

Lecture 4: Constraint Handling

1. Review of the last lecture
 - (a) Recombination on the real-valued representation
 - (b) Mutation on the real-valued representation
2. Constraints and Constraint Handling Techniques
3. Different Penalty Methods
4. Stochastic Ranking
5. Summary

What Is Constrained Optimisation

The general problem we considered here can be described as:

$$\min_{\mathbf{x}} \{f(\mathbf{x})\}$$

subject to

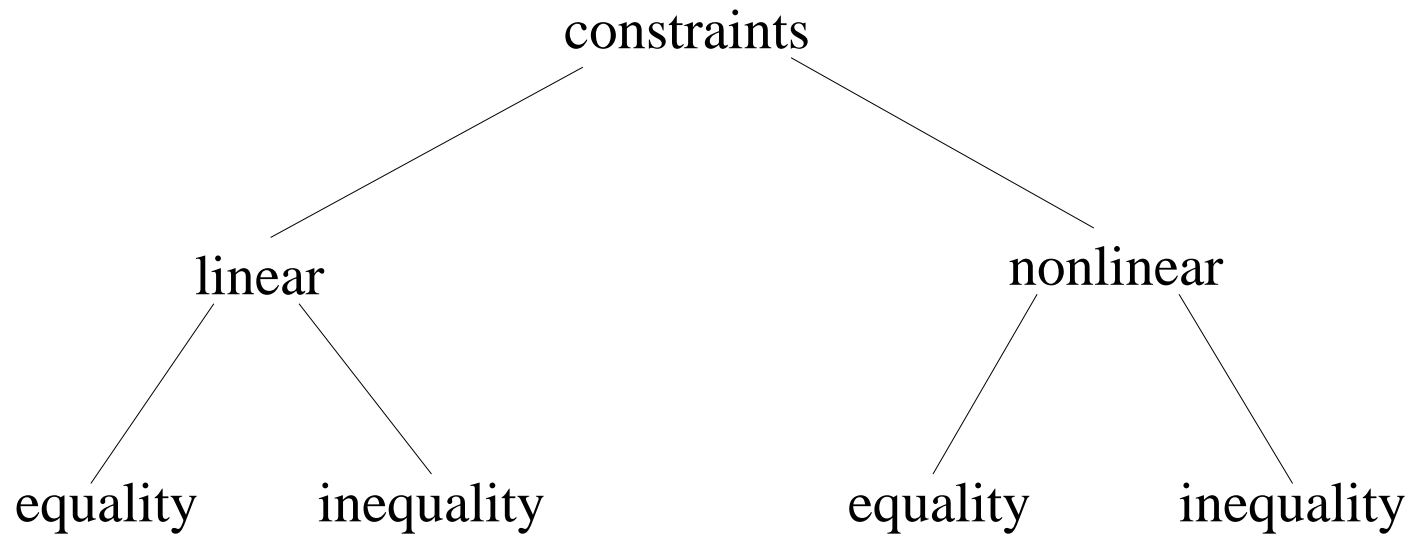
$$g_i(\mathbf{x}) \leq 0, \quad i = 1, 2, \dots, m$$

$$h_j(\mathbf{x}) = 0, \quad j = 1, 2, \dots, p$$

where \mathbf{x} is the n dimensional vector, $\mathbf{x} = (x_1, x_2, \dots, x_n)$; $f(\mathbf{x})$ is the objective function; $g_i(\mathbf{x})$ is the inequality constraint; and $h_j(\mathbf{x})$ is the equality constraint.

Denote the whole search space as \mathcal{S} and the feasible space as \mathcal{F} , $\mathcal{F} \subset \mathcal{S}$. It is important to note that the global in \mathcal{F} might not be the same as that in \mathcal{S} .

Different Types of Constraints



Linear constraints are relatively easy to deal with. Nonlinear equality constraints can be hard to handle.

Constraint Handling Techniques

The penalty function approach converts a constrained problem into an unconstrained one by introducing a penalty function into the objective function.

The repair approach maps (repairs) an infeasible solution into a feasible one.

The purist approach rejects all infeasible solutions in search.

The separatist approach considers the objective function and constraints separately.

The hybrid approach mixes two or more different constraint handling techniques.

