

Evolutionary Algorithms

Examples of Application

Prof. Dr. Rudolf Kruse **Pascal Held**

`{kruse,pheld}@iws.cs.uni-magdeburg.de`

Otto-von-Guericke University Magdeburg

Faculty of Computer Science

Institute of Knowledge and Language Engineering

Outline

1. Planning Flight Routes: ROGENA

Problem

Concept of Solution

Solution Approach Using Evolutionary Algorithms

2. Learning Fuzzy Control

Project: Rogena (DLR, 1995)

aircrafts move within the airspace

- on standard routes between airports
- with a fixed minimum distance on the same route
- on different altitudes depending on the flight's direction

Advantages of this solution:

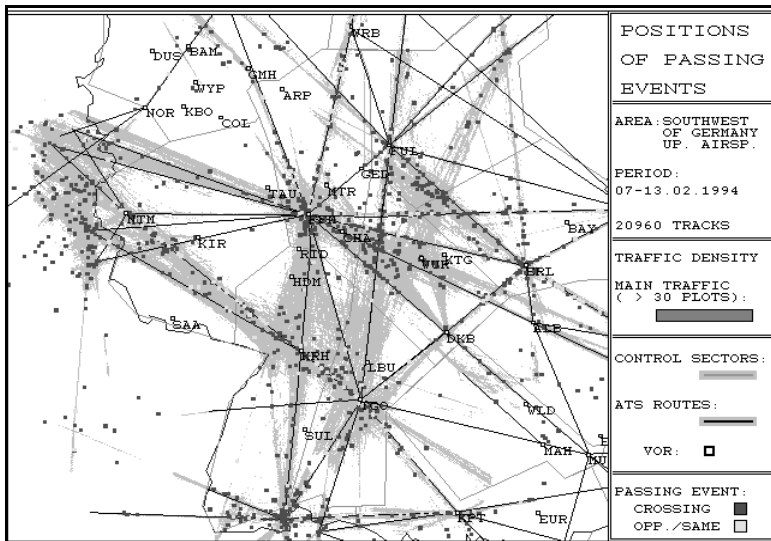
- simple rules/regulations for air traffic,
- easy supervision of flight routes by air traffic controllers,
- safe air traffic (e.g. preventing collisions)

Disadvantages of this solution:

- exploiting only little of the air space,
- aircraft density is very limited near the airports

not an adequate **solution** for today's air traffic

Air Space of the Surroundings of Frankfurt



Concept of Solution: Support for Air Traffic Controllers

airspace between standard routes has to be used for future air traffic increase

problem of this approach:

- more difficulties in controlling freely moving planes
- ⇒ increasing workload for air traffic controllers

task: develop an assistance tool supporting the design of safe and efficient routes between entry and exit points of one controller's airspace

fixed constraints:

- entry time, entry/exit locations within controlled zone,
- different flight characteristics of the controlled aircrafts,
- movements of other aircrafts (avoid collisions),
- restricted areas (because of buildings, military restrictions, stormy weather)

Solution Approach Using Evolutionary Algorithms

- search space for potential flight routes is huge
- it is difficult to find an analytical generation method
search with EA: **ROGENA** (free ROuting with GENetic Algorithms) [Gerdes, 1994, Gerdes, 1995]
- required data and calculation formulas for a description of an aircraft's attributes and behaviour: BADA-database of EUROCONTROL
- consider airspace excerpt of 200×200 nautical miles (NM) and heights of 0 up to 10,000 feet (ft)
- minimum distance between aircrafts: 5 NM, when landing 2.5–6 NM (dependent on the weight of both airplanes)
- EA only starts its job when there are flight route conflicts (violating the safety distance)

Coding of the Candidate Solutions

flight route = sequence of line segments

chromosomes of variable length/number of genes

every gene: a

point to pass;

additionally with
assigned air speed

	x	y	z	v
gene 1	86.2	15.8	1.65	258
gene 2	88.9	24.7	1.31	252
\vdots	\vdots	\vdots	\dots	\vdots
gene k	105.0	98.0	0.00	120

constraints, e.g.

- monotonous decreasing flight route (because of modelling landings),
- maximum acceleration/slowdown between points,
- minimum angle between line segments (radius of curves) etc.

