Hybrid Soft Computing: Where Are We Going?

Piero P. Bonissone
GE

Hybrid SC and EA - Outline

• Soft Computing Overview
  - SC Components: PR, FL, NN, EA
• Modeling with FL and EA
• Hybrid SC Systems
  - FLC Parameter Tuning by EA
  - EA Parameter Setting
• Conclusions

Soft Computing

• Soft Computing (SC): the symbiotic use of many emerging problem-solving disciplines.
• According to Prof. Zadeh:
  "...in contrast to traditional hard computing, soft computing exploits the tolerance for imprecision, uncertainty, and partial truth to achieve tractability, robustness, low solution-cost, and better rapport with reality"
• Soft Computing Main Components:
  - Approximate Reasoning:
    » Probabilistic Reasoning, Fuzzy Logic
  - Search & Optimization:
    » Neural Networks, Evolutionary Algorithms

Problem Solving Techniques

HARD COMPUTING
  Precise Models
  Symbolic Logic Reasoning

SOFT COMPUTING
  Approximate Models
  Traditional Numerical Modeling and Search
  Approximate Reasoning

Soft Computing: Hybrid Probabilistic Systems

Approximate Reasoning
Functional Approximation/Randomized Search

Probabilistic Models
Bayesian Belief Nets

Multivalued & Fuzzy Logics
Dempster Shafer

Neural Networks
Evolutionary Algorithms
Soft Computing: Hybrid EA Systems

- Probabilistic Models
- Multivalued & Fuzzy Logics
- Neural Networks
- Evolution Strategies
- Evolutionary Programs
- Genetic Algorithms

HYBRID EA SYSTEMS

EA parameters (N, Pcr, Pmu) controlled by FLC
EA-based search inter-twined with hill-climbing
EA parameters (Pop size, select.) controlled by EA

Hybrid SC and EA – Outline (2)

- Soft Computing Overview
  - SC Components: PR, FL, NN, EA
- Modeling with FL and EA
  - Hybrid SC Systems
    - FLC Parameter Tuning by EA
    - EA Parameter Setting
  - Conclusions

Fuzzy Logic Genealogy

- Origins: MVL for treatment of imprecision and vagueness
  - 1930s: Post, Kleene, and Lukasiewicz attempted to represent undetermined, unknown, and other possible intermediate truth-values.
  - 1937: Max Black suggested the use of a consistency profile to represent vague (ambiguous) concepts
  - 1965: Zadeh proposed a complete theory of fuzzy sets (and its isomorphic fuzzy logic), to represent and manipulate ill-defined concepts

Fuzzy Logic: Linguistic Variables

- Fuzzy logic give us a language (with syntax and local semantics), in which we can translate our qualitative domain knowledge.
- Linguistic variables to model dynamic systems
- These variables take linguistic values that are characterized by:
  - a label - a sentence generated from the syntax
  - a meaning - a membership function determined by a local semantic procedure

Fuzzy Logic: Reasoning Methods

- The meaning of a linguistic variable may be interpreted as a elastic constraint on its value.
- These constraints are propagated by fuzzy inference operations, based on the generalized modus-ponens.
- A FL Controller (FLC) applies this reasoning system to a Knowledge Base (KB) containing the problem domain heuristics.
- The inference is the result of interpolating among the outputs of all relevant rules.
- The outcome is a membership distribution on the output space, which is defuzzified to produce a crisp output.
**Fuzzy Logic Control: Inference Method**

![Diagram of FLC Inference Method]

**FLC Inference Method (cont.)**

- A FLC (KB + Reasoning Mechanism) defines a deterministic response surface in the cross product of state and output spaces, which approximates the original relationship.

- The FLC leverages the interpolation properties of this reasoning mechanism, to exhibit robustness with respect to parameter variations, disturbances, etc.

---

**Example (MISO): Max-min Composition with Centroid Defuzzification**

- If X is SMALL and Y is SMALL then Z is NEG. LARGE
- If X is SMALL and Y is LARGE then Z is NEG. SMALL
- If X is LARGE and Y is SMALL then Z is POS. SMALL
- If X is LARGE and Y is LARGE then Z is POS. LARGE

---

**Evolutionary Algorithms (EA)**

EA are part of the Derivative-Free Optimization and Search Methods:
- Evolutionary Algorithms
- Simulated annealing (SA)
- Random search
- Downhill simplex search
- Tabu search

EA consists of:
- Evolution Strategies (ES)
- Evolutionary Programming (EP)
- Genetic Algorithms (GA)
- Genetic Programming (GP)

---

**Evolutionary Algorithms Characteristics**

- Most Evolutionary Algorithms can be described by
  \[ x[t+1] = s(v(x[t])) \]

  - \( x[t] \): the population at time \( t \) under representation \( x \)
  - \( v \): is the variation operator(s)
  - \( s \): is the selection operator

---

**Evolutionary Algorithms Characteristics**

- EA exhibit an adaptive behavior that allows them to handle non-linear, high dimensional problems without requiring differentiability or explicit knowledge of the problem structure.