Penalty Function Approach: Introduction

NewObjectiveFunction = OriginalObjectiveFunction +
PenaltyCoefficient * DegreeOfConstraintViolation

The general form of the exterior penalty function method:

$$\psi(\mathbf{x}) = f(\mathbf{x}) + \left(\sum_{i=1}^m r_i G_i(\mathbf{x}) + \sum_{j=1}^p c_j H_j(\mathbf{x}) \right),$$

where $\psi(\mathbf{x})$ is the new objective function to be minimised, $f(\mathbf{x})$ is the original objective function, r_i and c_j are penalty factors (coefficients), and

$$G_i(\mathbf{x}) = (\max(0, g_i(\mathbf{x})))^\beta, \quad H_j(\mathbf{x}) = \max(0, |h_j(\mathbf{x})|^\gamma),$$

β and γ are usually chosen as 1 or 2.

Penalty Function Approach: Techniques

Static Penalties The penalty function is pre-defined and fixed during evolution.

Dynamic Penalties The penalty function changes according to a pre-defined sequence, which often depends on the generation number.

Adaptive and Self-Adaptive Penalties The penalty function changes adaptively. There is no fixed sequence to follow.

Static Penalty Functions

$$\psi(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^m r_i (G_i(\mathbf{x}))^2$$

where r_i 's are pre-defined and fixed.

- Equality constraints can be converted into inequality ones:

$$h_j(\mathbf{x}) \implies |h_j(\mathbf{x})| - \varepsilon \leq 0$$

where $\varepsilon > 0$ is a small number.

- Simple and easy to implement.
- Requires rich domain knowledge to set r_i 's.
- r_i 's can be divided into a number of different levels. When to use which is determined by a set of heuristic rules.

Dynamic Penalty Functions: An Example

General principle: The larger the generation number, the larger the penalty coefficient. *Why?*

Joines and Houck's method:

$$\psi(\mathbf{x}) = f(\mathbf{x}) + (Ct)^\alpha \left(\sum_{i=1}^m |g_i(\mathbf{x})|^\beta + \sum_{j=1}^p |h_j(\mathbf{x})|^\beta \right),$$

where C , α and β are constants defined by the user, e.g., $C = 0.5$, $\alpha = 1, 2$, $\beta = 2$, t is the generation number.

- Many different forms of dynamic penalties are possible, e.g., $\frac{t}{T}$. The best one can be difficult to find.
- Different coefficients can be used for inequality and equality constraints.

Dynamic Penalties: Generalisation

$$\psi(\mathbf{x}) = f(\mathbf{x}) + r(t) \sum_{i=1}^m G_i^2(\mathbf{x}) + c(t) \sum_{j=1}^p h_j^2(\mathbf{x}),$$

where $r(t)$ and $c(t)$ are two penalty coefficients.

Polynomials a_i and b_i are user-defined parameters.

$$r(t) = a_0 + a_1 t + a_2 t^2 + \dots, \quad c(t) = b_0 + b_1 t + b_2 t^2 + \dots.$$

Exponentials a and b user-defined parameters.

$$r(t) = e^{at}, \quad c(t) = e^{bt}.$$

Hybrid

$$r(t) = e^{a_0 + a_1 t + a_2 t^2 + \dots}, \quad c(t) = e^{b_0 + b_1 t + b_2 t^2 + \dots}.$$

Adaptive Penalties: An Example

Bean and Hadj-Alouane's method:

$$\psi(\mathbf{x}) = f(\mathbf{x}) + \lambda(t) \left(\sum_{i=1}^m G_i^2(\mathbf{x}) + \sum_{j=1}^p |h_j(\mathbf{x})| \right),$$

where λ is updated at generation t as follows:

$$\lambda(t) = \begin{cases} \frac{1}{\beta_1} \lambda(t), & \text{if case 1,} \\ \beta_2 \lambda(t), & \text{if case 2,} \\ \lambda(t), & \text{otherwise,} \end{cases}$$

where case 1 (or 2) indicates that the best individual in the last k generations was always feasible (or infeasible), $\beta_2 > \beta_1 > 1$.

What does the technique mean?

Fitness Function and Selection

- Let $\Phi(\mathbf{x}) = f(\mathbf{x}) + rG(\mathbf{x})$, where $G(\mathbf{x}) = \sum_{i=1}^m G_i(\mathbf{x})$ and $G_i(\mathbf{x}) = \max\{0, g_i(\mathbf{x})\}$.
- Given two individuals \mathbf{x}_1 and \mathbf{x}_2 . Their fitness values will now be determined by $\Phi(\mathbf{x})$.
- Because fitness values are used primarily in selection,

Changing fitness \iff changing selection probabilities.

