The Practical Handbook of GENETIC ALGORITHMS

Applications

SECOND EDITION

© 2001 by Chapman & Hall/CRC

The Practical Handbook of GENETIC ALGORITHMS

Applications

SECOND EDITION

Edited by
Lance Chambers

CHAPMAN & HALL/CRC

Boca Raton London New York Washington, D.C.

Library of Congress Cataloging-in-Publication Data

The practical handbook of genetic algorithms, applications / edited by
Lance D. Chambers.—2nd ed.
p. cm.
Includes bibliographical references and index.
ISBN 1-58488-2409-9 (alk. paper)
1. Genetic algorithms. I. Chambers, Lance.
QA402.5 .P72 2000
519.7—dc21 00-064500
CIP

This book contains information obtained from authentic and highly regarded sources. Reprinted material is quoted with permission, and sources are indicated. A wide variety of references are listed. Reasonable efforts have been made to publish reliable data and information, but the author and the publisher cannot assume responsibility for the validity of all materials or for the consequences of their use.

Neither this book nor any part may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, microfilming, and recording, or by any information storage or retrieval system, without prior permission in writing from the publisher.

All rights reserved. Authorization to photocopy items for internal or personal use, or the personal or internal use of specific clients, may be granted by CRC Press LLC, provided that $.50 per page photocopied is paid directly to Copyright Clearance Center, 222 Rosewood Drive, Danvers, MA 01923 USA. The fee code for users of the Transactional Reporting Service is ISBN 1-58488-240-9/01/$0.00+$.50. The fee is subject to change without notice. For organizations that have been granted a photocopy license by the CCC, a separate system of payment has been arranged.

The consent of CRC Press LLC does not extend to copying for general distribution, for promotion, for creating new works, or for resale. Specific permission must be obtained in writing from CRC Press LLC for such copying.

Direct all inquiries to CRC Press LLC, 2000 N.W. Corporate Blvd., Boca Raton, Florida 33431.

Trademark Notice: Product or corporate names may be trademarks or registered trademarks, and are used only for identification and explanation, without intent to infringe.

© 2001 by Chapman & Hall/CRC

No claim to original U.S. Government works
International Standard Book Number 1-58488-240-9
Library of Congress Card Number 00-064500
Printed in the United States of America 1 2 3 4 5 6 7 8 9 0
Printed on acid-free paper

Preface

Bob Stern of CRC Press, to whom I am indebted, approached me in late 1999 asking if I was interested in developing a second edition of volume I of the *Practical Handbook of Genetic Algorithms*. My immediate response was an unequivocal “Yes!” This is the fourth book I have edited in the series and each time I have learned more about GAs and people working in the field. I am proud to be associated with each and every person with whom I have dealt with over the years. Each is dedicated to his or her work, committed to the spread of knowledge and has something of significant value to contribute.

This second edition of the first volume comes a number of years after the publication of the first. The reasons for this new edition arose because of the popularity of the first edition and the need to perform a number of functions for the GA community. These “functions” fall into two main categories: the need to keep practitioners abreast of recent discoveries/learning in the field and to very specifically update some of the best chapters from the first volume.

The book leads off with chapter 0, which is the same chapter as the first edition by Jim Everett on model building, model testing and model fitting. An excellent “How and Why.” This chapter offers an excellent lead into the whole area of models and offers some sensible discussion of the use of genetic algorithms, which depends on a clear view of the nature of quantitative model building and testing. It considers the formulation of such models and the various approaches that might be taken to fit model parameters. Available optimization methods are discussed, ranging from analytical methods, through various types of hill-climbing, randomized search and genetic algorithms. A number of examples illustrate that modeling problems do not fall neatly into this clear-cut hierarchy. Consequently, a judicious selection of hybrid methods, selected according to the model context, is preferred to any pure method alone in designing efficient and effective methods for fitting parameters to quantitative models.

Chapter 1 by Roubos and Setnes deals with the automatic design of fuzzy rule-based models and classifiers from data. It is recognized that both accuracy and transparency are of major importance and we seek to keep the rule-based models small and comprehensible. An iterative approach for developing such fuzzy rule-based models is proposed. First, an initial model is derived from the data. Subsequently, a real-coded GA is applied in an iterative fashion, together with a rule-based simplification algorithm to optimize and simplify the model, respectively. The proposed modeling approach is demonstrated for a system identification and a classification problem. Results are compared to other

© 2001 by Chapman & Hall/CRC

approaches in the literature. The proposed modeling approach is more compact and interpretable.

Goldberg and Hammerham in Chapter 2, have extended their contribution to Volume III of the series (Chapter 6, pp 119–238) by describing their current research, which applies this technology to a different problem area, designing automata that can recognize languages given a list of representative words in the language and a list of other words not in the language. The experimentation carried out indicates that in this problem domain also, smaller machine solutions are obtained by the MTF operator than the benchmark. Due to the small variation of machine sizes in the solution spaces of the languages tested (obtained empirically by Monte Carlo methods), MTF is expected to find solutions in a similar number of iterations as the other methods. While SFS obtained faster convergence on more languages than any other method, MTF has the overall best performance based on a more comprehensive set of evaluation criteria.

Taplin and Qiu, in Chapter 3, have contibuted material that very firmly grounds GA in solving real-world problems by employing GAs to solve the very complex problems associated with the staging of road construction projects. The task of selecting and scheduling a sequence of road construction and improvement projects is complicated by two characteristics of the road network. The first is that the impacts and benefits of previous projects are modified by succeeding ones because each changes some part of what is a highly interactive network. The change in benefits results from the choices made by road users to take advantage of whatever routes seem best to them as links are modified. The second problem is that some projects generate benefits as they are constructed, whereas others generate no benefits until they are completed.

There are three general ways of determining a schedule of road projects. The default method has been used to evaluate each project as if its impacts and benefits would be independent of all other projects and then to use the resulting cost-benefit ratios to rank the projects. This is far from optimal because the interactions are ignored. An improved method is to use rolling or sequential assessment. In this case, the first year's projects are selected, as before, by independent evaluation. Then all remaining projects are reevaluated, taking account of the impacts of the first-year projects, and so on through successive years. The resulting schedule is still sub-optimal but better than the simple ranking.

Another option is to construct a mathematical program. This can take account of some of the interactions between projects. In a linear program, it is easy to specify relationships such as a particular project not starting before another specific project or a cost reduction if two projects are scheduled in succession. Fairly simple traffic interactions can also be handled but network-wide traffic effects have to be analysed by a traffic assignment model (itself a complex programming task). Also, it is difficult to cope with deferred project benefits. Nevertheless,

© 2001 by Chapman & Hall/CRC

mathematical programming has been used to some extent for road project scheduling.

The novel option, introduced in this chapter, is to employ a GA which offers a convenient way of handling a scheduling problem closely allied to the travelling salesman problem while coping with a series of extraneous constraints and an objective function which has at its core a substantial optimising algorithm to allocate traffic.

