

EVOLUTIONARY COMPUTATION AND MULTIOBJECTIVE OPTIMIZATION

Gary G. Yen, *FIEEE, FIET, FIAPR*

gyen@okstate.edu

Regents Professor, Oklahoma State University

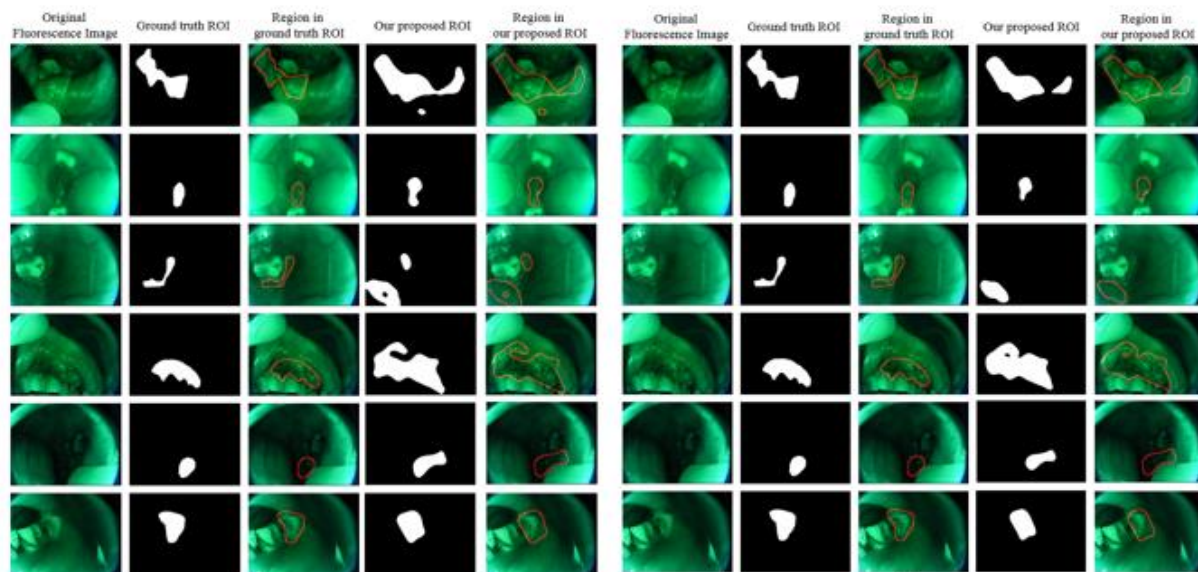
四川大学高端外籍讲座教授



**Summer Course at
Sichuan University, Chengdu, CHINA
Day Six of EIGHT, July 5, 2022**

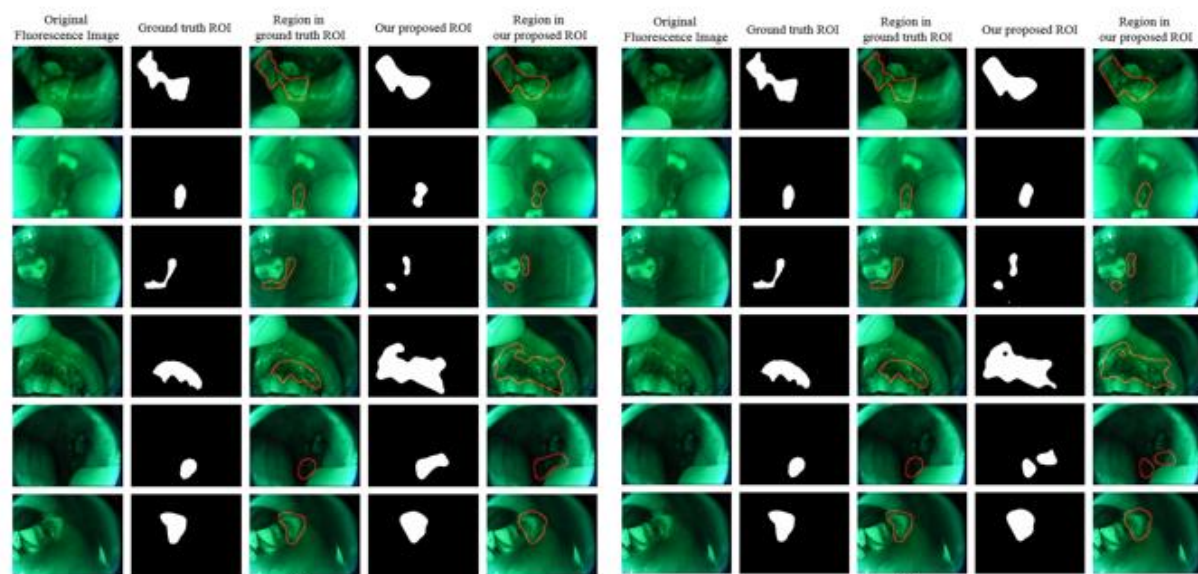
Case Study 6: Oral/Skin Cancer Screening

- **Motivation:** According to NIH, the 5-year survival rate for oral cancer diagnosed early is 75% compared to 20% for diagnosed late. Increased awareness and early intervention to those at greatest risk (i.e., African American men, people who regularly use tobacco products or consume alcohol, individuals subject to high radiation exposure and those infected with oral human papillomavirus) are believed the most effective means to save lives from oral cancer. Unlike tumors of the kidney, prostate, or other internal organs, oral cancers are readily accessible.
- **Approach:** In this study, our long-term objective is exploiting the modern technologies of high-resolution image sensing, AI for machine learning, and cost-effective edge computing as a viable alternative to address this long-lasting threat to personal oral health through early detection. Aided with big data analytics and eXplainable AI on the learnt process, we believe this study will improve medical knowledge and build a data proven instrument for improving outcomes for oral cancer.



(a)

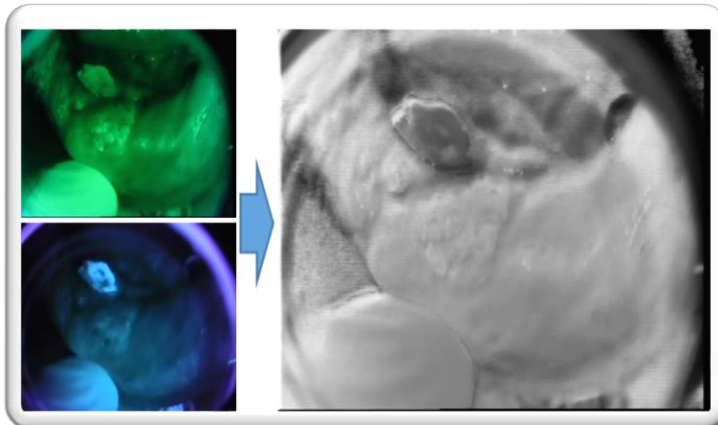
(b)



(c)

(d)

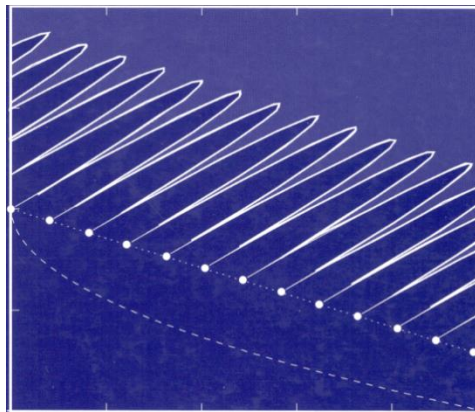
- Automatically designing a *block-growing CNN* given preference on boundary solution with maximizing sensitivity (**eliminating miss detection or false negative**) among specificity and complexity in oral cancer and skin cancer early detection systems



“Evolutionary neural architecture search for automatic esophageal lesion identification,” Zhou Y., Yuan X., Zhang X., Liu W., Wu Y., Yen G.G., Hu B., and Zhang Y., *IEEE Transactions on Artificial Intelligence*, 3(3), 2022. pp. 436-450.

