Nature Inspired Search and Optimisation
Advanced Aspects of Nature Inspired Search and Optimisation

Lecture 9: Constraint Handling in Evolutionary Algorithms (II)

Shan He

School for Computational Science
University of Birmingham

February 10, 2020

Outline of Topics

Penalty function, Fitness Function and Selection

- **Minimisation problem** with a penalty function $\Phi(\mathbf{x}) = f(\mathbf{x}) + rG(\mathbf{x})$, where $r > 0$
- **Question**: why $r > 0$? Hint: $G(\mathbf{x}) = \max(0, g(\mathbf{x}))^{\beta}$,
- Given two individual $\mathbf{x_1}$ and $\mathbf{x_2}$, their fitness values are now determined by $\Phi(\mathbf{x})$
- Because fitness values are used primarily in selection: Changing fitness $\rightarrow$ changing selection probabilities
- How penalty coefficient r affects selection?

Impact of r on selection: explanation

Comparison of the fitness values of two individuals of a **minimisation problem** with a penalty function $\Phi(\mathbf{x_1}) < \Phi(\mathbf{x_2})$:

$$f(\mathbf{x_1}) + rG(\mathbf{x_1}) < f(\mathbf{x_2}) + rG(\mathbf{x_2}),$$

where $r > 0$

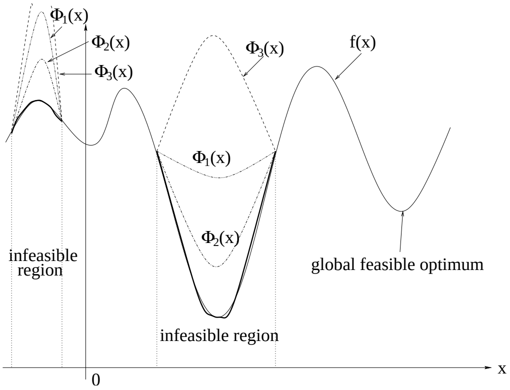
- Rewrite as

$$f(\mathbf{x_1}) - f(\mathbf{x_2}) + r(G(\mathbf{x_1}) - G(\mathbf{x_2})) < 0$$

  - $f(\mathbf{x_1}) < f(\mathbf{x_2})$ and $G(\mathbf{x_1}) < G(\mathbf{x_2}) \rightarrow f(\mathbf{x_1}) - f(\mathbf{x_2}) < 0$ and $G(\mathbf{x_1}) - G(\mathbf{x_2}) < 0$: change r ($r > 0$) has no impact on the comparison
  - $f(\mathbf{x_1}) < f(\mathbf{x_2})$ and $G(\mathbf{x_1}) > G(\mathbf{x_2}) \rightarrow f(\mathbf{x_1}) - f(\mathbf{x_2}) < 0$ and $G(\mathbf{x_1}) - G(\mathbf{x_2}) > 0$: Increasing r will eventually change the comparison
  - $f(\mathbf{x_1}) > f(\mathbf{x_2})$ and $G(\mathbf{x_1}) < G(\mathbf{x_2}) \rightarrow f(\mathbf{x_1}) - f(\mathbf{x_2}) > 0$ and $G(\mathbf{x_1}) - G(\mathbf{x_2}) < 0$: Decreasing r will eventually change the comparison

Penalties and Fitness Landscape Transformation

- In essence, different r lead to different ranking of individuals in the population
- Inappropriate penalty functions lead to infeasible results: setting r is difficult



Penalty Functions Demystified

- Penalty function essentially:
  - Transforms fitness
  - Changes rank $\rightarrow$ changes selection
- Why not change the rank directly in an EA?
- Stochastic Ranking

Stochastic Ranking

- Proposed by Dr Runarsson and Prof. Xin Yao in our school in 2000. Paper is here.
- A **special rank-based selection scheme** that handles constraints
- There is no need to use any penalty functions
- It's self-adaptive: few parameters to set and also **not sensitive**
- Became the one of the popular constraint handling techniques due to its effectiveness and simplicity