Fitness Function

Consider the following attributes of a flight route:

- safety distance to be maintained by other aircrafts,
- no passing of restricted airspace,
- length of a flight route to runway,
- arrival according to schedule (punctuality),
- keep difference to the optimum descent (calculated according to BADA-database) as small as possible,
- no acute angles between line segments, to keep the difference to the actual flight path as small as possible

these **attributes** contribute to the fitness function (according to weighted factors)

user may change weight factors for single attributes via slider bars

Processing the Evolutionary Algorithm

Aim: Generating a safe and efficient flight route for an aircraft that has just entered the controlled airspace

basis: modified form of an EA (one chromosome is subjected to either crossover or mutation)

population size: 60 individuals

initialization of the **initial population:** by repeatedly altering the standard flight route randomly

selection method:

basically roulette-wheel selection, with chromosomes violating a certain fitness threshold not contributing to the calculation of the selection probabilities (minimum goodness)

threshold is decreasing with t according to average fitness of the population (combination of roulette-wheel selection and flooding algorithm)

Applying Genetical Operators

20 chromosomes are directly taken into the next generation,
among them are the 5 best chromosomes (elitism)

20 chromosomes are processed by **2-point-crossover**

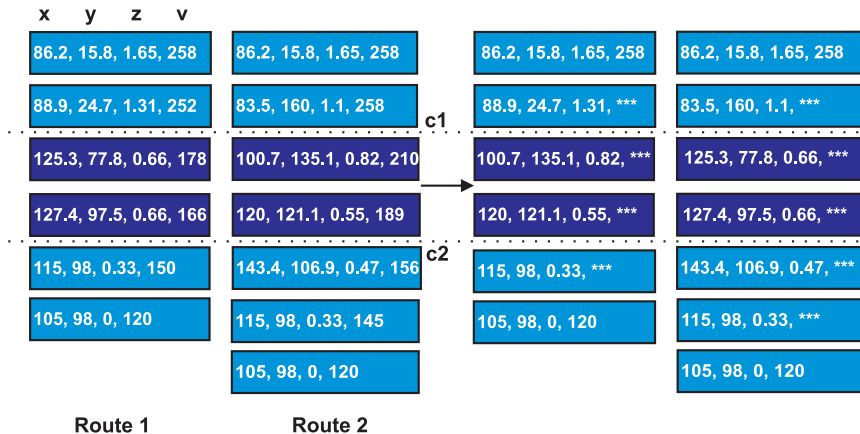
20 chromosomes are processed by a special **mutation operator**:

- either change the coordinates completely by chance (global search)
- or medium random change to a point close by,
- or small random change to a point nearby (hill climbing)
- additionally: change the number of genes (with low probability):
random location of deletion/insertion within the chromosome, new
point is initialized randomly close to its neighbors

repair methods:

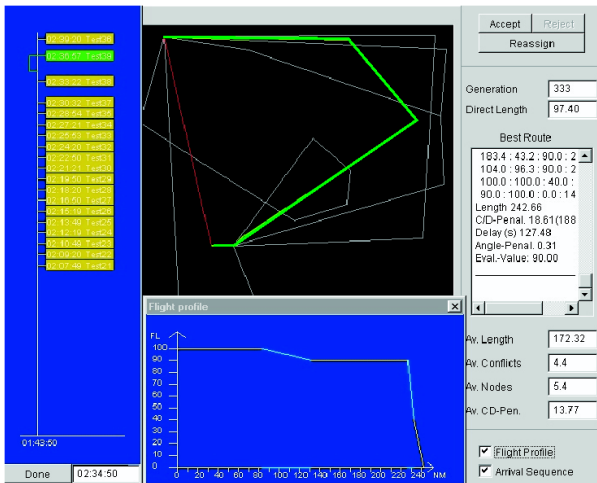
sequence of descending and climbing is uneconomical, as well as
sequence of accelerations and slowdowns

2-Point-Crossover



reason: exchanging subsequences in routes

User Interface



Flight Test Cases List:

- 02:30:30 Test28
- 02:30:47 Test38
- 02:33:22 Test28
- 02:30:32 Test27
- 02:29:44 Test26
- 02:27:21 Test24
- 02:24:43 Test23
- 02:24:20 Test22
- 02:22:50 Test21
- 02:21:21 Test20
- 02:14:50 Test24
- 02:13:20 Test28
- 02:10:50 Test27
- 02:10:30 Test26
- 02:10:30 Test25
- 02:10:19 Test24
- 02:10:09 Test23
- 02:09:20 Test22
- 02:07:09 Test21

Map: Shows a flight path with a highlighted green route and a red line segment.