7. MULTIOBJECTIVE OPTIMIZATION EVOLUTIONARY ALGORITHMS- Part II

多目标优化进化算法

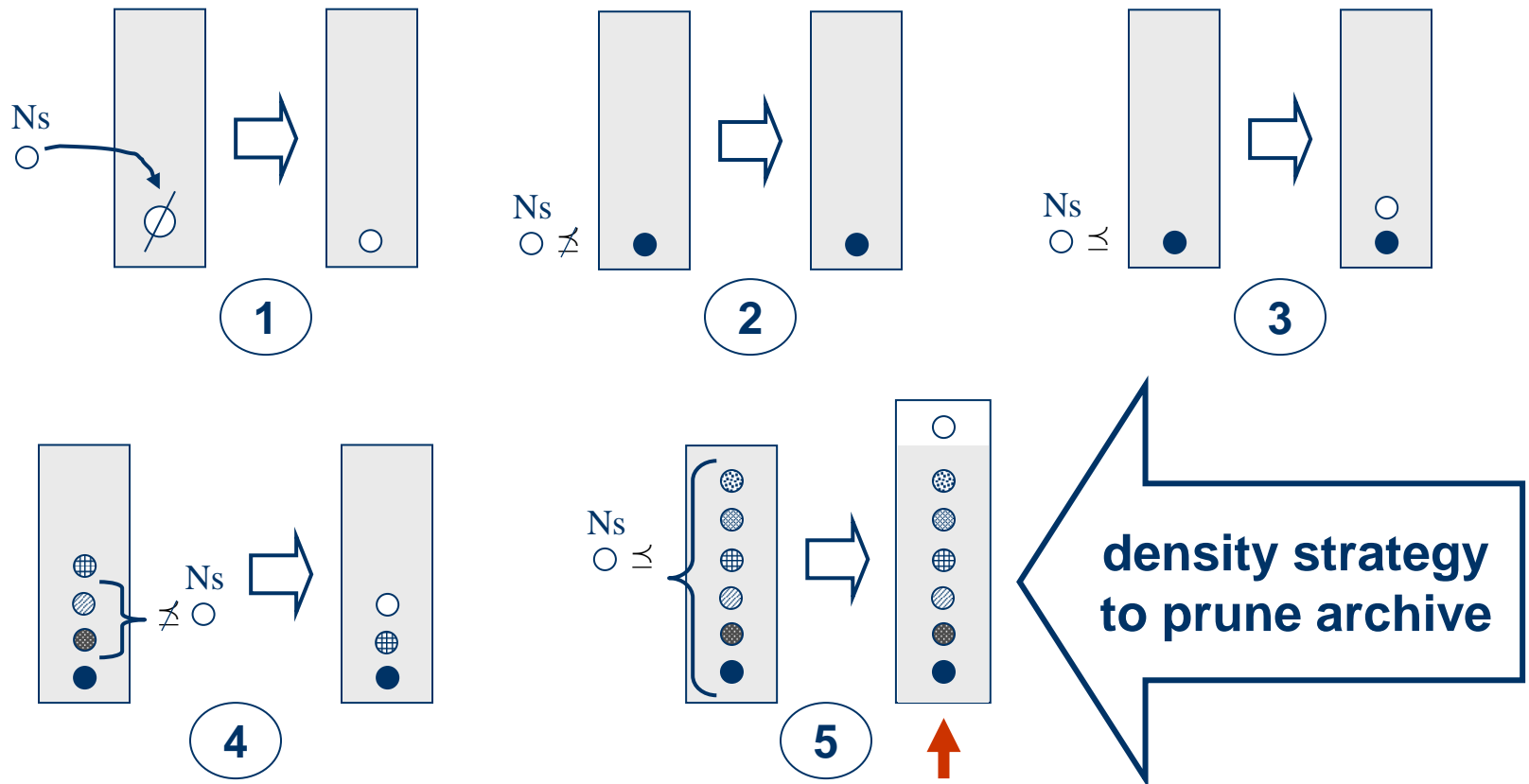


8- Strength Pareto EA (SPEA)

- Algorithm maintains elitism by explicitly maintaining an external population \bar{P}
- This population stores a fixed number of non-dominated solutions.
- At every generation, newly found non-dominated solutions are compared with the existing external population and the resulting non-dominated solutions are preserved
- Elites participate in genetic operations along with the current population in the hope of steering population towards the Pareto front

- The algorithm begins with a randomly created population P_0 of size N and an empty external population P'_0 with a maximum capacity of N'
- In any generation t the best non-dominated solutions (belonging to the first non-dominated front) of the population P_t are copied to the external population P'_t .
- Then the dominated solutions in the modified external population are found and deleted from the external population. \Rightarrow previously found elites which are now dominated by a new elite solution are deleted from the external population.
- Therefore external population contains the best non-dominated solutions of both old and new elites.
- To avoid external population overcrowded, the size of the external population is bounded to a limit, N' . That is, when the size of the external population is less than N' , all elites are kept in the population. When the size exceed N' , not all elites can be accommodated in the external population. \Rightarrow Elites which are less crowded in the non-dominated front are kept. A clustering method is designed.

Archive Maintenance



Exceed allocation Archive size !

- Once the new elites are preserved for the next generation, the algorithm then turns to the current population and uses genetic operators to find a new population.
- In addition to the assigning of fitness to the current population members, fitness is also assigned to external population members. In fact, SPEA assigns a fitness (called the strength) S_i to each member i of the external population first. The strength is proportional to the number (n_i) of current population members that are external solution i dominates

$$S_i = \frac{n_i}{N + 1}$$

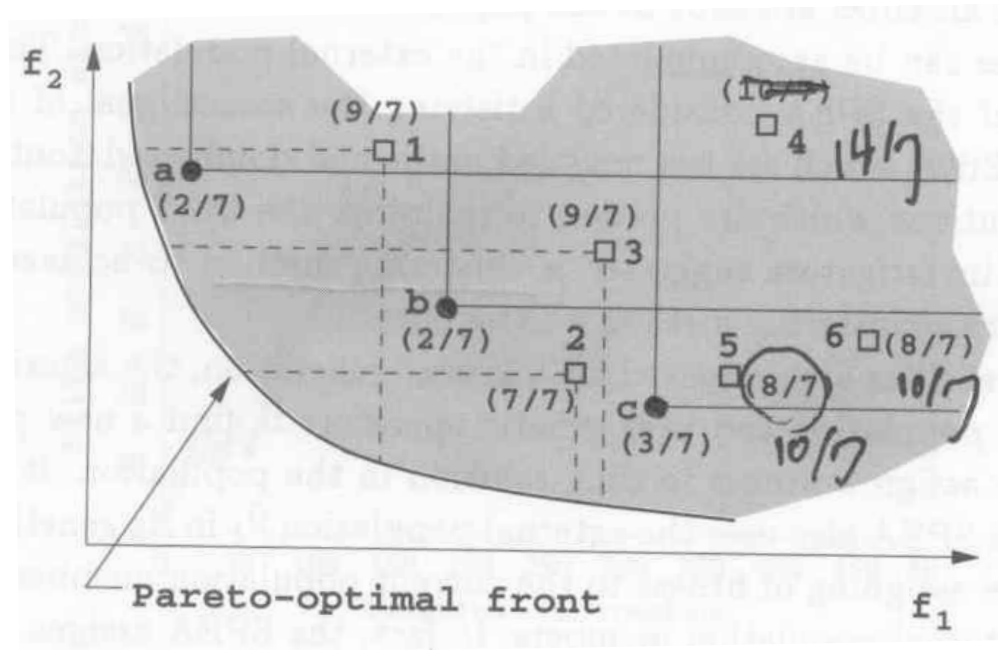
Fitness of the external population
- In other words, SPEA assigns more strength to an elite which dominates more solutions in the current population. Division by ($N+1$) ensures that the maximum value of the strength of any external population members is never one or more. A non-dominated solution dominating a fewer solutions has a smaller (or better) fitness.

