MY FIRST GATE TO GENETIC ALGORITHM

Farah Fairuz Zahirah
Nagoya University

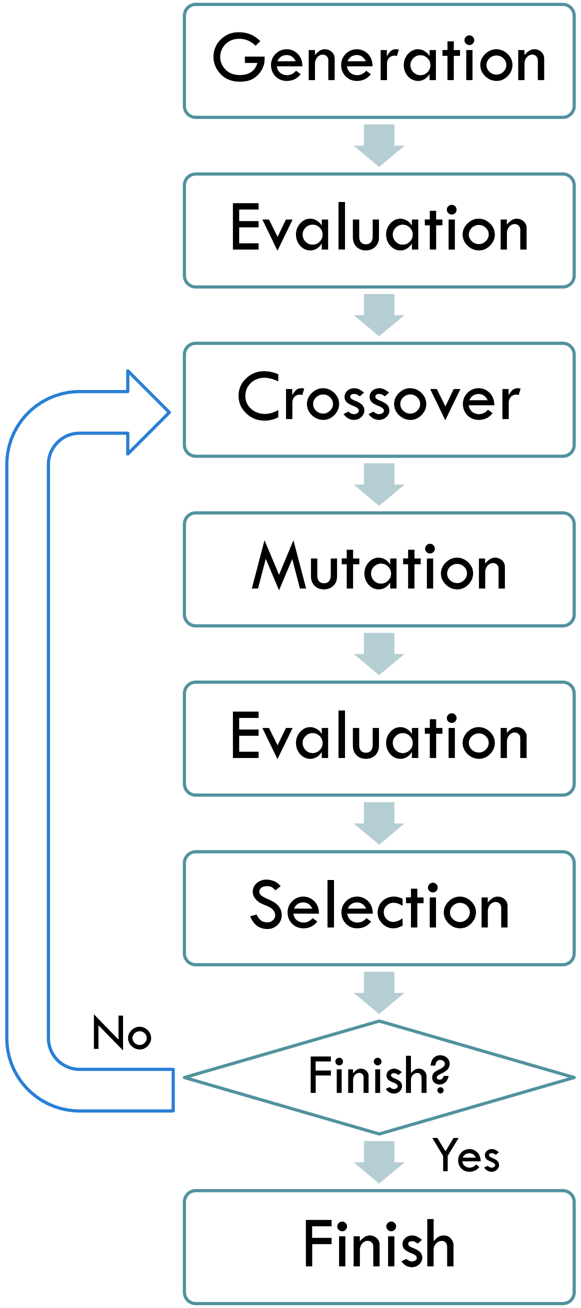
PURPOSE

Basic GA, as the first learning experience
of actually building the program based on a real problem, also of processing the data

FLOW

Setting

Design Variable	32
Constraint	22
Population	50
Number of Evaluations	9950



FIRST GENERATION

Randomized generation for the seed, between [0, 1]

as a standardized value of each variables

x_1	x_2	x_3		x_{31}	x_{32}
0.114849	0.808894	0.677372	…	0.28848	0.056884

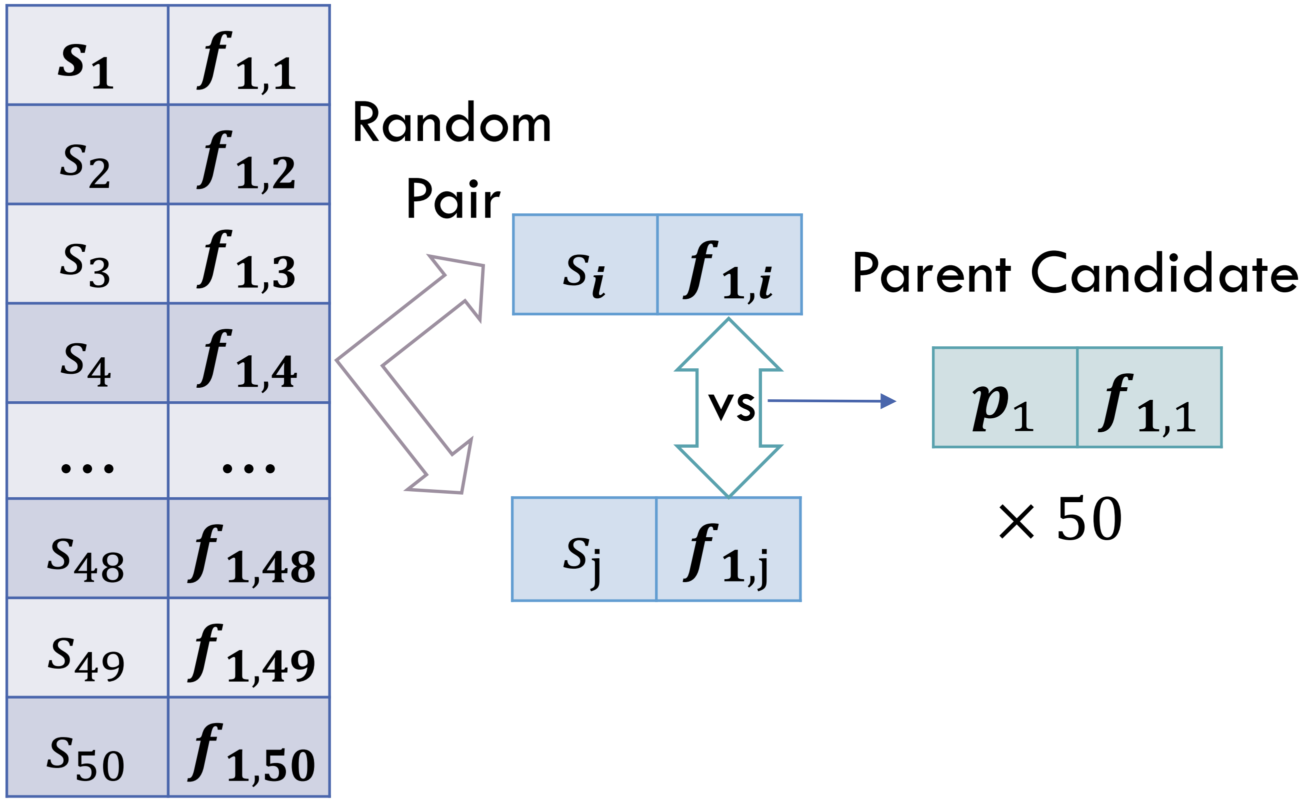
PARENT SELECTION

Tournament Method

Randomly selected 2 individual

-> Individual with lower objective value are chosen

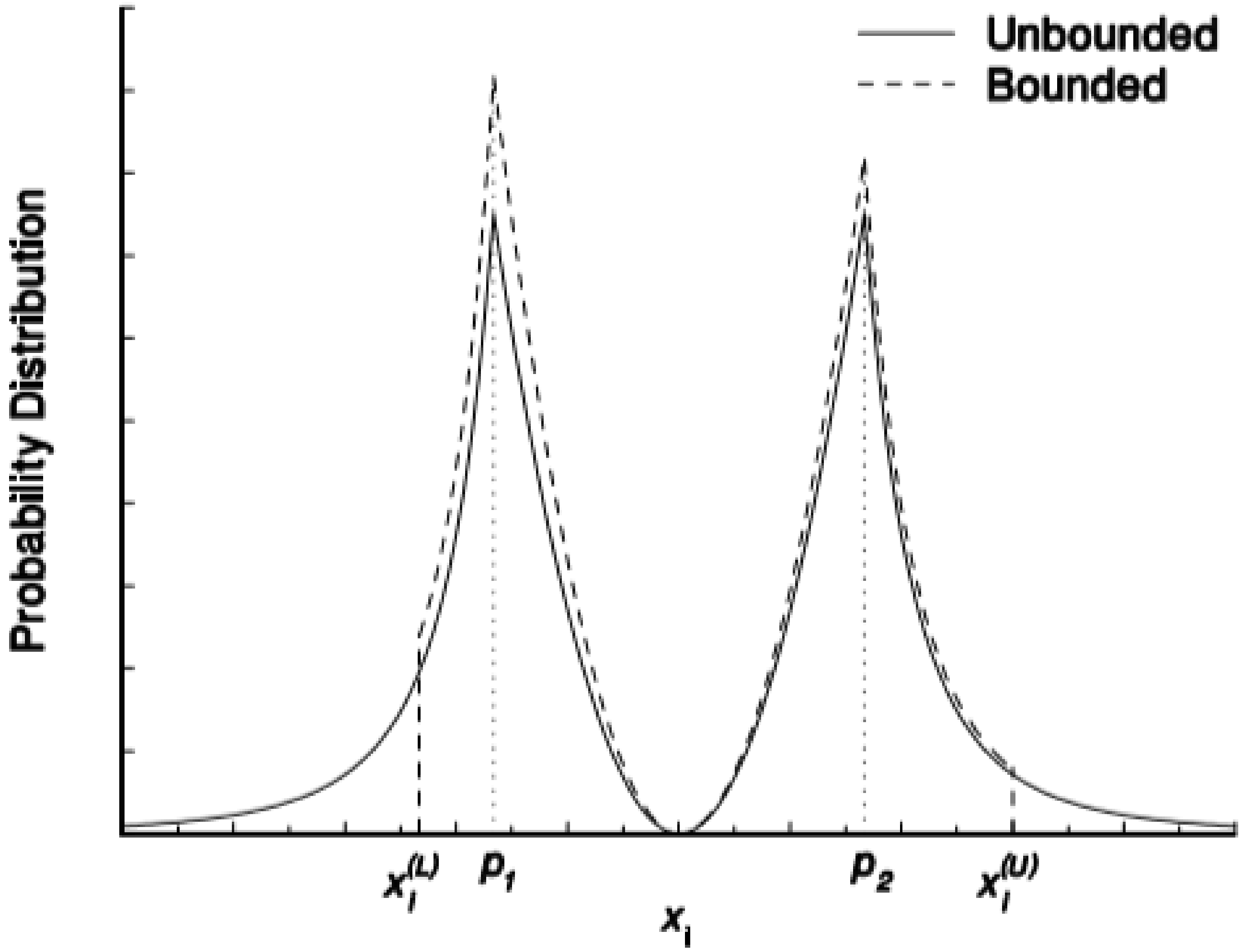
-> repeated enough times to get a full population



CROSSOVER

Simulated Binary Crossover – SBX (Deb and Goyal, 1996)

- Symmetric -> Avoid any bias towards particular parent
- When parents values are distant, distant children values possible
  When parents values are close, distant children values unlikely
  → Converging Search



CROSSOVER

Following the equation of:

$$\bar{\beta} = \begin{cases} (2u)^{\frac{1}{n+1}}, & if\ u \leq 0.5 \\ \left(\frac{1}{2(1-u)}\right)^{\frac{1}{n+1}}, & otherwise. \end{cases}$$

Children → Parents

$$x_i^{(1,t+1)} = 0.5\left[\left(1+\bar{\beta}\right)x_i^{(1,t)} + \left(1-\bar{\beta}\right)x_i^{(2,t)}\right],$$

$$x_i^{(2,t+1)} = 0.5\left[\left(1-\bar{\beta}\right)x_i^{(1,t)} + \left(1+\bar{\beta}\right)x_i^{(2,t)}\right],$$

Parameter

$$n = 15$$

Steps:

- Randomize u
- Get $\bar{\beta}$
- Calculate children value

MUTATION

The Mutated Value is calculated following the probability function defined by Deb and Goyal (1996) that depends on the perturbance factor δ:

$$P(\delta) = 0.5(n+1)(1-|\delta|)^n$$

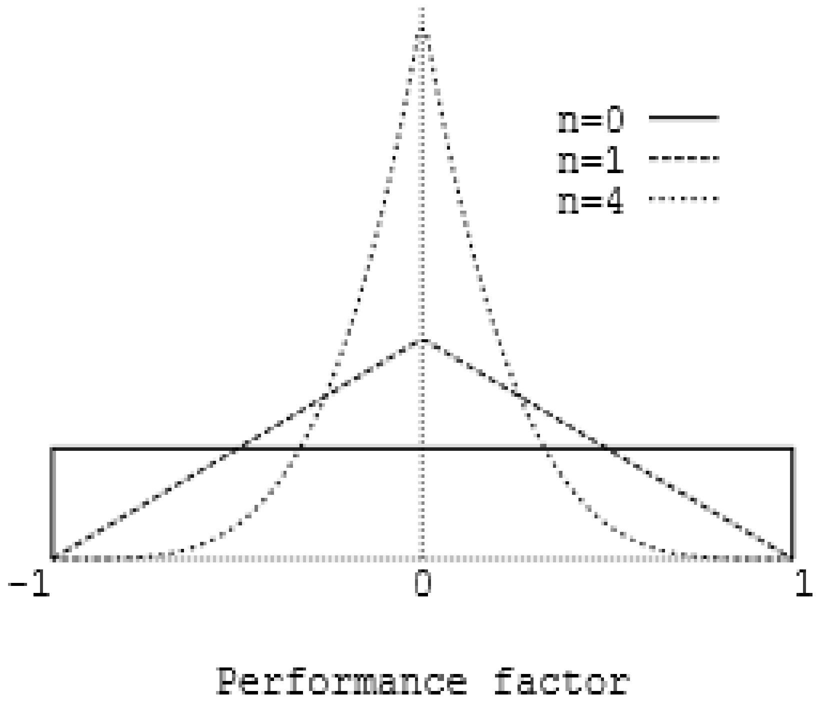
In accordance to the following equation,

$$\bar{\delta} = \begin{cases} (2u)^{\frac{1}{n+1}} - 1, & if\ u < 0.5 \\ 1 - [2(1-u)]^{\frac{1}{n+1}}, & if\ u \geq 0.5. \end{cases}$$

$$c = p + \bar{\delta}\Delta_{max}$$

Parameters:

$$p_m = \frac{1}{32},\ \Delta_{max} = 1,\ \ n = 15$$



Steps:

- ➢Randomize u
- ➢Get $\bar{\delta}$
- ➢Calculate Mutated Value c

CONSTRAINT HANDLING

Penalty points to objective value
-> Harder to be selected

$$f' = f + \alpha \times \Omega(x)$$

$$\Omega(x) = \sum_{i=1}^{22} w_i ,$$

$$w_i = \begin{cases} 0, & g_i(x) > 0 \\ |g_i(x)|, & g_i(x) \leq 0 \end{cases}$$

Parameter: $\alpha = 1$

OBTAINED RESULT

From the solution that satisfied all the constraint conditions,

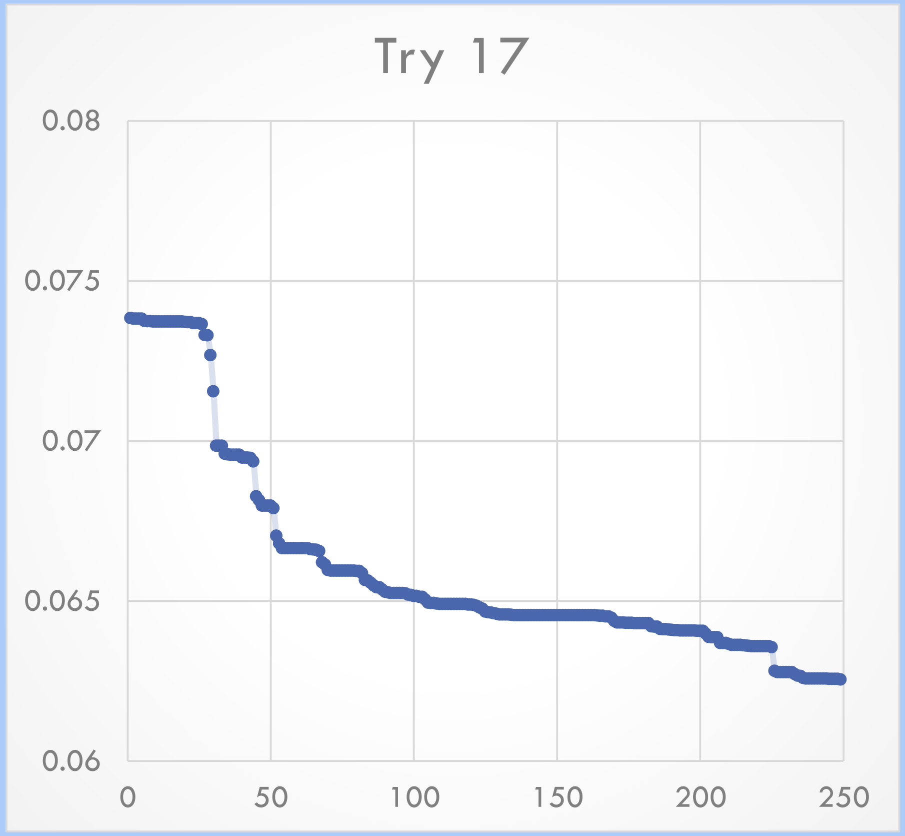
The best solution is of $f = 0.062543495$

Found on Generation 99

$$^{11}/_{21}$$

The median solution is of

- Only from satisfying trials: $f = 0.069217208$
- From all 21 Trials: $f = 0.078854218$



Satisfied Individual's f in the Best Trial

CONCLUSION

This project helped me build my foundation:

- GA and Techincal Programming Skills

Further study:

- Try making adjustment and research the effects
  - Ideas: Change the n from small to big along the loop so that the search can be even more converging
- Try the multi-objective techniques

Infeasibility Driven Evolutionary Algorithm with Bump Hunting

Third Evolutionary Computation Competition 2019

Category: Single Objective

Group Number: S02

UNSW AUSTRALIA

Never Stand Still

Kamrul Hasan Rahi

Research Masters Student

Supervisors: Dr. Hemant Kumar Singh and Professor Tapabrata Ray

Multidisciplinary Design Optimization (MDO) Group

School of Engineering and Information Technology (SEIT)

The University of New South Wales, Australia.

IDEA with Bump Hunting

Constrained single objective wind turbine design optimization problem.

- Population based stochastic optimization algorithms are preferred since the objective and constraint functions may be highly nonlinear with functional/slope discontinuity.
- To deal with constraints, strategies often prefer a feasible solution over infeasible ones. They are referred as Feasibility First constraint handling strategies e.g. NSGA-II.

However, preserving marginally infeasible solutions during the course of search and actively recombining them can result in faster rate of convergence over feasibility first strategies. **Infeasibility Driven Evolutionary Algorithm (IDEA)[1]** is one such scheme known for its superior performance on constrained optimization problems.

