Effective Black-Box Testing with Genetic Algorithms

Mark Last and Shay Eyal
Department of Information Systems Engineering
Ben-Gurion University of the Negev, Beer-Sheva, Israel

Abraham Kandel
National Institute for Systems Test and Productivity (NISTP)
University of South Florida, Tampa, FL, USA

IBM Verification Conference 2005

Agenda

- What is NISTP?
- Black-Box Testing: State-of-the-Art
- Research Goal and Objectives
- Overview of Genetic Algorithms (GA)
  - Fuzzy-Based Age Extension of Genetic Algorithms (FAexGA)
- GA-Based Test Case Generation
- Initial Case Studies
- Summary and Discussion

National Institute for Systems Test and Productivity

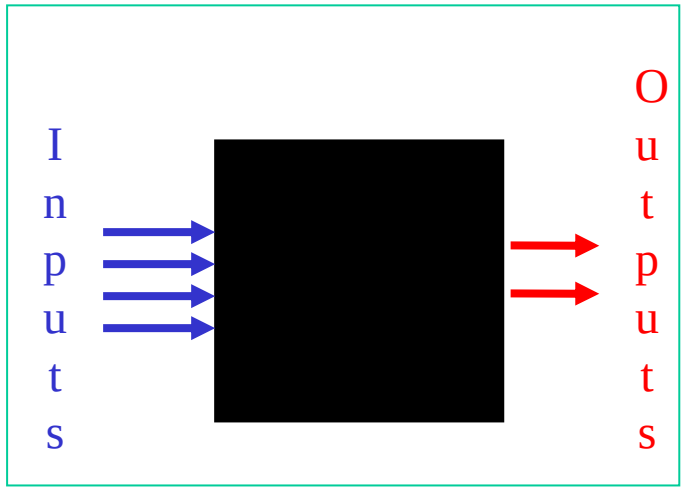
Funding Source: SPAWAR - US Space and Naval Warfare Systems Command (Grant No. N00039-01-1-2248). WWW: http://nistp.csee.usf.edu/

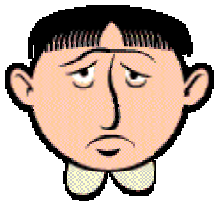
The Institute's mission is to engage in innovative research and to provide direct support to government agencies (and their development agents) which will enable them to substantially reduce cost and improve the development schedule and reliability of large-scale systems

- University of South Florida, Tampa, FL, USA
  - College of Engineering: PI - Prof. Abraham Kandel (Executive Director)
  - College of Business Administration: CO-PI - Prof. Alan R. Hevner
- Subcontractors
  - Arizona State University: PI - Dr. Wei-Tek Tsai
  - Drexel University: PI - Dr. Rosina Weber
  - QTSI: PI - Chick Garcia
  - Ben Gurion University: PI - Dr. Mark Last
  - Echelon: PIs - Dr. Jay Bayne, Dr. Norm Brown

Functional (Black-Box) Testing

- Based on software specification only
- Compares the observed outputs to expected outputs
- Needs an “oracle” for a tested system
- Exhaustive (combinatorial) testing is computationally infeasible for large-scale software systems
- BB Test generation methods
  - Random Testing (within / without the operational profile)
  - Special value testing
  - Boundary value testing
  - Equivalence classes
  - Decision tables
  - Input-Output Analysis





All BB testing methods cover only a tiny portion of possible software inputs!

National Institute for Systems Test and Productivity NISTP 2001

Effective Black-Box Test Cases

- Test case
  - A particular choice of input data to be used in testing a program
- An *effective* set of test cases
  - A test set that has a high probability of detecting faults presenting in a computer program
- Testers' Objectives
  - Generate good test cases
    - A good test case is one that has a high probability of detecting an as-yet undiscovered error
    - Several test cases causing the same bug may show a pattern that might lead to the real cause of the bug
  - Prioritize test cases
    - The dominant criterion: *rate of fault detection* – how quickly a test case detect faults during the testing process

National Institute for Systems Test and Productivity NISTP 2001

Research Goal and Objectives

- Goal
  - Develop novel methodology for generation of **effective black-box test cases** with genetic algorithms
- Basic Idea
  - Eliminate "bad" test cases that are unlikely to expose any error, while increasing the number of "good" test cases that have a high probability of producing an erroneous output
- Objectives
  - Define a general framework for evolving an effective set of test cases with a genetic algorithm
  - Enhance the proposed GA-based methodology by the novel Fuzzy-Based Age Extension of Genetic Algorithms (FAexGA)
  - Evaluate the effectiveness and the efficiency of the proposed approach using synthetic and real computer programs