- EA are very robust to time-varying behavior, even though they may exhibit low speed of convergence.
Modeling

- **Model** = 
  \[ \text{Structure} + \text{Parameters} + \text{Search Method} \]

- Classical control theory:
  - Structure: order of the differential equations
  - Parameters: coefficients of differential equation.
  - Search method: LMSE, Pole-placement, etc.

Modeling Using FLC (Mamdani type)

- A Mamdani-type FLC approximates a relationship between a state \( X \) and an output \( Y \) by using a KB and a reasoning mechanism (generalized modus-ponens).

- The Knowledge Base (KB) is defined by:
  - Scaling factors (SF): ranges of values of state and output variables
  - Termset (TS): membership functions of values
  - Ruleset (RS): a syntactic mapping of symbols from \( X \) to \( Y \)

Modeling Using FLC (Mamdani type)

- The structure of the model is the ruleset.

- The parameters of the model are the scaling factors and termsets.

- The search method is initialized by knowledge engineering and refined with some other external methods (SOFC, error minimization, etc.)

Modeling Using EA

- Similarly, for EA:
  - The structure of the model is the representation of an individual in the population (e.g., binary string, vector, parse tree, Finite State Machine).
  - The parameters of the model are the Population Size, Probability of Mutation, Prob. of Recombination, Generation Gap, etc.
  - The search method is a global search based on maximization of population fitness function

Hybrid SC and EA – Outline (3a)

- Soft Computing Overview
  - SC Components: PR, FL, NN, EA

- Modeling with FL and EA

  - Hybrid SC Systems
    - FLC Parameter Tuning by EA
    - EA Parameter Setting

  - Conclusions
FL Controllers Tuned by EAs

- **FLC**
  - FLC = KB + Inference Engine (with Defuzz.)
  - KB parameters:
    - Scaling factors (SF)
    - Membership Functions (MF)
    - Rule set (RS)

- **EA**
  - Encoding: binary or real-valued
  - Chromosome: string or table
  - Fitness function: Sum quadratic errors, entropy
  - Operators: one-point crossover, max-min arithmetical crossover, point-radius crossover.

FL Controllers tuned by EAs (cont.)

- **Historical Approaches:**
  - **Karr 91-93:**
    - Chromosome = concatenation of all termsets.
    - Each value in a termset was represented by 3 binary-encoded parameters.
  - **Lee & Takagi 93:**
    - Chromosome = 1 TSK rule (LHS: memb. fnct. RHS pol.)
    - Binary encoding of 3-parameter repr. of each term
  - **Surman et al: 93:**
    - Fitness function with added entropy term describing number of activated rules

SC in Train Handling: An Example

- **Problem Description**
  - Develop an automated train handler to control a massive, distributed system with little sensor information
  - Freight trains consist of several hundred heavy railcars connected by couplers (train length up to two miles)
  - Each coupler typically has a dead zone and a hydraulically damped spring
  - Railcars can move relative to each other while in motion, leading to a train that can change its length by 50 – 100 ft.
  - The position of the cars and couplers cannot be electronically sensed

SC in Train Handling: An Example

- **Solution Requirements**
  - An automated system has to satisfy multiple goals:
    - Tracking a velocity reference (defined over distance) to enforce speed limits and respect the train schedule
    - Providing a degree of train-handling uniformity across all crews
    - Operating the train in fuel-efficient regimes
    - Maintaining a smooth ride by avoiding sudden accelerations or brake applications (slack control)

Multi-body regulation problem, subject to proper slack management, without sensors for most of the state

SC in Train Handling: An Example

- **Description of Our Approach**
  - Use a Velocity Profile externally generated (using classical optimization or Evolutionary Algorithms)
  - Use a Fuzzy Logic Control (FLC) to track the velocity reference (Fuzzy PI Control)
  - Use an Evolutionary Algorithms to tune the FLC parameters to minimize velocity tracking error and number of throttle changes
  - Implement control actions with fuzzy rule set to maintain slack control
FLC tuned by EAs: Our Approach

- **Chromosome (real-valued encoding):**
  - Chr. 1 = Scaling factors;
  - Chr. 2 = Termsets;
  - Chr. 3 = Rules (not used)

