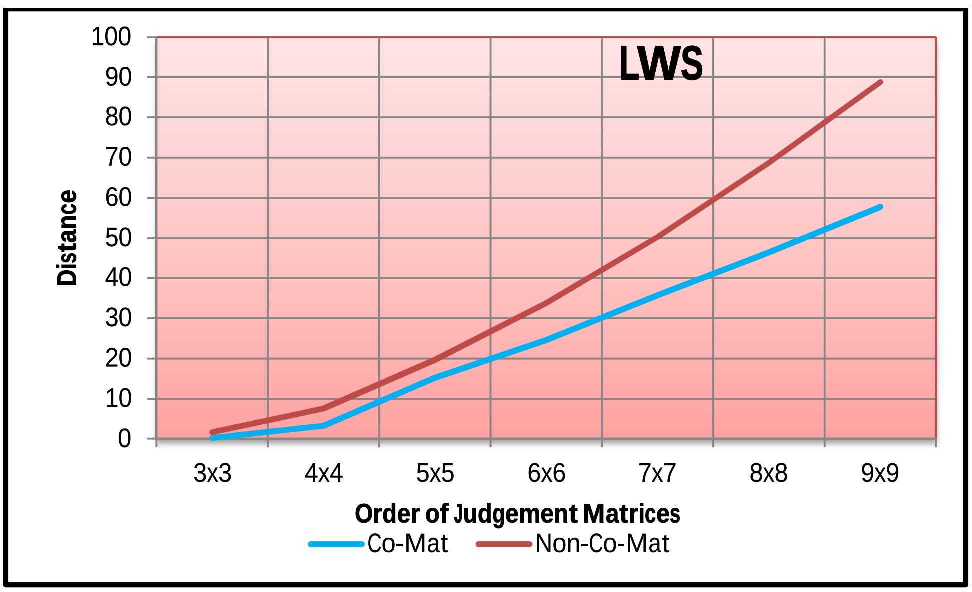


Figure 5.17: Comparing contradictory and non-contradictory matrices w.r.t. distance function LWS

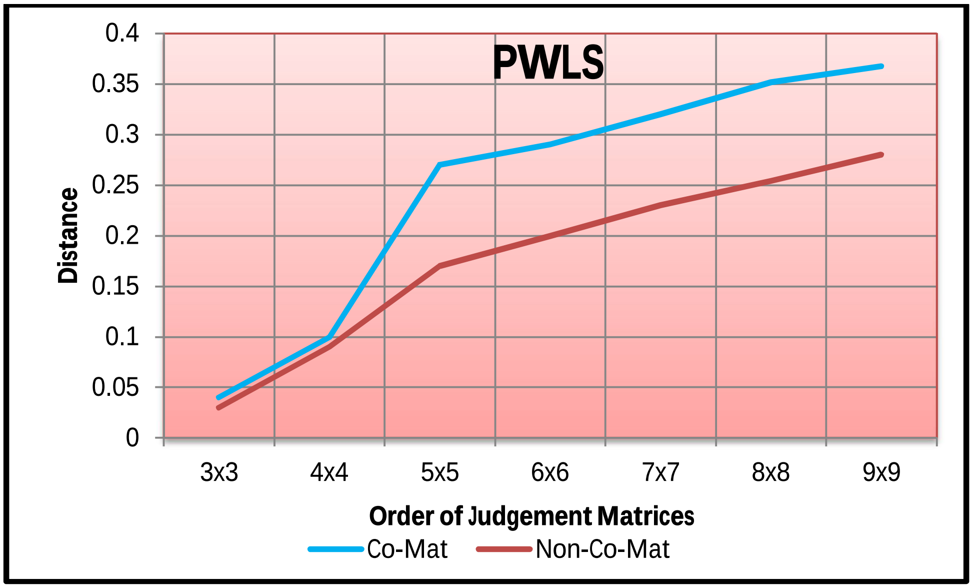


Figure 5.18: Comparing contradictory and non-contradictory matrices w.r.t. distance function PWLS

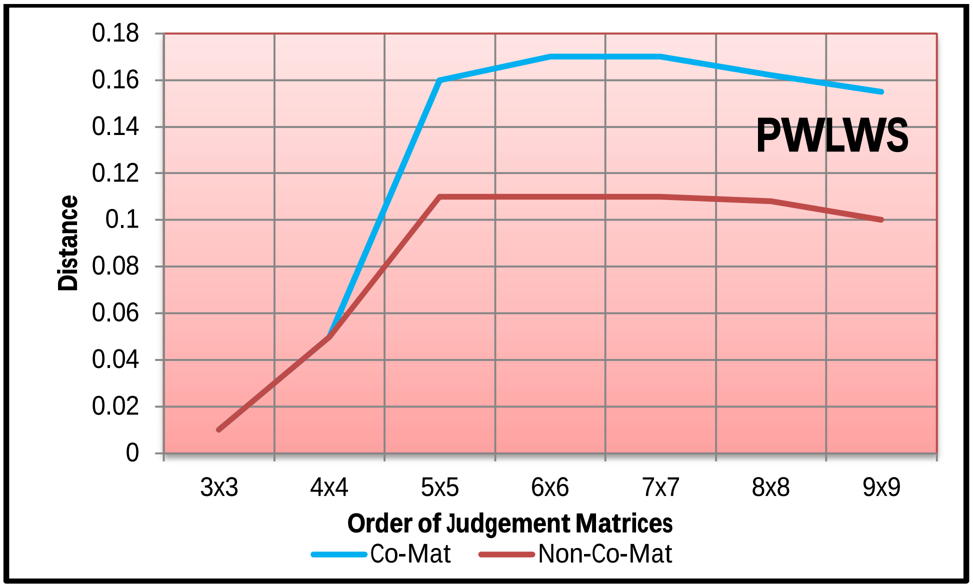


Figure 5.19: Comparing contradictory and non-contradictory matrices w.r.t. distance function PWLWS

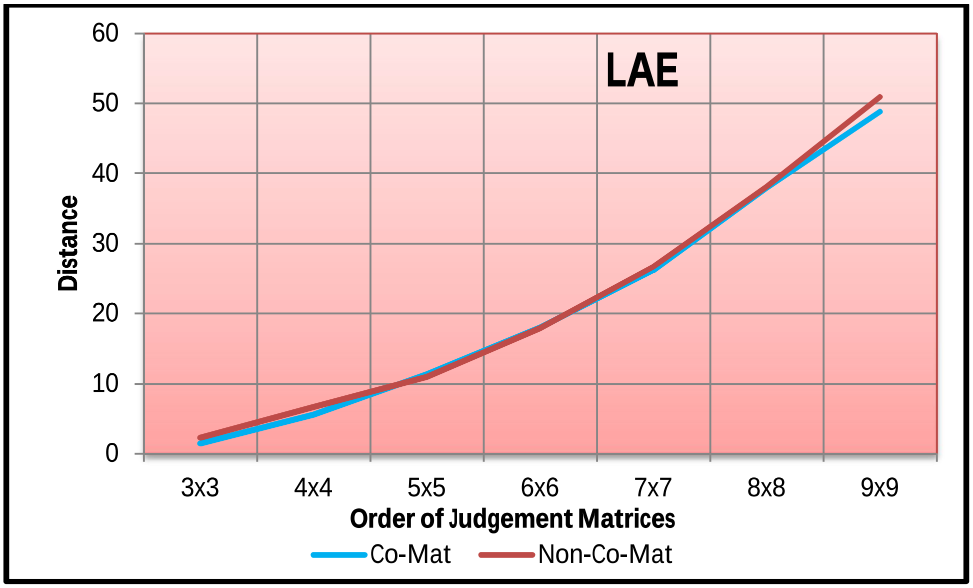


Figure 5.20: Comparing contradictory and non-contradictory matrices w.r.t. distance function LAE

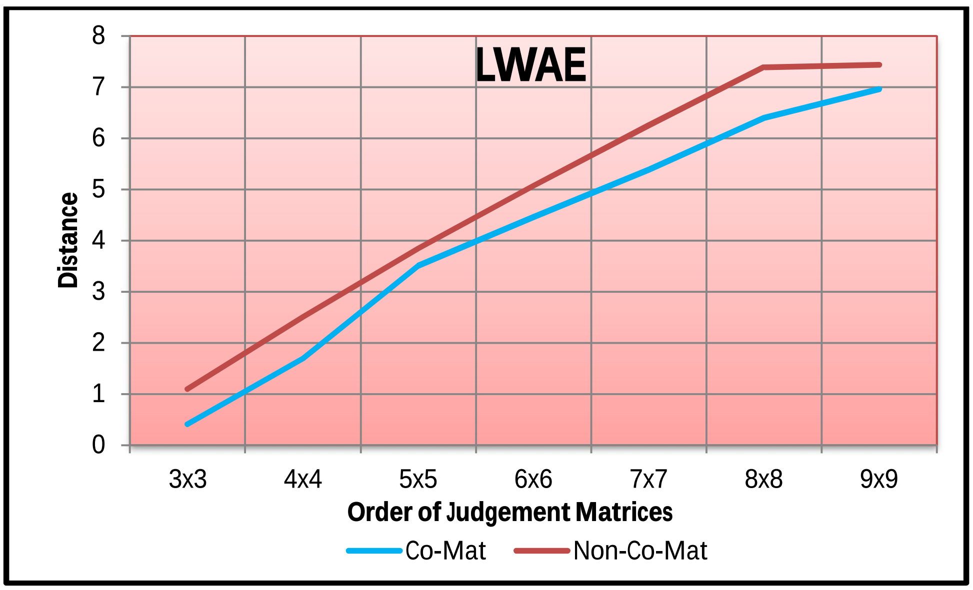


Figure 5.21: Comparing contradictory and non-contradictory matrices w.r.t. distance function LWAE

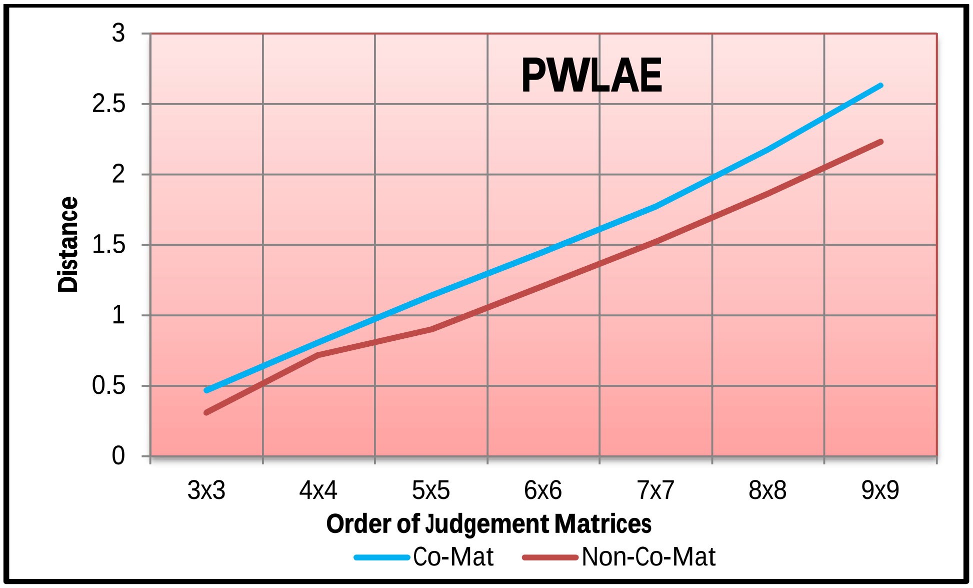


Figure 5.22: Comparing contradictory and non-contradictory matrices w.r.t. distance function PWLAE

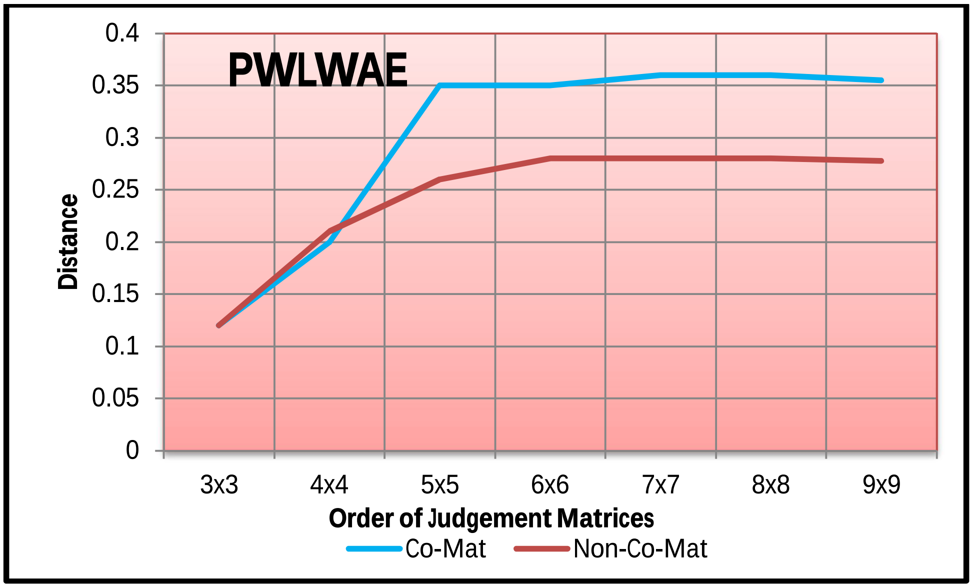


Figure 5.23: Comparing contradictory and non-contradictory matrices w.r.t. distance function PWLWAE

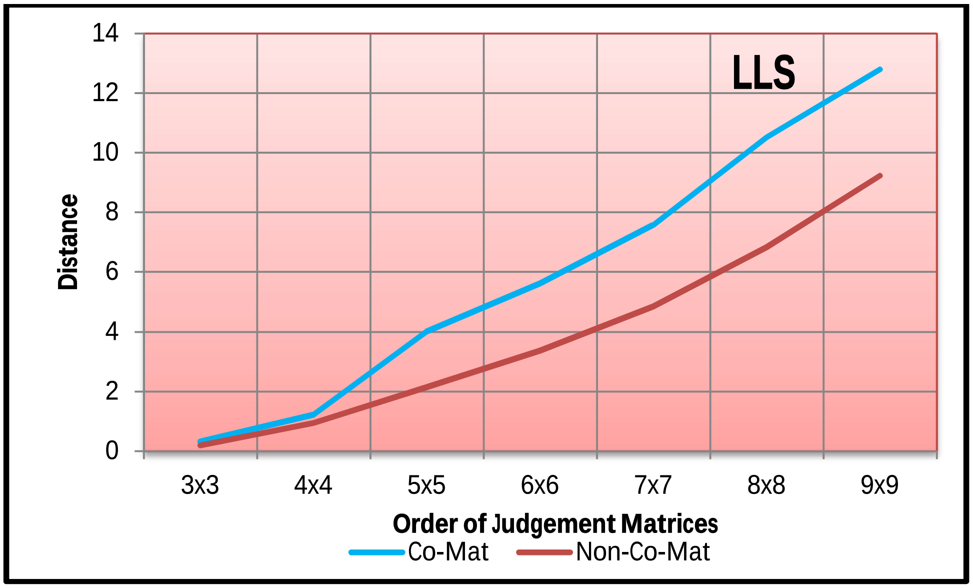


Figure 5.24: Comparing contradictory and non-contradictory matrices w.r.t. distance function LLS

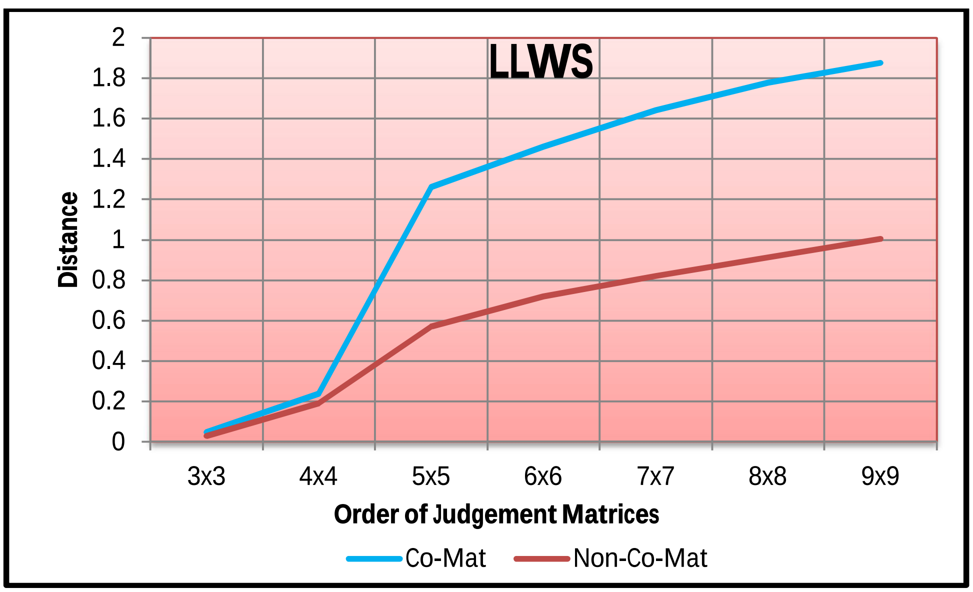


Figure 5.25: Comparing contradictory and non-contradictory matrices w.r.t. distance function LLWS

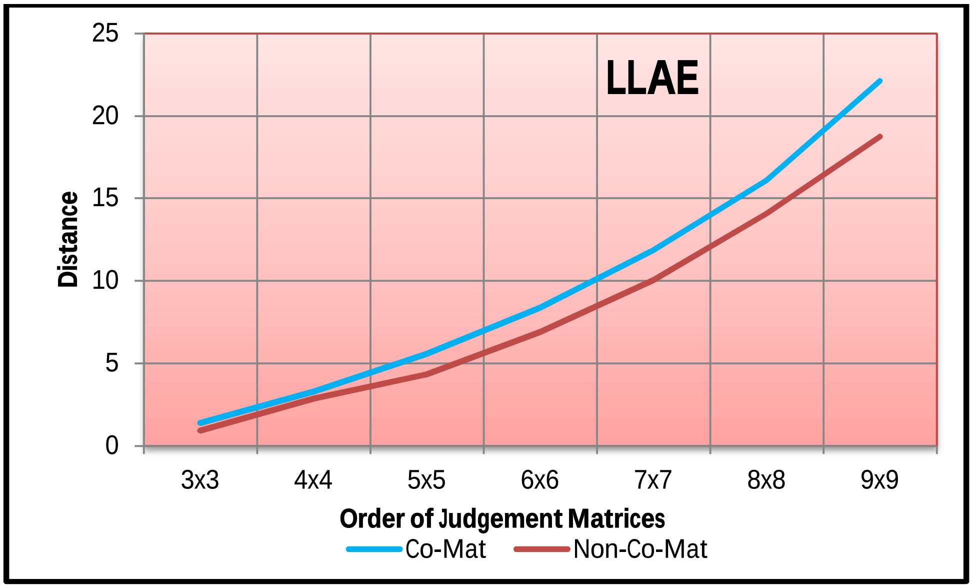


Figure 5.26: Comparing contradictory and non-contradictory matrices w.r.t. distance function LLAE

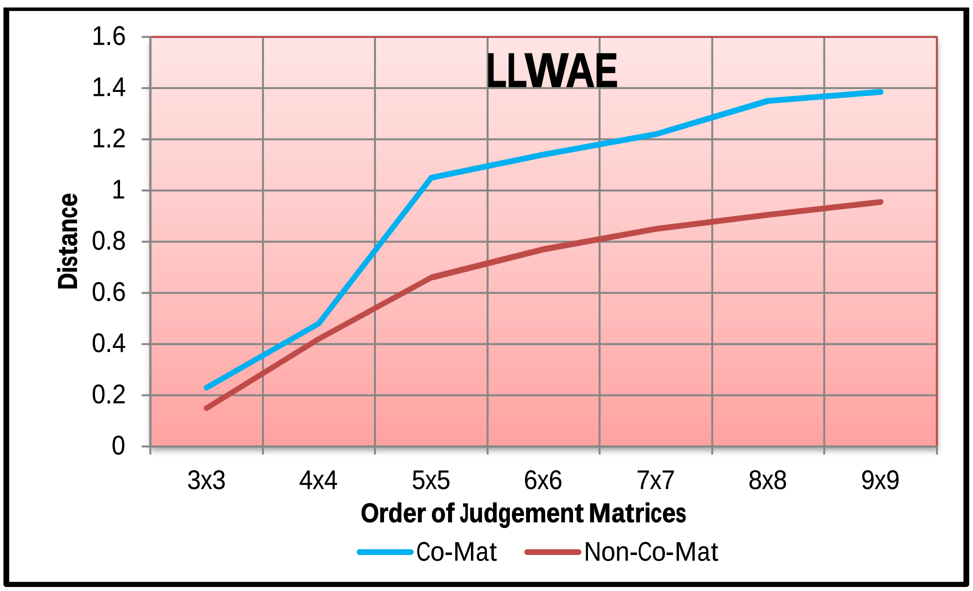


Figure 5.27: Comparing contradictory and non-contradictory matrices w.r.t. distance function LLWAE

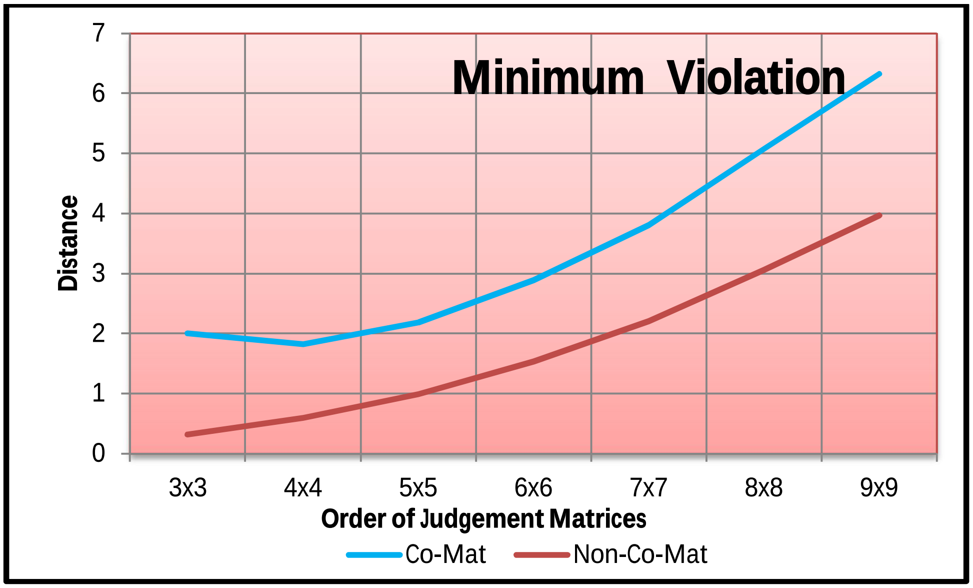


Figure 5.28: Comparing contradictory and non-contradictory matrices w.r.t. Minimum Violation

5.4 Feedback based pair-wise comparisons

The results of Section 5.3 conclude that contradictory judgement matrices in AHP should be avoided, if not *eliminated*. Elimination will require re-engineering of the notion of "Consistency Index" or "Consistency Ratio". Instead, the rest of this section explains a novel feedback based technique for pair-wise comparisons. Using it as soon as the decision maker makes an entry in the pair-wise comparison matrix and if this newly added entry "contradicts" with the previous judgements, a feedback is generated. This facilitates the decision maker to correct its contradictory decision as soon as it is entered in the pair wise comparison matrix.

Let there be N decision elements, d_1, d_2....d_N and these are to be compared using pair-wise comparison matrix. Eq 5.9 shows such pair-wise comparison matrix. There are two ways in which the decision maker can enter decision values in the matrix:

a) Diagonal-wise entries
b) Row-wise entries

5.4.1 Diagonal-wise entries

Here the decision maker starts comparing the elements in the first diagonal above the main diagonal, then second diagonal above the main diagonal, and continuing up to $(N-1)^{th}$ diagonal above the main diagonal. In other words, as in Eq 5.9, first the decision maker makes the entries corresponding to w_{12}, w_{23}, w_{34}....., w_{N-1N}. Then the entries corresponding to w_{13}, w_{24}...., $w_{N-2\ N}$ are made. It continues up to the last diagonal, i.e. w_{1N}.

5.4.1.1 Diagonal-wise Contradictory entry

Here the notion of *contradictory matrix* is extended to *diagonal-wise contradictory entry*. This is based on assumption that the pair-wise entries in the judgement matrix are strictly done diagonal-wise in its upper triangle, starting from first diagonal above the main diagonal.

Let

N be the order of pair-wise judgement matrix. It has N rows (numbered from 1 to N) and N columns (numbered from 1 to N).

w_{ac} be the entry in a^{th} row and c^{th} column, where c > a.

D_i be the i^{th} diagonal above the main diagonal with respect to column number i+1, where $1 \le i \le (N-1)$.

The following two statements (St1 and St2) are true:

St1: *Entry w_{ac} lies on diagonal $D_{(c-a)}$*

Since pair-wise entries are done diagonal-wise, while entering w_{ac} clearly the decision maker has moved (c-a-1) columns to the right of w_{aa}. Therefore, St1 is true.

St2: While entering w_{ac}, pair-wise decisions in (c-a-1) diagonals are entered.

Since pair-wise decisions are entered strictly diagonal-wise and St1 is true, St2 is also true.

Based on St1 and St2, entry w_{ac} is a *diagonal-wise contradictory entry* if the following two conditions are true:

a) (c-a) > 1, i.e. w_{ac} lies on D_2 and above. Eq 5.22

b) ∀ b, where a < b < c, any of the following conditions (Eq 5.23 to Eq 5.28) is true:

$w_{ac} < 1$ AND $w_{ab} > 1$ AND $w_{bc} > 1$ Eq 5.23

$w_{ac} > 1$ AND $w_{ab} < 1$ AND $w_{bc} < 1$ Eq 5.24

$w_{ac} > 1$ AND $w_{ab} = 1$ AND $w_{bc} < 1$ Eq 5.25

$w_{ac} < 1$ AND $w_{ab} = 1$ AND $w_{bc} > 1$ Eq 5.26

$w_{ac} = 1$ AND $w_{ab} = 1$ AND $w_{bc} < 1$ Eq 5.27

$w_{ac} = 1$ AND $w_{ab} = 1$ AND $w_{bc} > 1$ Eq 5.28

5.4.1.2 Diagonal-wise feedback algorithm

The algorithm, as shown in Fig 5.29, takes the entries diagonal-wise and based on Eq 5.22 to Eq 5.28 decides whether the entry is contradictory or not. If it is contradictory, an indication is given to the decision maker. Since it is argued to avoid such contradictory entry (which makes the judgement matrix as contradictory judgement matrix), it is up to the decision maker whether the decision is to be corrected or not. The time complexity of the algorithm is O(1), i.e. constant with respect to limiting the number of decision elements to be compared to 9.

Assumptions:
W is the pair-wise comparison matrix of order NxN
Algorithm:

```
for i = 1 to (N-1)
  for j = 1 to (N-i)
    Decision maker makes an entry W[j][j+i].
    for b = (j+1) to (j+i-1)
      a=j;
      c = j+i;
      if [(W[a][c] < 1 AND W[a][b] > 1 AND W[b][c] >1) OR
         (W[a][c] > 1 AND W[a][b] < 1 AND W[b][c] <1) OR
         (W[a][c] > 1 AND W[a][b] = 1 AND W[b][c] <1) OR
         (W[a][c] < 1 AND W[a][b] = 1 AND W[b][c] >1) OR
         (W[a][c] = 1 AND W[a][b] = 1 AND W[b][c] <1) OR
         (W[a][c] = 1 AND W[a][b] = 1 AND W[b][c] >1) ]
      then PRINT "CONTRADICTORY ENTRY"
```

Figure 5.29: Algorithm to check diagonal-wise contradictory entry.

5.4.2 Row-wise entries

Here the decision maker starts comparing the elements in the first row of the upper triangle, then second row of the upper triangle, and continuing up to (N-1)th row of the upper triangle. In other words, as in Eq 5.9, first the decision maker makes the entries corresponding to w_{12}, w_{13}, w_{14}....., w_{1N}. Then the entries corresponding to w_{23}, w_{24}...., w_{2N} are made. It continues up to the last row, i.e. $w_{N-1\ N}$.

5.4.2.1 Row-wise Contradictory entry

Here the notion of *contradictory matrix* is extended to row-*wise contradictory entry*. It is assumed that the pair-wise entries in the judgement matrix are strictly done row-wise in the upper triangle of the judgement matrix, starting from the first row. Let

N be the order of pair-wise judgement matrix. It has N rows (numbered from 1 to N) and N columns (numbered from 1 to N).

w_{ac} be the entry in a^{th} row and c^{th} column, where c > a.

R_i be the i^{th} row, where $1 \le i \le (N-1)$.

The following statement (St1) is true:

St1: While entering w_{ac}, pair-wise decisions in (a-1) rows are already entered.

Since pair-wise decisions are entered strictly row-wise and by definition of w_{ac}, St1 is also true.

Based on St1, entry w_{ac} is a *row-wise contradictory entry* if the following two conditions are true:

a) a > 1, i.e. w_{ac} lies on R_2 and above. Eq 5.29

b) $\forall$ b, where $1 \le b < a$ any of the following conditions (Eq 5.30 to Eq 5.35) is true:

$w_{ac} < 1$ AND $w_{ab} > 1$ AND $w_{bc} > 1$ Eq 5.30

$w_{ac} > 1$ AND $w_{ab} < 1$ AND $w_{bc} < 1$ Eq 5.31

$w_{ac} > 1$ AND $w_{ab} = 1$ AND $w_{bc} < 1$ Eq 5.32

$w_{ac} < 1$ AND $w_{ab} = 1$ AND $w_{bc} > 1$ Eq 5.33

$w_{ac} = 1$ AND $w_{ab} = 1$ AND $w_{bc} < 1$ Eq 5.34

$w_{ac} = 1$ AND $w_{ab} = 1$ AND $w_{bc} > 1$ Eq 5.35

5.4.2.2 Row-wise feedback algorithm

The algorithm, as shown in Fig 5.30, takes the entries row-wise and based on Eq 5.29 to Eq 5.35 decides whether the entry is contradictory or not. If it is contradictory, an indication is given to the decision maker; and it is up to him whether the decision is to be corrected. Again, the time complexity of the algorithm is O(1), i.e. constant with respect to limiting the number of decision elements to be compared to 9.

```
Assumptions:
W is the pair-wise comparison matrix of order NxN
Algorithm:
        for i = 1 to (N-1)
          for j = (i+1) to N
            Decision maker makes an entry W[i][j].
            for b = 1 to (i-1)
              a=i;
              c = j;
              if [(W[a][c] < 1 AND W[a][b] > 1 AND W[b][c] >1) OR
                 (W[a][c] > 1 AND W[a][b] < 1 AND W[b][c] <1) OR
                 (W[a][c] > 1 AND W[a][b] = 1 AND W[b][c] <1) OR
                 (W[a][c] < 1 AND W[a][b] = 1 AND W[b][c] >1) OR
                 (W[a][c] = 1 AND W[a][b] = 1 AND W[b][c] <1) OR
                 (W[a][c] = 1 AND W[a][b] = 1 AND W[b][c] >1) ]
              then PRINT "CONTRADICTORY ENTRY"
```

Figure 5.30: Algorithm to check row-wise Contradictory entry.