National Institute for Systems Test and Productivity NISTP 2001

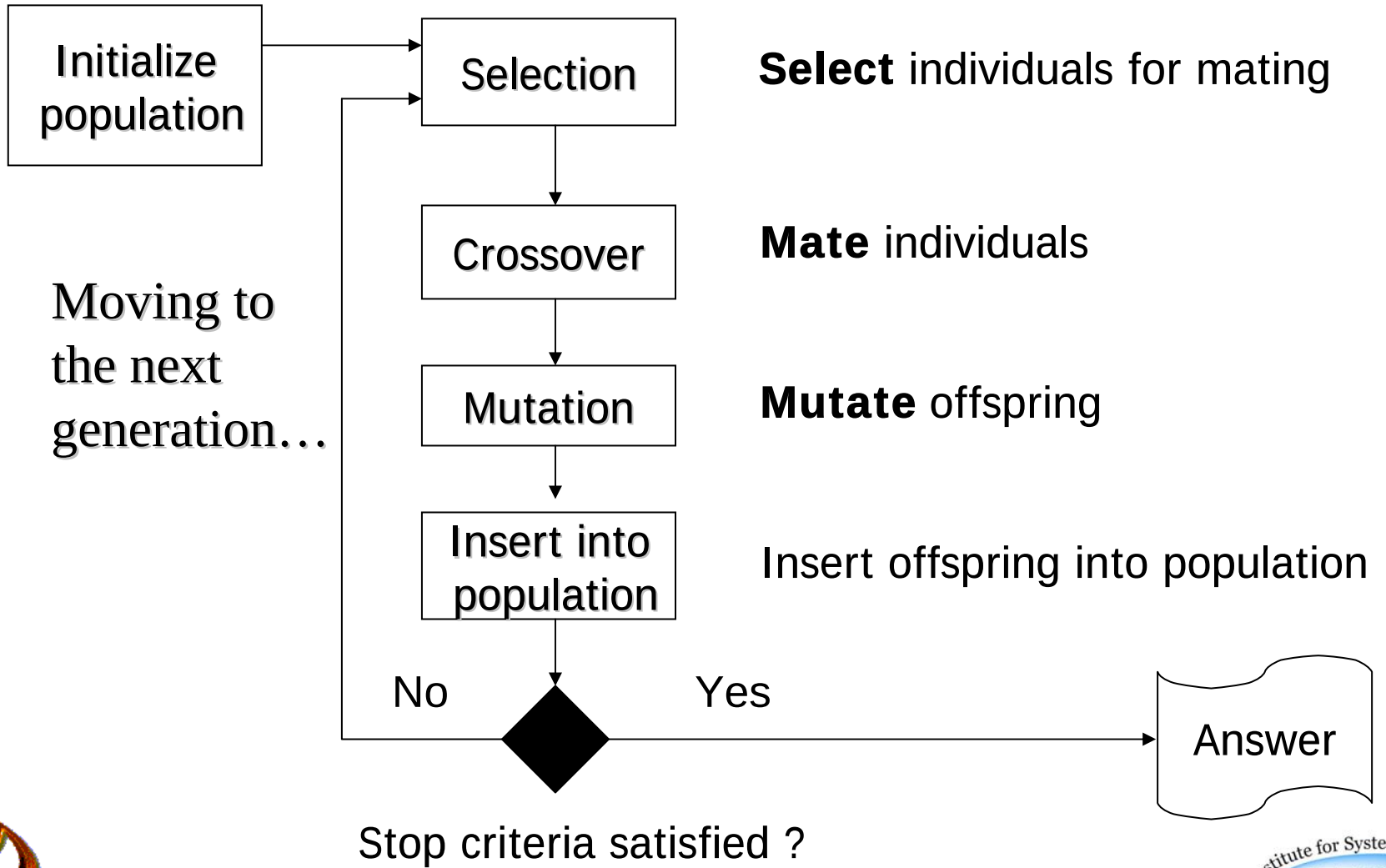
Overview of Genetic Algorithms (GA)

> *"Genetic algorithm is a computerized simulation of Darwin's theories"*
>
> Holland J. H.

- **Search and optimization method** developed by mimicking the evolutionary principles and chromosomal processing in natural genetics
- Application Areas
  - Optimization: numerical, combinatorial (job-shop scheduling).
  - Machine learning tasks: classification, prediction (weights for neural networks).
  - Economics: bidding strategies.

National Institute for Systems Test and Productivity NISTP 2001

GA Working Principle



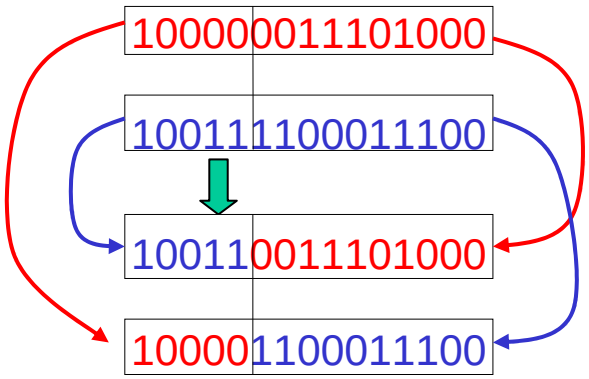
General GA Settings

- Solution encoding (binary, real, tree…).
- Initialization
  - How to create an initial population of potential solutions?
- Genetic operators
  - Selection method (ranking, roulette wheel…).
  - Crossover-operator (one-point, uniform…).
  - Mutation-operator (inversion, flip…).
- Evaluation function
  - Rate the candidate solutions quality
- Parameter values
  - N - population size
  - P_c - crossover probability
  - P_m - mutation probability
  - N_{gen} – number of generations

