INTRODUCTION TO FUZZY LOGIC

JAMES K. PECKOL

WILEY

Introduction to Fuzzy Logic

Introduction to Fuzzy Logic

James K. Peckol

Principal Lecturer Emeritus, Department of Electrical and Computer Engineering, University of Washington, Seattle, WA, USA
and
President, Oxford Consulting, Ltd, Seattle, WA, USA

WILEY

This edition first published 2021

Registered Office(s)
John Wiley & Sons, Inc., 111 River Street, Hoboken, NJ 07030, USA
John Wiley & Sons Ltd, The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

Editorial Office
The Atrium, Southern Gate, Chichester, West Sussex, PO19 8SQ, UK

For details of our global editorial offices, customer services, and more information about Wiley products visit us at www.wiley.com.

Wiley also publishes its books in a variety of electronic formats and by print-on-demand. Some content that appears in standard print versions of this book may not be available in other formats.

Library of Congress Cataloging-in-Publication Data

Name: Peckol, James K., author. | John Wiley & Sons, publisher.
Title: Introduction to fuzzy logic / James K. Peckol.
Description: Hoboken, NJ : Wiley, 2021. | Includes bibliographical references and index.
Identifiers: LCCN 2021011123 (print) | LCCN 2021011124 (ebook) | ISBN 9781119772613 (cloth) | ISBN 9781119772620 (adobe pdf) | ISBN 9781119772637 (epub)
Subjects: LCSH: Fuzzy logic. | Fuzzy sets. | Logic, Symbolic and mathematical.
Classification: LCC QA9.64 .P43 2021 (print) | LCC QA9.64 (ebook) | DDC 511.3/13–dc23
LC record available at https://lccn.loc.gov/2021011123
LC ebook record available at https://lccn.loc.gov/2021011124

Cover Design: Wiley
Cover Image: © Rasi Bhadramani/iStock/Getty Images

Set in 9.5/12.5pt STIXTwoText by Straive, Pondicherry, India

10 9 8 7 6 5 4 3 2 1

Dedication

To my family: Near and Extended, Close and Distant,
Present and Departed, So Similar,
So Different, So Known, So Surprising . . .
especially to our youngest brother Karl,
taken from us out of season during the last voyage
of the Edmund Fitzgerald.

Contents

Preface

Starting to Think Fuzzy and Beyond

Let's begin with these questions: "Exactly what is fuzzy logic?" "Why is the logic called fuzzy?" "Who might use fuzzy logic?" These are very good questions. People may have heard something about fuzzy logic and other kinds of logic but may not be quite sure what these terms mean or quite understand the applications.

Does fuzzy logic mean that someone's comment in a discussion is very confused? Let's try to answer that question and several of the other more common ones over the course of this text by starting with some simple fuzzy examples.

Our daily language is often routinely fuzzy; yet most of the time we easily understand it. Let's start by looking at some familiar expressions from our everyday exchanges.

Where did you park the car?
  I parked up *close* to the front door of the building.
Please put the box in the trunk of the car.
  I can't lift it. It's *very heavy*.
Are we close to the city yet?
  We're *roughly about* thirty minutes away.
Is that shower warm?
  It's *very, very* hot.
Is he tall?
  Yes, he's *very, very* tall.
Is she smart?
  Trust me, she's *incredibly* smart.

Each of the responses to the questions above is somewhat vague and imprecise yet, for the most part, each provides a reasonable answer that is probably well understood. Each expression in italics is called a fuzzy *linguistic variable* rather than a crisp real number or a simple "yes" or "no." The expressions give a high-level view of fuzzy logic or fuzzy reasoning. Accompanying such reasoning we also find threshold logic and perceptrons, which model the brain.

In daily life, we find that there are two kinds of imprecision: statistical and nonstatistical. Statistical imprecision is that which arises from such events as the outcome of a coin toss or card game. Nonstatistical imprecision, on the other hand, is that which we find in expressions such as "*We're roughly about thirty minutes away*." This latter type of imprecision is what we call fuzzy.

Children learn to understand and to manipulate such instructions at an early age. They quite easily understand phrases such as "Be home by around 5:00." Perhaps children understand too well. They are adept at turning such a fuzzy expression into one that is also fuzzy. When they arrive home shortly after 6:00, they argue that 6:00 is about 5:00.