5.5 Observations and their interpretations

Kwiesielewicz et al (2004) shown and proved that AHP allows the consideration of contradictory judgement matrices. They also proved that no ranking of decision elements exists for contradictory matrices which satisfy all the judgements. With respect to it, this chapter had shown the measuring of the magnitude of this problem and comparing contradictory matrices with non-contradictory matrices on RR-SI and 13 different quantitative metrics. Following are the observations:

- As the order of judgement matrices increases from 3x3 to 9x9, the percentage of contradictory matrices also increases from ≈ 0.7% to ≈ 72.8% respectively.
- Based on the comparison with qualitative metric and 13 different quantitative metrics, results again clearly show that for most of these, contradictory judgement matrices are not better than non-contradictory matrices.

- All this gives a clear implication that contradictory judgement matrices should be *avoided* (if not *eliminated,* because *elimination* requires re-engineering the AHP technique completely). For avoiding, a feedback should be given to the decision maker on each *contradictory entry*, rather than on complete *contradictory matrix*. For the same, two algorithms are proposed using which the decision maker can enter the values in the pair-wise comparison matrix and avoid contradictory judgement matrices.
- This also suggests that the solution designers should avoid asking the decision maker to compare many decision elements at a time. Rather limit the number of decision elements at a time and increase such corresponding pair-wise comparisons.

CHAPTER 6

Modeling User Preferences for Vertical Handover

6.1 Introduction

User satisfaction is one of the ultimate aims of vertical handover process. Different users can have different preferences on the same parameter. Limiting the scope to infrastructure WLAN's, as a hypothesis, this chapter proposes a model to capture user preferences. The purpose of proposing this hypothesis is to show the usefulness of beacon stuffing and feedback based algorithms proposed in Section 5.4 to avoid contradictory judgement matrices.

6.2 Problem Formulation for Network Selection using AHP

Many research articles [*Kassar et al, 2008a; Song et al,2005a; Wang et al, 2012; Zhang et al, 2010; Song et al, 2005; Alkhawlani et al, 2008;Taheri et al, 2011*] have modeled the problem of vertical handover decision using Analytic Hierarchy Process (AHP). Figure-6.1 shows a generalized AHP tree structure for the purpose. It shows N criteria and M alternatives. Some of the criteria can have sub criteria also. The value of N depends upon the model and the value of M depends upon the number of networks available at a particular instance of time. Using AHP, N criteria are assigned weights W_1 to W_N, such that Eq 6.1 is true.

$$\sum_{i=1}^{N} W_i = 1 \qquad \text{(Eq 6.1)}$$

Similarly, for each criteria K, M alternatives are assigned weights A_{K1} to A_{KM}, such that Eq 6.2 is true.

$$\sum_{j=1}^{M} A_{Kj} = 1, where\ 1 \leq K \leq N \qquad \text{(Eq 6.2)}$$

These weights actually correspond to the ranking of criteria, and ranking of alternatives for each criterion.

The problem is that in Figure-6.1 if one of the criteria is user preferences and based on the results derived in Chapter-5, how M alternative networks can be assigned weights such that Eq 6.2 is true.

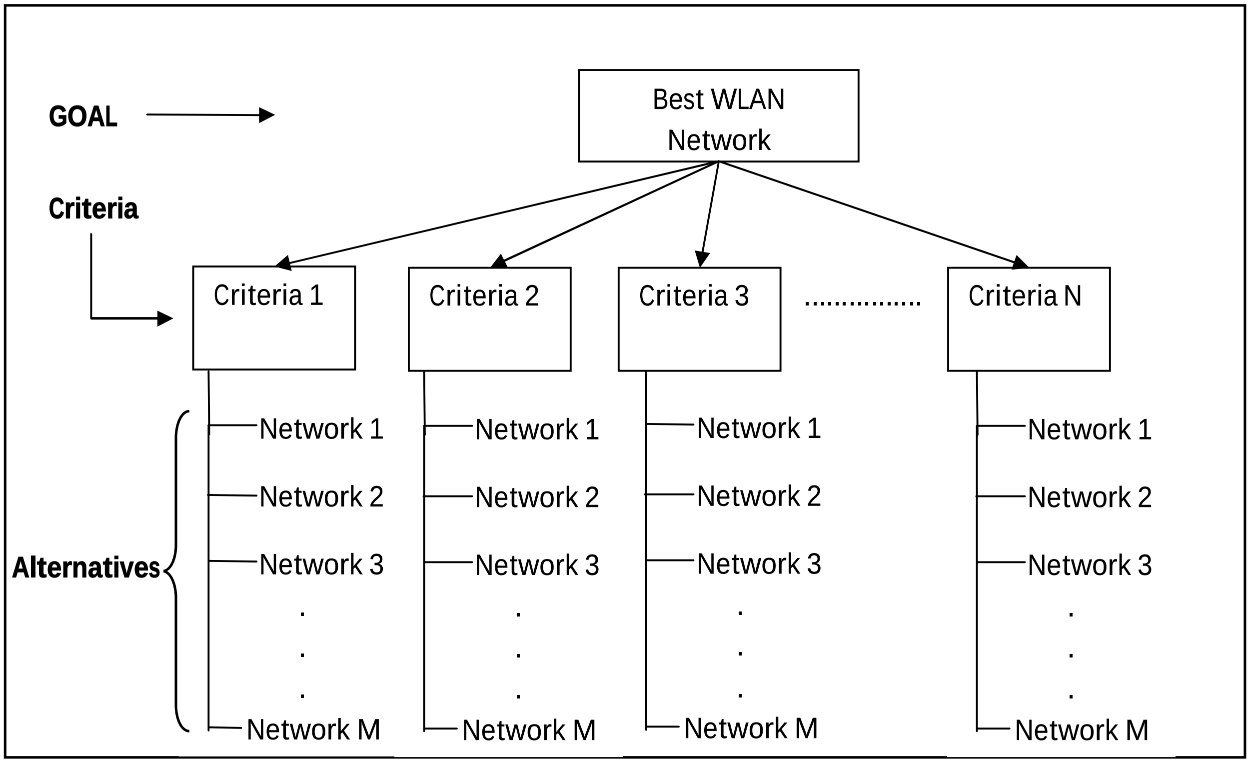


Figure-6.1: Generalized AHP model for Vertical Handover decision.

6.3 Scope of Work

The scope of the model proposed is limited to the following:

a) The proposed model is based on the conclusions drawn from Chapter-5 about avoiding contradictory judgement matrices, shows the necessity and usefulness of beacon stuffing and algorithms proposed in Section 5.4.
b) The parameters are mainly derived from the amendments proposed by 802.11u in the beacon frame. IEEE 802.11u specifies clearly in its section 1.2 under the heading "Purpose" [*IEEE 802.11u, 2011*]:

"*Defines functions and procedures aiding network discovery and selection by STA's, information transfer from external networks using QoS mapping, and a general mechanism for the provision of emergency services*"

Clearly, one of the goals of this amendment is aiding network discovery and selection. Now since beacon frame is an advertising frame for 802.11 network, 802.11u allows the insertion of few fields (as discussed in Section 3.4.2) in the beacon frame for aiding network discovery. Assuming that these newly added fields are necessary for aiding network discovery, the parameters for user preferences are mainly selected from this set of newly added fields; the ones which directly affect the end user.

c) Using the proposed model, available networks can be ranked based on only user preferences. Of course, it can be used *as such* with any vertical handover decision model which uses AHP as a decision technique.

6.4 UMTS QoS Classes

As already explained in Section 3.7, ETSI TS 123 107 [*3GPP TS 23.107, 2012*] defines four different traffic service classes: Conversational Class, Streaming Class, Interactive Class, and Background Class. Since users can have different preferences with respect to each of the class, the model facilitates the user to specify the preferences for each of them individually.

6.5 Parameters for User Preferences

ETSI TS 123 107 [*3GPP TS 23.107, 2012*] specifies that end users only care about the issues that are visible to them. Based on this, it specifies few conclusions about the involvement of the user in specifying the QoS requirements. From the end users point of view, these are [*3GPP TS 23.107, 2012*]:

- The number of user defined attributes has to be small.
- What only matters is the QoS perceived by end user.
- QoS definitions have to be future proof.
- Selected QoS attributes should be able to support all applications that are used.

- From the application requirements, deriving/defining the QoS attributes should be simple.

Considering the above five points, the following five criteria on which the user can give the preferences have been selected. These are directly related to QoS perceived by end user. Out of five criteria, three are directly derived from the amendment proposed in the beacon frame by 802.11u. So until there is no further changes done to 802.11u by IEEE these can be considered as future proof. Also deriving these from the beacon frame is simple and is applicable to all applications that are used. The other two criteria, i.e. cost and security, are the ones which affect the users directly and will continue to do so in future. The user can specify his/her choice by comparing, and thus rating, these criteria among each other.

1) Access Network Type: As stated in section 3.4.2, this feature is advertised in the beacon frame of 802.11u and specifies the type of access network. From the end users point of view, the type of access network actually correlates to the features associated with it (e.g. cost, security etc) and end user can give the preferences for selecting the network based on its perceived features. According to 802.11u, there can be eight different types of network which can be advertised, but here only the following four types of access networks are considered.

a) Private Network (PN): Examples of this type of access networks are home networks, or enterprise networks. These may employ user accounts.
b) Private Network with Guest Access (PNGA): These types of access network are the same as that of Private Networks, but guest accounts are available.
c) Chargeable Public Network (CPN): Example of this type of access network is the hotspot in a coffee shop or a hotel room, where the network is accessible to anyone but will be charged.
d) Free Public Network (FPN): Examples of this type of access network is hotspot at a public place such as airport etc where the network is accessible to anyone and there is no charge for the usage of the network.

The other four types of network which can be advertised are personal device network (which is network of personal devices like laptop attached to a printer), Emergency services only network (which is a network dedicated and limited to providing emergency services only), test or experimental network (which is a network used for test or experimental set up only), and wildcard network. Clearly these four types of networks are for dedicated purposes and are generally not to be used as the networks for offloading data services. In fact, generally, such kind of networks are setup for their respective purposes only and not for general purpose.

2) Roaming Consortium: Using this feature advertised again in 802.11u beacon frame, the user can give the choice for the preference of the WLAN network which has roaming partnership with his/her corresponding primary service provider. The *type of* roaming partnership is not in the scope.

3) Internet Connectivity provided: Having a WLAN connected to the internet or not depends upon the implementation and the choice of the owner. 802.11u has one bit dedicated for specifying this feature. If WLAN want to advertise that internet connectivity is provided, this bit is set to 1. If this bit is set to 0, it means that it is not specified whether the internet connectivity is provided or not. Since accessing the internet and related services is one of the most desired features of WLAN, the users can give the choice for the preference of the WLAN which provides internet connectivity.

4) Cost: Cost of using the WLAN is again very important criteria which affects the end user. Here cost means the money which the user has to pay (directly or indirectly) for using the WLAN and typically can be specified as rupees/Mbps. This cost can vary from WLAN to WLAN. There is no direct field in the 802.11u beacon frame specifying the cost of using the WLAN. But this can be achieved using Beacon stuffing, as shown in Chapter 4.

5) Security: It is also one of the most important criteria to be considered for user preference. It is the user who should decide that whether he/she want to prefer secure network or not. Here security is considered to be a feature which can be rated on a scale of 1 to 5. How it is to be done is not in the scope.

To capture the user preference by applying AHP, the above explained criteria and sub criteria are arranged in the form of hierarchy as shown in Figure-6.2. Here class X means the four QoS classes as explained in Section 6.4 above.

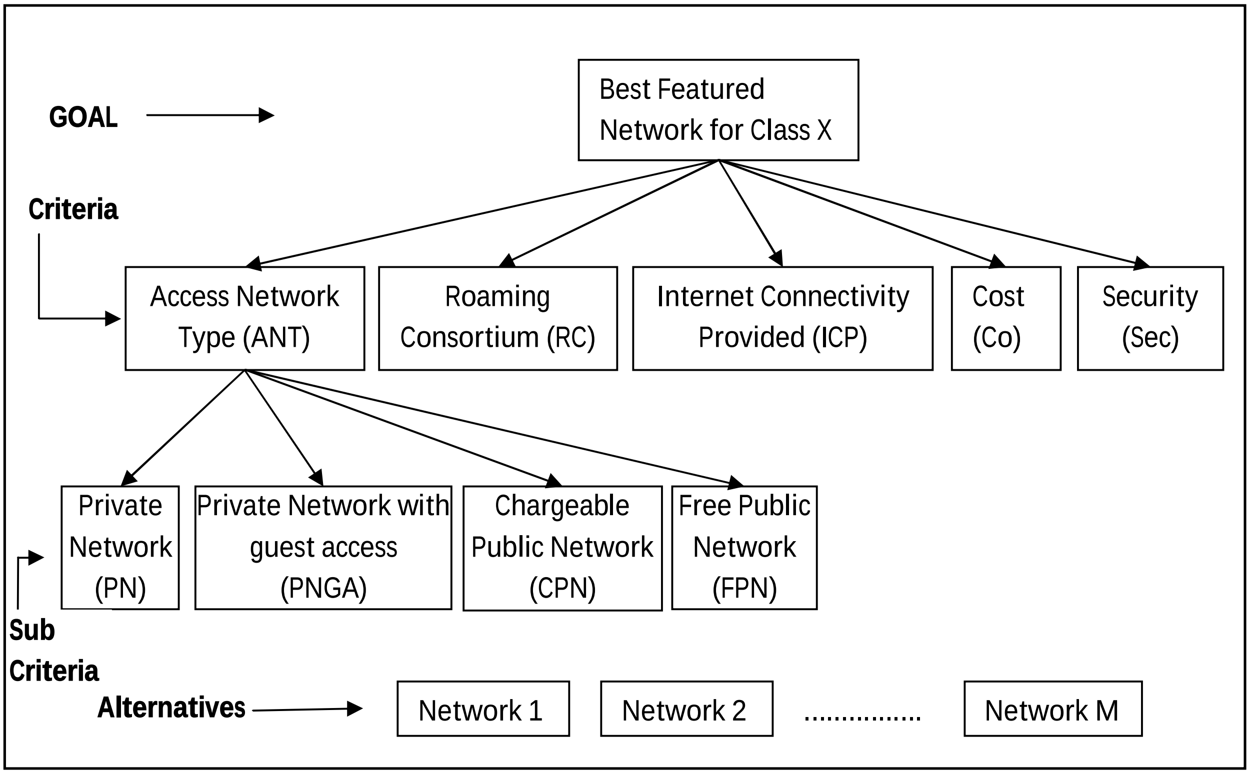


Figure 6.2: AHP decision tree for Class X, where X can be Conversational, Streaming, Interactive, or Background class.

The model justifies the following:

a) The order of judgement matrix which the user has to fill is of the order of 4x4 and 5x5, thus minimizes the probability of getting a contradictory judgement matrix. The corresponding judgement matrices are also comparatively less prone to local rank reversal.
b) Since all the values of alternative networks do have numerical (and not subjective) values, the corresponding judgement matrices will always be perfectly consistent (with CR = 0). Therefore, any number of alternative networks can be considered by the model without bothering about contradictory matrices or rank reversal. The only limiting factor here is the computational capability of the

device. This is because with N available networks, the judgement matrix will be of the order of NxN. As N become large, the number of corresponding matrix multiplications may be an issue.

c) Although the parameters are mainly derived from 802.11u, but even if the WLAN is not 802.11u compatible the values of the parameters can be passed using beacon stuffing.

6.6 Methodology

Figure 6.3 (a) and 6.3 (b) shows the pair-wise judgement matrices corresponding to five criteria and four sub-criteria of the model respectively. The upper triangle of the matrices is required to be filled up by the users.

	ANT	RC	ICP	Co	Sec
ANT	1				
RC		1			
ICP			1		
Co				1	
Sec					1

6.3 (a)

	PN	PNGA	CPN	FPN
PN	1			
PNGA		1		
CPN			1	
FPN				1

6.3 (b)

Figure 6.3: Pair-wise judgement matrices corresponding to Figure 6.2.

Forty two people, aged between 18 to 24 years, participated to give their respective preferences corresponding to the model as in Figure 6.2. These all participants were having the following traits:

a) All the participants were aware of the issues of Mobile Data Offloading. They were having a prior knowledge about the advantages and disadvantages of WLAN with respect to offloading 3G services onto it.

b) All the participants were habitual users of Wi-Fi network, wherever possible, to use internet related activities (surfing/downloading/uploading etc) on their mobile device.
c) All the participants were aware of basics of AHP; specifically about pair-wise comparisons and FCoS. If the participant was not aware, he/she was made aware using a training session.
d) All the participants knew the four classes of QoS as per [*3GPP TS 23.107*]. Again, if the participant did not know, he/she was made to know using a training session.

Corresponding to the model in Figure-6.2 and pair-wise-comparison matrices in Figure 6.3, each participant user gave the preferences as following:

a) For five criteria of the model, one pair-wise comparison matrix of order 5x5 for one QoS class. Since there are four QoS classes (i.e. conversational, streaming, interactive, and background), this resulted in four pair-wise comparison judgement matrices of order 5x5.
b) For four sub-criteria of the model, one pair-wise comparison matrix of order 4x4 for one QoS class. Again, since there are four QoS classes, this resulted in four pair-wise comparison judgement matrices of order 4x4.

This way each participant user ended up giving eight judgement matrices; four matrices of order 4x4 and four matrices of order 5x5. The reciprocal of the values in upper triangle of each matrix was replicated in the corresponding lower triangle, as per AHP process. Also, the consistency ratio (CR) for each judgement matrix was calculated and the decisions were accepted if CR < 10%. This way the responses of all the participant users were collected.

Moreover, the above procedure of taking input from the users was repeated two times:

1) Without any support of giving the feedback about whether the newly added entry contradicts with the previously added entries.

2) Using the tool built on algorithms proposed in Section 5.4. It checks the newly added entry for making the matrix contradictory and gives the appropriate feedback.

6.7 Results and Discussion

Table D.1 shows the responses of 42 participants for 4x4 judgement matrices without any feedback mechanism. Correspondingly, Table D.2 shows the same but with feedback mechanism.

Similarly, Table D.3 shows the responses for 5x5 judgement matrices without any feedback mechanism and correspondingly, Table D.4 shows the same but with feedback mechanism.

It is clearly observed that the number of contradictory judgement matrices is reduced drastically when the pair-wise ratios are supplemented with feedback. Though, theoretically the contradictory matrices should be reduced by 100%, it is not. This is because it is still up to the user whether to ignore the contradictory message or to consider it.

Figure 6.4 shows the percent of contradictory matrices resulted when participant users filled up the judgement matrices of order 4x4 and 5x5. It again clearly shows that for all the four QoS classes, the percent of contradictory ones are always higher when there is no feedback mechanism. In fact, Figure 6.5 shows the exact percent reduction achieved when feedback mechanism was used. It shows that, on average, there is a reduction of 60.27% for 4x4 judgement matrices and 65.74% for 5x5 judgement matrices.
It is also observed, that with the feedback mechanism, the consistency ratio (or average consistency) for 4x4 matrices increased from 6.567 to 5.608 and for 5x5 matrices from 5.995 to 4.23.

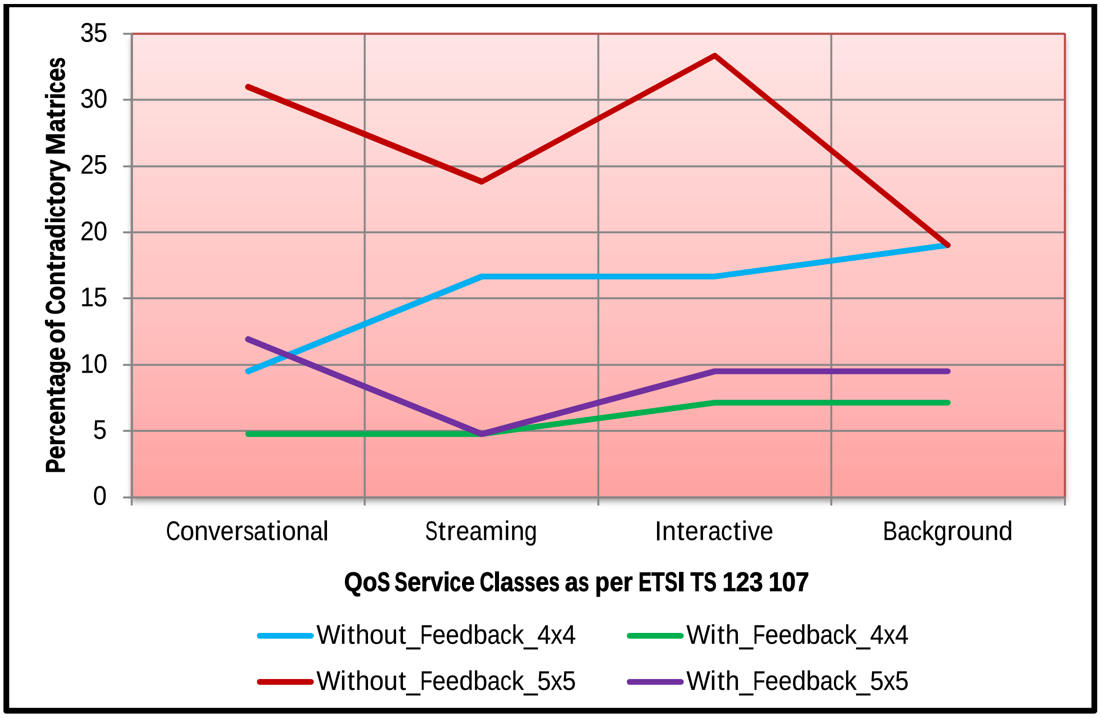


Figure 6.4: Percent of Contradictory matrices with and without feedback mechanism for 4x4 and 5x5 judgement matrices.

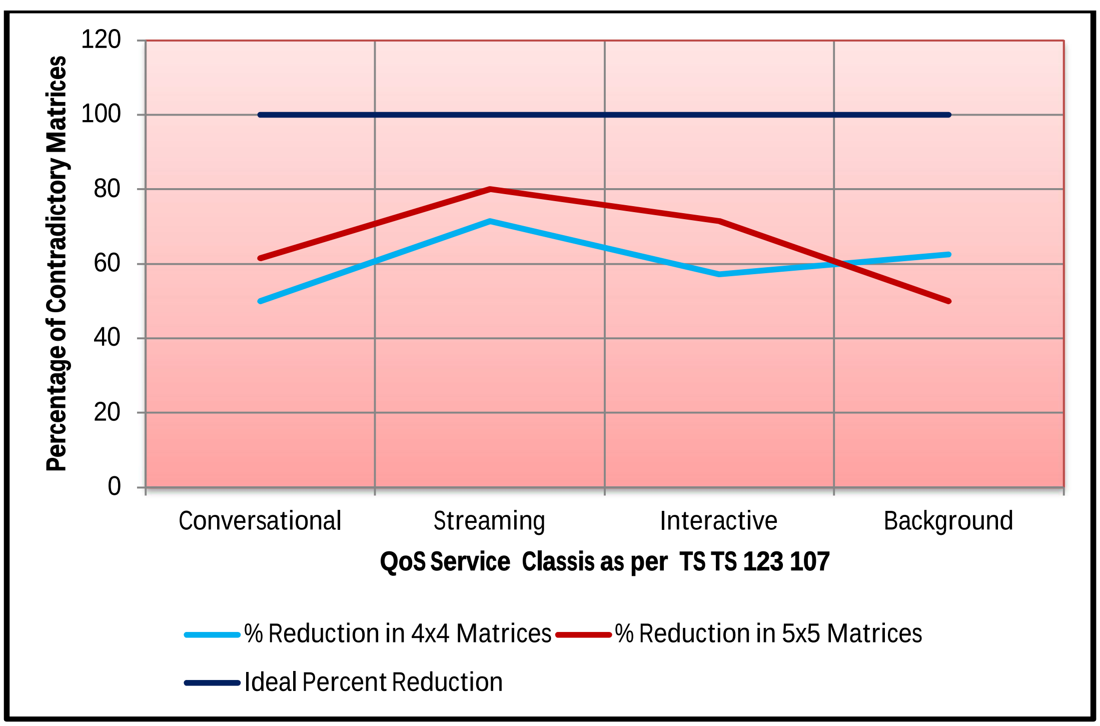


Figure 6.5: Percent reduction in contradictory matrices with feedback mechanism for order 4x4 and 5x5 respectively

CHAPTER 7

Conclusions, Limitations and Scope for Further Research

This chapter concludes the research work presented in this doctoral thesis. It also lists the limitations of the proposed work, thus giving the direction to future scope of work.

7.1 Conclusions

Following are the conclusions drawn from the research work:

a) The improved Beacon Stuffing technique has been successfully implemented in ns-3. It is enhanced to use the unused bits in the Length field of each Information Element, and to use multiple Vendor Specific Information Elements for maximizing the information contents to overload. Though the size of the beacon frame is increased, the maximum length is as per the standard.

b) Assuming that the first 30 frame body elements (both Information Elements and non-information elements) are the part of beacon frame and each Information Element is added successively in the order as specified by IEEE 802.11 standard, results show that the resulting bandwidth from improved beacon stuffing will be in the range of 294 Kbps to 1775 Kbps. This is better than the previous technique [*Chandra et al, 2007*] where the achieved bandwidth is 229 Kbps.

c) AHP allows subjective as well as normative criteria. If the criteria are subjective, the percentage of contradictory matrices increases with increase in order of the judgement matrix. Experimentally it is proved to increase from 0.7% (for 3x3 matrices) to 72.87% (for 9x9 matrices). This clearly indicates that as the order of judgement matrix increases, the probability of it being contradictory also increases sharply.

d) Comparing contradictory judgement matrices with that of non-contradictory ones on qualitative metric (called Local Rank Reversal) and 13 different quantitative metrics again shows that contradictory ones should be avoided as much as possible. This suggests that while designing the problem into hierarchy, the number of subjective criteria to be compared at a time should be as less as possible.

e) The notion of contradictory matrix is enhanced to the notion of contradictory entry. Based on this a feedback based mechanism for getting the input in pair-wise comparison matrix is designed. Using it, if the user corrects the contradictory decisions at all the places, the proposed mechanism can reduce the contradictory judgement matrices by 100%.
f) Since it is argued to avoid (and not *eliminate*) contradictory judgement matrices, to show the practical usefulness of the feedback based technique, as a hypothesis, a model is proposed to capture user preferences during vertical handover decision. Results show that the percentage of contradictory judgement matrices is reduced by 60.27% for order 4x4 and 65.74% for order 5x5.

7.2 Limitations

The research work presented in this thesis has the following limitations:

a) The proposed beacon stuffing technique requires changes in the WLAN driver. This is because stuffing additional information in the Length field of Information Elements and Vendor Specific element cannot be read at the mobile device unless the corresponding WLAN driver is modified.
b) The proposed beacon stuffing technique can of course stuff addition information in the successive beacon frames. But the situation in which one beacon is lost or not received correctly at the mobile device is not within the scope.
c) Though it is said that the size of information which can be embedded in the beacon frame can be unlimited, it is actually constrained by the sequence number field of the beacon frame.
d) Security of the embedded information is again not within the scope. This is because it is assumed that the information embedded is to be broadcast in the public domain.

e) Intrusion detection or authenticity of the embedded information in the beacon frame is again not within the scope. This is because it is assumed that any existing intrusion detection or authenticity-check algorithm will continue working in any such required environment.
f) The information embedding and extracting algorithms listed in section 4.6 will work in single hop scenarios (i.e. AP to mobile device) or in multi-hop scenarios where intermediate routers do not change the beacon header and/or its frame body.
g) Simulation of judgement matrices while experimental analysis in AHP was limited up to the order of 9x9 only. This is because of the computational limitations. Another reason towards this limitation is that the mean consistency index of randomly matrices is given by Saaty for order 3x3 to 9x9 only.
h) While generating inconsistent matrices for simulation, the errors introduced are considered to be additive. This is because the work is based on the assumptions of A. Ishizaka et al [*Ishizaka et al, 2006*].
i) The model proposed to capture user preferences is just a hypothesis for the research work in this thesis and its validation is not within the scope. This is because its main purpose is to show the usefulness of the proposed technique to avoid contradictory judgement matrices in AHP.
j) The participant users were aware with the AHP process and the four different QoS classes. This is because the purpose is to show the usefulness of the proposed technique for avoiding contradictory judgement matrices. Practically, a questionnaire is required to be designed for the purpose.
k) The proposed model could only show the usefulness of the proposed technique for avoiding contradictory matrices of order 4x4 and 5x5. This is because the number of criteria selected for comparison was 4 and 5 respectively.

7.3 Future Research

a) To work towards making beacon stuffing technique a practically useful one by designing a patch which can be downloaded on any mobile device and is able to read the overloaded information contents on it. This also requires studying existing implementations of beacon frames.

b) To enhance the beacon stuffing technique which allow the consideration of lost beacon frames also.

c) Section 4.7 shows the amount of information which can be embedded in the beacon frame. Here it is assumed that the frame body of the beacon is of maximum size. It is quite possible that during practical implementations the size of frame body may be reduced (because of noise etc.). Analysis of such environments will be taken up as a future work.

d) AHP is also criticized for other phenomenon (s), for example Global Rank reversal. Similar results are to be examined during Aggregation phase.

e) Though the model for user preferences is just a hypothesis in this doctoral thesis, validating it is a future course of study. All the users cannot be assumed to know the AHP process. This requires the design of suitable questionnaire to capture user preferences.

REFERENCES

3GPP TR 22.934, Feasibility study on 3GPP system to wireless local area network (WLAN) interworking, Technical Report 3GPP TR 22.934 version 9.1.0, 2010.

3GPP TR 22.935, Feasibility study on Location services (LCS) for wireless local area network (WLAN) interworking, Technical Report 3GPP TR 22.935 version 9.1.0 release 9, 2010.

3GPP TR 22.937, Requirements for service continuity between mobile and wireless local area network (WLAN) networks, Technical Report 3GPP TR 22.937 version 9.0.0 release 9, 2010.

3GPP TS 22.234, Requirements on 3GPP system to wireless local area network (WLAN) interworking, Technical Report 3GPP TS 22.234 version 9.0.0 release 9, 2010.

3GPP TS 23.107, Digital cellular telecommunications system (Phase 2+); Universal Mobile Telecommunications System (UMTS); LTE; Quality of Service (QoS) concept and architecture, Technical Report 3GPP TS 23.107 version 10.2.0 Release 10, 2012.

3GPP TS 23.234, 3GPP system to wireless local area network (WLAN) interworking; system description, Technical Specification 3GPP TS 23.234 version 9.0.0 release 9, 2010.

3GPP TS 23.237, Mobility between 3GPP – wireless local area network (WLAN) interworking and 3GPP systems, Technical Specification 3GPP TS 23.237 version 9.0.0 release 9, 2010.

A. Ghosal, “Mobile data offload: can Wi-Fi deliver”, Intellinet Technologies, Jan 2010. URL: http://www.intellinettech.com

Alkhawlani, M. M., & Ayesh, A. (2008). Access network selection for coexisted WWAN, WMAN and WLAN using combined fuzzy logic and AHP. *International Journal of Innovative Computing and Applications*, *1*(4), 219-231.

Asthana N. C., Nirmal A. (2009). *Urban Terrorism: Myths and realities*. Pointer publishers, India.

Attaullah, H., Iqbal, F., & Javed, M. Y. (2008, November). Intelligent vertical handover decision model to improve QoS. In *Digital Information Management, 2008. ICDIM 2008. Third International Conference on* (pp. 119-124). IEEE.

Bajwa,G., Choo,E.U., & Wedley,W.C. (2008) Effectiveness Analysis of Deriving Priority Vectors from Reciprocal Pairwise Comparison Matrices. *Asia-Pacific Journal of Operational Research*, 25(3), 279-299.

Banerjee, N., Agarwal, S., Bahl, P., Chandra, R., Wolman, A., & Corner, M. (2010). Virtual compass: relative positioning to sense mobile social interactions. *Pervasive Computing*, 1-21.