- Therefore, the fitness of a current population member j is assigned as one more than the sum of the strength values of all external population members which dominate j

$$F_j = 1 + \sum_{i \in P_t \wedge i \leq j} S_i$$

Fitness of the
main population

- The addition of one makes the fitness of any current population member P_t to be more than the fitness of any external population member P'_t . This method of fitness assignment suggests that a solution with a smaller fitness is better.
- With these fitness values, a binary tournament selection procedure is applied to the combined $(P'_t \cup P_t)$ population to choose solution with smaller fitness values. As usual, crossover and mutation operators are applied to the mating pool, and a new population P_{t+1} of size N is created.
- Computational complexity: $O(MN^2)$



SPEA Pseudo Codes

Step 1: Find the best non-dominated set of the current population P_t (i.e., $F_1(P_t)$) and copy it to the external population P'_t , or perform $P'_t = P'_t \cup F_1(P_t)$

Step 2: Find the best non-dominated solutions of the modified external population P'_t (i.e., $F_1(P'_t)$) and delete all dominated solutions from the external population (i.e., $P'_t = F_1(P'_t)$)

Step 3: If $|P'_t| > N'$, use a clustering technique to reduce the population size to N' . Otherwise, keep P'_t unchanged. The resulting population is the external population P'_{t+1} of the next generation.

Step 4: Assign fitness to each elite solution $i \in P'_{t+1}$ by using $S_i = \frac{n_i}{N+1}$ and for the current population using $F_j = 1 + \sum_{i \in P'_{t+1} \wedge i \leq j} S_i$

Step 5: Apply a binary tournament selection with these fitness values, a crossover and a mutation operator to create the new population P_{t+1} of size N from the combined population $(P'_{t+1} \cup P_t)$ of size $(N' + N)$.

Clustering Algorithm

Step C1: Initially, each solution belongs to a distinct cluster or $C_i = \{i\}$, so that

$$C = \{C_1, C_2, \dots, C_{N'}\}$$

Step C2: If $|C| \leq N'$, go to Step 5, otherwise go to Step 3

Step C3: For each pair of clusters, calculate the cluster distance by using

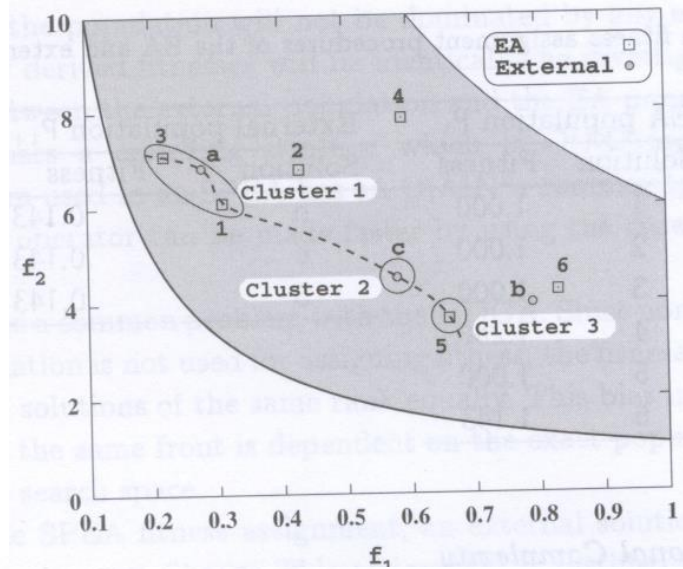
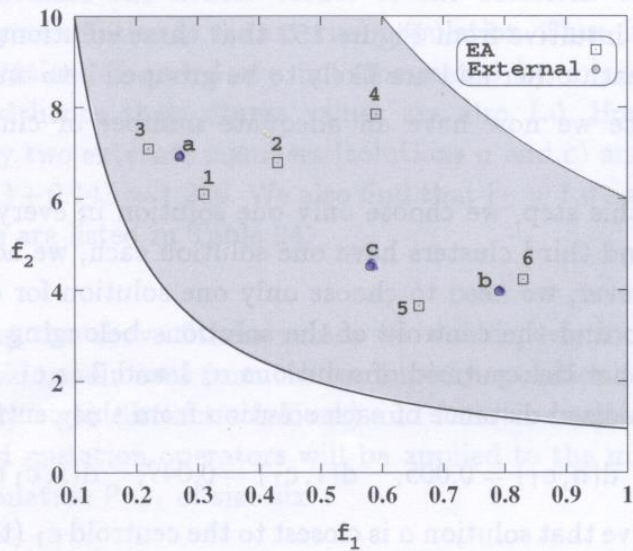
$$d_{12} = \frac{1}{|C_1||C_2|} \sum_{i \in C_1, j \in C_2} d(i, j)$$

where $d(i, j)$ is the Euclidian distance between the points i, j and find the pair (i_1, i_2) which corresponds to the minimum cluster distance.

Step C4: Merge the two clusters C_1 and C_2 together. This reduces the size of C by one, go to step C2.

Step C5: Choose only one solution from each cluster and remove the others from the clusters. The solution having the minimum average distance from other solutions in the cluster can be chosen as the representative solution of a cluster.

SPEA Example



EA population P_t					External population \bar{P}_t				
Solution	x_1	x_2	f_1	f_2	Solution	x_1	x_2	f_1	f_2
1	0.31	0.89	0.31	6.10	a	0.27	0.87	0.27	6.93
2	0.43	1.92	0.43	6.79	b	0.79	2.14	0.79	3.97
3	0.22	0.56	0.22	7.09	c	0.58	1.62	0.58	4.52
4	0.59	3.63	0.59	7.85					
5	0.66	1.41	0.66	3.65					
6	0.83	2.51	0.83	4.23					

- $P_t = \{1, 2, 3, 4, 5, 6\}$ and $P'_t = \{a, b, c\}$, $N=6$, $N'=3$.
- Step 1: Find the non-dominated solutions of P_t , $F_1(P_t) = \{1, 3, 5\}$.
 $P'_t = \{a, b, c, 1, 3, 5\}$
- Step 2: Find the non-dominated solutions of this modified population, $F_1(P'_t) = \{a, c, 1, 3, 5\}$. This is the new external population.
- Step 3: Since the size of P'_t is 5, which is greater than the allowed space $N' = 3$, we need to use the clustering algorithm to find which three will remain in the external population.
 - Step C1: Initially, all five solutions belong to a separate cluster.
 - Step C2: Since there are 5 clusters, we move to Step C3.
 - Step C3: We use $f_1^{\max}=1$, $f_1^{\min}=0.1$, $f_2^{\max}=60$, $f_2^{\min}=1$. and find $d_{12}=0.35$, $d_{13}=0.05$, $d_{14}=0.06$, $d_{15}=0.44$, $d_{23}=0.30$, $d_{24}=0.40$, $d_{25}=0.09$, $d_{34}=0.09$, $d_{35}=0.39$, $d_{45}=0.49$
 we observe that minimum cluster distance occurs between the first and the third clusters.