As we note, humans are quite facile at understanding fuzzy expressions and linguistic variables. For a computer, however, the opposite is true. With fuzzy logic, threshold logic, and perceptrons, increasingly both computer hardware and software are evolving to more challenging and interesting areas of logic such as neural networks, machine learning, and artificial intelligence.

Despite its amusing and seemingly contradictory name, fuzzy logic is not a logic that is fuzzy. On the contrary, fuzzy logic is a way of capturing the vagueness and imprecision that are so common in everyday human language. This capturing of vagueness and imprecision is also found in threshold logic and has significant application in artificial neurons called perceptrons. Capturing and representing the vagueness and imprecision of everyday language in terms that a computer can understand and work with is one of the objectives of fuzzy logic.

The computers we are all so familiar with operate using classical or crisp logic. Classical logic, around since Aristotle, divides the world into precise, nonoverlapping groups such as: yes–no, up–down, true–false, black–white, etc. Like a light bulb that can only be on or off, a classical logic statement can only be true or false. Those of you who have just said, "Wait a minute, what if the light's on a dimmer?" have just taken the first step to understanding fuzzy logic, threshold logic, and perceptrons. Like the light on a dimmer, a fuzzy logic statement can also be completely true or completely false, but it can also be partially true or partially false.

Fuzzy logic is simply a flexible variation and extension of classical logic. Fuzzy logic can represent statements that are completely true or false, and it can also represent those that are partially true. Classical logic lives in a black-and-white world. Fuzzy logic, threshold logic, and perceptrons, like humans, admit shades of gray. This ability to represent degrees of truth makes such tools very powerful for representing vague or imprecise ideas. We can now say, for example, that the tolerance on one capacitor is tighter than that on another or one program runs faster than another and not be concerned about specific values.

Organizing the Book

It is often all too easy to hack together a one-off crisp logic application that appears to work. Trying to replicate a million or more copies of such a design (with elastic timing constraints, variable path impedance, or flexible data values) very quickly runs into the real-world gremlins that are waiting for us. A solid, secure, robust, reliable design must always be based on the proven underlying theory, a thorough problem analysis, and a disciplined development approach. Such methods are growing increasingly important as we continue to push the envelope of designs that are impacting the daily lives of an ever-increasing number of people.

This book takes a developer's perspective to first refreshing the basics of classic or crisp logic, teaching the concept of fuzzy logic, then applying such concepts to approximate reasoning systems such as threshold logic and perceptrons. This book examines, in detail, each of the important theoretical and practical aspects that one must consider when designing today's applications.

These applications must include the following:

1) The formal hardware and software development process (stressing safety, security, and reliability)
2) The digital and software architecture of the system
3) The physical world interface to external analog and digital signals
4) The debug and test throughout the development cycle and finally
5) Improving the system performance

The Chapters

Introduction and Background

The Introduction gives an overview of the topics covered in the book. These topics include some of the vocabulary that is part of the fuzzy logic, threshold logic, and perceptron worlds. The Introduction also includes a bit of background and history, applications where such tools can be used, and a few contemporary examples.

History and Infrastructure

With the preliminary background set, the next two chapters introduce some of the early work that provided the foundation for fuzzy logic, the reasoning process for solving problems, and a brief review of the essentials of classic or crisp logic.

Chapter 1 presents some of the early views on reality, learning, logic, and reasoning that founded the first classic laws of thought that ultimately laid the foundations for fuzzy logic. Working from these fundamentals, the chapter introduces and discusses the basic mathematics and set theory underlying crisp and fuzzy logic and examines the similarities and differences between the two forms of logic. The chapter concludes with the introduction and study of fuzzy membership functions.

Chapter 2 opens with an introduction of the fundamental concepts of crisp logic underlying a classic algebra or algebraic system. The study follows with a review of the basics of Boolean algebra. We then introduce the concept and purpose of a truth table and demonstrate algebraic proofs using such tables. We then learn that the entries in such a table are called minterms and that a minterm is a binary aggregate of logical 0s and 1s that sets the logical value, true or false, of single cell entries in truth tables.

Next the K-Map is introduced and reviewed as a pictorial tool for grouping logical expressions with shared or common factors. Such sharing enables the elimination of unwanted variables thereby simplifying a logical expression. These studies introduce and teach the groundwork for relaxing the precision of classic logic and the concepts and tools similar to those that we'll apply and work with in the worlds of fuzzy logic, threshold logic, and perceptrons.