Flight profile:

FL vs. NM graph showing altitude (0-100) over distance (0-240 NM).

Control Panel:

Accept | Reject
Reassign

Generation: 333
Direct Length: 97.40

Best Route:
183.4 : 43.2 : 90.0 : 2
104.0 : 96.3 : 90.0 : 2
100.0 : 100.0 : 40.0 :
90.0 : 100.0 : 0.0 : 14
Length 242.66
CID+Penal. 18.61(188)
Delay (s) 127.48
Angle-Penal. 0.31
Eval-Value: 90.00

Av. Length: 172.32
Av. Conflicts: 4.4
Av. Nodes: 5.4
Av. CD-Pen.: 13.77

Flight Profile
 Arrival Sequence

Done | 02:34:50

Test Runs and Results

User Interface

- original route in red (direct connection of entry and exit location)
- best current route in green
- alternative routes in gray
- height chart of the route in a separate window
- left: timeline with landing times of all planes

several **simulation runs** with real data (describing entry/exit location and actual flight path)

routes generated by **ROGENA**: clearly shorter than real routes, without conflicts with other aircrafts

effective **support** for air traffic controllers

shortening routes allows for better aircraft throughput

Outline

1. Planning Flight Routes: ROGENA

2. Learning Fuzzy Control

- Fuzzy Logic

- Fuzzy Set Theory

- Linguistical Variables

- Fuzzy Control

- Generating/Optimizing Fuzzy Controls with EA

Brief Introduction to Fuzzy Theory

classical logic: only boolean values *true* and *false*

classical set theory: either *is in* or *not*

⇒ principle of bivalence: often inappropriate

Example: Sorites paradox (greek *sorites*: heap)

true: „One million grains of sand is a heap of sand.“

true: „A heap of sand minus one grain is still a heap.“

true: “999 999 grains of sand is still a heap.“

repeated application of this conclusion:

false: „1 grain of sand is still a heap.“

question: At which number of grains of sand is this conclusion not true?

Brief Introduction to Fuzzy Theory

obviously: no fixed number of grains of sand can be defined for this conclusion to become false for the first time

problem: terms of natural language (e.g. „heap of sand“, „bald“, „warm“, „fast“, „high pressure“, „easy“ etc.) are **vague**

vague terms are *inexact*, but not *useless*

- even for vague terms there are situations/objects for which they are *certainly applicable*, and such for which they are *certainly not applicable*
- in between: **Penumbra** (from the Latin: *partial eclipse*) of situations where it is not clear whether these terms are applicable, or where they are only applicable with restrictions (“small heap of sand”).
- Fuzzy Theory: mathematical model of penumbra

Fuzzy Logic

extension of classical logic to intermediate values inbetween *true* and *false*

boolean value: every value in $[0, 1]$, with $0 \hat{=} \textit{false}$ und $1 \hat{=} \textit{true}$

extension of the logical operators

classical logic		fuzzy logic	
operation	notion	operation	notion
negation	$\neg a$	fuzzy negation	$1 - a$
conjunction	$a \wedge b$	<i>t</i> -norm	$\min(a, b)$
disjunction	$a \vee b$	<i>t</i> -conorm	$\max(a, b)$

basic principles of this extension:

- for extreme values 0 and 1 the operations should work just like the their classical alike (constraints)
- for intermediate values their behaviour should be monotonous
- laws of classical logic should be preserved

Fuzzy Set Theory

classical set theory is based on the term „*is element of*“ (\in)
alternatively: membership in a set is described by *indicator function*:

let X be a set, then

$$I_M : X \rightarrow \{0, 1\}, \quad I_M(x) = \begin{cases} 1, & \text{if } x \in M, \\ 0, & \text{sonst,} \end{cases}$$

is called the **indicator function** of the set M according to the universe X

fuzzy set theory: indicator function is determined by a *membership function*:

$\mu_M : X \rightarrow [0, 1], \mu_M(x) \hat{=}$ Zugehörigkeitsgrad von x zu M ,
membership function of the **fuzzy set** M according to the universe X

fuzzy set: a set defined by its membership function

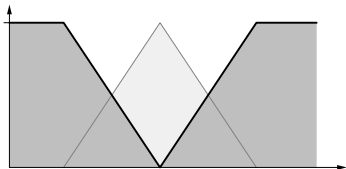
Operations

basic principle of this extension:

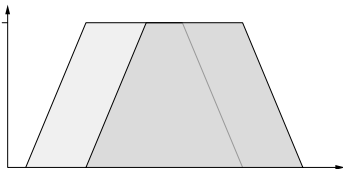
element-wise application of logical operators for
(fuzzy) sets A and B over universe X

complement	classical	$\bar{A} = \{x \in X \mid x \notin A\}$
	fuzzy	$\mu_{\bar{A}}(x) = 1 - \mu_A(x)$
intersection	classical	$A \cap B = \{x \in X \mid x \in A \wedge x \in B\}$
	fuzzy	$\mu_{A \cap B}(x) = \min(\mu_A(x), \mu_B(x))$
union	classical	$A \cup B = \{x \in X \mid x \in A \vee x \in B\}$
	fuzzy	$\mu_{A \cup B}(x) = \max(\mu_A(x), \mu_B(x))$

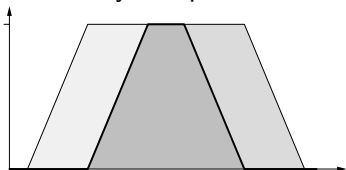
Fuzzy Set Operators: Examples



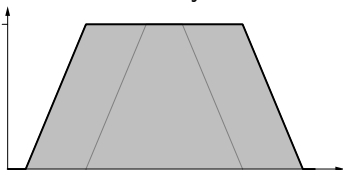
fuzzy complement



two fuzzy sets



fuzzy intersection



fuzzy union

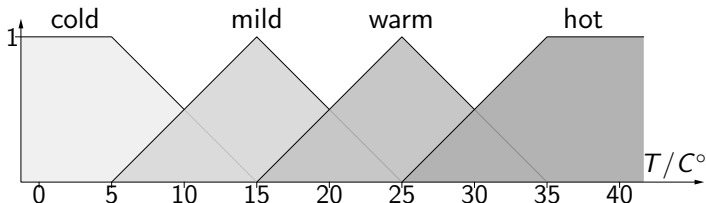
Fuzzy Partitions and Linguistical Variables

to be able to describe the domain by (linguistical) terms it is partitioned with the help of fuzzy sets

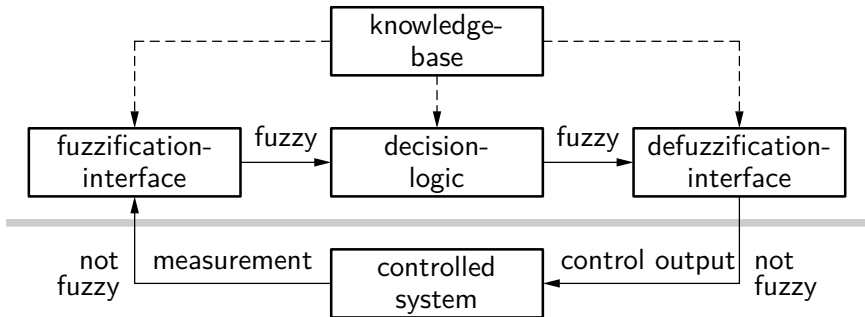
- every fuzzy set of this partitioning is assigned to a linguistical term
- condition: for every domain value the membership grades must sum up to 1

Example: fuzzy set partitioning for temperatures

linguistical variable with its values *cold*, *mild*, *warm* and *hot*.

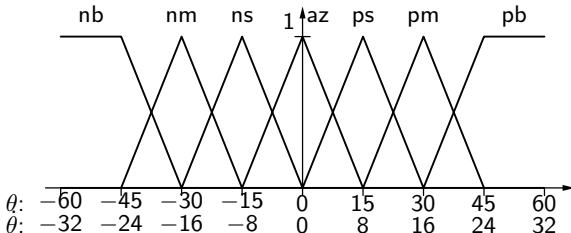
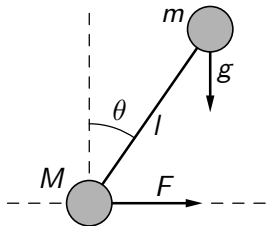


