EE496 : COMPUTATIONAL INTELLIGENCE
EA04 : SWARM INTELLIGENCE

UGUR HALICI
METU: Department of Electrical and Electronics Engineering (EEE)
METU-Hacettepe U: Neuroscience and Neurotechnology (NSNT)

Swarm- and Population-Based Optimization

swarm intelligence

- part of AI's developing intelligent multi-agent systems
- inspired by the behaviour of certain species, in particular
    - social insects (e.g. ants, termites, bees etc.) and
    - animals living in swarms (e.g. fish, birds etc.)

these species are capable of solving complex tasks by cooperation.

main idea

- generally quite simple individuals with limited skills
- self-coordinated without central control unit
- individuals exchanging information (cooperation)

techniques are classified by their way of information exchange

swarm and population based algorithms

- swarm and population based algorithms: heuristics for solving optimization problems
- **purpose:** finding a good approximation of the solution
- attempt to reduce the **problem of local optima** (by improving exploration of the search space)
- important: **exchange of information** between individuals (depending on the principle: different types of algorithms)
- **particle swarm optimization**
  - optimization of a function with real agruments
  - exchange of information by watching the neighbors
- **ant colony optimization**
  - search for best routes (abstract: within a decision graph)
  - exchange of information: manipulation of the environment (stigmergy)

Techniques

Genetic/Evolutionary Algorithms

- biological pattern: evolution of life
- exchange of information by recombination of genotypes
- every individual serves as a candidate solution

Population Based Incremental Learning

- biological pattern: evolution of life
- exchange of information by prevalence in population
- every individual serves as a candidate solution

Techniques

Particle Swarm Optimization

- biological pattern: foraging of fish or bird swarms for food
- exchange of information by aggregation of single solutions
- every individual serves as a candidate solution

Ant Colony Optimization

- biological pattern: ants searching a route for food
- exchange of information by manipulating their environments (stigmergy, extended phenotype to Darwin)
- individuals generate a candidate solution

Population based incremental learning (PBIL)

- genetic algorithm without population
- instead: only store population statistics ⇒ by $G = \{0, 1\}^L$ for all L bits the frequency of „1“
- specific individuals (e.g. for evaluation) are generated randomly according to the statistical frequency
- recombination: uniform crossover ⇒ implicitly when generating an individual
- selection: choosing the best individuals B for updating the population statistics $Pr^{(t)}_k \leftarrow B_k \cdot \alpha + Pr^{(t-1)}_k (1 - \alpha)$
- mutation: bit-flipping ⇒ slightly random changes within the population statistics

Population Based Incremental Learning

Algorithm 1 PBIL   POPULATION BASED INCREMENTAL LEARNING

Input: evaluation function F
Output: best individual A_{best}

$t \leftarrow 0$
$A_{\text{best}} \leftarrow$ create random individual from $\mathcal{G} = \{0,1\}^L$ → Genetic representation
$Pr^{(t)} \leftarrow (0.5, \ldots, 0.5) \in [0,1]^L$
while termination condition not satisfied {
  $P \leftarrow \emptyset$
  for $i \leftarrow 1, \ldots, \lambda$ {
    $A \leftarrow$ generate individual from $\{0,1\}^L$ according to $Pr^{(t)}$
    $P \leftarrow P \cup \{A\}$
  }
  evaluate P according to F
  $B \leftarrow$ select best individuals P
  if $F(B) \succ F(A_{\text{best}})$ {
    $A_{\text{best}} \leftarrow B$
  }
  $t \leftarrow t + 1$
  for each $k \in \{1, \ldots, L\}$ {
    $Pr_k^{(t)} \leftarrow B_k \cdot \alpha + Pr_k^{(t-1)}(1-\alpha)$
  }
  for each $k \in \{1, \ldots, L\}$ {
    $u \leftarrow$ draw a random number according to $U((0,1])$
    if $u < p_m$ {
      $u' \leftarrow$ draw a random number according to $U(\{0,1\})$
      $Pr_k^{(t)} \leftarrow u' \cdot \beta + Pr_k^{(t)}(1-\beta)$
    }
  }
}
return A_{best}

λ: population size

α: learning rate
low: exploration
high: exploitation

p_m: mutation rate

β: mutation constant

PBIL: Typical Parameters

learning rate α

- low: emphasizes exploration
- high: emphasizes fine tuning

parameter	co-domain
population size	20–100
λ learning rate α	0.05–0.2
mutation rate p_m	0.001–0.02
mutation constant β	0.05