The authors from City University of Hong Kong are Zhang, Chung, Lo, Hui, and Wu. Their contribution, Chapter 4, deals with the optimization of electronic circuits. It presents an implementation of a decoupled optimization technique for the design of switching regulators. The optimization process entails selection of the component values in the regulator to meet the static and dynamic requirements. Although the proposed approach inherits characteristics of evolutionary computations that involve randomness, recombination, and survival of the fittest, it does not perform a whole-circuit optimization. Consequently, intensive computations that are usually found in stochastic optimization techniques can be avoided. In the proposed optimization scheme, a regulator is decoupled into two components, namely, the power conversion stage (PCS) and the feedback network (FN). The PCS is optimized with the required static characteristics such as the input voltage and output load range, whils"t the FN is optimized with the required static characteristics of the whole system and the dynamic responses during the input and output disturbances. Systematic procedures for optimizing circuit components are described. The proposed technique is illustrated with the design of a buck regulator with overcurrent protection. The predicted results are compared with the published results available in the literature and are verified with experimental measurements.

Chapter 5 by Hallinan discusses the problems of feature selection and classification in the diagnosis of cervical cancer. Cervical cancer is one of the most common cancers, accounting for 6% of all malignancies in women. The standard screening test for cervical cancer is the Papanicolaou (or "Pap") smear, which involves visual examination of cervical cells under a microscope for evidence of abnormality.

Pap smear screening is labour-intensive and boring, but requires high precision, and thus appears on the surface to be extremely suitable for automation. Research has been done in this area since the late 1950s; it is one of the "classical" problems in automated image analysis.

In the last four decades or so, with the advent of powerful, reasonably priced computers and sophisticated algorithms, an alternative to the identification of malignant cells on a slide has become possible.

The approach to detection generally used is to capture digital images of visually normal cells from patients of known diagnosis (cancerous/precancerous condition or normal). A variety of features such as nuclear area, optical density, shape and

© 2001 by Chapman & Hall/CRC

texture features are then calculated from the images, and linear discriminant analysis is used to classify individual cells as either "normal" or "abnormal." An individual is then given a diagnosis on the basis of the proportion of abnormal cells detected on her Pap smear slide.

The problem with this approach is that while all visually normal cells from "normal" (i.e., cancer-free) patients may be assumed to be normal, not all such cells from "abnormal" patients will, in fact, be abnormal. The proportion of affected cells from an abnormal patient is not known *a priori*, and probably varies with the stage of the cancer, its rate of progression, and possibly other factors. This means that the "abnormal" cells used for establishing the canonical discriminant function are not, in fact, all abnormal, which reduces the accuracy of the classifier. Further noise is introduced into the classification procedure by the existence of two more-or-less arbitrary cutoff values – the value of the discriminant score at which individual cells are classified as "normal" or "abnormal," and the proportion of "abnormal" cells used to classify a patient as "normal" or "abnormal."

GAs are employed to improve the ability of the system to discriminate and therefore enhance classification.

Chapter 6, dealing with "Algorithms for Multidimensional Scaling," offers insights into looking at the potential for using GAs to map a set of objects in a multidimensional space. GAs have a couple of advantages over the standard multidimensional scaling procedures that appear in many commercial computer packages. The most frequently cited advantage of Gas – the ability to avoid being trapped in a local optimum – applies in the case of multidimensional scaling. Using a GA or at least a hybrid GA, offers the opportunity to freely choose an appropriate objective function. This avoids the restrictions of the commercial packages, where the objective function is usually a standard function chosen for its stability of convergence rather than for its applicability to the user's particular research problem. The chapter details genetic operators appropriate to this class of problem, and uses them to build a GA for multidimensional scaling with fitness functions that can be chosen by the user. The algorithm is tested on a realistic problem, which shows that it converges to the global optimum in cases where a systematic hill-descending method becomes entrapped at a local optimum. The chapter also looks at how considerable computation effort can be saved with no loss of accuracy by using a hybrid method. For hybrid methods, the GA is brought in to "fine-tune" a solution, which has first been obtained using standard multidimensional scaling methods.

Chapter 7 by Lam and Yin describes various applications of GAs to transportation optimization problems. In the first section, GAs are employed as solution algorithms for advanced transport models; while in the second section, GAs are used as calibration tools for complex transport models. Both sections show that, similar to other fields, GAs provide an alternative powerful tool to a wide variety

© 2001 by Chapman & Hall/CRC

of problems in the transportation domain.

It is well-known that many decision-making problems in transportation planning and management could be formulated as bilevel programming models (single-objective or multi-objectives), that are intrinsically non-convex and it is thus difficult to find the global optimum. In the first example, a genetic-algorithms-based (GAB) approach is proposed to solve the single-objective models. Compared with the previous heuristic algorithms, the GAB approach is much simpler in principle and more efficient in applications. In the second example, the GAB approach to accommodate multi-objective bilevel programming models is extended. It is shown that this approach can capture a number of Pareto solutions efficiently and simultaneously which can be attributed to the parallelism and globality of GAs.

Varela, Vela, Puente, Gomez and Vidal in Chapter 8 describe an approach to solve job shop scheduling problems by means of a GA which is adapted to the problem in various ways. First, a number of adjustments of the evaluation function are suggested; and then it is proposed that a strategy to generate a number of chromosomes of the initial population allows the introduction of heuristic knowledge from the problem domain. In order to do that, the variable and value ordering heuristics proposed by Norman Sadeh are exploited. These are a class of probability-based heuristics which are, in principle, set to guide a backtracking search strategy. The chapter validates all of the refinements introduced on well known benchmarks and reports experimental results showing that the introduction of the proposed refinements has an accumulative and positive effect on the performance of the GA.

Chapter 9, developed by Raich and Ghaboussi, discusses an evolutionary-based method called the implicit redundant representation genetic algorithm (IRR GA) is applied to evolve synthesis design solutions for an unstructured, multi-objective frame problem domain. The synthesis of frame structures presents a design problem that is difficult, if not impossible, for current design and optimization methods to formulate, let alone search. Searching for synthesis design solutions requires the optimization of structures with diverse structural topology and geometry. The topology and geometry define the number and the location of beams and columns in the frame structure. As the topology and geometry change during the search process, the number of design variables also change. To support the search for synthesis design solutions, an unstructured problem formulation that removes constraints that specify the number of design variables is used. Current optimization methods, including the simple genetic algorithm (SGA), are not able to model unstructured problem domains since these methods are not flexible enough to change the number of design variables optimized. The unstructured domain can be modeled successfully using the location-independent and redundant IRR GA representation.

© 2001 by Chapman & Hall/CRC

The IRR GA uses redundancy to encode a variable number of location-independent design variables in the representation of the problem domain. During evolution, the number and locations of the encoded variables dynamically change within each individual and across the population. The IRR GA provides several benefits: redundant segments protect existing encoded design variables from the disruption of crossover and mutation; new design variables may be designated within previously redundant segments; and the dimensions of the search space dynamically change as the number of design variables represented changes. The IRR GA synthesis design method is capable of generating novel frame designs that compare favorably with solutions obtained using a trial-and-error design process.

