DOWNLOAD PDF RIGOROUS DEVELOPMENT OF COMPLEX FAULT-TOLERANT SYSTEMS (LECTURE NOTES IN COMPUTER SCIENCE)

Chapter 1 : Papers and Reports on Layered Queueing

*Rigorous Development of Complex Fault-Tolerant Systems (Lecture Notes in Computer Science) [Michael Butler, Cliff Jones, Alexander Romanovsky, Elena Troubitsyna] on calendrierdelascience.com *FREE* shipping on qualifying offers. This book brings together 19 papers focusing on the application of rigorous design techniques to the development of fault-tolerant.*

The continuous sophistication of video games represents a stimulating challenge for Artificial Intelligence researchers. As part of our work on improving the behaviour of military units in Real-Time Strategy Games we are testing different techniques and methodologies for computer-controlled players. In this paper Evolutionary Programming is explored, establishing a first approach to develop an automatic controller for the classic maze chase game Ms. After the initial training, we evaluate the results obtained by all these combinations, identifying best choices and discussing the performance improvement that can be obtained with similar techniques in complex games. In this context, different techniques and methodologies for computer-controlled players as part of that research project are being tested. Some of these machines are very complex, having multiple parameters that should be adjusted by experimentation playtesting and by experienced game designers. We have decided to choose a simple game to start trying different techniques and methodologies as it is the case of 2 Evolutionary Programming. If significant results are found, we could study their applicability for improving the more complex FSMs of RTS games. For this paper we have developed an automatic controller for a version of the game Ms. The behaviour of the protagonist Ms. Pac-Man in the game is implemented by a FSM with some variable values that act as a threshold to control the state transitions. A genetic algorithm is used to find the best values for the performance of this FSM, aiming to probe the utility of this approach improving the game score. The structure of this paper is as follows: Section 2 is focused on the definitions of the concepts and the background needed to understand the rest of the paper. Section 3 presents our computer-controlled player for the Ms. Section 4 describes how we apply evolutionary computation to improve the performance score of the player and Section 5 shows and discusses the results of the experiments. Finally, in Section 6 we present our conclusions, foreseen the next steps of this project. Before applying evolutionary programming to complex games, we are performing experiments using more simple games, such as Pac-Man. Produced by Namco, it has been consider as an icon since its launch, not only for the videogames industry, but for the twentieth century popular culture [3]. There are also randomly-located fruits in the maze that add extra points to the score when eaten. The game is over when Pac-Man loses three lifes. The punctuation in the score grows exponentially after each ghost been eaten during this period. This version is slightly faster than the original game and, in contrast with the first one, the ghosts do not have a deterministic behaviour, their path through the maze is not predefined [4]. This makes the game more difficult, being much more challenging the creation of strategies to avoid being killed. Over the years there have been several proposals in the academy in relation to using AI for maze chase games as Pac-Man, both for controlling the protagonist or the antagonists the game. Ghosts League was a competition for developing completely automated controllers for Ms. Pac-Man with the usual goal of optimiz- 3 ing the score. With respect to Evolutionary Computing, genetic algorithms were originally developed by Cramer [5] and popularized by Koza [6] among others. This paradigm has become a large field of study, being widely-used in AI challenges and for optimizing the behaviour of intelligent automata. These algorithms could be useful for optimizing controllers in video games, even in sophisticated titles using autonomous agents [7]; but for our purposes, a controlled and limited scenario as the simple mazes of Ms. Pac-Man is perfect to test the methodology for evaluating the performance of a computer-controlled player. The controller implemented in this class is one of the simplest of the framework and it has been changed to transform it in a simple FSM with just three states: Pac-Man goes to the closest pill in order to eat it. In case there were several pills at the same distance, it will follow a preference order according to the direction toward these and clockwise starting from the top.