When r Plays an Important Role

$$\Phi(\mathbf{x}_1) < \Phi(\mathbf{x}_2)$$

means

$$f(\mathbf{x}_1) + rG(\mathbf{x}_1) < f(\mathbf{x}_2) + rG(\mathbf{x}_2).$$

1. $f(\mathbf{x}_1) < f(\mathbf{x}_2)$ and $G(\mathbf{x}_1) < G(\mathbf{x}_2)$: r has no impact on the comparison.
2. $f(\mathbf{x}_1) < f(\mathbf{x}_2)$ and $G(\mathbf{x}_1) > G(\mathbf{x}_2)$: Increasing r will eventually change the comparison.
3. $f(\mathbf{x}_1) > f(\mathbf{x}_2)$ and $G(\mathbf{x}_1) < G(\mathbf{x}_2)$: Decreasing r will eventually change the comparison.

In essence, different r 's lead to different rankings of individuals in the population.

Penalty Functions Demystified

Penalty function

⇒ Fitness transformation

⇒ Rank change (selection change)

Why not change the rank directly in an EA?

Stochastic Ranking — Self-adaptive

Stochastic ranking is a special rank-based *selection scheme* that handles constraints. There is no need to use any penalty functions. It's self-adaptive since there are few parameters to set.

1. DO the following until there is no more change in the ranking.
2. FOR $i := 1$ TO $\mu - 1$ DO
 - (a) $u := U(0, 1)$, where u is a uniformly distributed random number;
 - (b) IF $G(\mathbf{x}_i) = G(\mathbf{x}_{i+1}) = 0$ OR $u \leq P_f$ THEN
IF $f(\mathbf{x}_i) > f(\mathbf{x}_{i+1})$ THEN $swap(\mathbf{x}_i, \mathbf{x}_{i+1})$;
 - (c) ELSE
IF $G(\mathbf{x}_i) > G(\mathbf{x}_{i+1})$ THEN $swap(\mathbf{x}_i, \mathbf{x}_{i+1})$;

P_f indicates the probability of using the objective function for comparison in ranking.

The Role of P_f

$P_f > 0.5$: Most comparisons are based on $f(\mathbf{x})$ only. Infeasible solutions are likely to occur.

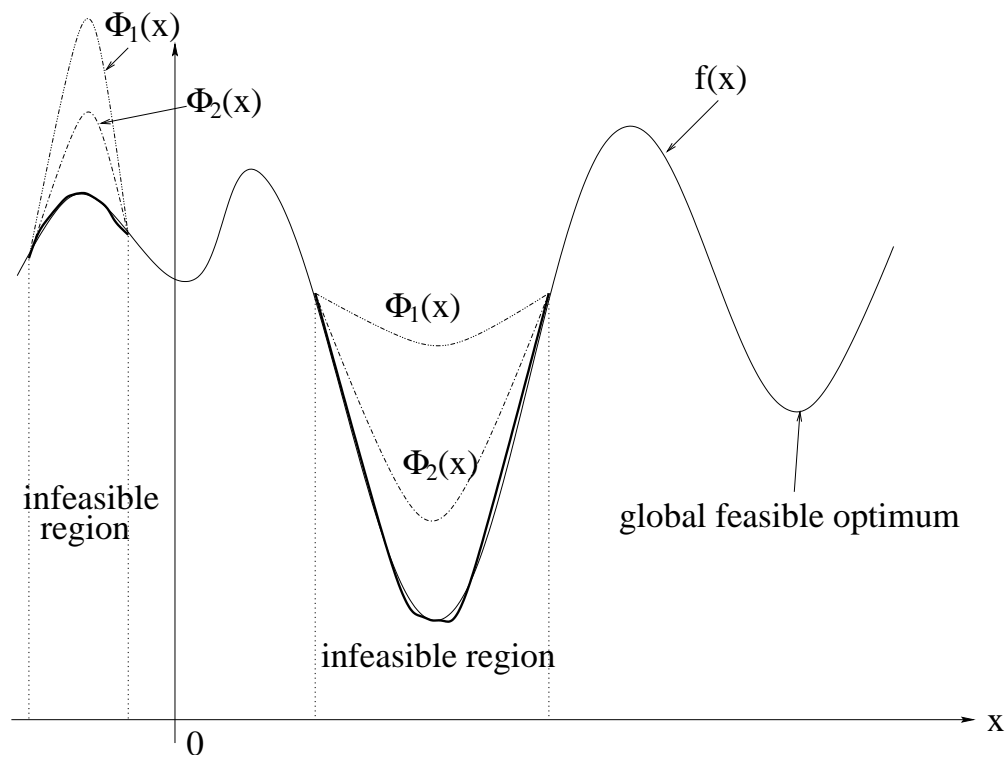
$P_f < 0.5$: Most comparisons are based on $G(\mathbf{x})$ only. Infeasible solutions are less likely to occur, but the solutions might be poor.

