# Genetic Algorithms in Game Theory

**Lóránt Bódis**

**Babes-Bolyai University**

**Internet: www.geocities.com/lbodis**

**e-mail: lbodis@yahoo.com**

# Where I am coming from?

**Romania**

# Romania – Transylvania

# Cluj/Kolozsvár/Klausenburg

**City picture from “Fellegvár”**

# Cluj/Kolozsvár/Klausenburg

**City center**

# Babes-Bolyai University

**Central building**

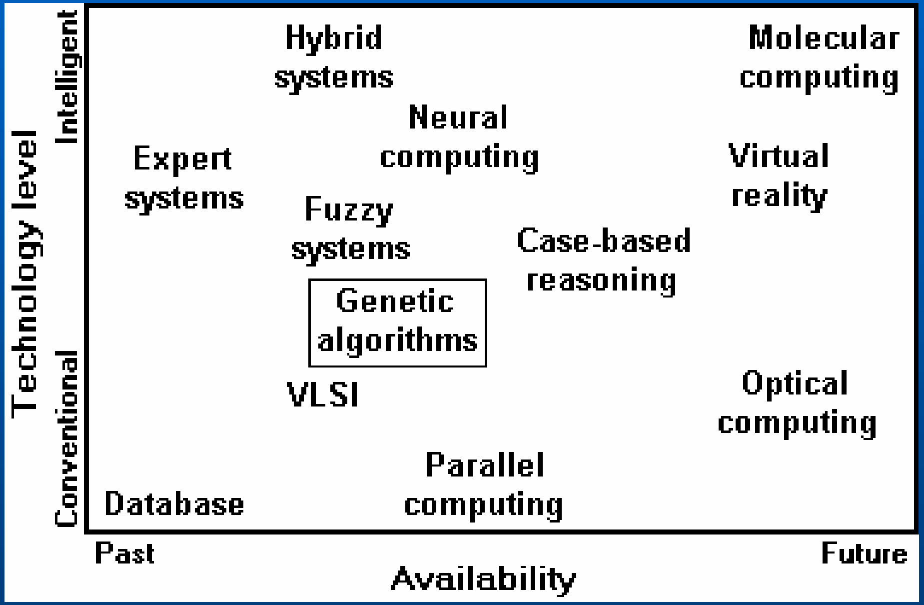
**Arms of BBU**

# Objectives

- Presentation of 2 very popular fields of Artificial Intelligence (AI) and a specific combination of these
  - Study of genetic algorithms
  - Review of game theory terminology
- Application  of evolutionary methods for problems from game theory
- Development of optimal strategies for games

# Introduction

- **Place of Genetic Algorithms between AI technologies**
  - it has a central position
  - it is used by several other methods
  - it's a relative new field




- **Games and Game Theory**
  - important research fields of the AI
  - the aim is the development of efficient search algorithms
  - the research results can be used by other fields as well

# Genetic Algorithms (GAs)

- **First introduced by John H. Holland**
- **Global optimization method**
- **Stochastic algorithm**
- **Adaptive search technique**
- **Provides a domain independent search heuristics**
- **Problem independent algorithm**
- **It has a robust structure**
- **Artificial selection**

# GA – Steps, Components

- Based on the principle of natural selection, it simulates several biological processes
- Simple representation of a problem's solutions using strings (bit strings - if possible) – chromosome (individual) representation
- The GA simultaneously works with several solutions (individuals) – generates a sequence of populations
- Evaluation function, which has the role of the environment, the estimation of solutions in pursuance of the fitness – fitness function
- Genetic operators, which are changing the content of the offspring individuals during reproduction

# GA Components – Chromosome

- **Coding of solutions**
- **A syntactically well and easy to handle letter- or number sequence**
- **The positions (indexes) of chromosome (genotype) are the genes, the values (letter or number, character) on this positions are the alleles**

# GA Components – Fitness Function

- It serves for the evaluation of solutions
- Measures the performance, competence, suitability, fitness of individuals
- Definition of fitness function can be the most difficult but also the most important task
- The aim is to find the global optimum of this function

# Genetic Operators

- **Simple transformations on chromosomes**
- **Genetic operators can be classified in 3 main groups:**
  - **selection, recombination**
  - **mutation**
  - **crossover, reproduction**
- **For the GA it can be given as parameter the mutation and crossover rates (probabilities) in order to be used these operators only for a certain number of individuals**