Sets, Sets, and More Sets

Building on the work of those who opened the path and set the trail for us, the next two chapters introduce and study the fundamental concepts, properties, and operations of sets and set membership first for classic sets and then for fuzzy sets.

Chapter 3 introduces the fundamental concept of sets, focusing on what are known as classical or crisp sets. The chapter begins with an introduction of some of the elementary vocabulary and terminology and then reviews the principle definitions and concepts of the theory of ordinary or classical sets. The concepts of *subsets* and *set membership* are then presented and explored. *Set membership* naturally leads to the concept of *membership functions*.

With the fundamentals of sets and set membership established, we study the basic theory of classic or crisp logic. We then move to the details of the properties and logical operations of using crisp sets and of developing crisp membership applications. Crisp sets and crisp membership applications are a prelude to the introduction of *fuzzy logic*, *fuzzy sets*, fuzzy *set membership,* and threshold logic.

Chapter 4 moves to the fuzzy world introducing and focusing on what are termed fuzzy sets. The chapter reviews some of the principle definitions and concepts of the theory of ordinary or classical sets and illustrates how these are identical to fuzzy subsets when the degree of membership in the subset is expanded to include all real numbers in the interval [0.0, 1.0]. We learned that vagueness and imprecision are common in everyday life. Very often, the kind of information we encounter may be placed into two major categories: statistical and nonstatistical.

The fundamental fuzzy terminology is presented and followed by the introduction of the basic fuzzy set properties and applications. With properties and applications understood, the focus shifts to membership functions and the grade of membership. Up to this point, data has been expressed in numerical form. Often a graphical presentation is a more effective and convenient tool. Such graphs can be expressed in both linear and curved graphical format.

Linguistic Variables and Hedges

Chapter 5, we begin this chapter with a look at early symbols and sounds and their evolution to language and knowledge. Building on such origins, we introduce the concept of *sets* and move into the worlds of formal s*et theory*, *Boolean algebra* and introduce *crisp variables*.

From the crisp world, we migrate into the fuzzy world and introduce the concept and term *linguistic variable* as a variable whose values are words or phrases in a natural (or synthetic) language rather than real numbers. Such words are interpreted as representing labels on fuzzy subsets within a universe of discourse, which refers to a collection of entities that are currently being discussed, analyzed, or examined.

We learn that the values for a linguistic variable are generated from a set of primary terms, a collection of modifiers called *hedges*, and a collection of *connectives*. Hedges affect the value of a linguistic variable by either concentrating or diluting the membership distribution of the primary terms. We learn that concentrating or diluting a membership distribution can be very clearly represented graphically as discussed in Chapter 4.

We conclude with a discussion of the purpose, creation and use, and manipulation of hedges.

Fuzzy Inference and Approximate Reasoning

Chapter 6 introduces the concepts of fuzzy inference and approximate reasoning. As part of developing an effective reasoning methodology, we introduce and demonstrate the fundamental concepts and various relationships of equality, containment and entailment, conjunction and disjunction, and union and intersection among sets and subsets.

We stress that for engineers, reasoning is an essential and significant element of effective design and the successful solution to problems. We also introduce and demonstrate a variety of relationships between and among fuzzy sets and subsets as an important part of that whole process.

We conclude with the inference rules, modus ponens and modus tollens, which govern deduction and abduction, the forward and backward reasoning processes.

Chapter 7 introduces and extends the crisp world development process into the fuzzy world. The first step effectively applies the fuzzification process to extend crisp world data and basic knowledge of fuzzy world fundamentals into the domain of fuzzy system design and development. The fuzzy design process is then illustrated through its application in the design and development of several basic systems. The closing step, called defuzzification, is then applied to return crisp data to the real world.

Doing the Work

Chapter 8, we open the chapter with a summary of the growing strengths and major attractions of fuzzy logic. We identify two of the major attractions. The first attraction is fuzzy logic's ability to facilitate expressing problems in linguistic terms. The second is its ability to support applications in the areas where a designer can encounter vagueness and where the numerical mathematical model of a system may be too complex or impossible to build using conventional techniques.