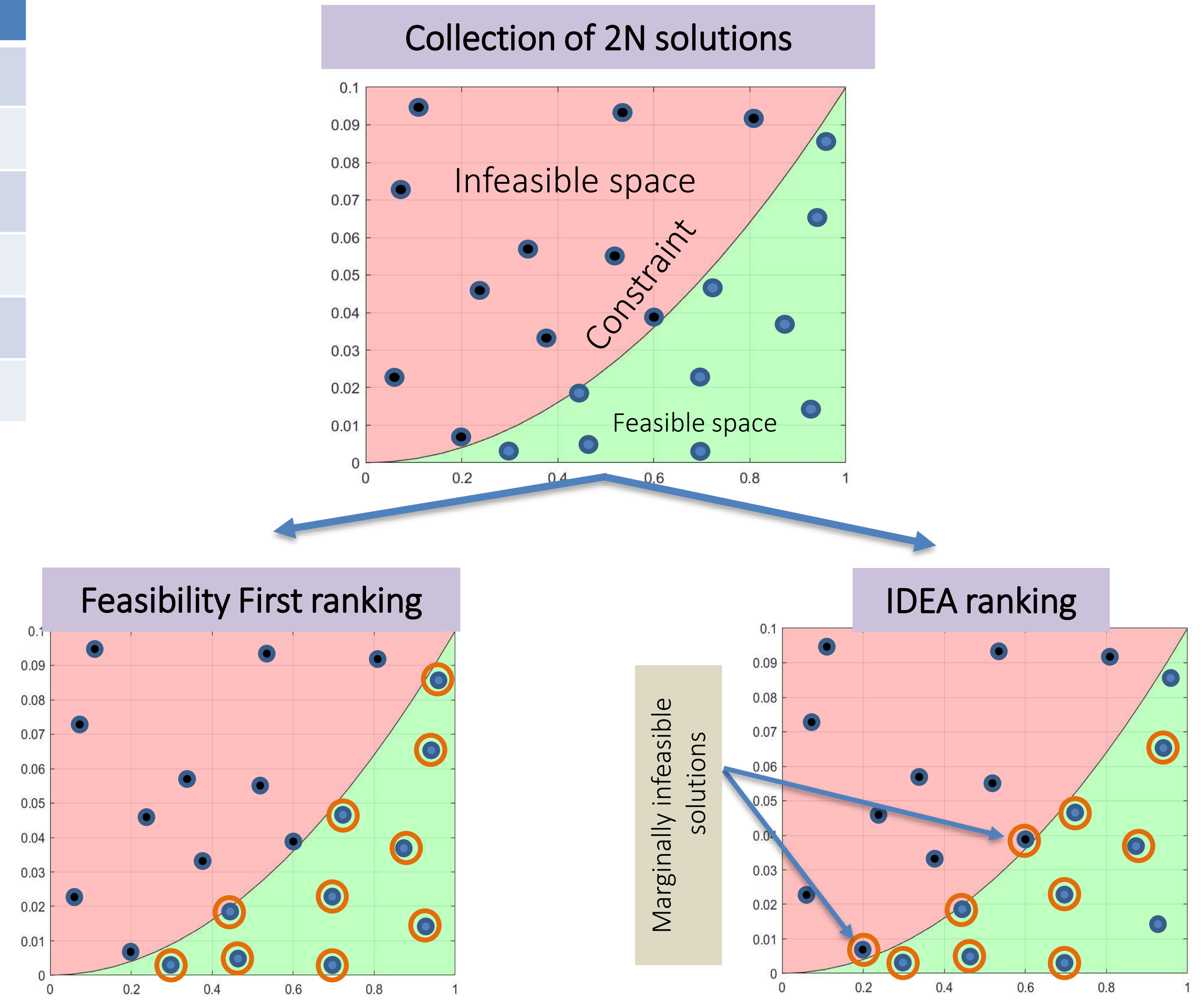
Smart reduction in variable space is yet another scheme that can offer significant benefits to the process of recombination. **Bump Hunting[2]** is an approach that can be used to identify potential regions of interest.

The proposed approach employs IDEA with original variable bounds until 50% of the computational budget is exhausted. Thereafter, it identifies reduced variable bounds using Bump Hunting and runs IDEA using these reduced bounds for the remaining computational budget.

1. Ray, T., Singh, H. , Isaacs, A. , and Smith, W.,(2009) "Infeasibility driven evolutionary algorithm for constrained optimization," in Constraint Handling in Evolutionary Optimization (Mezura-Montes, E. ed.), Studies in Computational Intelligence, vol. 198, pp. 147–167, Springer.
2. Friedman, J. H., & Fisher, N. I. (1999). Bump hunting in high-dimensional data. Statistics and Computing, 9(2), 123-143.

Infeasibility Driven Evolutionary Algorithm (IDEA)

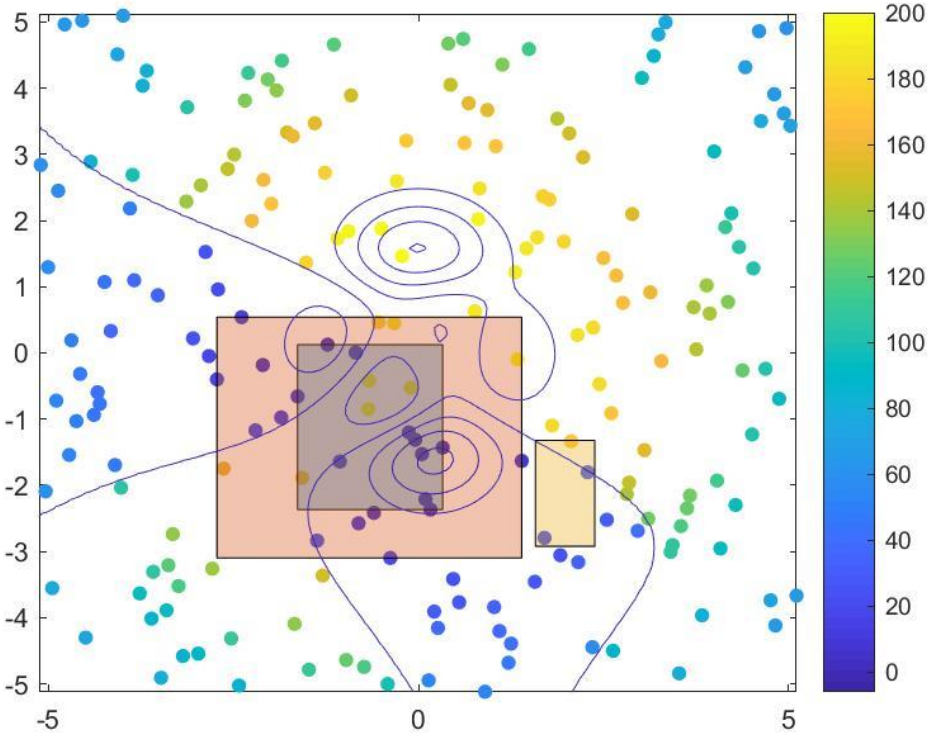
Parameter	Value
Population Size *N*	100
Crossover probability	1.0
Mutation probability	0.1
Distribution index: Crossover	20
Distribution index: Mutation	20
Infeasibility ration (α)	0.1

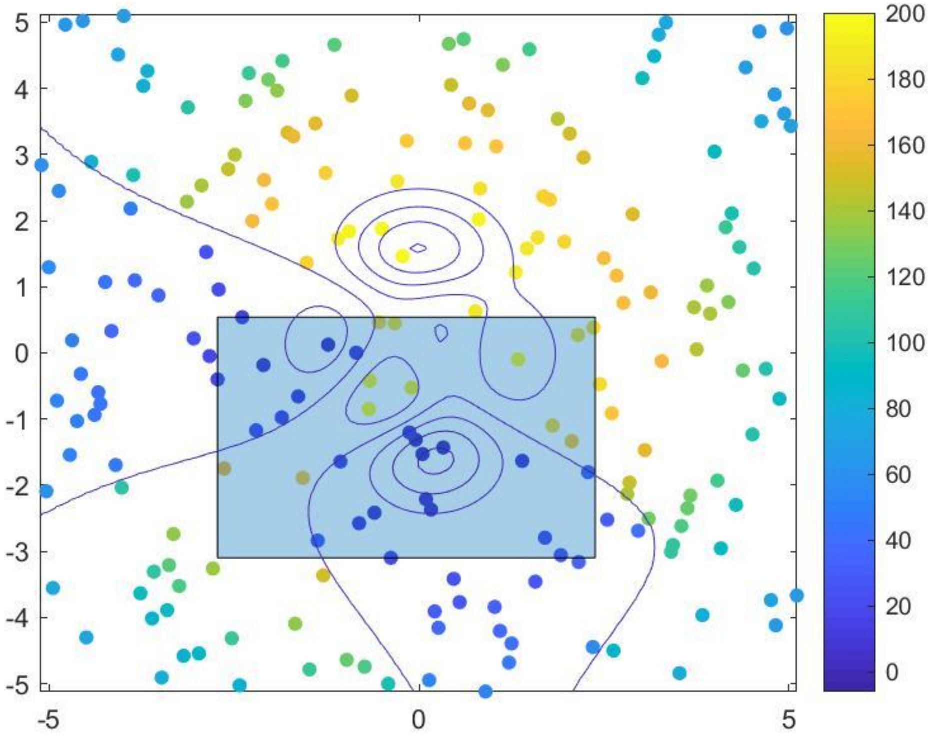


Steps:

1. Generate N Initial solutions using LHS sampling.
2. Evaluate these N solutions.
3. Create N offspring using SBX and PM.
4. Evaluate these N offspring solutions.
5. Select N solutions from these N parent and N offspring solutions using IDEA ranking.

Bump Hunting (BH) for Space Reduction: Illustration





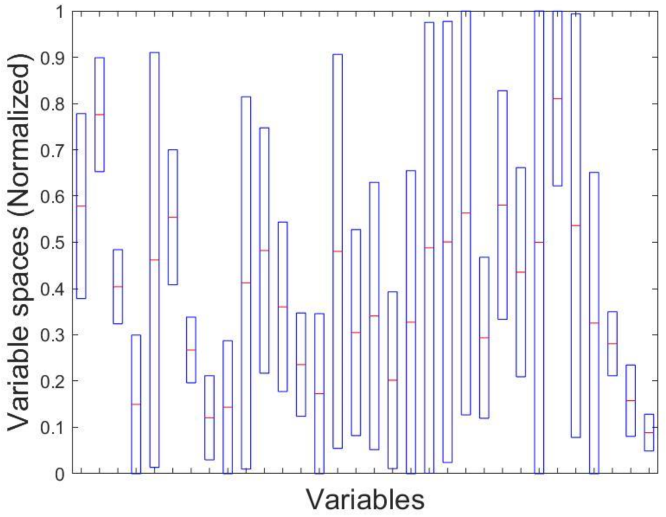
Promising hyper-rectangles identified using best 50% solutions(Minimization sense).

Lower bound = max(min(all variables from all boxes), global LB)

Upper bound = min(max(all variables from all boxes), global UB)

IDEA with BH for Wind Turbine Design Problem

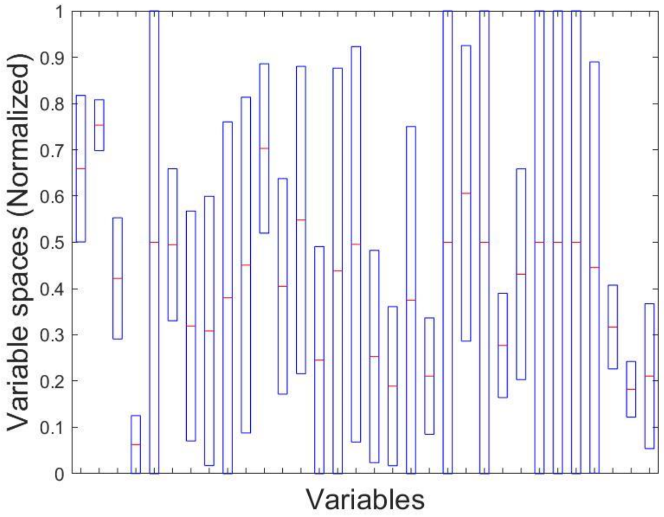
Reduced variable bounds identified from best solutions.



Potential regions of interest. Bounds identified using results of 21 runs of NSGA-II, IDEA for 10,000 and 30,000 functions evaluations.

Volume=1.4429E-13

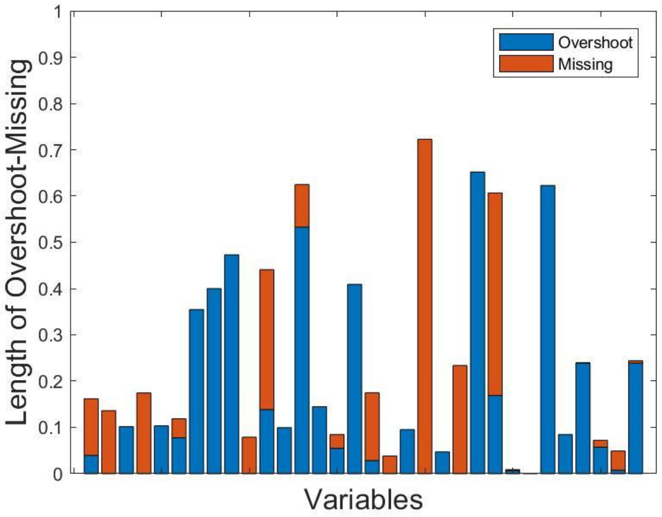
Reduced variable bounds identified after 5,000 function evaluations



Top 50% of solutions (with constraint violation less than 1e-3 or feasible solutions) were used to identify the reduced variable bounds.

Volume=2.9170E-11

Overshoots and Missed spaces



Low height of red and blue bar is preferred.

MULTIDISCIPLINARY DESIGN OPTIMIZATION

Thank you for your attention

http://www.mdolab.net/

UNSW
THE UNIVERSITY OF NEW SOUTH WALES

s03

Applying Differential Evolution to Wind Turbine Optimization in Considering Constraint Violations

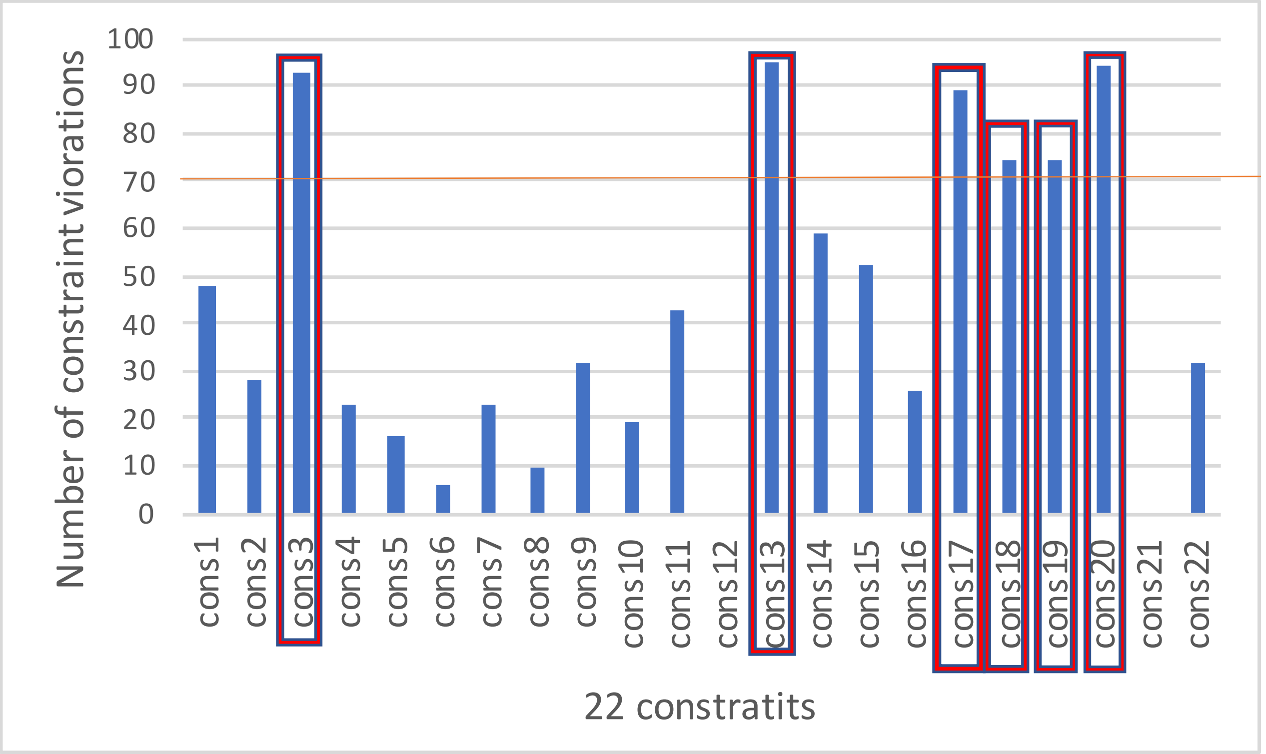
Hori Takato　Uchitane Takeshi　(Aichi Institute of Technology)

Evolutionary Computing Symposium 2019 @ Minamiawaji sity

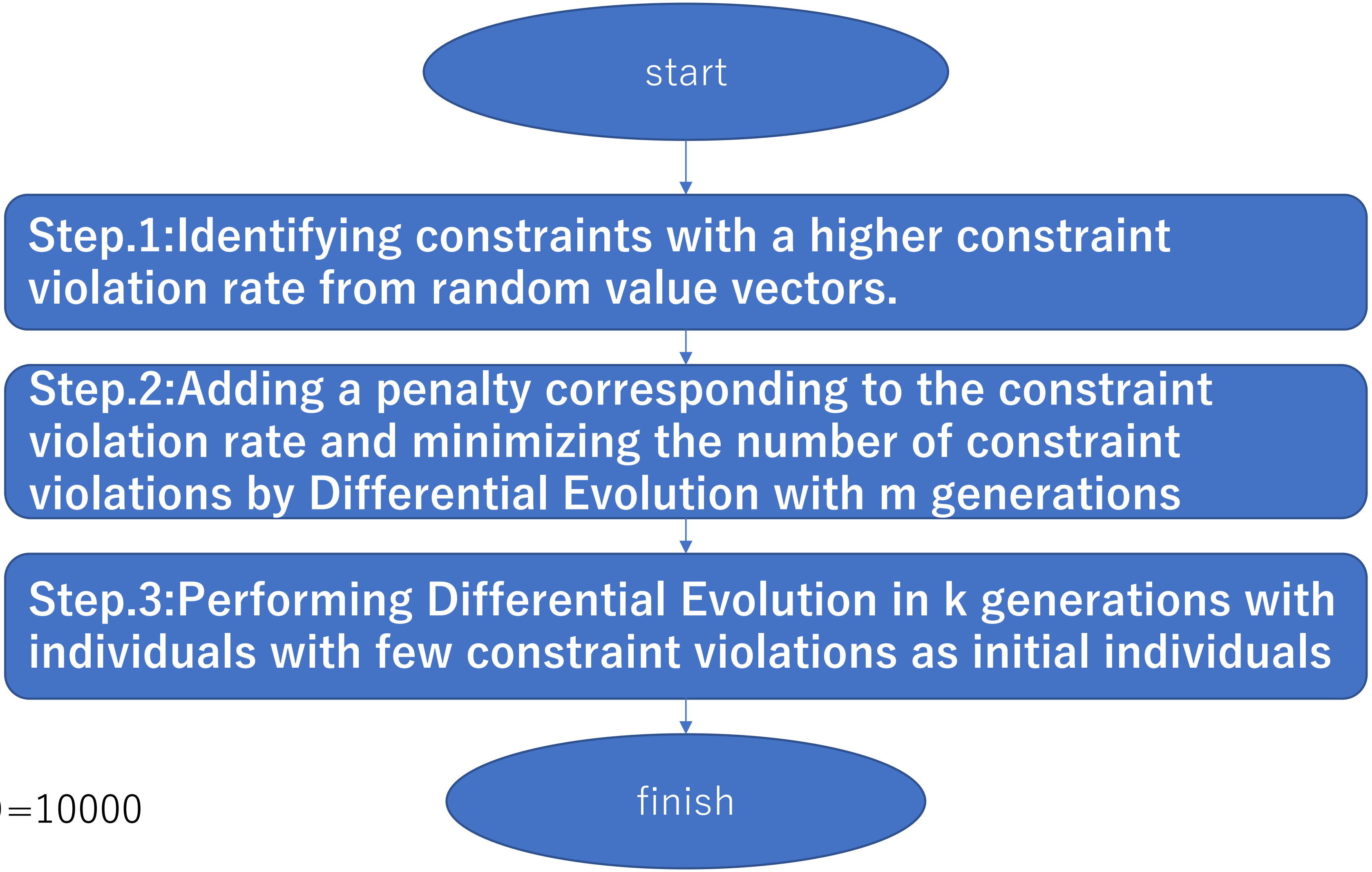
Wind turbine optimization constraints

There are many constraints on the problem of Wind turbine optimization.