# Genetic Operators – Selection

- **Problem independent**
- **It chooses an individual from the population taking into account its fitness**
- **Variants:**
  - **fitness proportionate selection**
  - **tournament selection**

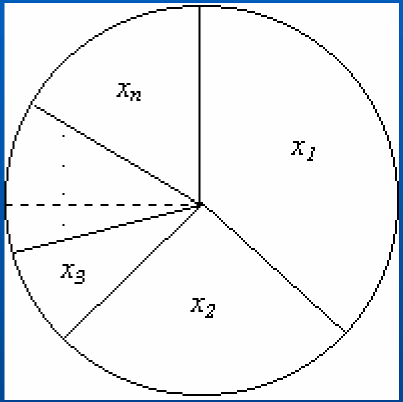
# Genetic Operators – Selection (cont.)

- **Fitness Proportionate Selection**
  - **the probability of selection of a solution is greater if his fitness is more greater compared to the population's average fitness value**
  - **the probability of selection for each element of the population:**

$$P(e) = \frac{f(e)}{nf(Pop)}$$

**where *f(e)* is the fitness value, *n* the population size, and *f(Pop)* is the average fitness of the population's elements**

  - **in practice the roulette wheel method is used, where each element of the population is represented by a slice/niche of the roulette wheel, which is straightforward proportional with the individual's fitness score**



# Genetic Operators – Selection (cont.)

- **Tournament Selection**
  - a group (typically between 2 and 7 individuals) are selected at random from the population and the best is chosen
- **Elitism**
  - an elitist genetic algorithm is one that always retains in the new population the best individual found so far
- The selection operators are used as many individuals is needed in the new population

# Genetic Operators – Mutation

- **The aim is the refreshment of the individuals; leads to additional genetic diversity**
- **Help the search process escape local optima traps**
- **Changes the values of randomly selected genes**

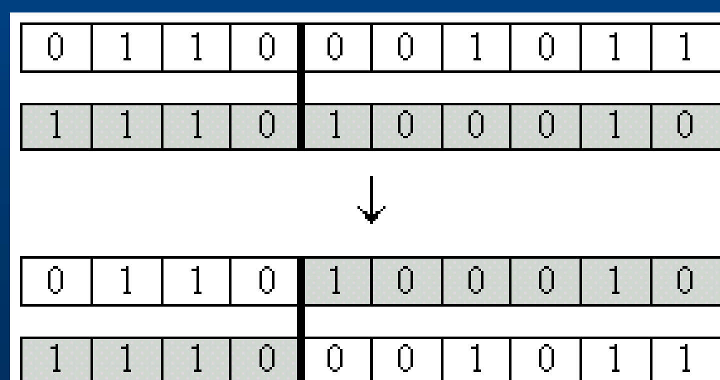
| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | 1 |

↓

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 1 | 1 |

# Genetic Operators – Crossover

- Several variants exits
- One-point crossover
  - the aim is to generate fitter individuals (offspring) by combining (exchanging bits) the properties of different individuals (parents) through
  - at crossover point the two half codes are swapped, creating new individuals (offspring)