Craenen, Eiben and Marchiori in Chapter 10 develop a contribution that describes evolutionary algorithms (EAs) for constraint handling. Constraint handling is not straightforward in an EA because the search operators mutation and recombination are "blind" to constraints. Hence, there is no guarantee that if the parents satisfy some constraints the offspring will satisfy them as well. This suggests that the presence of constraints in a problem makes EAs intrinsically unsuited to solve this problem. This should especially hold when the problem does not contain an objective function to be optimized, but only constraints – the category of constraint satisfaction problems. A survey of related literature, however, indicates that there are quite a few successful attempts to evolutionary constraint satisfaction. Based on this survey, the authors identify a number of common features in these approaches and arrive at the conclusion that EAs can be effective constraint solvers when knowledge about the constraints is incorporated either into the genetic operators, in the fitness function, or in repair mechanisms. The chapter concludes by considering a number of key questions on research methodology.

Chapter 11 provides a very valuable approach to fine-tuning fuzzy rules. The chapter presents the design of a fuzzy logic controller (FLC) for a boost-type power factor corrector. A systematic offline design approach using the genetic algorithm to optimize the input and output fuzzy subsets in the FLC is proposed. Apart from avoiding complexities associated with nonlinear mathematical modeling of switching converters, circuit designers do not have to perform time-consuming procedures of fine-tuning the fuzzy rules, which require sophisticated experience and intuitive reasoning as in many classical fuzzy-logic-controlled applications. Optimized by a multi-objective fitness function, the proposed control scheme integrates the FLC into the feedback path and a linear programming rule on controlling the duty time of the switch for shaping the input current waveform, making it unnecessary to sense the rectified input voltage. A 200-W experimental prototype has been built. The steady-state and transient responses of the converter under a large-signal change in the supply voltage and in the output load are investigated.

© 2001 by Chapman & Hall/CRC

In Chapter 12, Grundler, from the University of Zagreb describes a new method of complex process control with the coordinating control unit based upon a genetic algorithm. The algorithm for the control of complex processes controlled by PID and fuzzy regulators at the first level and coordinating unit at the second level has been theoretically laid out. A genetic algorithm and its application to the proposed control method have been described in detail. The idea has been verified experimentally and by simulation in a two-stage laboratory plant. Minimal energy consumption criteria limited by given process response constraints have been applied, and improvements in relation to other known optimizing methods have been made. Independent and non-coordinating PID and fuzzy regulator parameter tuning have been performed using a genetic algorithm and the results achieved are the same or better than those obtained from traditional optimizing methods while at the same time the method proposed can be easily automated. Multilevel coordinated control using a genetic algorithm applied to a PID and a fuzzy regulator has been researched. The results of various traditional optimizing methods have been compared with an independent non-coordinating control and multilevel coordinating control using a genetic algorithm.

Chapter 13 discusses GA approaches to cancer treatment. The aim of radiation therapy is to cure the patient of malignant disease by irradiating tumours and infected tissue, whilst minimising the risk of complications by avoiding irradiation of normal tissue. To achieve this, a *treatment plan*, specifying a number of variables, including beam directions, energies and other factors, must be devised. At present, plans are developed by radiotherapy physicists, employing a time-consuming iterative approach. However, with advances in treatment technology which will make higher demands on planning soon to be available in clinical centres, computer optimisation of treatment plan parameters is being actively researched. These optimisation systems can provide treatment solutions that better approach the aims of therapy. However, direct optimisation of treatment goals by computer remains a time-consuming and computationally expensive process. With the increases in the demand for patient throughput, a more efficient means of planning treatments would be beneficial. Previous work by Knowles (1997) described a system which employs artificial neural networks to devise treatment plans for abdominal cancers. Plan parameters are produced instantly upon input of seven simple values, easily measured from the CT-scan of the patient. The neural network used in Knowles (1997) was trained with fairly standard backpropagation (Rumelhart et al., 1986) coupled with an adaptive momentum scheme. This chapter focuses on later work in which the neural network is trained using evolutionary algorithms. Results show that the neural network employing evolutionary training exhibits significantly better generalisation performance than the original system developed. Testing of the evolutionary neural network on clinical planning tasks at Royal Berkshire Hospital in Reading, UK, has been carried out. It was found that the system can readily produce clinically useful treatment plans, considerably quicker than the

© 2001 by Chapman & Hall/CRC

human-based iterative method. Finally, a new neural network system for breast cancer treatment planning was developed. As plans for breast cancer treatments differ greatly from plans for abdominal cancer treatments, a new network architecture was required. The system developed has again been tested on clinical planning tasks at Royal Berkshire Hospital and results show that, in some cases, plans which improve on those produced by the hospital are generated.

For those of you who are well-entrenched in the field, there are authors that you will recognise as being some of the best; and for those of you who are new to Gas, the same will apply – these are names you will certainly come to know and respect. The contributors to this edition come from a cross-section of academia and industry – theoreticians and practitioners. All make a significant contribution to our understanding of and ability to use GAs.

One of the main objectives of the series has been to develop a work that will allow practitioners to take the material offered and use it productively in their own work. This edition maintains that objective. To that end, some contributors have also included computer code so that their work can be duplicated and used productively in your own endeavours. I will willingly e-mail the code to you if you send a request to lchambers@transport.wa.gov.au or it may be found on the CRC Press web site at **www.crcpress.com**.

The science and art of GA programming and application has come a long way in the last 5 years since the publication of the first edition. However, I consider GAs as still being a "new science" that has a long way to go before the bounds of the effects are well-defined and their ability to contribute in a meaningful manner to many fields of human endeavour are exhausted. We are, metaphorically, still "scratching the surface" of our understanding and applications of GAs. This book is designed to help scratch that surface just a little bit deeper and a little bit more.

As in the previous volumes, authors have come from countries around the world. In a world, which we are told is continually shrinking, it is pleasing to obtain first hand evidence of this shrinkage. As in the earlier volumes all communications were by e-mail which has dramatically sped up the whole process. But even so, a work of this nature invariably takes time.

The development of a chapter contribution to any field of serious endeavour is a task that must, of need, be taken on only after serious consideration and contemplation. I am happy to say that I believe all the authors contributing to this volume have gone through those processes and I believe that because of the manifest quality of the work presented.

© 2001 by Chapman & Hall/CRC

Lance Chambers
Perth, Western Australia
lchambers@transport.wa.gov.au

Note: I have not Americanised (sic) the spelling of English spelling contributors. So, as you read, you will find a number of words with s's where you may expect z's, and you may find a large number of u's where you might least expect them as in the word, "colour" and "behaviour." Please do not be perturbed. I believe the authors have the right to see their work in a form each recognises. I also have not altered the referencing forms used (we all understand the various forms and this should not detract from the book, but hopefully add some individuality) by the authors.

Ultimately, however, I am responsible for all alterations, errors and omissions.