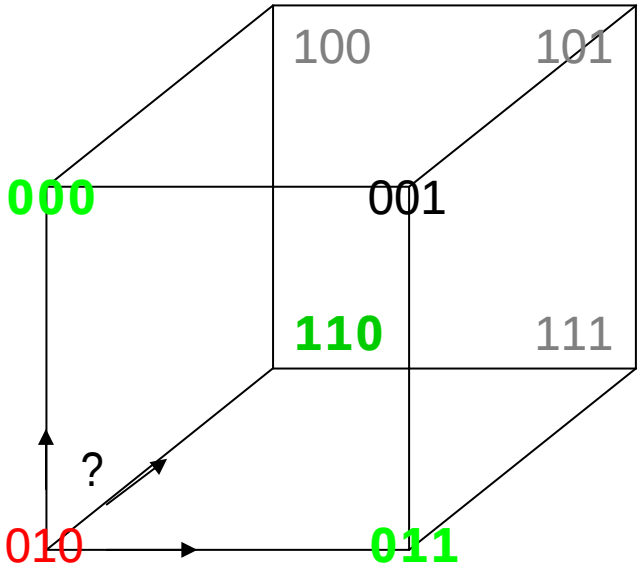
National Institute for Systems Test and Productivity NISTP 2001

Crossover and Mutation

Single-Point Crossover



Bit-flip Mutation

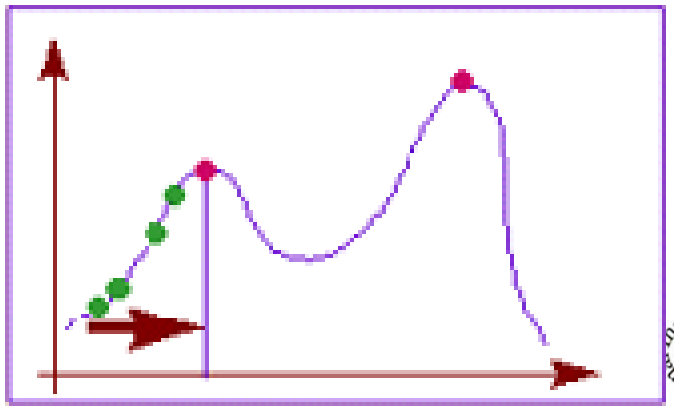


Premature Convergence of GAs

- "Poor parameter setting might make exploitation /exploration relationship (EER) disproportionate, produce lack of diversity and lead to *Premature Convergence*"

  F. Herrera

- Parameters: population size, crossover probability, etc.
- Exploration – overall search in the solution space
- Exploitation – localized search in the promising regions discovered so far in that space



Fuzzy-Based Age Extension of Genetic Algorithms (FAexGA)

(Last and Eyal, 2005)

- Goal
  - Find an optimal policy for controlling the Exploration/Exploitation relationship in GA based on the age attribute.
    - Age = number of generations

Example:

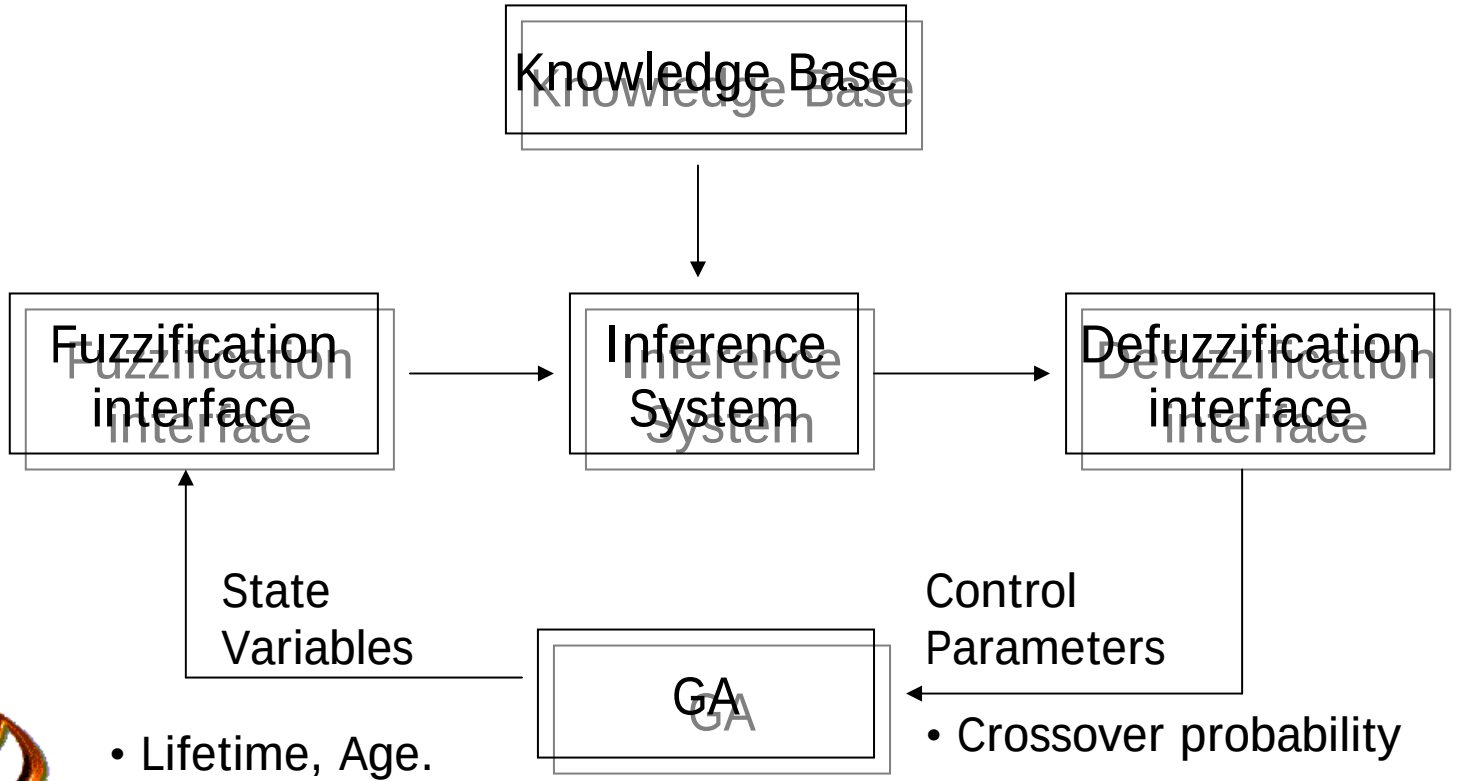
If A is Old and B is Old then crossover probability is Low