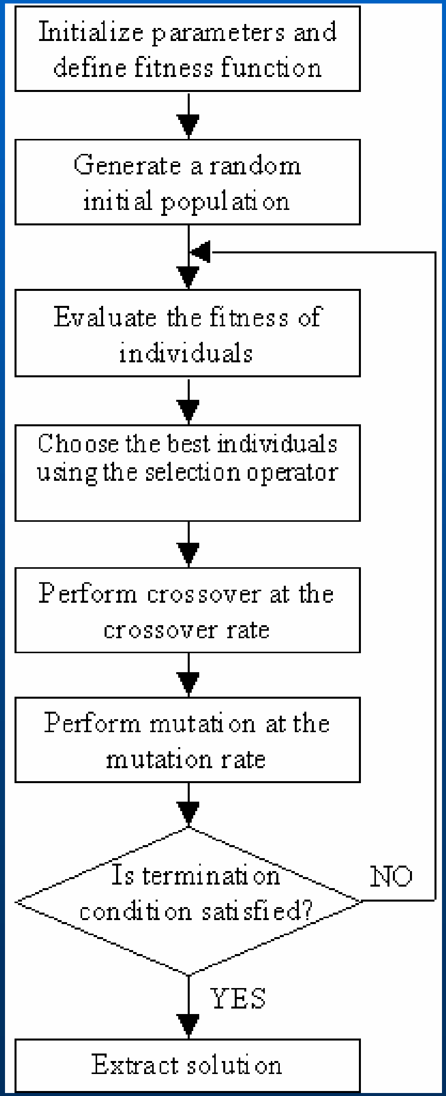


# GA – Algorithm

- Several variants known

```
Coding of individuals, genetic representation
Definition of fitness function
Setting the parameters
Generating initial population
While not(termination condition) do
  Creation of new population from parent
  population using the selection operator
  Crossover, mutation in the new population
  The new population will take the role of
  parents in the next generation/iteration
End while
```

- The **initial population** is usually generated randomly
- **Termination condition** can be the iteration number



# Connection with Metaheuristics

- GA belongs to the class of problem-independent metaheuristics
- Most common metaheuristic algorithms:
  - simulated annealing
  - tabu search
  - hill climber
- Advanced searching methods
- Global methods, containing local optima avoidance techniques

# Simulated Annealing

- **Stochastic computational technique derived from statistical mechanic**
- **Used for large optimization tasks (VLSI, wire routing)**

```
t ← 0
Initialize T initial temperature
Select randomly a v_c string
repeat
  repeat
    Select a v_n new string from the neighborhood of v_c by changing
    single bits of v_c
    if f(v_c)<f(v_n) then v_c ← v_n
                     else if random[0,1)<exp{(f(v_n)-f(v_c))/T}
                          then v_c ← v_n
  until(stop condition) //temperature equilibrium; iteration number
  T ← g(T,t)
  t ← t + 1
until(terminition condition) //T reached a low value; sys. has frozen
```

# Tabu Search

- **Iterative corrective algorithm**
- **Used for problems where the solutions are situated on the nodes of a graph**
- ***T* tabu list**
  - **contains the recently checked/examined solutions, the earliest is deleted after a time**
  - **recency based (temporary) memory**

```
s ← (initial allowed solutions) //s – current solution
s* ← s //s* - best solution found during search
k ← 1
while not(termination condition) do
  s′ ← best element from the neighbors of s – T
  Update T with s′
  s ← s′
  if s′ is better then s* then s* ← s′
  k ← k + 1
endwhile
```

# Hill Climber

- **Simple iterated (steepest ascent) hillclimbing algorithm**
- **The success of the algorithm's single iteration depends on the initial string**

```
t ← 0
repeat
  local ← FALSE
  Select randomly a v_c string
  repeat
    Select n new strings from the neighborhood of v_c by changing
    single bits of v_c
    Select the v_n string from the set of new strings, where the f
    object function value is the greatest
    if f(v_c)<f(v_n) then v_c ← v_n
                     else local ← TRUE
  until local
  t ← t + 1
until t = MAX //MAX - iteration number
```

# Comparing Metaheuristics – Example

- Find global maximum of the function:
  - $f(v) = |11*one(v)-150|$,
    where one(v) = number of 1s in the 30 length v binary string
  - global maximum: $v_g = (111...111)$, $f(v_g) = |11*30 - 150| = 180$
  - local maximum: $v_l = (000...000)$, $f(v_l) = |11*0 - 150| = 150$
- HC – sometimes finds only the local maximum
  - ex. initial string contains 13 ones (function value 7)
  - 14 ones – function value 4; 12 ones – function value 18
- SA – handles easier this task, because with certain probabilities accepts worse solutions, which helps the algorithm to get out of local optima
  - ex. $v_c$ has 12 ones, $v_n$ 13 ones
  - $p=\exp\{(f(v_n)-f(v_c))/T\}=\exp\{(7-18)/T\}$; if T=20 then $p=e^{-11/20}=0,576$
- GA – finds the global maximum using relatively low iteration number and avoids easily local optima traps

Application

# GA – Applications

- It's worth to use for tasks, which:
  - have a large search-space
  - don't have domain-specific description, knowledge
- NP-hard problems
  - graph coloring, traveling salesman problem (TSP), binpacking, backpack problem, SAT
- Problems with large search-space
  - Function optimization, machine learning, evolving artificial neural networks (ANN), combinatorial problems
- Applying for game theory problems