We then introduce and present a formal approach to fuzzy logic design. We point out that a fundamental and essential key to any successful design process is that knowing and understanding the problem and recognizing potential hazards that might occur during the design process. We stress that without such knowledge, one cannot and should not proceed with a design until such issues are rectified. In addition, we recommend periodic design reviews throughout the design process and stress the performance of *failure modes*, *effects*, and *criticality analysis* (FMECA) testing.

We then introduce and walk through the execution of the major steps in a formal design methodology. In particular, we refer to the requirements and design documentation in Appendix A and the Unified Modeling Language tools and touch on text in Appendix B.

Introducing Threshold Logic

Chapter 9, as we now move forward, we introduce and explore two tools that build on and extend the knowledge gained from crisp and fuzzy logic: threshold logic and perceptrons. The incorporated features ultimately make important contributions to the foundation of advanced tools called neural networks, machine learning, and artificial intelligence.

Threshold logic brings into the world of digital logic design the ability to alter the value of input signals using what are designated as weights. The logic also introduces the capability to set a threshold that the summed weighted inputs need to exceed to assert a true output. Although effective with most logical operations, the threshold devices fail at the implementation of the exclusive OR.

Moving to Perceptron Logic

Chapter 10, we open the chapter with an introduction to the architecture of the threshold logic device, which is a basic building block at the heart of what is called a *perceptron*. The perceptron is also known as an *artificial neuron*. Starting with a high-level model of the basic biological neuron, we introduce the vocabulary describing the elementary components of the device.

From there, we move to the *McCulloch–Pitts* (MCP) artificial neuron and illustrate possibilities of implementing the fundamental logic devices using the MCP model. We then bring in the concept of weights from threshold logic and discuss, develop, and implement an MCP neuron network. The next step is to introduce and present the basic perceptron. We walk through each of the major functional blocks from inputs to output in the basic perceptron and also point out potential implementation and operational concerns and possible solutions of which to be aware.

We then present and describe perceptron learning and develop the learning rule. The chapter concludes with a presentation of the essential steps for testing the perceptron.

The Appendices

Two supporting appendices are included. The first provides an introduction to the preparation of formal Requirements Specifications and Design Specifications. The second provides a tutorial of the Unified Modeling Language (UML) and associated tools and some important issues to consider in the testing process.

Appendix A introduces the traditional product development life cycle and stresses the need to thoroughly understand both the operating environment and the system being designed. Further stressed is the difference between a *requirements specification* and a *design specification*. Design is only one element of product development. Each design must also be thoroughly tested to confirm and ensure that it meets specified requirements and national and international standards within its operating environment.

Appendix B introduces and provides an overview of the Unified Modeling Language and associated diagrams. The major static diagrams and the utility of each are presented and discussed. Of particular importance are the *class* and *use case* diagrams. The need for dynamic modeling is also presented. In addition, important test considerations are introduced.

This appendix also restresses the need for serious system test and the test process. It also provides a brief outline and summary of the key elements and components of a design that potentially can cause (serious) problems during operation and therefore should be considered and closely and carefully examined during the test process.

The Audience

The book is intended for students with a broad range of background and experience and also serves as a reference text for those working in the fields of electrical engineering and computer science. The core audience should have at least one quarter to one semester of study in Boolean algebra and crisp logic design, facility with a high-level programming language such as C, C++, or Java, and some knowledge of logic devices and operating systems, and should be an upper-level junior or senior or lower-level graduate student. Some background in formal system design, test, embedded systems, and analog fields would also be helpful.

Notes to the Instructor

This book can be a valuable tool for students in the traditional undergraduate electrical engineering, computer engineering, or computer science programs as well as for practicing engineers who also wish to review the basic concepts in these programs. Here, students may study the essential aspects of the development of contemporary fuzzy, neural, and approximate reasoning systems.

Students are also given a solid presentation of hardware and software architecture fundamentals, a good introduction to the design process and the formal methods used therein (including safety, security, and reliability). They are also given a comprehensive presentation of the interface to local and distributed external world devices and guidance on how to debug and test their designs.

Key to the presentation is a substantial number of worked examples illustrating fundamental ideas as well as how some of the subtleties in application go beyond basic concepts. Each chapter opens with a list of *Things to Look For* that highlights the more important material in the chapter and concludes with review questions and thought questions.