- **Order of tuning (as in Zheng '92):**
  - Initialize rulebase with standard PI structure and termsets with uniformly distributed terms
  - Apply EAs to find best scaling factors
  - Apply EAs to find best termsets
  - Apply EAs to find best rule set (not used)

- Transition from large to small granularity

### FLC Sensitivity to Parameter Changes

<table>
<thead>
<tr>
<th>Changing a Scaling Factor</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Low</td>
<td>PLC</td>
<td>ZE</td>
</tr>
<tr>
<td>Low</td>
<td>PLC</td>
<td>ZE</td>
</tr>
<tr>
<td>Medium</td>
<td>PLC</td>
<td>ZE</td>
</tr>
<tr>
<td>High</td>
<td>PLC</td>
<td>ZE</td>
</tr>
<tr>
<td>Very High</td>
<td>PLC</td>
<td>ZE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Changing a Term in X1</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>ZE</td>
<td>NL</td>
</tr>
<tr>
<td>Medium</td>
<td>ZE</td>
<td>NL</td>
</tr>
<tr>
<td>Low</td>
<td>ZE</td>
<td>NL</td>
</tr>
<tr>
<td>Very Low</td>
<td>ZE</td>
<td>NL</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Changing a Rule</th>
<th>X1</th>
<th>X2</th>
</tr>
</thead>
<tbody>
<tr>
<td>High</td>
<td>ZE</td>
<td>NL</td>
</tr>
<tr>
<td>Medium</td>
<td>ZE</td>
<td>NL</td>
</tr>
<tr>
<td>Low</td>
<td>ZE</td>
<td>NL</td>
</tr>
<tr>
<td>Very Low</td>
<td>ZE</td>
<td>NL</td>
</tr>
</tbody>
</table>

### Architecture: Modules, Fitness Funct.

- **Architecture**
  - EA: pop.size=50; P(cross)=.6; P(mut)=.001
  - Three Types of fitness functions
  - Train Simulator: NSTD (STD+TEM)
  - Fuzzy PI (Ke, Kedot, K∆u)

- **Fitness functions (f₁, f₂, f₃):**

\[
\begin{align*}
  f₁ &= \min\left(\sum_i (\text{notch}_i - \text{notch}_{i+1}) + \text{dynbrake}_i + \text{dynbrake}_{i+1}\right) \\
  f₂ &= \min\left(\sum_i \left| v_i - v_{i+1} \right| \right) \\
  f₃ &= \min\left(\left(\sum_i (\text{notch}_i - \text{notch}_{i+1}) \right) K₁ + \left(\sum_i \left| v_i - v_{i+1} \right| \right) K₂\right) \\
  f₄ &= \min\left(\left| \text{notch}_1 - \text{notch}_{12} \right| \right)
\end{align*}
\]

### FLC tuned by GAs

- **GA (GENESIS)**
- **Fitness Function**
- **Train Simulator**
- **FLC (PI)**

### Experiment Design

- **12 test (4 for each fitness function)**
  - Initial SF with initial MF;
  - EA tuned SF with initial MF;
  - Initial SF with EA tuned MF;
  - EA tuned SF with EA tuned MF

- **Train Simulation:**
  - 14 miles long flat track
  - 1 uniformly heavy train with 100 cars and 4 locomotives
  - Analytically computed velocity profile

- **Constraints to maintain 0.5 terms overlap, for best interpolation**
Experiments Results

• Experiment Results with $f_1$

<table>
<thead>
<tr>
<th>Description</th>
<th>Time</th>
<th>Journey</th>
<th>Fuel</th>
<th>Fitness</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial SF; Initial MF</td>
<td>26.5</td>
<td>14.26</td>
<td>878</td>
<td>73.2</td>
<td></td>
</tr>
<tr>
<td>EA tuned SF; Initial MF</td>
<td>27.8</td>
<td>14.21</td>
<td>870</td>
<td>15.15</td>
<td>4</td>
</tr>
<tr>
<td>Initial SF; EA tuned MF</td>
<td>26.00</td>
<td>14.18</td>
<td>870</td>
<td>70.93</td>
<td>2</td>
</tr>
<tr>
<td>EA tuned SF; EA tuned MF</td>
<td>28.3</td>
<td>14.12</td>
<td>872</td>
<td>14.64</td>
<td>10</td>
</tr>
</tbody>
</table>