# Game and Game Theory

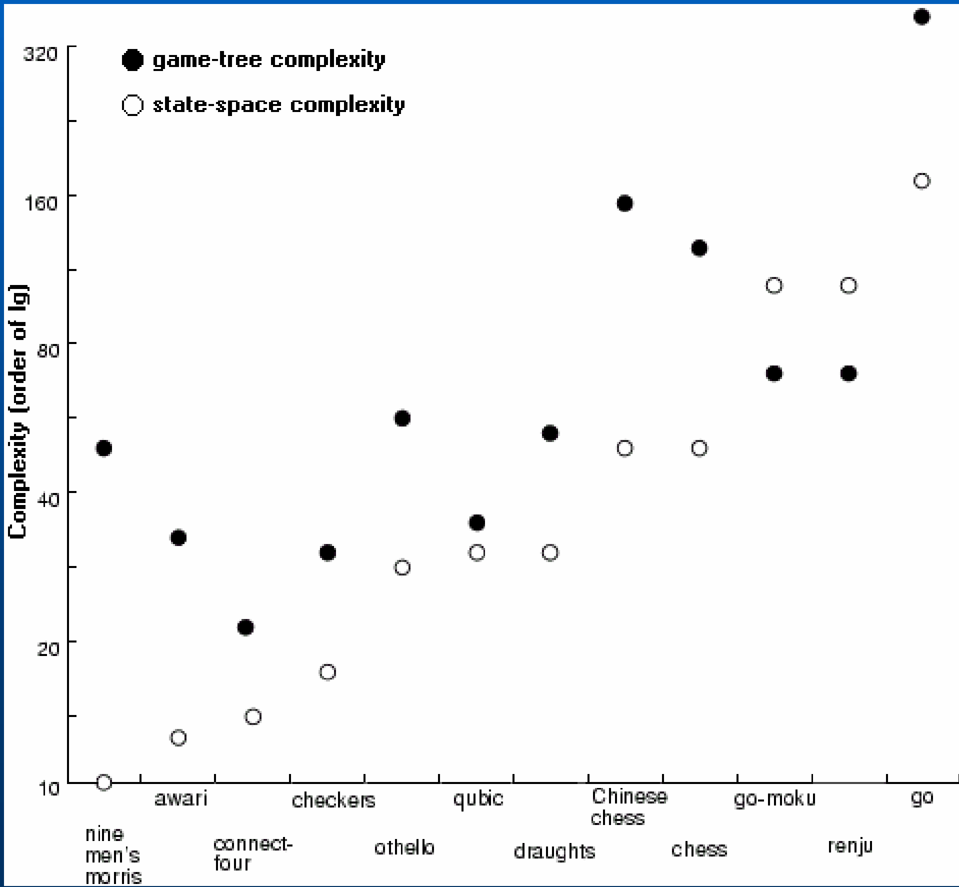
- **Game**
  - it requires from a person a high level of intelligence, cognitive activity
  - task, whose solution is searched by the AI with the help of computers
  - the games were the first to pique the interest of researchers, because it was a great challenge: creating programs that are capable of exceeding the performance and ability of humans
  - *Deep Blue* chess program is a great achievement – in 1997 defeated the World Chess Champion
  - finding the solutions of strategic games is the research area of machine learning
- **Game Theory**
  - symbiosis of mathematics, economics and computer science
  - Neumann analyzed economic behaviors through the games
  - used to explain strategic reasoning, conclusions

# Game Properties, Classification

- **Players number**
  - one, two or more players
- **Information**
  - perfect information – every player has access to all information, they know the rules of the game, the previously done moves and the current state
  - imperfect information
- **Zero-sum**
  - the sum of a player's wins and loses is zero
- **Finiteness**
  - finite – from a given state there are finite number of possibilities and the game ends in a finite time
  - infinite

# Game Complexity

- **State-space complexity**
  - number of legal game positions reachable from the initial position
    - TTT: $3^9 = 19683$ upper bound, 5478 sharper upper bound
    - Go-Moku: $3^{225} \approx 10^{105}$
    - Chess: $10^{50}$
- **Game-tree complexity**
  - number of leaf nodes in the solution search tree of the initial position
    - TTT: 9!
    - Go-Moku: $210^{30} \approx 10^{70}$
    - Chess: $35^{80} \approx 10^{123}$