© 2001 by Chapman & Hall/CRC

Contents

Chapter 0 Model Building, Model Testing and Model Fitting

0.1 Uses of Genetic Algorithms

0.1.1 Optimizing or Improving the Performance of Operating Systems

0.1.2 Testing and Fitting Quantitative Models

0.1.3 Maximizing vs. Minimizing

0.1.4 Purpose of this Chapter

0.2 Quantitative Models

0.2.1 Parameters

0.2.2 Revising the Model or Revising the Data?

0.2.3 Hierarchic or Stepwise Model Building: The Role of Theory

0.2.4 Significance and Meaningfulness

0.3 Analytical Optimization

0.3.1 An Example: Linear Regression

0.4 Iterative Hill-Climbing Techniques

0.4.1 Iterative Incremental Stepping Method

0.4.2 An Example: Fitting the Continents Together

0.4.3 Other Hill-Climbing Methods

0.4.4 The Danger of Entrapment on Local Optima and Saddle Points

0.4.5 The Application of Genetic Algorithms to Model Fitting

0.5 Assay Continuity in a Gold Prospect

0.5.1 Description of the Problem

0.5.2 A Model of Data Continuity

0.5.3 Fitting the Data to the Model

0.5.4 The Appropriate Misfit Function

0.5.5 Fitting Models of One or Two Parameters

0.5.6 Fitting the Non-homogeneous Model 3

0.6 Conclusion

Reference

© 2001 by Chapman & Hall/CRC

Chapter 1 Compact Fuzzy Models and Classifiers through Model Reduction and Evolutionary Optimization

1.1 Introduction

1.2 Fuzzy Modeling

1.2.1 The Takagi-Sugeno Fuzzy Model

1.2.2 Data-Driven Identification by Clustering

1.2.3 Estimating the Consequent Parameters

1.3 Transparency and Accuracy of Fuzzy Models

1.3.1 Rule Base Simplification

1.3.2 Genetic Multi-objective Optimization

1.4 Genetic Algorithms

1.4.1 Fuzzy Model Representation

1.4.2 Selection Function

1.4.3 Genetic Operators

1.4.4 Crossover Operators

1.4.5 Mutation Operators

1.4.5.1 Constraints

1.5 Examples

1.5.1 Nonlinear Plant

1.5.2 Proposed approach

1.6 TS Singleton Model

1.7 TS Linear Model

1.7.1 Iris Classification Problem

1.7.2 Solutions in the literature

1.7.3 Proposed Approach

1.8 Conclusion

References

Chapter 2 On the Application of Reorganization Operators for Solving a Language Recognition Problem

2.1 Introduction

2.1.1 Performance across a New Problem Set

2.1.2 Previous Work

2.2 Reorganization Operators

2.2.1 The Jefferson Benchmark

© 2001 by Chapman & Hall/CRC

2.2.2 MTF
2.2.3 SFS
2.2.4 Competition
2.3 The Experimentation
2.3.1 The Languages
2.3.2 Specific Considerations for the Language Recognition Problem
2.4 Data Obtained from the Experimentation
2.5 General Evaluation Criteria
2.6 Evaluation
2.6.1 Machine Size
2.6.2 Convergence Rates
2.6.3 Performance of MTF
2.7 Conclusions and Further Directions
References

Chapter 3 Using GA to Optimise the Selection and Scheduling of Road Projects

3.1 Introduction
3.2 Formulation of the Genetic Algorithm
3.2.1 The Objective
3.2.2 The Elements of the Project Schedule
3.2.3 The Genetic Algorithm
3.3 Mapping the GA String into a Project Schedule and Computing the Fitness
3.3.1 Data Required
3.3.2 Imposing Constraints
3.3.3 Calculation of Project Benefits
3.3.4 Calculating Trip Generation, Route Choice and Link Loads
3.4 Results
3.4.1 Convergence of Solutions to the Problem
3.4.2 The Solutions
3.4.3 Similarity and Dissimilarity of Solutions: Euclidean Distance
3.5 Conclusions: Scheduling Interactive Road Projects by GA
3.5.1 Dissimilar Construction Schedules with High and Almost Equal Payoffs
3.5.2 Similar Construction Schedules with Dissimilar Payoffs

© 2001 by Chapman & Hall/CRC

References

Chapter 4 Decoupled Optimization of Power Electronics Circuits Using Genetic Algorithms

4.1 Introduction

4.2 Decoupled Regulator Configuration

4.2.1 Optimization Mechanism of GA

4.2.2 Chromosome and Population Structures

4.2.3 Fitness Functions

4.3 Fitness Function for PCS

4.3.1 OF_1 for Objective (1)

4.3.2 OF_2 for Objective (2)

4.3.3 OF_3 for Objective (3)

4.3.4 OF_4 for Objective (4)

4.4 Fitness function for FN

4.4.1 OF_5 for Objective (1)

4.4.2 OF_6 and OF_8 for Objective (2) and Objective (4)

4.4.3 OF_8 of Objective (3)

4.5 Steps of Optimization

4.6 Design Example

4.7 Conclusions

References

Chapter 5 Feature Selection and Classification in the Diagnosis of Cervical Cancer

5.1 Introduction

5.2 Feature Selection

5.3 Feature Selection by Genetic Algorithm

5.3.1 GA Encoding Schemes

5.3.2 GAs and Neural Networks

5.3.3 GA Feature Selection Performance

5.3.4 Conclusions

5.4 Developing a Neural Genetic Classifier

5.4.1 Algorithm Design Issues

5.4.2 Problem Representation

© 2001 by Chapman & Hall/CRC

5.4.3 Objective Function
5.4.4 Selection Strategy
5.4.5 Parameterization
5.5 Validation of the Algorithm
5.5.1 The Dataset
5.5.2 Experiments on Two-Dimensional Data
5.5.3 Results of Two-Dimensional Data Experiments
5.5.4 Lessons from Artificial Data
5.5.5 Experiments on a Cell Image Dataset
5.6 Parameterization of the GA
5.6.1 Parameterization Experiments
5.6.2 Results of Parameterization Experiments
5.6.3 Selecting the Neural Network Architecture
5.7 Experiments with the Cell Image Dataset
5.7.1 Slide-Based vs. Cell-Based Features
5.7.2 Comparison with the Standard Approach
5.7.3 Discussion
References

Chapter 6 Algorithms for Multidimensional Scaling

6.1 Introduction
6.1.1 Scope of This Chapter
6.1.2 What is Multidimensional Scaling?
6.1.3 Standard Multidimensional Scaling Techniques
6.2 Multidimensional Scaling Examined in More Detail
6.2.1 A Simple One-Dimensional Example
6.2.2 More than One Dimension
6.2.3 Using Standard Multidimensional Scaling Methods
6.3 A Genetic Algorithm for Multidimensional Scaling
6.3.1 Random Mutation Operators
6.3.2 Crossover Operators
6.3.3 Selection Operators
6.3.4 Design and Use of a Genetic Algorithm for Multidimensional Scaling
6.4 Experimental Results
6.4.1 Systematic Projection

© 2001 by Chapman & Hall/CRC

6.4.2 Using the Genetic Algorithm
6.4.3 A Hybrid Approach
6.5 The Computer Program
6.5.1 The Extend Model
6.5.2 Definition of Parameters and Variables
6.5.3 The Main Program
6.5.4 Procedures and Functions
6.5.5 Adapting the Program for C or C++
6.6 Using the Extend Program
References

Chapter 7 Genetic Algorithm-Based Approach for Transportation Optimization Problems