Where A and B are two chromosomes chosen for crossover

National Institute for Systems Test and Productivity NISTP 2001

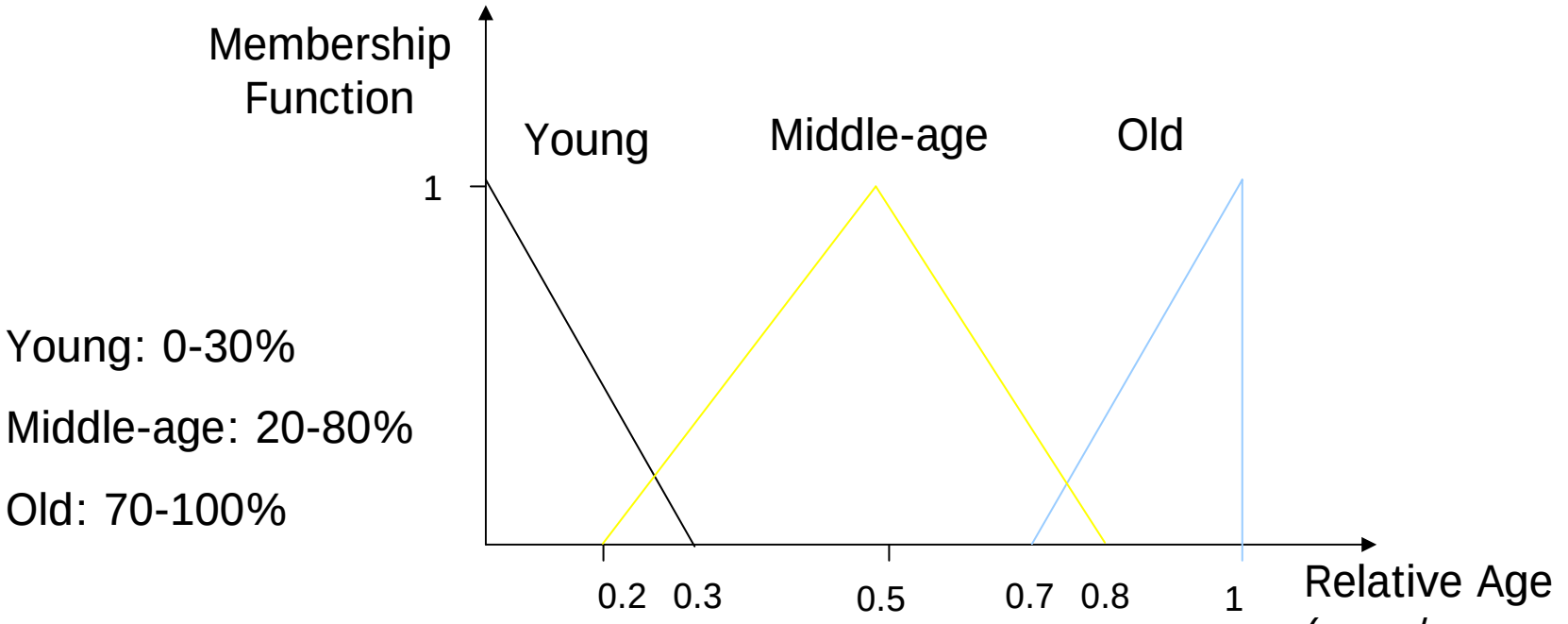
FAexGA Method

- Adapt the crossover probability (*Pc*) with Fuzzy Logic Controller (FLC):



Fuzzification of *Age*

- Age ∈ [Young, Middle-age, Old]