# IPD – Presentation

- **Iterated Prisoner's Dilemma – IPD**
  - two player, non zero-sum, imperfect information (non cooperative), infinite, social game
  - during a move the players have to make a decision separately (choice between cooperation and defection); any previous communication between the players is not allowed
  - according to the decisions the players will receive points, which is also told to the other player
  - this process is repeated/iterated *n* times, none of players know when ends the game
  - the aim is to maximize the accumulated points by each of the players
  - a game in IPD is a choice by each player in one

# IPD – Game Strategies

- **Robert Axelrod** handled the choice-pairs as parameters – they have names and values, relations between them
- a problem is IPD, if: *T>R>P>S* and *2R>S+T*
- simulates different social, economical, military and political interactions (“**arms race**”)
- choices in one move might affect the future choices of the other player (partner)

| | | Second player | |
|---|---|---|---|
| | | *Cooperate* | *Defect* |
| First player | *Cooperate* | (3 , 3) *Reward for mutual cooperation* | (0 , 5) *Sucker's payoff, and Temptation to defect* |
| | *Defect* | (5 , 0) *Temptation to defect, and Sucker's payoff* | (1 , 1) *Punishment for mutual defection* |

- **TF2T**: cooperates except if opponent has defected twice consecutively
- **PAVLOV**: cooperates if and only if both players choose the same option in the previous move
- **SPITEFUL**: cooperates until partner defects, subsequently always defect
- **ALLC**, **ALLD**, **RANDOM** etc.
- for years the **TFT** was considered the best strategy

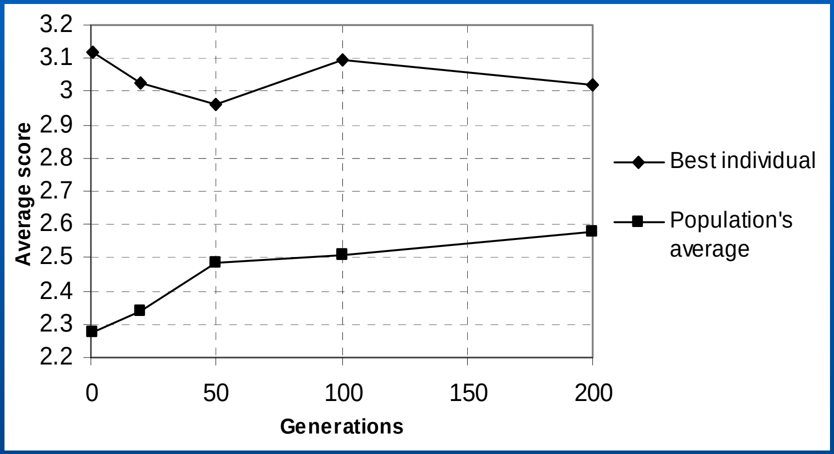
# IPD – Evolving Strategy with GA

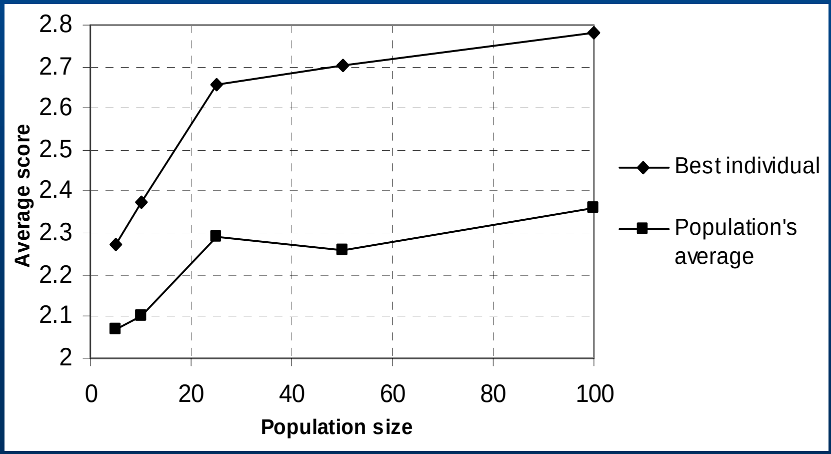
- **Representation**
  - a player in the current game makes a choice (C/D, 1/0) on the basis of the outcomes of the previous 3 games (6 moves) – $2^6$=64 different combinations
  - the strategy is a 64 bits long string which contains the answers for every possible game-sequence
  - we add the initial 6 moves in order to start the series of games, the chromosome becomes a 70 length bit string
- **Fitness function**
  - competitive fitness function – every individual plays a certain number of games against every other population member (full competition)
  - fitness value = sum of the points (based on the payoff matrix) received by the individual through all the games (game numbers*population size if is self-play)
- **Genetic operators**
  - 1-point crossover (rate of 25%) and mutation (rate of 1%)