• Experiment Results with $f_3$

<table>
<thead>
<tr>
<th>Description</th>
<th>Time</th>
<th>Journey</th>
<th>Fuel</th>
<th>Fitness</th>
<th>Gen.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial SF; Initial MF</td>
<td>26.5</td>
<td>14.26</td>
<td>878</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>EA tuned SF; Initial MF</td>
<td>27.2</td>
<td>14.35</td>
<td>871</td>
<td>0.817</td>
<td>4</td>
</tr>
<tr>
<td>Initial SF; EA tuned MF</td>
<td>26.26</td>
<td>14.18</td>
<td>871</td>
<td>0.942</td>
<td>20</td>
</tr>
<tr>
<td>EA tuned SF; EA tuned MF</td>
<td>27.3</td>
<td>14.35</td>
<td>872</td>
<td>0.817</td>
<td>10</td>
</tr>
</tbody>
</table>

Tuning of FLC with EA: Remarks

• Verified tuning order proposed by Zheng (92)
  » SF tuning: major impact
  » MF tuning: minor impact
  » RS tuning: almost no impact

• For both $f_1$ and $f_3$, fuel minimization is implicitly derived from throttle jockeying minimization

• Complex fitness function (requiring simulation run - 23 sec for each chromosome evaluation) limited trials number - with no apparent impact

• Successfully tested on simulated 43 mile long track with altitude excursions
  » (Selkirk, NY→Framingham, MA)
Hybrid SC and EA – Outline (3b)

- Soft Computing Overview
  - SC Components: PR, FL, NN, EA
- Modeling with FL and EA
- Hybrid SC Systems
  - FLC Parameter Tuning by EA
  - EA Parameter Setting
- Conclusions

EA Parameter Setting

- EA Model:
  - Structure, Parameters
- EA Parameter Setting:
  - EA Parameter Tuning
  - EA Parameter Control
- An Application to Agile Manufacturing
  - Object-level Representation and Complexity
- Solution
  - FLC KB
  - Statistical Experiments
  - Analysis and Summary of 1200 Experiments
- Remarks

EA Model

Object-level Problem

Object-level GA

Structure & Parameters

EA Structure

- GA Structural Design Selections:
  - GA Type:
    » {Simple, Steady-State, Niche,...}
  - Chromosome Encoding:
    » {Binary, Integer, Real,...}
  - Constraints Representation:
    » {Penalty function, data structure, filters, ...}
  - Fitness Function:
    » {Scalar function, Weighted aggregation of multiple functions, Vector-valued function, ...}

EA Parameters

- Adjustable parameters for a GA
  - N = Population size
  - Large pop. prevent premature convergence
  - P_c = Crossover rate:
    P_c * N = # crossovers per generation
  - P_m = Mutation rate:
    P_m * N * L = # mutations per generation
  - G = Generation Gap
    Percentage of population to be replaced
  - W = Scaling Window Size = [1, 7]
  - S = Selection Strategy = (Elitist, Non-Elitist)

EA Parameter Setting - Outline

- EA Model:
  - Structure, Parameters
- EA Parameter Setting:
  - EA Parameter Tuning
  - EA Parameter Control
- An Application to Agile Manufacturing
  - Object-level Representation and Complexity
- Solution
  - FLC KB
  - Statistical Experiments
  - Analysis and Summary of 1200 Experiments
- Remarks
EAs Parameter Setting

Before the run
Parameter Setting
Parameter Tuning
Parameter Control

During the run
Parameter Setting
Parameter Tuning
Parameter Control

Parameter Tuning

Adaptive
Deterministic

Self-Adaptive

EAs Parameter Setting: Parameter Tuning

- **Off-line Tuning**
- Determined before running the GAs on the object-level problem by
  - Studying a subset of five diverse problems (DeJong, 1975)
  - Running a Meta-Genetic Algorithm (Grefenstette, 1986)

Off Line Tuning of GA Parameters
*(DeJong, 1975)*

Object-level GA

Suite of 5 problems:
- Parabola
- Rosenbrock’s saddle
- Step function
- Quartic Noise
- Shekel’s foxholes

Object-level Problem

Object-level GA

Population Size: 50
Crossover Rate: 0.6
Mutation Rate: 0.001
Replacement 100%
Scaling Window n = inf
Selection Strategy Elitist

Object-level GA

On-Line Performance
Population Size: 30
Crossover Rate: 0.96
Mutation Rate: 0.01
Replacement 100%
Scaling Window n = inf
Selection Strategy Elitist

SC Hybrid Systems: EA Tuning EA

Approximate Reasoning Approaches
Probabilistic Models

Search/Optimization Approaches
Neural Networks
Evolutionary Algorithms

Evolution Strategies
Evolutionary Programs

Hybrid EA Systems
EA parameters (Pop size, select.) controlled by EA
GAs Parameter Setting: Deterministic Control