The review questions are based directly on material covered in the chapter and mirror and expand on the *Things to Look For* list. The questions provide students a self-assessment of their understanding and recall of the material covered. Though based on the material covered in the chapter, the thought questions extend the concepts as well as provide a forum in which students can explore, discuss, and synthesize new ideas based on those concepts with colleagues.

The text is written and organized much as one would develop a new system, i.e. from the top-down, building on the basics. Ideas are introduced and then revisited throughout the text, each time to a greater depth or in a new context.

Safety, security, and reliability are absolutely essential components in the development of any kind of system and system design process today. Such material is not an integral component in this text but should be stressed as companion technology.

As we stated in the opening of this Preface, finding a good balance between depth and breadth in today's approximate reasoning systems is a challenge. To that end, a couple of decisions were made at the outset. First, the text is not written around a specific microprocessor or software language. Rather, the material is intended to be relevant to (and has been used to develop) a wide variety of applications running on many different kinds of processors. Second, the artificial intelligence field and approaches to approximate reasoning are rapidly changing even as this sentence is being typed and read. In lieu of trying to pursue and include today's latest technologies, the focus is on the basics that apply to any of the technologies. The underlying philosophy of this book is that the student is well-grounded in the fundamentals will be comfortable working with and developing state-of-the-art systems using the newest ideas. Ohm's law hasn't changed for many years; the fields of computer science and electrical and computer engineering have and are.

The core material has been taught as a one-quarter, senior-level course in fuzzy logic development in several universities around the world. Based on student background, the text is sufficiently rich to provide material for a two-to-three-quarter or two-semester course in fuzzy logic and approximate reasoning systems development at the junior to senior level in a traditional four-year college or university engineering program.

Beyond the core audience, the sections covering the assumed foundation topics can provide a basis on which the student with a limited hardware or software background can progress through the remainder of the material. The logic and software sections are not sufficiently deep to replace the corresponding one- or two-quarter courses in the topics. For those with adequate background in such areas, the material can either be skipped or serve as a brief refresher.

Acknowledgments

Over the years, as I've collected the knowledge and experiences to bring this book together, there have been many, many people with whom I have studied, worked, and interacted. Our discussions, debates, and collaborations have led to the ideas and approach to design presented on the pages that follow.

While there are far too many to whom I owe a debt of thanks to try to list each here, I do want to give particular thanks to David L. Johnson, Corrine Johnson, Greg Zick, Tom Anderson, David Wright, Gary Anderson, Patrick Donahoo, Dave Bezold, Steve Swift, Paul Lantz, Mary Kay Winter, Steve Jones, Kasi Bhaskar, Brigette Huang, Jean-Paul Calvez, Gary Whittington, Olivier Pasquier, Charles Staloff, Gary Ball, John Davis, Patrick F. Kelly, Margaret Bustard, William and Betty Peckol, and Donna Karschney for all they've done over the years. William Hippe, Alex Talpalatskiy, and my brother William Peckol, who have all spent many hours proofreading, commenting, and making valuable suggestions to improve the early, working versions of the text, deserve a special thank you. From John Wiley, I want to thank Sandra Grayson who supported the original idea of publishing this text and especially Louis Manoharan (Project Editor) and Kanchana Kathirvelu (Production Editor), both of whom helped to guide the manuscript through the editing and production phases, and the unknown copyeditors, compositors, and others whose efforts on and contributions to this project have been invaluable.

In any project, design reviews are an essential part of producing a quality product. I wish to express my appreciation and thanks to this project's many reviewers for their evaluations and constructive comments, which helped guide its development.

I want to extend special thanks to my family William, Suzanne, Joe, Paulette, Karl, and Lori, and to my daughters Erin and Robyn, and grandchildren Kyleen, Jordan, and Tara.

Finally, I extend a thank you to my many teachers, friends, colleagues, and students who I've had the pleasure of knowing and working with over the years.

About the Author

James K. Peckol, Ph.D., is a Principal Lecturer Emeritus in the Department of Electrical and Computer Engineering at the University of Washington, Seattle, USA, where he has been named Teacher of the Year three times and Outstanding Faculty twice. He is also the founder of Oxford Consulting, Ltd., a product design and development consulting firm. The author is a member of *Tau Beta Pi, Who's Who in the World, Who's Who in Science and Engineering*, and has been presented with the Marquis *Who's Who* Lifetime Achievement Award.

His background spans over 50 years as an engineer and educator in the fields of software, digital, medical, and embedded systems design and development. Also, the author has published first and second editions of the book *Embedded Systems: A Contemporary Design Tool*.