Application

# IPD – Evaluation, Results

- **crossover probability: 0,25**
- **mutation probability: 0,01**
- **game-pair numbers: 150**
- **roulette  wheel selection**
- **generation-number: for lower values the defection is common, but in the course of iterations more and more individuals begin to cooperate, thus the average score tends to 3**
  - **population size: 50**
- **population size: with more individuals a better strategy is evolved, adapting to a strong and diversified/varied environment**
  - **iteration-number: 50**
- **the results are the average of 10 consecutive runs**



# Puzzle Game – Presentation

- one person, perfect information
- 9-cell puzzle
  - on a 3x3-as grid 9 numbered squares/cells
  - the aim is to reach a given final state starting from an initial random state/configuration
  - the cells in the same row or column can be exchanged
- 8-cell puzzle
  - 8 blocks and an empty square
  - the cells can be moved on the empty space

# 9-cell Puzzle – GA

| | | |
|---|---|---|
| 8 | 4 | 7 |
| 2 | 1 | 9 |
| 6 | 5 | 3 |

→

| | | |
|---|---|---|
| 1 | 2 | 3 |
| 4 | 5 | 6 |
| 7 | 8 | 9 |

- **Representation**
  - exchange of cells is coded with 0-18 numbers
  - chromosome: sequence of these codes, which transforms the initial state to another state (final state when the solution is found)
- **Fitness function**
  - difference between the current and final state
  - the more cells are in correct position the best is the state
  - individual's fitness:

$$25 - \sum_{i=1}^{9} d(A_i, V_i)$$

where $A_i$ denotes the current position for number $i$, $V_i$ the correct position and $d$ distance between the two states

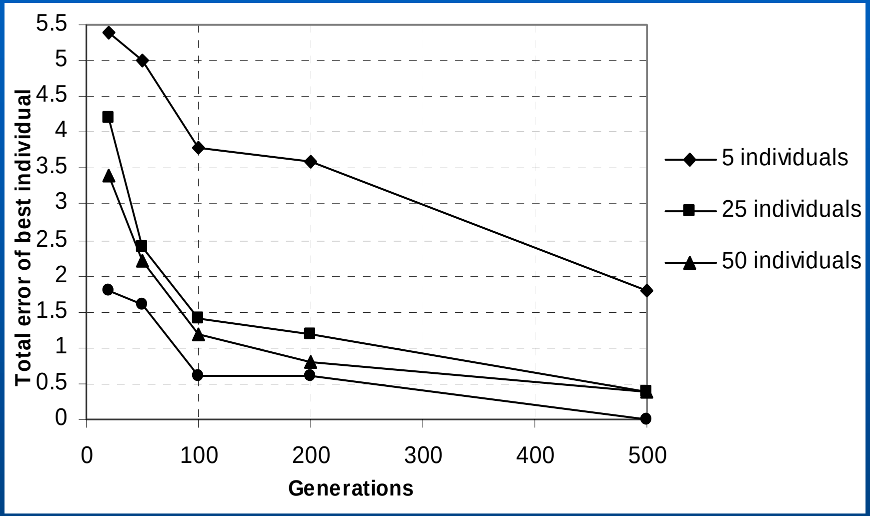
ex. total distance between the two states on the figure is 20

- **Genetic operators**
  - 1-point crossover (rate of 50%) and mutation (rate of 2%)

Application

# 9-cell Puzzle – Evaluation, Results

- Results of the different population sizes in the function of generations
  - error: discrepancy between the current and final state
  - crossover probability: 0,5
  - mutation probability: 0,02
  - depth of game-tree: 15
  - elitist tournament selection