Young: 0-30%

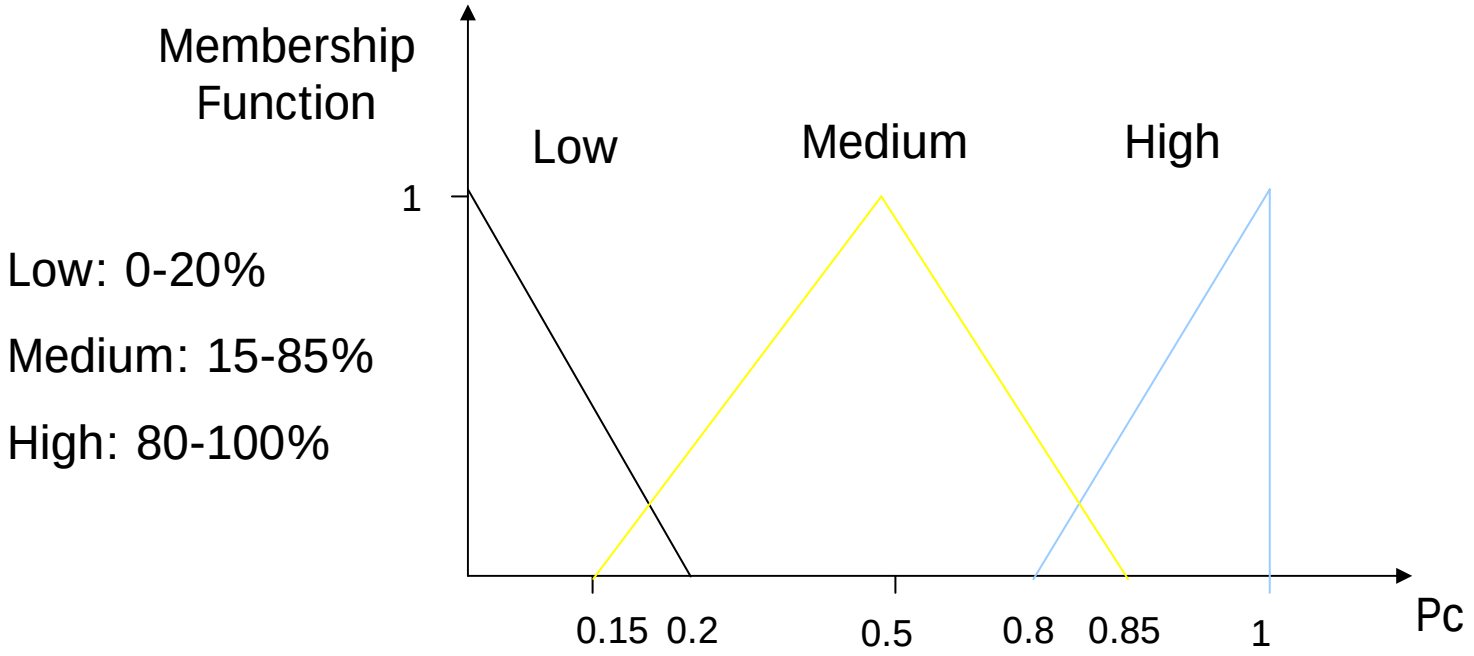
Middle-age: 20-80%

Old: 70-100%

Lifetime = number of generations a chromosome stays alive

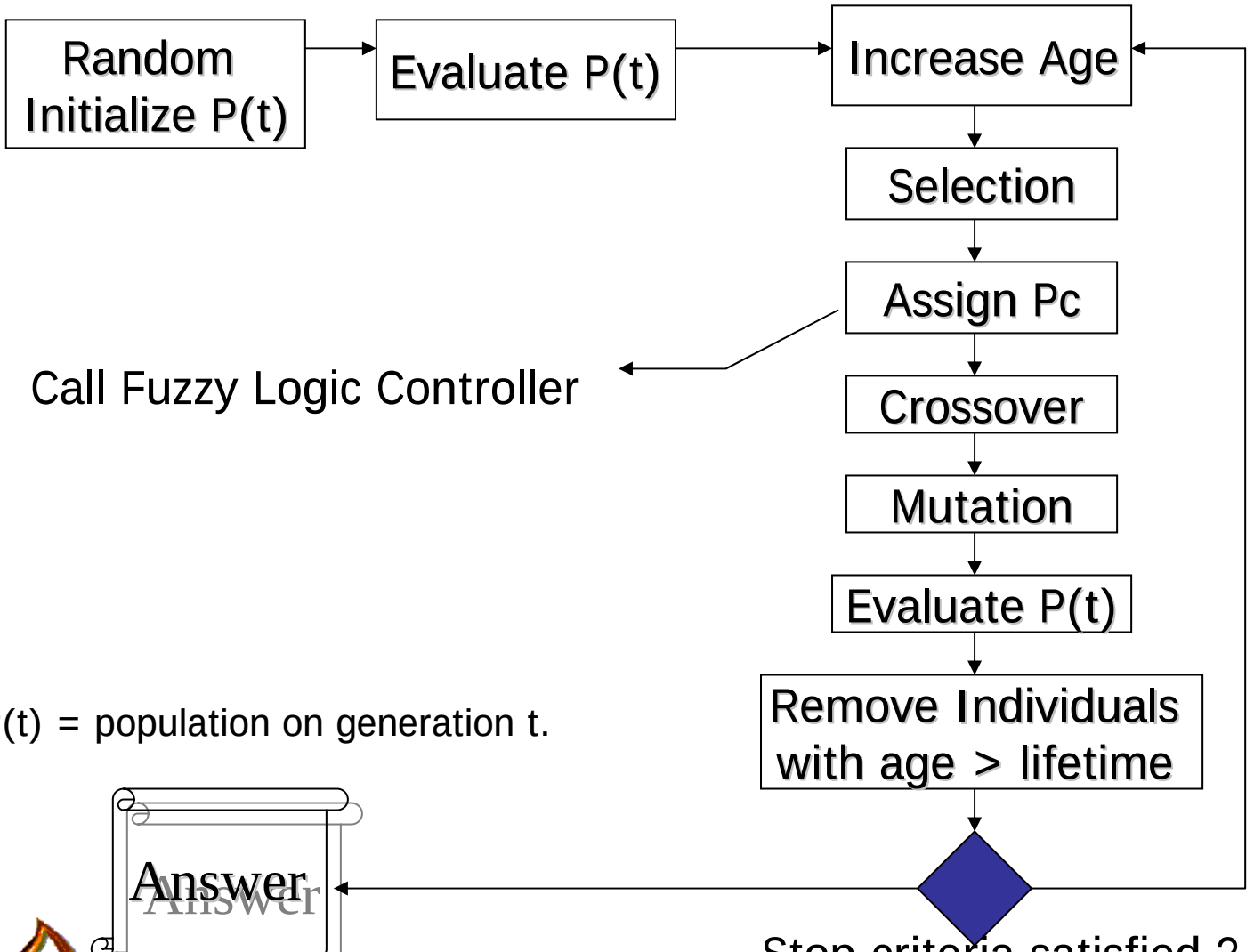
Fuzzification of *Crossover Probability* P_C