Pac-Man runs away from the closest ghost. In the implementation of this FSM we use four numerical variables that later on will compose the chromosome of the individuals of the population that the genetic algorithm will be using: The minimum distance a ghost should be to start running away from him. The minimum number of pills that should stay in the level to start going toward them proactively instead of hiding from or eating ghosts. Using these variables, the FSM has these transition rules: Pac-Man automatic player controller is explained, we perform the study to test if genetic algorithms can improve the FSM in terms of performance in the game. In this case the punctuation of the game itself will be used, so the game is executed with the parameters generated by the algorithm and average values are calculated after a constant number of game sessions played by a phenotype set in The average score from a set of played games acts as the fitness function for our algorithm. These genes are real numbers that take values between 0 and 1. The value of the genes are multiplied by in order to evaluate the results. Indeed the FSM of Ms. Pac-Man controller needs values between 0 and So on, every individual of the population represents a Ms. We have implemented two selection operators, six crossovers, two mutations and four substitutions also called regrouping , in order to perform different tests and determine which combination of operators get the best results. These are our two types of selection: Individuals are ordered in a list according to their fitness. The probability for an individual to be chosen for the crossover is higher as higher is its average score. N individuals of the population are selected randomly. Among these individuals the one with the better fitness value is selected. These are our six types of crossover: The parental chromosomes genes are interchanged from a given gene position. The parental chromosomes genes are interchanged from two given positions. Two progenitors take part and two new descendants are created. A binary mask determines the division of the genes which are going to be crossed. N descendants are generated. The value of the gene of the descendant in the position i is chosen randomly in a range defined by the genes of the progenitors that are located in the same position. This is a generalization of the plain one, called BLX-alpha. The value of the gene of the position i in the descendant is chosen randomly from an interval. Two new descendants are generated according to an arithmetic operation. The value of the gene i in the descendant X is the result of the operation being A and B the progenitors and Ai the value of the gene i of the chromosome A: And the value of the gene i in the descendant Y is the result of the operation: These are our two types of mutations: A gene is randomly selected and it mutates. The value of the gene is replaced by another randomly generated. Two genes are selected and they interchanged positions. These are our four types of substitution: The descendants replace the individuals with worse fitness from all the population. The individuals that are going to be substituted are randomly chosen. Groups of N individuals are selected and the worst of each group is replaced by a descendant. The descendants replace their own parents. At the beginning, the population is initialized with a certain number of individuals by default , all of them created with random genes. Each one is evaluated before it is added to the population structure a tree structure is used to maintain the population ordered by the fitness value of each individual. Then, the minimum, maximum and average fitness of this population is calculated and the next generation is produced. This process is repeated several times generations of individuals are created by default. Instead of testing all the possible combinations 96 different experiments and studying the interactions between operators one by one, as a first exploration of the problem we have taken a different approach. The graphics which are displayed below represent the results of the experiments. The X axis represents the number of the generation produced and the Y axis the values obtained in this generation. The Worst and Random substitution 7 Fig. Tournament and Generational substitution As can be seen in the results, the substitution operator of the Worst was the one with better results in this algorithm. The random one is a little inconsistent, with big variations and making worse the average fitness it is even worst: The substitution by Tournament seems to work quite well but in other experiments, not shown in these graphs, it never improves the best individual. Finally, the generational replacement just does not improve anything. Therefore, the Substitution of the Worst operator is chosen as the best operator in its category. Then, the different mutation operators are tested. The combination with the uniform mutation has already been tested in the previous step so we only have to test mutation by

interchange. Mutation by interchange 8 This method of mutation slightly improves the Uniform mutation operator, both in the average and maximum fitness where it gets some important jumps. Therefore we chose this method. One-point-based and Multipoint-based Crossover Fig. Plain and Combined Crossover 9 Fig. Arithmetic Crossover The One-Point-based operator get better results than the Uniform sometimes, but others it does not improve. The Multi-Point-based improves the results much more. The plain crossover produces few hops improvements in the best individual but these hops are very large, more than in the Multi-Point. The combined crossover also produces good results but, again, it does not produce as good improvements as the plain. Finally the arithmetic crossover produced a great improvement in the worse individuals and thus the average fitness , however it does not improve the best. Therefore the Plain crossover is selected for the remaining experiments.

DOWNLOAD PDF RIGOROUS DEVELOPMENT OF COMPLEX FAULT-TOLERANT SYSTEMS (LECTURE NOTES IN COMPUTER SCIENCE)

Chapter 2 : Lecture Notes in Computer Science | Awards | LibraryThing

PDF DOWNLOAD Read Rigorous Development of Complex Fault-Tolerant Systems (Lecture Notes in Computer Science / Programming and Software Engineering) () PDF Online Edition to the Documentary Film Directed by Raoul Peck (Vintage International) BOOK ONLINE PDF FREE DOWNLOAD I Am Not.