As an engineer in the aerospace, commercial, and medical electronics industries, the author has worked on and contributed to the design and development of test systems for military aircraft navigation systems and radar systems, the *Apollo* color camera, various weather satellites, the Mars *Viking Lander*, flight control systems for a number of commercial aircraft, production of high-quality electronic test instruments and measurement systems, and several defibrillation systems. Academic experience spans more than 25 years of developing and teaching software, digital design, fuzzy logic, approximate reasoning, networking, and embedded systems design courses for students from academe and industry with experience ranging from limited hardware or software backgrounds to those at the junior, senior, graduate, and industrial levels.

Introduction

THINGS TO LOOK FOR...

- The topics that will be covered in the book.
- Important first steps when beginning a new design.
- Some of the strengths, applications, and advantages of fuzzy logic.
- The differences between crisp and fuzzy logic.
- Is fuzzy logic a newly developed technology?
- Who is using fuzzy logic?
- Should fuzzy logic be used for all designs?
- Should a fuzzy system be implemented in hardware or software?
- Where might tools called perceptrons and threshold logic be used?
- What should we do after we have designed and built the hardware and firmware for our system?

I.1 Introducing Fuzzy Logic, Fuzzy Systems, and • • • • •

This section begins with some personal philosophy about fuzzy logic, fuzzy systems, and other devices. It begins with what fuzzy logic is and presents some introductory questions and answers about the field. It compares and contrasts fuzzy logic with the traditional classic or crisp logic and concludes with a high-level view of the basic design and development process and potential applications for both approaches. In addition to fuzzy logic, this text provides a brief review of classical logic and presents and discusses the related areas of threshold logic and artificial neurons also termed perceptrons.

I.2 Philosophy

The chapters ahead bring us into the interesting world of threshold logic, fuzzy logic, and perceptrons. Welcome. The approach and views on solving engineering problems presented in this work reflect things that I have learned throughout my career. Yours will probably be a bit different, particularly as you learn and develop your skills and as the technology evolves. My approach has been augmented by the views and approaches of many very creative engineers, scientists, mathematicians, and philosophers dating back centuries.

We see here the two main themes that will be interwoven through each of the chapters ahead. With each new design, our first look should be from the outside. What are we designing? How will people use it – *what is its behavior*? What effect will it have on its operating environment – *what are the outputs*? What will be the effect of its operating environment on it – *what are its inputs*? How well do we have to do the job – *what are the constraints*?

We want to look at the high-level details first and then go on to the lower. We can borrow the idea of the public interface (an outside view) to our system from our colleagues working on object-centered designs. Our first view should be of the public interface to our system – we should view it from the outside and then move to the details inside.

I.3 Starting to Think Fuzzy – Fuzzy Logic Q&A

We'll open this book by introducing fuzzy logic. Along our path, we use, design, and develop tools, techniques, and knowledge from just about every other discipline in electrical engineering and computing science.

OK, let's start. Other than an interesting name, exactly what is fuzzy logic? Many people have heard something about fuzzy logic but are not quite sure what it is, what it means, or what's fuzzy about it. Let's try to answer those questions and several of the other more common ones.

Despite its amusing and seemingly contradictory name, fuzzy logic is not a logic that is fuzzy. Exactly the opposite is true. It is a way of capturing the vagueness and imprecision that are so common in our everyday languages and thinking. Capturing and representing such vagueness and imprecision in terms that a computer or learning system can understand and work with becomes the challenge.

Fuzzy logic can represent statements that are completely true or false, and it can also represent those that are partially true and/or partially false. Classical crisp logic lives in a black-and-white, "yes" and "no" world. Fuzzy logic admits shades of gray. Such an ability to represent degrees of truth using what are called hedges and linguistic variables makes fuzzy logic very powerful for representing vague or imprecise ideas.

I.4 Is Fuzzy Logic a Relatively New Technology?

Not really. Although fuzzy logic has been generating a lot of interest in this country recently, it is far from new. Lotfi Zadeh (1965), of the University of California at Berkeley, proposed many of the original ideas when he published his first famous research paper on fuzzy sets in 1965. Japanese companies have been using fuzzy logic for over 50 years. They have been granted over 2000 patents and have designed fuzzy logic into hundreds of products ranging from elevator and traffic control systems to video cameras and refrigerators.