In practice, P_f is often set between 0.45 and 0.5. *Is feasibility guaranteed in this case?*

Penalties and Fitness Landscape Transformation

NewObjectiveFunction = OriginalObjectiveFunction +
PenaltyCoefficient * DegreeOfConstraintViolation

Different penalty functions lead to different new objective functions.



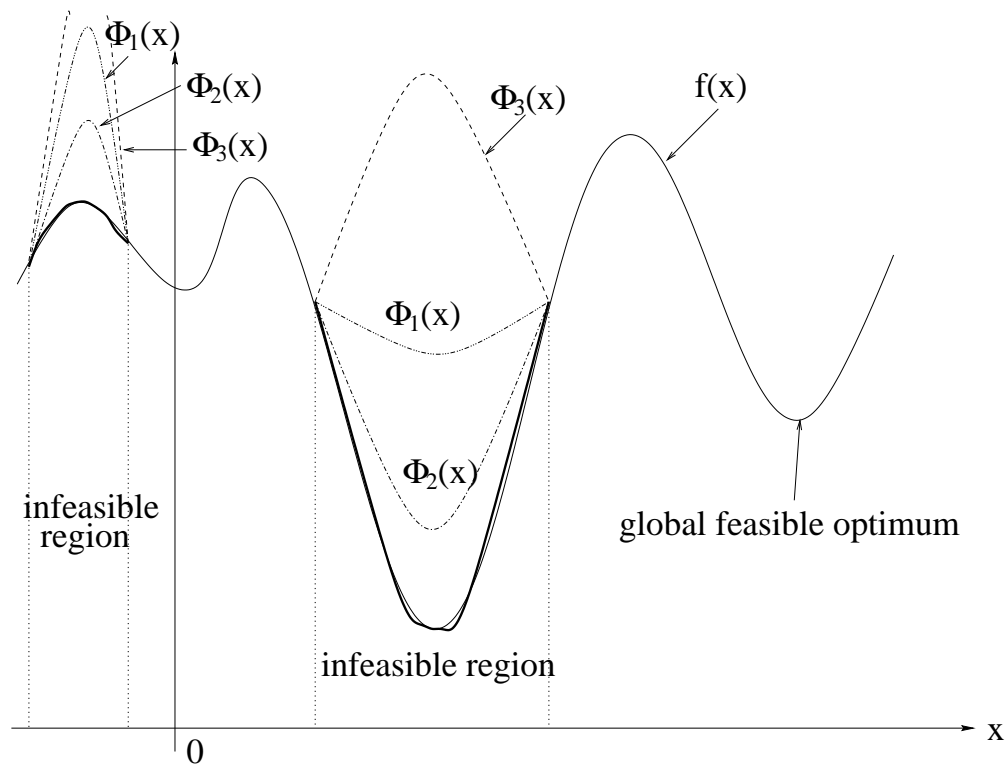
Well Transformed Objective Functions

1. $\Phi_1(\mathbf{x})$ on the previous page is well transformed because the global optimum of $\Phi_1(\mathbf{x})$ is the feasible optimum we want.
2. $\Phi_2(\mathbf{x})$ is poorly transformed because its global optimum is infeasible. This is a typical case of *under-penalisation*.

Does it mean the penalty function should be really large?