Architecture of a Fuzzy Control



- knowledge base contains fuzzy rules for control and fuzzy partitioning of the variables' domains
- fuzzy rule: **if** X_1 **is** $A_{i_1}^{(1)}$ **and** ... **and** X_n **is** $A_{i_n}^{(n)}$ **then** Y **is** B .
 X_1, \dots, X_n are measured quantities and Y is a control quantity
 $A_{i_k}^{(k)}$, B : linguistical terms (associated to fuzzy sets)

Example of a Fuzzy Control: Pole Balancing

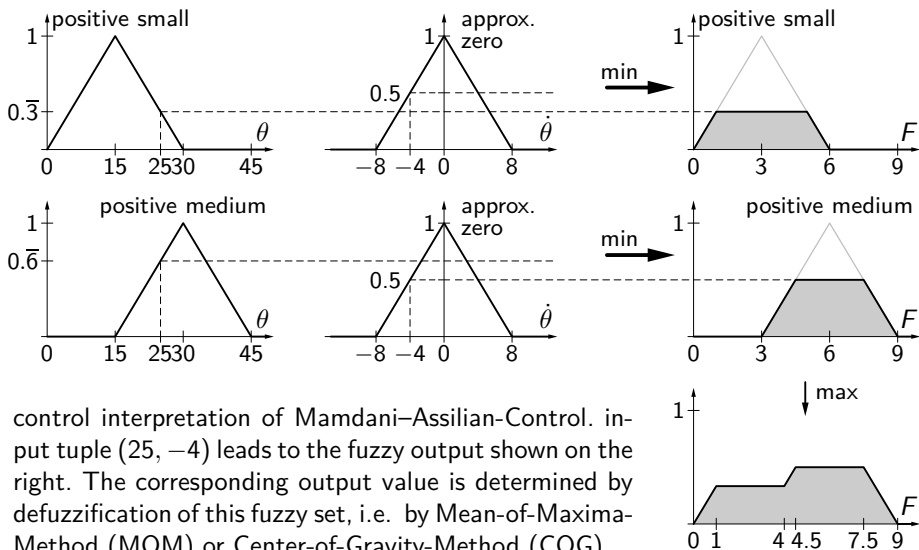


abbreviations

- pb – positive big
- pm – positive medium
- ps – positive small
- az – approximately zero
- ns – negative small
- nm – negative medium

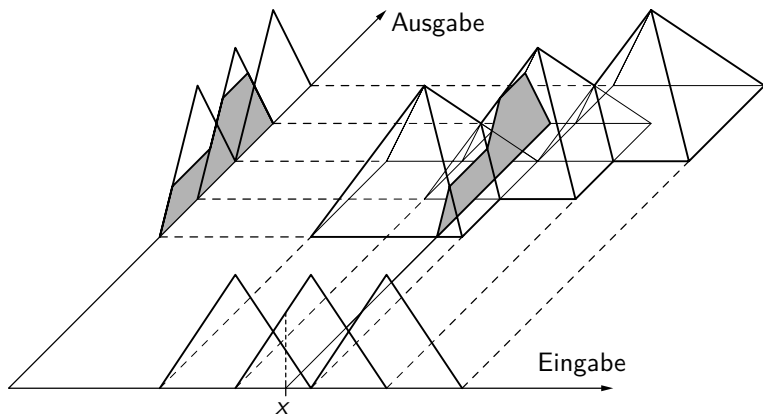
$\dot{\theta} \setminus \theta$	nb	nm	ns	az	ps	pm	pb
pb			ps	pb			
pm				pm			
ps	nm		az	ps			
az	nb	nm	ns	az	ps	pm	pb
ns				ns	az		pm
nm				nm			
nb				nb	ns		

Fuzzy Control according to Mamdani-Assilian



control interpretation of Mamdani-Assilian-Control. input tuple $(25, -4)$ leads to the fuzzy output shown on the right. The corresponding output value is determined by defuzzification of this fuzzy set, i.e. by Mean-of-Maxima-Method (MOM) or Center-of-Gravity-Method (COG).