# TTT – Presentation

- **Tic-Tac-Toe – TTT**
  - **Two person, perfect information, zero-sum, board game**
  - **players alternate placing their markers (*X* respectively *O*) on a 3x3 grid and the first player to obtain 3 in a row horizontally, vertically or diagonally wins**
  - **if there is no winner and no more free cell the game is called draw game**

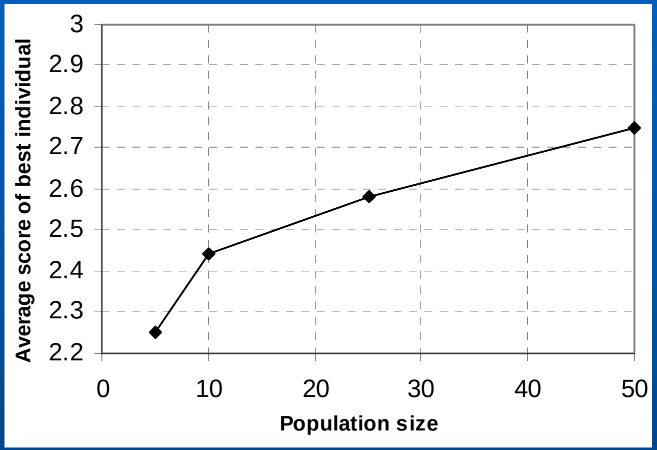
# TTT - Evolving Strategy with GA

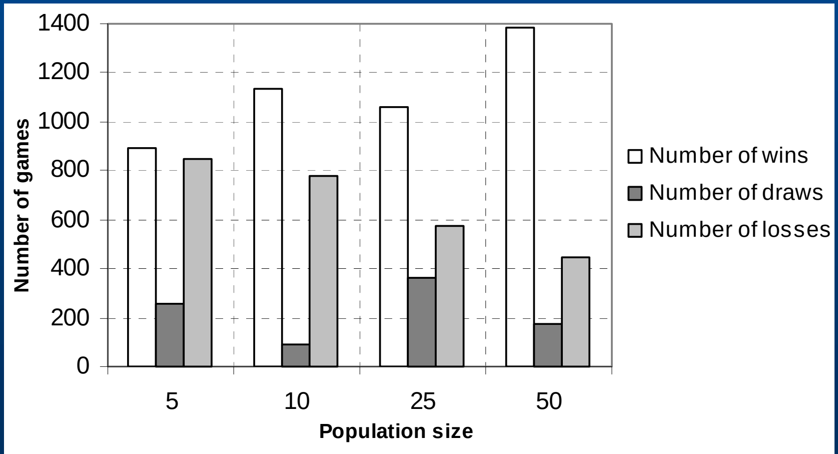
- **Representation**
  - every possible position which does not have a symmetric form, does not contain a win, has at least to opened squares, it's not initial - or final state – totally 593 positions
  - chromosome: 593 genes, the allele contains the position number which the player marks in the current state
- **Fitness function**
  - competitive fitness function – every individual plays a certain number of games against every other population member (full competition)
  - during a game-pair players alternate starting order
  - at the end of a game, strategies receive scores based on their results (win, loss or draw)
  - fitness function = sum of the points received by the individual through all the games (game numbers*population size if is self-play)
- **Genetic operators**
  - 1-point crossover and mutation

Application

# TTT – Evaluation, Results

- **population size: at greater values the individuals are learning more**
  - **maximum average score is 3, (*Win*=3, *Draw*=2, *Lose*=1)**
  - **iteration number: 100**
  - **crossover probability: 0,05**
  - **mutation probability: 0,08**
  - **number of game-pairs: 2**
  - **tournament selection**
- **testing the strategy: 1000 pairs of games were played with the RANDOM strategy**



# Conclusions

- The genetic algorithms have proved to be an efficient global optimization method
- GA can be well used for different problems from game theory, where, the search-space is large and there is no concrete domain-specific knowledge
- The GA was able to evolve intelligent behavior patters in a relative short time for the Iterated Prisoners Dilemma problem

**Further information...**

- Diploma Work
- References

# Acknowledgements


- Dr. Anna Soós for the guidance in writing my diploma work
- TU München for the access to the rich bibliography provided by the library
- Professor Dr. Ernö Pretsch for giving me the opportunity and providing the necessary conditions to prepare this presentation at ETH Zürich
- My parents for the material and spiritual support