- Pc ∈ [Low, Medium, High]



- Defuzzification method: *Center of Gravity*

Fuzzy-Based Age Extension of GA



Fuzzy Rule Base

- Age ∈ [Young, Middle-age, Old]
- Crossover Probability *Pc* ∈ [Low, Medium, High]

Parent I / Parent II	**"Young"**	**"middle-age"**	**"Old"**
"Young"	Low	Medium	Low
"middle-age"	Medium	High	Medium
"Old"	Low	Medium	Low

- Inference method: *MAX-MIN*

National Institute for Systems Test and Productivity NISTP 2001

GA-Based Test Case Generation

- Representation
  - Each test case can be represented as a vector of binary or continuous values related to the inputs of the tested software
- Initialization
  - An initial test set can be generated randomly in the space of possible input values.
- Genetic operators.
  - Selection, crossover, and mutation operators can be adapted to representation of test cases for a specific program.
- Evaluation function.
  - Test cases can be evaluated by their *fault-exposing-potential* using buggy (mutated) versions of the original program.

National Institute for Systems Test and Productivity NISTP 2001

Design of Experiments

- Tested Algorithms:
  - Simple GA [Goldberg].
  - GA with varying population size (GAVaPS).
  - Crisp Age extension of GA.
  - Fuzzy-based Age extension of GA (three configurations).

Parameter	**FAexGA#1**	**FAexGA#2**	**FAexGA#3**
FLC – "Young" upper limit	30%	25%	20%
FLC – "Old" lower limit	70%	75%	80%

Evaluation Measures

Performance measure	Formula
Solution Quality	$\frac{\#\,runs\ yielding\ the\ optimal\ solution}{total\ number\ of\ runs}$
Unuse factor	$u = 1 - \frac{g}{g_{\max}}$ g = # of generations for the first solution
Genotypical diversity	$GD = \frac{\bar{d} - d_{\min}}{d_{\max} - d_{\min}}$ d=distance from the best solution
Phenotypical diversity	$PD = \frac{f_{best}}{\bar{f}}$ f = fitness

Case Study 1: Boolean Expression

- **Correct version** - Boolean expression composed of 100 attributes and three logical operators: AND, OR, and NOT.
  - This expression is generated randomly using an external application
  - Number of combinatorial test cases: 2^{100}
- **Erroneous version** - same Boolean expression as the correct one, except for a single error injected in the correct expression by randomly selecting an OR operator and flipping it to the AND one (or vice versa).
- GA parameters:
  - Number of generations = 200.
  - Population size = 100.
  - Chromosome length = 100.
  - Crossover probability = 0.9 (for simple GA and GAVaPS).
  - Mutation probability = 0.01
  - Discrete evaluation function (Error = Yes / No)