7.1 GA-Based Solution Approach for Transport Models
7.1.1 Introduction
7.1.2 GAB Approach for Single-Objective Bilevel Programming Models
7.1.3 GAB Approach for Multi-Objective Bilevel Programming Models
7.1.4 Summary
7.2 GAB Calibration Approach for Transport Models
7.2.1 Introduction
7.2.2 Review of TFS
7.2.3 Calibration Measures
7.2.4 GAB Calibration Procedure
7.2.5 Calibration of TFS
7.2.6 Case Study
7.2.7 Summary
7.3 Concluding Remarks
References
Appendix I: Notation

Chapter 8 Solving Job-Shop Scheduling Problems by Means of Genetic Algorithms

8.1 Introduction
8.2 The Job-Shop Scheduling Constraint Satisfaction Problem
8.3 The Genetic Algorithm

© 2001 by Chapman & Hall/CRC

8.4 Fitness Refinement

8.4.1 Variable and Value Ordering Heuristics

8.5 Heuristic Initial Population

8.6 Experimental Results

8.7 Conclusions

References

Chapter 9 Applying the Implicit Redundant Representation Genetic Algorithm in an Unstructured Problem Domain

9.1 Introduction

9.2 Motivation for Frame Synthesis Research

9.2.1 Modeling the Conceptual Design Process

9.2.2 Research in Frame Optimization

9.3 The Implicit Redundant Representation Genetic Algorithm

9.3.1 Implementation of the IRR GA Algorithm

9.3.2 Suitability of the IRR GA in Conceptual Design

9.4 The IRR Genotype/Phenotype Representation

9.4.1 Provision of Dynamic Redundancy

9.4.2 Controlling the Level of Redundancy in the IRR GA Initial Population

9.5 Applying the IRR GA to Frame Design Synthesis in an Unstructured Domain

9.5.1 Unstructured Design Problem Formulation

9.5.2 IRR GA Genotype/Phenotype Representation for Frame Design Synthesis

9.5.3 Use of Repair Strategies on Frame Design Alternatives

9.5.4 Generation of Horizontal Members in Design Synthesis Alternatives

9.5.5 Specification of Loads on Unstructured Frame Design Alternatives

9.5.6 Finite-Element Analysis of Frame Structures

9.5.7 Deletion of Dynamically Allocated Nodal Linked Lists

9.6 IRR GA Fitness Evaluation of Frame Design Synthesis Alternatives

9.6.1 Statement of Frame Design Objectives Used as Fitness Functions

9.6.2 Application of Penalty Terms in IRR GA Fitness Evaluation

9.7 Discussion of the Genetic Control Operators Used by the IRR GA

9.7.1 Fitness Sharing among Individuals in the Population

9.7.2 Tournament Selection of New Population Individuals

© 2001 by Chapman & Hall/CRC

9.7.3 Multiple Point Crossover of Binary Strings

9.7.4 Single-Bit Mutation of Binary Strings

9.8 Results of the Implicit Redundant Representation Frame Synthesis Trials

9.8.1 Evolved Design Solutions for the Frame Synthesis Unstructured Domain

9.8.2 Synthesis versus Optimization of Frame Design Solutions Using IRR GA

9.9 Concluding Remarks

References

Chapter 10 How to Handle Constraints with Evolutionary Algorithms

10.1 Introduction

10.2 Constraint Handling in EAs

10.3 Evolutionary CSP Solvers

10.3.1 Heuristic Genetic Operators

10.3.2 Knowledge-Based Fitness and Genetic Operators

10.3.3 Glass-Box Approach

10.3.4 Genetic Local Search

10.3.5 Co-evolutionary Approach

10.3.6 Heuristic-Based Microgenetic Method

10.3.7 Stepwise Adaptation of Weights

10.4 Discussion

10.5 Assessment of EAs for CSPs

10.6 Conclusion

References

Chapter 11 An Optimized Fuzzy Logic Controller for Active Power Factor Corrector Using Genetic Algorithm

11.1 Introduction

11.2 FLC for the Boost Rectifier

11.2.1. Switching Rule for the Switch SW

11.2.2 Fuzzy Logic Controller (FLC)

11.2.3 Defuzzification

11.3 Optimization of FLC by the Genetic Algorithm

11.3.1 Structure of the Chromosome

11.3.2 Initialization of S_i

© 2001 by Chapman & Hall/CRC

11.3.3 Formulation of Multi-objective Fitness Function
11.3.4 Selection of Chromosomes
11.3.5 Crossover and Mutation Operations
11.3.6 Validation of S_I: Recovery of Valid Fuzzy Subsets
11.4 Illustrative Example
11.5 Conclusions
References
Chapter 12 Multilevel Fuzzy Process Control Optimized by Genetic Algorithm
12.1 Introduction
12.2 Intelligent Control
12.3 Multilevel Control
12.3.1 Optimal Control Concept
12.3.2 Process Stability during Genetic Algorithm Optimizing
12.3.3 Optimizing Criteria
12.4 Optimizing Aided by Genetic Algorithm
12.4.1 Genetic Algorithm Parameters
12.5 Laboratory Cascaded Plant
12.6 Multilevel Control Using Genetic Algorithm
12.6.1 Non-coordinated Multilevel Control Using a PID Controller
12.7 Fuzzy Multilevel Coordinated Control
12.7.1 Decision Control Table
12.8 Conclusions
References
Chapter 13 Evolving Neural Networks for Cancer Radiotherapy
13.1 Introduction and Chapter Overview
13.2 An Introduction to Radiotherapy
13.2.1 Radiation Therapy Treatment Planning (RTP)
13.2.2 Volumes
13.2.3 Treatment Planning
13.2.4 Recent Developments and Areas of Active Research
13.2.5 Treatment Planning
13.3 Evolutionary Artificial Neural Networks

© 2001 by Chapman & Hall/CRC

13.3.1 Evolving Network Weights
13.3.2 Evolving Network Architectures
13.3.3 Evolving Learning Rules
13.3.4 EPNet
13.3.5 Addition of Virtual Samples
13.3.6 Summary
13.4 Radiotherapy Treatment Planning with EANNs
13.4.1 The Backpropogation ANN for Treatment Planning
13.4.2 Development of an EANN
13.4.3 EANN Results
13.4.4 Breast Cancer Treatment Planning
13.5 Summary
13.6 Discussion and Future Work
Acknowledgments
References

© 2001 by Chapman & Hall/CRC

Figures

Figure 0.1 Simple linear regression

Figure 0.2 Iterative incremental stepping method

Figure 0.3 Fitting contours on the opposite sides of an ocean

Figure 0.4 Least misfit for contours of steepest part of continental shelf

Figure 0.5 The fit of the continents around the Atlantic

Figure 0.6 Entrapment at a saddle point

Figure 0.7 Cumulative distribution of gold assays, on log normal scale

Figure 0.8 Assay continuity

Figure 0.9 Log correlations as a function of r, the inter-assay distance

Figure 0.10 Correlations as a function of r, the inter-assay distance

Figure 0.11 Fitting model 0: $\rho(r) = a$

Figure 0.12 Fitting model 1: $\rho(r) = \exp(-kr)$

Figure 0.13 Fitting model 2: $\rho(r) = a.\exp(-kr)$

Figure 0.14 Comparing model 0, model 1 and model 2

Figure 0.15 Fit of model 3 using systematic projection

Figure 0.16 Fit of model 3 using the genetic algorithm

Figure 1.1 Example of a linguistic fuzzy rule

© 2001 by Chapman & Hall/CRC

Figure 1.2 Fuzzy sets are defined by fitting parametric functions (solid lines) to the projections (dots) of the point-wise defined fuzzy sets in the fuzzy partition matrix U