- No feedback information is used.
- A time-varying schedule is used to modify a GA parameter $p$
- $p$ is replaced by $p(t)$
- Correct design of $p(t)$ is very difficult

EAs Parameter Setting: Self-Adaptive Control

- Incorporate parameters into chromosome making them subject to evolution
- Typically used to determine Mutation Step $S$
  - $[g_1, g_2, ... , g_n, S]$
  - Mutation Step for Entire Genome
  or $[g_1, g_2, ... , g_n, S_1, S_2, ... , S_t]$
  - Mutation Steps for Each Genome Value

EAs Parameter Setting: Deterministic Control - Example

Control of Population size
By decreasing Population Size toward the last part of the Evolution we are trying to improve the solution refinement (e.g., more generations with same number of trials)

- Constant Population size: $N = 338$
- Number of trials = $338 \times \text{MaxGen}$

GAs Parameter Setting: Adaptive Control

- Feedback from the search is used to determine the direction and/or magnitude of the change in the parameter value.
- A Fuzzy Logic Controller is used to obtain parameter changes in
  - Population Size
  - Mutation Rate
  as a function of
  - Genotypic Diversity
  - Percentage Completed Trials

SC Hybrid Systems: FLC Tuning EA

Approximate Reasoning Approaches
- Probabilistic Models
- Neural Networks
- Evolutionary Algorithms
- Fuzzy Logic
MV-Algebra

Search/Optimization Approaches
- Evolution Strategies
- Genetic Algorithms
- Genetic Programs

Evolutionary Programs

Genetic Prog.

Evolution Strategies

Fuzzy Logic

Fuzzy Controller

EA parameters controlled by FLC

Fuzzy Logic Controlled GA (FLC-GA)

State Variables describing the evolution stage
- Genotypic Diversity
- Percentage Completed Trials
- Population Size
- Mutation Rate

Fuzzy Logic Controller

KB

Controlled GA parameters

Object-level Problem

Object-level Problem
EA Parameter Setting

• EA Model:
  - Structure, Parameters
• EA Parameter Setting
  - EA Parameter Tuning
  - EA Parameter Control
• An Application to Agile Manufacturing
  - Object-level Representation and Complexity
• Solution
  - FLC KB
  - Statistical Experiments
  - Analysis and Summary of 1200 Experiments
• Remarks

Object-level Problem Representation

Search Space Size

• For EA Statistical Analysis: O(10^7)
• For EA Performance Validation: O(10^{18}) and O(10^{21})

EA Parameter Setting - Outline

• EA Model:
  - Structure, Parameters
• EA Parameter Setting
  - EA Parameter Tuning
  - EA Parameter Control
• An Application to Agile Manufacturing
  - Object-level Representation and Complexity
• Solution
  - FLC KB
  - Statistical Experiments
  - Analysis and Summary of 1200 Experiments
• Remarks

Solution Architecture
### Untuned GA (U-TGA)

- Population Size: 50
- Generations: 250
- Crossover Rate: 0.6
- Mutation Rate: 0.001

### Guidance for Experiments

- Minimize high-level search space size for FLC-EA by
- Identify primary drivers (influences) of EA search
  - DOE determined that the two main drivers were:
    - Population Size ($N$) and Mutation Rate ($P_m$)
- Control primary drivers by few simple heuristic rules
  - Built two FLC controllers with heuristic rule sets and SF
    - Changed on input (state variable) to capture evolution stage
- Determining FLC firing rate
  - Take a control action every 10 generation
- Extensive & statistically significant empirical evidence
  - Use t-test and F-tests to analyze $\mu$ and $\sigma$ improvements

### Fuzzy Logic Controller for EAs: Knowledge Base

- **Inputs**
  - **Genotypic Diversity** ($GD$): A(Very Low), B(Low), C(Medium), D(High), E(Very High)
  - **Percentage Completed Trials** ($PFE$): A(Very Low), B(Low), C(Medium), D(High), E(Very High)

- **Outputs (for both $\Delta N$ and $\Delta P_m$):**
  - A(Neg. High), B(Neg. Medium), C(No Change), D(Pos. Medium), E(Pos. High)

### Fuzzy Controller for $\Delta N$ and $\Delta P_m$:

**Termsets**

- **Inputs:**
  - **GD**: A(Very Low), B(Low), C(Medium), D(High), E(Very High)
  - **PFE**: A(Very Low), B(Low), C(Medium), D(High), E(Very High)
- **Outputs (for both $\Delta N$ and $\Delta P_m$):**
  - A(Neg. High), B(Neg. Medium), C(No Change), D(Pos. Medium), E(Pos. High)
Statistical Experiments: EA Structure