The constraint violation rate is depending on the constraint conditions. Particular, 3, 13, 17, 18, 19, and 20th constraints have many constraint violations.



Three steps to applying Differential Evolution to eliminate constraint violations



i. If n and m are too large, fewer generations to minimize objective function.
ii. If n and m are too small, the objective function is optimized with constraint violations remaining.

num of individual × (n+m+k)=10000

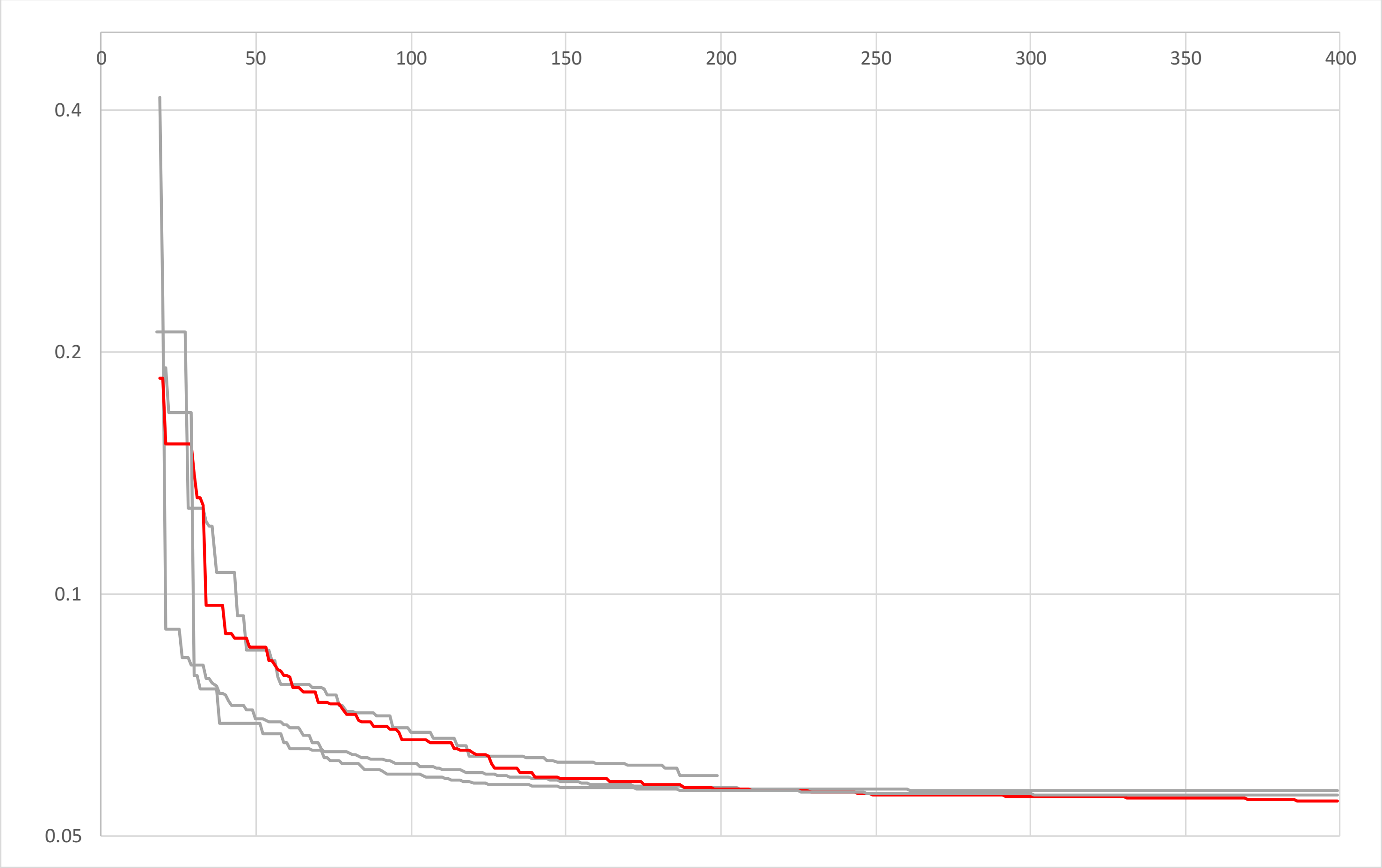
Result

parameter

F	CR	num of individuals	n	m	k
0.5	0.2	50	5	25	170
0.5	0.2	25	10	20	370
0.3	0.2	25	10	20	370
0.3	0.5	25	10	20	370

Best ：0.055301
Var. ：1.9632E-07
Ave. ：0.056092571
Med.：0.05605

Transition of evaluation function



Mutation based on Variance of Individuals in IDE

P4-01　Ryukoku University

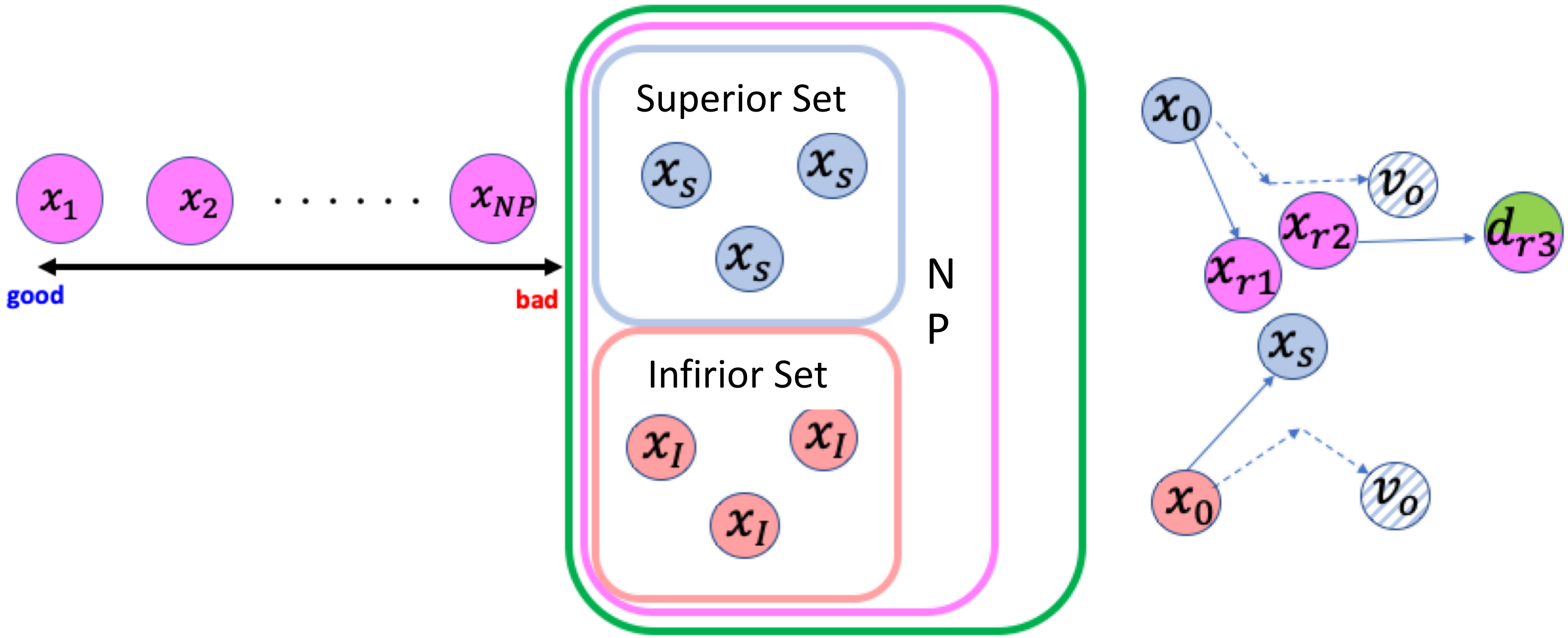
○Yuta Furukawa　Keiko Ono
Kenta Matsuo

Overview

IDE

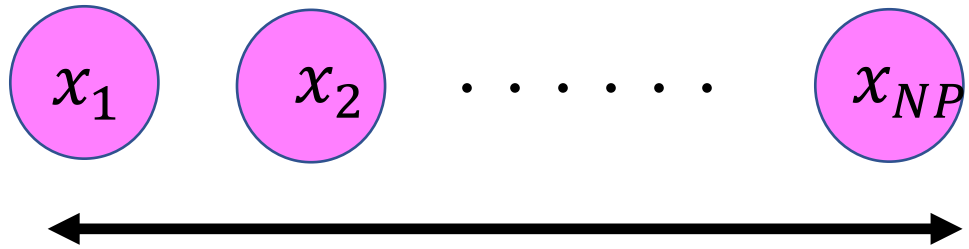
Differential EvolutionWith an Individual-Dependent Mechanism

- ◆ A kind of differential evolution method proposed by Lixin Tang et al.
- ◆ Efficient search is possible by IDP setting to set parameters based on individuals' fitness and IDM strategy to set an appreciate search direction and its range.



Mutation:IDP Setting

◆Sort population based on fitness



good | bad

F -> $F_o = (randn\left(\frac{o}{NP}\right), 0.1)$ $(o = 1,2,\dots,NP)$

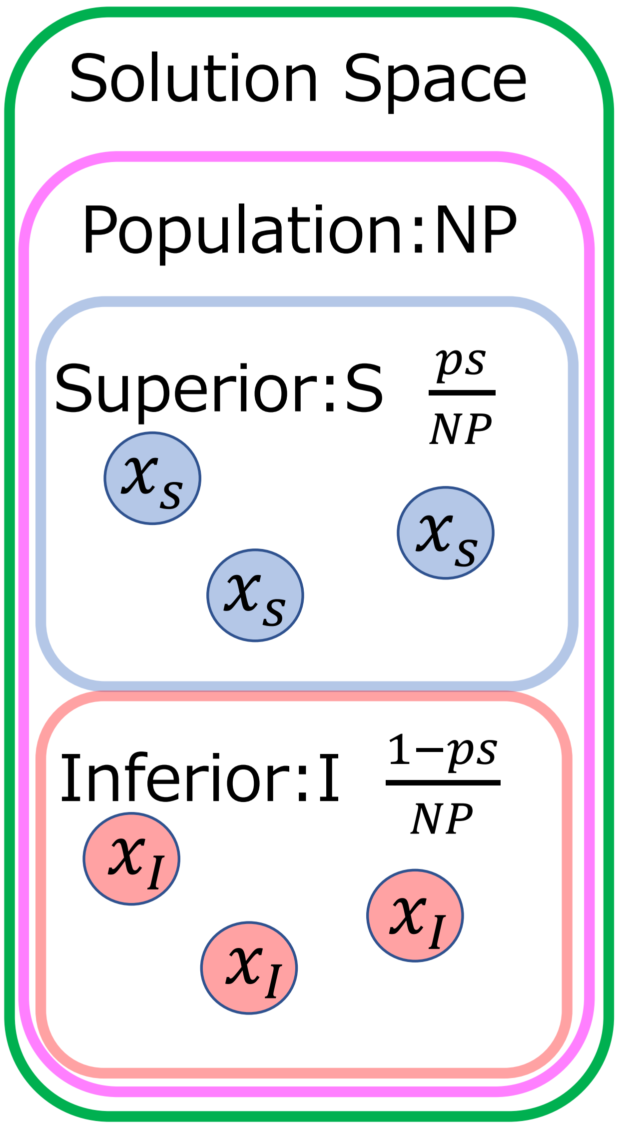
◆ Individuals with good fitness are set to smaller F and each search range is reduced.
◆Individuals with bad fitness are set to larger F and its search range is expanded.

CR-> $CR_i = (randn\left(\frac{i}{NP}\right), 0.1)$ $(i = 1,2,\dots,NP)$

◆Individuals with good fitness are set to smaller CR to inherit more information from parents.
◆Individuals with bad fitness have a larger CR to inherit more information from mutant individuals.

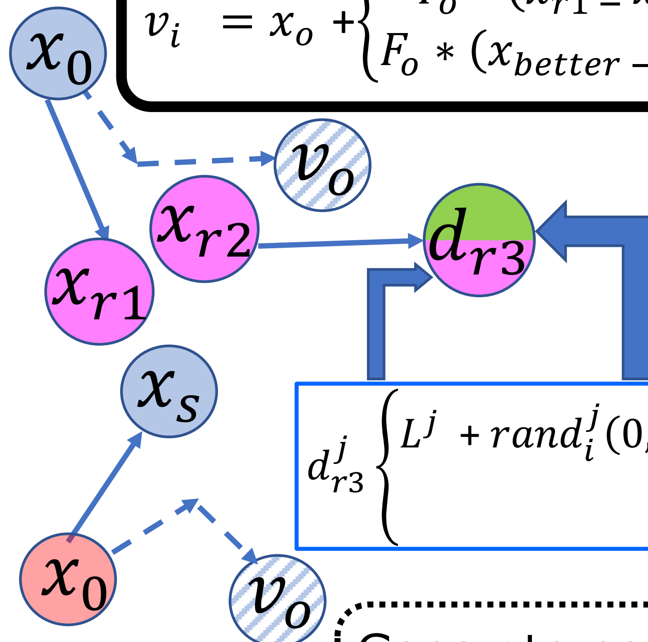
龍谷大学
RYUKOKU UNIVERSITY

Mutation:IDM Strategy



$$v_i = x_o + \begin{cases} F_o * (x_{r1} - x_o) + F_o * (x_{r2} - d_{r3}) & o \in S \\ F_o * (x_{better} - x_o) + F_o * (x_{r2} - d_{r3}) & o \in I \end{cases}$$

S:Superior set
I:Inferior set
x_{better}:Individuals randomly selected from the superior set



$$d^j_{r3} \begin{cases} L^j + rand^j_i(0,1) \cdot (U^j - L^j), \text{if}(rand^j_i(0,1) < P_d) \\ x^j_{r3}, otherwise \end{cases}$$

Generate new individuals randomly or selected from a current population

ps:Determine the composition of dr3 and the proportion of individuals in the superior and inferior sets

龍谷大学
RYUKOKU UNIVERSITY

Proposed Method

Point

◆ Dimensional compression with SOM

- ◆ When the target problem is high-dimensional problem

- ◆ Introduce the Self-Organizing Map(SOM)

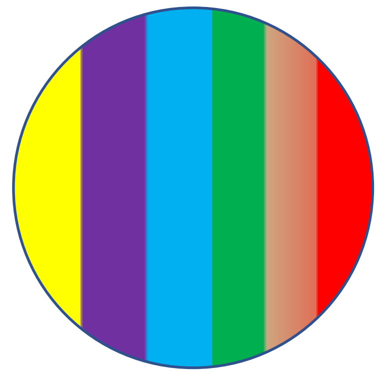
$$NP = \{\boldsymbol{x}_{1,}, \boldsymbol{x}_{2,}, \ldots, \boldsymbol{x}_i\}$$

$$\boldsymbol{x}_i = \{x_i^1, x_i^2, \ldots, x_i^j\} \text{ -> } \{x_i^1, x_i^2\}$$

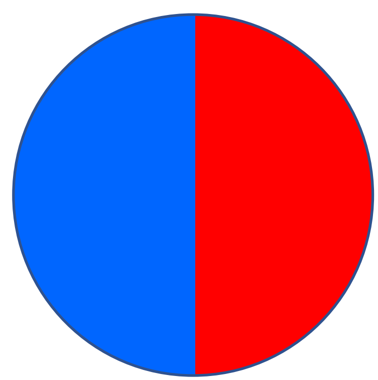
- ◆ Normalize the Solution space

$$0 \leq x_i^1 \leq 1,\ 0 \leq x_i^2 \leq 1$$

$$\boldsymbol{x}_i = \{x_i^1, x_i^2, \ldots, x_i^j\}$$



$$\boldsymbol{x}_i = \{x_i^1, x_i^2\}$$



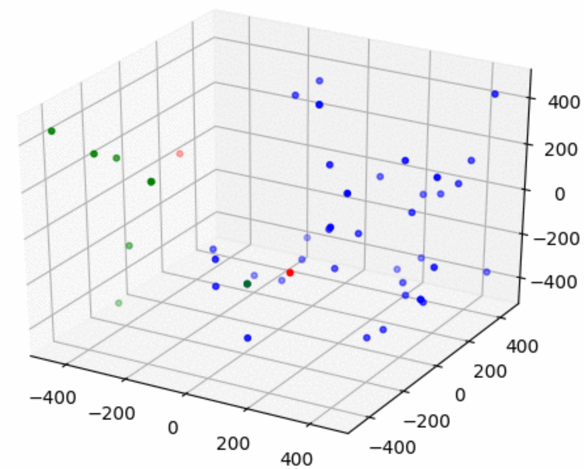
龍谷大学
RYUKOKU UNIVERSITY

Proposed Method

Point ◆ Clustering for estimating a population

◆In the previous method, individual diversity is not considered, because ps calculates only based on the number of generations.

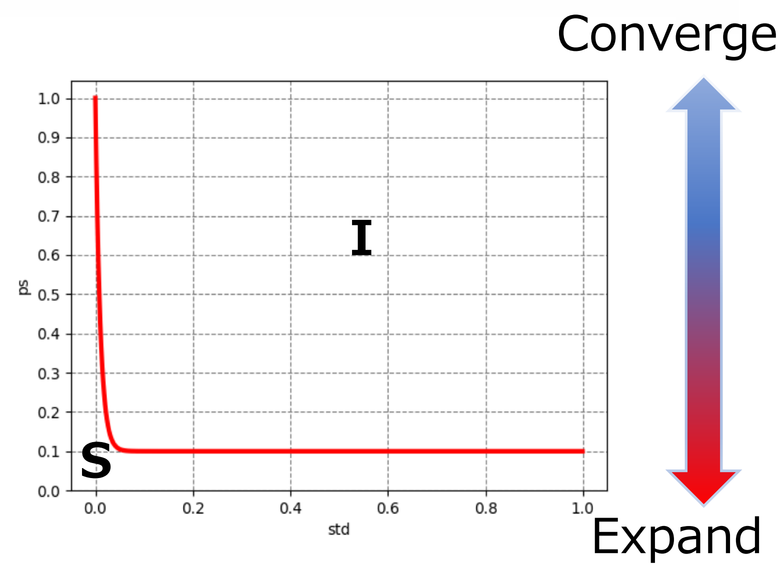
◆The proposed method adopts a population clustering method (Dirichlet process gaussian mixture model)to capture a landscape by using ps.