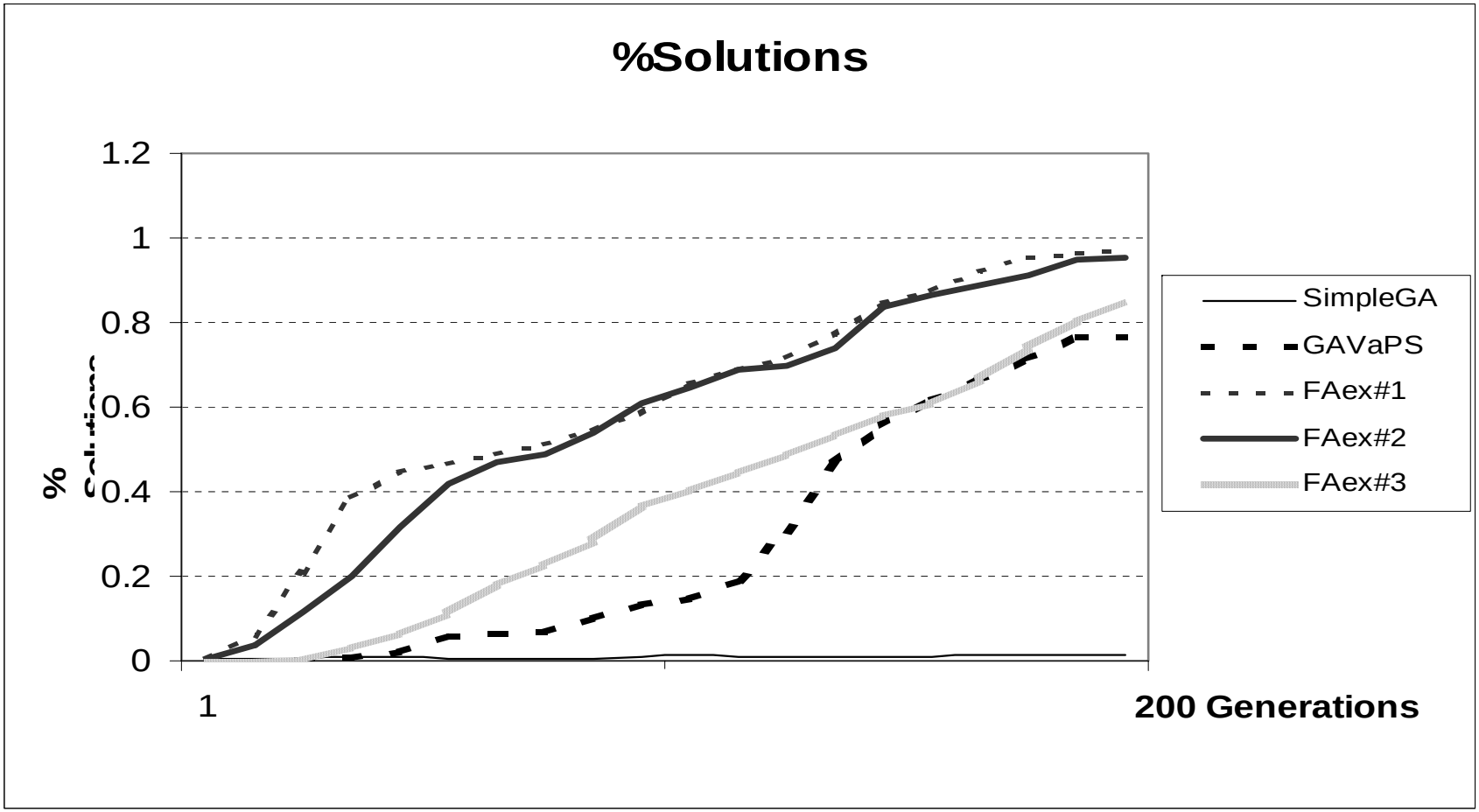
Boolean Expression Results

Measure\Algorithm	SimpleGA	GAVaPS	FAexGA#1	FAexGA#2	FAexGA#3
% Runs with solution	70%	60%	95%	70%	75%
% Solutions in the final population	1.27%	81.52%	99.93%	98.86%	87.18%
Unuse Factor	65.57%	62.27%	70.66%	68.71%	46.27%

- FAexGA#1 reached a solution in most of the runs with significant differences compared to the others algorithms.
- FAexGA#1 averaged the highest number of solutions (this should assist locating the error).
- FAexGA#1 averaged the highest Unuse factor

National Institute for Systems Test and Productivity NISTP 2001

Boolean Expression Results (cont.)