One frequently cited example is a one-button washing machine. This machine senses the size of the load of clothes, the amount and type of dirt and then selects the proper quantity of soap, water level, water temperature, and washing time. Fuzzy logic has also been applied to the classic driverless truck-backer-upper problem and automatic flight control for helicopters.

I.5 Who Is Using Fuzzy Logic in the United States?

Companies in the United States utilizing fuzzy logic in contemporary designs include Eaton Industrial Controls, Motorola, NCR, Intel, Rockwell, Togai InfraLogic, NASA, Gensym, Allen-Bradley Co., General Electric, and General Motors. Some of the fields using fuzzy logic–related technologies include linear and nonlinear control, data analysis, pattern recognition, operations research, and financial systems.

The list of companies outside of the United States becoming involved in developing products that use fuzzy logic or in producing tools for designing fuzzy logic systems is growing at rapid rate.

I.6 What Are Some Advantages of Fuzzy Logic?

Fuzzy logic works very well in conjunction with other technologies. In particular, it provides accurate responses to ambiguous, imprecise, or vague data. Because fuzzy logic allows ideas to be expressed in linguistic terms, it offers a formal mathematical system for representing problems using familiar words.

As a result, fuzzy logic has proven to be a powerful and effective tool for modeling systems with uncertainties in their inputs or outputs or for use when precise models of a system are either unknown or extremely complex.

I.7 Can I Use Fuzzy Logic to Solve All My Design Problems?

Perhaps, however, you should not use fuzzy logic in those systems for which you already have a good or optimal solution using traditional methods. If there is a simple and clearly defined mathematical model for the system, use it. Fuzzy logic, like any other tool, must be used properly and carefully.

Fuzzy logic has been found to give excellent results in several general areas. The most common usage today is in systems for which complete or adequate models are difficult to define or develop and in systems or tasks that use human observations as input, control rules, or decision rules.

The Hitachi's control system for the Sendai subway near Tokyo and Matsushita's one-button washing machine are very good examples. A fuzzy logic approach also works well in systems that are continuous and complex and that have a nonlinear transfer function or in which vagueness is common.

I.8 What's Wrong with the Tools I'm Using Now?

Nothing. Fuzzy logic does not replace your existing tools; it gives you an additional one. Fuzzy logic simplifies the task of representing and working with vague, imprecise, or ambiguous information often common in human speech, ideas, or reasoning. It provides a means for solving a set of problems that have been difficult or impossible to solve using traditional methods. Consider working examples such as automatic flight control for a helicopter, or the precise management of the freezing of fish in a home freezer.

I.9 Should I Implement My Fuzzy System in Hardware or Software?

Either one. Good tools and solutions are available for hardware and software. However, today fuzzy logic is essentially a "software type" technology and should probably be considered and evaluated that way. This is good news to us as designers. It means that we should be able to take advantage of all that we've learned about developing good software and hardware systems and apply it to developing good fuzzy logic solutions.

As with software, continuous technological improvements will ensure a migration of fuzzy logic into hardware. Hardware solutions may certainly run faster, but software solutions are more flexible and do offer a hedge against the possible unavailability of parts. Such systems, as we'll see, also have the ability to learn. Whichever approach is finally selected, one still needs to go through the rigorous process of analyzing the product under development, making the same trade-offs we have always made, and thoroughly testing the resulting design and product for safety and reliability.

I.10 Introducing Threshold Logic

As we now move forward, we introduce and explore two tools that build on and extend the knowledge gained from crisp and fuzzy logic. These tools are called *threshold logic* and *perceptrons*. Threshold logic builds on and extends the capability of the traditional combinational logic gates. The incorporated features ultimately make important contributions to the foundation of advanced tools called *neural networks*, *machine learning,* and *artificial intelligence*.

I.11 Moving to Perceptron Logic

We move next to a very fascinating device called a *perceptron*. This device is also known as an *artificial neuron*. The perceptron incorporates capabilities from both fuzzy logic and threshold logic and includes the capability to learn. Starting with a high-level model, we introduce the vocabulary describing the elementary components and capabilities of the device. From there, we move to the *McCulloch–Pitts* (MCP) artificial neuron, which we will examine and then implement a basic model. We also illustrate implementing fundamental classic logic devices using the MCP model.