Ps utilizes a standard deviation for each cluster and determine accoding to the following formula:

$$ps = 0.1 + 0.9 * (1 - \bar{\sigma}(C_n)^{100})$$

$\bar{\sigma}(C_n)$:Standard deviation of each cluster



龍谷大学
RYUKOKU UNIVERSITY

Crossover

◆Insert an individual randomly

$FOR\ j = 1\ to\ D$

$$u_{i,g}^{j} = \begin{cases} v_{i,g}^{j} \ \text{if}(rand_i^j(0,1) \leq CR_i\ or\ j = j_{rand}) \\ x_{i,g}^{j}, otherwise \end{cases}$$

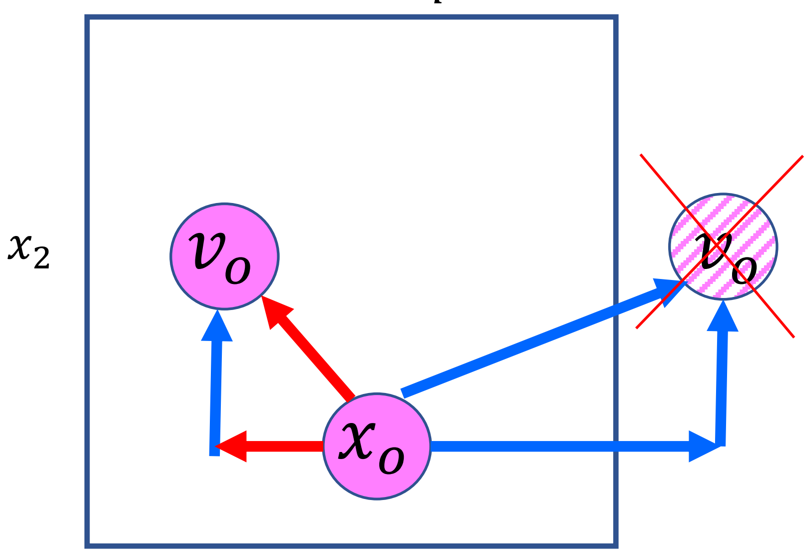
◆ Select crossover points at random

$IF(u_{i,g}^{j} < L \quad or\ u_{i,g}^{j} > U \quad)$

$$u_{i,g}^{j} = L \quad + rand_i^j(0,1) \cdot (U \quad - L \quad)$$

◆ If the value of the j-th dimension is out of solution space, a next individual gemerated at ramdom in the solution space instead of pulling back

Solution Space



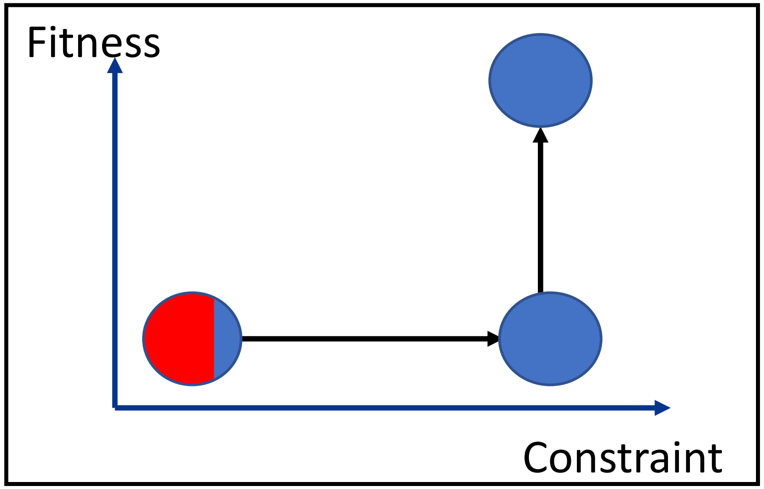
Selection

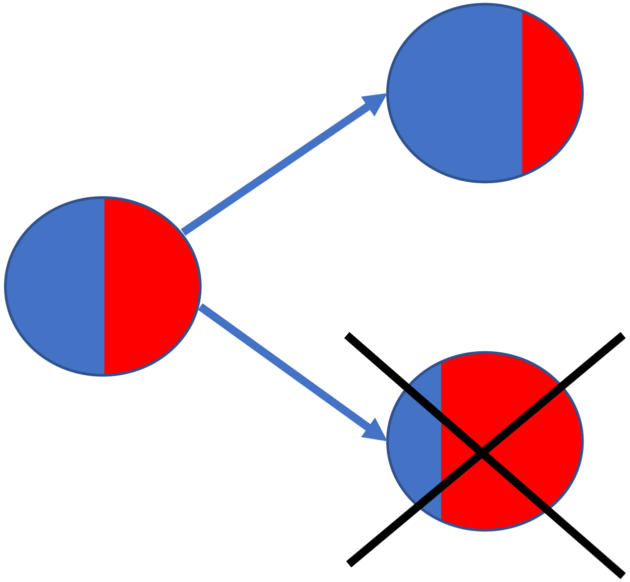
◆Optimize individuals in two stages

◆ Update an individual by comparing the number of elements that satisfy the constraints

↓

◆ If all constraints are satisfied and its fitness is better than before, an individual is update.





Parameter Settnigs

◆The proposed method doesn’t need a parameter fitting method

3rd Evolutionary Computation Competition, December 14, 2019

Algorithm Presentation (s05, m05)

Jernej Zupančič, Aljoša Vodopija, Tea Tušar,
Erik Dovgan, Bogdan Filipič

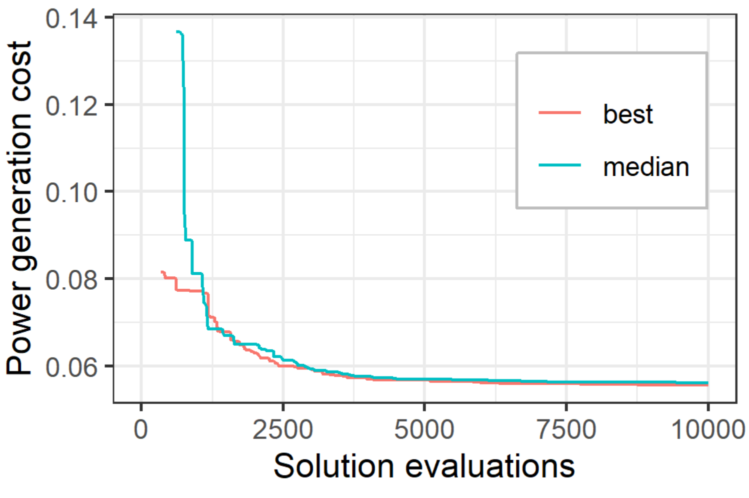
Jožef Stefan Institute (JSI), Ljubljana, Slovenia

Single-objective optimization algorithm (s05)

- Algorithm: jDE (Python Package: pygmo, function: saDE)
- DoE method: Latin hypercube sampling
- Constraint handling technique (CHT): dynamic penalty function

$$\bar{f}(x) = f(x) + (ct)^{\alpha} \sum_{i} v_i(x)$$

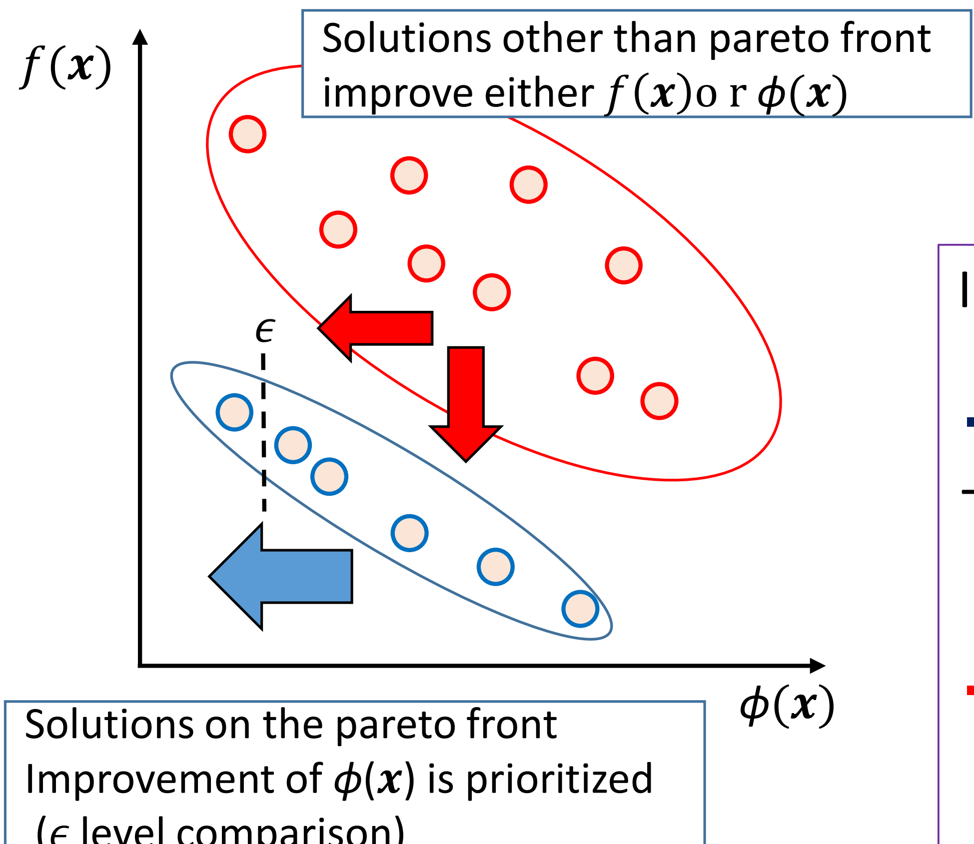
- Parameters and configuration:
  - Population size: 20
  - No. of generations: 500
  - DE variant: rand/1/bin
  - CHT parameters: c = 1.0, α = 1.0



Evolution Computation Competition 2019
Single objcective optimization

Jun-ichi Kushida (Hiroshima City University)

- **Method:** DE with ε constraint method and pareto approach
- Optimize constraint violation $\Phi(\boldsymbol{x})$ and $f(\boldsymbol{x})$ separately



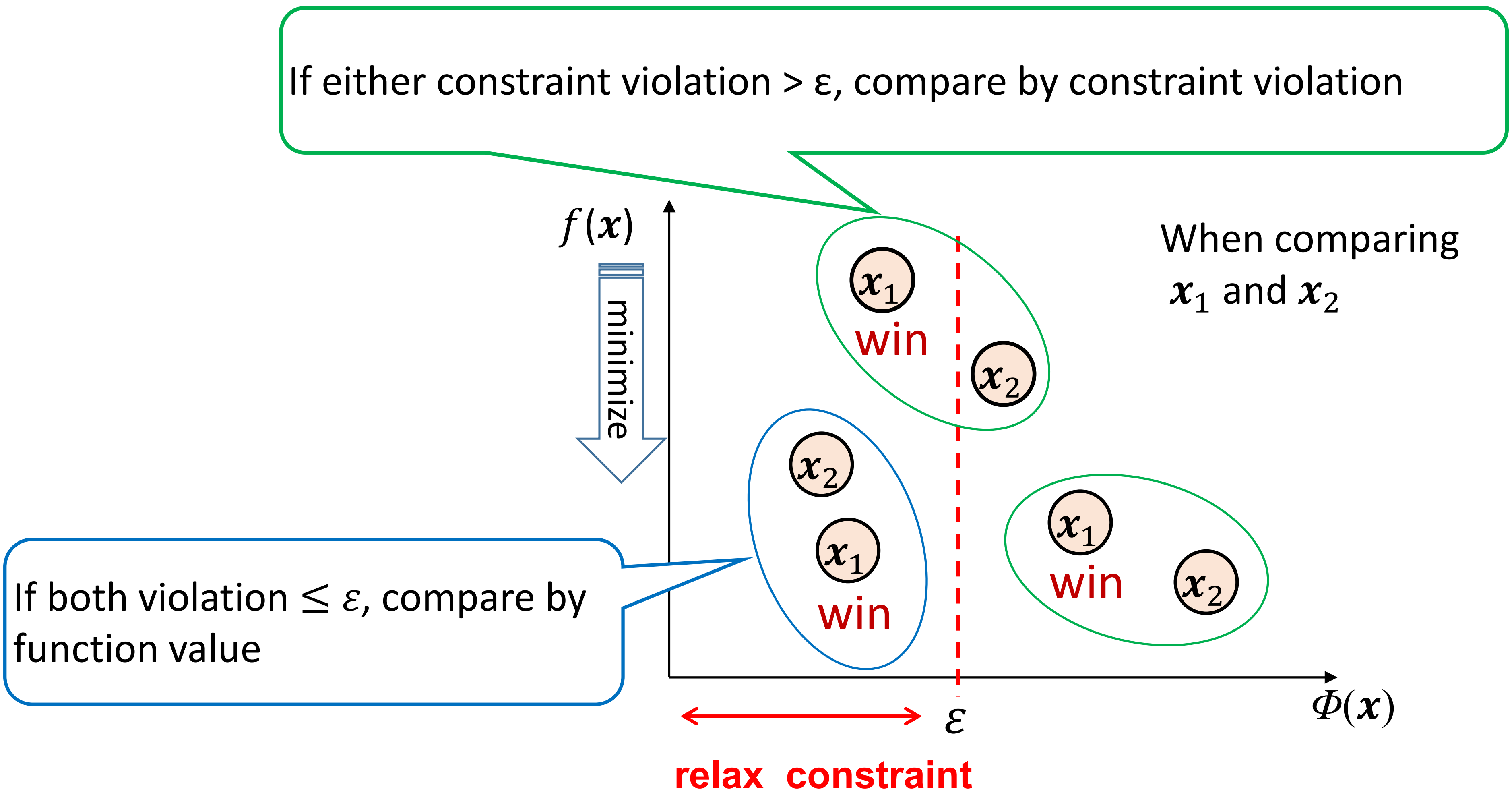
In the two-objective space (f, Φ space)

▪ Individual with Pareto rank = 1
→ Prioritize improvement of $\Phi(\boldsymbol{x})$
(Search feasible solution)

▪ Individuals with Pareto rank !=1
→ Preserve diversity

ϵ epsilon level comparison

Relax constraints by ϵ and compare parent and child



Control of ϵ level

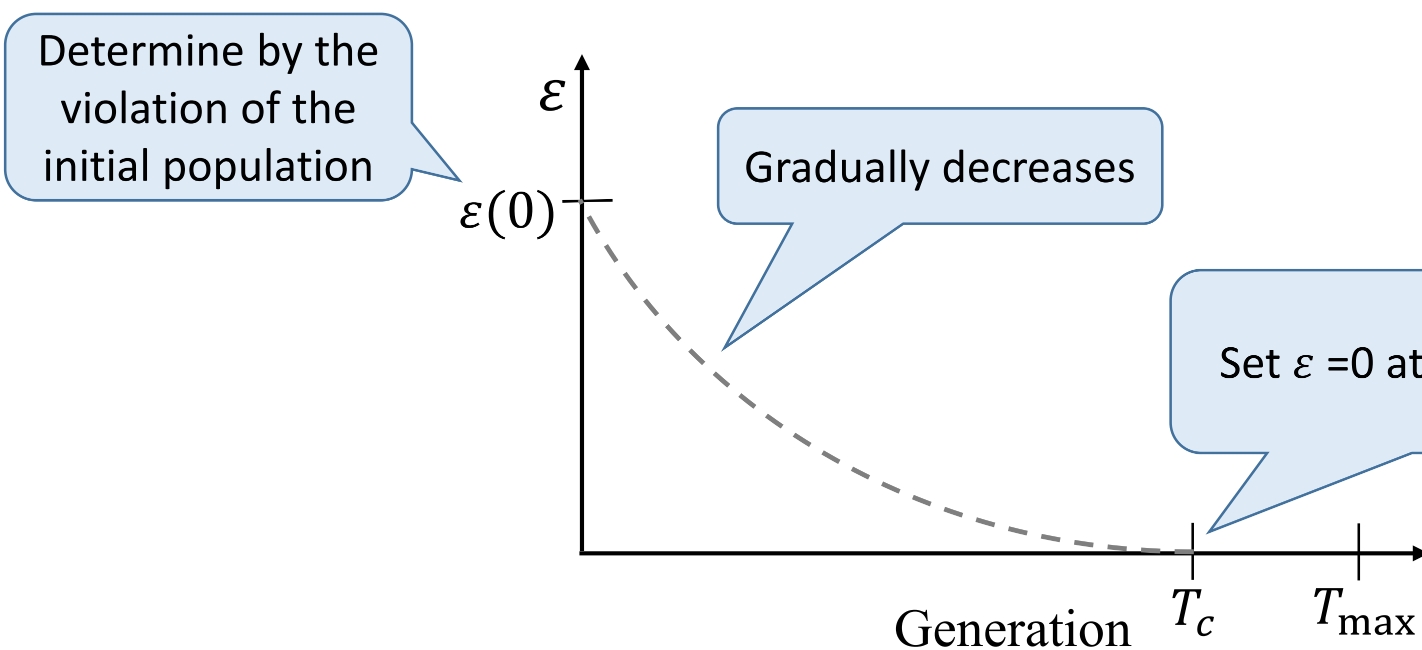
ε value value in generation t

$$\epsilon(t) = \begin{cases} \epsilon(0)(1 - \frac{t}{T_c})^{cp}, & 0 < t < T_c \\ 0, & t \geq T_c \end{cases}$$

cp: parameter for control of ε

Initial ε value : $\varepsilon(0) = \Phi(\boldsymbol{x}_\theta)$

$\boldsymbol{x}_\theta$ is the θth individual among the initial individuals sorted in ascending order by constraint violation ($\theta = r \times NP$)



Setup

Parameter	value
Population size NP	50 -> 10 (after T_c gen.)
T_c	140th generation
r	0.1
cp	3

- Strategy of ε DE: rand/1 /bin

  Mutant vector $\boldsymbol{v} = \boldsymbol{x}_{r1} + F_t(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3})$