Barja,J.M., Calafate,C.T., Cano,J.C. & Manzoni,P.(2011,June) An overview of vertical handover techniques: Algorithms, protocols and tools.*Computer Communications*,34(8),985-997.

Bazzi, A., Pasolini, G., & Gambetti, C. (2006, June). SHINE: simulation platform for heterogeneous interworking networks. In *Communications, 2006. ICC'06. IEEE International Conference on* (Vol. 12, pp. 5534-5539). IEEE.

Belton, V., & Gear, A. (1983). On a Shortcoming of Saaty's Method of Analytical Hierarchies*. Omega*, 11, 228-230.
Belton, V., & Gear, A. (1985). The Legitimacy of Rank Reversal—A Comment. *Omega*, 13, 143-144.

Belton, V., & Gear, A. (1985). The legitimacy of rank reversal – A comment. Omega, 13, 143–144.

Bernaschi, M., Cacace, F., & Iannello, G. (2004, August). Vertical handoff performance in heterogeneous networks. In *Proceedings of the 2004 International Conference on Parallel Processing Workshops* (pp. 100-107). IEEE Computer Society.

Brugha, C. M. (2004,November). Structure of Multiple Criteria Decision-Making. *The Journal of the Operational Research Society*,55(11),1156-1168.

Bruno, R., Conti, M., & Gregori, E. (2005). Mesh networks: commodity multihop ad hoc networks. *Communications Magazine, IEEE*, *43*(3), 123-131.

Buburuzan T., Nyamen L. N. (2008), Performance evaluation of an enhanced IEEE 802.21 handover model, in: 1st workshop on wireless broadband access for communities and rural developing regions, Karlstad.

Budescu, D. V., Zwick, R., & Rapoport, A. (1986). A comparison of the eigenvalue method and the geometric mean procedure for ratio scaling. *Applied psychological measurement*, *10*(1), 69-78.

Buddhikot M., Chandramenon G., Han S., Lee Y. W., Miller S., Salgarelli L., “Integration of 802.11 and third-generation wireless data networks”, Twenty-Second Annual Joint Conference of the IEEE Computer and Communications, July 2003, pp. 503- 512 vol.1

Carlos, A., Costa, B., Vansnick , J.(2008,June 16) A critical analysis of the eigenvalue method used to derive priorities in AHP.*European Journal of Operational Research*,187(3), 1422-1428.

Cebeci, U. (2009). Fuzzy AHP-based decision support system for selecting ERP systems in textile industry by using balanced scorecard. Expert Systems with Applications, 36, 8900–8909.

Celik, M., Kandakoglu, A., & Er, D. (2009). Structuring fuzzy integrated multi-stages evaluation model on academic personnel recruitment in MET institutions. Expert Systems with Applications, 36, 6918–6927.

Chamodrakas, I.,Batis, D. & Martakos, D. (2010,January) Supplier selection in electronic marketplaces using satisficing and fuzzy AHP. *Expert Systems with Applications*,37(1), 490-498.

Champion, C.A., Li,X., Zhai,Q., Teng J., & Xuan, D. (2012).*Enclave: Promoting Unobtrusive and Secure Mobile Communications with a Ubiquitous Electronic World*. WASA (pp. 235-247). Springer

Chandra, R., & Bahl, P. (2004, March). MultiNet: Connecting to multiple IEEE 802.11 networks using a single wireless card. In *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies* (Vol. 2, pp. 882-893). IEEE.

Chandra, R., Padhye, J., & Ravindranath, L. (2008, February). Wi-Fi neighborcast: enabling communication among nearby clients. In *Proceedings of the 9th workshop on Mobile computing systems and applications* (pp. 38-42). ACM.

Chandra, R., Padhye, J., Ravindranath, L., & Wolman, A. (2007, March). Beacon-stuffing: Wi-fi without associations. In *Mobile Computing Systems and Applications, 2007. HotMobile 2007. Eighth IEEE Workshop on* (pp. 53-57). IEEE.

Chang, B. J., & Chen, J. F. (2008). Cross-layer-based adaptive vertical handoff with predictive RSS in heterogeneous wireless networks. *Vehicular Technology, IEEE Transactions on*, *57*(6), 3679-3692.

Chang, C.-W., Wu, C.-R., & Lin, H.-L. (2009). Applying fuzzy hierarchy multiple attributes to construct an expert decision making process. Expert Systems with Applications, 36, 7363–7368.

Chen, G. U., Mei, S. O. N. G., Yong, Z. H. A. N. G., Li, W. A. N. G., & Jun-de, S. O. N. G. (2009, May). Novel network selection mechanism using AHP and enhanced GA. In Communication Networks and Services Research Conference, 2009. CNSR'09. Seventh Annual (pp. 397-401). IEEE.

Chen,L.J., Sun,T., Chen,B., Rajendran,V. & Gerla,M.(2004,May) *A smart decision model for vertical handoff*:The 4th Int'l Workshop on Wireless Internet and Reconfigurability

Choo, E.U. & Wedley, W.C.(2004,May) A common framework for deriving preference values from pairwise comparison matrices. *Computers & Operations Research*,31(6), 893-908.

Cisco Visual Networking index (2013). *http://www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-520862.pdf* <accessed March 10 2013>

Corvaja, R. (2006). QoS analysis in overlay Bluetooth-Wifi networks with profile-based vertical handover. *Mobile Computing, IEEE Transactions on*, *5*(12), 1679-1690.

Crawford G. & Williams,C.(1985, December) A note on the analysis of subjective judgment matrices.*Journal of Mathematical Psychology*, 29(4),387-405.

Dagdeviren, M., Yavuz, S., & KilInç, N. (2009). Weapon selection using the AHP and TOPSIS methods under fuzzy environment. Expert Systems with Applications, 36, 8143–8151.

Dutta, A., Das, S., Famolari, D., Ohba, Y., Taniuchi, K., Fajardo, V., Lopez R.M., Kodama T., & Schulzrinne, H. (2007). Seamless proactive handover across heterogeneous access networks. *Wireless Personal Communications*, *43*(3), 837-855.

Dyer, J. (1990a). A clarification of ‘‘remarks on the analytic hierarchy process’’. Management Science, 36, 274–275.

Dyer, J. (1990b). Remarks on the analytic hierarchy process. Management Science, 36, 249–258.

Fall, K., & Varadhan, K. (2000), ns notes and documents The VINT Project. UC Berkeley, LBL, USC/ISI, and Xerox PARC, February 2000.

Fishburn, P.C. (1967). Additive Utilities with Incomplete product set: applications to priorities and assignments. Operations Research society of America, Baltimore, MD, USA.

Forman E., Peniwati K. (1998), "Aggregating individual judgments and priorities with the analytic hierarchy process", European Journal of Operational Research, Volume 108, Issue 1, 1 July 1998, pp 165-169.

Golany,B. & Kress,M.(1993,September 10). A multicriteria evaluation of methods for obtaining weights from ratio-scale matrices. *European Journal of Operational Research*, 69(2) , 210-220.

Goyal, P., & Saxena, S. K. (2008). A dynamic decision model for vertical handoffs across heterogeneous wireless networks. *World Academy of Science, Engineering and Technology*, *31*, 677-682.

Grunenberger, Y., & Rousseau, F. (2010, April). Virtual access points for transparent mobility in wireless LANs. In *Wireless Communications and Networking Conference (WCNC), 2010 IEEE* (pp. 1-6). IEEE.

Hasswa, A., Nasser, N., & Hossanein, H. (2005, March). Generic vertical handoff decision function for heterogeneous wireless. In *Wireless and Optical Communications Networks, 2005. WOCN 2005. Second IFIP International Conference on* (pp. 239-243). IEEE.

Herman, M. W., & Koczkodaj, W. W. (1996). A Monte Carlo study of pairwise comparison. *Information processing letters*, *57*(1), 25-29.

Holder, R. (1990). Some comment on the analytic hierarchy process. Journal of the Operational Research Society, 41, 1073–1076.

Holder, R. (1991). Response to holder’s comments on the analytic hierarchy process: Response to the response. Journal of the Operational Research Society, 42,914–918.

Horrich, S., Ben Jamaa, S., & Godlewski, P. (2007, April). Neural networks for adaptive vertical handover decision. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks and Workshops, 2007. WiOpt 2007. 5th International Symposium on* (pp. 1-7). IEEE.

Hotspot usage to reach 120 billion connects by 2015, Scottasdale, August 2011. http://www.prweb.com/releases/2011/8/prweb8751194.htm <Accessed January, 2013>

Hsu, S. H., Kao, C.-H., & Wu, M.-C. (2009). Design facial appearance for roles in video games. Expert Systems with Applications, 36, 4929–4934.

İç, Y. T., & Yurdakul, M. (2009). Development of a decision support system for machining center selection. *Expert systems with Applications*, *36*(2), 3505-3513.

IEEE standard 802.11 (2007), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications

IEEE standard 802.11k (2008), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 1: Radio resource management of wireless LANs.

IEEE standard 802.11n (2009), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 5: Enhancements for Higher Throughput.

IEEE standard 802.11p (2010), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 6: wireless access in vehicular environments.

IEEE standard 802.11r (2008), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 2: Fast Basic Service Set (BSS) transition.

IEEE standard 802.11s (2011), Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications – amendment 10: mesh networking.

IEEE standard 802.11u (2011), Part 11: wireless LAN medium access control (MAC) and physical layer (PHY) specifications – amendment 9: interworking with external networks.

IEEE standard 802.11v (2011), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 8: IEEE 802.11 wireless network management.

IEEE standard 802.11w (2009), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 4: Protected Management Frames.

IEEE standard 802.11y (2008), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 3: 3650 - 3700 MHz operation in USA.

IEEE standard 802.11z (2010), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications - amendment 7: Extensions to Direct-link setup (DLS).

IEEE standard 802.11 (2012), Part 11: wireless LAN Medium Access Control (MAC) and Physical Layer specifications

Inzerilli, T., Vegni, A. M., Neri, A., & Cusani, R. (2008, October). A location-based vertical handover algorithm for limitation of the ping-pong effect. In *Networking and Communications, 2008. WIMOB'08. IEEE International Conference on Wireless and Mobile Computing,* (pp. 385-389). IEEE.

Ishizaka, A.& Labib,A.(2011,October).Review of the main developments in the analytic hierarchy process. *Expert Systems with Applications*, 38(11), 14336-14345.

Ishizaka, A. & Lusti, M. (2006). How to derive priorities in AHP: a comparative study. *Central European Journal of Operations Research*, 14 (4). pp. 387-400.

Joe, I., & Hong, S. (2007, May). A mobility-based prediction algorithm for vertical handover in hybrid wireless networks. In *Broadband Convergence Networks, 2007. BcN'07. 2nd IEEE/IFIP International Workshop on* (pp. 1-5). IEEE.

Johnson, C.R., Beine,W. B. & Theodore J.W.(1979,February).Right-left asymmetry in an eigenvector ranking procedure. *Journal of Mathematical Psychology*,19(1),61-64.

Jones, D. F. & Mardle, S. J. (2004,August)A Distance-Metric Methodology for the Derivation of Weights from a Pairwise Comparison Matrix. *The Journal of the Operational Research Society*,55(8), 869-875.

Karapetrovic, S., & Rosenbloom, E. S. (1999). A quality control approach to consistency paradoxes in AHP. *European Journal of Operational Research*, *119*(3), 704-718.

Kainulainen,T., Leskinen, P., Korhonen, P., Haara, A. & Hujala, T.(2009,February) . A statistical approach to assessing interval scale preferences in discrete choice problems.*Journal of the Operational Research Society*, 60 (2), 252-258.

Kassar, M., Kervella, B., & Pujolle, G. (2007, December). Architecture of an intelligent inter-system handover management scheme. In *Future generation communication and networking (fgcn 2007)* (Vol. 1, pp. 332-337). IEEE.

Kassar M., Kervella B., Pujolle G., (2008a), An overview of vertical handover decision strategies in heterogeneous wireless networks, Elsevier Computer Communications, 31 (10) (2008).

Kassar, M., Kervella, B., & Pujolle, G. (2008b). Autonomic-oriented architecture for an intelligent handover management scheme. In *Communication Networks and Services Research Conference, 2008. CNSR 2008. 6th Annual* (pp. 139-146). IEEE.

Kim K. Y., Yun D. G., Shim J. C., Cho K. S., Choi S. G., QoS-aware seamless handover schemes between heterogeneous networks in NGN, in: 12th International IEEE Conference on Advanced Communication Technology, 2010. .

Kim, T., Han, S. W., & Han, Y. (2010). A QoS-aware vertical handoff algorithm based on service history information. *Communications Letters, IEEE*, *14*(6), 527-529.

Khosla, R., Goonesekera, T., & Chu, M.-T. (2009). Separating the wheat from the chaff: An intelligent sales recruitment and benchmarking system. Expert Systems with Applications, 36, 3017–3027.

Köksalan, M., Wallenius, J., and Zionts, S. (2011). Multiple Criteria Decision Making: From Early History to the 21st Century. Singapore: World Scientific.

Kwiesielewicz, M., & Uden V., E. (2004). Inconsistent and contradictory judgements in pairwise comparison method in the AHP. *Computers & Operations Research*, *31*(5), 713-719.

Labib, A.W. (2011). A supplier selection model: A comparison of fuzzy logic and the analytic hierarchy process, International Journal of Production Research, 49.

Lassoued I., Bonnin J., Hamouda Z. B., & Belghith A., (2008), A Methodology for evaluating vertical handoff decision mechanisms, International Conference on Networking, pp. 377 - 384, Seventh International Conference on Networking (ICN 2008).

Lee, S.-H. (2010). Using fuzzy AHP to develop intellectual capital evaluation model for assessing their performance contribution in a university. Expert Systems with Applications

Li, M., Chen, S., & Xie, D. (2007, October). A multi-step vertical handoff mechanism for cellular multi-hop networks. In *Proceedings of the 2nd ACM workshop on Performance monitoring and measurement of heterogeneous wireless and wired networks* (pp. 119-123). ACM.

Li, T. S., & Huang, H. H. (2009). Applying TRIZ and Fuzzy AHP to develop innovative design for automated manufacturing systems. *Expert systems with applications*, *36*(4), 8302-8312.

Li, Y., Tang, J., & Luo, X. (2010). An ECI-based methodology for determining the final importance ratings of customer requirements in MP product improvement. Expert Systems with Applications.

Liao, H., Tie, L., & Du, Z. (2006, June). A vertical handover decision algorithm based on fuzzy control theory. In *Computer and Computational Sciences, 2006. IMSCCS'06. First International Multi-Symposiums on* (Vol. 2, pp. 309-313). IEEE.

Lin, C. (2007). A revised framework for deriving preference values from pairwise comparison matrices. European Journal of Operational Research, 176, 1145–1150.

Lin, C.C. (2007, January 16).A revised framework for deriving preference values from pairwise comparison matrices. *European Journal of Operational Research*, 176(2),1145-1150.

Lin, C.-L., Chen, C.-W., & Tzeng, G.-H. (2010). Planning the development strategy for the mobile communication package based on consumers' choice preferences.Expert Systems with Applications

Liu, C.C. & Chen, S.Y. (2009,July). Prioritization of digital capital measures in recruiting website for the national armed forces. *Expert System with Application,*36(5), 9415-9421.

Liu, C., & Zhou, C. (2005, March). An improved interworking architecture for UMTS-WLAN tight coupling. In *Wireless Communications and Networking Conference, 2005 IEEE* (Vol. 3, pp. 1690-1695). IEEE.

Mhatre, V., Lundgren, H., Baccelli, F., & Diot, C. (2007, December). Joint MAC-aware routing and load balancing in mesh networks. In *Proceedings of the 2007 ACM CoNEXT conference* (p. 19). ACM.

Mani, M., & Crespi, N. (2006, November). WLC14-5: handover criteria considerations in future convergent networks. In *Global Telecommunications Conference, 2006. GLOBECOM'06. IEEE* (pp. 1-5). IEEE.

Marquez-Barja, J., Calafate, C. T., Cano, J. C., & Manzoni, P. (2011, March). Evaluation of a technology-aware vertical handover algorithm based on the IEEE 802.21 standard. In *Wireless Communications and Networking Conference (WCNC), 2011 IEEE* (pp. 617-622). IEEE.

MathWorks, Inc. (2005). *MATLAB: the language of technical computing. Desktop tools and development environment, version 7* (Vol. 9). MathWorks.

Mehbodniya, A., & Chitizadeh, J. (2005, March). An intelligent vertical handoff algorithm for next generation wireless networks. In *Wireless and Optical Communications Networks, 2005. WOCN 2005. Second IFIP International Conference on* (pp. 244-249). IEEE.

Miller D.,W., Starr M.,K., (1969). Executive decisions and operations research. Prentice Hall, 2nd edition.

mobithinking.com (2013). *http://www.mobithinking.com/mobile-marketing-tools/latest-mobile-stats/a* <accessed March 10 2013>

Modeler, O. P. N. E. T. (2009). OPNET Technologies Inc. *Bethesda, MD. URL: http://www. opnet. com*.

Nay, P., & Zhou, C. (2009, January). Vertical handoff decision algorithm for integrated UMTS and LEO satellite networks. In *Communications and Mobile Computing, 2009. CMC'09. WRI International Conference on* (Vol. 2, pp. 180-184). IEEE.

Niaraki, A. S., & Kim, K. (2009). Ontology based personalized route planning system using a multi-criteria decision making approach. Expert Systems with Applications, 36, 2250–2259.

Nicholson,A.J., Wolchok, S. & Noble, B.D.(Jan. 2010). Juggler: Virtual Networks for Fun and Profit.*IEEE Transactions on Mobile Computing*,9(1), 31-43.

Niyato, D., & Hossain, E. (2009). Dynamics of network selection in heterogeneous wireless networks: an evolutionary game approach. *Vehicular Technology, IEEE Transactions on*, *58*(4), 2008-2017.

Patil C.,S., Karhe R.,R., Aher M.,A. (2012). development of Mobile Technology: A Survey. International journal of Advanced research in electrical, electronics, and instrumentation engineering. Vol 1, issue 5, pp 374-379.

Pawar, P., Wac, K., Van Beijnum, B. J., Maret, P., van Halteren, A., & Hermens, H. (2008, March). Context-aware middleware architecture for vertical handover support to multi-homed nomadic mobile services. In *Proceedings of the 2008 ACM symposium on Applied computing* (pp. 481-488). ACM.

portioresearch.com(2013). *http://www.portioresearch.com/media/3986/Portio%20Research%20Mobile%20Factbook%202013.pdf* <accessed March 15, 2013.

Saaty T.L., 1980. The Analytic Hierarchy Process, McGraw Hill, New York

Saaty, T. L. (1977). A scaling method for priorities in hierarchical structures. *Journal of mathematical psychology*, *15*(3), 234-281.

Saaty, T. L. (1986). Axiomatic foundation of the analytic hierarchy process. *Management science*, *32*(7), 841-855.

Saaty, T. L. (1994). How to make a decision: The Analytic Hierarchy Process. Interfaces November/December 1994 24: 19-43

Saaty T.L., 2000. fundamentals of Analytic Hierarchy Process. RWS Publications, Pittsburgh, PA.

Saaty, T. L. (2003). Decision-making with the AHP: Why is the principal eigenvector necessary. *European journal of operational research*, *145*(1), 85-91.

Saaty, T. L. (2008). Decision making with the analytic hierarchy process. International Journal of Services Sciences, vol. 1, No 1.

Saaty, T., & Forman, E. (2003). The Hierarchon: A dictionary of hierarchies (Vol. V). Pittsburgh: RWS Publications.

Saaty, T., & Hu, G. (1998). Ranking by eigenvector versus other methods in the analytic hierarchy process. Applied Mathematics Letters, 11, 121–125.

Saaty, T. L., & Vargas, L. G. (1984a). Comparison of eigenvalue, logarithmic least squares and least squares methods in estimating ratios. *Mathematical Modelling*, *5*(5), 309-324.

Saaty, T., & Vargas, L. (1984b). The legitimacy of rank reversal. Omega, 12, 513–516.

Salomon, V. A., & Montevechi, J. A. (2001). A compilation of comparisons on the analytic hierarchy process and others multiple criteria decision making methods: some cases developed in Brazil. In 6th International Symposium on the Analytic Hierarchy Process, Bern.

Sen,J. & Ukil,A. (2010) A QoS-aware end-to-end connectivity management algorithm for mobile applications:Proceedings of the Third Annual ACM Bangalore Conference (COMPUTE '10). ACM, New York, NY, USA.

Sgora, A., Chatzimisios, P., & Vergados, D. D. (2010). Access network selection in a heterogeneous environment using the AHP and fuzzy TOPSIS methods. In Mobile Lightweight Wireless Systems (pp. 88-98). Springer Berlin Heidelberg.

Shere S. V. (2009), design and analysis of application architecture For opportunistic networks Using ad hoc wi-fi, Master of Science in Computer Engineering thesis, the university of texas at arlington.

Siddiqui, F., & Zeadally, S. (2006). Mobility management across hybrid wireless networks: Trends and challenges. *Computer Communications*, *29*(9), 1363-1385.

Singhrova, A., & Prakash, N. (2007, September). A review of vertical handoff decision algorithm in heterogeneous networks. In *Proceedings of the 4th international conference on mobile technology, applications, and systems and the 1st international symposium on Computer human interaction in mobile technology* (pp. 68-71). ACM.

Song Q., Jamalipour A. (2005). "Network selection in an integrated wireless LAN and UMTS environment using mathematical modeling and computing techniques", Wireless Communications, IEEE , vol.12, no.3, pp. 42- 48, June 2005.

Song, Q. & Jamalipour, A. (2005a, May). A network selection mechanism for next generation networks, Communications, 2005. ICC 2005. 2005 IEEE International Conference on , vol.2, no., pp 1418-1422, 16-20 May 2005

Stam, A., & Duarte Silva, P. (2003). On multiplicative priority rating methods for AHP. European Journal of Operational Research, 145, 92–108.

Stevens-Navarro, E., & Wong, V. W. (2006, May). Comparison between vertical handoff decision algorithms for heterogeneous wireless networks. In *Vehicular technology conference, 2006. VTC 2006-Spring. IEEE 63rd* (Vol. 2, pp. 947-951). IEEE.

Stevens-Navarro, E., Lin, Y., & Wong, V. W. (2008). An MDP-based vertical handoff decision algorithm for heterogeneous wireless networks. *Vehicular Technology, IEEE Transactions on*, *57*(2), 1243-1254.

Steuer, R. E., & Choo, E. U. (1983). An interactive weighted Tchebycheff procedure for multiple objective programming. Mathematical programming, 26(3), 326-344.

Summers, W. C., & DeJoie, A. (2004, October). Wireless security techniques: an overview. In Proceedings of the 1st annual conference on Information security curriculum development (pp. 82-87). ACM.

Sur, A., & Sicker, D. C. (2005, October). Multi layer rules based framework for vertical handoff. In *Broadband Networks, 2005. BroadNets 2005. 2nd International Conference on* (pp. 571-580). IEEE.

Taheri, M., Kazemi, M. A. A., & Ashlaghi, A. T. (2011). Improvement of Mobility Management in Heterogeneous Wireless Networks by Using Multiple Attribute Decision Making. *Computer and Information Science*, *4*(4), p100.

The world in 2010: ICT facts and figures, http://www.itu.int/ITU-D/ict/material/FactsFigures2010.pdf <accessed March 10 2013>

Thurstone, L. (1927). A law of comparative judgments. Psychological Review, 34, 273–286.

Triantaphyllou, E., Pardalos, P. M., & Mann, S. H. (1990). Minimization approach to membership evaluation in fuzzy sets and error analysis. *Journal of Optimization Theory and Applications*, *66*(2), 275-287.

Tseng, Y.-F., & Lee, T.-Z. (2009). Comparing appropriate decision support of human resource practices on organizational performance with DEA/AHP model. Expert Systems with Applications, 36, 6548–6558.

Wang, H. S., Che, Z. H., & Wu, C. (2010). Using analytic hierarchy process and particle swarm optimization algorithm for evaluating product plans. Expert Systems with Applications, 37, 1023–1034.

Wang, N. C.; Chiang, Y.; Wong, J. (2012, April). A Hierarchical Mobile IPv6 Handoff Algorithm for Heterogeneous Wireless Networks. Advanced Science Letters, Volume 9, Number 1, pp. 445-450.

Wang, T.-Y., & Yang, Y.-H. (2009). A fuzzy model for supplier selection in quantity discount environments. Expert Systems with Applications, 36, 12179–12187.

Wright, D. J. (2007, July). Maintaining QoS during handover among multiple wireless access technologies. In *Management of Mobile Business, 2007. ICMB 2007. International Conference on the* (pp. 10-10). IEEE.

Xia, L., Ling-ge, J., Chen, H., & Hong-wei, L. (2008, June). An intelligent vertical handoff algorithm in heterogeneous wireless networks. In *Neural Networks and Signal Processing, 2008 International Conference on* (pp. 550-555). IEEE.

Yan, X., Ahmet Şekercioğlu, Y., & Narayanan, S. (2010). A survey of vertical handover decision algorithms in Fourth Generation heterogeneous wireless networks. *Computer Networks*, *54*(11), 1848-1863.

Yang,C., Chuang,S.P., Huang,R.H.(2009,October) Manufacturing evaluation system based on AHP/ANP approach for wafer fabricating industry.*Expert Systems with Applications*,36(8),11369-11377.

Ying, W., Jun, Y., Yun, Z., Gen, L., & Ping, Z. (2008, March). Vertical handover decision in an enhanced media independent handover framework. In *Wireless Communications and Networking Conference, 2008. WCNC 2008. IEEE* (pp. 2693-2698). IEEE.

Yokoyama, M. (1921). The nature of the affective judgment in the method of paired comparisons. *The American journal of psychology*, *32*(3), 357-369.

Yoon P., K., Hwang C., L., (1995). Multiple Attribute Decision Making: an Introduction, SAGE Publications Inc., California.

Zhang, P., Zhou, W., Xie, B., & Song, J. (2010, September). A novel network selection mechanism in an integrated WLAN and UMTS environment using AHP and modified GRA. In *Network Infrastructure and Digital Content, 2010 2nd IEEE International Conference on* (pp. 104-109).

APPENDIX A

Table A.1 shows free bits present in the Length field of each Information Element of the frame body of the beacon frame. Order number is relative order in which the elements are present. Element ID uniquely identifies each Information Element. The base standard 802.11-2007 had only 21 defined information elements and more have been added with the amendments to it. The total number of free bits up to 802.11-2012 is 191. A vendor specific element can also be present and is always the last element. It has no free bits in the length field.

Table A.1: Number of Free Bits available in Length field of each beacon frame

S. No.	Order No.	Information Element	Element ID	Max Length Field	Number of free bits in Length Field
802.11-2007					
1	4	Service Set Identifier (SSID)	0	32	2
2	5	Supported rates	1	8	4
3	6	Frequency-Hopping (FH) Parameter Set	2	5	5
4	7	DS Parameter Set	3	1	7
5	8	CF Parameter Set	4	6	5
6	9	IBSS Parameter Set	6	2	6
7	10	Traffic indication map (TIM)	5	254	0
8	11	Country	7	254	0
9	12	FH Parameters	8	2	6
10	13	FH Pattern Table	9	254	0
11	14	Power Constraint	32	1	7
12	15	Channel Switch Announcement	37	3	6
13	16	Quiet	40	6	5
14	17	IBSS DFS	41	253	0
15	18	TPC Report	35	2	6
16	19	ERP Information	42	1	7
17	20	Extended Supported Rates	50	255	0
18	21	RSN	48	254	0
19	22	BSS Load	11	5	5

S. No.	Orde r No.	Information Element	Element ID	Max Length Field	Number of free bits in Length Field
20	23	EDCA Parameter Set	12	18	3
21	24	QoS Capability	46	1	7
802.11k					
23	25	AP Channel Report	51	255	0
24	26	BSS Average Access Delay	63	1	7
25	27	Antenna Information	64	1	7
26	28	BSS Available Admission Capacity	67	24	3
27	29	BSS AC Access Delay	68	4	5
28	30	Measurement Pilot Transmission Information	66	255	0
29	31	Multiple BSSID	71	255	0
30	32	RRM Enabled Capabilities	70	5	5
802.11r					
31	33	Mobility Domain	54	3	6
802.11y					
32	34	DSE Registered Location	58	20	3
33	35	Extentended Channel Switch Announcement	60	4	5
34	36	Supported Regulatory Classes	59	253	0
		802.11n			
35	37	HT Capabilities	45	26	3
36	38	HT Operation	61	22	3
37	39	BSS Coexistence	72	1	7
38	40	Overlapping BSS Scan Parameters	74	14	4
39	41	Extended Capabilities	127	6	5
802.11v					
40	42	FMS Descriptor	86	255	0
41	43	QoS Traffic Capability	89	3	6
42	44	Time Advertisement	69	16	3
802.11u					
43	45	Interworking	107	9	4
44	46	Advertisement Protocol	108	variable	0

S. No.	Order No.	Information Element	Element ID	Max Length Field	Number of free bits in Length Field
45	47	Roaming Consortium	109	1	7
46	48	Emergency Alert Identifier	112	8	4
		802.11s			
47	49	Mesh ID	114	32	2
48	50	Mesh Configuration	113	7	5
49	51	Mesh Awake Window	119	2	6
50	52	Beacon Timing	120	253	0
51	53	MCCAOP Advertisement Overview	174	6	5
52	54	MCCAOP Advertisement	123	255	0
53	55	Mesh Channel Switch Parameters	118	6	5

APPENDIX B

Appendix B shows the changes done in network simulator (ns-3) to implement beacon stuffing.