Box , Doha, Qatar E-mail: All rights are reserved, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, re-use of illustrations, recitation, broadcasting, reproduction on microfilms or in any other way, and storage in data banks. Duplication of this publication or parts thereof is permitted only under the provisions of the German Copyright Law of September 9, , in its current version, and permission for use must always be obtained from Springer. Violations are liable to prosecution under the German Copyright Law. The use of general descriptive names, registered names, trademarks, etc. This series of conferences has been held annually since and has become one of the premier international conferences in the areas of neural networks. Over the past few decades, the neural information processing community has witnessed tremendous efforts and developments from all aspects of neural information processing research. These include theoretical foundations, architectures and network organizations, modeling and simulation, empirical study, as well as a wide range of applications across different domains. Recent developments in science and technology, including neuroscience, computer science, cognitive science, nano-technologies, and engineering design, among others, have provided significant new understandings and technological solutions to move neural information processing research toward the development of complex, large-scale, and networked brain-like intelligent systems. This long-term goal can only be achieved with continuous efforts from the community to seriously investigate different issues of the neural information processing and related fields. To this end, ICONIP provided a powerful platform for the community to share their latest research results, to discuss critical future research directions, to stimulate innovative research ideas, as well as to facilitate multidisciplinary collaborations worldwide. ICONIP received tremendous submissions authored by scholars coming from 60 countries and regions across six continents. Based on a rigorous peerreview process, where each submission was evaluated by at least two reviewers, about high-quality papers were selected for publication in the prestigious series of Lecture Notes in Computer Science. These papers cover all major topics of theoretical research, empirical study, and applications of neural information processing research. One was on Challenges and Promises in Computational Intelligence with panelists: We highly appreciate all the organizers of special sessions and workshop for their tremendous efforts and strong support. Our conference would not have been successful without the generous patronage of our sponsors. We are most grateful to our platinum sponsor: We would also like to take this opportunity to express our deepest gratitude to the members of the Program Committee and all reviewers for their professional review of the papers. Furthermore, we would also like to thank Springer for publishing the proceedings in the prestigious series of Lecture Notes in Computer Science. We would, moreover, like to express our heartfelt appreciation to the keynote, plenary, panel, and invited speakers for their vision and discussions on the latest Preface VII research developments in the field as well as critical future research directions, opportunities, and challenges. Finally, we would like to thank all the speakers, authors, and participants for their great contribution and support that made ICONIP a huge success. Kamel, and Mohamed A.

DOWNLOAD PDF RIGOROUS DEVELOPMENT OF COMPLEX FAULT-TOLERANT SYSTEMS (LECTURE NOTES IN COMPUTER SCIENCE)

Chapter 3 : Department of Computer Science < Colorado State University

This paper examines the hypothesis that rigorous fault tolerance can be achieved by using aspect oriented software development in conjunction with formal methods of verification and analysis. After brief summaries on fault tolerance, aspect-oriented.

Hillston, LNCS , pp. Tauseef Israr, Murray Woodside, Greg Franks, "Interaction tree algorithms to extract effective architecture and layered performance models from traces", Journal of Systems and Software, v 80 n 4, April , pp Pan, "Efficient performance models for layered server systems with replicated servers and parallel behavior", Journal of Systems and Software, v 80, no 4, April , pp Tregunno, "Layered bottlenecks and their mitigation", Proc of 3rd Int. Lecture Notes in Computer Science: Champeau, Eds , pp. Litoiu, "The use of optimal filters to track parameters of performance models";, Proc. Dorin Bogdan Petriu, Murray Woodside, "Software performance models from system scenarios", Performance Evaluation, v 61 issue 1 pp Kim, Ra and C. Jarvis, Ligang He, Daniel P. Spooner and Graham R. Van de Meerssche K. Romanovsky, pp , Dec Woodside, "Analyzing the Effectiveness of fault-management architectures in layered distributed systems", Performance Evaluation, v 56 pp 93 - , Mar. Jarvis, Ligang He and Graham R. LQ has difficulty with cache size. An especially simple model for just two layers. Workshop on Software and Performance, Rome, July Lecture Notes in Computer Science , pp. Bayarov, "Automated performance modeling of software generated by a design environment", Performance Evaluation, v 45, pp - , July El-Sayed, Don Cameron, C. PDF for a version with some corrections C. Transactions on Software Engineering, Vol. Lecture Notes in Computer Science Vol.

Chapter 4 : Instant Messaging in Java: The Jabber Protocols - Book Pdf Djvu

Rigorous Development Of Complex Fault Tolerant Systems Lecture Notes In Computer Science Programming And Software Engineering ebooks and guide Handbook Of Frauds Scams And Swindles Failures Of Ethics In Leadership.

Chapter 5 : Lecture Notes in Computer Science: - calendrierdelascience.com

Julien ChiÃ¨ze a quelque chose Ã vous dire sur les notes dans le jeu vidÃ©o.

Chapter 6 : Lecture Notes in Computer Science PDF

Moreover, it enables a formal representation of essential abstractions used in the development of fault tolerant agent systems, including scopes, roles, locations, and agents. Application of the proposed approach results in designing fault tolerant agent systems in which inter-consistency and inter-operability of agents is ensured by construction.

Chapter 7 : Rigorous Development of Complex Fault-Tolerant Systems - ePrints Soton

Lecture Notes in Computer Science Rigorous Development of Complex Fault-Tolerant Systems Towards a Method for Rigorous Development of Generic Requirements.