  $\boldsymbol{x}_{r1}$ is randomly selected from individuals with pareto rank = 1

Parameter of t-th generation

- $F_t = 0.6 - 0.3 * \frac{\varepsilon(t)}{\varepsilon(0)}$

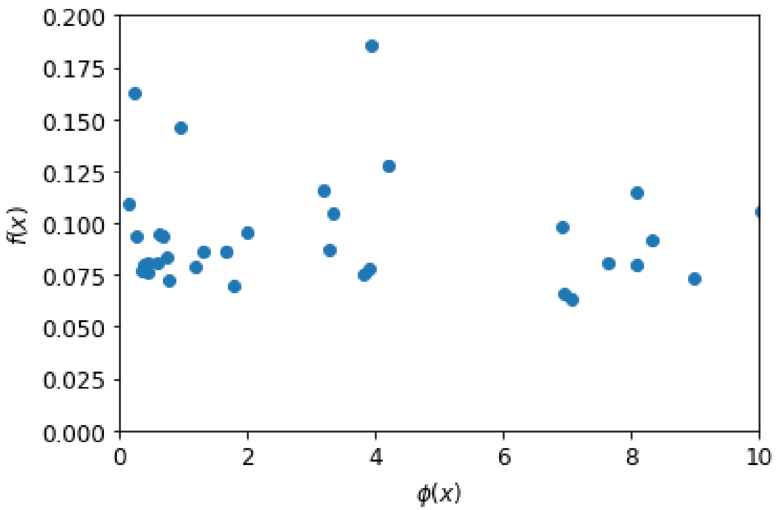
  Gradually increase 0.3 -> 0.6

- $CR_t = 0.1 + 0.9 * \frac{\varepsilon(t)}{\varepsilon(0)}$

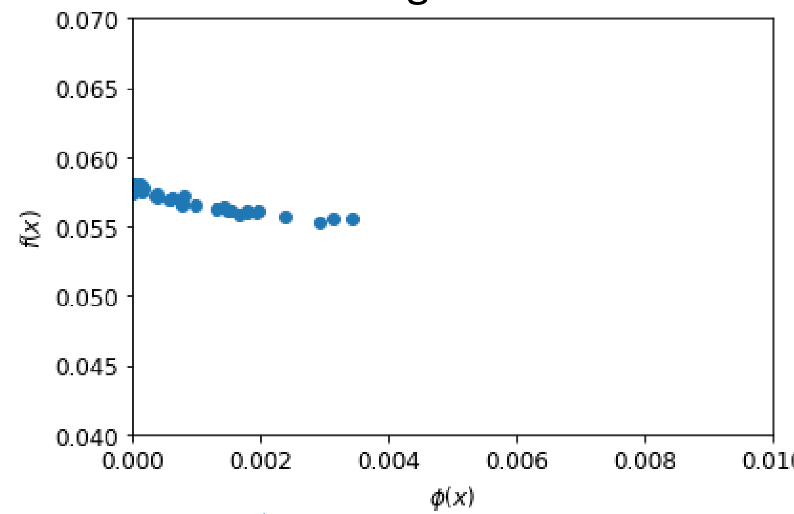
  Gradually decrease 0.9 -> 0.1

Convergence of populations in f, ϕ space

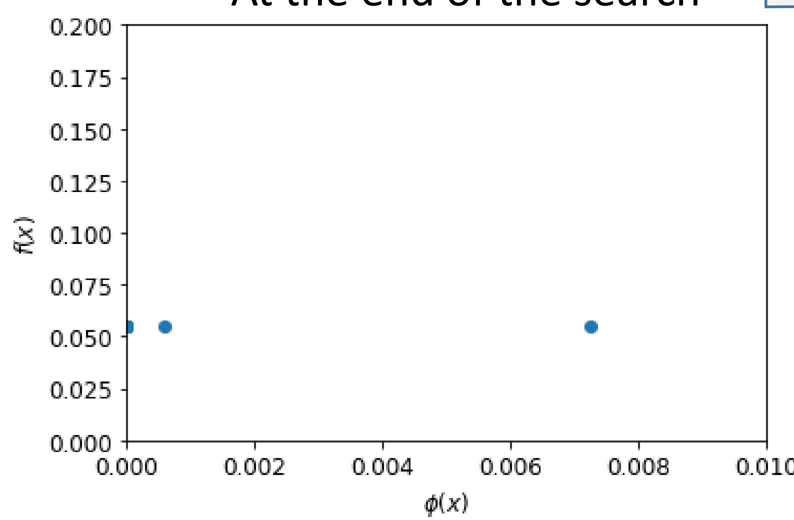
Early stage of search



Later generations



At the end of the search

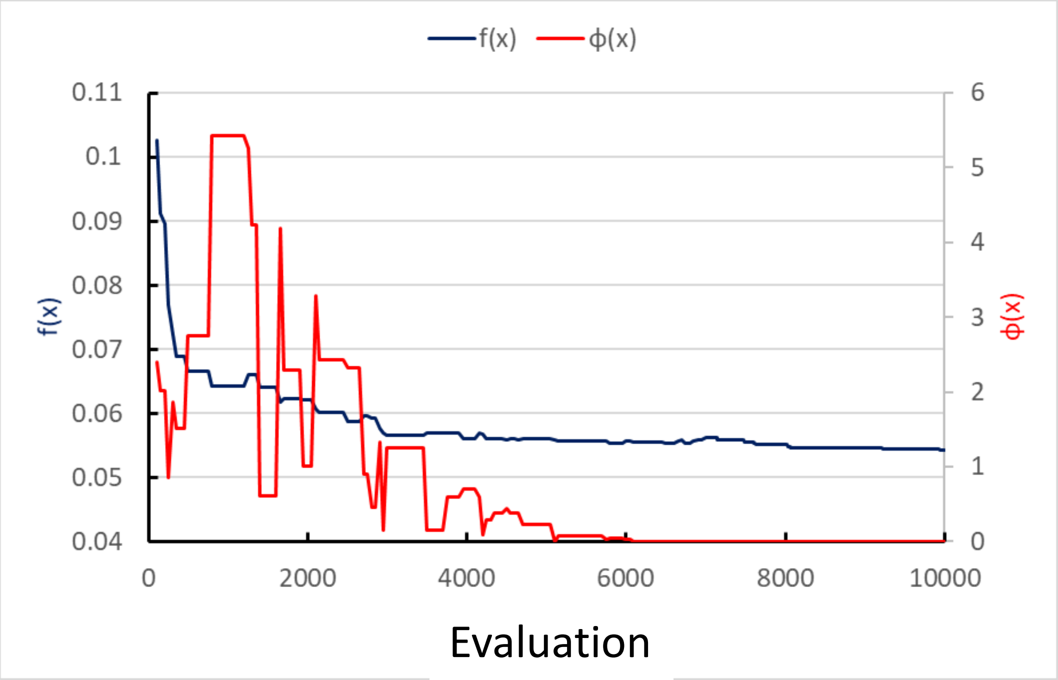


Experimental result

Average, maximum, median, and average values over 21 trials

min	0.054334
max	0.057439
median	0.055819
average	0.055805

Transition of the best solution for the trial of median



Complexity Reduction Fast Moving Natural Evolution Strategy

Number: s08

Takuya kato, Kazuki Kamata and Isao Ono
Tokyo Institute of Technology

Natural Evolution Strategy (NES) [Wierstra 08]

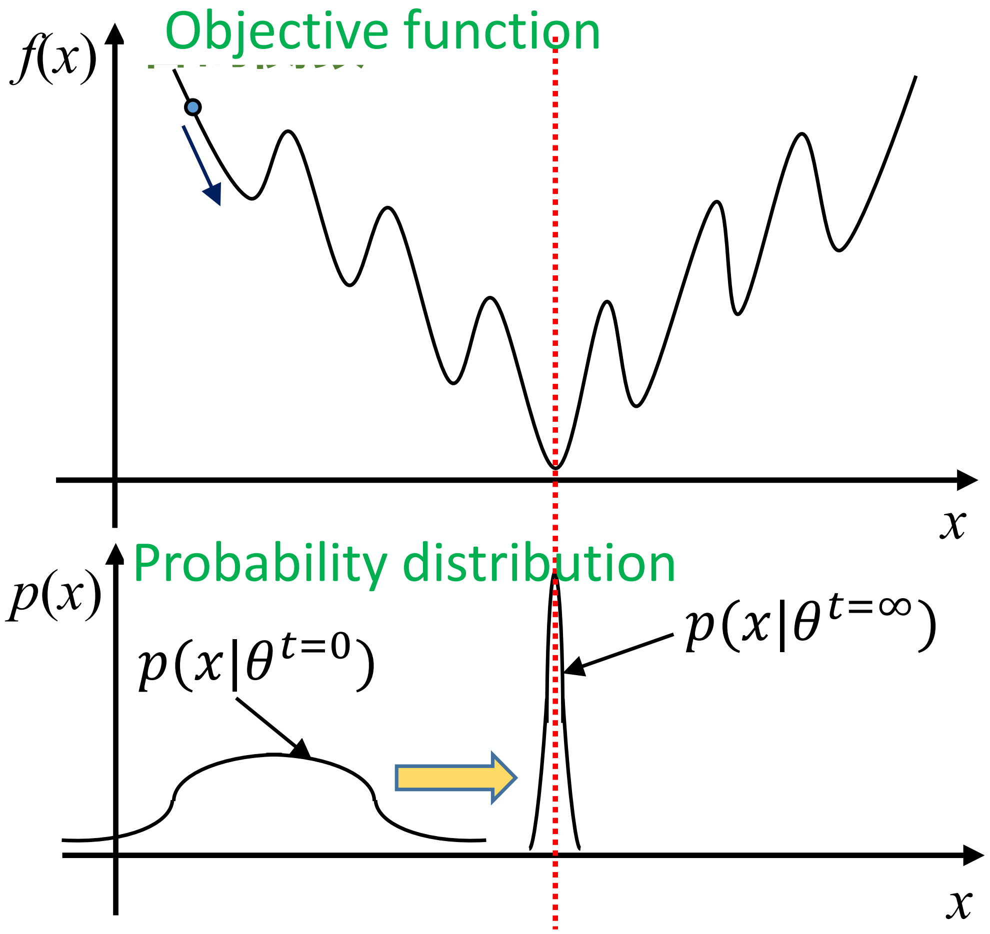
- Minimizes the expected objective function value:

$$J(\boldsymbol{\theta}) = \int f(\boldsymbol{x}) p(\boldsymbol{x}|\boldsymbol{\theta}) \boldsymbol{dx}.$$

  - ➢ $f(\boldsymbol{x})$: objective function
  - ➢ $p(\boldsymbol{x}|\boldsymbol{\theta})$: probability distribution
  - ➢ $\boldsymbol{\theta}$: parameter of probability distribution
- Natural gradient descent method [Amari 85]:

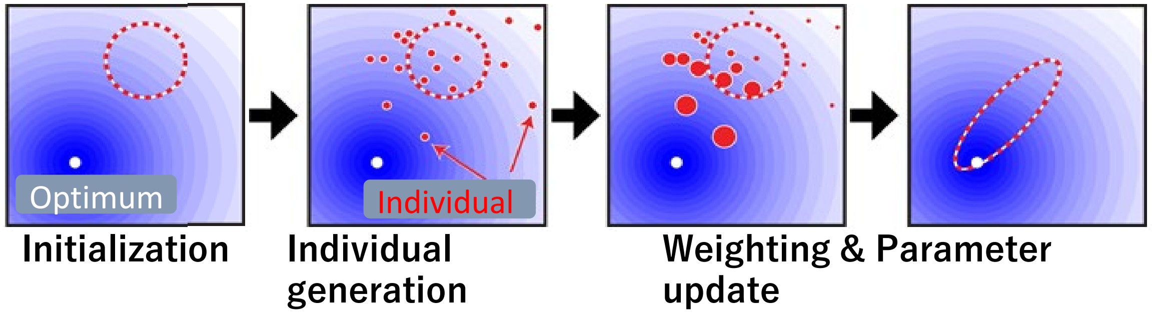
$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \eta \boldsymbol{F}^{-1}(\boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}).$$

  - ➢ η: learning rate
  - ➢ $\boldsymbol{F}(\boldsymbol{\theta})$: Fisher's information matrix
    - ◆ $\boldsymbol{F}(\boldsymbol{\theta}) = E_{\boldsymbol{x}}[\nabla_{\boldsymbol{\theta}} \ln p(\boldsymbol{x}|\boldsymbol{\theta}) (\nabla_{\boldsymbol{\theta}} \ln p(\boldsymbol{x}|\boldsymbol{\theta}))^{\mathrm{T}}]$



Natural Evolution Strategy (NES) [Wierstra 08]

- Algorithm when $p(x|\theta)$ is a normal distribution
  1. Initialize the generation $g = 0$ and the probability distribution $N\left(\boldsymbol{m}^{(g)}, \boldsymbol{C}^{(g)}\right)$.
  2. Make λ individuals $\{\boldsymbol{x}_i\}_{i=1}^{\lambda}$ according to $N(\boldsymbol{m}^{(g)}, \boldsymbol{C}^{(g)})$.
  3. Evaluate $\boldsymbol{x}_i$, and update $\boldsymbol{m}$ and $\boldsymbol{C}$ as follows.
$$\boldsymbol{m} \leftarrow \boldsymbol{m} - \eta \sum_{i=1}^{\lambda} \frac{f(\boldsymbol{x}_i)}{\lambda} (\boldsymbol{x}_i - \boldsymbol{m})$$
$$\boldsymbol{C} \leftarrow \boldsymbol{C} - \eta \sum_{i=1}^{\lambda} \frac{f(\boldsymbol{x}_i)}{\lambda} \left((\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^{\mathrm{T}} - \boldsymbol{C}\right)$$
  4. If a stop condition is not met, $g = g + 1$ and go to step 2.



Natural Evolution Strategy (NES) [Wierstra 08]

- Fitness shaping [Wierstra 08]
  - ➢Makes the algorithm invariant under monotonically increasing transformation.
  - ➢Replaces $-\frac{f(\boldsymbol{x}_i)}{\lambda}$ with a normalized weight w_i.

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \eta \sum_{i=1}^{\lambda} w_i (\boldsymbol{x}_i - \boldsymbol{m})$$

$$\boldsymbol{C} \leftarrow \boldsymbol{C} - \eta \sum_{i=1}^{\lambda} w_i \left((\boldsymbol{x}_i - \boldsymbol{m})(\boldsymbol{x}_i - \boldsymbol{m})^{\mathrm{T}} - \boldsymbol{C} \right)$$

◆The better $f(\boldsymbol{x}_i)$ is, the larger w_i is.

$$w_i^{\mathrm{rank}} = \frac{\widehat{w}_i}{\sum_{j=1}^{\lambda} \widehat{w}_j} - \frac{1}{\lambda},$$

$$\widehat{w}_i = max\left(0, \ln\left(\frac{\lambda}{2} + 1\right) - \ln(i_{\mathrm{ord}})\right), \text{where } i_{\mathrm{ord}}$$ is rank relating to $f(\boldsymbol{x})$.

Complexity Reduction Fast Moving Natural Evolution Strategy (CR-FM-NES)[Nomura 17]

- Uses a normal distribution with a restricted covariance matrix as $p(x|\boldsymbol{\theta})$.
  - Covariance matrix: $\sigma^2 \boldsymbol{D}(I + \boldsymbol{v}\boldsymbol{v}^T)\boldsymbol{D}$ [Akimoto 14]
    - $\boldsymbol{D}$: diagonal matrix
    - $\boldsymbol{v}$: vector
    - σ : scalar
  - Mean vector: $\boldsymbol{m}$
  - Parameter of normal distribution:

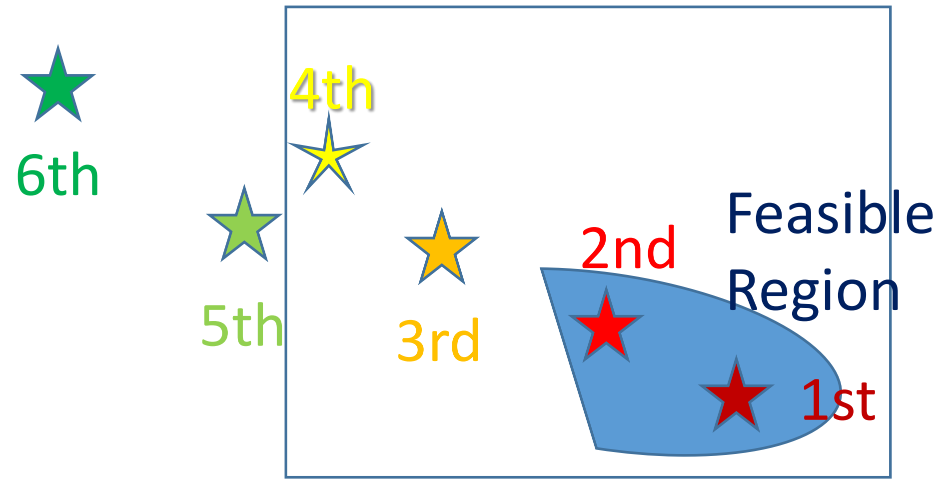
$$\boldsymbol{\theta} = (\boldsymbol{m}, \sigma, \boldsymbol{v}, \boldsymbol{D})$$

- Reduces time and space complexity.

- Algorithm:
  1. Initialize each variables.
  2. Generate λ samples: $\boldsymbol{x}_i \sim p(\boldsymbol{x}|\boldsymbol{\theta})$.
  3. Sort $\boldsymbol{x}_i$ with a preference order operator $<_{\mathrm{p}}$.
  4. Switch learning rates of $\sigma, \boldsymbol{v}$ and $\boldsymbol{D}$ according to search situation.
  5. Update $\boldsymbol{\theta} = (\boldsymbol{m}, \sigma, \boldsymbol{v}, \boldsymbol{D})$ using natural gradient.
  6. If the stopping condition is met, stop, otherwise $g \leftarrow g + 1$ then go to step 2.

CR-FM-NES for Wind Turbine Design Optimization Problem

- Parameters:
  - ➢Sample size: 48
  - ➢Others: default
- Preference order operator: $<_p$
  1. Upper and lower constraint violation
  2. Problem constraint violation
  3. Objective function
- Solutions which do not meet upper and lower constraint are not simulated.