B.1 Implementation at AP

The following global variables are defined in [*ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc*]

```
uint8_t oui[3] = {255,255,255};
uint8_t bits_used[50] = {6, 4, 3, 1, 3, 8, 2, 8, 2, 8, 8, 3, 5, 8, 8, 8,
8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 8, 1, 8, 8, 2, 8, 2, 8, 8,
3, 8, 1, 8, 5, 1};
uint8_t element_ids[20] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 32, 35,
37, 40, 41, 42, 45, 46};
uint32_t max_beacon_size;

uint8_t info[10000];
uint16_t info_length;
uint16_t info_read;
uint8_t num_elements;

uint8_t fragment_info[3000];
uint16_t fragment_info_length;
uint16_t current_index;
int current_bit;

uint8_t control_ssid;
uint8_t control_rates;

uint16_t length_bits_used;
uint16_t length_bits;

intmax_vendor;
int max_vendor_octets;
intvendor_used;
int vendor_octets_used;

uint8_t sta_info[3000];
uint16_t sta_current_index;
int sta_current_bit;

uint8_t sta_control_ssid;
uint8_t sta_control_rates;
uint8_t sta_bssid[6];
uint8_t sta_oui[3];
```

B.1.1 Initializing AP

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc]

```
void initializeVariables(){
int i;

max_beacon_size = 2304;

memset(info, 0, 10000);
info_read = 0;

memset(fragment_info, 0, 3000);
fragment_info_length = 0;

memset(sta_bssid, 0, 6);
memset(sta_info, 0 ,3000);

length_bits_used = 0;
length_bits = 0;
control_ssid = 0;
control_rates = 0;
current_index = 0;
current_bit = 7;
sta_control_ssid = 0;
sta_control_rates = 0;

srand( time(NULL) );
info_length = rand() % 5000;
num_elements = rand() % 18;

for (i=0; i< info_length; i++){
info[i] = rand() % 127 + 1;
  }
}
```

[ns-3.15/scratch/final.cc]

```
int i;
extern uint8_t info[10000];
extern uint16_t info_length;
extern uint8_t num_elements;
extern uint8_t control_rates;

std::cout<< "Starting Simulator...\n";
std::cout<< "Initializing variables...\n";
initializeVariables();
std::cout<< "Variables Initialized.\n";
```

```
std::cout<< "\n";

control_rates = 2; // set to any value from 0-7

std::cout<< "Control Rates: " << (int) control_rates << "\n";
std::cout<< "Number of information elements excluding SSID, Rates, Vendor Specific: " << (int) num_elements << "\n";
std::cout<< "Information length: " << (int) info_length << " octets.\n";
std::cout<< "Information is as follows:\n";

for(i = 0; i < info_length; i++){
std::cout<< (int) info[i] << " ";
}
std::cout<< "\n";
```

B.1.2 Inserting data in BSSID:

To accomplish this, following code is modified/added in the function ***ApWifiMac::SendOneBeacon [ns-3.15/ns-3.15/src/wifi/model/ap-wifi-mac.cc]***

```
WifiMacHeaderhdr;
Mac48Address bss;
uint16_t remaining;

ResetAP(); // defined in [ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc] which just resets variables to default state
std::cout<< "\n";

if(isBssidChanged(control_rates)){
bss.CopyFrom(info + info_read);
info_read += 6;
}
else{
bss = GetAddress();
}

remaining = info_length - info_read;
if (remaining < 7 &&isBssidChanged(control_rates)){
control_ssid = 0;
}
else{
control_ssid = 1;
}
hdr.SetAddr3 (bss);
```

B.1.3 Get number of octets to be added to length fields

To accomplish this, following code is modified/added in the function ***MgtProbeResponseHeader::GetSerializedSize***

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc]

```
uint32_t size = 0;      // size of the beacon frame body
int info_octets = 0;
int j;
int remaining;

for(j = 0; j < num_elements; j++){
length_bits = length_bits + 8 - bits_used[element_ids[j+2]];
size += 2;
size += 1 << bits_used[element_ids[j+2]];
}

info_octets = length_bits / 8;
length_bits = info_octets * 8;

if (hasMoreData() && info_octets &&isLengthChanged(control_rates)){
remaining = info_length - info_read;

if (remaining <= info_octets){
control_ssid = 0;
fragment_info_length = remaining;
  }
else{
control_ssid = 1;
fragment_info_length = info_octets;
  }

std::cout<< "Octets in length fields: " << (int) fragment_info_length<<
"\n";
length_bits = fragment_info_length * 8;
info_octets = length_bits / 8;

memcpy(fragment_info, info + info_read, fragment_info_length);
info_read += fragment_info_length;
}
else{
info_octets = 0;
}
```

B.1.4 Get number of octets to be added to Vendor Specific elements.

To accomplish this, following code is modified/added in the function ***MgtProbeResponseHeader::GetSerializedSize***

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc]

```
if (hasMoreData()&&isVendorChanged(control_rates)){
remaining = info_length - info_read;

max_vendor = (max_beacon_size - size) / 257;
max_vendor_octets = max_vendor * 252;

if ((max_beacon_size - size) % 257 > 5){
max_vendor += 1;
max_vendor_octets += ((max_beacon_size - size) % 257) - 5;
  }

if (remaining <= max_vendor_octets){
control_ssid = 0;
vendor_octets_used = remaining;
  }
else{
control_ssid = 1;
vendor_octets_used = max_vendor_octets;
  }

vendor_used = vendor_octets_used / 252;
size += 257 * vendor_used;

if (vendor_octets_used % 252){
size += 5;
size += vendor_octets_used % 252;
  }

std::cout<< "Octets in vendor fields: " << (int) vendor_octets_used<<
"\n";
memcpy(fragment_info + info_octets, info + info_read,
vendor_octets_used);
info_read += vendor_octets_used;
}
```

B.1.5 Adding control data

To accomplish this, following code is modified/added in the function ***WifiInformationElement::Serialize***

[ns-3.15/ns-3.15/src/wifi/model/wifi-information-element.cc]

```
uint8_t mask;
uint8_t length = GetInformationFieldSize();

if (ElementId() == IE_SSID){
mask = control_ssid << 6;
length = mask | length;
}

if (ElementId() == IE_SUPPORTED_RATES){
mask = control_rates << 4;
length = mask | length;
}

i.WriteU8 (ElementId ());
i.WriteU8 (length);
```

B.1.6 Adding data to the length fields

To accomplish this, following code is added in the file

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc]

```
uint8_t GetNextBit() {
uint8_t bit;

if (length_bits_used == length_bits || current_index >
fragment_info_length)
bit = 0;
else{
bit = fragment_info[current_index] & (1 << current_bit);
bit = bit >> current_bit;
current_bit -= 1;

if (current_bit < 0){
current_bit = 7;
current_index += 1;
length_bits_used += 1;
    }
  }
return bit;
}
```

```
uint8_t GetNextBits(uint8_t count){
uint8_t bits, j, next_bit;
bits = 0;

for(j = 0; j < count; j++){
next_bit = GetNextBit();
bits = bits | next_bit<< (7 - j);
  }
return bits;
}
```

These lines are added *to* ***MgtProbeResponseHeader::Serialize in***

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc] file

```
uint8_t j;
uint8_t length = 0;
uint8_t bits_available = 0;
uint8_t dummy[255] = {0};

for(j = 0; j < num_elements; j++){
i.WriteU8 (element_ids[j+2]);
length = 0;

bits_available = 8 - bits_used[element_ids[j+2]];
if (isLengthChanged(control_rates)){
length = GetNextBits(bits_available);
i.WriteU8 (length);
  }
else
i.WriteU8(length);
i.Write (dummy, 1<<bits_used[element_ids[j+2]]);
}
```

B.1.7 Adding data to the vendor specific elements

To accomplish this, following code is modified/added in the function ***MgtProbeResponseHeader::Serialize***

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc]

```
while (vendor_octets_used > 0){
i.WriteU8(IE_VENDOR_SPECIFIC);
if (vendor_octets_used >= 252)
length = 255;
else
length = vendor_octets_used + 3;
```

```
i.WriteU8(length);
i.Write(oui, 3);
i.Write(fragment_info + current_index, length - 3);
current_index += length - 3;
vendor_octets_used -= length - 3;
}
```

B.2 Implementation at STA

We have dealt with part of adding information to the beacon. Now, we have to read the data from beacons received by the STAs. The following code performs this.

B.2.1 Reading BSSID

To accomplish this, following code is modified/added in the function

StaWifiMac::Receive

[ns-3.15/ns-3.15/src/wifi/model/sta-wifi-mac.cc]

```
Mac48Address bssid;

ResetSTA();
bssid = hdr->GetAddr3();
bssid.CopyTo(sta_bssid);
```

B.2.2 Reading control data:

To accomplish this, following code is modified/added in the function ***WifiInformationElement::DeserializeIfPresent***

[ns-3.15/ns-3.15/src/wifi/model/wifi-information-element.cc]

```
uint8_t mask;

//length contains the length of the information element read

if (ElementId() == IE_SSID){
sta_control_ssid = length >> 6;
mask = (1 << 6) - 1;
length = length & mask;
}

else if (ElementId() == IE_SUPPORTED_RATES){
sta_control_rates = length >> 4;
mask = (1 << 4) - 1;
```

```
length = length & mask;
}
```

B.2.3 Copying BSSID to sta_info

To accomplish this, following code is modified/added in the function

MgtProbeResponseHeader::Deserialize

[*ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc*]

```
if (isBssidChanged(sta_control_rates)
memcpy(sta_info, sta_bssid, 6);
sta_current_index += 6;
}
```

B.2.4 Reading data from the length fields

To accomplish this, following code is added in the file

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc]

```
void PutNextBit(uint8_t bit){

sta_info[sta_current_index] = sta_info[sta_current_index] | (bit <<
sta_current_bit);
sta_current_bit -= 1;

if ( sta_current_bit < 0){
sta_current_index += 1;
sta_current_bit =7;
  }
}

void PutNextBits(uint8_t data, uint8_t count){
uint8_t bit = 0;
uint8_t j;

for(j = 0; j < count; j++){
bit = 0;
bit = data & (1 << (7-j));
bit = bit >> (7-j);
PutNextBit(bit);
  }
}
```

Following code is added in the ***MgtProbeResponseHeader::Deserialize*** function of [*ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc*] file.

```
elementId = i.ReadU8 ();
while (elementId< IE_VENDOR_SPECIFIC &&elementId> IE_SUPPORTED_RATES){
length = i.ReadU8 ();
if (isLengthChanged(sta_control_rates))
PutNextBits(length, 8 - bits_used[elementId]);
i.Next (1<<bits_used[elementId]);
elementId = i.ReadU8 ();
}
```

B.2.5 Reading data from the Vendor Specific elements

To accomplish this, following code is added in the ***MgtProbeResponseHeader::Deserialize*** *function of*

[ns-3.15/ns-3.15/src/wifi/model/mgt-headers.cc] *file.*

```
while (elementId == IE_VENDOR_SPECIFIC){
length = i.ReadU8();
i.Read(sta_oui, 3);
if (isCustomOUI() &&isVendorChanged(sta_control_rates))
i.Read (sta_info + sta_current_index, length - 3);
else
i.Next(length - 3);
sta_current_index += (length - 3);
elementId = i.ReadU8();
}
```

APPENDIX C

Table C.1: Percentage of Contradictory Judgement Matrices for order 3x3

	Order of Matrix = 3x3		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
0	248	0	0
1	102	0	0
2	102	0	0
3	60	0	0
4	60	6	10
5	66	0	0
6	78	0	0
7	30	0	0
8	54	0	0
9	54	0	0

Table C.2: Percentage of Contradictory Judgement Matrices for order 4x4

	Order of Matrix = 4x4		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
0	12800	0	0
1	21228	1920	9.04
2	30972	3324	10.73
3	35616	3000	8.42
4	42660	5052	11.84
5	47652	4632	9.72
6	53988	7572	14.02
7	60756	9408	15.48
8	63492	8328	13.11
9	68652	8301	12.09

Table C.3: Percentage of Contradictory Judgement Matrices for order 5x5

	Order of Matrix = 5x5		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
Judgement matrices w.r.t. Case-1 as explained in section 5.3			
0	243747	55632	22.82
1	240541	50438	20.96
2	271937	51163	18.81
3	277169	52782	19.04
4	328240	90881	27.68
5	317322	85819	27.04
6	299721	70192	23.41
7	297261	69753	23.46
8	313095	83480	26.66
9	313429	85628	27.31
Judgement matrices w.r.t. Case-2 as explained in section 5.3			
0	258523	70137	27.12
1	279454	67710	24.22
2	310358	69443	22.37
3	303689	69146	22.76
4	338950	94113	27.76
5	333265	98009	29.40
6	323819	87798	27.11
7	317354	85038	26.79
8	332541	97437	29.30
9	330205	97982	29.67
Judgement matrices w.r.t. Case-3 as explained in section 5.3			
0	259348	69999	26.99
1	280614	68298	24.33
2	311864	69887	22.40
3	306211	69925	22.83
4	339798	94084	27.68
5	334074	98407	29.45
6	324961	87623	26.96
7	319213	84839	26.57
8	333885	97952	29.33
9	330950	99054	29.93

Table C.4: Percentage of Contradictory Judgement Matrices for order 6x6

	Order of Matrix = 6x6		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
Judgement matrices w.r.t. Case-1 as explained in section 5.3			
0	246775	78258	31.71
1	260250	74620	28.67
2	299136	87769	29.34
3	331110	108960	32.90
4	330324	105127	31.82
5	345015	115195	33.38
6	358658	125415	34.96
7	383395	148304	38.68
8	404412	166569	41.18
9	419294	181283	43.23
Judgement matrices w.r.t. Case-2 as explained in section 5.3			
0	259717	98938	38.09
1	295881	98231	33.19
2	327868	110549	33.71
3	356456	132273	37.10
4	358378	131307	36.63
5	374167	141683	37.86
6	387886	151672	39.10
7	413943	172649	41.70
8	434041	189575	43.67
9	453455	205379	45.29
Judgement matrices w.r.t. Case-3 as explained in section 5.3			
0	260403	99534	38.22
1	297146	99300	33.41
2	329295	110348	33.51
3	356770	132662	37.18
4	359642	132109	36.73
5	373435	141705	37.94
6	384907	150129	39.00
7	412570	173049	41.94
8	433585	189201	43.63
9	453197	205020	45.23

Table C.5: Percentage of Contradictory Judgement Matrices for order 7x7

	Order of Matrix = 7x7		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
Judgement matrices w.r.t. Case-1 as explained in section 5.3			
0	197292	78546	39.81
1	234078	84697	36.18
2	280789	108068	38.48
3	305057	121325	39.77
4	330529	137059	41.46
5	363444	165478	45.53
6	394455	192098	48.69
7	427316	219313	51.32
8	464106	249587	53.77
9	504805	282434	55.94
Judgement matrices w.r.t. Case-2 as explained in section 5.3			
0	209862	100284	47.78
1	261615	111119	42.47
2	307361	137318	44.67
3	331921	152583	45.96
4	359013	171430	47.75
5	394312	198003	50.21
6	429425	226101	52.65
7	465420	256906	55.19
8	506575	291711	57.58
9	552785	329987	59.69
Judgement matrices w.r.t. Case-3 as explained in section 5.3			
0	211311	101218	47.90
1	261354	110898	42.43
2	309263	138508	44.78
3	333994	153564	45.97
4	359025	170710	47.54
5	396069	200582	50.64
6	433318	229716	53.01
7	468938	259244	55.28
8	510560	294134	57.61
9	553637	330658	59.72

Table C.6: Percentage of Contradictory Judgement Matrices for order 8x8

	Order of Matrix = 8x8		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
Judgement matrices w.r.t. Case-1 as explained in section 5.3			
0	134697	62532	46.42
1	193581	85431	44.13
2	240519	111081	46.18
3	279549	137335	49.12
4	325472	174307	53.55
5	370648	211145	56.96
6	420849	254187	60.39
7	475351	300183	63.14
8	534153	349759	65.47
9	593549	400178	67.42
Judgement matrices w.r.t. Case-2 as explained in section 5.3			
0	148712	81781	54.99
1	215926	110526	51.18
2	265445	142496	53.68
3	305423	171016	55.99
4	355245	210137	59.15
5	406611	253393	62.31
6	463567	302134	65.17
7	526938	356936	67.73
8	589389	409881	69.54
9	652562	462951	70.94
Judgement matrices w.r.t. Case-3 as explained in section 5.3			
0	148529	81876	55.12
1	216059	110210	51.00
2	267332	143780	53.78
3	308660	173064	56.06
4	358589	212188	59.17
5	408178	253682	62.14
6	462540	301324	65.14
7	526431	356545	67.72
8	591358	411251	69.54
9	651495	461222	70.79

Table C.7: Percentage of Contradictory Judgement Matrices for order 9x9

	Order of Matrix = 9x9		
Bin No.	**Number of inconsistent matrices**	**Number of contradictory matrices**	**Percentage**
Judgement matrices w.r.t. Case-1 as explained in section 5.3			
0	78342	40218	51.33
1	142064	71928	50.63
2	192019	102688	53.47
3	239530	140603	58.69
4	291122	184113	63.24
5	346379	231735	66.90
6	412361	289345	70.16
7	483371	349271	72.25
8	548930	406758	74.10
9	616752	465994	75.55
Judgement matrices w.r.t. Case-2 as explained in section 5.3			
0	90084	53891	59.82
1	160375	93401	58.23
2	213179	131341	61.61
3	263137	172899	65.70
4	320453	221713	69.18
5	385130	278347	72.27
6	459554	343607	74.76
7	531404	406399	76.47
8	606630	470842	77.61
9	679441	533724	78.55
Judgement matrices w.r.t. Case-3 as explained in section 5.3			
0	91109	54363	59.66
1	160638	94003	58.51
2	210497	129839	61.68
3	261800	170747	65.22
4	320926	221347	68.97
5	385508	278410	72.21
6	457966	342346	74.75
7	530674	405380	76.38
8	603918	468784	77.62
9	675224	529836	78.46

Table C.8: Best-Case and Any-Case Local Rank Reversals for contradictory and non-contradictory judgement matrices of order 4x4

Order of Judgement Matrices: 4x4				
	Contradictory Matrices		**Non-Contradictory Matrices**	
Bin	**Best-Case**	**Any-Case**	**Best-Case**	**Any-Case**
Bin[0]	0	0	0.03	0.22
Bin[1]	0	0	0.03	0.28
Bin[2]	0	0	1.18	3.31
Bin[3]	0	0	1.39	3.18
Bin[4]	0	0.02	0.96	3.19
Bin[5]	0.54	9.93	1.84	4.41
Bin[6]	1.58	6.37	1.25	3.9
Bin[7]	1.79	6.89	1.41	4.16
Bin[8]	0.86	6.05	1.22	4.01
Bin[9]	1.45	5.52	1.83	5.02

Table C.9: Best-Case and Any-Case Local Rank Reversals for contradictory and non-contradictory judgement matrices of order 5x5

Order of Judgement Matrices: 5x5				
	Contradictory Matrices		**Non-Contradictory Matrices**	
Bin	**Best-Case**	**Any-Case**	**Best-Case**	**Any-Case**
Bin[0]	0	0	0.18	0.56
Bin[1]	0.53	1.27	0.56	2.38
Bin[2]	1.29	5.32	0.91	3.13
Bin[3]	2.02	10.54	1.08	3.84
Bin[4]	1.58	7.91	1.28	5.17
Bin[5]	2.09	8.99	1.41	5.09
Bin[6]	2.09	9.06	1.52	5.56
Bin[7]	1.99	10.83	1.37	6.19
Bin[8]	2.82	11.82	1.56	6.39
Bin[9]	3.3	13.79	1.62	6.63

Table C.10: Best-Case and Any-Case Local Rank Reversals for contradictory and non-contradictory judgement matrices of order 6x6

Order of Judgement Matrices: 6x6				
	Contradictory Matrices		**Non-Contradictory Matrices**	
Bin	**Best-Case**	**Any-Case**	**Best-Case**	**Any-Case**
Bin[0]	0.57	1.48	0.46	2.24
Bin[1]	1.19	4.79	1.09	4.78
Bin[2]	2.46	12.53	1.45	6.67
Bin[3]	2.46	12.71	1.54	7.76
Bin[4]	2.66	13.89	1.75	8.57
Bin[5]	2.95	15.78	1.89	9.42
Bin[6]	3.23	17.8	1.89	10.23
Bin[7]	3.63	19.64	2.05	11.14
Bin[8]	3.94	21.94	2.17	11.89
Bin[9]	4.26	24.55	3.39	12.77

Table C.11: Best-Case and Any-Case Local Rank Reversals for contradictory and non-contradictory judgement matrices of order 7x7

Order of Judgement Matrices: 7x7				
	Contradictory Matrices		**Non-Contradictory Matrices**	
Bin	**Best-Case**	**Any-Case**	**Best-Case**	**Any-Case**
Bin[0]	0.74	3.62	0.79	4.68
Bin[1]	1.88	11.1	1.42	7.96
Bin[2]	2.63	16.1	1.77	10.49
Bin[3]	2.98	18.45	1.96	12.08
Bin[4]	3.36	21.37	2.1	13.31
Bin[5]	3.79	24.13	2.26	14.38
Bin[6]	4.22	26.88	2.44	15.58
Bin[7]	4.63	29.77	2.68	16.99
Bin[8]	5.07	32.88	2.94	18.28
Bin[9]	5.47	35.73	3.18	19.69

Table C.12: Best-Case and Any-Case Local Rank Reversals for contradictory and non-contradictory judgement matrices of order 8x8

Order of Judgement Matrices: 8x8				
	Contradictory Matrices		**Non-Contradictory Matrices**	
Bin	**Best-Case**	**Any-Case**	**Best-Case**	**Any-Case**
Bin[0]	1.374	9.038	1.154	9.338
Bin[1]	2.46	16.852	1.868	12.648
Bin[2]	3.168	22.384	2.192	15.696
Bin[3]	3.734	26.632	2.318	17.494
Bin[4]	4.3	30.56	2.51	19.134
Bin[5]	4.838	34.43	2.766	20.72
Bin[6]	5.366	38.158	3.084	22.446
Bin[7]	5.884	41.636	3.53	24.29
Bin[8]	6.334	44.888	3.748	26.166
Bin[9]	6.746	48.006	4.06	28.002

Table C.13: Best-Case and Any-Case Local Rank Reversals for contradictory and non-contradictory judgement matrices of order 9x9

Order of Judgement Matrices: 9x9				
	Contradictory Matrices		**Non-Contradictory Matrices**	
Bin	**Best-Case**	**Any-Case**	**Best-Case**	**Any-Case**
Bin[0]	1.842	13.366	1.638	13.016
Bin[1]	2.712	21.204	2.21	16.948
Bin[2]	3.456	28.286	2.458	20.282
Bin[3]	4.098	33.532	2.594	22.422
Bin[4]	4.8	38.592	2.852	24.414
Bin[5]	5.502	43.324	3.2	26.434
Bin[6]	6.242	47.672	3.654	28.676
Bin[7]	6.862	51.548	4.106	32.546
Bin[8]	7.364	54.92	4.528	33.27
Bin[9]	7.74	58.104	4.82	35.566

Table C.14: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. distance functions LSM and LWS

Order of Judgement Matrix	Distance Function = LSM		Distance Function = LWS	
	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices
3x3	0.41	2.39	0.17	1.58
4x4	6.26	12.22	3.31	7.49
5x5	26.76	30.83	15.23	19.63
6x6	47.79	58.19	24.62	33.71
7x7	76.24	95.34	35.72	50.31
8x8	113.58	147.212	46.382	68.662
9x9	156.265	208.0425	57.685	88.7775

Table C.15: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. distance functions PWLS and PWLWS

Order of Judgement Matrix	Distance Function = PWLS		Distance Function = PWLWS	
	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices
3x3	0.04	0.03	0.01	0.01
4x4	0.1	0.09	0.05	0.05
5x5	0.27	0.17	0.16	0.11
6x6	0.29	0.2	0.17	0.11
7x7	0.32	0.23	0.17	0.11
8x8	0.352	0.254	0.162	0.108
9x9	0.3675	0.28	0.155	0.1

Table C.16: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. distance functions LAE and LWAE

Order of Judgement Matrix	**Distance Function = LAE**		**Distance Function = LWAE**	
	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices
3x3	1.48	2.24	0.41	1.09
4x4	5.56	6.67	1.69	2.51
5x5	11.31	10.99	3.51	3.84
6x6	17.98	17.95	4.46	5.07
7x7	26.18	26.68	5.38	6.24
8x8	37.976	38.092	6.402	7.38
9x9	48.8425	50.925	6.9525	7.4375

Table C.17: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. distance functions PWLAE and PWLWAE

Order of Judgement Matrix	**Distance Function = PWLAE**		**Distance Function = PWLWAE**	
	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices
3x3	0.47	0.31	0.12	0.12
4x4	0.81	0.72	0.2	0.21
5x5	1.14	0.9	0.35	0.26
6x6	1.45	1.21	0.35	0.28
7x7	1.77	1.52	0.36	0.28
8x8	2.174	1.864	0.36	0.28
9x9	2.63	2.2325	0.355	0.2775

Table C.18: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. distance functions LLS and LLWS

Order of Judgement Matrix	**Distance Function = LLS**		**Distance Function = LLWS**	
	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices
3x3	0.32	0.18	0.05	0.03
4x4	1.22	0.94	0.24	0.19
5x5	4.02	2.14	1.26	0.57
6x6	5.63	3.37	1.46	0.72
7x7	7.58	4.85	1.64	0.82
8x8	10.506	6.832	1.778	0.914
9x9	12.8	9.2175	1.875	1.005

Table C.19: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. distance functions LLAE and LLWAE

Order of Judgement Matrix	**Distance Function = LLAE**		**Distance Function = LLWAE**	
	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices	Average distance for Contradictory Matrices	Average distance for Non-Contradictory Matrices
3x3	1.39	0.93	0.23	0.15
4x4	3.27	2.85	0.48	0.42
5x5	5.59	4.34	1.05	0.66
6x6	8.36	6.89	1.14	0.77
7x7	11.81	10.05	1.22	0.85
8x8	16.118	14.064	1.348	0.904
9x9	22.1275	18.7375	1.385	0.955

Table C.20: Average Distance for contradictory and non-contradictory judgement matrices w.r.t. Minimum Violation metric

Order of Judgement Matrix	**Minimum Violation metric**	
	Contradictory Matrices	Non-Contradictory Matrices
3x3	2	0.32
4x4	1.82	0.6
5x5	2.19	0.99
6x6	2.89	1.54
7x7	3.81	2.21
8x8	5.076	3.056
9x9	6.3275	3.9725

APPENDIX D

Appendix D shows the responses of users wrt to Figure 6.2. Each row depicts the upper triangle values and the ones shown as bold correspond to the contradictory matrix.

Table D.1: User Preferences without feedback mechanism for four sub-criteria wrt Figure 6.2

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
CONVERSATIONAL CLASS	4	6	4	4	0.5	0.333333	9.02
	4	6	4	4	2	0.25	9.71
	7	9	6	4	1	0.25	9.80
	6	9	4	5	1	0.333333	8.09
	0.111111	**0.125**	**0.142857**	**1**	**0.5**	**2**	**7.99**
	9	1	9	0.111111	1	9	0.00
	4	9	6	4	1	0.333333	4.07
	8	9	4	4	0.5	0.25	9.76
	2	4	8	2	4	2	0.00
	4	8	4	4	2	0.25	6.48
	2	1	1	0.5	0.5	1	0.00
	4	6	4	4	2	0.333333	7.50
	4	9	5	5	1	0.2	6.53
	5	9	4	7	1	0.333333	9.73
	0.125	**1**	**0.5**	**7**	**6**	**1**	**1.71**
	5	9	4	7	1	0.2	9.85
	7	9	6	4	1	0.333333	8.19
	5	9	6	6	1	0.25	9.87
	0.333333	**0.5**	**1**	**3**	**2**	**0.5**	**6.55**
	9	3	1	0.333333	9-Jan	0.333333	0.00
	5	9	6	5	1	0.333333	6.73
	5	9	5	4	1	0.333333	3.81
	1	2	2	2	2	1	0.00
	4	8	4	4	1	0.2	4.84
	2	**4**	**9**	**1**	**4**	**1**	**8.45**
	5	6	5	4	1	0.333333	8.64
	5	7	4	4	1	0.2	8.14
	0.333333	0.5	0.5	3	2	0.333333	6.25
	6	8	4	4	1	0.333333	6.75
	5	9	4	5	0.5	0.333333	8.44
	4	6	4	4	0.5	0.25	9.71
	4	6	5	4	2	0.333333	7.90
	4	8	4	4	0.5	0.333333	6.20

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	5	9	7	4	2	0.333333	6.82
	4	6	4	4	0.5	0.333333	9.02
	5	7	4	4	0.5	0.2	9.53
	4	8	5	4	2	0.333333	4.71
	5	8	6	4	1	0.25	8.33
	4	9	4	5	0.5	0.25	7.23
	5	8	8	4	2	0.333333	9.41
	4	6	4	4	1	0.2	8.25
	0.25	0.5	0.5	3	2	0.333333	4.85
STREAMING CLASS	0.5	1	0.25	2	1	0.5	2.63
	0.5	**1**	**0.25**	**0.5**	**0.5**	**1**	**9.67**
	0.333333	0.5	0.142857	3	0.25	0.25	6.93
	2	4	9	3	9	7	8.77
	1	4	2	3	6	1	5.82
	0.25	0.5	0.142857	2	0.2	0.25	8.46
	2	3	6	2	7	7	8.65
	1	**1**	**0.5**	**0.5**	**1**	**0.5**	**6.94**
	1	2	5	2	9	5	2.35
	1	2	5	2	8	9	7.44
	0.5	1	0.25	2	2	0.5	8.18
	0.5	1	0.5	3	4	0.5	9.98
	4	6	4	4	2	0.25	9.71
	2	1	8	0.5	8	4	8.38
	2	**1**	**8**	**2**	**9**	**4**	**8.24**
	2	3	7	3	8	6	8.62
	1	1	7	1	2	2	9.23
	5	7	6	4	2	0.333333	9.57
	2	4	9	3	9	7	8.77
	0.2	0.5	0.125	2	0.25	0.142857	4.67
	7	9	4	4	0.5	0.25	8.10
	2	1	3	0.5	2	2	1.86
	0.5	**2**	**4**	**1**	**7**	**7**	**8.57**
	2	1	3	0.5	1	8	9.93
	1	2	2	2	2	2	2.51
	5	8	4	5	2	0.333333	9.81
	1	2	2	2	5	1	4.84
	0.5	1	0.25	2	0.5	0.5	2.57
	0.5	**2**	**8**	**1**	**6**	**9**	**9.86**
	2	4	9	3	8	1	7.46
	2	4	9	3	7	5	5.16
	6	8	4	4	0.5	0.25	8.19

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	0.5	4	3	3	3	0.5	6.24
	2	**1**	**6**	**1**	**2**	**5**	**3.83**
	5	8	7	5	2	0.333333	9.86
	0.5	2	2	2	6	3	4.38
	1	4	6	3	6	3	2.32
	2	2	5	1	3	8	6.74
	0.125	**0.111111**	**0.2**	**1**	**1**	**2**	**1.70**
	5	8	4	4	1	0.166667	7.84
	0.5	3	2	3	7	3	7.64
	0.5	1	3	2	6	3	0.00
INTERACTIVE CLASS	0.166667	0.25	0.5	2	2	0.5	6.47
	0.2	0.5	0.5	3	2	0.5	1.55
	2	4	9	3	9	7	8.77
	0.5	1	0.25	3	0.5	0.5	4.72
	0.166667	**0.25**	**0.5**	**1**	**1**	**0.5**	**9.78**
	0.25	0.333333	0.142857	1	0.166667	0.25	8.32
	0.166667	0.333333	0.5	2	2	0.333333	8.82
	0.333333	0.5	0.5	2	2	0.333333	6.89
	0.25	0.333333	0.5	1	0.5	0.5	8.38
	0.333333	**0.5**	**1**	**3**	**2**	**0.5**	**6.55**
	0.111111	**0.111111**	**0.111111**	**1**	**1**	**3**	**6.96**
	0.2	0.5	0.5	2	1	0.5	3.57
	0.2	0.5	0.5	2	2	0.333333	6.08
	0.25	0.5	0.5	3	2	0.5	1.88
	0.5	1	1	2	2	1	0.00
	0.333333	0.5	0.5	3	2	0.5	3.09
	0.333333	**0.5**	**1**	**3**	**1**	**0.5**	**9.60**
	0.333333	0.5	0.5	3	1	0.5	3.59
	0.25	0.5	0.5	3	1	0.2	9.62
	0.25	0.333333	0.142857	1	0.166667	0.25	8.32
	1	1	4	1	4	4	0.00
	0.166667	0.333333	0.5	3	2	0.333333	7.69
	0.111111	**1**	**0.333333**	**4**	**3**	**1**	**5.58**
	0.25	0.333333	0.5	2	1	0.5	4.84
	0.2	0.5	0.5	2	2	0.25	9.96
	0.166667	0.333333	0.5	2	2	0.333333	8.82
	0.333333	**0.5**	**1**	**1**	**1**	**0.5**	**9.11**
	0.2	0.333333	0.5	2	1	0.333333	8.52
	0.2	0.333333	0.5	3	1	0.333333	9.37
	0.333333	0.5	0.5	3	2	0.5	3.09
	0.25	0.5	0.5	3	1	0.5	3.54