Fuzzy Control according to Mamdani–Assilian



Fuzzy control system of one measured quantity, one control quantity and three fuzzy rules: every pyramid = one fuzzy rule, input x leads to fuzzy output drawn gray

Defuzzification

Interpretation of fuzzy rules gives **output-fuzzy-set**, which is to be transformed into a **sharp threshold**. This process is called **Defuzzification**.

main defuzzification methods:

- **Center of Gravity, COG**

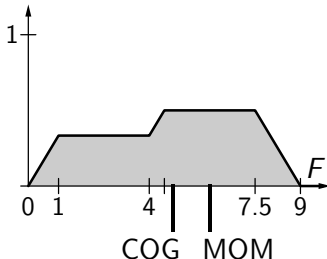
centers of the area underneath the output fuzzy set

- **Center of Area, COA**

point separating the area underneath the output fuzzy set in equally sized parts

- **Mean of Maxima, MOM**

arithmetic mean of the points of maximum membership degree



Generating/Optimizing Fuzzy Controls with EA

Mamdani-Assilian Control can be optimized by changing

- **rule base** (which rules, which output)
- **fuzzy sets/fuzzy partitions**
(shape, location, size, number of fuzzy sets)
- ***t*-norm** or ***t*-conorm** for rule interpretation (rarely)
- parameters of the **defuzzification method** (if applicable; rarely)
- which **input quantities** to use for rules
(*feature selection*)

now: only optimizing rule base and fuzzy sets with fixed choice of input/measured quantities

rule interpretation: by minimum and maximum,

defuzzification: center-of-gravity method

Possible Approaches

1. *optimize rule base and fuzzy partitioning at the same time*
disadvantage: parallel optimization of many parameters
2. *optimize fuzzy partitioning with fixed rule base first, afterwards optimize rule base with best fuzzy partitioning*
disadvantage: expertise needed for creating rule base
(choosing a random rule base is not promising)
3. *optimize rule base for fixed fuzzy sets first, afterwards fuzzy partitioning with fixed rule base*
fuzzy sets can be distributed (e.g.) equidistantly
in this case: user needs to provide a fixed number of fuzzy sets per measured quantity and for control quantity.

now: only third alternative

Fitness Function

a good control system should satisfy the following criteria:

- find the goal state for every potential situation
- reach the goal state as fast as possible
- find the goal state with minimum effort

The control is used multiple times to control a test system (in the following: computer simulation of a pole balancing problem with several different initial states)

according to success of the rules (rule goodness) the control gets credits (number of situations, time till goal state, energy consumption)

- **note:**

In this case assessment of the individuals is the main task. Every individual needs to be simulated while controlling for a minimum number of time steps.

Evaluating an Individual's Control Success

- actual value differs from the goal state a lot \Rightarrow abort (failure)
(e.g. pole balancing problem: actual value needs to be within $[-90^\circ, 90^\circ]$)
- actual value should get (and remain) close to the desire value after a certain time (range of tolerance).
otherwise abort, too (failure)
- range of tolerance is reduced with t (slowly leading to the goal state)
 - within the first generations: it is sufficient to not knock down the pole
 - later on: pole needs to stay vertically within a small angle
- absolute values of the control quantity differences are summed up and used within a penalty term
(in balanced state there is no difference in quick switching between high forces and controlling with small forces. High forces should be avoided.)

Generating/Optimizing the Rule Base: Coding

- only complete rule bases (there must be a rule for every combination of input fuzzy sets)
- for every combination of input fuzzy sets only the term of the control quantity has to be determined (table filling)

example: rule base for pole balancing controller:

$\dot{\theta} \setminus \theta$	nb	nm	ns	az	ps	pm	pb
pb	az	ps	ps	pb	pb	pb	pb
pm	ns	az	ps	pm	pm	pb	pb
ps	nm	ns	az	ps	pm	pm	pb
az	nb	nm	ns	az	ps	pm	pb
ns	nb	nm	nm	ns	az	ps	pm
nm	nb	nb	nm	nm	ns	az	ps
nb	nb	nb	nb	nb	ns	ns	az

schematically

Coding of the Rule Base

linearization (conversation to vector)

table is processed in arbitrary, but fixed order, and table entries are listed in a vector

example: listing line by line

problem: neighborhood relations

between the rows are lost

(neighboring entries should have similar
linguistical terms; this should
be considered e.g. for crossover)

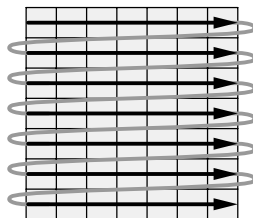


table (direct application of the scheme)

two- or multidimensional chromosomes

(special genetical operators are needed)