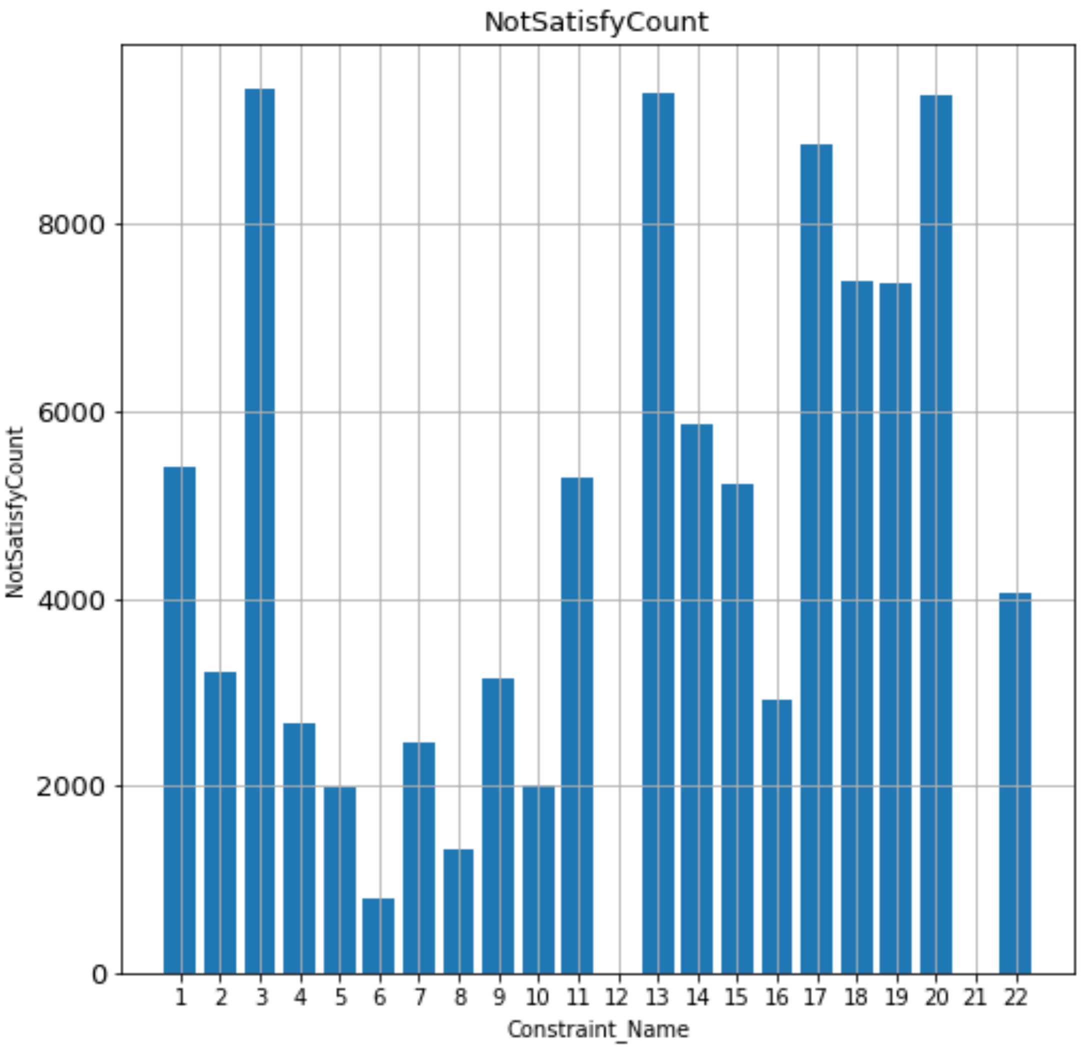
	1st	2nd	3rd	4th	5th	6th
Objective function	1	2	1	1	-	-
Problem constraint violation	0	0	1	2	-	-
Upper and lower constraint violation	0	0	0	0	1	2

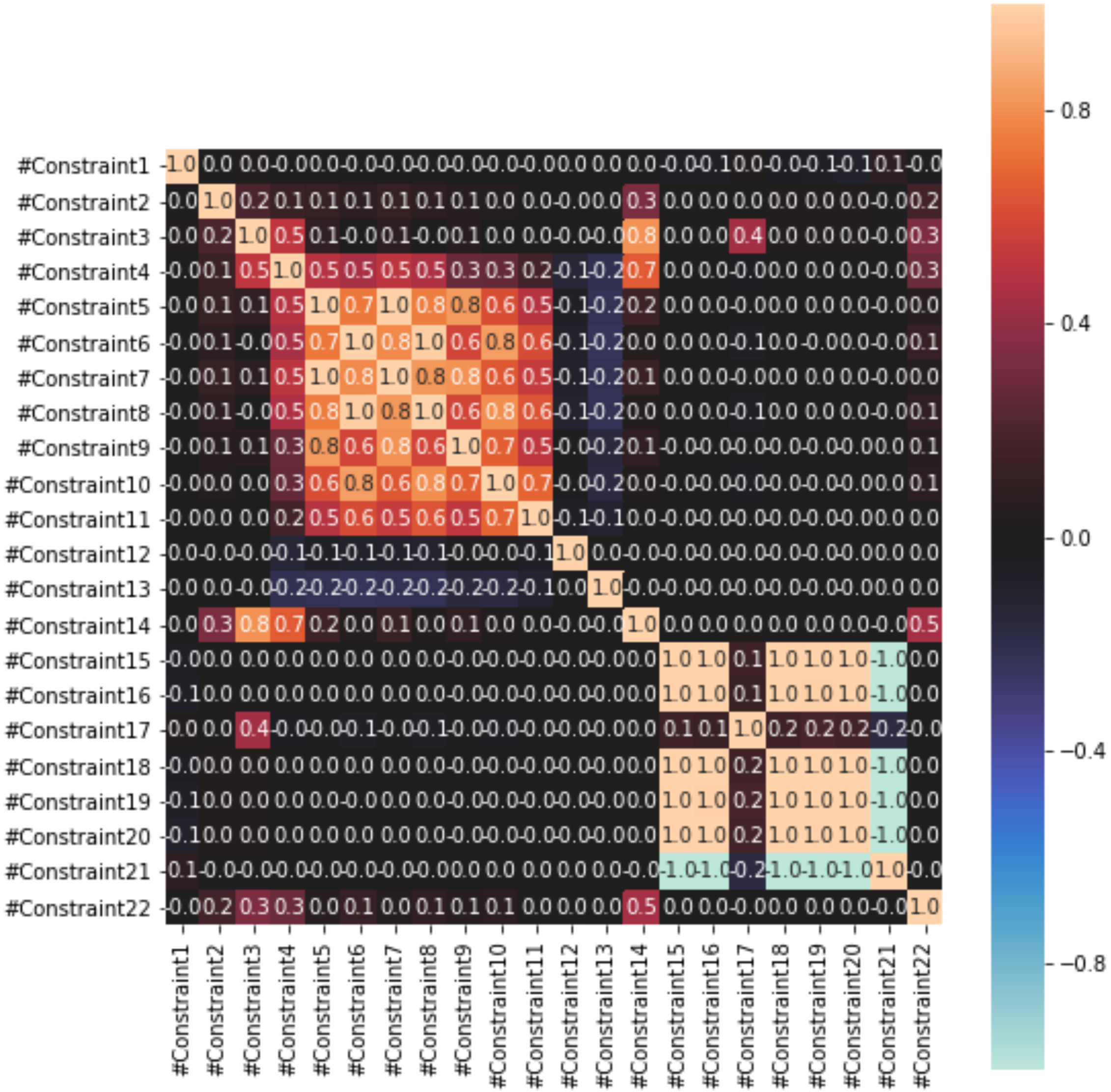
The 3rd Evolutionary Computation Competition

University of Tsukuba, M1 Yuta Kobayashi

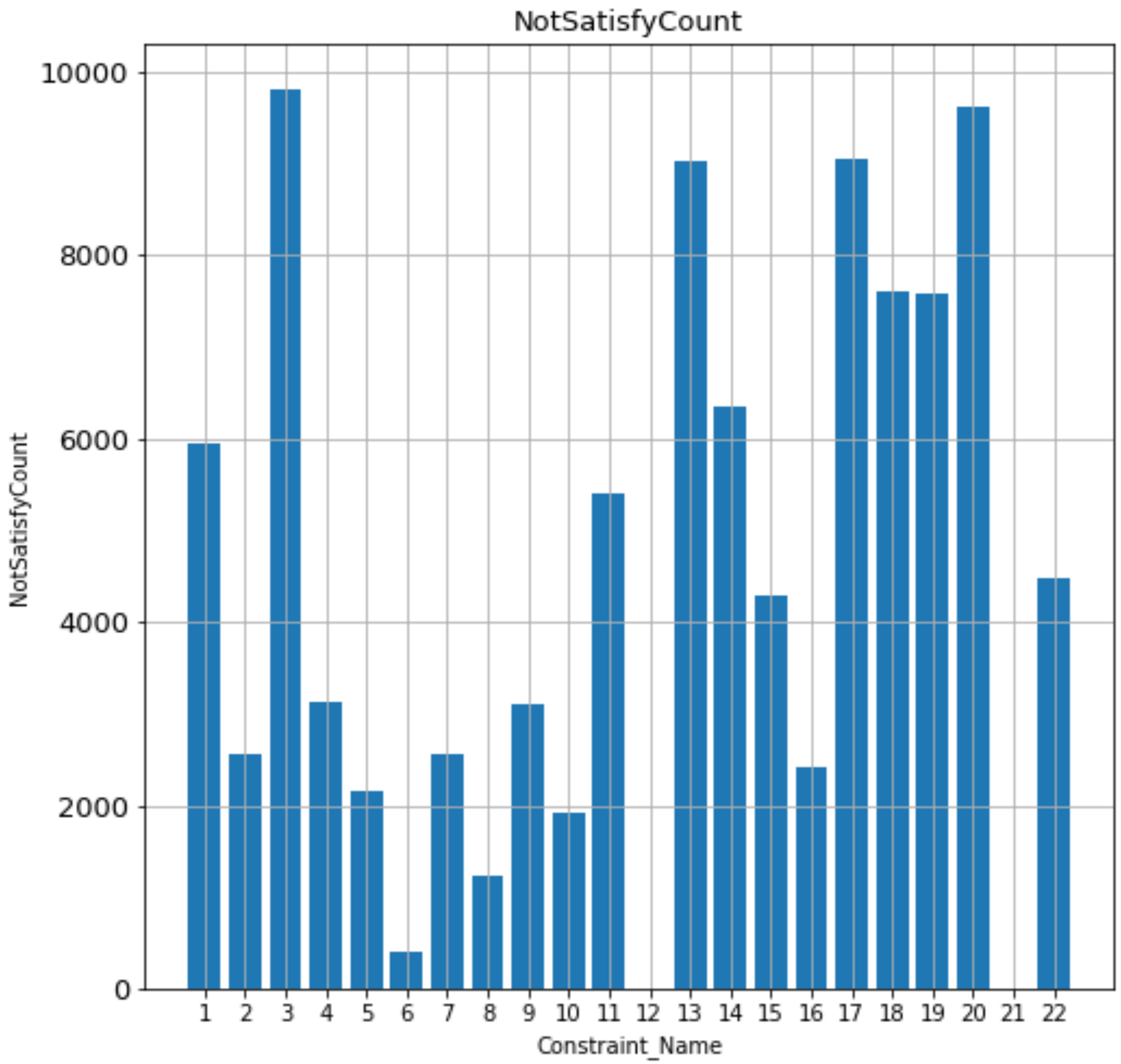
University of Tsukuba, Claus Aranha

Problem Exploration

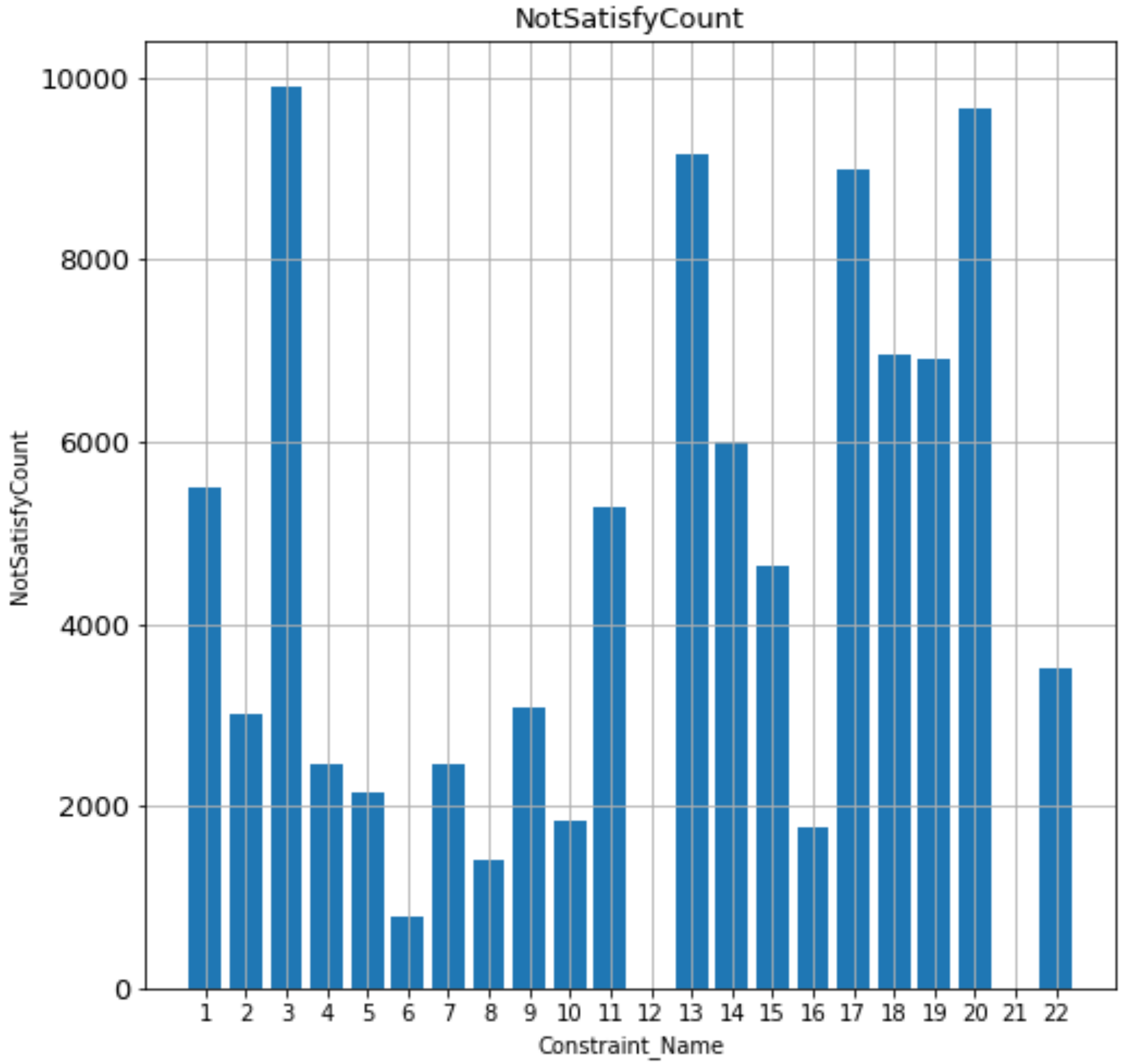




Problem Exploration



3-13



3-13-20

2-step Search

- finding non-constrained initial population:
- $z_i = \frac{c_i - \mu_i}{\sigma_i}$
- $f = \sum_{i=1}^{22} w_i z_i$
- w_i: weight of constraint
- c_i: value of constraint
- μ_i : mean of constraint
- σ_i: std of constraint

8000 evaluations

・Initialize by Latin Hyper Cube Sampling
・Search spaces that satisfy constraint by using DE

DE

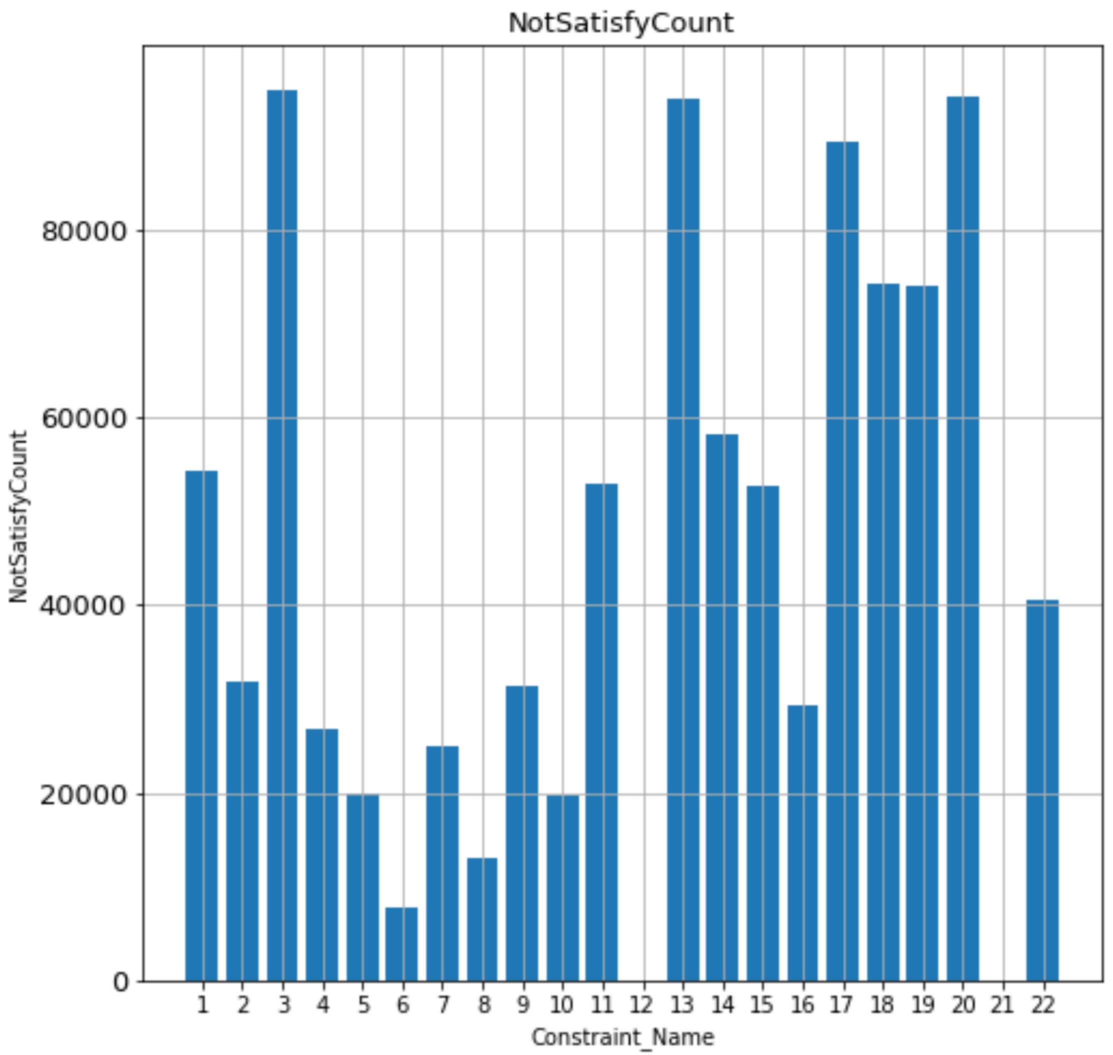
2000 evaluations

・Initialize populations by using solutions that satisfy all constraints
・Search better solutions by MOEA/D-DE

MOEA/D-DE

Pre-experiment - how to find constraint weight

Constraints	not_satisfy_propotion
#Constraint1	0.54192
#Constraint2	0.31853
#Constraint3	0.94864
#Constraint4	0.26685
#Constraint5	0.20023
#Constraint6	0.7698
#Constraint7	0.24959
#Constraint8	0.13028
#Constraint9	0.31362
#Constraint10	0.19650
#Constraint11	0.52900
#Constraint12	0
#Constraint13	0.93847
#Constraint14	0.58237
#Constraint15	0.52680
#Constraint16	0.29401
#Constraint17	0.89310
#Constraint18	0.74274
#Constraint19	0.74027
#Constraint20	0.94039
#Constraint21	0
#Constraint22	0.40636