Figure 1.3 Transparency of the fuzzy rule base premise

Figure 1.4 Similarity-driven simplification

Figure 1.5 Two modeling schemes with multi-objective GA optimization

Figure 1.6 Input $u(k)$, unforced system $g(k)$, and output $y(k)$ of the plant in (Equations 15 and 16)

Figure 1.7 Initial fuzzy sets and fuzzy sets in the reduced model

Figure 1.8 Local singleton models and the response surface

Figure 1.9 Simulation of the six-rule TS singleton model and error in the estimated output

Figure 1.10 Local linear TS-model derived in five steps: (a) initial model with ten clusters, (b) set merging, (c) GA-optimization, (d) set-merging, (e) final GA optimization

Figure 1.11 Simulation of the six-rule TS singleton model and the error in the estimated output

Figure 1.12 Local linear TS model and the response-surface

Figure 1.13 Iris data: *setosa* (×), *versicolor* (O), and *virginica* (∇)

Figure 1.14 Initial fuzzy rule-based model with three rules and 33 misclassifications

Figure 1.15 Optimized fuzzy rule-based model with three rules and three misclassifications (Table 1.3-B)

Figure 1.16 Optimized and reduced fuzzy rule-based model with three rules and four misclassifications (Table 1.3-E)

Figure 2.1 16-state/148-bit FSA genome (G1) map

© 2001 by Chapman & Hall/CRC

Figure 2.2 Outline of the Jefferson benchmark GA. The two inserts will be extra steps used in further sections as modifications to the original algorithm

Figure 2.3 An example of the crossover used

Figure 2.4 An example of the mutation operator used

Figure 2.5 Outline of the MTF operator

Figure 2.6 Four tables depiction of MTF algorithm on a four-state FSM genome

Figure 2.7 Outline of the SFS operator

Figure 2.8 Standardization formula for SFS algorithm (Step 2b, Figure 2.7)

Figure 2.9 Pictorial description of Figure 2.8 for max_num_states = 32

Figure 2.10 Table depiction of SFS algorithm on a four-state FSM genome

Figure 2.11 Outline of competition procedure

Figure 2.12 16-state/148-bit FSA genome (G2) map

Figure 2.13 Table of parameters for the languages

Figure 2.14 The seeds used to initialize the random number generator for each run

Figure 2.15 Number of generations required to find a solution

Figure 2.16 Number of generations required to find a solution

Figure 2.17 Minimal number of states found in a solution

Figure 2.18 Minimal number of states found in a solution

Figure 2.19 Rankings of methods for each language based on machine size

© 2001 by Chapman & Hall/CRC

Figure 2.20 Recommendations of methods for each language based on efficiency

Figure 2.21 Recommendations of languages for each method based on efficiency

Figure 3.1 The genetic algorithm for the road project construction timetable problem

Figure 3.2 Relationship between the timetable analysis period and project sub-periods

Figure 3.3 Procedure for calculation of the objective function value

Figure 3.4: Comparison of the Steps in the Improvement of the Objective Function Values of the best individuals over GA generations in ten experiments

Figure 3.5 Euclidean distance between two vectors in a *R3* space

Figure 3.6 Hypothetical superior solutions and surrounding inferior solutions

Figure 4.1 Block diagram of power electronics circuits: chromosome structures and the fitness functions

Figure 4.2 Objective functions

Figure 4.3 Typical transient response of v_d

Figure 4.4 Flowchart of the optimization steps of PCS

Figure 4.5 Reproducion process

Figure 4.6 Buck regulator with overcurrent protection

Figure 4.7 Φ_p and Φ_F vs. the number of generation *gen*

Figure 4.8 Simulated start-up transients when v_{in} is 20 V and R_L is 5 Ω

Figure 4.9 Experimental start-up transients when v_{in} is 20 V and R_L is 5 Ω

© 2001 by Chapman & Hall/CRC

Figure 4.10 Simulated start-up transients when v_{in} is 60 V and R_L is 5 Ω

Figure 4.11 Experimental start-up transients when v_{in} is 60 V and R_L is 5 Ω

Figure 4.12 Simulated transient responses when v_{in} is changed from 20 V to 40 V

Figure 4.13 Experimental transient responses when v_{in} is changed from 20 V into 40 V

Figure 4.14 Simulated transient responses when $\mathrm{R_L}$ is changed from 5 Ω to 10 Ω and $\mathrm{v_{in}}$ is 40 V

Figure 4.15 Experimental transient responses when R_L is changed from 5 Ω to 10 Ω and v_{in} is 40 V

Figure 4.16 Simulated transient responses when R_L is changed from 10 Ω to 5 Ω and v_{in} is 40 V

Figure 4.17 Experimental transient responses when R_L is changed from 10 Ω to 5 Ω and v_{in} is 40 V

Figure 5.1 Automated diagnosis from digital images

Figure 5.2 Architecture of the neural network

Figure 5.3 Organization of a chromosome coding for a simple three-layer neural network

Figure 5.4 Two dimensional training data

Figure 5.5 ROC curves for 2-D data: select 2 from 7 features, training set

Figure 5.6 ROC curves for 2-D data: select 2 from 7 features, test set

Figure 5.7 Performance of a “good” classifier (Run 1) compared with that of a “poor” classifier (Run 3) on training and validation data

Figure 5.8 Histogram of cell nuclear area

© 2001 by Chapman & Hall/CRC

Figure 5.9 Correlation of AUC on the training data with maximum fitness for the parameterization experiments

Figure 5.10 The presence of abnormal cells shifts the distribution of a feature measured across all cells on a slide

Figure 5.11 ROC curves for test on train results

Figure 5.12 ROC curves for test on test results

Figure 5.13 ROC curves for test on train results

Figure 5.14 ROC curves for test on test results

Figure 5.15 Generalizability of the MACs classifiers

Figure 6.1 Global and local optima for the one-dimensional example

Figure 6.2 Misfit function (Y) for the one-dimensional example

Figure 6.3 Projected mutation

Figure 6.4 The genetic algorithm control panel

Figure 6.5 Systematic projection from ten random starting configurations

Figure 6.6 Genetic algorithm using the same ten random starting configurations

Figure 6.7 Starting from Eigen vectors and from the Alscal solution

Figure 6.8 The Extend model

Figure 6.9 The Extend simulation setup screen

Figure 7.1 Example network 1

Figure 7.2 Demand multiplier versus generation number

Figure 7.3 Example network 2

Figure 7.4 Pareto optimal solutions

© 2001 by Chapman & Hall/CRC

Figure 7.5 Flowchart of GAB calibration algorithm

Figure 7.6 Tuen Mun corridor network

Figure 7.7 Integral network cost vs. perception error coefficient

Figure 7.8 Total trip cost vs. perception error coefficient

Figure 7.9 Link choice entropy vs. perception error coefficient

Figure 7.10 Path choice entropy vs. perception error coefficient

Figure 7.11 NCV vs. OD variation coefficient

Figure 7.12 Path choice entropy vs. perception error coefficient in the pilot tests