What Would be an Appropriate Penalty

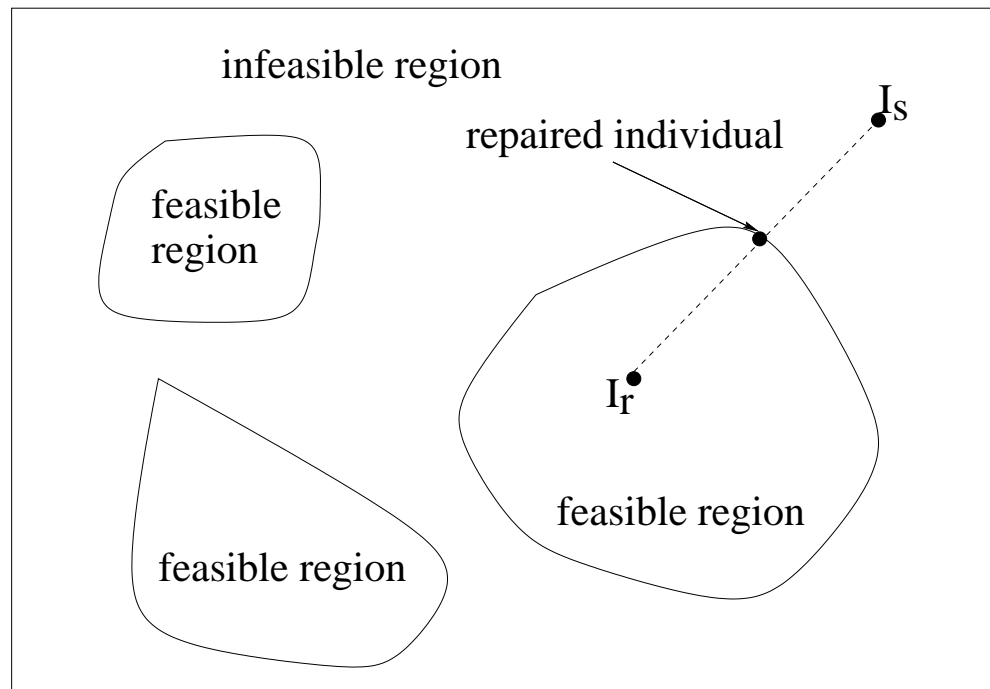
Is $\Phi_2(x)$ a good choice? Why or why not?



The balance between objective function and penalties is important.

Repair Approach to Constraint Handling

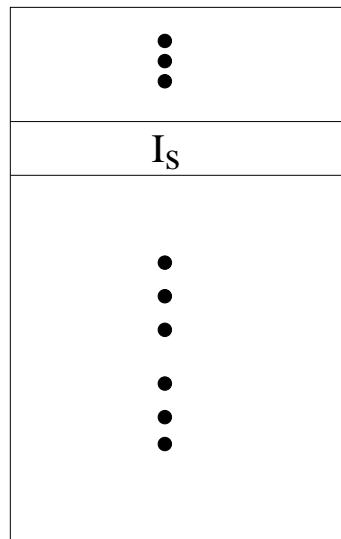
Instead of modifying an EA or fitness function, infeasible individuals can be “repaired” into feasible ones.



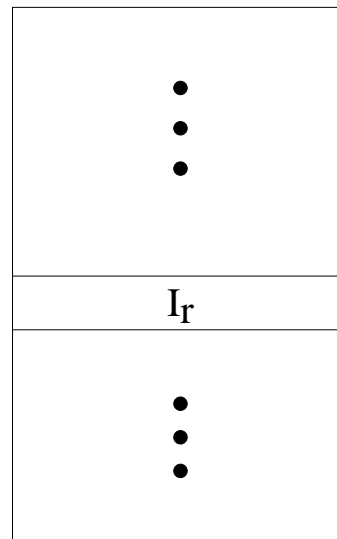
Repairing Infeasible Individuals

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

population of evolving
individuals (feasible or
infeasible)



population of feasible
reference individuals
(changing but not evolving)