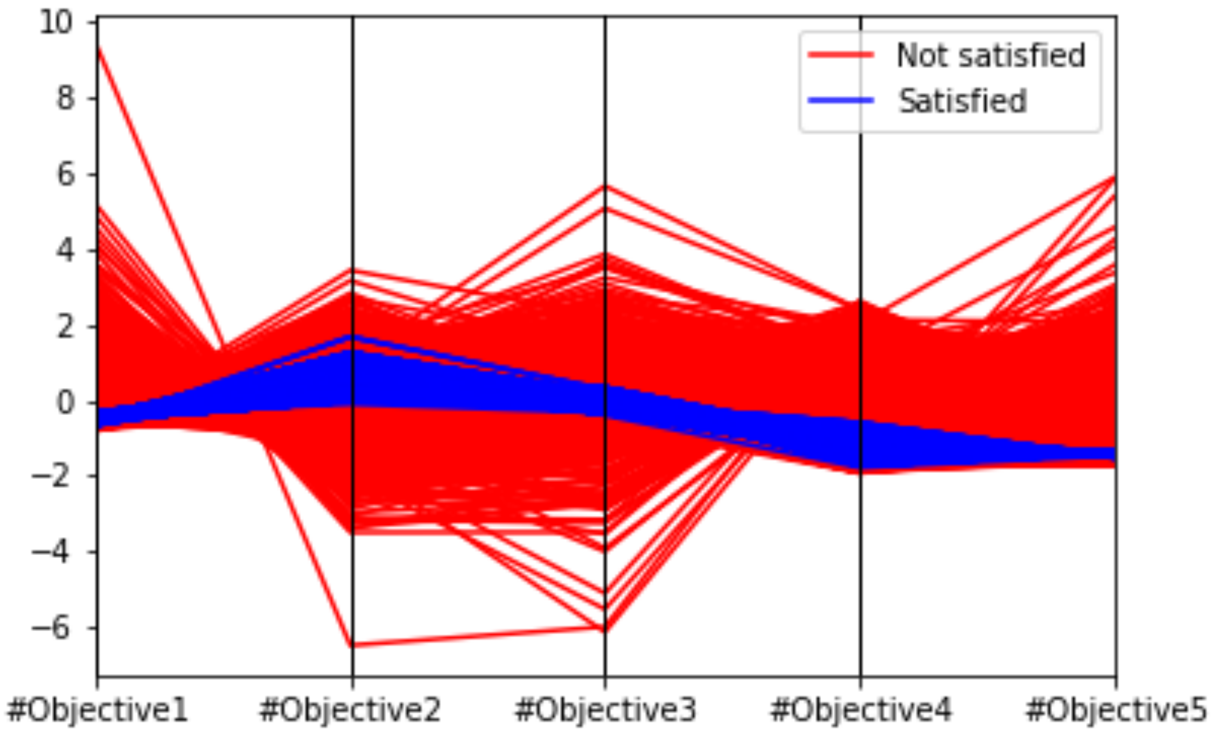


Parameter

DE		MOEAD-DE	
Population	100	Population	70
Crossover rate, C	1.0	Crossover rate, C	1.0
Scaling Factor, F	0.5	Scaling Factor	0.5
index parameter, η_m	20	index parameter, η_m	20
Mutation Rate	1/32	Mutation Rate	1/32
		Decomposition method	SLD
		Scalar aggregation function	Weighted Tchebycheff

Result

Trials	HyperVolume	satisfycount
1	2.791	696
2	2.337	586
3	3.246	716
4	3.126	609
5	2.431	585
6	3.177	570
7	2.833	648
8	3.301	603
9	3.276	506
10	2.923	782
11	3.004	590
12	3.302	579
13	2.863	713
14	3.200	536
15	2.963	617
16	2.995	540
17	3.029	593
18	2.452	689
19	2.949	610
20	2.402	615
21	2.772	638



Evolutionary Computation Symposium Competition 2019

Application of CM2T

Yuto Fujii[1], Taiki Hanada[1], Yiping Liu[1], Naoki Masuyama[1],Yusuke Nojima[1], and Hisao Ishibuchi[2]

[1]Osaka Prefecture University

[2]Southern University of Science and Technology

Introduction

Characteristics of Competition

1. Wind Turbine Optimization Problem is a severe constrained problem.

   ➡ Utilizing various infeasible solutions

2. The reference point for HV calculation is far from the true nadir point.

   ➡ Modifying initial weight vectors to obtain solutions near the edges of the Pareto front

Introduction

Characteristics of Competition

1. Wind Turbine Optimization Problem is a severe constrained problem.

   ➡ Utilizing various infeasible solutions

2. The reference point for HV calculation is far from the true nadir point.

   ➡ Modifying initial weight vectors to obtain solutions near the edges of the Pareto front

The Algorithm Utilizing Various Infeasible Solutions: CM2T

CM2T (Constrained Multi-objective to Two-objective) is a constrained multi-objective evolutionary algorithm based on decomposition.

Characteristic of CM2T

Solutions are evaluated and selected in each transformed two-objective optimization problem.

Minimize Objective 1: Scalarizing function value

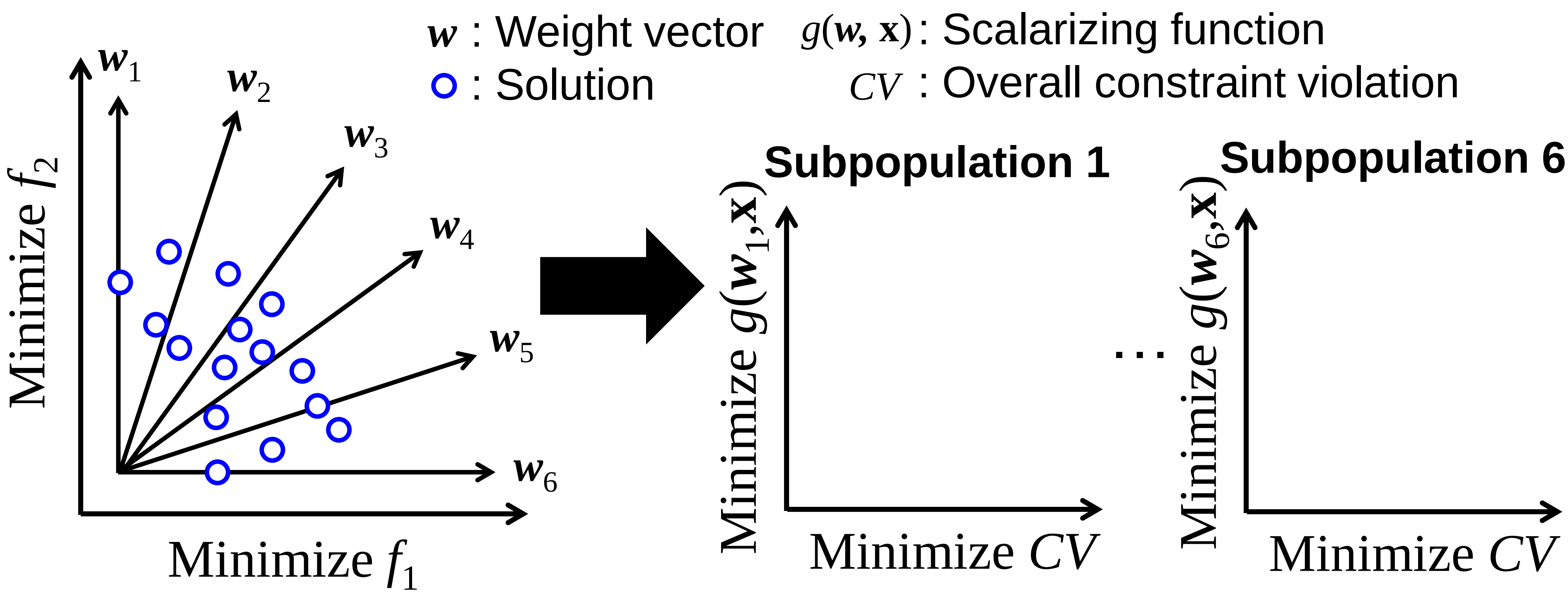
Minimize Objective 2: Overall constraint violation value

CM2T proposed paper

T. Fukase, N. Masuyama, Y. Nojima, and H. Ishibuchi, “A Constrained Multi-objective Evolutionary Algorithm Based on Transformation to Two-objective Optimization Problems,” In *Proc. of Intelligent System Symposium* (FAN2019), Toyama, 2019 (Japanese).

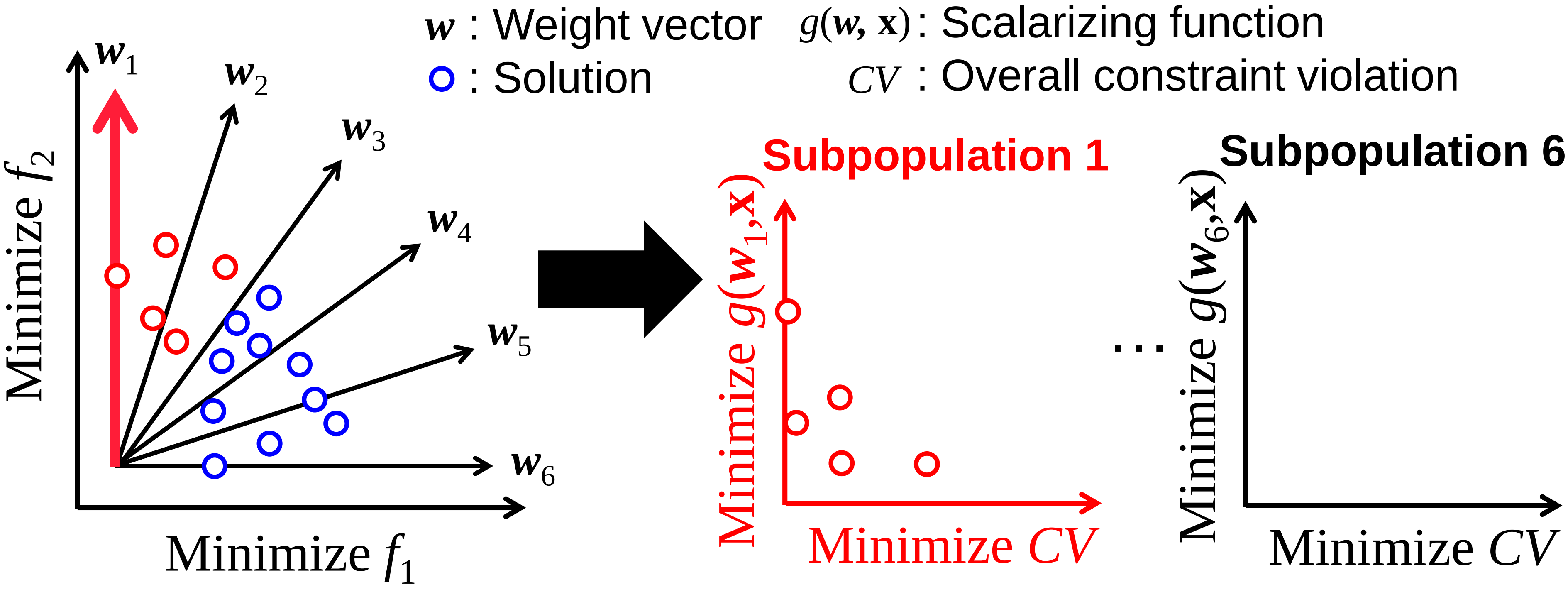
Problem Transformation

Solutions corresponding to each vector are evaluated in the transformed two-objective (scalarizing function and overall constraint violation) space.



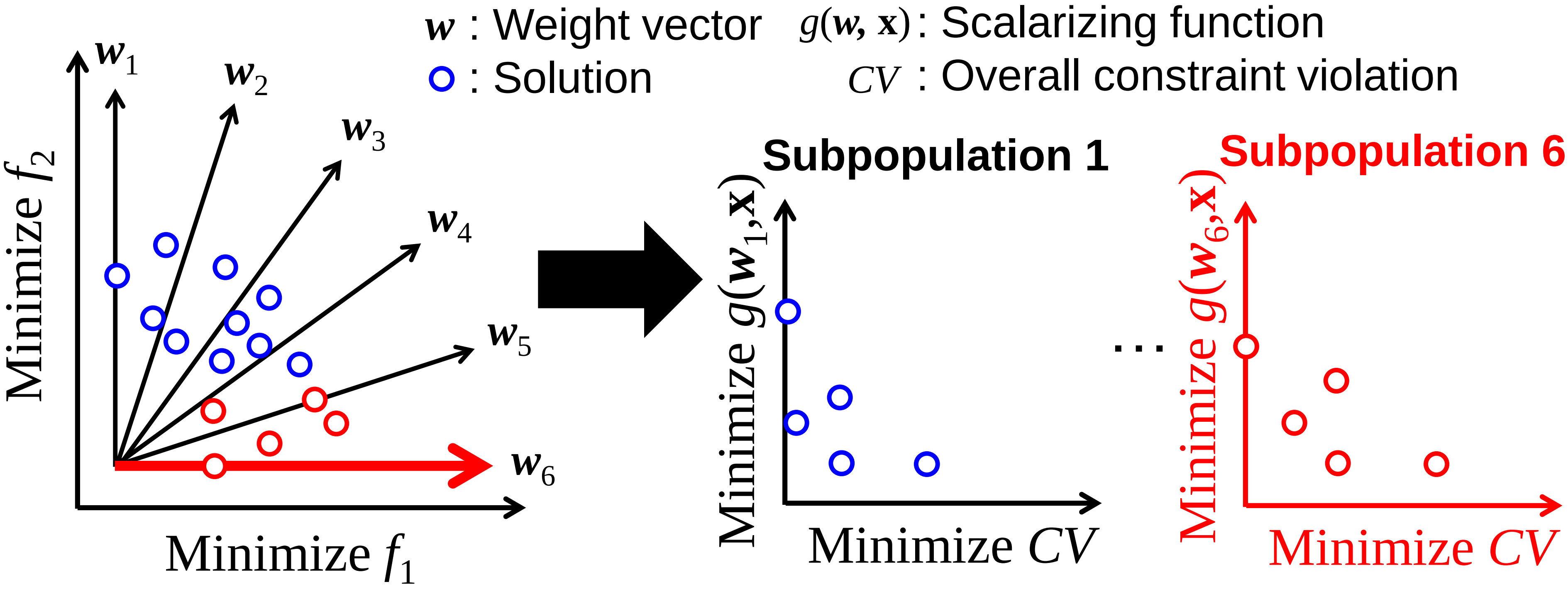
Problem Transformation

Solutions corresponding to each vector are evaluated in the transformed two-objective (scalarizing function and overall constraint violation) space.



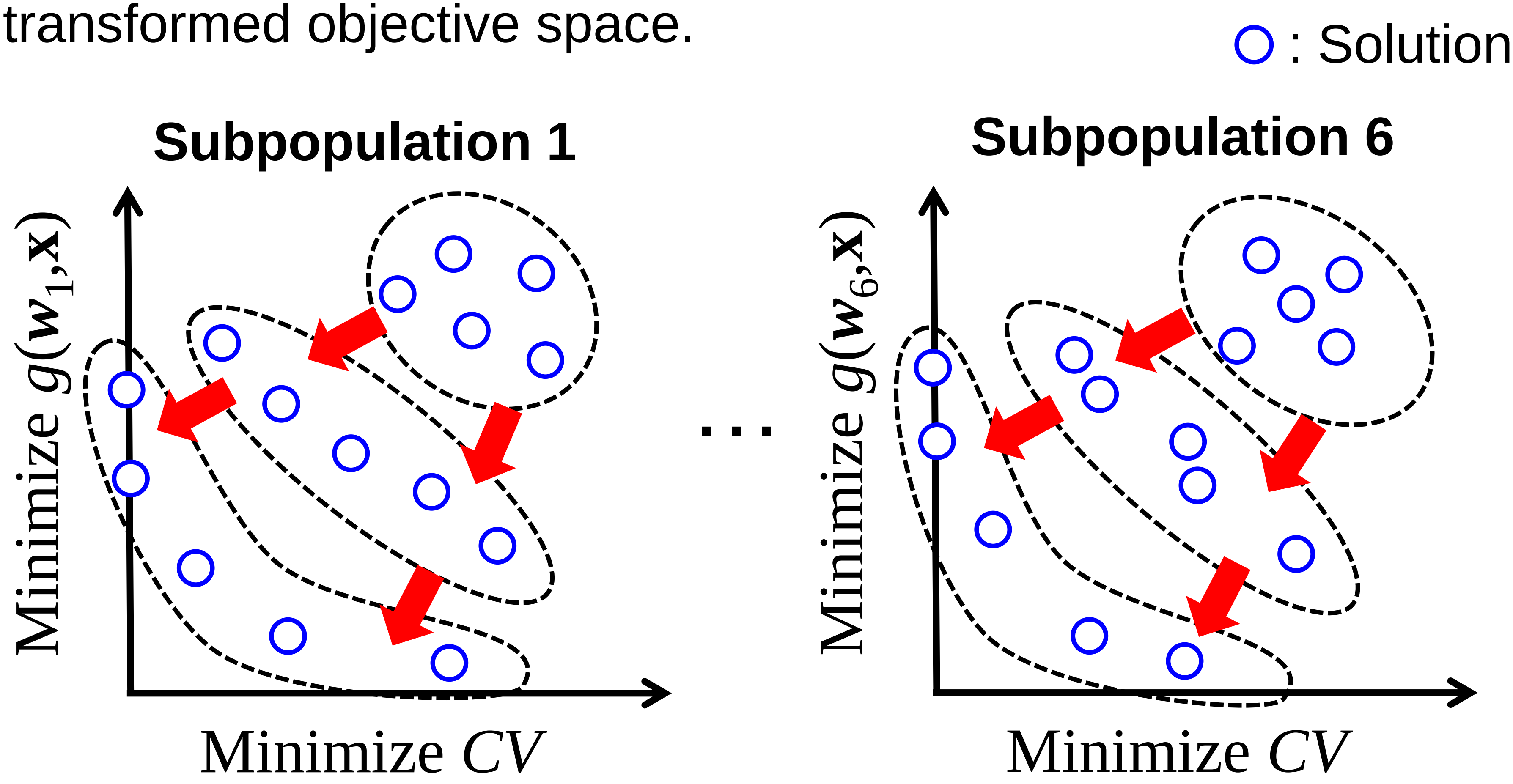
Problem Transformation

Solutions corresponding to each vector are evaluated in the transformed two-objective (scalarizing function and overall constraint violation) space.



Search Method in Each Subpopulation

Solutions in each subpopulation are evaluated and selected by the Pareto ranking and the crowding distance in the transformed objective space.