PBIL: Problems

- algorithm might learn depencencies between certain single bits
- PBIL considers single bits isolated of each other

example:

population 1					population 2			
1	1	0	0	individual 1	1	0	1	0
1	1	0	0	individual 2	0	1	1	0
0	0	1	1	individual 3	0	1	0	1
0	0	1	1	individual 4	1	0	0	1
0.5	0.5	0.5	0.5	population statistics	0.5	0.5	0.5	0.5

- same population statistics can represent different populations

Particle Swarm Optimization

- fish or birds are searching for rich food resources in swarms
- orientation based on individual search (cognitive part) and other individuals close to them within the swarm (social part)
- also: living within a swarm reduces the risk of getting eaten by a predator

Particle Swarm Optimization [Kennedy and Eberhart, 1995]

- **motivation:** behaviour of swarms of fish (e.g.) when searching for food: randomly swarming out, but always returning to the swarm to exchange information with the other individuals
- **approach:** use a “swarm“ of m candidate solutions instead of single ones
- **preconditions:** $\Omega \subseteq$ IRn and thus the function f, $f : R^n \rightarrow R$ to be maximized (w.l.o.g.)
- **procedure:** take every candidate solution as a “particle“ searching for food at the position x_i with a velocity of v_i. (i = 1, . . . ,m)

⇒ combine elements of ground-oriented search (e.g. gradient descent approach) and population-based search (e.g. EA)

Particle Swarm Optimization

update for position and velocity of particle i:

$$v_i (t + 1) = \alpha v_i (t) + \beta_1(x^{(local)}{}_i(t) - x_i(t)) + \beta_2(x^{(global)}(t) - x_i (t))$$

$$x_i (t + 1) = x_i (t) + v_i (t)$$

- parameter: β_1, β_2 randomly for every step, α decreasing with t
- $x^{(local)}{}_i$ is **local memory** of an individual (particle): the best coordinates being visited by this individual within the search space, i.e.

  $$x^{(local)}{}_i = x_i(\arg\max^t{}_{u=1} f (x_i (u)))$$
- $x^{(global)}$ is **global memory** of the swarm: the best coordinates being visited by any individual of the swarm within the search space (best solution so far), i.e.

  $$x^{(global)}(t) = x^{(local)}{}_j(t) \text{ with } j = \arg\max^m{}_{i=1} f (x_i{}^{(local)})$$

Algorithm 2 Particle swarm optimization

```
for each particle i {
    x_i ← choose randomly within search space Ω
    v_i ← 0
}
do {
    for each particle i {
        y ← f(x_i)
        if y ≥ f(x_i^(local)) {
            x_i^(local) ← x_i
        }
        if y ≥ f(x_~~i~~^(global)) {
            x^(global) ← x_i
        }
    }
    for each particle i {
        v_i(t+1) ← α · v_i(t) + β_1 (x_i^(local)(t) − x_i(t)) + β_2 (x^(global)(t) − x_i(t))
        x_i(t+1) ← x_i(t) + v_i(t)
    }
} while termination condition is not satisfied
```

Line 11 (math form): if $y \geq f\left(\boldsymbol{x}_{\cancel{i}}^{(\text{global})}\right)$ {

Line 16 (math form): $\boldsymbol{v}_i(t+1) \leftarrow \alpha \cdot \boldsymbol{v}_i(t) + \beta_1 \left(\boldsymbol{x}_i^{(\text{local})}(t) - \boldsymbol{x}_i(t)\right) + \beta_2 \left(\boldsymbol{x}^{(\text{global})}(t) - \boldsymbol{x}_i(t)\right)$

Extensions

- **reduced search space:** if Ω is a proper subset of R^n (e.g. hypercube $[a, b]^n$), then all particles will be reflected and bounce off the boundaries of the search space
- **local environment of a particle:** use best local memory of a subgroup instead of global swarm memory, e.g. particles surrounding the currently updated one
- **automatic parameter adjustment:** e.g. changing the swarm size (particles being much worse than the currently updated one are extinguished)
- **diversity control:** prevent early convergence to suboptimal solutions e.g. by introducing a new random number for updating the speed to increase diversity