- Step C4: we merge these cluster together and have only four clusters.
- Step C2: Since there are four clusters, we move to Step C3 to reduce one more cluster.
- Step C3: Now average distance between the first and second clusters is the average distance between the two pairs of solutions (a, c) and (1,c). The distance between a and c is 0.35 and that between solutions 1 and c is 0.30. Thus, the average distance $d_{12}=0.325$. Similarly, we find $d_{12}=0.325$, $d_{13}=0.075$, $d_{14}=0.415$, $d_{23}=0.400$, $d_{24}=0.090$, and $d_{34}=0.490$. The minimum distance occurs between clusters 1 and 3.
- Step C4: The we merge clusters 1 and 3 and have the following three clusters: $C_1=\{a, 1, 3\}$, $C_2=\{c\}$, and $C_3=\{5\}$
- Step C5: We choose only one solution in every cluster. Since the second and third clusters have only one solution each, we accept them as they are. However, we need to choose only one solution for cluster 1. The first step is to find the centroid of the solutions belong to the cluster, $c_1=(0.27, 6.71)$. Now the normalized distance of each solution from this centroid is as follows: $d(a,c_1)=0.005$, $d(1,c_1)=0.049$, and $d(3,c_1)=0.052$.

- We observe that solution a is closest to the centroid. Thus we choose solution a and delete solutions 1 and 3 from this cluster. There the new external population is $P_{t+1}' = \{a, c, 5\}$.

- Step 4: We assign fitness values to the solutions of populations P_t and P_{t+1}' . First we concentrate on the external population. We observe that solution a dominates only one solution (solution 4) in P_t . Thus, its fitness is assigned as $F_a = 1/(6+1) = 0.143$ ($n_a = 1$). Similarly, we find $n_c = n_5 = 1$, and their fitness values are also $F_c = F_5 = 0.143$.
- Next we calculate the fitness values of the solutions of P_t . Solution 1 is dominated by no solution in the external population. Thus, its fitness is $F_1 = 1.0$. Similarly, solutions 2 and 3 are not dominated by any external population members and hence their fitness values are also 1.0. However, solution 4 is dominated by two external members (solutions a and c) and hence its fitness is $F_4 = 1 + 0.143 + 0.143 = 1.286$. We also find that $F_5 = 1.0$ and $F_6 = 1.143$.

- Step 5: Now, using the above fitness values we would perform six tournament s by randomly picking solutions from the combined populations of size nine and form the mating pool. Thereafter, crossover and mutation operators will be applied to the mating pool to create the new population P_{t+1} of size six.

9- Advanced Strength Pareto EA

- Areas for improvements:
 1. *Improved fitness assignment scheme*, which takes each individual into account how many individuals it dominates and it is dominated by.
 2. A *nearest neighbor density technique* is incorporated which allows a more precise guidance of the search process.
 3. New *archive truncation method* guarantees the preservation of boundary solutions.

Fitness Assignment

- To avoid individuals dominated by the same archive members having identical fitness values, both dominating and dominated solutions are taken into account.
- In detail, each individual i in the archive P_t' and the population P_t is assigned a strength value $S(i)$, representing the number of solutions it dominates

$$S(i) = \left| \{ j \mid j \in P_t + \overline{P_t'} \wedge i \succ j \} \right|$$

where $|\cdot|$ denotes the cardinality of a set, $+$ stands for multiset union and the symbol \succ corresponds to the Pareto dominance relation.

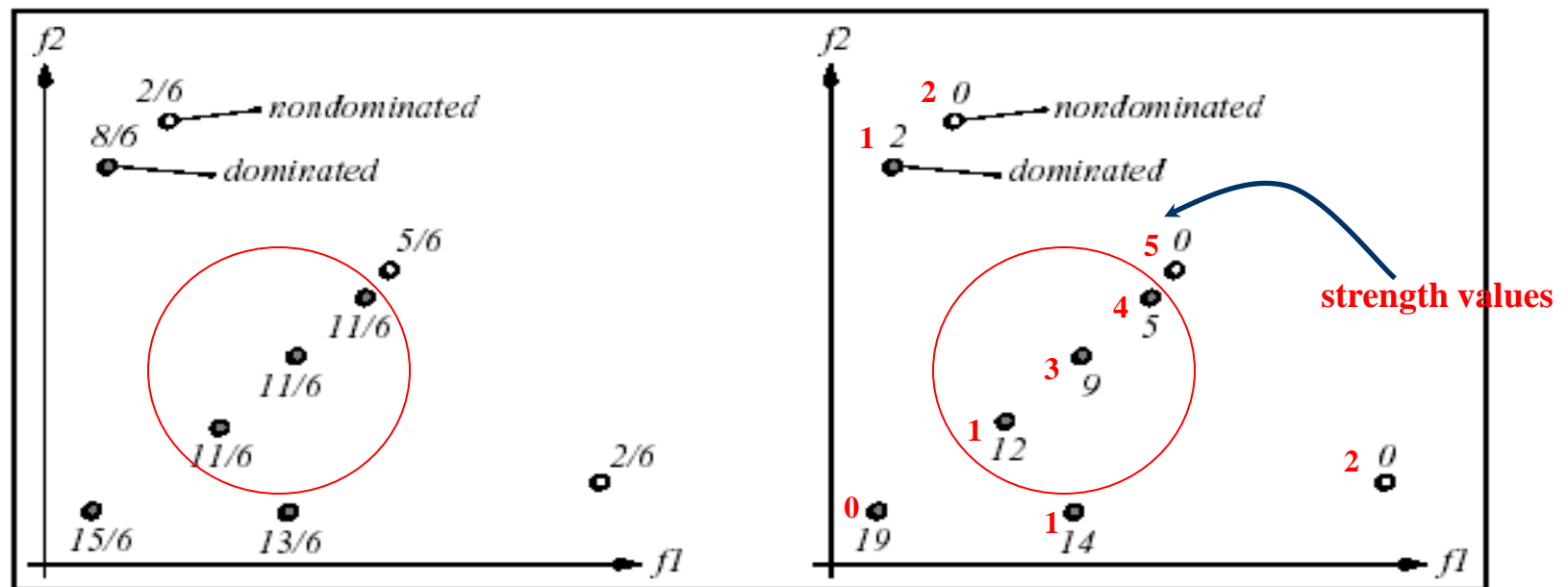
SPEA2 Design Procedure

- On the basis of the S values, the raw fitness $R(i)$ of an individual i is calculated as

$$R(i) = \sum_{j \in P_t + \bar{P}_t, j \succ i} S(j)$$

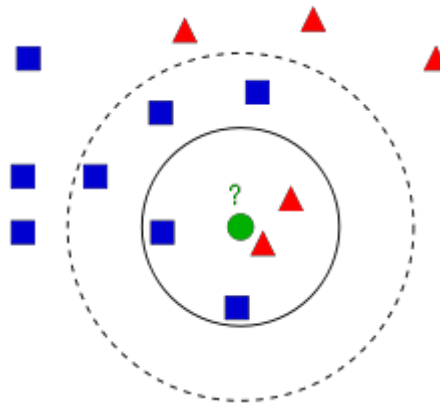
- Fitness is determined by the strengths of its dominators in both archive and population, as opposed to SPEA where only archive members are considered in this context.
- Fitness is to be minimized here, i.e., $R(i) = 0$ corresponds to a non-dominated individual, while a high $R(i)$ value means that i is dominated by many individuals (which in turn dominate many individuals).
- SPEA2 is available at www.tik.ee.ethz.ch/pisa