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	0.333333	0.5	0.5	1	0.5	0.333333	7.05
	0.333333	0.25	0.125	1	0.25	0.166667	6.75
	0.111111	**0.166667**	**0.142857**	**1**	**1**	**3**	**8.13**
	0.333333	0.5	0.5	2	0.5	0.5	6.04
	0.166667	0.25	0.5	2	1	0.5	8.68
	0.2	0.333333	0.5	3	1	0.5	7.13
	0.166667	0.2	0.125	2	0.25	0.25	9.92
	0.166667	0.333333	0.5	2	2	0.5	4.38
	0.25	0.5	0.5	3	1	0.5	3.54
	0.25	0.25	0.5	3	2	0.5	8.85
	0.25	0.5	0.5	3	1	0.25	7.08
BACKGROUND CLASS	0.166667	0.2	0.125	1	0.25	0.2	9.22
	0.333333	**0.2**	**0.5**	**1**	**1**	**0.5**	**9.39**
	2	4	9	3	9	7	8.77
	0.333333	0.5	0.142857	3	0.25	0.25	6.93
	0.5	1	0.25	3	1	0.5	3.12
	1	3	8	2	7	4	1.09
	0.166667	0.2	0.125	1	0.25	0.25	7.21
	0.333333	0.5	0.142857	3	0.25	0.142857	5.64
	0.111111	**0.25**	**0.5**	**1**	**2**	**1**	**6.58**
	0.166667	**0.333333**	**0.5**	**1**	**2**	**0.5**	**8.96**
	0.25	0.5	0.125	2	0.166667	0.166667	8.67
	0.166667	0.25	0.125	1	0.25	0.2	7.84
	0.333333	0.5	0.125	3	0.25	0.166667	4.45
	0.2	0.5	0.125	1	0.25	0.125	8.47
	0.2	**0.5**	**0.5**	**1**	**2**	**0.5**	**7.87**
	0.25	0.5	0.125	2	0.2	0.125	5.91
	0.2	**0.333333**	**0.5**	**1**	**2**	**0.5**	**8.14**
	0.2	0.5	0.125	1	0.2	0.2	7.20
	0.5	3	3	2	5	3	6.58
	0.111111	0.111111	0.2	1	2	4	3.43
	0.25	0.5	0.125	1	0.25	0.166667	3.92
	0.166667	0.25	0.125	1	0.25	0.2	7.84
	0.166667	0.5	0.142857	2	0.25	0.166667	7.54
	0.111111	**0.333333**	**0.111111**	**1**	**1**	**0.5**	**4.84**
	0.5	1	2	2	4	2	0.00
	0.333333	0.5	0.125	1	0.166667	0.166667	3.89
	0.2	0.333333	0.125	2	0.25	0.25	5.94
	2	2	8	1	4	4	0.00
	0.333333	0.5	0.125	1	0.125	0.166667	7.36
	0.166667	0.2	0.125	1	0.25	0.2	9.22

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	0.25	**0.5**	**0.5**	**1**	**2**	**0.5**	**6.89**
	0.2	0.25	0.125	1	0.166667	0.25	9.81
	0.25	0.5	0.125	1	0.142857	0.142857	9.41
	0.2	0.25	0.125	1	0.25	0.2	6.43
	0.333333	0.5	0.125	3	0.166667	0.125	9.68
	0.25	0.25	0.125	1	0.25	0.25	3.49
	0.25	0.5	0.125	1	0.166667	0.166667	6.84
	0.2	0.25	0.125	1	0.25	0.2	6.43
	0.166667	0.25	0.125	1	0.25	0.25	6.36
	0.333333	0.333333	0.125	1	0.25	0.166667	4.15
	0.25	**0.25**	**0.5**	**1**	**1**	**0.5**	**7.39**
	0.25	0.5	0.142857	1	0.166667	0.25	8.55

Table D.2: User Preferences with feedback mechanism for four sub-criteria wrt Figure 6.2

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
CONVERSATIONAL CLASS	4	6	4	4	1	0.333333	5.19
	4	7	4	4	1	0.333333	3.68
	5	9	5	4	2	0.333333	6.28
	6	9	4	5	1	0.333333	8.09
	7	9	4	4	0.5	0.25	8.10
	5	9	4	4	0.5	0.25	5.76
	4	9	7	4	2	0.333333	4.91
	8	9	4	4	0.5	0.25	9.76
	2	4	8	2	4	2	0.00
	4	8	4	4	2	0.25	6.48
	2	1	1	0.5	0.5	1	0.00
	4	9	5	5	1	0.333333	4.05
	4	9	5	4	1	0.333333	2.62
	5	9	4	7	1	0.333333	9.73
	0.125	**1**	**0.5**	**7**	**6**	**1**	**1.71**
	5	9	4	7	1	0.2	9.85
	7	9	6	4	1	0.333333	8.19
	5	9	6	6	1	0.25	9.87
	5	9	4	4	1	0.25	3.98
	0.25	0.5	0.5	2	2	0.5	2.32
	0.333333	0.5	0.5	3	1	0.5	3.59
	4	8	4	4	1	0.333333	2.57
	5	9	5	4	1	0.25	5.06
	4	8	4	4	1	0.2	4.84
	2	**4**	**9**	**1**	**4**	**1**	**8.45**
	5	6	5	4	1	0.333333	8.64
	5	7	4	4	1	0.2	8.14
	0.333333	0.5	0.5	3	2	0.333333	6.25
	6	8	4	4	1	0.333333	6.75
	4	7	4	4	2	0.333333	5.93
	4	6	4	4	0.5	0.25	9.71
	4	7	5	4	1	0.333333	4.97
	4	8	4	4	1	0.333333	2.57
	0.5	1	0.25	3	0.5	0.5	4.72
	4	6	4	4	0.5	0.333333	9.02
	2	2	4	2	4	3	3.67
	4	8	5	4	2	0.333333	4.71
	0.25	0.5	0.5	3	1	0.5	3.54
	4	9	4	5	0.5	0.25	7.23
	7	9	6	4	1	0.333333	8.19
	7	9	4	4	1	0.333333	8.01
	0.25	0.5	0.5	3	2	0.333333	4.85
	0.5	1	0.25	2	1	0.5	2.63

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	0.5	1	0.25	3	1	0.5	3.12
	2	2	9	1	4	7	1.49
	2	4	9	3	9	7	8.77
	1	4	2	3	6	1	5.82
	0.5	4	7	3	8	2	4.46
	2	3	6	2	7	7	8.65
	2	2	8	1	4	3	0.48
	1	2	5	2	9	5	2.35
	0.5	1	6	2	3	5	9.39
	2	1	7	0.5	4	2	9.64
	0.5	1	0.5	3	4	0.5	9.98
	0.5	1	1	3	2	1	0.93
	2	1	8	0.5	8	4	8.38
	1	**1**	**7**	**0.333333**	**3**	**5**	**5.71**
	2	2	9	1	4	3	0.80
	1	1	7	1	2	2	9.23
	4	6	4	4	1	0.333333	5.19
	7	9	6	4	1	0.25	9.80
	0.2	0.5	0.125	2	0.25	0.142857	4.67
	7	9	4	4	0.5	0.25	8.10
	2	1	3	0.5	2	2	1.86
	2	1	4	0.333333	1	3	1.81
	2	1	3	0.5	1	8	9.93
	1	2	2	2	2	2	2.51
	0.5	2	7	3	8	2	2.57
	1	2	2	2	5	1	4.84
	0.5	1	0.25	2	0.5	0.5	2.57
	2	**4**	**8**	**1**	**2**	**1**	**6.18**
	2	4	9	3	8	1	7.46
	0.5	4	7	3	9	1	8.88
	6	8	4	4	0.5	0.25	8.19
	0.5	4	3	3	3	0.5	6.24
	1	1	7	1	4	3	3.16
	5	8	7	5	2	0.333333	9.86
	0.5	1	0.5	3	2	0.5	1.97
	2	2	8	1	3	6	2.05
	2	2	5	1	3	8	6.74
	2	1	4	0.25	3	8	4.13
	1	1	7	1	4	7	1.77
	0.5	4	7	3	5	2	6.95
	0.111111	0.111111	0.111111	0.5	0.5	1	2.91
Interac.	0.166667	0.25	0.5	2	2	0.5	6.47
	0.2	0.5	0.5	3	2	0.5	1.55
	0.166667	0.25	0.5	2	2	0.5	6.47
	0.5	1	0.25	3	0.5	0.5	4.72
	0.2	0.25	0.5	2	2	0.5	6.63
	0.333333	***0.5***	***0.5***	***1***	***2***	***0.5***	***6.47***

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	0.166667	0.333333	0.5	2	2	0.333333	8.82
	0.166667	0.25	0.125	1	0.25	0.2	7.84
	0.25	0.333333	0.5	1	0.5	0.5	8.38
	0.333333	0.25	0.125	1	0.25	0.166667	6.75
	0.333333	0.5	0.5	3	2	0.5	3.09
	0.2	0.5	0.5	2	1	0.5	3.57
	0.25	0.333333	0.5	1	0.5	0.5	8.38
	0.166667	0.5	0.5	2	2	0.5	3.11
	0.5	1	1	2	2	1	0.00
	0.25	0.5	0.5	2	2	0.333333	6.11
	0.333333	***0.5***	***1***	***3***	***1***	***0.5***	***9.60***
	0.333333	0.5	0.5	3	1	0.5	3.59
	0.25	0.5	0.5	3	1	0.2	9.62
	0.25	***0.333333***	***0.5***	***1***	***1***	***0.5***	***5.57***
	1	1	4	1	4	4	0.00
	0.166667	0.333333	0.5	3	2	0.333333	7.69
	0.166667	0.333333	0.5	2	2	0.5	4.38
	0.2	0.5	0.5	2	2	0.25	9.96
	0.2	0.5	0.5	2	2	0.25	9.96
	0.166667	0.333333	0.5	2	2	0.333333	8.82
	0.2	0.25	0.125	1	0.25	0.166667	8.53
	0.2	0.333333	0.5	2	1	0.333333	8.52
	0.2	0.333333	0.5	3	1	0.333333	9.37
	0.333333	0.5	0.5	3	2	0.5	3.09
	0.333333	0.333333	0.5	0.5	0.5	0.5	8.93
	0.333333	0.5	0.5	1	0.5	0.333333	7.05
	0.166667	0.5	0.5	3	2	0.5	1.74
	0.2	0.5	0.5	3	1	0.5	4.18
	0.333333	0.5	0.5	2	0.5	0.5	6.04
	0.166667	0.25	0.5	2	1	0.5	8.68
	0.333333	0.5	0.5	2	1	0.5	1.86
	0.25	0.5	0.5	3	1	0.5	3.54
	0.166667	0.333333	0.5	2	2	0.5	4.38
	0.25	0.5	0.5	3	1	0.5	3.54
	0.2	0.333333	0.5	2	2	0.5	4.23
	0.25	0.5	0.5	3	1	0.25	7.08
Background	0.166667	0.25	0.125	1	0.25	0.25	6.36
	0.333333	0.333333	0.125	1	0.2	0.2	3.17
	0.25	0.333333	0.142857	1	0.25	0.25	3.68
	0.333333	0.5	0.142857	3	0.25	0.25	6.93
	4	6	4	4	1	0.333333	5.19
	0.166667	0.5	0.5	2	2	0.5	3.11
	0.166667	0.2	0.125	1	0.25	0.25	7.21
	0.333333	0.5	0.142857	2	0.25	0.25	3.28
	0.2	0.2	0.125	1	0.25	0.25	5.92
	0.111111	***0.333333***	***0.111111***	***1***	***1***	***0.5***	***4.84***
	0.25	0.333333	0.125	1	0.2	0.2	5.04

	PN-PNGA	PN-CPN	PN-FPN	PNGA-CPN	PNGA-FPN	CPN-FPN	Consistency
	0.166667	0.25	0.125	1	0.25	0.2	7.84
	0.333333	0.5	0.125	3	0.25	0.166667	4.45
	0.2	0.5	0.125	1	0.25	0.125	8.47
	0.2	***0.333333***	***0.5***	***1***	***2***	***0.5***	***8.14***
	0.25	0.5	0.125	2	0.2	0.125	5.91
	0.166667	0.5	0.125	1	0.25	0.25	7.28
	0.2	0.5	0.125	1	0.2	0.2	7.20
	0.166667	0.333333	0.125	1	0.25	0.2	6.91
	0.25	0.333333	0.125	1	0.25	0.2	3.64
	0.25	0.5	0.125	1	0.25	0.166667	3.92
	0.333333	0.25	0.142857	1	0.2	0.25	4.97
	0.166667	0.5	0.142857	2	0.25	0.166667	7.54
	0.2	0.333333	0.125	1	0.25	0.25	4.26
	0.5	1	2	2	4	2	0.00
	0.333333	0.5	0.125	1	0.166667	0.166667	3.89
	0.2	0.333333	0.125	2	0.25	0.25	5.94
	2	2	8	1	4	4	0.00
	0.333333	0.5	0.125	1	0.125	0.166667	7.36
	0.166667	0.2	0.125	1	0.25	0.2	9.22
	0.25	**0.5**	**0.5**	**1**	**2**	**0.5**	**6.89**
	0.2	0.25	0.125	1	0.166667	0.25	9.81
	0.25	0.5	0.125	1	0.142857	0.142857	9.41
	0.2	0.25	0.125	1	0.25	0.2	6.43
	0.25	0.5	0.125	1	0.25	0.166667	3.92
	0.333333	0.333333	0.125	1	0.25	0.25	1.24
	0.25	0.5	0.125	1	0.166667	0.166667	6.84
	0.2	0.25	0.125	1	0.25	0.25	4.84
	0.2	0.333333	0.5	3	2	0.5	4.36
	0.333333	0.333333	0.125	1	0.25	0.166667	4.15
	0.166667	0.333333	0.5	3	2	0.5	4.07
	0.166667	0.5	0.125	2	0.25	0.2	6.40

Table D.3: User Preferences without feedback mechanism for five criteria wrt Figure 6.2

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
CONVERSATIONAL CLASS	0.25	0.5	2	0.333333	8	9	3	5	0.5	0.166667	9.62
	0.166667	***0.5***	***2***	***1***	***3***	***6***	***2***	***2***	***0.333333***	***0.333333***	***6.61***
	0.333333	0.5	0.5	0.111111	3	2	0.25	0.333333	0.111111	0.333333	4.52
	4	0.25	0.5	0.125	0.333333	0.111111	0.111111	0.5	0.333333	0.333333	9.18
	0.25	***0.5***	***0.5***	***0.111111***	***3***	***0.5***	***0.5***	***1***	***0.166667***	***0.166667***	***9.13***
	0.166667	0.333333	1	0.25	5	9	6	5	1	0.25	8.83
	0.166667	0.500001	1.000002	0.166667	3	6	1.000002	2	0.333334	0.166667	0.00
	0.2	***1***	***0.5***	***0.333333***	***6***	***2***	***1***	***1***	***0.25***	***0.333333***	***3.06***
	3	1	3	9	0.25	2	9	6	9	6	8.67
	4	***0.166667***	***0.5***	***0.25***	***0.111111***	***1***	***0.25***	***6***	***2***	***0.25***	***8.38***
	0.5	0.111111	0.5	4	0.142857	0.5	4	4	9	7	8.21
	0.2	1	2	0.5	6	5	3	2	0.5	0.333333	2.49
	0.25	0.25	1	0.166667	3	6	2	2	1	0.2	4.70
	0.333333	1	0.5	0.125	5	2	0.5	0.5	0.125	0.25	0.48
	0.166667	***0.5***	***1***	***1***	***4***	***9***	***2***	***7***	***1***	***0.2***	***8.71***
	0.2	1	1	0.25	3	3	1	1	0.25	0.25	0.77
	0.2	0.333333	0.5	0.25	4	9	2	3	0.5	0.25	5.31
	0.25	***1***	***0.5***	***0.2***	***5***	***1***	***1***	***1***	***0.2***	***0.2***	***6.58***
	0.2	1	1	0.166667	5	5	0.833335	1	0.166667	0.166667	0.00
	0.166667	1	0.5	0.333333	9	4	4	0.5	0.5	0.2	8.84
	6	0.5	2	9	0.125	0.5	2	4	9	5	2.48
	0.25	***0.111111***	***0.5***	***0.5***	***0.2***	***1***	***5***	***4***	***8***	***1***	***9.11***
	2	0.2	0.333333	0.142857	0.25	0.111111	0.111111	2	1	0.5	5.30
	0.2	1	0.5	0.125	5	3	1	0.166667	0.125	0.25	5.45
	0.25	1	2	0.5	9	8	1	2	0.5	0.333333	7.70
	0.166667	1	2	0.333333	7	9	1	2	0.333333	0.166667	2.35
	0.166667	***0.5***	***2***	***1***	***4***	***9***	***2***	***8***	***0.5***	***0.2***	***8.47***
	0.166667	1	1	0.333333	6	6	2	1	0.333333	0.333333	0.00
	0.25	0.25	0.5	0.111111	5	2	0.333333	2	0.333333	0.166667	9.10
	0.166667	1	0.5	0.166667	7	4	1	0.2	0.125	0.333333	3.02
	0.2	***1***	***1***	***1***	***7***	***7***	***6***	***1***	***1***	***0.25***	***6.25***
	0.333333	***1***	***2***	***3***	***4***	***8***	***2***	***4***	***1***	***0.25***	***8.00***

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	2	***0.125***	***0.5***	***0.125***	***0.142857***	***1***	***0.25***	***5***	***1***	***0.25***	***5.60***
	0.333333	1	0.5	0.111111	3	7	1	0.5	0.166667	0.333333	8.20
	1	0.125	0.5	0.25	0.125	0.5	0.25	4	2	0.5	0.00
	0.25	0.111111	0.5	0.25	0.25	2	3	9	6	0.5	6.14
	0.2	***1***	***0.5***	***0.333333***	***4***	***4***	***1***	***1***	***0.2***	***0.166667***	***3.79***
	0.333333	***0.333333***	***2***	***0.333333***	***4***	***9***	***1***	***1***	***0.333333***	***0.2***	***8.01***
	1	0.2	0.857142	0.111111	0.2	0.5	0.111111	6	1.2	0.5	5.44
	0.2	0.5	2	0.5	6	9	7	2	0.5	0.333333	7.27
	0.25	1	2	1	6	9	5	6	1	0.25	6.35
	0.25	0.142857	0.5	0.2	0.166667	1	0.25	6	2	0.5	5.65
STREAMING CLASS	0.2	0.111111	1	2	0.2	2	4	6	9	2	6.98
	0.25	0.111111	1	0.5	0.2	5	3	6	3	0.5	9.95
	0.25	***0.2***	***1***	***2***	***0.333333***	***2***	***1***	***8***	***6***	***2***	***9.69***
	0.333333	0.166667	0.5	0.5	0.333333	1	3	3	5	5	5.76
	0.2	0.111111	0.5	2	0.333333	1	4	6	9	4	5.31
	0.25	***0.125***	***0.5***	***0.25***	***0.2***	***1***	***1***	***6***	***3***	***0.5***	***2.87***
	6	***0.5***	***1***	***6***	***0.333333***	***0.166667***	***1***	***0.333333***	***3***	***9***	***7.27***
	0.25	0.142857	0.5	0.2	0.166667	1	0.25	6	2	0.5	5.65
	2	0.5	3	7	0.166667	0.5	4	7	9	5	8.39
	0.2	0.111111	1	4	0.333333	2	8	5	9	5	9.11
	0.166667	0.5	0.5	0.25	5	5	4	1	0.333333	0.25	6.80
	0.5	0.166667	2	2	0.333333	2	9	8	5	3	9.56
	0.333333	0.111111	1	2	0.125	2	4	9	9	2	8.71
	0.333333	0.111111	0.5	3	0.25	0.5	3	1	9	5	9.01
	0.5	0.111111	0.5	2	0.25	0.5	2	7	9	5	8.96
	2	***0.25***	***0.25***	***0.5***	***0.125***	***0.125***	***1***	***1***	***2***	***2***	***6.72***
	0.25	0.111111	0.5	0.25	0.25	1	0.5	4	2	0.5	1.40
	0.5	0.125	0.5	3	0.111111	0.5	3	5	9	5	7.85
	0.5	***0.166666***	***0.5***	***1.333332***	***0.25***	***2.666664***	***2***	***8***	***7***	***1***	***3.80***
	0.25	0.111111	0.5	0.111111	0.142857	0.5	0.111111	6	1	0.5	9.31
	0.2	1	0.5	0.166667	5	3	1	0.5	0.166667	0.333333	0.16
	0.333333	0.111111	0.5	0.5	0.142857	1	1	6	6	1	1.99
	0.333333	***0.142857***	***1***	***0.25***	***0.333333***	***2***	***0.5***	***8***	***2***	***1***	***6.45***

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	0.5	0.111111	0.5	1	0.166667	0.5	6	5	9	4	7.75
	0.5	0.125	0.5	0.5	0.25	1	1	4	4	1	0.00
	0.333333	***0.2***	***0.5***	***1***	***0.333333***	***1***	***3***	***1***	***9***	***7***	***5.46***
	1	0.2	1	0.5	0.2	1	0.5	5	2.5	0.5	0.00
	0.2	0.2	1	4	0.333333	3	9	8	9	4	8.39
	0.5	0.166667	1	1	0.333333	2	2	6	6	1	0.00
	0.5	***0.111111***	***1***	***3***	***0.2***	***1***	***4***	***7***	***9***	***3***	***5.32***
	0.25	***0.111111***	***0.5***	***1***	***0.142857***	***0.5***	***1***	***7***	***9***	***3***	***8.38***
	0.333333	0.111111	0.5	1	0.2	1	3	5	9	3	1.66
	0.2	0.111111	0.5	0.5	0.125	0.5	0.5	6	6	1	9.18
	0.5	0.166667	1	1	0.333333	2	2	6	6	1	0.00
	1	0.142857	2	0.5	0.2	2	0.333333	8	3	0.333333	1.91
	5	0.5	2	9	0.166667	1	4	8	9	4	6.90
	0.2	0.166667	0.5	0.25	0.2	1	0.5	6	2	0.5	5.49
	2	0.25	0.5	0.142857	0.125	0.25	0.111111	2	0.333333	0.5	4.16
	0.5	***0.25***	***0.5***	***0.5***	***0.25***	***1***	***1***	***2***	***8***	***3***	***7.52***
	1	0.25	0.25	0.111111	0.25	0.25	0.111111	1	0.25	0.25	2.17
	3	0.2	1.2	0.6	0.142857	1.2	0.25	6	6	0.5	7.27
	6	2	0.333333	1	0.333333	0.111111	0.333333	0.2	0.2	3	6.10
INTERACTIVE CLASS	2	0.125	1	0.333333	0.125	0.333333	0.5	9	3	0.333333	7.12
	0.2	***1***	***1***	***0.5***	***6***	***9***	***4***	***5***	***1***	***0.2***	***7.89***
	6	***2***	***0.2***	***1***	***0.333333***	***0.111111***	***1***	***0.111111***	***1***	***7***	***8.57***
	0.333333	0.111111	1	4	0.333333	3	9	8	9	4	7.33
	1	0.111111	0.25	0.25	0.142857	0.5	0.25	7	2	0.5	4.86
	1	0.2	2	0.5	0.2	2	0.166667	8	3	0.333333	5.68
	3	0.2	0.5	0.5	0.111111	0.5	0.25	4	2	0.5	2.34
	1	***0.111111***	***2***	***0.25***	***0.111111***	***1***	***0.5***	***4***	***2***	***0.5***	***8.81***
	0.25	1	1	0.333333	5	5	1	1	0.333333	0.333333	0.89
	3	***0.111111***	***1***	***0.5***	***0.125***	***2***	***0.5***	***8***	***5***	***0.5***	***7.76***
	0.25	0.111111	1	3	0.333333	2	9	4	9	3	6.22
	2	***0.111111***	***1***	***0.166667***	***0.111111***	***2***	***0.2***	***8***	***3***	***0.25***	***6.90***
	2	0.25	2	2	0.125	1	0.25	8	2	0.25	5.46
	1	0.111111	0.5	0.25	0.111111	0.142857	0.25	5	3	0.5	6.06

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	3	***0.25***	***1***	***2***	***0.111111***	***1***	***0.333333***	***7***	***3***	***0.333333***	***5.94***
	3	0.111111	0.333333	0.25	0.125	0.5	0.25	3	2	0.5	6.28
	4	0.2	0.333333	0.166667	0.2	0.111111	0.125	0.333333	0.5	0.5	9.36
	2	0.25	2	0.5	0.125	1	0.25	8	2	0.25	0.00
	4	0.25	2	1	0.2	0.5	0.25	4	2	0.5	3.63
	1	***0.111111***	***2***	***0.25***	***0.2***	***1***	***0.333333***	***8***	***2***	***0.25***	***3.35***
	1	0.142857	1	0.5	0.142857	1	0.5	8	4	0.5	0.13
	4	***0.166667***	***0.5***	***0.25***	***0.333333***	***0.333333***	***0.142857***	***1***	***1***	***0.5***	***9.60***
	0.333333	1	1	0.333333	4	8	2	1	0.333333	0.333333	2.85
	3	0.2	0.2	0.125	0.25	0.111111	0.111111	0.5	0.25	0.5	6.29
	2	0.25	2	1	0.333333	5	0.5	4	2	0.5	9.42
	0.166667	1	2	0.25	9	9	1	2	0.25	0.25	4.55
	3	***0.2***	***0.5***	***0.333333***	***0.142857***	***0.2***	***0.111111***	***1***	***2***	***0.333333***	***6.59***
	3	***0.25***	***0.5***	***0.333333***	***0.142857***	***1***	***0.125***	***8***	***1***	***0.25***	***7.78***
	4	0.25	2	0.2	0.25	0.5	0.125	2	0.5	0.25	7.45
	3	0.125	2	0.5	0.142857	0.333333	0.125	3	1	0.25	9.33
	1	0.125	0.333333	0.111111	0.111111	0.142857	0.2	3	1	0.25	6.23
	1	***0.111111***	***2***	***0.125***	***0.111111***	***0.5***	***0.125***	***3***	***1***	***0.25***	***9.51***
	4	0.2	2	1	0.166667	3	0.333333	8	4	0.5	8.88
	1	0.166667	1	0.5	0.333333	1	0.5	8	4	0.5	3.60
	0.5	0.111111	0.5	3	0.111111	0.5	3	3	9	6	7.08
	2	***0.111111***	***1***	***2***	***0.166667***	***1***	***2***	***7***	***8***	***0.5***	***9.06***
	3	0.25	1	0.25	0.142857	0.5	0.125	4	1	0.25	0.76
	4	0.25	0.25	0.142857	0.25	0.25	0.111111	1	0.25	0.25	8.93
	2	0.166667	2	0.5	0.111111	0.25	0.333333	8	4	0.333333	7.30
	2	0.25	1	0.25	0.125	0.5	0.125	4	1	0.25	0.00
	2	***0.166667***	***1***	***0.5***	***0.333333***	***2***	***0.2***	***7***	***1***	***0.5***	***9.43***
	0.25	***1***	***1***	***0.5***	***3***	***3***	***1***	***3***	***0.333333***	***0.333333***	***4.70***
	2	0.5	2	5	0.25	1	9	3	9	3	5.93
	4	1	2	9	0.333333	4	2	5	9	2	8.87
	3	***0.5***	***2***	***5***	***0.2***	***1***	***1***	***9***	***5***	***3***	***8.01***
	6	1	2	9	0.166667	1	3	6	9	7	7.97
	3	0.333333	0.5	4	0.111111	0.2	2	2	9	8	1.62

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	2	0.5	0.5	6	0.333333	0.333333	1	2	4	5	6.18
	5	0.5	1	8	0.333333	0.5	4	3	9	8	5.83
	2	0.25	0.25	0.5	0.125	0.125	0.25	1	2	2	0.00
	0.166667	0.5	0.5	0.25	5	5	4	1	0.333333	0.25	6.80
	6	***2***	***0.25***	***2***	***0.25***	***0.111111***	***1***	***0.142857***	***1***	***7***	***6.26***
	1	0.25	1	0.25	0.25	1	0.25	8	0.5	0.25	5.32
	3	0.5	1	7	0.142857	0.125	3	3	9	8	8.05
	3	0.333333	0.166667	0.5	0.111111	0.111111	0.5	0.333333	1	7	7.59
	1	0.166667	0.333333	0.166667	0.166667	0.2	0.166667	2	3	0.5	5.87
	6	0.333333	0.166667	0.142857	0.111111	0.111111	0.111111	0.5	0.5	1	7.63
	5	0.333333	0.166667	0.5	0.166667	0.111111	0.5	0.5	2	4	5.77
	0.2	***0.5***	***1***	***0.111111***	***5***	***2***	***0.5***	***0.5***	***0.111111***	***0.25***	***4.83***
	2	***0.333333***	***1***	***9***	***0.333333***	***2***	***4***	***2***	***9***	***3***	***7.33***
	6	0.5	0.166667	2	0.111111	0.111111	1	0.5	3	5	8.50
	3	0.5	0.5	4	0.111111	0.111111	1	1	8	8	0.51
	3	0.333333	0.666666	0.5	0.111111	0.222222	0.666666	2	9	2	9.30
	0.25	0.333333	1	0.333333	3	9	3	2	0.25	0.333333	8.21
	4	0.5	1	7	0.142857	0.25	0.5	2	6	3	4.08
	5	***1***	***0.333333***	***5***	***0.25***	***0.111111***	***0.333333***	***0.333333***	***1***	***3***	***7.79***
	0.25	0.5	2	1	4	9	2	6	2	0.166667	6.90
	2	0.25	0.333333	1	0.111111	0.142857	0.166667	3	6	2	6.67
	4	0.5	2	8	0.142857	0.5	4	8	9	4	7.75
	4	***0.5***	***1***	***2***	***0.125***	***0.111111***	***0.5***	***0.5***	***3***	***7***	***4.90***
	6	0.333333	2	9	0.125	0.5	4	5	9	7	9.45
	2	0.333333	0.333333	7	0.166667	0.142857	4	1	8	8	6.15
	5	0.25	2	2	0.25	0.333333	1	2	6	3	8.26
	1	***0.2***	***1***	***0.25***	***0.2***	***5***	***0.25***	***5***	***1***	***0.25***	***9.37***
	0.2	0.111111	0.5	3	0.2	1	4	6	9	3	9.06
	4	0.5	2	9	0.125	0.5	4	4	9	7	5.22
	6	0.5	3	6	0.125	0.5	0.25	6	9	2	8.70
	0.333333	0.111111	0.5	2	0.142857	0.5	2	6	9	5	7.88
	2	***0.25***	***1***	***0.25***	***0.25***	***1***	***0.166667***	***5***	***0.333333***	***0.25***	***6.42***
	5	1	0.2	0.5	0.333333	0.111111	0.333333	0.2	0.5	6	7.30
	0.25	0.111111	0.5	2	0.166667	1	3	5	9	3	5.52
	4	1	0.5	1	0.25	0.125	0.25	0.5	1	2	0.00

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	4	0.333333	3	9	0.142857	0.5	4	9	9	4	9.93
	0.5	0.125	0.5	1	0.142857	3	3	5	9	2	8.03

Table D.4: User Preferences with feedback mechanism for five criteria wrt Figure 6.2