Characteristics of Competition

1. Wind Turbine Optimization Problem is a severe constrained problem.

   ➡ Utilizing various infeasible solutions

2. The reference point for HV calculation is far from the true nadir point.

   ➡ Modifying initial weight vectors to obtain solutions near the edges of the Pareto front

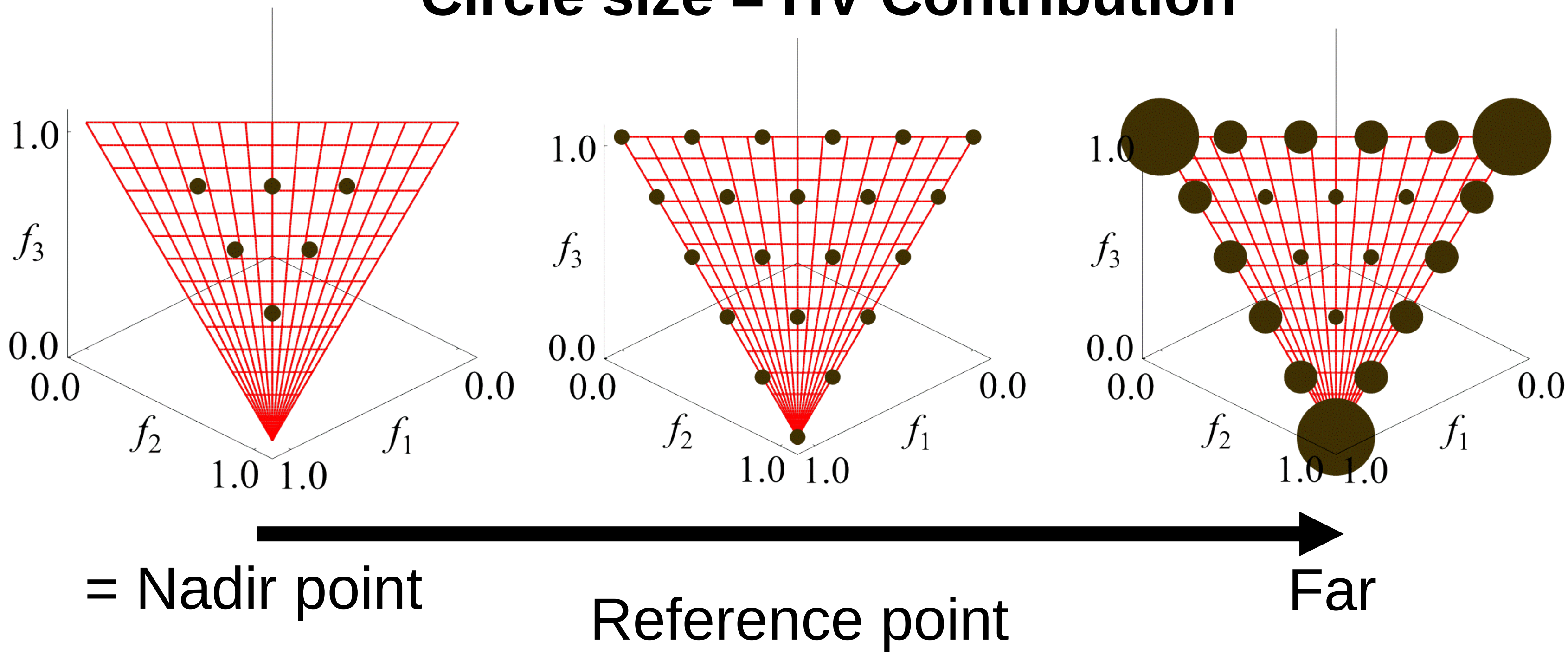
Modifying Initial Weight Vectors

Motivation

In the previous study [Ishibuchi et al. GECCO2017]

When the reference point is far from the nadir point, solutions at the edges of the Pareto front have larger HV contribution.

Circle size = HV Contribution



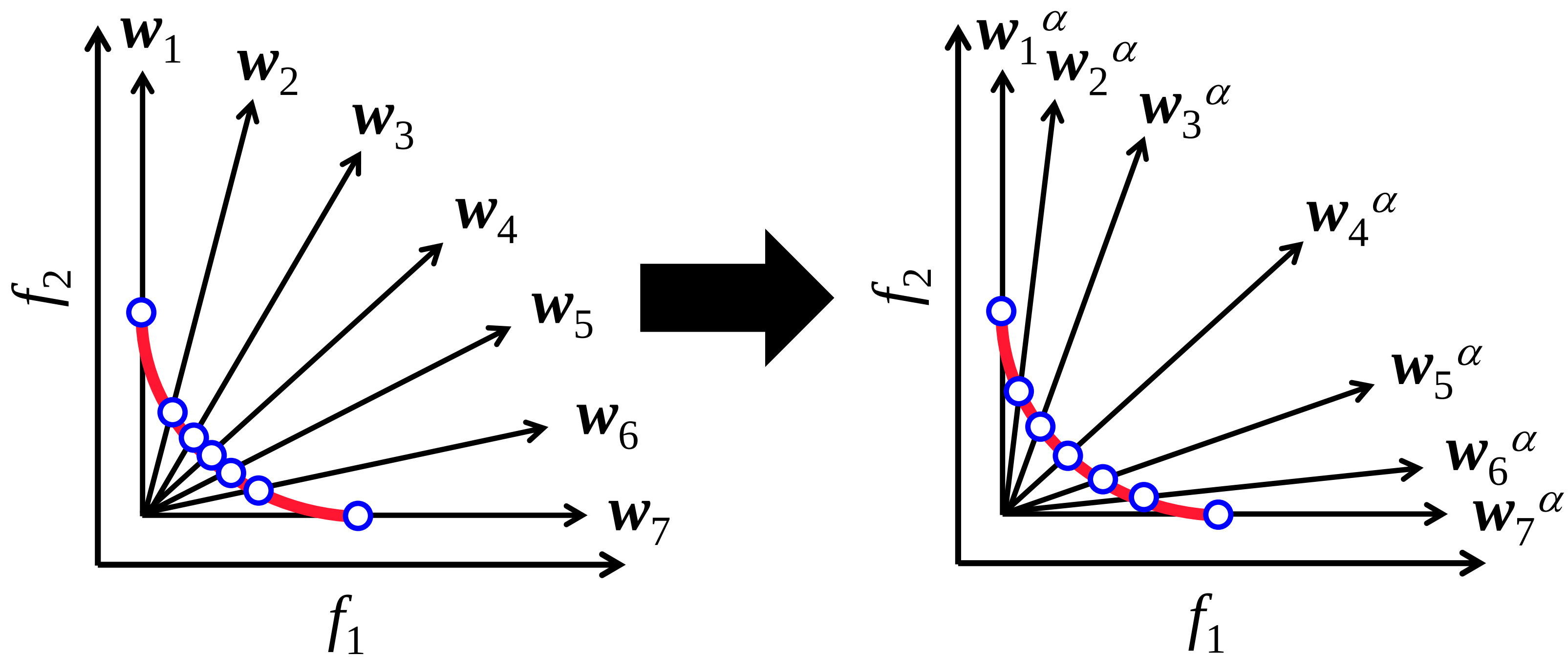
Modifying Initial Weight Vectors

Our Method

To search for solutions near the edges of the Pareto front, we raised the initial weight vectors to the power of α ($\alpha > 1$).

Before: $\boldsymbol{w} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_n)$ ➡ **After:** $\boldsymbol{w}' = (\boldsymbol{w}_1^{\alpha}, \boldsymbol{w}_2^{\alpha}, \ldots, \boldsymbol{w}_n^{\alpha})$

$\boldsymbol{w}$: Weight vector — : Pareto front ○ : Solution



Initial weight vectors ($\alpha = 1$)

$\alpha > 1$

Parameters

Population size : 210

Subpopulation size : 21

Scalarizing function : Normalized Tchebycheff

Crossover : SBX (DI: 20)

Probability of crossover : 1.0

Mutation : Polynomial Mutation (DI: 20)

Probability of mutation : 1 / 32

The power of weight vectors : 4

The parameter tuning is not applied.

Wind Turbine Design Optimization using a Many-objective Evolutionary Algorithm with a Single Set of Reference Vectors

Ahsanul Habib

Research Associate

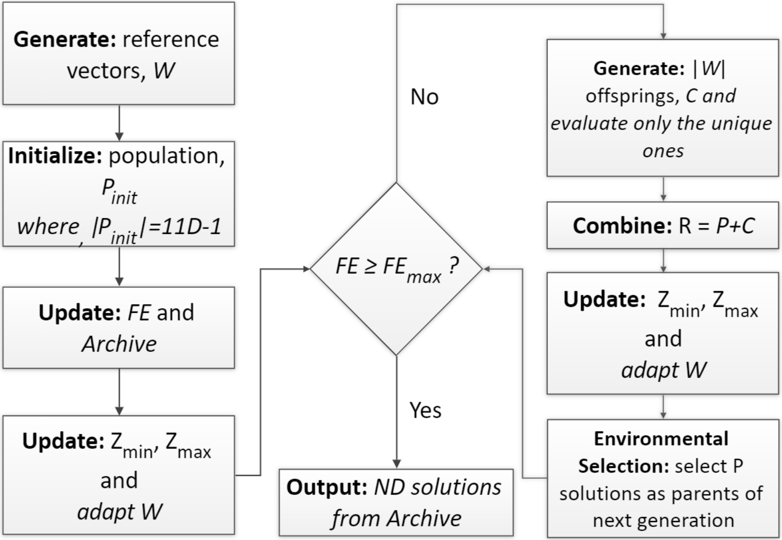
Multidisiplinary Design Optimization (MDO) Group

School of Engineering and Information Technology (SEIT)
University of New South Wales (UNSW), Canberra, Australia

14 December, 2019

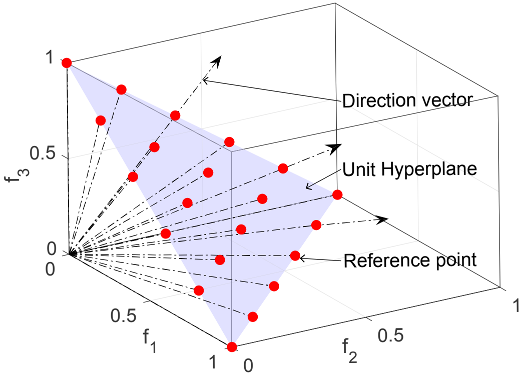
Outline

- A multi-objective wind turbine design optimization problem as part of the Evolutionary Computation Competition 2019.
- The problem involves 5 objectives, 32 continuous variables and 22 constraints, which are evaluated using WISDEM and OpenMDAO tools.
- The design optimization problem needs to be solved with a computational budget of 10,000 function evaluations.



Different steps of SRMEA.

Generation of Reference Vectors

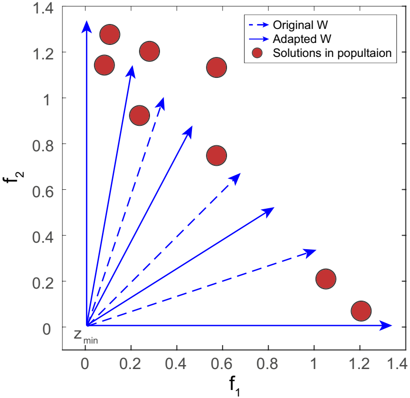


W, Reference vectors originating from z_{min}.

- The size of the initial population is predefined by the user (N_{init}).
- The solutions are initialized within the variable bounds using Latin hypercube sampling (LHS).

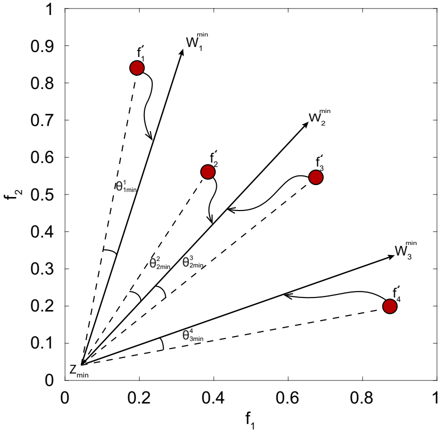
The update scheme for the i^{th} reference vector is presented below:

$$W_i = \frac{W_{0,i} \odot (z_{max} - z_{min})}{||W_{0,i} \odot (z_{max} - z_{min})||}; \quad i = 1, \ldots N_W$$



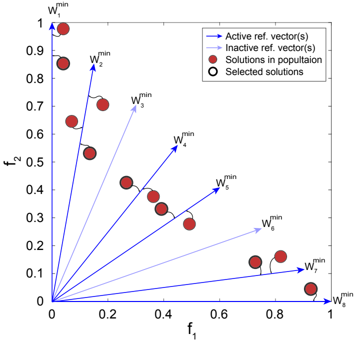
Adaptation of W for a 2 objective problem.

- In each generation, N_W offspring solutions are generated using simulated binary crossover (SBX) and differential evolution (DE) operator with an equal probability.
- For DE, the first parent is from the sorted list of parents and the other two parents are randomly chosen.
- Each offspring solution undergoes polynomial mutation (PM).

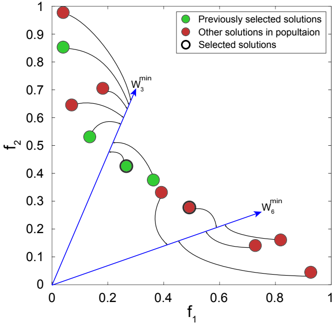


Assignment of solutions to *W*.

Environmental Selection



Selecting solutions from active reference vectors.



Using inactive reference vectors later to select more solutions.

A ND-based constraint handling method is employed here to maintain the solution diversity during the search of feasible region(s). Two possible scenarios can occur:

- **All parent+offspring solutions are infeasible:** The solutions are normalized according to z_{min} and z_{max}. Then, a non-domination (ND) sort is performed taking the CV of the solutions as an objective and the ED of the solutions (calculated with the normalized objective values) to the origin as the second objective. Finally, the population is sorted based on the ranks obtained from this ND sort algorithm.
- **Some solutions are feasible:** If some solutions are feasible in the combined population (number of feasible solutions is less than N_W), they are automatically selected and the rest of the infeasible solutions are sorted according to the ND-based scheme mentioned in the previous step.

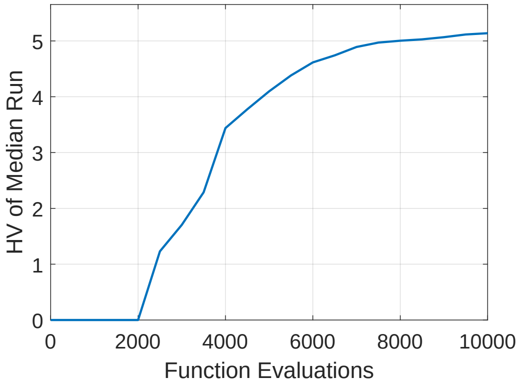
- Number of initial solutions, N_{init}: $11D - 1$.
- Maximum number of function evaluation, FE_{max}: 10,000.
- Uniform spacing on unit hyperplane in NBI method, H: 7.
- Number of reference vectors, N_W: 330.
- Population size, $N = N_W$
- Number of independent runs: 21.
- Crossover (p_c) and mutation probability (p_m): 1.0 and $1/D$.
- Distribution index of crossover (η_c) and mutation (η_m): 30 and 20.
- Crossover probability (CR) and differential weight (F) for DE: 1.0 and 0.5.
- Performance metrics: Hypervolume.

Hypervolume statistics for feasible non-dominated solutions obtained in 21 independent runs are as follows:

Worst	Mean	Best	Median	Std	Success Rate (%)
4.9745	5.1195	5.2525	5.1378	0.0742	100

A success rate of 100% means that all 21 independent runs were able to obtain feasible solutions.

Median Hypervolume Convergence



Hypervolume convergence for the median run.

Thank you for listening!

3rd Evolutionary Computation Competition, December 14, 2019

Algorithm Presentation (s05, m05)

Jernej Zupančič, Aljoša Vodopija, Tea Tušar,
Erik Dovgan, Bogdan Filipič

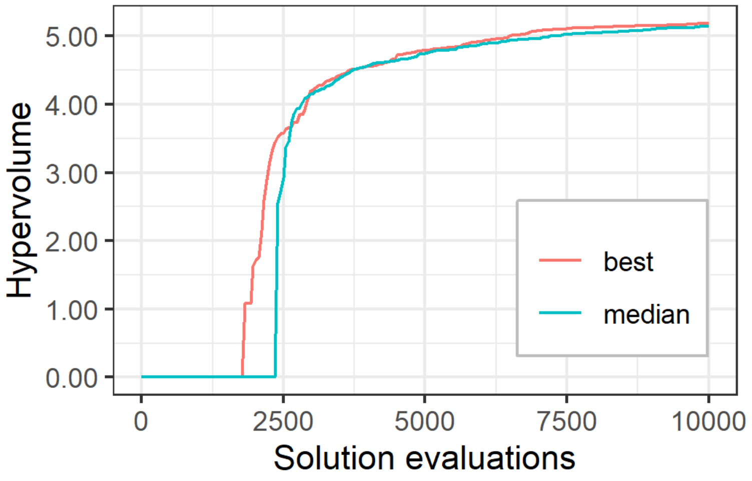
Jožef Stefan Institute (JSI), Ljubljana, Slovenia

Multi-objective optimization algorithm (m05)

- Algorithm: Modified version of NSGA-II capable of including various CHTs (our implementation in Python)
- DoE method: Latin hypercube sampling
- CHT: dynamic penalty function

$$\bar{f}(x) = f(x) + (ct)^{\alpha} \sum_{i} v_i(x)$$

- Parameters:
  - Population size: 48
  - No. of generations: 208
  - Crossover probability: 1.0
  - Mutation probability: 0.15
  - CHT parameters: c = 0.5, α = 2.0



Algorithm overview for EC competition 2019

Hayato Noguchi (Ritsumeikan University)

Tomohiro Harada (Tokyo Metropolitan University)

Overview

- Use $I_{SDE}+$ as evaluation indicator
- Give the constraint processing to conventional $I_{SDE}+$
- Change Simulated Binary crossover (SBX) to Differential Evolution operator (DE) as crossover method
- Exclude similar solutions

$I_{SDE}+$

1. Assign the total fitness of all m objectives to each individual
2. Give maximum $I_{SDE}+$ value to the individual with the minimum fitness
3. Compare each $I_{SDE}+$ values of remaining individuals
4. Take the top N individuals to the next generation

If individual p is feasible …

$$I_{SDE}+(p) = \min_{q \in P_{feasible},\ p \neq q} \{dist(p, q'_1), dist(p, q'_2), \ldots, dist(p, q'_{N_{feasible}-1})\}$$

$$q'(j) = \begin{cases} p(j) & q(j) < p(j) \\ q(j) & \text{otherwise} \end{cases} \quad j \in (1, 2, \ldots, m)$$

Convergence to the optimal Pareto front can be expected while keeping the diversity of the population.

Constraint Processing

- Calculate violations based on constraints

$$violation(p) = \sum_{i=1}^{k} \max\left\{0, -\frac{g_k(p)}{g_k^{max}}\right\} \geq 0$$

(g_k: constraint function)

$$cI_{SDE}+(p) = \begin{cases} I_{SDE}+(p) & \text{(If } p \text{ is feasible)} \\ -violation(p) & \text{(If } p \text{ is infeasible)} \end{cases}$$

The higher $violation(p)$ is,
the less likely the individual p remains in the next generation

Other Improvements

- Replace Simulated Binary crossover (SBX) to Differential Evolution operator (DE) as crossover method

$$p_j^i = \begin{cases} p_j^{r_1} + F \times \left(p_j^{r_2} - p_j^{r_3} \right) & rand_j(0,1) \leq CR \ \vee \ j = j_{rand} \\ p_j^i & otherwise \end{cases}$$

$$i \in (1,2,\ldots,N)$$
$$j \in (1,2,\ldots,Individual\ Size)$$

- Exclude similar solutions

$$\min_{a \in A} \{dist(p,a)\} < eps \quad (A\text{: All evaluated solutions})$$

If many similar solutions exist, the threshold of exclusion eps is reduced gradually

$$eps = 0.5 \times eps$$

Parameters

Population size: 100

Number of generations: 100

Crossover probability (CR): 0.9

Scaling factor (F): 0.5

Mutation probability: 1.0

Threshold of excluding similar solutions: 0.01

Thank you for listening.