Repairing Algorithm

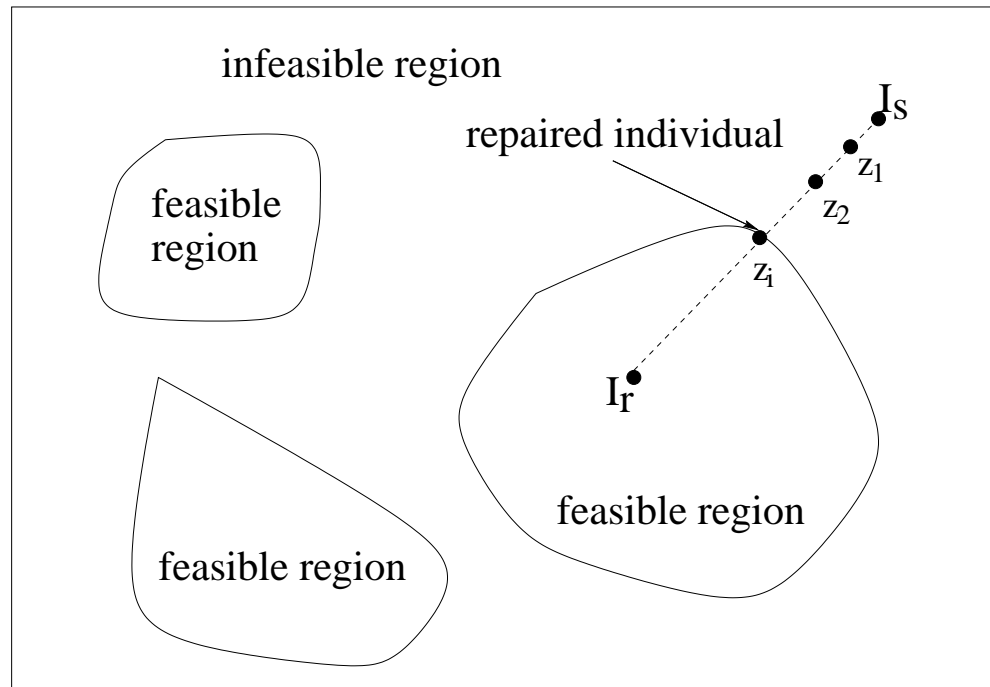
1. Select a reference individual I_r .
2. Create a sequence of candidate individuals z_i between I_s and I_r :

$$z_i = a_i I_s + (1 - a_i) I_r,$$

where $0 < a_i < 1$ can be generated at random or deterministically. The process of creating z_i stops when z_i is feasible (i.e. ,when the first feasible z_i is found).

3. Let this z_i be the repaired individual of I_s . Its objective function value will be used as I_s 's fitness value.
4. Replace I_s by z_i with probability P_r . (Even if I_s is not replaced by z_i , its fitness is still that of z_i 's.)
5. If z_i is better than I_r , replace it.

Repairing Algorithm: Visualisation



Repairing Algorithm: Implementation Issues

How to find initial reference individuals? 1. Preliminary exploration

2. Human knowledge

How to select I_r ? 1. Uniformly at random

2. According to the fitness of I_r

3. According to the distance between I_r and I_s

How to determine a_i ? 1. Uniformly at random between 0 and 1

2. Using a fixed sequence, e.g., $\frac{1}{2}, \frac{1}{4}, \dots$

How to choose P_r ? A small number, usually < 0.5 .

Repairing, Lamarckian Evolution and Baldwin Effect

1. The repairing approach is closely linked to the wider issues regarding Lamarckian evolution and Baldwin effect. This is due to two characteristics:
 - (a) Repairing occurs only within one generation.
 - (b) Repaired individuals does not always replace the original ones.
2. Individual repairing \iff individual learning.
3. Although learned characteristics (i.e., repairs done) are not inherited, (individual) learning helps and guide (population) evolution.
4. Learning appears to smooth out bumps in the fitness landscape and thus help evolution.

Summary

1. Adding a penalty term to the objective function is equivalent to changing the fitness function, which is in turn equivalent to changing selection probabilities.
2. It is easier and more effective to change the selection probabilities directly and explicitly. Stochastic ranking enables us to do this.
3. There are other constraint handling techniques than the penalty method.
4. We have covered numerical constraints only. We have not dealt with constraints in a combinatorial space.

References

1. T. Bäck, D. B. Fogel, and Z. Michalewicz (eds.), Handbook of Evolutionary Computation, IOP Publ. Co. & Oxford University Press, 1997. Sections C5.1 — C5.6. (In the school library)
2. T. P. Runarsson and X. Yao, “Stochastic Ranking for Constrained Evolutionary Optimization,” *IEEE Transactions on Evolutionary Computation*, **4**(3):284-294, September 2000.
3. T. Runarsson and X. Yao, “Search Bias in Constrained Evolutionary Optimization,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, **35**(2):233-243, May 2005.