Figure 7.13 NCV vs OD variation coefficient in the pilot tests

Figure 7.14 Maximum fitness vs population size, generation, length of chromosome

Figure 7.15 Maximum fitness vs. crossover probability and mutation probability

Figure 7.16 Fitness vs perception error coefficient in the TFS calibration

Figure 7.17 Fitness vs OD variation coefficient in the TFS calibration

Figure 8.1 A JSS problem instance with three jobs

Figure 8.2 (a) Scheduling produced by the fitness1 strategy to the problem of Figure 8.1 from the individual (3 3 1 1 1 2 2 2). The fitness1 value is 13. (b) Scheduling produced from the same individual by the fitness2 strategy. The fitness2 value is 11

Figure 8.3 Results of convergence of six versions of the GA

Figure 8.4 Results about convergence of four versions of the GA along 1000 generations

© 2001 by Chapman & Hall/CRC

Figure 8.5 Comparison of various versions of the GA in solving the FT10 problem instance

Figure 9.1 C++ code for *main()* function that implements the IRR GA

Figure 9.2 *SIndividual* data structure used for the population individuals

Figure 9.3 Comparison of generic IRR GA and SGA genotype representations

Figure 9.4 Dynamic redundancy provided by the IRR GA compared to the SGA

Figure 9.5 Models of structured and unstructured frame design problem formulations

Figure 9.6 Definition of design variables encoded in the IRR GA genotype

Figure 9.7 SNodeData structure for storing design variables

Figure 9.8 Definition of *SaveNodes()* function called by *EvaluateBinary()*

Figure 9.9 Definition of *CreateNodeForList()* and *slsStore()* called by *SaveNodes()*

Figure 9.10 Assembly of complete structure from design variables

Figure 9.11 Linked lists of *SNodeData* structures for frame structure defined in Figure 9.10

Figure 9.12 Definition of *SStructure* and *SNode* data structure for frame alternatives

Figure 9.13 *EvaluateBinary()* code segment for structures with less than two supports

Figure 9.14 Code segment for *EvaluateBinary()* and function *DeleteSingleNode()*

Figure 9.15 *EvaluateBinary()* code segment and function *MakeSameNodes()*

© 2001 by Chapman & Hall/CRC

Figure 9.16 Common list functions called by *DeleteSingleNode()* and *MakeSameNodes()*

Figure 9.17 Implementation of *CreateHorzMembers()*

Figure 9.18 *SLoadVector* data structure for structural loads and forces

Figure 9.19 Application of alternating span live loading to an example structure

Figure 9.20 Implementation of *SetGravityLoad()*

Figure 9.21 Application of wind loading to the exterior nodes of two example structures

Figure 9.22 *SetWL()* applies wind loading in each direction to frame structures

Figure 9.23 Deletion of arrays of linked lists created dynamically by the IRR GA program

Figure 9.24 Implementation of *CalcVolumeFitness()* and *CalcFloorFitness()*

Figure 9.25 Code segment of *CalcHorzDeflPenalty()*

Figure 9.26 Implementation of *CalcVertDeflPenalty()*

Figure 9.27 Implementation of *CalcNodeSymPenalty()*

Figure 9.28 Code segment from *SelectString()* implementing tournament selection

Figure 9.29 *CrossoverBinary()* code to set the number and location of multiple crossover sites

Figure 9.30 Frame design solutions for four trials represented by the fittest population individual of each IRR GA trial

Figure 9.31 Individuals in top 25% of the population ranked by fitness after one generation

© 2001 by Chapman & Hall/CRC

Figure 9.32 Individuals in top 25% of the population after 50 generations

Figure 9.33 Individuals in top 25% of the population after 200 generations

Figure 9.34 Maximum fitness and average fitness of the IRR GA population over 500 generations for a single trial

Figure 11.1 Block diagram of the boost rectifier with APFC and FLC

Figure 11.2 Behavioral model of the APFC

Figure 11.3 Structure of the fuzzy subsets and chromosomes

Figure 11.4 Inference method

Figure 11.5 Flowcharts

Figure 11.6 Typical output response of the boost rectifier

Figure 11.7 Crossover and mutation operations

Figure 11.8 Validation of S_i

Figure 11.9 GA-trained membership functions

Figure 11.10 Steady-state experimental waveforms when $R_L = 110\ \Omega$

Figure 11.11 Transient responses when R_L is changed from 110 Ω to 220 Ω

Figure 11.12 Transient responses when R_L is changed from 220 Ω to 110 Ω

Figure 11.13 Transient responses when v_{in} is changed from 110 V to 90 V

Figure 11.14 Transient responses when v_{in} is changed from 90 V to 130 V

Figure 11.15 Transient output and control voltages when v_{in} is changed from 90 V to 130 V (Ch 1: output voltage (100 V/div); Ch2: control voltage (2 V/div); Timebase: 20 ms/div)

© 2001 by Chapman & Hall/CRC

Figure 12.1 Block diagram of a coordinate control concept

Figure 12.2 Block diagram of laboratory plant

Figure 12.3 Photo of laboratory plant

Figure 12.4 Block diagram of laboratory plant

Figure 12.5 Block diagram of the first stage of plant

Figure 12.6 Block diagram of the second stage of plant

Figure 12.7 Block diagram of the connecting tube

Figure 12.8 First process stage response for Zeigler-Nichols and GA tuned PID, controller for step input q_{k1u} from $q_{k1u} = 0.5$ l/min to $q_{k1u} = 1.0$ l/min

Figure 12.9 Second process stage response for Ziegler-Nicholos and GA tuned PID_2 controller for step input q_{k1u} from $q_{k1u} = 0.5$ l/min to $q_{k1u} = 1.0$ l/min

Figure 12.10 First stage response to step disturbance q_{k1u} (from $q_{k1u} = 0.5$ l/min to $q_{k1u} = 1.0$ l/min) controlled with genetic algorithm tuned decision tables

Figure 12.11 First stage response to step disturbance q_{k1u} (from $q_{k1u} = 0.5$ l/min to $q_{k1u} = 0.2$ l/min) controlled with genetic algorithm tuned decision tables

Figure 12.12 Second stage response to step disturbance q_{k1u} (from $q_{k1u} = 0.5$ l/min to $q_{k1u} = 1.0$ l/min) controlled with genetic algorithm tuned decision tables

Figure 12.13 Second stage response to step disturbance q_{k1u} (from $q_{k1u} = 0.5$ l/min to $q_{k1u} = 0.2$ l/min) controlled with genetic algorithm-tuned decision tables

Figure 12.14 Comparison of energy consumption for both stages, at different input step disturbances