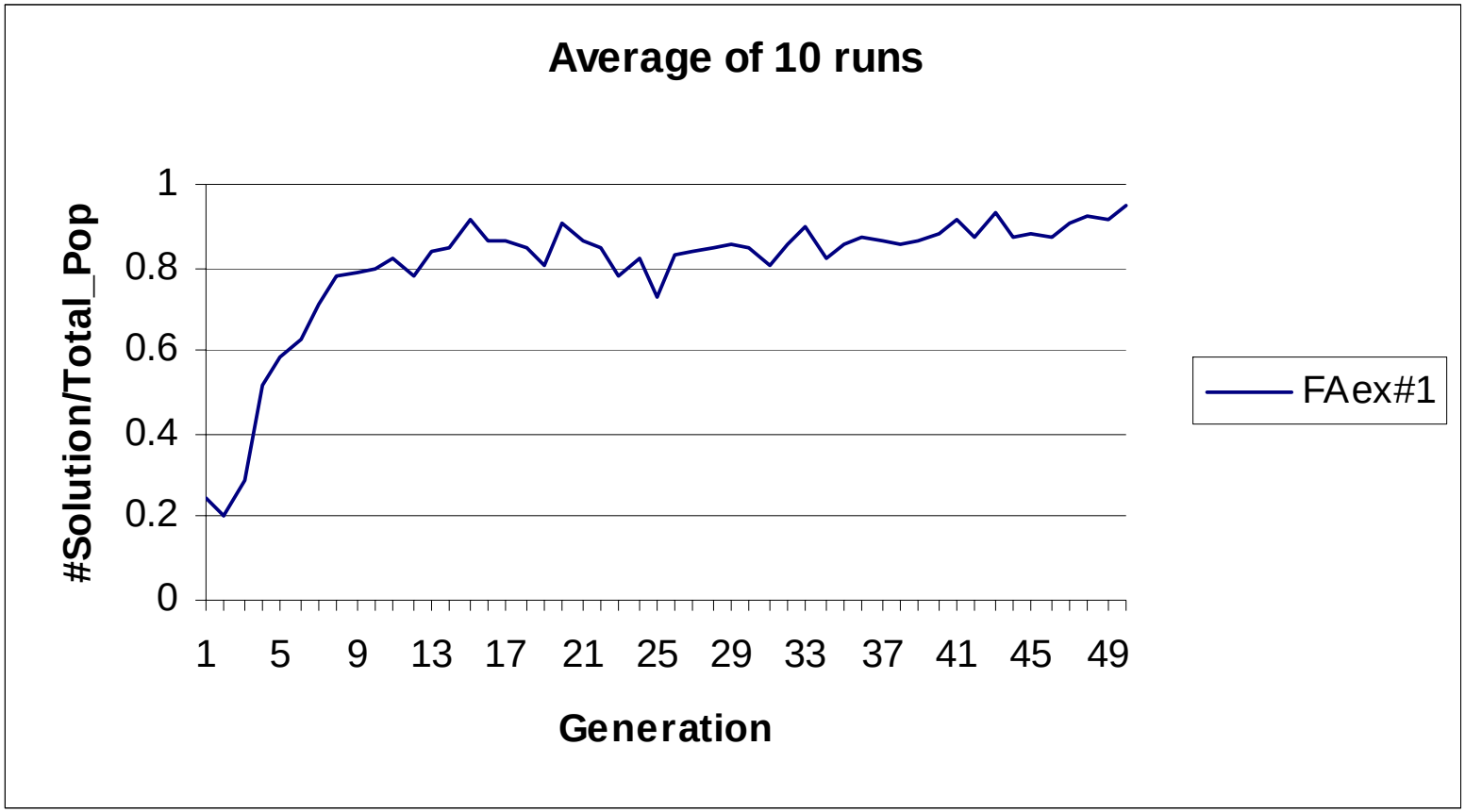
Case Study 2: Credit Application

(Source: Last and Kandel, 2003)

- Number of inputs: 8
  - Five nominal and three continuous
- Number of outputs: 2
  - Decision (accept / reject)
  - Credit limit (continuous)
- Physical lines of code: 63
- Number of program flow paths: 52
- Number of combinatorial tests: 11,312,000

National Institute for Systems Test and Productivity NISTP 2001

Credit Application Results



Summary

- We have introduced a new, GA-based approach to generation of effective black-box test cases
- The Fuzzy-Based Age Extension of Genetic Algorithm (FAexGA) appears to be much more efficient for this problem than the two other evaluated algorithms (SimpleGA and GAVaPS):
  - FAexGA has a much higher probability to find an error in the tested software
  - The error would be found much faster, which should result in saving a lot of resources for the testing team.
  - In addition, the number of distinct solutions produced by FAexGA is significantly higher, which may be useful for investigation and identification of the error itself by the software programmers
- Future research
  - Experimentation with larger software programs
  - Development of more sophisticated, possibly continuous evaluation functions

National Institute for Systems Test and Productivity NISTP 2001

Acknowledgments

- This work was partially supported by the National Institute for Systems Test and Productivity at University of South Florida under the USA Space and Naval Warfare Systems Command Grant No. N00039-01-1-2248 and by the Fulbright Foundation that has awarded Prof. Kandel the Fulbright Senior Specialists Grant at Ben-Gurion University of the Negev in November-December 2005.

Related Publications

- Books
  - M. Last, A. Kandel, and H. Bunke (Editors), "Artificial Intelligence Methods in Software Testing". World Scientific, Series in Machine Perception and Artificial Intelligence, Vol. 56, 2004.
- Papers
  - M. Last and A. Kandel, "Automated Test Reduction Using an Info-Fuzzy Network", T.M. Khoshgoftaar (Ed.), Software Engineering with Computational Intelligence, Kluwer Academic Publishers, pp. 235 – 258, 2003.
  - M. Last, M. Friedman, and A. Kandel, "The Data Mining Approach to Automated Software Testing", Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD-2003), pp. 388 - 396, Washington, DC, USA August 24 - 27, 2003
  - M. Last, M. Friedman, and A. Kandel, "Using Data Mining for Automated Software Testing", International Journal of Software Engineering and Knowledge Engineering (IJSEKE), Special Issue on Data Mining for Software Engineering and Knowledge Engineering, Vol. 14, No. 4, pp. 369-393, August 2004.
  - M. Last and S. Eyal, A Fuzzy-Based Lifetime Extension of Genetic Algorithms, Fuzzy Sets and Systems, Vol. 149, Issue 1, January 2005, 131-147.

National Institute for Systems Test and Productivity NISTP 2001