Variation (Standard Mutation)

rule/table entry is chosen randomly

linguistical term of the output is changed randomly

if necessary **multiple rules/table entries** are changed simultaneously

maybe useful: **limit mutation of a rule base** in such a way, that table entries can only be changed to linguistic terms that are neighboring (or sufficiently close to) the current entry

Beispiele: „positive small“ can only be changed to

„approximately zero“ or „positive medium“

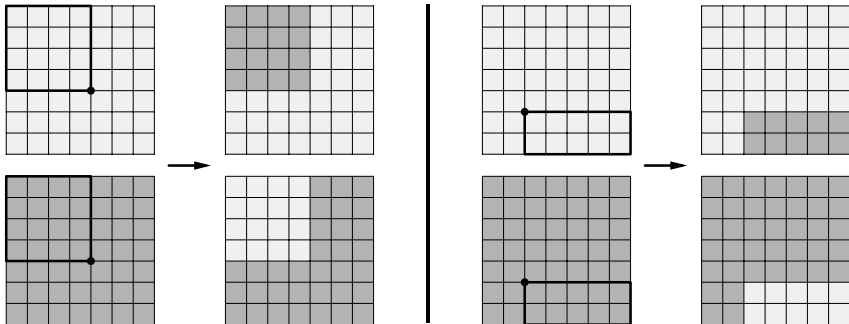
„negative big“ can only be changed to

„negative medium“ or „negative small“

(prevents fast scattering of learned information; rule base is changed “with caution“)

Recombination (One Point Crossover)

choose one inner grid point and a table corner randomly
exchange subtables defined by these two points between parents

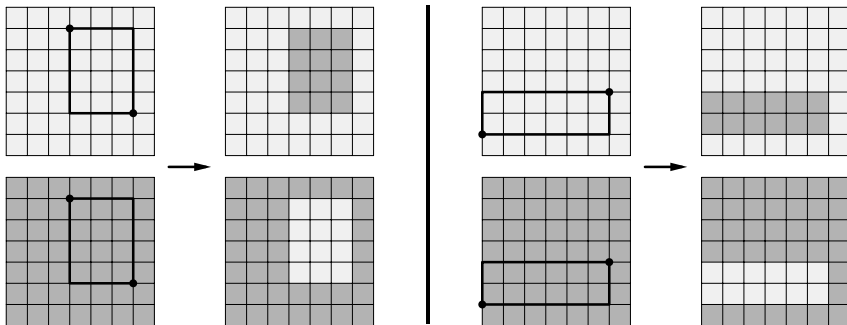


note: to prefer joint transferation of neighboring rules, crossover *should* have positional bias!

Recombination (Two Point Crossover)

choose two grid points randomly within the table (points lying on the border, too)

Exchange the subtable defined by these two points between parents



Two point crossover is more appropriate, because subsolutions can be exchanged more flexibly.

Optimizing the Fuzzy Sets

given: optimized rule base with fixed equidistantly distributed fuzzy sets

looking for: further improvements of the control unit by changing the fuzzy sets, with constant rule base („Fine-Tuning“)

coding the fuzzy sets: (first option)

- choose shape of the fuzzy sets
(e.g. triangle, trapezoidal, gaussian, parabola, spline etc.)
- list defining parameters of the fuzzy sets
(e.g. triangle: left margin, center, right margin)

e.g. pole balancing control with triangle fuzzy sets (excerpt):

...	nm			ns			az			ps			...
...	-45	-30	-15	-30	-15	0	-15	0	15	0	15	30	...

Disadvantages of this Coding

coding is very “inelastic“ regarding the shape of the fuzzy sets
e.g. previously fixed, whether triangles or trapezoidals are used

genetical operators can destroy ordering of the parameters
(e.g. for triangles “left \leq center \leq right“ must hold)

fuzzy sets „overtaking“ each other possibly: meaningful ordering
amongst the fuzzy sets might be ruined by mutation or crossover
(e.g. “ns right of ps“ should be true)

condition „sum of membership degrees = 1“ might be violated
(can be corrected by unique representation of identical
parameters)

...	-45	-15	15	-15	15	30	15	30	45	30	45	60	...
...	-45	-30	-20	-30	-20	-10	-20	-10	0	-10	10	30	...