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
CONVERSATIONAL CLASS	0.2	1	0.5	0.111111	4	2	0.5	0.5	0.111111	0.2	0.20
	0.166667	1	1	0.2	7	7	1	1	0.2	0.2	0.44
	0.333333	0.5	0.5	0.111111	3	2	0.25	0.333333	0.111111	0.333333	4.52
	0.2	1	1	0.333333	5	5	2	1	0.333333	0.333333	0.14
	0.166667	1	1	0.166667	5	5	1	1	0.166667	0.166667	0.16
	0.166667	0.333333	1	0.25	5	9	6	5	1	0.25	8.83
	0.333333	1	1	0.25	3	3	0.333333	1	0.125	0.333333	4.55
	0.25	1	0.5	0.111111	3	2	0.333333	0.5	0.111111	0.166667	0.56
	3	1	3	9	0.25	2	9	6	9	6	8.67
	0.2	1	1	0.333333	5	5	2	1	0.333333	0.333333	0.14
	0.2	0.5	1	0.2	3	9	1	4	0.5	0.2	1.86
	0.2	0.5	1	0.111111	3	9	1	3	0.333333	0.25	2.79
	0.25	0.5	0.500001	0.166667	6	3	0.999999	0.333333	0.166667	0.333333	3.97
	0.333333	1	0.5	0.125	5	2	0.5	0.5	0.125	0.25	0.48
	3	0.2	0.5	0.111111	0.125	0.5	0.111111	5	1	0.333333	4.77
	0.333333	1	2	0.333333	3	6	1	2	0.333333	0.166667	0.00
	2	0.25	2	0.5	0.125	1	0.25	9	2	0.25	0.06
	0.166667	1	0.5	0.333333	9	4	4	0.5	0.5	0.2	8.84
	3	0.25	1	0.25	0.142857	0.5	0.125	4	1	0.25	0.76
	2	0.25	2	1	0.111111	2	0.333333	6	2	0.333333	4.20
	2	0.2	0.333333	0.142857	0.25	0.111111	0.111111	2	1	0.5	5.30
	0.2	1	0.5	0.125	5	3	1	0.166667	0.125	0.25	5.45
	0.25	1	2	0.5	9	8	1	2	0.5	0.333333	7.70
	0.166667	1	2	0.333333	7	9	1	2	0.333333	0.166667	2.35
	0.2	***1***	***1***	***1***	***3***	***6***	***6***	***2***	***2***	***0.333333***	***5.31***
	0.2	0.5	0.5	0.111111	3	3	0.5	1	0.166667	0.166667	0.36
	0.2	0.5	1	0.2	4	8	2	2	0.4	0.2	1.31
	0.166667	1	0.5	0.166667	7	4	1	0.2	0.125	0.333333	3.02
	1	***0.111111***	***1***	***0.166667***	***0.142857***	***0.333333***	***0.25***	***8***	***2***	***0.25***	***5.44***
	0.166667	0.5	1	0.142857	3	9	1	3	0.5	0.2	1.28
	0.25	0.5	1	0.5	4	9	2	6	1	0.166667	6.41

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	0.2	0.111111	0.5	0.5	0.25	1	1	5	6	1	2.69
	0.25	***1***	***1***	***0.5***	***3***	***3***	***1***	***3***	***0.333333***	***0.333333***	***4.69***
	0.2	***0.5***	***2***	***0.333333***	***4***	***9***	***1***	***5***	***1***	***0.2***	***4.40***
	1	0.166667	2	0.5	0.166667	2	0.5	9	3	0.333333	0.44
	0.5	0.166667	1	3	0.333333	3	9	8	9	3	4.11
	0.25	0.5	1	0.25	4	8	2	2	0.5	0.25	1.70
	0.2	0.5	2	0.333333	7	7	2	4	1	0.2	8.04
	1	0.2	0.333333	0.111111	0.2	0.333333	0.111111	2	0.2	0.333333	6.58
	0.25	1	1	0.25	4	4	1	1	0.25	0.25	0.00
	2	***0.111111***	***1***	***2***	***0.166667***	***1***	***2***	***7***	***8***	***0.5***	***9.06***
	0.25	0.142857	0.5	0.2	0.166667	1	0.25	6	2	0.5	5.65
STREAMING CLASS	0.2	0.111111	1	2	0.2	2	4	6	9	2	6.98
	0.25	0.111111	0.5	2	0.25	2	8	6	9	4	5.66
	0.5	0.111111	0.5	0.25	0.2	1	0.25	4	2	0.5	1.33
	0.5	0.125	0.5	1	0.25	1	2	4	8	2	0.00
	2	0.5	3	7	0.166667	0.5	4	7	9	5	8.39
	0.2	0.111111	1	4	0.333333	2	8	5	9	5	9.11
	0.166667	0.5	0.5	0.25	5	5	4	1	0.333333	0.25	6.80
	0.5	0.166667	2	2	0.333333	2	9	8	5	3	9.56
	0.25	1	0.5	0.111111	4	2	0.25	0.5	0.111111	0.25	2.11
	0.5	0.142857	1	3	0.333333	2	8	5	9	4	3.45
	0.333333	0.111111	1	2	0.125	2	4	9	9	2	8.71
	0.5	0.111111	0.5	0.5	0.166667	1	1	9	9	1	2.81
	0.5	0.111111	0.5	2	0.25	0.5	2	7	9	5	8.96
	2	***0.25***	***0.25***	***0.5***	***0.125***	***0.125***	***1***	***1***	***2***	***2***	***6.72***
	1	0.111111	0.5	0.2	0.111111	0.5	0.125	4	1	0.25	0.90
	0.5	0.125	0.5	3	0.111111	0.5	3	5	9	5	7.85
	0.333333	0.111111	0.5	3	0.142857	0.5	3	4	9	6	8.06
	0.25	0.111111	0.5	0.111111	0.142857	0.5	0.111111	6	1	0.5	9.31
	0.25	0.111111	0.5	1	0.25	1	2	4	8	2	1.36
	0.5	0.125	1	1	0.25	2	2	8	8	1	0.00
	0.25	0.111111	0.5	1	0.25	1	2	6	9	2	1.76

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	0.25	0.142857	0.5	0.2	0.166667	1	0.25	6	2	0.5	5.65
	0.5	0.166666	0.999999	0.999999	0.333333	1.999998	1.999998	6	6	1	0.00
	0.333333	0.111111	0.5	0.5	0.25	1	3	5	9	5	5.92
	0.5	0.125	0.5	0.5	0.25	1	1	4	4	1	0.00
	0.333333	***0.2***	***0.5***	***1***	***0.333333***	***1***	***3***	***1***	***9***	***7***	***5.46***
	1	0.2	1	0.5	0.2	1	0.5	5	2.5	0.5	0.00
	0.2	0.2	1	4	0.333333	3	9	8	9	4	8.39
	0.5	0.166667	1	1	0.333333	2	2	6	6	1	0.00
	0.166667	1	0.5	0.125	9	4	1	0.5	0.125	0.25	0.39
	0.333333	0.111111	0.5	4	0.2	1	7	3	9	7	8.26
	0.333333	0.111111	0.5	1	0.2	1	3	5	9	3	1.66
	2	0.25	2	1	0.111111	1	0.333333	5	2	0.333333	1.88
	0.5	0.111111	0.5	1	0.142857	1	2	3	6	2	3.55
	0.5	0.166667	2	4	0.333333	3	9	9	9	4	6.59
	5	0.5	2	9	0.166667	1	4	8	9	4	6.90
	0.2	0.166667	0.5	0.25	0.2	1	0.5	6	2	0.5	5.49
	2	0.25	0.5	0.142857	0.125	0.25	0.111111	2	0.333333	0.5	4.16
	0.2	0.111111	0.5	0.5	0.25	1	1	5	6	1	2.69
	1	0.25	0.25	0.111111	0.25	0.25	0.111111	1	0.25	0.25	2.17
	3	0.2	1.2	0.6	0.142857	1.2	0.25	6	6	0.5	7.27
	0.25	0.111111	0.5	0.5	0.166667	0.5	0.5	4	4	1	5.09
INTERACTIVE	2	0.125	1	0.333333	0.125	0.333333	0.5	9	3	0.333333	7.12
	3	0.111111	0.5	0.2	0.142857	0.5	0.142857	5	1	0.333333	5.61
	3	0.2	0.5	0.5	0.111111	0.5	0.25	4	2	0.5	2.34
	1	0.2	2	0.5	0.2	2	0.166667	8	3	0.333333	5.68
	3	0.2	0.5	0.5	0.111111	0.5	0.25	4	2	0.5	2.34
	1	0.2	2	1	0.2	2	1	8	2	0.333333	2.62
	3	0.111111	0.333333	0.25	0.125	0.5	0.25	3	2	0.5	6.28
	2	0.25	2	1	0.125	1	0.333333	6	2	0.333333	1.16
	0.25	0.111111	1	3	0.333333	2	9	4	9	3	6.22
	0.333333	***0.111111***	***1***	***1***	***0.2***	***2***	***2***	***7***	***7***	***0.5***	***3.04***
	0.25	0.111111	0.5	3	0.333333	2	9	7	9	6	7.83

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	4	0.2	0.333333	0.166667	0.2	0.111111	0.125	0.333333	0.5	0.5	9.36
	0.2	0.111111	0.5	0.5	0.25	1	1	5	6	1	2.69
	3	0.111111	0.2	0.111111	0.111111	0.2	0.111111	2	0.5	0.25	7.98
	2	0.111111	1	0.25	0.125	0.5	0.25	5	3	0.5	3.42
	4	0.25	2	1	0.2	0.5	0.25	4	2	0.5	3.63
	1	***0.111111***	***2***	***0.25***	***0.2***	***1***	***0.333333***	***8***	***2***	***0.25***	***3.35***
	1	0.142857	1	0.5	0.142857	1	0.5	8	4	0.5	0.13
	0.25	0.111111	1	3	0.333333	2	9	4	9	3	6.22
	0.333333	1	1	0.333333	4	8	2	1	0.333333	0.333333	2.85
	3	0.2	0.2	0.125	0.25	0.111111	0.111111	0.5	0.25	0.5	6.29
	1	0.25	0.5	0.25	0.25	0.5	0.25	2	1	0.5	0.00
	0.166667	1	2	0.25	9	9	1	2	0.25	0.25	4.55
	2	0.25	0.25	0.111111	0.125	0.125	0.111111	1	0.5	0.333333	2.59
	4	0.25	2	3	0.142857	0.5	0.25	4	2	0.5	8.45
	4	0.25	2	0.2	0.25	0.5	0.125	2	0.5	0.25	7.45
	3	0.125	2	0.5	0.142857	0.333333	0.125	3	1	0.25	9.33
	1	0.2	1	0.333333	0.25	1	0.333333	7	2	0.333333	0.91
	1	0.25	2	1	0.25	2	1	8	4	0.5	0.00
	4	0.2	2	1	0.166667	3	0.333333	8	4	0.5	8.88
	1	0.166667	1	0.5	0.333333	1	0.5	8	4	0.5	3.60
	2	***0.25***	***1***	***1***	***0.125***	***1***	***0.333333***	***5***	***3***	***0.333333***	***2.47***
	0.5	***0.111111***	***2***	***0.25***	***0.2***	***1***	***0.5***	***5***	***2***	***0.5***	***6.60***
	2	0.25	0.25	0.111111	0.142857	0.142857	0.111111	1	0.25	0.25	5.61
	0.5	0.111111	0.5	3	0.111111	0.5	3	3	9	6	7.08
	4	0.25	0.5	0.125	0.333333	0.111111	0.111111	0.5	0.333333	0.333333	9.18
	2	0.25	1	0.333333	0.125	0.333333	0.111111	5	1	0.25	1.30
	4	0.25	0.25	0.142857	0.25	0.25	0.111111	1	0.25	0.25	8.93
	1	0.166667	2	0.5	0.166667	2	0.5	9	3	0.333333	0.44
	2	0.25	1	0.25	0.125	0.5	0.125	4	1	0.25	0.00
	3	0.2	2	0.333333	0.25	1	0.25	5	3	0.25	7.69
	0.5	0.125	1	0.5	0.25	2	1	6	3	0.5	0.50

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
BACKGROUND CLASS	2	0.5	0.5	3	0.25	0.25	1.5	1	6	6	0
	4	1	0.333333	2	0.333333	0.2	0.5	0.5	2	2	3.27
	3	0.333333	3	3	0.333333	0.5	3	2	9	6	8.81
	2	0.5	0.5	6	0.333333	0.333333	1	2	4	5	6.17
	2	1	0.333333	2	0.333333	0.111111	0.5	0.333333	2	8	1.76
	2	0.25	0.25	0.5	0.125	0.125	0.25	1	2	2	0
	0.166667	0.5	0.5	0.25	5	5	4	1	0.333333	0.25	6.8
	0.166667	1	0.5	0.111111	7	1	0.333333	0.25	0.111111	0.2	4.64
	4	0.5	0.25	1	0.125	0.111111	0.5	0.5	3	6	1.67
	1	0.166667	0.333333	0.166667	0.166667	0.2	0.166667	2	3	0.5	5.87
	3	1	1	9	0.333333	0.333333	3	1	9	9	0
	5	0.333333	0.166667	0.5	0.166667	0.111111	0.5	0.5	2	4	5.77
	0.2	0.5	1	0.2	4	8	1	2	0.333333	0.2	1.01
	3	0.333333	1	3	0.142857	0.333333	1	2	6	3	0.58
	2	0.25	0.5	0.5	0.2	0.25	0.5	2	2	2	1.9
	1	0.25	0.5	0.142857	0.142857	0.25	0.142857	5	2	0.5	6.03
	6	2	0.5	0.5	0.333333	0.111111	0.111111	0.25	0.25	1	0.4
	3	0.5	0.5	4	0.111111	0.111111	1	1	8	8	0.5
	4	0.5	1	7	0.142857	0.25	0.5	2	6	3	4.09
	3	***0.5***	***0.25***	***1***	***0.166667***	***0.111111***	***0.5***	***0.25***	***1***	***8***	***5.63***
	5	1	2	2	0.142857	0.333333	0.333333	2	2	1	0.23
	2	0.25	0.333333	1	0.111111	0.142857	0.166667	3	6	2	6.67
	4	0.5	2	8	0.142857	0.5	4	8	9	4	7.75
	4	0.5	1	1	0.125	0.25	0.25	2	2	1	0.00
	5	0.5	2	2	0.142857	0.5	0.5	4	4	1	0.27
	6	0.5	0.25	3	0.142857	0.142857	1	0.5	6	4	7.00
	5	0.25	2	2	0.25	0.333333	1	2	6	3	8.26
	0.25	***0.111111***	***0.5***	***1***	***0.2***	***1***	***1***	***5***	***5***	***1***	***4.02***
	0.2	0.111111	0.5	3	0.2	1	4	6	9	3	9.06
	4	0.5	2	9	0.125	0.5	4	4	9	7	5.22
	4	1	2	9	0.333333	4	2	5	9	2	8.87
	6	***1***	***1***	***9***	***0.166667***	***0.5***	***3***	***4***	***8***	***6***	***6.20***
	1	0.2	0.2	0.111111	0.2	0.2	0.111111	1	0.25	0.25	4.09
	5	0.333333	2	2	0.142857	0.5	0.5	5	5	1	1.33

	ANT-RC	ANT-ICP	ANT-COST	ANT-SEC	RC-ICP	RC-COST	RC-SEC	ICP-COST	ICP-SEC	COST-SEC	Consistency
	0.25	0.111111	0.5	0.25	0.142857	0.5	0.25	3	2	0.5	6.38
	0.25	0.111111	0.5	2	0.166667	1	3	5	9	3	5.52
	3	0.5	1	7	0.142857	0.125	3	3	9	8	8.05
	2	0.5	0.25	5	0.333333	0.2	3	1	9	9	2.46
	4	0.333333	3	9	0.142857	0.5	4	9	9	4	9.93
	0.5	0.111111	0.5	0.166667	0.142857	0.333333	0.166667	5	2	0.5	3.19
	3	0.5	0.166667	1	0.333333	0.111111	0.5	0.333333	2	5	1.59
	0.2	***0.111111***	***0.5***	***1***	***0.142857***	***0.5***	***1***	***3***	***9***	***2***	***8.47***