SPEA vs. SPEA2 (maximization problem)



On the left, the fitness values for a given population according to the SPEA Scheme. On the right, the raw SPEA2 fitness values for the same population are depicted. No individual dominated by the same archive members will have the identical fitness values.

kNN Classification



- Nearest Neighbor- green circle belong to the class of red triangle
- $k=3$ - green circle belong to the class of red triangle
- $k=7$ - green circle belong to the class of blue square
- Modified kNN version with distance weighting

Nearest Neighbor Density Technique

- Additional density information is incorporated to discriminate between individuals having identical raw fitness values. The density estimation technique used is an adaptation of the k -th nearest neighbor method (Silverman 1986), where the density at any point is a (decreasing) function of the distance to the k -th nearest data point. Here, we simply take the inverse of the distance to the k -th nearest neighbor as the density estimate.
- For each individual i the distances (in objective space) to all individuals j in archive and population are calculated and stored in a list in increasing order, the k -th element gives the distance sought, denoted as σ_i^k and commonly $k = \sqrt{N + \bar{N}}$

The density $D(i)$ corresponding to i is defined by $D(i) = \frac{1}{\sigma_i^k + 2}$
 $0 < D(i) < 1$

Fitness of individual i is given by $F(i) = R(i) + D(i)$

Archive Update

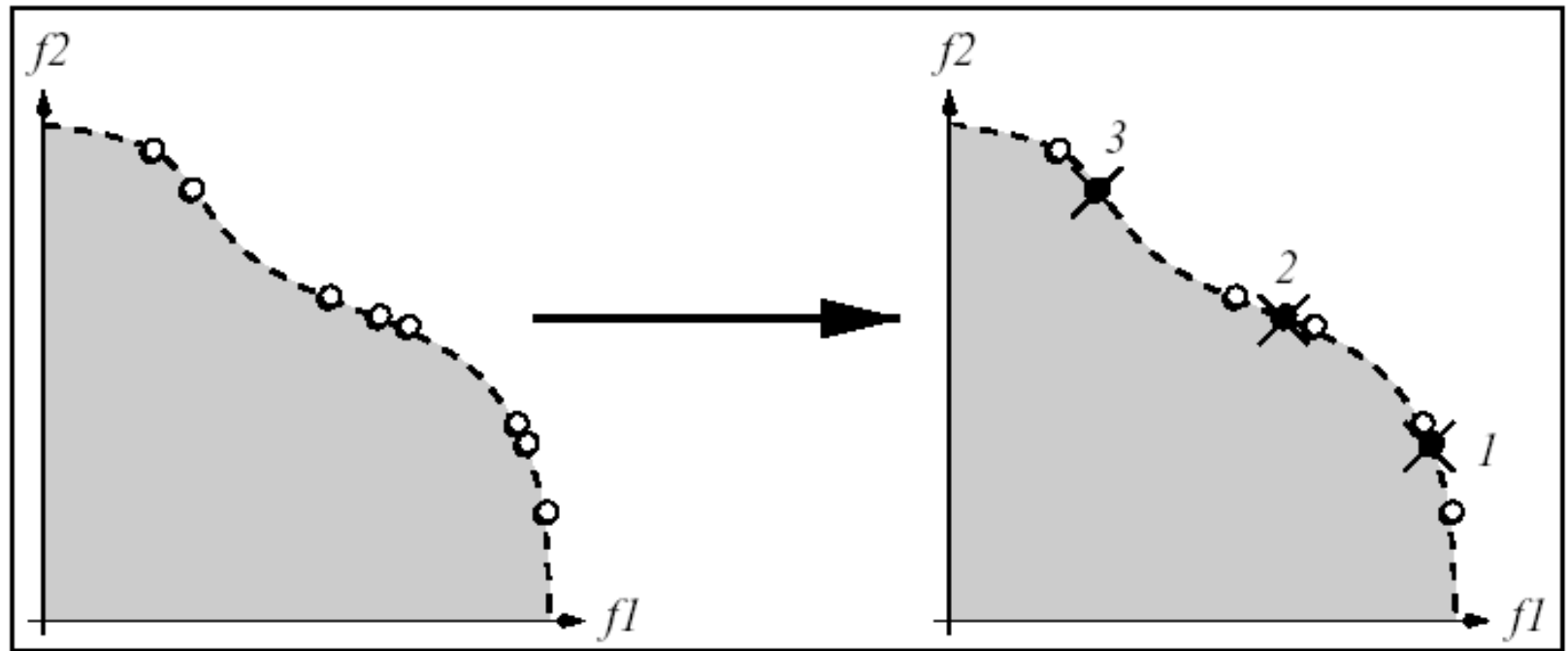
- The archive update operation in SPEA2 differs from the one in SPEA in two respects:
 - the number of individuals contained in the archive is constant over time
 - the truncation method prevents boundary solutions being removed.

Archive Truncation

- When the size of the current non-dominated set needs to be reduced, the archive truncation procedure is used, where at each iteration that individual i is chosen for removal for which for all $j \in \bar{P}_{t+1}$ with $i \leq_d j$

$$i \leq_d j \quad \Leftrightarrow \quad \forall 0 < k < |\bar{P}_{t+1}| : \sigma_i^k = \sigma_j^k \vee \\ \exists 0 < k < |\bar{P}_{t+1}| : \left[\left(\forall 0 < l < k : \sigma_i^l = \sigma_j^l \right) \wedge \sigma_i^k < \sigma_j^k \right]$$

where σ_i^k denotes the distance of i to its k th nearest neighbor in $|\bar{P}_{t+1}|$, this means the individual which has the minimum distance to another individual is chosen at each stage; if there are several individuals with minimum distance the tie is broken by considering the second smallest distances and so forth.



On the left, a non-dominated set is shown. On the right, it is depicted which solutions are removed in which order by the truncate operator (assuming that $N^* = 5$).

SPEA2 Pseudo Codes

Step 1: *Initialization*: Generate an initial population P_0 and create the empty archive (external set) $P_0' = \emptyset$; Set $t = 0$.