- **Data Set for Experiments**: Seven part classes corresponding to a complexity of O(10^7)
- **EA Structure**:
  - Type: Simple, Steady-State
  - Chromosome Encoding: Integer
  - Fitness Function: Three type of cost functions
  - Selection Method: Proportional Roulette
  - Crossover Operator: Uniform
  - Mutation Operator: Exponentially

Statistical Experiments: Set-Up

- **Set-Up for 1200 experiments**:
  - We defined 4 EA configurations:
    - (a) Untuned Simple EA (U-SEA)
    - (b) FL Controlled Simple EA (FLC-SEA)
    - (c) Untuned Steady State EA (U-SSEA)
    - (d) FL Controlled Steady State EA (FLC-SSEA)

Statistical Experiments: Measures

- For each of the four configurations (a-d) we ran 20 experiments with the same parameters
- Then we considered the following measures:
  \[ \hat{B} = \text{sample average over 20 experiments of} \]
  \[ \text{Best score frequency (number of time cost function J reached its minimal value -}
  \text{known a priori for small size experiment)} \]
  \[ \hat{\mu} = \text{average of population best} \]
  \[ \hat{\sigma} = \text{standard deviation of population best} \]
**Statistical Experiments: Analysis**

- We performed an ANOVA test (both t and F test - with p < 0.05 ) to see if:
  - Cost (U-SEA) >> cost (FLC-SEA)
  - Cost (U-SEA) >> cost (U-SSEA)
  - Cost (U-SSEA) >> cost (FLC-SSEA)

- We verified if the FLC caused the controlled EA to perform worse than its corresponding untuned EA, i.e.:
  - Cost (U-SEA) << cost (FLC-SEA)
  - Cost (U-SSEA) << cost (FLC-SSEA)

<table>
<thead>
<tr>
<th>Function</th>
<th>U-SEA</th>
<th>FLC-SEA</th>
<th>U-SSEA</th>
<th>FLC-SSEA</th>
</tr>
</thead>
<tbody>
<tr>
<td>C*T</td>
<td>-</td>
<td>-</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>C*T²</td>
<td>-</td>
<td>-</td>
<td>2</td>
<td>3</td>
</tr>
<tr>
<td>C*e(T-10)/3</td>
<td>1</td>
<td>-</td>
<td>-</td>
<td>1</td>
</tr>
<tr>
<td>µσ</td>
<td>7%</td>
<td>47%</td>
<td>60%</td>
<td>7%</td>
</tr>
</tbody>
</table>

- Total changes in µ:
- Total changes in σ:

**EA Parameter Setting**

- **EA Model:**
  - Structure, Parameters
- **EA Parameter Setting**
  - EA Parameter Tuning
  - EA Parameter Control

**An Application to Agile Manufacturing**

- Object-level Representation and Complexity

- **Solution**
  - FLC KB
  - Statistical Experiments

- **Remarks**

**Remarks (cont.)**

- **Main Result**
  - By using the FLC with the above State and Control variables, we achieved a good improvement of the population average and an even better improvement of the population variance.
  - No major negative effects on EA performance using FLC

**Remarks**

- **FLC State Representation:** [Evolution Stage]
  - Evolution time needs to be an explicit state variable since we have different control goals during the EA’s stages.
  - Diversity measures the evolutionary stage:
    - Percentage Fitness Evaluations (PFE)
    - Genotypic Diversity (GD)

- **FLC Control Variables:** [EA Adaptable Parameters]
  - \( \Delta N \) = Change in Population Size
  - \( \Delta P_m \) = Change in Mutation Rate

**Hybrid SC and EA – Outline (4)**

- **Soft Computing Overview**
  - SC Components: PR, FL, NN, EA
- **Modeling with FL and EA**
- **Hybrid SC Systems**
  - FLC Parameter Tuning by EA
  - EA Parameter Setting

- **Conclusions**
Synergy in SC: Reasons & Approaches

**Hybrid Soft Computing**
- Leverages **tolerance for imprecision**, uncertainty, and incompleteness - intrinsic to the problems to be solved
- Generates **tractable, low-cost, robust** solutions to such problems by **integrating knowledge and data**

**Tight Hybridization**
- Data-driven Tuning of Knowledge-derived Models
  - Translate domain knowledge into initial structure and parameters
  - Use Global or local data search to tune parameters
- Knowledge-driven Search Control
  - Use Global or local data search to derive models (Structure + Parameters)
  - Translate domain knowledge into an algorithm’s controller to improve/manage solution convergence and quality

Synergy in SC: Reasons & Approaches

**Loose Hybridization (Model Fusion)**
- Does not combine **features** of methodologies - only their results
- Their outputs are compared, contrasted, and aggregated, to increase reliability

**Hybrid Search Methods**
- Intertwining **local** search within **global** search
- Embedding knowledge in operators for global search

**Future:**
- Circle of SC’s related technologies will probably widen beyond its current constituents.
- Push for low-cost solutions and intelligent tools will result in deployment of hybrid SC systems that efficiently integrate reasoning and search techniques.