APPENDIX E

/* The following program is used to measure contradictory judgement matrices, compare contradictory judgement matrices with non-contradictory ones on different metrics. */

```
#include<stdio.h>
#include<math.h>
#include<time.h>
#include<float.h>

#define order 3 //order varies from 3  to 9
#define UTSize ((order*order-order)/2)
#define arrSize 17
#define MAX(a,b) ((a)>(b)?(a):(b))
#define MIN(a,b) ((a)<(b)?(a):(b))
#define numOfImpurities 9
void printmatrix(float[order][order]);
void printevector (float matrix[order]);
float findConsistency(float
matrix[order][order]);
void multiply_matrices (int row1,int
col1,int row2,int col2,float
first[order][order],float
second[order][order],float
multiply[order][order]);
void eigen_vector_by_square ( int p,int
k,float[p][k],float[order]);
void eigen_vector_by_root(int,int,float
mat[order][order],float[order]);

void complete_matrix(float
matrix[order][order]);
int check_contradiction( float
mat[order][order]);
int check_contradiction1(float
mat[order][order]);
void sort (float evector[order], int index
[order]);
void sort_again (float evector[order], int
index [order]);
int rank_reversal_best_case(float
matrix[order][order]);
int rank_reversal_any_case(float
matrix[order][order]);
void reverse_matrix(float
matrix[order][order] );
int choice1(float matrix[order][order],float
arr[]);
int choice2(float
matrix[order][order],float[]);
int choice3(float
matrix[order][order],float[]);
int choice4(float
matrix[order][order],float[]);
float roundSearch(float arr[],int low,int
high,float key);
float floorSearch(float arr[],int low,int
high,float key);
float ceilSearch(float arr[],int low,int
high,float key);
void add_impurity(float
matrix[order][order], float arr[]);
void copy_matrix(float matrix[order][order]
, float matrixnew[order][order]);


void results_3x3 (float arr[]);
void results_4x4 (float arr[]);
void results_5x5 (float arr[]);
void results_6x6 (float arr[]);
void results_7x7 (float arr[]);
void results_8x8 (float arr[]);
void results_9x9 (float arr[]);


typedef struct
{
     int row;
     int col;
}utMatrix;
utMatrix UTM[UTSize];

int main (void)
{
 float
arr[arrSize]={0.111111,0.125,0.142857,0.
166667,0.2,0.25,0.333333,0.5,1,2,3,4,5,6
,7,8,9};
 time_t seconds;
 time(&seconds);
 srand((unsigned int) seconds);

 switch (order)
 {
  case 3: results_3x3 (arr);
       break;
```

```
  case 4: results_4x4 (arr);
       break;
  case 5: results_5x5 (arr);
       break;
  case 6: results_6x6 (arr);
       break;
  case 7: results_7x7 (arr);
       break;
  case 8: results_8x8 (arr);
       break;
  case 9: results_9x9 (arr);
       break;
 }
 printf ("done");
}
void printmatrix(float mat[order][order])
{
   int i,j;

   for(i=0;i<order;i++)
   {
       for(j=0;j<order;j++)
       {
           printf("%f  ",mat[i][j]);
       }
       printf("\n");
   }
}
void printevector (float matrix[order])
{
 int i;
 for (i=0;i<order;i++)
 {
  printf ("%f\n",matrix[i]);
 }
}
void results_9x9 (float arr[])
{
 float matrix[order][order],consistency;
 float cr;
 int choice,flag,rep;
 int i,j,k,p,q,r,s,t,index,conchk;
 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
 } contradictory;

 typedef struct
 {
  int total_cardinal_mat;
  int total_rr_best;
  int total_rr_any;
 } cardinal;

 typedef struct
 {
  contradictory con;
  cardinal card;
 } inconsistent_matrices;

 zero_consistency zcon;
 inconsistent_matrices BIN[10]; //10 bins
for Inconsistent matrices.
 //INITIALIZE THE ELEMENTS OF
STRUCTURES
 zcon.total_consistent_mat = 0;
 zcon.total_rr_best = 0;
 zcon.total_rr_any = 0;
 for (j=0;j<10;j++)
 {
  BIN[j].con.total_contradictory_mat = 0;
  BIN[j].con.total_rr_best = 0;
  BIN[j].con.total_rr_any = 0;
  BIN[j].card.total_cardinal_mat = 0;
  BIN[j].card.total_rr_best = 0;
  BIN[j].card.total_rr_any = 0;
 }
 //INITIALIZATION ENDS

int ran[8], ra;
 for (ra=0;ra<8;ra++)
 {
  ran[ra] = (rand() % 3)+1;
  printf ("%d\n",ran[ra]);
 }
 int ter = (17.0/ran[0]) * (17.0/ran[1]) *
(17.0/ran[2]) * (17.0/ran[3]) *
(17.0/ran[4]) * (17.0/ran[5]) *
(17.0/ran[6]) * (17.0/ran[7]);
 printf ("total Count =%d\n",ter);
 for(i=0;i<17;i+=ran[0])
 {
  for(j=0;j<17;j+=ran[1])
  {
   for(k=0;k<17;k+=ran[2])
   {
    for(p=0;p<17;p+=ran[3])
    {
    for (q=0;q<17;q+=ran[4])
```

```
{
for (r=0;r<17;r+=ran[5])
{
for (s=0;s<17;s+=ran[6])
{
for (t=0;t<17;t+=ran[7])
{
 matrix[0][1]=arr[i];
 matrix[1][2]=arr[j];
 matrix[2][3]=arr[k];
 matrix[3][4]=arr[p];
 matrix[4][5]=arr[q];
 matrix[5][6]=arr[r];
 matrix[6][7]=arr[s];
 matrix[7][8]=arr[t];
     //SET DIAGONAL ELEMENTS AS 1
 int temp;
 for (temp=0;temp<order;temp++)
  matrix [temp][temp] = 1.0;
 for (choice=0;choice<4;choice++)
 {
  switch (choice)
  {
   case 0: flag = choice1 (matrix,arr);
          break;
   case 1: flag = choice2 (matrix,arr);
          break;
   case 2: flag = choice3 (matrix,arr);
        break;
   case 3: flag = choice4 (matrix,arr);
        break;
  } // SWITCH
  if (flag == 1) // Means Upper Triangle
is valid
  {
   flag = 0;
   complete_matrix (matrix); //complete
the lower triangle
   consistency = findConsistency
(matrix);
   cr = consistency * 100.0;
   if (cr == 0.0)
   {
    zcon.total_consistent_mat++;
    int check_rr_best = 0;
    check_rr_best =
rank_reversal_best_case(matrix);
    if (check_rr_best == 1)       //MEANS
RANK REVERSAL IS THERE
    {
     zcon.total_rr_best++;
    }
    int check_rr_any = 0;
        check_rr_any =
rank_reversal_any_case(matrix);
        if (check_rr_any == 1)        //MEANS
RANK REVERSAL IS THERE
        {
         zcon.total_rr_any++;
        }
       }           //CASE ENDS WHEN CR = 0
       if ((cr < 10.0) && (cr >= 0.0))
       {
        float matrixnew[order][order]; //add
impurity for a maximum of three times to
increase the probability of getting a
inconsistent matrix.
        copy_matrix (matrix ,
matrixnew);//copy matrix to matrixnew
        add_impurity (matrixnew,arr);
        consistency = findConsistency
(matrixnew);
        cr = consistency * 100.0;
        rep = 3;
        while ( ((cr >= 10.0)  ||  (cr < 0.0))
&& (rep > 0))
        {
         copy_matrix (matrix,matrixnew);
         add_impurity (matrixnew,arr);
         consistency = findConsistency
(matrixnew);
         cr = consistency * 100.0;
         rep--;
        } //while loop
        copy_matrix (matrixnew , matrix);
//copy matrixnew to matrix. matrixnew is
a matrix with added impurity.
        if ((cr < 10.0) && (cr >= 0.0))
        {
         // now we have a inconsistent
matrix.
        if (cr == 0.0)
         {
          zcon.total_consistent_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
//MEANS RANK REVERSAL IS THERE
          {
           zcon.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
//MEANS RANK REVERSAL IS THERE
```

```
         {
          zcon.total_rr_any++;
         }
        }          //CASE ENDS WHEN CR = 0
        else //CASE STARTS WHEN CR > 0
AND CR < 10
        {
         index = (int)cr;
         conchk = check_contradiction
(matrix);
         if (conchk == 1)  //If the matrix is
contradictory then get its Rank Reversal
         {

BIN[index].con.total_contradictory_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].con.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].con.total_rr_any++;
          }
         }
         else  //means that the matrix is
not contradictory and has ordinal
inconsistency present.
         {

BIN[index].card.total_cardinal_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].card.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].card.total_rr_any++;
          }
         }
        }  //CASE ENDS WHEN CR > 0
AND CR < 10
```

```
        } // INNER if ((cr < 10.0) && (cr >=
0.0))
       } //if ((cr < 10.0) && (cr >= 0.0))
      } //if (flag == 1)
     } // for choice = 0
    } //for i=0
   } //for j=0
  } //for k = 0
 }}}}} // for p=0 and q=0 and r=0 and s=0
and t=0
 //OUTPUT THE RESULTS
 FILE *fp;
 fp = fopen ("results_9x9.txt","w");
 if (fp == NULL)
 {
  printf ("Cannot open the file\n");
  return;
 }
 fprintf (fp,"********************* ANALYSIS
OF JUDGEMENT MATRICES WITH
ORDER %d X %d
***********************\n\n",order,order);
 fprintf (fp,"****************CONSISTENCY
= 0  ********************\n");
 fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);

 fprintf (fp,"****************CONSISTENCY
> 0 and CONSISTENCY < 10%
********************\n");
 for (i=0;i<10;i++)
 {
  fprintf
(fp,"\n*************************************
BIN[%d]: ****************************\n",i);
  fprintf (fp,"                           Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
```

```
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                    Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
}

void results_8x8 (float arr[])
{
 float matrix[order][order],consistency;
 float cr;
 int choice,flag,rep;
 int i,j,k,p,q,r,s,index,conchk;
 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
 } contradictory;

 typedef struct
 {
  int total_cardinal_mat;
  int total_rr_best;
  int total_rr_any;
 } cardinal;

 typedef struct
 {
  contradictory con;
  cardinal card;
 } inconsistent_matrices;

 zero_consistency zcon;
 inconsistent_matrices BIN[10]; //10 bins
for Inconsistent matrices.
 //INITIALIZE THE ELEMENTS OF
STRUCTURES
 zcon.total_consistent_mat = 0;
 zcon.total_rr_best = 0;
 zcon.total_rr_any = 0;
 for (j=0;j<10;j++)
 {
  BIN[j].con.total_contradictory_mat = 0;
  BIN[j].con.total_rr_best = 0;
  BIN[j].con.total_rr_any = 0;
  BIN[j].card.total_cardinal_mat = 0;
  BIN[j].card.total_rr_best = 0;
  BIN[j].card.total_rr_any = 0;
 }
 //INITIALIZATION ENDS

int ran[7], ra;
 for (ra=0;ra<7;ra++)
 {
  ran[ra] = (rand() % 2)+1;
  printf ("%d\n",ran[ra]);
 }
 int ter = (17.0/ran[0]) * (17.0/ran[1]) *
(17.0/ran[2]) * (17.0/ran[3]) *
(17.0/ran[4]) * (17.0/ran[5]) *
(17.0/ran[6]);
 printf ("totoal Count = %d\n",ter);
// int gh = 0;
 for(i=0;i<17;i+=ran[0])
 {
  for(j=0;j<17;j+=ran[1])
  {
   for(k=0;k<17;k+=ran[2])
   {
    for(p=0;p<17;p+=ran[3])
    {
    for (q=0;q<17;q+=ran[4])
    {
    for (r=0;r<17;r+=ran[5])
    {
    for (s=0;s<17;s+=ran[6])
    {
     //printf ("%d\n",gh++);
     matrix[0][1]=arr[i];
     matrix[1][2]=arr[j];
     matrix[2][3]=arr[k];
     matrix[3][4]=arr[p];
```

```
    matrix[4][5]=arr[q];
    matrix[5][6]=arr[r];
    matrix[6][7]=arr[s];
       //SET DIAGONAL ELEMENTS AS 1
    int temp;
    for (temp=0;temp<order;temp++)
     matrix [temp][temp] = 1.0;
    for (choice=0;choice<4;choice++)
    {
     switch (choice)
     {
      case 0: flag = choice1 (matrix,arr);
              break;
      case 1: flag = choice2 (matrix,arr);
              break;
      case 2: flag = choice3 (matrix,arr);
            break;
      case 3: flag = choice4 (matrix,arr);
            break;
     } // SWITCH
     if (flag == 1) // Means Upper Triangle
is valid
     {
      flag = 0;
      complete_matrix (matrix); //complete
the lower triangle
      consistency = findConsistency
(matrix);
      cr = consistency * 100.0;
      if (cr == 0.0)
      {
       zcon.total_consistent_mat++;
       int check_rr_best = 0;
       check_rr_best =
rank_reversal_best_case(matrix);
       if (check_rr_best == 1)        //MEANS
RANK REVERSAL IS THERE
       {
        zcon.total_rr_best++;
       }
       int check_rr_any = 0;
       check_rr_any =
rank_reversal_any_case(matrix);
       if (check_rr_any == 1)        //MEANS
RANK REVERSAL IS THERE
       {
        zcon.total_rr_any++;
       }
      }           //CASE ENDS WHEN CR = 0
      if ((cr < 10.0) && (cr >= 0.0))
      {
       float matrixnew[order][order]; //add
impurity for a maximum of three times to
increase the probability of getting a
inconsistent matrix.
        copy_matrix (matrix ,
matrixnew);//copy matrix to matrixnew
        add_impurity (matrixnew,arr);
        consistency = findConsistency
(matrixnew);
        cr = consistency * 100.0;
        rep = 3;
        while ( ((cr >= 10.0)  ||  (cr < 0.0))
&& (rep > 0))
        {
         copy_matrix (matrix,matrixnew);
         add_impurity (matrixnew,arr);
         consistency = findConsistency
(matrixnew);
         cr = consistency * 100.0;
         rep--;
        } //while loop
        copy_matrix (matrixnew , matrix);
//copy matrixnew to matrix. matrixnew is
a matrix with added impurity.
        if ((cr < 10.0) && (cr >= 0.0))
        {
         // now we have a inconsistent
matrix.
        if (cr == 0.0)
         {
          zcon.total_consistent_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
//MEANS RANK REVERSAL IS THERE
          {
           zcon.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
//MEANS RANK REVERSAL IS THERE
          {
           zcon.total_rr_any++;
          }
         }         //CASE ENDS WHEN CR = 0
         else //CASE STARTS WHEN CR > 0
AND CR < 10
         {
          index = (int)cr;
          conchk = check_contradiction
(matrix);
          if (conchk == 1)  //If the matrix is
contradictory then get its Rank Reversal
```

```
        {
BIN[index].con.total_contradictory_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].con.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].con.total_rr_any++;
          }
         }
         else   //means that the matrix is
not contradictory and has ordinal
inconsistency present.
         {
BIN[index].card.total_cardinal_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].card.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].card.total_rr_any++;
          }
         }
        }   //CASE ENDS WHEN CR > 0
AND CR < 10
       } // INNER if ((cr < 10.0) && (cr >=
0.0))
      } //if ((cr < 10.0) && (cr >= 0.0))
     } //if (flag == 1)
    } // for choice = 0
   } //for i=0
  } //for j=0
 } //for k = 0
}}}} // for p=0 and q=0 and r=0 and s=0
 //OUTPUT THE RESULTS
 FILE *fp;
 fp = fopen ("results_8x8.txt","w");
 if (fp == NULL)
 {
  printf ("Cannot open the file\n");
  return;
 }
 fprintf (fp,"******************** ANALYSIS
OF JUDGEMENT MATRICES WITH
ORDER %d X %d
**********************\n\n",order,order);
 fprintf (fp,"***************CONSISTENCY
= 0  *******************\n");
 fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);

 fprintf (fp,"***************CONSISTENCY
> 0 and CONSISTENCY < 10%
*******************\n");
 for (i=0;i<10;i++)
 {
  fprintf
(fp,"\n*************************************
BIN[%d]: ***************************\n",i);
  fprintf (fp,"                    Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
```

```
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
}

void results_7x7 (float arr[])
{
 float matrix[order][order],consistency;
 float cr;
 int choice,flag,rep,counter=0;
 int i,j,k,p,q,r,index,conchk;
 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
 } contradictory;

 typedef struct
 {
  int total_cardinal_mat;
  int total_rr_best;
  int total_rr_any;
 } cardinal;

 typedef struct
 {
  contradictory con;
  cardinal card;
 } inconsistent_matrices;

 zero_consistency zcon;
 inconsistent_matrices BIN[10]; //10 bins
for Inconsistent matrices.
 //INITIALIZE THE ELEMENTS OF
STRUCTURES
 zcon.total_consistent_mat = 0;
 zcon.total_rr_best = 0;
 zcon.total_rr_any = 0;

 for (j=0;j<10;j++)
 {
  BIN[j].con.total_contradictory_mat = 0;
  BIN[j].con.total_rr_best = 0;
  BIN[j].con.total_rr_any = 0;
  BIN[j].card.total_cardinal_mat = 0;
  BIN[j].card.total_rr_best = 0;
  BIN[j].card.total_rr_any = 0;
 }
 //INITIALIZATION ENDS

 for(i=0;i<17;i++)
 {
  for(j=0;j<17;j++)
  {
   for(k=0;k<17;k++)
   {
    for(p=0;p<17;p++)
    {
    for (q=0;q<17;q++)
    {
    for (r=0;r<17;r++)
    {
     matrix[0][1]=arr[i];
     matrix[1][2]=arr[j];
     matrix[2][3]=arr[k];
     matrix[3][4]=arr[p];
     matrix[4][5]=arr[q];
     matrix[5][6]=arr[r];
          //SET DIAGONAL ELEMENTS AS 1
     int temp;
     for (temp=0;temp<order;temp++)
      matrix [temp][temp] = 1.0;
     for (choice=0;choice<4;choice++)
     {
      switch (choice)
      {
       case 0: flag = choice1 (matrix,arr);
               break;
       case 1: flag = choice2 (matrix,arr);
               break;
       case 2: flag = choice3 (matrix,arr);
             break;
       case 3: flag = choice4 (matrix,arr);
             break;
      } // SWITCH
      if (flag == 1) // Means Upper Triangle
is valid
      {
       flag = 0;
       complete_matrix (matrix); //complete
the lower triangle
       consistency = findConsistency
(matrix);
       cr = consistency * 100.0;
```

```
        if (cr == 0.0)
        {
         zcon.total_consistent_mat++;
         int check_rr_best = 0;
         check_rr_best =
rank_reversal_best_case(matrix);
         if (check_rr_best == 1)        //MEANS
RANK REVERSAL IS THERE
         {
          zcon.total_rr_best++;
         }
         int check_rr_any = 0;
         check_rr_any =
rank_reversal_any_case(matrix);
         if (check_rr_any == 1)        //MEANS
RANK REVERSAL IS THERE
         {
          zcon.total_rr_any++;
         }
        }           //CASE ENDS WHEN CR = 0
        if ((cr < 10.0) && (cr >= 0.0))
        {
         float matrixnew[order][order]; //add
impurity for a maximum of three times to
increase the probability of getting a
inconsistent matrix.
         copy_matrix (matrix ,
matrixnew);//copy matrix to matrixnew
         add_impurity (matrixnew,arr);
         consistency = findConsistency
(matrixnew);
         cr = consistency * 100.0;
         rep = 3;
         while ( ((cr >= 10.0)  ||  (cr < 0.0))
&& (rep > 0))
         {
          copy_matrix (matrix,matrixnew);
          add_impurity (matrixnew,arr);
          consistency = findConsistency
(matrixnew);
          cr = consistency * 100.0;
          rep--;
         } //while loop
         copy_matrix (matrixnew , matrix);
//copy matrixnew to matrix. matrixnew is
a matrix with added impurity.
         if ((cr < 10.0) && (cr >= 0.0))
         {
          // now we have a inconsistent
matrix.
         if (cr == 0.0)
          {
           zcon.total_consistent_mat++;
           int check_rr_best = 0;
           check_rr_best =
rank_reversal_best_case(matrix);
           if (check_rr_best == 1)
//MEANS RANK REVERSAL IS THERE
           {
            zcon.total_rr_best++;
           }
           int check_rr_any = 0;
           check_rr_any =
rank_reversal_any_case(matrix);
           if (check_rr_any == 1)
//MEANS RANK REVERSAL IS THERE
           {
            zcon.total_rr_any++;
           }
          }          //CASE ENDS WHEN CR = 0
          else //CASE STARTS WHEN CR > 0
AND CR < 10
          {
           index = (int)cr;
           conchk = check_contradiction
(matrix);
           if (conchk == 1)  //If the matrix is
contradictory then get its Rank Reversal
           {

BIN[index].con.total_contradictory_mat++;
            int check_rr_best = 0;
            check_rr_best =
rank_reversal_best_case(matrix);
            if (check_rr_best == 1)
            {
             BIN[index].con.total_rr_best++;
            }
            int check_rr_any = 0;
            check_rr_any =
rank_reversal_any_case(matrix);
            if (check_rr_any == 1)
            {
             BIN[index].con.total_rr_any++;
            }
           }
           else  //means that the matrix is
not contradictory and has ordinal
inconsistency present.
           {

BIN[index].card.total_cardinal_mat++;
            int check_rr_best = 0;
            check_rr_best =
rank_reversal_best_case(matrix);
            if (check_rr_best == 1)
            {
             BIN[index].card.total_rr_best++;
```

```
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].card.total_rr_any++;
          }
         }
        }   //CASE ENDS WHEN CR > 0
AND CR < 10
        } // INNER if ((cr < 10.0) && (cr >=
0.0))
      } //if ((cr < 10.0) && (cr >= 0.0))
     } //if (flag == 1)
    } // for choice = 0
   } //for i=0
  } //for j=0
 } //for k = 0
}}} // for p=0 and q=0 and r=0
 //OUTPUT THE RESULTS
 FILE *fp;
 fp = fopen ("results_7x7.txt","w");
 if (fp == NULL)
 {
  printf ("Cannot open the file\n");
  return;
 }
 fprintf (fp,"********************* ANALYSIS
OF JUDGEMENT MATRICES WITH
ORDER %d X %d
***********************\n\n",order,order);
 fprintf (fp,"****************CONSISTENCY
= 0  ********************\n");
 fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);

 fprintf (fp,"****************CONSISTENCY
> 0 and CONSISTENCY < 10%
*******************\n");
 for (i=0;i<10;i++)
 {
  fprintf
(fp,"\n*************************************
BIN[%d]: ****************************\n",i);
  fprintf (fp,"                            Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                     Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
}
void results_6x6 (float arr[])
{
 float matrix[order][order],consistency;
 float cr;
 int choice,flag,rep;
 int i,j,k,p,q,r,index,conchk;
 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
```

```
} contradictory;

typedef struct
{
 int total_cardinal_mat;
 int total_rr_best;
 int total_rr_any;
} cardinal;

typedef struct
{
 contradictory con;
 cardinal card;
} inconsistent_matrices;

zero_consistency zcon;
inconsistent_matrices BIN[10]; //10 bins
for Inconsistent matrices.
//INITIALIZE THE ELEMENTS OF
STRUCTURES
zcon.total_consistent_mat = 0;
zcon.total_rr_best = 0;
zcon.total_rr_any = 0;
for (j=0;j<10;j++)
{
 BIN[j].con.total_contradictory_mat = 0;
 BIN[j].con.total_rr_best = 0;
 BIN[j].con.total_rr_any = 0;
 BIN[j].card.total_cardinal_mat = 0;
 BIN[j].card.total_rr_best = 0;
 BIN[j].card.total_rr_any = 0;
}
//INITIALIZATION ENDS
for(i=0;i<17;i++)
{
 for(j=0;j<17;j++)
 {
  for(k=0;k<17;k++)
  {
   for(p=0;p<17;p++)
   {
   for (q=0;q<17;q++)
   {
    matrix[0][1]=arr[i];
    matrix[1][2]=arr[j];
    matrix[2][3]=arr[k];
    matrix[3][4]=arr[p];
    matrix[4][5]=arr[q];
         //SET DIAGONAL ELEMENTS AS 1
    int temp;
    for (temp=0;temp<order;temp++)
     matrix [temp][temp] = 1.0;
    for (choice=0;choice<4;choice++)
    {
      switch (choice)
      {
       case 0: flag = choice1 (matrix,arr);
               break;
       case 1: flag = choice2 (matrix,arr);
               break;
       case 2: flag = choice3 (matrix,arr);
             break;
       case 3: flag = choice4 (matrix,arr);
             break;
      } // SWITCH
      if (flag == 1) // Means Upper Triangle
is valid
      {
       flag = 0;
       complete_matrix (matrix); //complete
the lower triangle
       consistency = findConsistency
(matrix);
       cr = consistency * 100.0;
       if (cr == 0.0)
       {
        zcon.total_consistent_mat++;
        int check_rr_best = 0;
        check_rr_best =
rank_reversal_best_case(matrix);
        if (check_rr_best == 1)        //MEANS
RANK REVERSAL IS THERE
        {
         zcon.total_rr_best++;
        }
        int check_rr_any = 0;
        check_rr_any =
rank_reversal_any_case (matrix);
        if (check_rr_any == 1)        //MEANS
RANK REVERSAL IS THERE
        {
         zcon.total_rr_any++;
        }
       }           //CASE ENDS WHEN CR = 0
       if ((cr < 10.0) && (cr >= 0.0))
       {
        float matrixnew[order][order]; //add
impurity for a maximum of three times to
increase the probability of getting a
inconsistent matrix.
        copy_matrix (matrix ,
matrixnew);//copy matrix to matrixnew
        add_impurity (matrixnew,arr);
        consistency = findConsistency
(matrixnew);
        cr = consistency * 100.0;
          rep = 3;
```

```
        while ( ((cr >= 10.0)  ||  (cr < 0.0))
&& (rep > 0))
        {
        copy_matrix (matrix,matrixnew);
         add_impurity (matrixnew,arr);
         consistency = findConsistency
(matrixnew);
         cr = consistency * 100.0;
         rep--;
        } //while loop
        copy_matrix (matrixnew , matrix);
//copy matrixnew to matrix. matrixnew is
a matrix with added impurity.
       if ((cr < 10.0) && (cr >= 0.0))
       {
        //  now we have a inconsistent
matrix.
         if (cr == 0.0)
        {
         zcon.total_consistent_mat++;
         int check_rr_best = 0;
         check_rr_best =
rank_reversal_best_case(matrix);
         if (check_rr_best == 1)
//MEANS RANK REVERSAL IS THERE
         {
          zcon.total_rr_best++;
         }
         int check_rr_any = 0;
         check_rr_any =
rank_reversal_any_case(matrix);
         if (check_rr_any == 1)
//MEANS RANK REVERSAL IS THERE
         {
          zcon.total_rr_any++;
         }
        }          //CASE ENDS WHEN CR = 0
        else //CASE STARTS WHEN CR > 0
AND CR < 10
        {
         index = (int)cr;
         conchk = check_contradiction
(matrix);
         if (conchk == 1)  //If the matrix is
contradictory then get its Rank Reversal
         {

BIN[index].con.total_contradictory_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].con.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].con.total_rr_any++;
          }
         }
         else
         {

BIN[index].card.total_cardinal_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].card.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].card.total_rr_any++;
          }
         }
        }   //CASE ENDS WHEN CR > 0
AND CR < 10
        } // INNER if ((cr < 10.0) && (cr >=
0.0))
      } //if ((cr < 10.0) && (cr >= 0.0))
     } //if (flag == 1)
    } // for choice = 0
   } //for i=0
  } //for j=0
 } //for k = 0
}} // for p=0 and q=0
 //OUTPUT THE RESULTS
 FILE *fp;
 fp = fopen ("results_6x6.txt","w");
 if (fp == NULL)
 {
  printf ("Cannot open the file\n");
  return;
 }
 fprintf (fp,"******************** ANALYSIS
OF JUDGEMENT MATRICES WITH
ORDER %d X %d
**********************\n\n",order,order);
 fprintf (fp,"****************CONSISTENCY
= 0  *******************\n");
```

```
 fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);

 fprintf (fp,"****************CONSISTENCY
> 0 and CONSISTENCY < 10%
********************\n");
 for (i=0;i<10;i++)
 {
  fprintf
(fp,"\n**************************************
BIN[%d]: ****************************\n",i);
  fprintf (fp,"                              Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                    Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
}

void results_5x5 (float arr[])
{
  int aaaa = 0;
 float matrix[order][order],consistency;
 float cr;
 int choice,flag,rep;
 int i,j,k,p,q,r,index,conchk;
 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
 } contradictory;

 typedef struct
 {
  int total_cardinal_mat;
  int total_rr_best;
  int total_rr_any;
 } cardinal;

 typedef struct
 {
  contradictory con;
  cardinal card;
 } inconsistent_matrices;

 zero_consistency zcon;
 inconsistent_matrices BIN[10]; //10 bins
for Inconsistent matrices.
 //INITIALIZE THE ELEMENTS OF
STRUCTURES
 zcon.total_consistent_mat = 0;
 zcon.total_rr_best = 0;
 zcon.total_rr_any = 0;

 for (j=0;j<10;j++)
 {
  BIN[j].con.total_contradictory_mat = 0;
  BIN[j].con.total_rr_best = 0;
  BIN[j].con.total_rr_any = 0;
  BIN[j].card.total_cardinal_mat = 0;
  BIN[j].card.total_rr_best = 0;
  BIN[j].card.total_rr_any = 0;
 }
 for(i=0;i<17;i++)
```

```
{
 for(j=0;j<17;j++)
 {
  for(k=0;k<17;k++)
  {
   for(p=0;p<17;p++)
   {
    matrix[0][1]=arr[i];
    matrix[1][2]=arr[j];
    matrix[2][3]=arr[k];
    matrix[3][4]=arr[p];
        //SET DIAGONAL ELEMENTS AS 1
    int temp;
    for (temp=0;temp<order;temp++)
     matrix [temp][temp] = 1.0;
    for (choice=0;choice<4;choice++)
    {
     switch (choice)
     {
      case 0: flag = choice1 (matrix,arr);
              break;
      case 1: flag = choice2 (matrix,arr);
              break;
      case 2: flag = choice3 (matrix,arr);
            break;
      case 3: flag = choice4 (matrix,arr);
            break;
     } // SWITCH
     if (flag == 1) // Means Upper Triangle
is valid
     {
      flag = 0;
      complete_matrix (matrix); //complete
the lower triangle
      //COMPUTE WEIGHT VECTOR FOR
CALCULATING THE DISTANCE
      consistency = findConsistency
(matrix);
      cr = consistency * 100.0;
      if (cr == 0.0)
      {
       zcon.total_consistent_mat++;
       int check_rr_best = 0;
       check_rr_best =
rank_reversal_best_case(matrix);
       if (check_rr_best == 1)        //MEANS
RANK REVERSAL IS THERE
       {
        zcon.total_rr_best++;
       }
       int check_rr_any = 0;
       check_rr_best =
rank_reversal_any_case(matrix);
       if (check_rr_any == 1)        //MEANS
RANK REVERSAL IS THERE
       {
        zcon.total_rr_any++;
       }
      }          //CASE ENDS WHEN CR = 0
      if ((cr < 10.0) && (cr >= 0.0))
      {
       float matrixnew[order][order]; //add
impurity for a maximum of three times to
increase the probability of getting a
inconsistent matrix.
       copy_matrix (matrix ,
matrixnew);//copy matrix to matrixnew
       add_impurity (matrixnew,arr);
       consistency = findConsistency
(matrixnew);
       cr = consistency * 100.0;
        rep = 3;
        while ( ((cr >= 10.0)  ||  (cr < 0.0))
&& (rep > 0))
        {
        copy_matrix (matrix,matrixnew);
         add_impurity (matrixnew,arr);
         consistency = findConsistency
(matrixnew);
         cr = consistency * 100.0;
         rep--;
        } //while loop
        copy_matrix (matrixnew , matrix);
//copy matrixnew to matrix. matrixnew is
a matrix with added impurity.
       if ((cr < 10.0) && (cr >= 0.0))
       {
        // now we have a inconsistent
matrix.
         if (cr == 0.0)
        {
         zcon.total_consistent_mat++;
         int check_rr_best = 0;
         check_rr_best =
rank_reversal_best_case(matrix);
         if (check_rr_best == 1)
//MEANS RANK REVERSAL IS THERE
         {
          zcon.total_rr_best++;
         }
         int check_rr_any = 0;
         check_rr_any =
rank_reversal_any_case(matrix);
         if (check_rr_any == 1)
//MEANS RANK REVERSAL IS THERE
         {
          zcon.total_rr_any++;
```

```
        }
       }          //CASE ENDS WHEN CR = 0
       else //CASE STARTS WHEN CR > 0
AND CR < 10
       {
        index = (int)cr;
        conchk = check_contradiction
(matrix);
        if (conchk == 1)  //If the matrix is
contradictory then get its Rank Reversal
        {

BIN[index].con.total_contradictory_mat++;


         int check_rr_best = 0;
         check_rr_best =
rank_reversal_best_case(matrix);
         if (check_rr_best == 1)
         {
          BIN[index].con.total_rr_best++;
         }
         int check_rr_any = 0;
         check_rr_any =
rank_reversal_any_case(matrix);
         if (check_rr_any == 1)
         {
            if (aaaa == 0)
         {
            if (index == 1)
            {
               if
(BIN[index].con.total_contradictory_mat >
290)
               {

//complete_matrix(matrix);
                    float evector[order];
                    printmatrix(matrix);
                    printf("Consistency =
%f\n", findConsistency(matrix));

eigen_vector_by_square(order,
order,matrix,evector);
                    printevector(evector);
                    reverse_matrix(matrix);
                    printmatrix(matrix);
                    printf("Consistency =
%f\n", findConsistency(matrix));

eigen_vector_by_square(order,
order,matrix,evector);
                    printevector(evector);
                    aaaa = 1;
                }
            }
          }
          BIN[index].con.total_rr_any++;
         }
        }
        else  //means that the matrix is
not contradictory and has cardinal
inconsistency present.
        {

BIN[index].card.total_cardinal_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].card.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].card.total_rr_any++;
          }
        }
       }  //CASE ENDS WHEN CR > 0
AND CR < 10
      } // INNER if ((cr < 10.0) && (cr >=
0.0))
     } //if ((cr < 10.0) && (cr >= 0.0))
    } //if (flag == 1)
   } // for choice = 0
  } //for i=0
 } //for j=0
} //for k = 0
} // for p=0
 //Compute the Average of all the
distance methods.

 FILE *fp;
 fp = fopen ("results_5x5.txt","w");
 if (fp == NULL)
 {
  printf ("Cannot open the file\n");
  return;
 }
 fprintf (fp,"********************* ANALYSIS
OF JUDGEMENT MATRICES WITH
ORDER %d X %d
***********************\n\n",order,order);
 fprintf (fp,"****************CONSISTENCY
= 0  *******************\n");
```

```
 fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);

 fprintf (fp,"****************CONSISTENCY
> 0 and CONSISTENCY < 10%
*******************\n");
 for (i=0;i<10;i++)
 {
  fprintf
(fp,"\n**************************************
BIN[%d]: ****************************\n",i);
  fprintf (fp,"                              Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                    Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
}
```

```
void results_4x4 (float arr[])
{
 float matrix[order][order],consistency;
 int i,j,k,p,q,r,index,conchk;
 //float ei_vector[order];
 //float wt_vector[order];
 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
 } contradictory;

 typedef struct
 {
  int total_cardinal_mat;
  int total_rr_best;
  int total_rr_any;
 } cardinal;

 typedef struct
 {
  contradictory con;
  cardinal card;
 } inconsistent_matrices;

 zero_consistency zcon;
 inconsistent_matrices BIN[10];
 //INITIALIZE Variables
 zcon.total_consistent_mat = 0;
 zcon.total_rr_best = 0;
 zcon.total_rr_any = 0;

 for (i=0;i<10;i++)
 {
   BIN[i].con.total_contradictory_mat = 0;
   BIN[i].con.total_rr_best = 0;
   BIN[i].con.total_rr_any = 0;
   BIN[i].card.total_cardinal_mat = 0;
   BIN[i].card.total_rr_best = 0;
   BIN[i].card.total_rr_any = 0;
 }

 for (i=0;i<17;i++)
 {
  for (j=0;j<17;j++)
  {
```

```
   for (k=0;k<17;k++)
   {
    for (p=0;p<17;p++)
    {
     for (q=0;q<17;q++)
     {
      for (r=0;r<17;r++)
      {
       matrix[0][1]=arr[i];
       matrix[0][2]=arr[j];
       matrix[0][3]=arr[k];
       matrix[1][2]=arr[p];
       matrix[1][3]=arr[q];
       matrix[2][3]=arr[r];
       matrix[0][0] = 1.0;
       matrix[1][1] = 1.0;
       matrix[2][2] = 1.0;
       matrix[3][3] = 1.0;
       complete_matrix (matrix);
       consistency = findConsistency
(matrix);
       float cr = consistency*100.0;
       if (cr == 0.0)
       {
         zcon.total_consistent_mat++;
         int check_rr_best = 0;
         check_rr_best =
rank_reversal_best_case(matrix);
         if (check_rr_best == 1)
//MEANS RANK REVERSAL IS THERE
         {
          zcon.total_rr_best++;
         }
         int check_rr_any = 0;
         check_rr_any =
rank_reversal_any_case(matrix);
         if (check_rr_any == 1)         //MEANS
RANK REVERSAL IS THERE
         {
          zcon.total_rr_any++;
         }
        } //if (cr == 0.0)
        if (cr > 0.0 && cr < 10.0)
        {
         index = (int)cr;
         conchk = check_contradiction1
(matrix);
         if (conchk == 1)
         {

BIN[index].con.total_contradictory_mat++;

          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].con.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].con.total_rr_any++;
          }
         }
         else
         {

BIN[index].card.total_cardinal_mat++;
          int check_rr_best = 0;
          check_rr_best =
rank_reversal_best_case(matrix);
          if (check_rr_best == 1)
          {
           BIN[index].card.total_rr_best++;
          }
          int check_rr_any = 0;
          check_rr_any =
rank_reversal_any_case(matrix);
          if (check_rr_any == 1)
          {
           BIN[index].card.total_rr_any++;
          }
         }
        } // if (cr > 0.0 && cr <= 10.0)
       }  //r
      }  //q
     }  //p
    }  //k
   }  //j
  } //i

  //OUTPUT THE RESULTS
  FILE *fp;
  fp = fopen ("results_4x4.txt","w");
  if (fp == NULL)
  {
   printf ("Cannot open file \n");
   return;
  }

  fprintf (fp,"****************CONSISTENCY
= 0  ********************\n");
  fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
```

```
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);


 fprintf (fp,"****************CONSISTENCY
> 0 and CONSISTENCY < 10%
********************\n");
 for (i=0;i<10;i++)
 {
  fprintf
(fp,"\n*************************************
BIN[%d]: ****************************\n",i);
  fprintf (fp,"                              Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                    Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
} //function
```

```
void results_3x3 (float arr[])
{
 float matrix[order][order],consistency;
 int i,j,k,index,conchk;

 typedef struct
 {
  int total_consistent_mat;
  int total_rr_best;
  int total_rr_any;
 } zero_consistency;

 typedef struct
 {
  int total_contradictory_mat;
  int total_rr_best;
  int total_rr_any;
 } contradictory;

 typedef struct
 {
  int total_cardinal_mat;
  int total_rr_best;
  int total_rr_any;
 } cardinal;

 typedef struct
 {
  contradictory con;
  cardinal card;
 } inconsistent_matrices;

 zero_consistency zcon;
 inconsistent_matrices BIN[10];
 //INITIALIZE EVERYTHING
 zcon.total_consistent_mat = 0;
 zcon.total_rr_best = 0;
 zcon.total_rr_any = 0;

 for (i=0;i<10;i++)
 {
   BIN[i].con.total_contradictory_mat = 0;
   BIN[i].con.total_rr_best = 0;
   BIN[i].con.total_rr_any = 0;
   BIN[i].card.total_cardinal_mat = 0;
   BIN[i].card.total_rr_best = 0;
   BIN[i].card.total_rr_any = 0;
 }
 //INITIALIZATION ENDS

 for (i=0;i<17;i++)
 {
  for (j=0;j<17;j++)
  {
```

```
  for (k=0;k<17;k++)
  {
   matrix[0][1] = arr[i];
   matrix[0][2] = arr[j];
   matrix[1][2] = arr[k];
   matrix[0][0] = 1.0;
   matrix[1][1] = 1.0;
   matrix[2][2] = 1.0;

   complete_matrix (matrix);
   consistency = findConsistency (matrix);
   float cr = consistency*100.0;
   if (cr == 0.0)
   {
    zcon.total_consistent_mat++;
    int check_rr_best = 0;
    check_rr_best =
rank_reversal_best_case(matrix);
    if (check_rr_best == 1)        //MEANS
RANK REVERSAL IS THERE
    {
     zcon.total_rr_best++;
    }
    int check_rr_any = 0;
    check_rr_any =
rank_reversal_any_case(matrix);
    if (check_rr_any == 1)        //MEANS
RANK REVERSAL IS THERE
    {
     zcon.total_rr_any++;
    }
   } //if (cr == 0.0)
   if (cr > 0.0 && cr < 10.0)
   {
    index = (int)cr;
    conchk = check_contradiction1
(matrix);
    if (conchk == 1)
    {

BIN[index].con.total_contradictory_mat++;
     int check_rr_best = 0;
     check_rr_best =
rank_reversal_best_case(matrix);
     if (check_rr_best == 1)
     {
      BIN[index].con.total_rr_best++;
     }
     int check_rr_any = 0;
     check_rr_any =
rank_reversal_any_case(matrix);
     if (check_rr_any == 1)
     {
      BIN[index].con.total_rr_any++;
     }
    }
    else
    {
     BIN[index].card.total_cardinal_mat++;
     int check_rr_best = 0;
     check_rr_best =
rank_reversal_best_case(matrix);
     if (check_rr_best == 1)
     {
      BIN[index].card.total_rr_best++;
     }
     int check_rr_any = 0;
     check_rr_any =
rank_reversal_any_case(matrix);
     if (check_rr_any == 1)
     {
      BIN[index].card.total_rr_any++;
     }
    }
   } // if (cr > 0.0 && cr <= 10.0)
  }
 }
}

 //OUTPUT THE RESULTS
 FILE *fp;
 fp = fopen ("results_3x3.txt","w");
 if (fp == NULL)
 {
  printf ("Cannot open file \n");
  return;
 }

 fprintf (fp,"****************CONSISTENCY
= 0  ********************\n");
 fprintf (fp,"Total Matrices :
%d\n",zcon.total_consistent_mat);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage = %f\n",zcon.total_rr_best,
((float)zcon.total_rr_best/zcon.total_consis
tent_mat) * 100.0);
 fprintf (fp,"Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage = %f\n",zcon.total_rr_any,
((float)zcon.total_rr_any/zcon.total_consist
ent_mat) * 100.0);


 fprintf (fp,"****************CONSISTENCY
> 0 and CONSISTENCY < 10%
*******************\n");
 for (i=0;i<10;i++)
```

```
 {
  fprintf
(fp,"\n****************************************
BIN[%d]: ******************************\n",i);
  fprintf (fp,"                          Total
Contradictory Matrices :
%d\n",BIN[i].con.total_contradictory_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].con.total_rr_best,
((float)BIN[i].con.total_rr_best/BIN[i].con.t
otal_contradictory_mat) * 100.0 );
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n\n",BIN[i].con.total_rr_any,
((float)BIN[i].con.total_rr_any/BIN[i].con.to
tal_contradictory_mat) * 100.0);

  fprintf (fp,"                    Total Cardinally
Inconsistent Matrices :
%d\n",BIN[i].card.total_cardinal_mat);
  fprintf (fp,"Number of Matrices in which
Rank reversal occurs in BEST case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_best,
((float)BIN[i].card.total_rr_best/BIN[i].card.
total_cardinal_mat) * 100.0);
  fprintf (fp," Number of Matrices in which
Rank reversal occurs in ANY case: %d,
Percentage =
%f\n",BIN[i].card.total_rr_any,
((float)BIN[i].card.total_rr_any/BIN[i].card.
total_cardinal_mat) * 100.0);
 }
}


/* RETURNS 0 IF THERE IS NO RANK
REVERSAL and returns 1 RANK
REVERSAL IN THE BEST CASE */
int rank_reversal_best_case(float
matrix[order][order])
{
 float evector1[order], evector2[order];
 int i,j;
 int index1[order], temp_index[order],
index2[order]; //to store the sorted
indexes of evector1 and evector2
 eigen_vector_by_square (order, order,
matrix, evector1);
 //printevector(evector1);
 sort(evector1,index1);
 reverse_matrix(matrix);
 eigen_vector_by_square (order, order,
matrix, evector2);
 sort_again(evector2,temp_index);
//temp_index contains the reverse sorted
sequence as that of index1
 for (i=0,j=order;i<order;i++,j--)
  index2[i] = temp_index[j-1];
 //printevector1(index1);
 //printevector1(index2);
 if (index1[0] != index2[0])
  return (1);
 else
  return (0);
}
//RETURNS 0 IF THERE IS NO RANK
REVERSAL
//RETURNS 1 IF THERE IS A RANK
REVERSAL IN ANY CASE
int rank_reversal_any_case(float
matrix[order][order])
{
 float evector1[order], evector2[order];
 int i,j;
 int index1[order], temp_index[order],
index2[order]; //to store the sorted
indexes of evector1 and evector2
 eigen_vector_by_square (order, order,
matrix, evector1);
 //printevector(evector1);
 sort(evector1,index1);
 reverse_matrix(matrix);
 eigen_vector_by_square (order, order,
matrix, evector2);
 sort_again(evector2,temp_index);
//temp_index contains the reverse sorted
sequence as that of index1
 for (i=0,j=order;i<order;i++,j--)
  index2[i] = temp_index[j-1];
 //printevector1(index1);
 //printevector1(index2);
 for (i=0;i<order;i++)
 {
    if(index1[i] != index2[i])
     return(1);
 }
 return (0);
}
//sort method do the sorting by
preventing the order of similar elements.
void sort (float evector[order], int index
[order])
{
 int i, j, min, k=0;
```

```
 for (i=0;i<order;i++)
 {
  min = 0;
  for (j=1;j<order;j++)
  {
      if ((evector[j] < evector[min]))// &&
(evector[min] != -1))
       min = j;
  }
  index[k++] = min;
  evector[min] = 100.0;
 //
printevector(evector);printf("\n");printevect
or1(index);printf("\n");
 }
}
// sort_again method do the sorting by
reversing the order of similar elemenets.
void sort_again (float evector[order], int
index [order])
{
 int i, j, min, k=0;
 for (i=0;i<order;i++)
 {
  min = 0;
  for (j=1;j<order;j++)
  {
      if ((evector[j] <= evector[min]))// &&
(evector[min] != -1))
       min = j;
  }
  index[k++] = min;
  evector[min] = 100.0;
 //
printevector(evector);printf("\n");printevect
or1(index);printf("\n");
 }
}
void complete_matrix(float
matrix[order][order])
{

     int l,m;

     for(l=1;l<order;l++)
    {
                for(m=0;m<l;m++)
                {

     matrix[l][m]=1.0/matrix[m][l];
                }
    }
}
```

```
void reverse_matrix (float
matrix[order][order])
{
     int i,j;
     for (i=0;i<order;i++)
     {
                for (j=0;j<order;j++)
                {
                           matrix[i][j] =
1.0/matrix[i][j];
                }
     }
}
int choice1(float matrix[order][order],float
arr[])
{
     int i,j,k,flag=0;
     for(i=0;i<order-2;i++)
     {
               for(j=i+2;j<order;j++)
               {
                         matrix[i][j]=matrix[i][j-
1]*matrix[j-1][j];
                         if( (matrix[i][j]>1/9.0)
&& (matrix[i][j]<(9.0) ))
                         {

for(k=0;k<arrSize;k++)
              {

if((int)(arr[k]*100000)==(int)(matrix[i][j]*10
0000))
                  {
                     flag=1;
                     break;
                  }
              }
              if(flag==0)
                  return 0;

          }
          else
              return 0;
               }
     }
     return 1;
}

int choice2(float matrix[order][order],float
arr[])//using fn ptrs the 3 fns choice2,3,4
can be replaced with a single fn.
{
     int i,j;
     for(i=0;i<order-2;i++)
```

```
    {
            for(j=i+2;j<order;j++)
            {

    matrix[i][j]=floorSearch(arr,0,arrSize-
1,matrix[i][j-1]*matrix[j-1][j]);
            }
    }
    return 1;
}
int choice3(float matrix[order][order],float
arr[])
{
    int i,j;
    for(i=0;i<order-2;i++)
    {
            for(j=i+2;j<order;j++)
            {

    matrix[i][j]=ceilSearch(arr,0,arrSize-
1,matrix[i][j-1]*matrix[j-1][j]);
            }
    }
    return 1;
}

int choice4(float matrix[order][order],float
arr[])
{
    int i,j;
    for(i=0;i<order-2;i++)
    {
            for(j=i+2;j<order;j++)
            {

    matrix[i][j]=roundSearch(arr,0,arrSize-
1,matrix[i][j-1]*matrix[j-1][j]);
            }
    }
      return 1;
}

float floorSearch(float arr[],int low,int
high,float key) //optimize by binary
search
{
    if(key<=arr[low])
            return arr[low];
    if(key>=arr[high])
            return arr[high];
    high--;
    for(;high>=low;high--)
    {
            if(arr[high]<=key)
                    return arr[high];
    }
}

float ceilSearch(float arr[],int low,int
high,float key) //optimize by binary
search
{
    //printf("\n %f \n",key);
    if(key<=arr[low])
            return arr[low];
    if(key>=arr[high])
            return arr[high];
    low++;
    for(;low<=high;low++)
    {
            if(arr[low]>=key)
                    return arr[low];//giving
wrong output for 1/9*3=0.333333->o.5
    }
}

float roundSearch(float arr[],int low,int
high,float key) //optimize by binary
search
{
    if(key<=arr[low])
            return arr[low];
    if(key>=arr[high])
            return arr[high];
    low++;
    for(;low<=high;low++)
    {
            if(arr[low]==key)
                    return arr[low];
            else if(arr[low]>key)
            {

                    float diff1=key-arr[low-
1];
                    float diff2=arr[low]-key;
                    if(diff1<diff2)
                            return arr[low-
1];
                    else
                            return arr[low];
            }
    }
}

void add_impurity(float
matrix[order][order],float arr[])
{
```

```
    int l=0,i=0,j=0,k=0,
randNum,randIndex,matIndex,arrIndex,ar
rLow,arrHigh,matValueFrequency;
    float
matValue,tempIndex,impurity[numOfImp
urities],randImpurity,indexArray[]={0,1,2,3
,4,5,6,7,8,9,10,11,12,13,14,15,16};

    for(i=0;i<order;i++)
    {
        for(j=i+1;j<order;j++)
        {
            UTM[k].row=i;
            UTM[k].col=j;
            k++;
        }
    }

    randNum=(rand())%UTSize;
//RANDOM NUMBER FOR NUMBER OF
IMPURITIES TO BE ADDED.
    for(j=0;j<randNum;j++)
    {

randIndex=rand()%UTSize;//RANDOM
INDEX WHOSE VALUE IS TO BE
PERTURBED

    matValue=matrix[UTM[randIndex].row
][UTM[randIndex].col];
        if(matValue>=1)

    tempIndex=arrSize/2+matValue-
1;//FIND THE INDEX OF THE VALUE
SELECTED IN THE ARRAY
CORRESPONDING TO SCALE.
        else
            tempIndex=arrSize/2-
(1/matValue-1);

    arrIndex=roundSearch(indexArray,0,ar
rSize-1,tempIndex);
        arrLow=MAX(arrIndex-4,0);

    arrHigh=MIN(arrIndex+4,arrSize-1);
        matValueFrequency=9-
(arrHigh-arrLow+1);
        //populating impurity array
        l=0;
        for(k=arrLow;k<arrIndex;k++)
        {
            impurity[l++]=arr[k];
        }

    for(k=1;k<=matValueFrequency;k++)
        {

    impurity[l++]=matValue;
        }
        for(k=arrIndex;k<=arrHigh;k++)
        {
            impurity[l++]=arr[k];
        }
        int temp1;

    randImpurity=impurity[rand()%numOf
Impurities];

    matrix[UTM[randIndex].row][UTM[rand
Index].col]=randImpurity;

    matrix[UTM[randIndex].col][UTM[randI
ndex].row] = 1.0/randImpurity;
    }
}

void copy_matrix (float
matrix[order][order] ,float
matrixnew[order][order])
{
    int i,j;
    for (i=0;i<order;i++)
    {
        for (j=0;j<order;j++)
        {
            matrixnew[i][j] =
matrix[i][j];
        }
    }
}

void eigen_vector_by_square (int row,int
col,float mat[row][col],float e_vector[row])
{
 float
temp[row][col],square[row][col],sum_rows[r
ow],prev_sum_rows[row],diff_sum_rows[ro
w];
 float total=0.0,diff;
 int i,j,repeat=1;
                    //INITIALIZATION
 for (i=0;i<row;i++)
 {
  sum_rows[i] = 0.0;  prev_sum_rows[i] =
0.0;
  for (j=0;j<col;j++)
   temp[i][j] = mat[i][j];
```

```
 }
 while (repeat == 1)
 {
  multiply_matrices
(row,col,row,col,temp,temp,square);
  //SUM UP THE ROWS AND NORMALIZE
THEM.
  for (i=0;i<row;i++)
  {
   for (j=0;j<col;j++)
   {
    sum_rows[i] += square[i][j];
    temp[i][j] = square[i][j];   // copy square
matrix in temp to square it again, if
required.
   }
   total += sum_rows[i];
  }
  //normalize the row sums.
  for (i=0;i<row;i++)
  {
   sum_rows[i] /= total;
   diff_sum_rows[i] = sum_rows[i] -
prev_sum_rows[i];
   prev_sum_rows[i] = sum_rows[i];
  }
  repeat = 0;
  total=0.0;
  for (i=0;i<row;i++)
  {
   if (diff_sum_rows[i] >= 0.001)
    repeat = 1;
  }
  if (repeat == 0)
   for (i=0;i<row;i++)
   {
    //printf (" %f  \n",sum_rows[i]);
    e_vector[i] = sum_rows[i];
   }
 }//end of while loop
}

/*
FUNCTION pre-multiplies MATRICES OF
ORDER row1xcol1 with matrix of
row2xcol2 AND STORES THE result in
matrix of orde  row1 x col2.
*/
void multiply_matrices (int row1,int
col1,int row2,int col2,float
first[row1][col1],float
second[row2][col2],float
multiply[row1][col2])
{
 int c,d,k; float sum=0.0;
 int i,j;
 //PRINT MATRICES
if (col1 != row2)
 {
  printf ("\nThe two matrices cannot be
multiplied as col1 of first is not equal to
row2 of the other\n");
  return;
 }
 for ( c = 0 ; c < row1 ; c++ )
  {
   for ( d = 0 ; d < col2 ; d++ )
   {
    for ( k = 0 ; k < row2 ; k++ )
    {
     sum = sum + first[c][k]*second[k][d];
    }
    multiply[c][d] = sum;
    sum = 0.0;
   }
  }
}

float findConsistency(float
matrix[order][order])
{
      float eVector[order];
      float colSum[order];
      float temp[order][order];
      float lambda=0.0,ci,cr;
      float
ri[]={0.0,0.0,0.58,0.9,1.12,1.24,1.32,1.41,
1.45,1.49};
      int i,j;
      //cnt4++;
      for(i=0;i<order;i++)
      {
            colSum[i]=0.0;
            eVector[i]=0.0;
            for(j=0;j<order;j++)
            {
                  temp[i][j]=matrix[i][j];
            }
      }
      for(i=0;i<order;i++)
      {
            for(j=0;j<order;j++)
            {
                  colSum[i]+=matrix[j][i];
            }
      }
      //Normalizing the Columns
      for(i=0;i<order;i++)
```

```
        {
                for(j=0;j<order;j++)
                {
                        temp[j][i]/=colSum[i];
                }
        }
        for(i=0;i<order;i++)
        {
                for(j=0;j<order;j++)
                {
                        eVector[i]+=temp[i][j];
                }
                eVector[i]/=order;
        }
        for(i=0;i<order;i++)
                lambda+=(eVector[i]*colSum[i]);
        ci=(lambda-order)/(order-1);
        cr=ci/ri[order-1];
        return (cr);
}
int check_contradiction( float
mat[order][order])
{
 int i,j,k;
 for (i=0;i<order;i++)
 {
  for (j=0;j<order;j++)
  {
   for (k=0;k<order;k++)
   {
    if ((i != j) && (i != k) && (j != k))
    {
     if (((log(mat[i][j]) * log(mat[i][k])) <=0)
&& ((log(mat[i][k]) * log(mat[j][k])) <0))
      return (1);
     else
      if (log(mat[i][j]) == 0 && log(mat[i][k])
== 0 && log (mat[j][k]) !=0)
       return (1);
    }
   }
  }
 }
 return (0);
}

int check_contradiction1(float
mat[order][order])
{
 int i,j,k;
 int rows = order;
 for (i=0;i<rows;i++)
 {
  for (j=0;j<rows;j++)
  {
   for (k=0;k<rows;k++)
   {
    if (mat[i][j]>1 && mat[i][k] < 1 &&
mat[j][k] > 1)
     return (1);
    if (mat[i][j]<1 && mat[i][k] > 1 &&
mat[j][k] < 1)
     return (1);
    if (mat[i][j]==1 && mat[i][k] > 1 &&
mat[j][k] < 1)
     return (1);
    if (mat[i][j]==1 && mat[i][k] < 1 &&
mat[j][k] > 1)
     return (1);
    if (mat[i][j]==1 && mat[i][k] == 1 &&
mat[j][k] < 1)
     return (1);
    if (mat[i][j]==1 && mat[i][k] == 1 &&
mat[j][k] > 1)
     return (1);
   }
  }
 }
 return (0);
}
```