Step 2: *Fitness assignment*: Calculate fitness values of individuals in P_t and P_t'

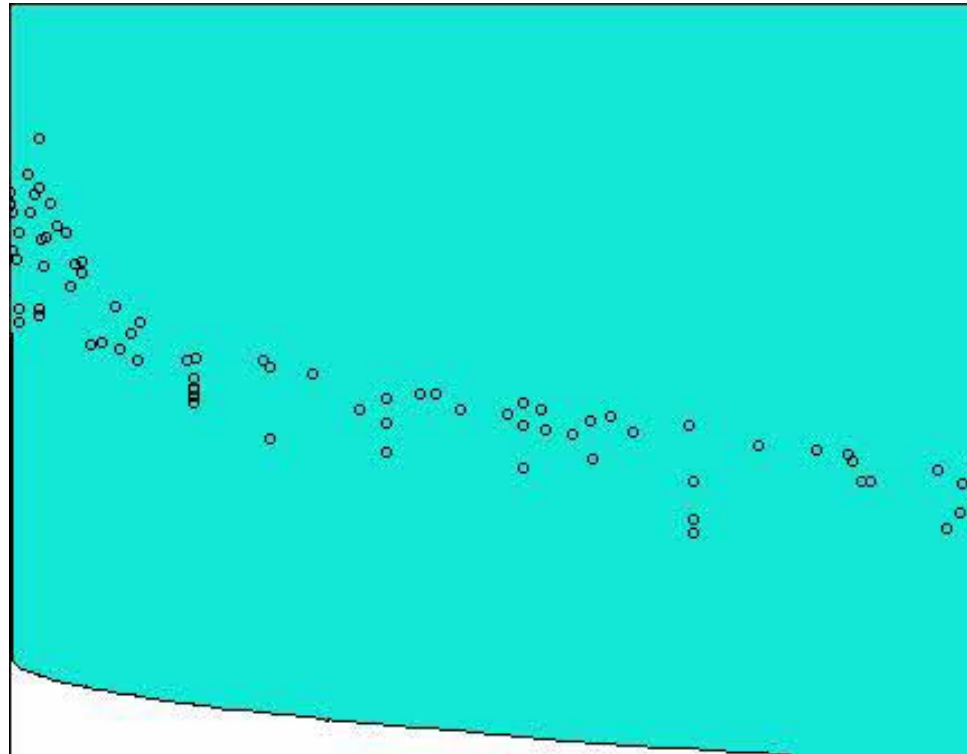
Step 3: *Environmental selection*: Copy all non-dominated individuals in P_t and P_t' to P_{t+1}' . If size of P_{t+1}' exceeds N' then reduce P_{t+1}' by means of the truncation operator, otherwise if size of P_{t+1}' is less than N' then fill P_{t+1}' with dominated individuals in P_t and P_t' .

Step 4: *Termination*: If $t < T$ or another stopping criterion is satisfied then set A to the set of decision vectors represented by the non-dominated individuals in P_{t+1}' . Stop.

Step 5: *Mating selection*: Perform binary tournament selection with replacement on P_{t+1}' in order to fill the mating pool.

Step 6: *Variation*: Apply recombination and mutation operators to the mating pool and set P_{t+1} to the resulting population. Increment generation counter ($t = t + 1$) and go to Step 2.

SPEA2 for ZDT1



SPEA2 Advantages & Disadvantages

- Advantages:
 - Once a solution in the Pareto-optimal front is found it gets stored in the external population.
 - Clustering ensures that a better spread is achieved among the obtained non-dominated solutions.
 - Clustering algorithm does not have any special parameter to be specified.
- Disadvantages:
 - There is an extra parameter N' , the size of the external population and the balance between the regular population size N and external population size N' is important in the success of SPEA as it directly affects the importance of elitism.
 - As in MOGA the fitness values do not favor all non-dominated solutions of the same rank equally.

10- Dynamic Population MOEA

- Goal: to exploit and avoid “genetic drift” phenomenon simultaneously
- If the predetermined population size is *too small*, there will not be enough schemas to be exploited, resulting into a non-uniformly distributed Pareto front. If the population size is *too large*, it may require unnecessarily large computational resources and result into an extremely long running time
- A fixed population size will have great difficulty in obtaining a Pareto front with a desired resolution because the size and shape of the true Pareto front is unknown *a priori* for most of the MOPs
- In most MOEAs, a “guessed” population size is used and a replacement method is usually designed to keep the population size
- A dynamic population size will be more reasonable for MOEA if the computational effort can be adaptively adjusted based on the complexity of the problem

“Dynamic multiobjective evolutionary algorithm: adaptive cell-based rank and density estimation,” Yen G.G. and Lu H., *IEEE Transactions on Evolutionary Computations*, 7(3), 2003, pp. 253-274.

Cost Benefit

- optimal population size depends on the complexity of the problem
- automatic stopping criteria
- time/computational complexity
- biological evidence

Cell-Based Rank & Density Calculation

- An adaptive grid density estimation approach is used. The length of adaptive grid cell of the i 'th dimension in objective space is computed as

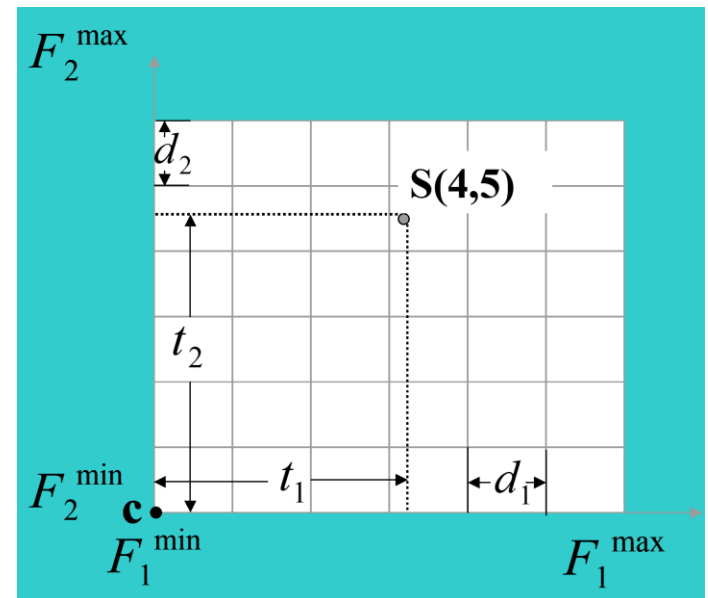
$$d_i = \frac{\max_{\mathbf{x} \in X} f_i(\mathbf{x}) - \min_{\mathbf{x} \in X} f_i(\mathbf{x})}{K_i}$$

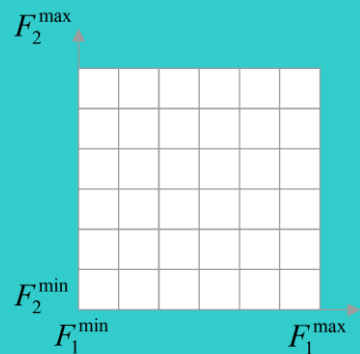
$$i = 1, \dots, n$$

$$t_i = s_i - F_i^{\min}$$

- Home address of individual S

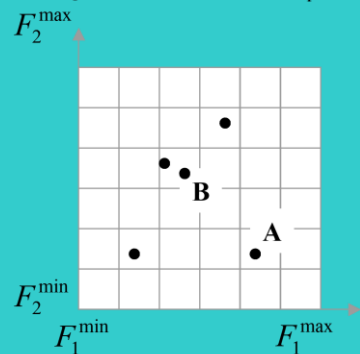
$$h_i = \text{mod}(t_i, d_i) + 1$$





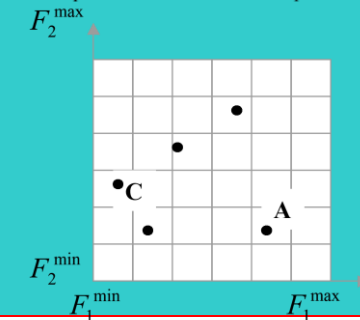
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1
1	1	1	1	1	1