FL Controllers tuned by EAs (cont.)

**Historical Approaches (cont.):**
- **Kinzel et al. 94:**
  - Chromosome = Rule Table
  - Point-radius crossover changing 3x3 rule window (similar to a two-point crossover for string representation)
  - Order of tuning:
    - Initialize rulebase according to heuristics
    - Apply GAs to find best rule table
    - Tune membership function of best rule set
- **Herrera et al. 95:**
  - Chromosome = concatenation of all rules
  - Real-valued encoding, Max-min arithmetical crossover

Evolutionary Algorithms: ES

**Evolutionary Strategies (ES)**
- Originally proposed for the optimization of continuous functions
- \((\mu, \lambda)\)-ES and \((\mu + \lambda)\)-ES
  - A population of \(\mu\) parents generate \(\lambda\) offspring
  - Best \(\mu\) offspring are selected in the next generation
  - \((\mu, \lambda)\)-ES: parents are **excluded** from selection
  - \((\mu + \lambda)\)-ES: parents are **included** in selection
- Started as \((1+1)\)-ES (Reschenberg) and evolved to \((\mu + \lambda)\)-ES (Schwefel)
- Started with Mutation only (with individual mutation operator) and later added a recombination operator
- Focus on behavior of individuals

Evolutionary Algorithms: EP

**Evolutionary Programming (EP)**
- Originally proposed for sequence prediction and optimal gaming strategies
- Currently focused on continuous parameter optimization and training of NNs
- Could be considered a special case of \((\mu + \mu)\)-ES without recombination operator
- Focus on behavior of species (hence no crossover)
- Proposed by Larry Fogel (1963)
Evolutionary Algorithms: GA

- Genetic Algorithms (GA)
  - Perform a randomized search in solution space using a genotypic rather than a phenotypic.
  - Each solution is encoded as a chromosome in a population (a binary, integer, or real-valued string).
  - Each string’s element represents a particular feature of the solution.
  - The string is evaluated by a fitness function to determine the solution’s quality.
  - Better-fit solutions survive and produce offspring.
  - Less-fit solutions are culled from the population.
  - Strings are evolved using mutation & recombination operators.
  - New individuals created by these operators form next generation of solutions.
  - Started by Holland (1962; 1975)

- Genetic Programming (GP)
  - A special case of Genetic Algorithms.
  - Chromosomes have a hierarchical rather than a linear structure.
  - Their sizes are not predefined.
  - Individuals are tree-structured programs.
  - Modified operators are applied to sub-trees or single nodes.
  - Proposed by Koza (1992)

GA Structure (cont.)

- GA Structural Design Selections:
  - Parent Selection Method:
    - {Proportional Roulette, Tournament, Rank, Uniform, ...}
  - Crossover Operator:
    - {Once-cut, Two-cuts, Uniform, BLX, Parent Weighted, ...}
  - Mutation Operator:
    - Mutation Rate: {Exponentially Decreasing, Uniform, ...}
    - Value: {Exponentially Decreasing, Uniform, Normally Distributed, ...}

GA Parameters (cont.)

- Other possible parameters that could be adjusted:
  - T = Number of Trials = \( \Sigma N_i \)
    where N is population size and \( i = 1, \text{Max}_\text{Gen} \)
  - \( \sigma_\mu \) = Mutation step
    \( \sigma \) in Normally distributed Mutation value
  - \( P_s \) = Probability of Selection
    Parametrized slope of probability distribut.
  - \( A_s \) = Arity of Parents
    number of parents in recombination

Fuzzy Controller for Mutation Rate: Rule Set

Fuzzy Controller for Mutation Rate: Rule Set

\[ GD, \text{PFE} \rightarrow \Delta P_m \]

<table>
<thead>
<tr>
<th>Genotypic Diversity</th>
<th>Percentage Fitness Evaluation (PFE)</th>
<th>Change in Mutation Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Very Low</td>
<td>Pos High</td>
<td>Pos High</td>
</tr>
<tr>
<td>Low</td>
<td>Pos High</td>
<td>Pos Medium</td>
</tr>
<tr>
<td>Medium</td>
<td>Pos Medium</td>
<td>Pos Medium</td>
</tr>
<tr>
<td>High</td>
<td>Pos Medium</td>
<td>No Change</td>
</tr>
<tr>
<td>Very High</td>
<td>No Change</td>
<td>No Change</td>
</tr>
</tbody>
</table>

Statistical Experiments: Set-Up (cont.)