/*
The following distance methods were used as quantitative metrics for comparing contradictory and non-contradictory judgement matrices. These were called from appropriate places in function results_3x3(), results_4x4(), etc. To avoid duplicating the code, the functions which call the distance functions are not shown.
***/**

```
float distMethod0(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod1(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod2(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod3(float
matrix[order][order],int order1,float
```

```
eVector[order][numOfWtMethods],int
colIndex);
float distMethod4(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod5(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod6(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod7(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod8(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod9(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod10(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);
float distMethod11(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex);

// Least Square Method [Method number
0 of the research paper]
float distMethod0(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
     int i,j;
     float sum=0.0;
     for(i=0;i<order1;i++)
     {
               for(j=0;j<order1;j++)
               {
                         float temp1, temp2;
                         temp1 = matrix[i][j]-
eVector[i][colIndex]/eVector[j][colIndex];
                         temp2 = temp1 * temp1;
                         sum+=temp2;
               }
     }
     return sum;
}

//average_distance[0][0] +=
distMethod0(matrix,order,wt_vector,0);
float distMethod0(float mat[order][order],
int temp2, float eVector[order]
[numOfWtMethods], int colIndex )
{
 int i,j;
 int row, col;
 row = col = order;
 float mv=0.0;//for calculting minimum
violation
 float e_eigen[order];
 for (i=0;i<order;i++)
  e_eigen[i] = eVector[i][colIndex];
 for (i=0;i<row;i++)
 {
  for (j=0;j<col;j++)
  {
   if((e_eigen[i] > e_eigen[j]) && mat[i][j] < 1)
    mv += 1;
   else
    if (((e_eigen[i] == e_eigen[j]) && mat[i][j]
!= 1)||((e_eigen[i] != e_eigen[j]) && mat[i][j]
==1))
     mv += 0.5;
  }
 }
 return (mv);
}

//Worst Least Square.
float distMethod1(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
     int i,j;
     float max=0.0;
     for(i=0;i<order1;i++)
     {
               for(j=0;j<order1;j++)
               {
                   float temp1, temp2;
                         temp1 = matrix[i][j]-
eVector[i][colIndex]/eVector[j][colIndex];
                         temp2 = temp1 * temp1;
                         if((i!=j)&&(temp2>max))
                         {
                                   max=temp2;
                         }
```

```
            }
        }
        return max;
}

//Preference Weighted Least Square
Method.
float distMethod2(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
        int i,j;
        float sum=0.0;
        for(i=0;i<order1;i++)
        {
                for(j=0;j<order1;j++)
                {
            float temp1, temp2;
            temp1 =
matrix[i][j]*eVector[j][colIndex]-
eVector[i][colIndex];
                        temp2 = temp1 * temp1;
                        sum+= temp2;
                }
        }
        return sum;
}
//Preference Weighted Least Worst
Square.
float distMethod3(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
        int i,j;
        float max=0.0;
        for(i=0;i<order1;i++)
        {
                for(j=0;j<order1;j++)
                {
                    float temp1, temp2;
            temp1 =
matrix[i][j]*eVector[j][colIndex]-
eVector[i][colIndex];
                        temp2 = temp1 * temp1;
                        if((i!=j)&&(temp2>max))
                        {
                                max=temp2;
                        }
                }
        }
        return max;
}
```

```
//Least absolute Error
float distMethod4(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
        int i,j;
        float sum=0.0;
        for(i=0;i<order1;i++)
        {
                for(j=0;j<order1;j++)
                {
                    float temp1, temp2;
                    temp1 = matrix[i][j]-
eVector[i][colIndex]/eVector[j][colIndex];
                    temp2 = fabsf (temp1);
                        sum+=temp2;
                }
        }
        return sum;
}
//Worst Least Absolute Error.
float distMethod5(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
        int i,j;
        float max=0.0;
        for(i=0;i<order;i++)
        {
                for(j=0;j<order;j++)
                {
                    float temp1, temp2;
                    temp1 = matrix[i][j]-
eVector[i][colIndex]/eVector[j][colIndex];
                    temp2 = fabsf (temp1);
                        if((i!=j)&&(temp2>max))
                        {
                                max=temp2;
                        }
                }
        }
        return max;
}
// Preference Weighted Least Absoute
Error
float distMethod6(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
        int i,j;
```

```
    float sum=0.0;
    for(i=0;i<order1;i++)
    {
            for(j=0;j<order1;j++)
            {
               float temp1, temp2;
               temp1 =
matrix[i][j]*eVector[j][colIndex]-
eVector[i][colIndex];
               temp2 = fabsf(temp1);
               sum+=temp2;
            }
    }
    return sum;
}
// Preference Weighted Least Worst
Absoute Error
float distMethod7(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
    int i,j;
    float max=0.0;
    for(i=0;i<order;i++)
    {
            for(j=0;j<order;j++)
            {
               float temp1, temp2;
               temp1 =
matrix[i][j]*eVector[j][colIndex]-
eVector[i][colIndex];
               temp2 = fabsf(temp1);
                    if((i!=j)&&(temp2>max))
                    {
                            max=temp2;
                    }
            }
    }
    return max;
}

//Logarithmic Least Square
 float distMethod8(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
    int i,j;
    float sum=0.0;
    for(i=0;i<order1;i++)
    {
            for(j=0;j<order1;j++)
            {
                    float temp1, temp2;
                    temp1 = logf
(eVector[i][colIndex]) - logf
(eVector[j][colIndex]);
                    temp2 = logf(matrix[i][j])
- temp1;
        temp2 = temp2 * temp2;
                    sum+=temp2;
            }
    }
    return sum;
}
//Logarithmic Least Worst Square
float distMethod9(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
    int i,j;
    float max=0.0;
    for(i=0;i<order1;i++)
    {
            for(j=0;j<order1;j++)
            {
                    float temp1, temp2;
                    temp1 = logf
(eVector[i][colIndex]) - logf
(eVector[j][colIndex]);
                    temp2 = logf(matrix[i][j])
- temp1;
                    temp2 = temp2 * temp2;
                    if ((i != j) && (temp2 >
max))
                      max = temp2;
            }
    }
    return max;
}

float distMethod10(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
    int i,j;
    float sum=0.0;
    for(i=0;i<order1;i++)
    {
            for(j=0;j<order1;j++)
            {
               float temp1, temp2, temp3;
               temp1 = logf
(eVector[i][colIndex]) -
logf(eVector[j][colIndex]);
```

```
                    temp2 = logf (matrix[i][j]) -
temp1;
                    temp3 = fabsf (temp2);
                          sum+=temp3;
                }
        }
        return sum;
}
float distMethod11(float
matrix[order][order],int order1,float
eVector[order][numOfWtMethods],int
colIndex)
{
        int i,j;
        float max=0.0;
        for(i=0;i<order;i++)
        {
                for(j=0;j<order;j++)
                {
                    float temp1, temp2, temp3;
                    temp1 = logf
(eVector[i][colIndex]) -
logf(eVector[j][colIndex]);
                    temp2 = logf (matrix[i][j]) -
temp1;
                    temp3 = fabsf (temp2);
                          if((i!=j)&& (temp3>max))
                          {
                                    max=temp3 ;
                          }
                }
        }
        return max;
}
```

APPENDIX F

Figure F.1 shows the flow chart depicting the logic behind adding impurities in a judgement matrix. It uses following definitions:

a) UTSize: for Matrix of Order N, UTSize= ((N*N)-N)/2

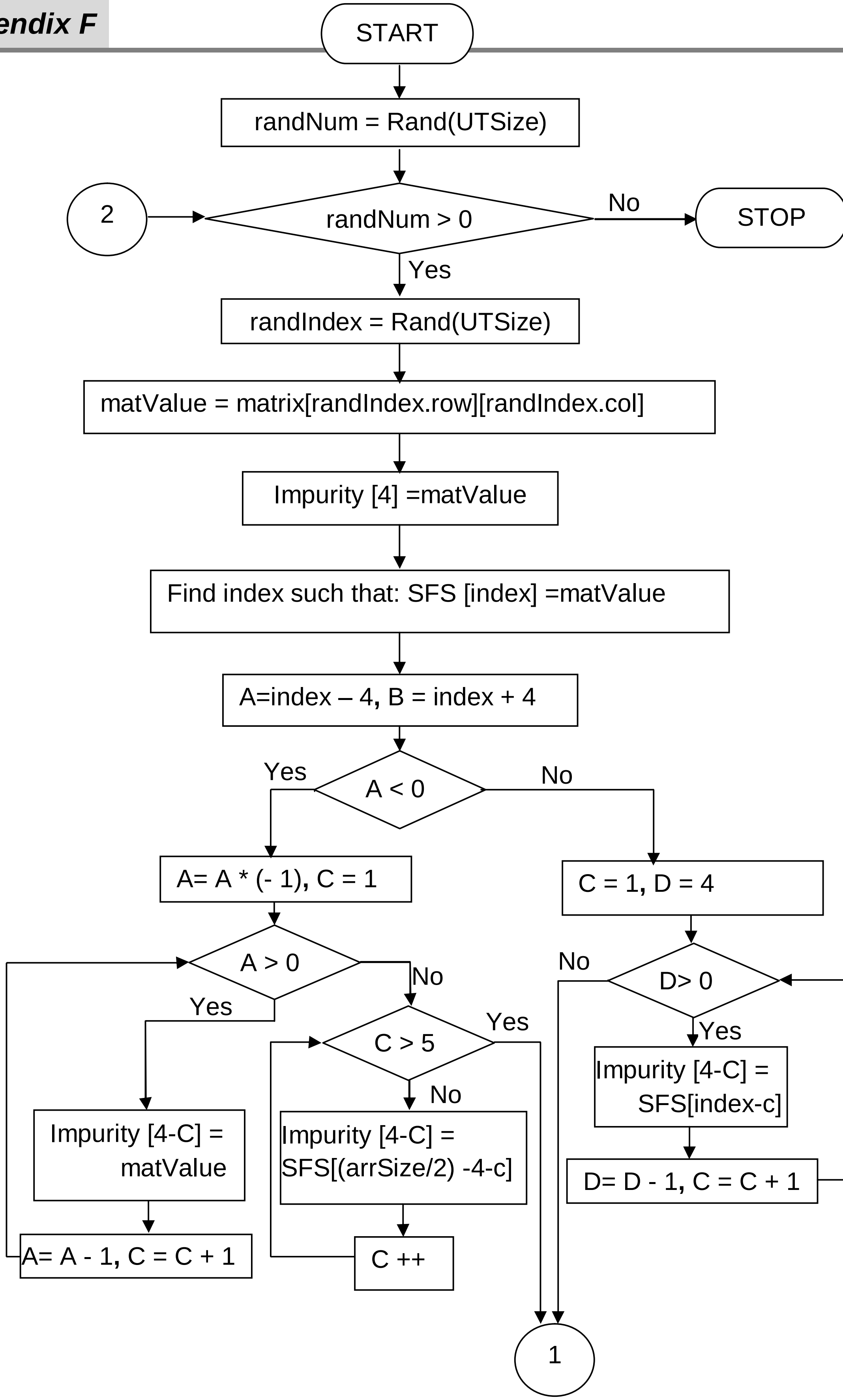
b) Saaty's Fundamental Scale: SFS[17] = { 1/9, 1/8, 1/7 … 1/2,1,2 … 8, 9}

c) Rand(N): Function which generates a Random number between 0 and (N-1)

d) Impurity[9]: Array of size 9 used for storing 9 values from which one is picked to replace a value in the matrix.

e) Matrix[order][order]: Matrix of size order x order in which the impurity has to be added.

f) randIndex[ ]: array of size UTSize such that each element of the array is associated with corresponding row and column.


START
randNum = Rand(UTSize)
2
randNum > 0
No
STOP
Yes
randIndex = Rand(UTSize)
matValue = matrix[randIndex.row][randIndex.col]
Impurity [4] =matValue
Find index such that: SFS [index] =matValue
A=index – 4, B = index + 4
Yes
A < 0
No
A= A * (- 1), C = 1
C = 1, D = 4
A > 0
No
Yes
C > 5
Yes
No
D> 0
Yes
Impurity [4-C] = matValue
No
Impurity [4-C] = SFS[(arrSize/2) -4-c]
Impurity [4-C] = SFS[index-c]
D= D - 1, C = C + 1
A= A - 1, C = C + 1
C ++
1

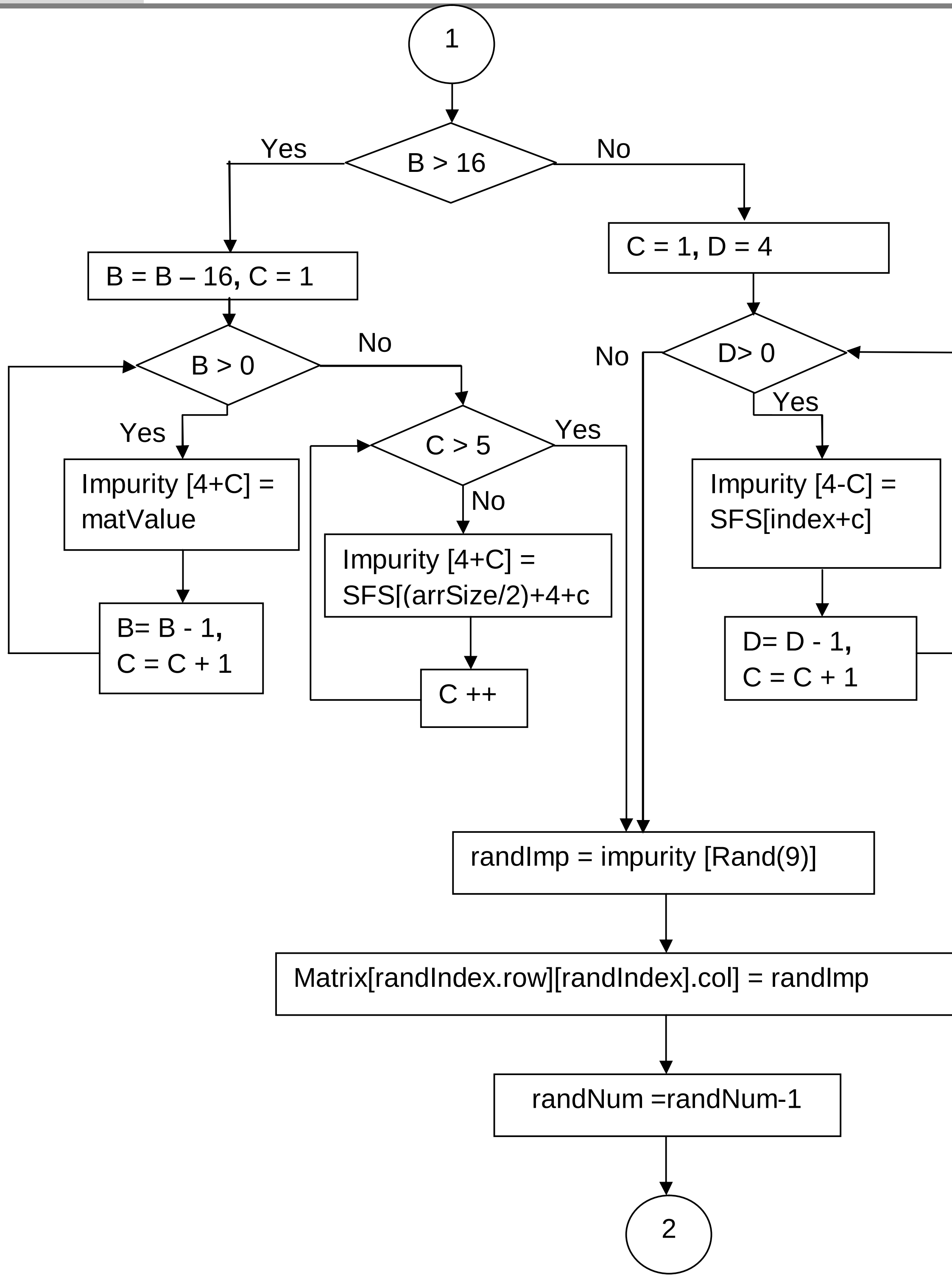


Figure F.1: Flow Chart for adding impurities in judgement matrices

LIST OF PUBLICATIONS AND PRESENTATIONS

The research study in this doctoral thesis resulted in following published / accepted research papers.

INTERNATIONAL JOURNALS

1) Gupta V. & Rohil M. K. (2013, February). Bit-Stuffing in 802.11 Beacon Frame: Embedding Non-Standard Custom Information. International Journal of Computer Applications 63(2): 6-12. Published by Foundation of Computer Science, New York, USA.
2) Gupta V. & Rohil M. K. (2012, August). Enhancing Wi-Fi with IEEE 802.11u for Mobile Data Offloading. International Journal of Mobile Network Communications and Telematics, volume 2, number 4, ISSN 1839-5678.
3) Gupta V. & Rohil M. K. (2012, February). Interworking of 3G and 802.11 networks: Present and the Future ahead. International Journal of Mobile and adhoc network, vol 2, issue 1, ISSN 2249-202X.

INTERNATIONAL CONFERENCES

4) Gupta V. & Rohil M. K. (2012, December). An experimental measurement of contradictory judgement matrices in AHP. Parallel Distributed and Grid Computing (PDGC), 2012 2nd IEEE International Conference on, pp.527 - 532, 6-8 Dec. 2012.
5) Gupta V. (2013, January). Network discovery and user preferences for network selection in 3G-WLAN interworking environment. Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on , pp.1,2, 7-10 Jan. 2013.
6) Gupta V. & Rohil M. K. (2013, February). Modeling User Preferences for Vertical Handover in 3G-WLAN Interworking Environment on top of IEEE 802.11u. 3rd IEEE International Advance Computing Conference, Feb 22-23, 2013 <Presented>.

7) Gupta V. & Swaroop S. (2012 April). Handover procedure using probability based mobility patterns in 3G-WLAN interworking environment. Recent Advances in Computing and Software Systems (RACSS), 2012 International Conference on, pp.224-227, 25-27 April 2012.

BOOK CHAPTER

8) Gupta V. & Rohil M. K. (2012). Mobile Data Offloading: Benefits, Issues, and Technological Solutions. Advances in Computer Science, Engineering & Applications, Springer Berlin Heidelberg, vol 167, 2012, pp 73 - 80.

NATIONAL CONFERENCES

9) Gupta V. & Rohil M. K. (2012, November). Information Embedding in IEEE 802.11 Beacon Frame. IJCA Proceedings on National Conference on Communication Technologies & its impact on Next Generation Computing 2012 CTNGC (3):12-16, November 2012.

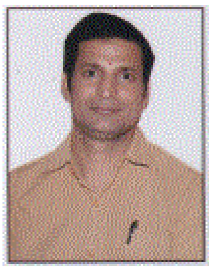
BRIEF BIOGRAPHY OF CANDIDATE

Vishal Gupta received the M. Tech. degree in Computer Science from Uttar Pradesh Technical University, Lucknow in the year 2008. He received B. E. degree in Computer Technology from Nagpur University in the year 2000.

In May, 2008 he joined the department of Computer Science and Information Systems, Birla Institute of Technology and Science - Pilani, Pilani Campus, as an Assistant Lecturer, and later in May 2009 was promoted to Lecturer within the same department. Prior to BITS Pilani, he served the department of MCA, Institute of Management Studies, Lal Quan, Ghaziabad for 2 years; Department of Computer Science and Engineering, Krishna Institute of Engineering and Technology, Ghaziabad for 3 years; and Bharat Institute of Technology, Meerut for 2 years.

His research interests include 3G-WLAN Interworking, Multi Criteria Decision Making, Ranking algorithms, Link Structure of the Web, and Search Engine ranking techniques.

BRIEF BIOGRAPHY OF THE SUPERVISOR



Prof. Mukesh Kumar Rohil received the M.Sc. (Hons.) degree in Physics, the B.E. (Hons.) degree in Civil Engineering, the M.E. degree in Systems & Information and the Ph.D. degree in Computer Science & Engineering from the Birla Institute of Technology and Science, Pilani – 333 031, Rajasthan, India, in 1993, 1993, 1995 and 2004 respectively.

He served M/s Asia Polytex India Limited, Mumbai (earlier Bombay) as Systems Analyst-cum-Programmer during Aug-Nov, 1993. He worked as ASSISTANT LECTURER during June 1995 and Dec 1998, as Lecturer during Jan 1998 and May 2006, Assistant Professor during June 2006 and January 2013, and since February 2013 he is working as an Associate Professor with Department of Computer Science and Information Systems, Birla Institute of Technology and Science, Pilani – 333 031, Rajasthan, India. His area of research interest includes Artificial Intelligence and Expert Systems, Computer Graphics, Digital Image Processing of Remote Sensing Data, Wireless Networks, Geographical Information Systems, Applications of Artificial Intelligence in Software Engineering and pattern recognition.

Prof. Rohil has published 29 articles in conferences and in Journals. He is a life member of Indian Society of Remote Sensing (ISRS). Prof. Rohil received OPTO-MECH award for the best paper presented titled "Exploring possible applications of fuzzy logic in simulation of three-dimensional visualization using remote sensing data" presented in annual convention of ISRS, in Nov 2000. Prof. Rohil has received commendation certificate for the article "Digital Imaging in Teaching Construction Process" presented in 6th Annual Convention & Seminar on "Education and Training of Building Professionals", New Delhi; April 20-22, 2000 & published in the Journal of Indian Buildings Congress; New Delhi; April 2000, 7:1, 47-53