Ranking Selection

- Sort population size of M from **best to worst** according to their fitness values:

$$x'_{(M-1)}, x'_{(M-2)}, x'_{(M-3)}, \cdots, x'_{(0)},$$

- Select the top γ-ranked individual x'_{γ} with probability $p(\gamma)$, where $p(\gamma)$ is a ranking function, e.g.
  - linear ranking
  - exponential ranking
  - power ranking
  - geometric ranking

Penalty Functions Demystified

- Penalty function essentially performs:
  - Fitness (objective) function transformation
  - Rank change $\longrightarrow$ selection change
- Why not change the **rank** directly in an EA?
- Ranking = sorting: we can modify the sorting algorithm in EA to consider constraint violation
- Stochastic ranking: essentially a modified bubble sort algorithm with some additional rules to handle G

Stochastic Ranking

Stochastic Ranking Algorithm (based on bubble sort)

for $j := 1$ to M **do** // *Double loops to iterate all pairs of individuals*
  for $i := 2$ to M **do**
    $u := U(0;1)$, // u *is a uniformly distributed random number*
    if $G(x'_{i-1}) = G(x'_i) = 0$ OR $u \leq P_f$ **then**
    // *Swap them so that the better (smaller) one is before the worse one*
    // *Note: from best to worst, so currently* x'_i *ranked before* x'_{i-1}
      if $f(x'_{i-1}) < f(x'_i)$ **then**
        swap(I_i, I_{i-1});
        swap($f(x'_i)$, $f(x'_{i-1})$);
        swap($G(x'_i)$, $G(x'_{i-1})$);
    else // *which means there are constraint violations*
      if $G(x'_{i-1}) < G(x'_i)$ **then** //*only compare constraint violations*
        swap(I_i; I_{i-1});
        swap($f(x'_i)$, $f(x'_{i-1})$);
        swap($G(x'_i)$, $G(x'_{i-1})$);

M is the number of individuals, I is the indices of the individuals, $G(\cdot)$ is the sum of constraint violation and P_f **is a constant that indicates the probability of using the objective function for comparison in ranking.**

The role of P_f

- **Question**: why introduce P_f, which essentially enables infeasible solutions (whose fitness values are better) to be ranked higher than feasible solution (whose fitness values are worse) with some probability?

The role of P_f

- $P_f > 0.5$:
  - Most comparisons are based on $f(x)$ only
  - Infeasible solutions are likely to occur
- $P_f < 0.5$:
  - Most comparisons are based on $G(x)$ only
  - Infeasible solutions are less like to occur, but the solutions might be poor
- As recommended in the paper, P_f is often set between 0.45 and 0.5

Penalty Functions Demystified

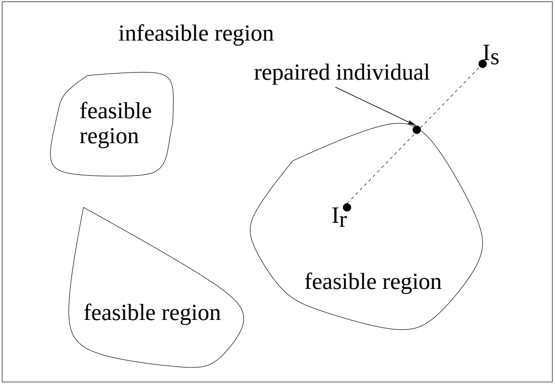
- Penalty function essentially performs:
  - Fitness transformation
  - Rank change $\longrightarrow$ selection change
- Stochastic ranking change ranks by changing the sorting algorithm
- Why not change **selection** directly in an EA?