Coding of the Fuzzy Partitions

Instead of equally distributed sampling points amongst the domain, the membership degrees of different fuzzy sets are given

$$\underbrace{\begin{pmatrix} \mu_1(x_1) \\ \vdots \\ \mu_n(x_1) \end{pmatrix}}_{\text{Gen 1}} \cdots \underbrace{\begin{pmatrix} \mu_1(x_i) \\ \vdots \\ \mu_n(x_i) \end{pmatrix}}_{\text{Gen } i} \cdots \underbrace{\begin{pmatrix} \mu_1(x_m) \\ \vdots \\ \mu_n(x_m) \end{pmatrix}}_{\text{Gen } m}$$

coding with $m \times n$ numbers from $[0, 1]$

pb	0	0	0	0	0	0	0	0	0	0	0	0	0	0	.5	1	1	1
pm	0	0	0	0	0	0	0	0	0	0	0	.5	1	.5	0	0	0	0
ps	0	0	0	0	0	0	0	0	0	.5	1	.5	0	0	0	0	0	0
az	0	0	0	0	0	0	0	.5	1	.5	0	0	0	0	0	0	0	0
ns	0	0	0	0	0	.5	1	.5	0	0	0	0	0	0	0	0	0	0
nm	0	0	0	.5	1	.5	0	0	0	0	0	0	0	0	0	0	0	0
nb	1	1	1	.5	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	-60	-45	-30		-15		0		15		30		45		60			

Genetical Operators

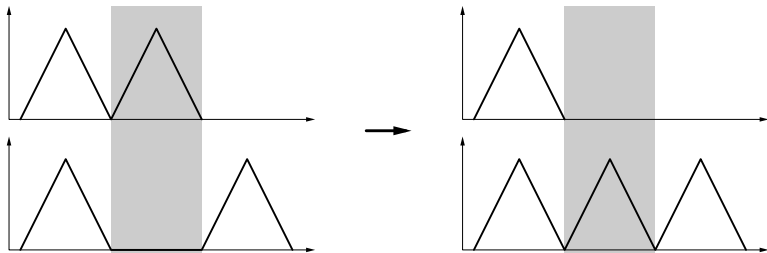
mutation: analogue to 1 bit mutation:

a randomly chosen entry is randomly manipulated

useful: limit the range of change, e.g. by defining an interval or a normal distribution

crossover: 1-point or 2-point crossover

note: crossover can extinguish fuzzy sets

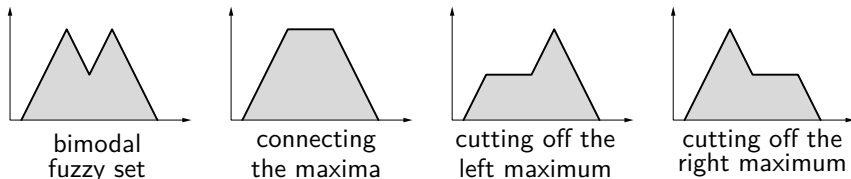


Repairing Fuzzy Sets

by mutation/crossover: membership function of fuzzy sets might not be *unimodal* anymore (a unimodal membership function has 1 local maximum only)

multimodal fuzzy sets: much more difficult to interpret, than unimodal fuzzy sets

if feasible, fuzzy sets are repaired (to make them unimodal)



extend / cut fuzzy sets in a way that they cover the total domain, but no point is covered by too many fuzzy sets

Generating/Optimizing Fuzzy Controls with GA

generation/optimization in two steps:

- optimize rule base for fixed fuzzy partitioning
- optimize fuzzy partitioning with the fixed generated rule base

during experiments: EA finds correct fuzzy control for the pole balancing problem

rule generator of EA: very costly (in terms of computation time)

advantage: only few background knowledge needed

other requirements for fitness function:

- **compactness:** little number of rules and fuzzy sets
- **completeness:** covering all relevant regions of the domain
- **consistent:** no rules with similar antecedent, but contradicting consequence
- **interpretability:** limited overlap of fuzzy sets

Literature for this Lecture I



Gerdes, I. (1994).

Application of genetic algorithms to the problem of free routing for aircraft.

In *Proceedings of the IEEE World Congress on Computational Intelligence: First IEEE Conference on Evolutionary Computation*, volume 2, page 536–541, Orlando, FL, USA.



Gerdes, I. (1995).

Application of genetic algorithms for solving problems related to free routing for aircraft.

In Biethan, J. and Nissen, V., editors, *Evolutionary Algorithms in Management Applications*, page 328–340. Springer, Berlin, Germany.