© 2001 by Chapman & Hall/CRC

Figure 12.15 Comparison of cumulative energy consumption for both stages of the laboratory plant for total of six steps input disturbances

Figure 12.16 Response of the first stage of a plant controlled by fuzzy controllers (decision tables are GA-tuned) for set point $T_r = 37°C$

Figure 12.17 Response of the second stage of a plant controlled by fuzzy controllers (decision tables are GA tuned) for set point $T_r = 64.4°C$

Figure 12.18 Behavior of the first stage of a plant controlled by fuzzy controllers (decision tables are GA tuned) for set point $T_r = 28.6°C$

Figure 12.19 Behavior of the second stage of a plant controlled by fuzzy controllers (decision tables are GA tuned) for set point $T_r = 47.5°C$

Figure 12.20 First stage response with nonlinear characteristic of thyristor converter

Figure 12.21 Second stage response with nonlinear characteristic of thyristor converter

Figure 12.22 First stage process response for various optimizing criteria

Figure 12.23 Second stage process response for various optimizing criteria

Figure 13.1 A schematic showing a typical beam setup for treatment of a prostate cancer

Figure 13.2 The Philips multi-leaf collimator

Figure 13.3 A typical plot of the dose to a target volume plotted on a dose-volume histogram

Figure 13.4 A cost function vs. gantry angle plot with the allowed gantry-angle-windows also displayed

Figure 13.5 A typical routine for evolution of connection weights. (From X. Yao, 1996.)

Figure 13.6 A typical cycle of the evolution of architectures. (From X. Yao, 1996.)

© 2001 by Chapman & Hall/CRC

Figure 13.7 A typical cycle of the evolution of learning rules. (From X. Yao, 1996.)

Figure 13.8 Input measurements taken from a patient's CT-scan for input to the neural network. Inputs 1, 2, and 3 are lengths and inputs 4, 5, and 6 are angles

Figure 13.9 Neural network architecture showing inputs and outputs (some connection lines are not shown)

Figure 13.10 Encoding of the connection weights on a chromosome

Figure 13.11 A plot of training set error and validation set error against generation for the EANN

Figure 13.12 A plot of training set error and validation set error against epoch for SAM

© 2001 by Chapman & Hall/CRC

Tables

Table 1.1 Singleton TS fuzzy models for the dynamic plant

Table 1.2 Linear TS fuzzy models for the dynamic plant

Table 1.3 Fuzzy rule-based classifiers for the Iris data derived by means of scheme 1 (A,B,C) and scheme 2 (D,E,F)

Table 2.1 Four-state FSM with start state Q13

Table 2.2 FSM with of Table 2.1 after Step 1 of MTF

Table 2.3 FSM of Table 2.2 after Next States for Q0 reassigned

Table 2.4 FSM of Table 2.1 after MTF

Table 2.5 Four-state FSM with start state Q13

Table 2.6 FSM with of Table 2.5 after Step 1 of SFS

Table 2.7 FSM of Table 2.6 after Next States for Q0 Reassigned

Table 2.8 FSM of Table 2.5 after SFS

Table 3.1 Details of road projects proposed for the rural road network in the Pilbara and adjoining regions in Western Australia

Table 3.2 Effects of a project on travel time (TT) on link *i*

Table 3.3 Vehicle travel time on link *i* in year *t*: TTi(t)

Table 3.4 Values of the best ten GA I\individuals in each of experiments 1 and 2

Table 3.5 Summary of the best ten investment sequences

© 2001 by Chapman & Hall/CRC

Table 3.6 Project sequence for the best solution converted to annual investment

Table 3.7 Road project construction timetable determined by the best solution

Table 3.8 Euclidean distances between the best ten solutions

Table 3.9 Differences between solutions: Euclidean distance and program similarities

Table 3.10 Comparison of project implementation in the best and second best solutions (Euclidean distance = 4.99)

Table 4.1 Parameters in GA optimization

Table 4.2(a) Initial values of *L* and *C* and the results after 500 generations

Table 4.2(b) Initial component values for the controller and the results after 500 generations

Table 5.1 Variables in the 2-D artificial data set

Table 5.2 Two-dimensional data: Selecting two features from seven

Table 5.3 Performance of run 3 with early stopping

Table 5.4 Description of BCCA dataset.

Table 5.5 Parameterization of the genetic algorithm

Table 5.6 Performance of slide-based and cell-based classifiers at various operating points

Table 5.7 Confusion matrix for stepwise linear discriminant analysis at operating point X

Table 5.8 Confusion matrix for best GA/NN at operating point Y

Table 5.9 Performance of the GA/NN and SLDA at the QC and PS operating points

© 2001 by Chapman & Hall/CRC

Table 6.1 An example data matrix of inter-object distances d_{ij}

Table 6.2 Inter-city flying mileages

Table 7.1 Input data for example network 1

Table 7.2 Solutions with alternative algorithms

Table 7.3 Input data for example network

Table 7.4 Pareto optimal solutions

Table 7.5 OD matrix (passenger car units per hour)

Table 7.6 The link data of the network

Table 8.1 Individual and aggregate demands of the initial state of the problem of Figure. 8.1 for all tasks and resources over the time intervals

Table 8.2 Survivabilities of all ten tasks in the initial state of the problem of Figure 8.1 over the time intervals

Table 8.3 Comparison of six versions of the GA against the *ORR & FSS* heuristics

Table 8.4 Comparison of the heuristic strategies to generate individuals

Table 9.1 Values of scalar constants for calculating the fitness and penalty function

Table 10.1 Specific features of three implemented versions of H-GA

Table 10.2 Specific features of Arc-GA

Table 10.3 Main features of Glass-Box GA

Table 10.4 Main features of the GLS algorithm

Table 10.5 Main features of the co-evolutionary algorithm

Table 10.6 Main features of heuristic-based microgenetic algorithm

© 2001 by Chapman & Hall/CRC

Table 10.7 Main features of the SAW-ing algorithm

Table 12.1 Comparison of optimizing results of PID controllers

Table 12.2 49-element control decision table

Table 12.3 Comparison of energy consumption for fuzzy controllers

Table 12.4 Decision control table tuned by genetic algorithm for the first process

Table 12.5 Decision control table tuned by genetic algorithm for the second process

Table 13.1 Summary of EANN training times

Table 13.2 Comparison of SAM and EANN generalisation performance

Table 13.3 Summary of EANN and SAM generalisation performance

Table 13.4 Best validation set errors at various training set errors for EANN and SAM

Table 13.5 Best validation set errors at various low training set errors for EANN and SAM

Table 13.6 Summary of breast cancer treatment plans produced by the EANN

© 2001 by Chapman & Hall/CRC

Genetic Algorithms is an advanced topic. Even though the content has been prepared keeping in mind the requirements of a beginner, the reader should be familiar with the fundamentals of Programming and Basic Algorithms before starting with this tutorial. Genetic algorithm is an optimization technique, which tries to find out such values of input so that we get the best output values or results.Â Introduction to Genetic Algorithm & their application in data science. Shubham Jain, July 31, 2017. Introduction.