Feasibility rule

- Proposed by Deb in 2000
- Based on (binary) tournament selection
- After randomly choose k ($k = 2$) individuals to form a tournament, apply the following rules:
  - Between 2 feasible solutions, the one with better fitness value wins
  - Between a feasible and an infeasible solutions, the feasible one wins
  - Between 2 infeasible solutions, the one with lowest G wins
- Pros: simple and parameter free
- Cons: premature convergence

Repair Approach to Constraint Handling

- Instead of modifying an EA or fitness function, infeasible individuals can be "repaired" into feasible ones.
- Let I_s be an infeasible individual and I_r a feasible individual

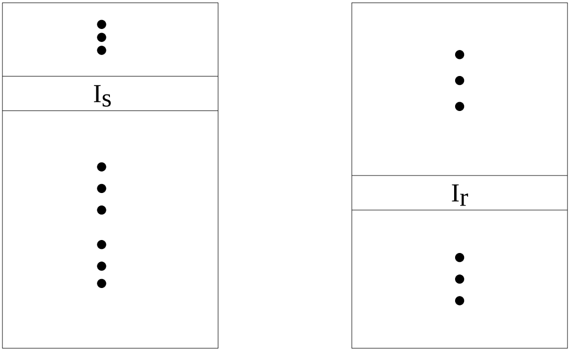


Repairing Infeasible Individuals

- We maintain two populations of individuals:

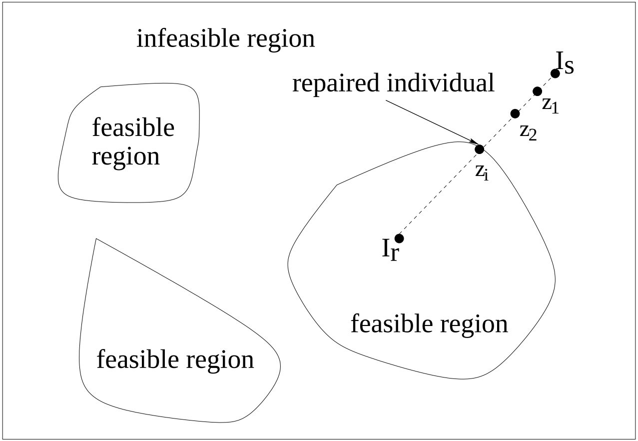
population of evolving individuals (feasible or infeasible)

population of feasible reference individuals (changing but not evolving)



Repairing Infeasible Individuals

Let I_s be an infeasible individual and I_r a feasible individual.



Repairing Algorithm

Repairing Algorithm

Select a reference individual I_r.
DO the following until individual z_i is feasible

$z_i = a_i I_s + (1 - a_i) I_r$ where $0 < a_i < 1$.

Calculate the fitness value of z_i: $f(z_i)$
IF $f(z_i) \leq f(I_r)$ THEN Replace I_s by z_i
ELSE

$u := U(0; 1)$, // u *is a uniformly distributed random number;*
IF $u \leq Pr$ THEN Replace I_s by z_i

Question: Why we still replace I_s by z_i with some probability Pr even z_i is worse than I_r, i.e., $f(z_i) > f(I_r)$

Repairing Algorithm: Implementation Issues

- How to find initial feasible reference individuals?
  - Preliminary exploration
  - Human knowledge
- How to select I_r
  - Uniformly at random
  - According to the fitness of I_r
  - According to the distance between I_r and I_s
- How to determine a_i
  - Uniformly at random between 0 and 1
  - Using a fixed sequence, e.g., $\frac{1}{2}$, $\frac{1}{4}$, $\cdots$
- How to choose Pr: A small number, usually < 0.5

Conclusion

- Adding a penalty term to the objective function is equivalent to changing the fitness function, which is in turn equivalent to changing selection probabilities.
- It is easier and more effective to change the selection probabilities directly and explicitly: stochastic ranking and feasibility rules
- There are other constraint handling techniques such as repairing methods (see a review paper in the reading list)

Further reading

- T. P. Runarsson, X. Yao, Stochastic ranking for constrained evolutionary optimization, IEEE Transactions on Evolutionary Computation, 4(3):284-294, September 2000.
- S. He, E. Prempain and Q. H. Wu, An improved particle swarm optimizer for mechanical design optimization problems, Engineering Optimization, 36(5):585-605, 2004
- E. Mezura-Montesa, C. A. Coello Coellob, Constraint-handling in nature-inspired numerical optimization: Past, present and future. Swarm and Evolutionary Computation, 2011