- GA Parameters
  - \( N \) = Base Population size: 50
  - \( P_c \) = Crossover rate: 0.600
  - \( P_m \) = Mutation rate: 0.005
  - \( G \) = Generation Gap 100% replacement
    - Simple GA (SGA)
    - Steady State GA (SSGA)
  - \( S \) = Selection Strategy: Elitist
Summary of 1200 Experiments

GAs controlled by FL (cont.)
- **Probability of Selection:**
  - Parametrized slope distribution ranging from:
    - Uniform probability: ignore fitness function and perform random selection of parents - extreme case of exploration, to
    - Proportional selection with rescaling and other intermediate strategies - compromise between exploration and exploitation cases, and
    - Ranking: always select the best N and ignore the rest - extreme case of exploitation
  - Probability as function of fitness and genotypical distance with other solutions - enforcing diversity and favoring exploration
- **Probability of crossover:**
  - Constraints applicability to mostly good solutions
- **Customized-crossover operators** (for real-coded GAs):
  - Selection of crossovers based on T-norms and T-conorms causes offsprings to take more extreme values (exploration)
  - Selection of crossovers based on aggregating operators causes offsprings to take average values (exploitation)

Fuzzy Controller for $\Delta N$ and $\Delta P_m$: Outputs
- **Outputs**
  - $\Delta N = \text{Change in Population Size (Mult. Factor)}$
    - $\Delta N$ range is [0.5, 1.5] == [Neg High, Pos High] so that NC corresponds to 100% of previous Pop Size
    - Population Size is clamped within [25, 150]
  - $\Delta P_m = \text{Change in Mutation Rate (Mult. Factor)}$
    - $\Delta P_m$ range is [0.5, 1.5] == [Neg High, Pos High] so that NC corresponds to 100% of previous Pm
    - Mutation Rate is clamped within [0.005, 0.10]

Next Steps: Controlling Other Parameters
- **Run-time Controlled GAs Parameters:**
  - Population size:
    - Larger size: increase parallel search in solution space
    - Smaller size: focus on current existing regions
  - Probability mutation:
    - Higher prob. of mutation disrupts current solutions - exploration
    - Lower probability of mutation favors current solutions - exploitation
- **Other Possible Run-time Controllable GAs Parameters:**
  - **Customized mutation operators:**
    - Variable amount of changes - smaller for good solutions, larger for bad ones
  - **Fitness function:**
    - Evolving fitness function (variable weights in multi-criteria aggregating function)

Fuzzy Controller for $\Delta N$ and $\Delta P_m$: Control Parameters
- **Frequency of Control Actions**
  - Control Action: mutation rate changed every 10 generations
  - Population size change every generation
- **Mutation Rate**
  - Mutation rates drops exponentially after a control action that increases it
- **Inference Engine Parameters**
  - Left Hand Side (LHS) evaluation: Minimum operator
  - Rule Firing: Minimum operator
  - Rule Output Aggregation: Maximum operator
  - Defuzzification: Center of Gravity (COG)

Fusion of Reasoning Models
- **Develop Collection of Quasi-independent Models**
  - Each Model Generates:
    - Output Value ($v_i$) - Prediction
    - Confidence parameter ($c_i$) derived from training stats - Introspection
  - Intelligent Fusion Rules
    - Consider discrepancies among Output values ($v$)
    - Consider dynamic confidence value ($c$) associated with each output

Example of Fusion for Mortgage Collateral Evaluation

\[ s_F = \{ V_1, C_1 \} \]
Synergy in SC: Reasons & Approaches

- SC Leverages Knowledge and Data to Derive the Model
- Model = Structure + Parameters (& Search)
- Data-driven Tuning of Knowledge-derived Models
  - Translate domain knowledge to initial structure & parameters
  - Use Global or local data search to tune parameters
- Knowledge-driven Search Control
  - Use Global or local data search to derive models (Structure + Parameters)
  - Translate domain knowledge into an algorithm’s controller to improve/manage solution convergence and quality
- Hybrid Search Methods
  - Embedding local search within global search
  - Embedding knowledge in operators for global search
  - Fusion of models to increase accuracy and reliability