0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0



1	2	4	5	6	6
1	2	4	4	6	6
1	2	2	4	5	5
1	2	2	2	3	3
1	1	2	2	2	3
1	1	1	1	1	1

0	0	0	0	0	0
0	0	0	1	0	0
0	0	2	0	0	0
0	0	0	0	0	0
0	1	0	0	1	0
0	0	0	0	0	0

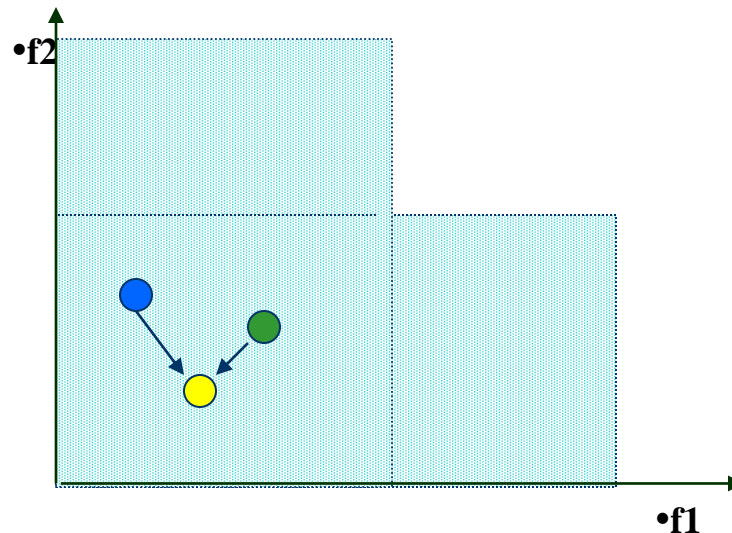


2	3	4	5	6	6
2	3	4	4	6	6
2	3	3	4	5	5
1	3	3	3	4	4
1	1	2	2	2	3
1	1	1	1	1	1

0	0	0	0	0	0
0	0	0	1	0	0
0	0	1	0	1	0
1	0	0	0	0	0
0	1	0	0	1	0
0	0	0	0	0	0

Population Growing Strategy

- Goal: to focus on pure population increment to ensure that each individual survives enough generations so that it can contribute its valuable schemas.



Population Declining Strategy

- Goal: to prevent the population size from growing unboundedly for an individual c at generation n , we introduce:

- Health indicator

$$H(c, n) = \frac{1}{\text{rank}(c, n)}$$

- Global density indicator

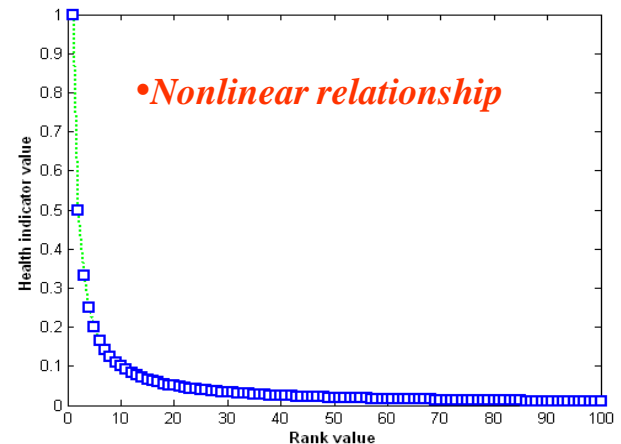
$$DG(n) = \frac{D_{avg}}{ppv}$$

- Local density indicator

$$DL(c, n) = \frac{\text{density}(c, n)}{ppv}, \text{ if } \text{density}(c, n) > ppv$$

0, otherwise

where, ppv denotes the desired number of individuals in each cell and D_{avg} is the average density value of the current population



- Age indicator

$$A(y, n) = \frac{\text{age}(y, n) - A_{th}}{n}, \text{ where } A_{th} \text{ denotes the predetermined threshold of the age}$$

- Probabilities of removing an individual

- Likelihood of removing the **most unhealthy** individuals

$$l_1^i = (1 - H(c_i, n))^2 \times A(y_i, n)$$

$$H(c_i, n) = 1 / r_{\max}$$

- Likelihood of removing the **unhealthy** individuals in the **most crowded** cells

$$l_2^i = (1 - H(c_i, n))^2 \times DG(n) \times (DL(c_i, n) - 1) \times A(y_i, n)$$

- Likelihood of removing a **nondominated** individual from the **most crowded** cells *after the entire population converges*

$$l_3^i = (DL(c_i, n) - 1) \times DG(n) \times A(y_i, n)$$

- To determine whether an individual y_i will be eliminated, three random numbers between [0, 1] are generated to compare with the concerned likelihoods according to the situation of the given individual.

Observations

- Because the age indicator $A(y_i, n)$ influences all of three likelihoods, l_1 , l_2 and l_3 will be 0 if the age of the concerned individual is less than the age threshold A_{th} . This implies that if an individual is not old enough, it will not be eliminated from the population no matter how high its rank and density values are.
- At each generation, DMOEA will remove those *most unhealthy* individuals according to likelihood l_1 , based on their rank values and ages. Assume the age indicator of an individual y is $A(y, n)=1$, the relationship between its rank value and l_1 value is an exponentially growing function. Without considering the effects of other indicators, when an *unhealthy* individual in the set has a very high rank value, it will have a very high likelihood (l_1) to be eliminated, since it is too far away from the current Pareto front. Moreover, as r_{max} drops and gets closer to 1, l_1 will decrease very fast, and the concerned individual will not be removed easily because it is very likely to be evolved into an elitist in the future. Therefore, this “shell removing” strategy will keep eliminating the individuals located on the outside layer with an adaptive probability until the entire population converges into a non-dominated set.

- Because all the individuals in the same cell share the fixed computation resource (or “living resource”), the individuals located in a crowded cell have to compete much harder for the limited resource than those located in a sparse cell. Therefore, another elimination scheme based on a *crowdedness* indicator is designed in DMOEA in order to remove some *unhealthy* individuals that stay in the most crowded areas. From I_2 likelihood Equation, at each generation, if an individual belongs to the set Y_{dr} it will have the likelihood of I_2 to be eliminated based on its *age*, *health*, and local rank value and density condition. From this scheme, the population tends to be distributed homogeneously by eliminating the redundant individuals.
- After every individual has converged into a Pareto point, another elimination scheme is implemented based on I_3 values. Therefore, the resulting trade-off hyperareas $A_{to}(n)$ are counted, and the final population is truncated to ensure that each cell contains *ppv* number of individuals; thus the optimal population size can be calculated by

$$dps(n) = ppv \times A_{to}(n)$$

Objective Space Compression Strategy

Goal— to ensure the precise of resulting Pareto front. Three criteria need to be met:

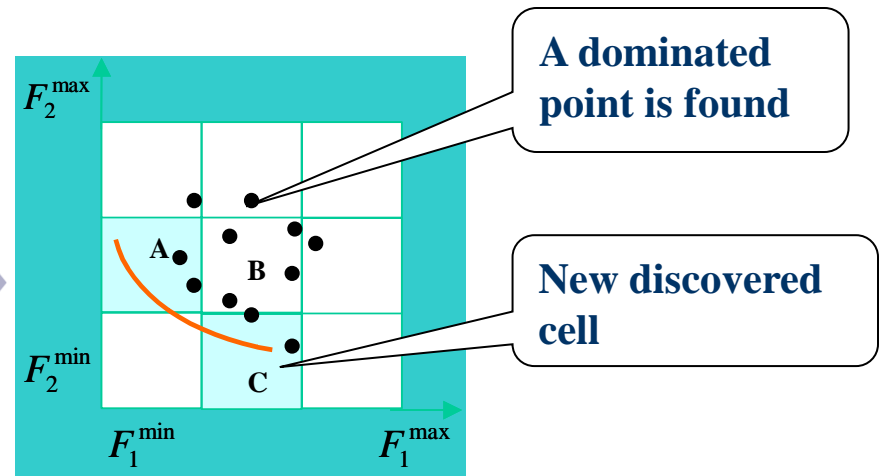
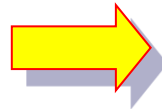
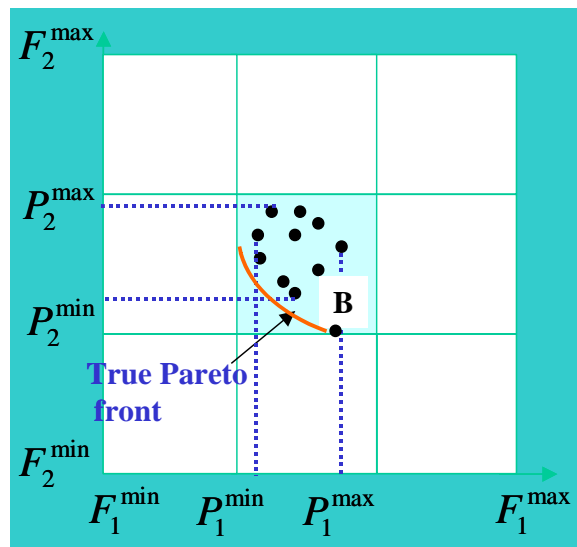
- maximum cell rank value of all the individuals is 1-- all discovered cells are non-dominated;
- $(F_i^{\max} - P_i^{\max}) > 0.1(F_i^{\max} - F_i^{\min})$ or $(P_i^{\min} - F_i^{\min}) > 0.1(F_i^{\max} - F_i^{\min})$ -- there is still some room for compression;
- minimum age value of all the individuals is greater than the predefined age threshold A_{th} -- no new generated individuals since most of the resulting points are Pareto optimal;

Therefore, new boundaries of the objective space will be

$$F_i^{\max} = (P_i^{\max} + F_i^{\max}) / 2$$

$$F_i^{\min} = (P_i^{\min} + F_i^{\min}) / 2$$

A Graphical Illustration



Stopping Criteria

- The rank values of all cells are 1s
- The objective space cannot be compressed anymore
- Each resulting non-dominated cell contains ppv individuals
- Afterwards, DMOEA is refined by using the traditional Pareto ranking scheme instead of cell-based ranking to ensure all the resulting individuals are Pareto points

F-4: High Dimensional Decision Space

- Deb (2001)

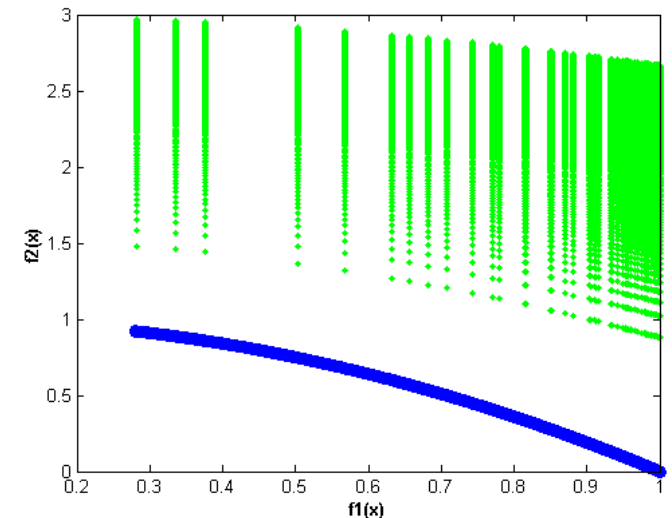
Minimize:

$$f_1(x) = 1 - e^{-4x_1} \sin^6(6\pi x_1)$$

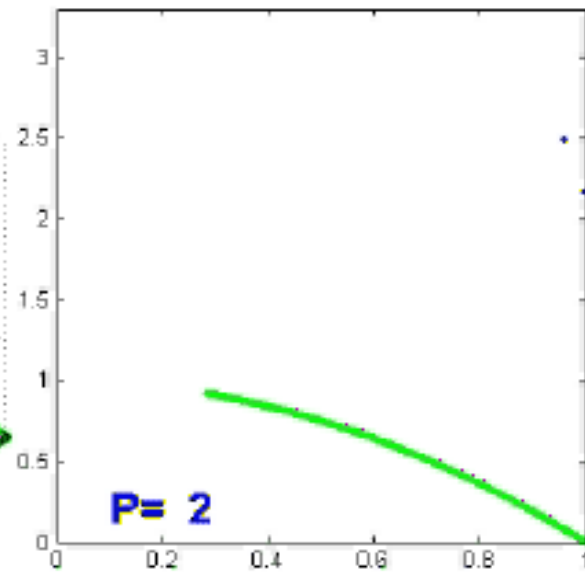
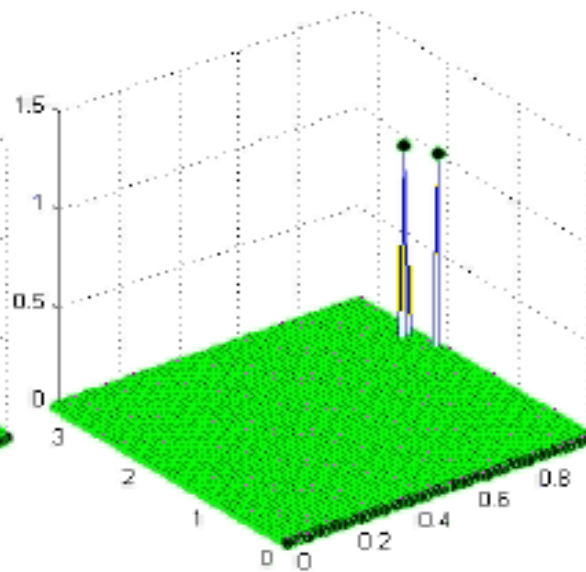
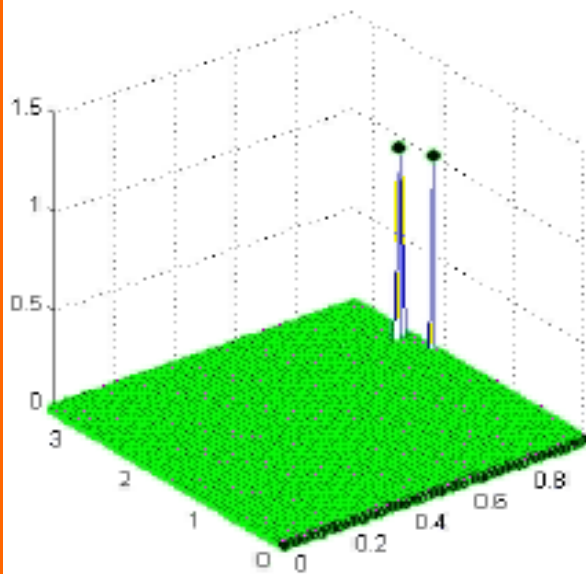
$$f_2(x) = g \left(1 - \left(\frac{f_1(x)}{g} \right)^2 \right)$$

Subject to

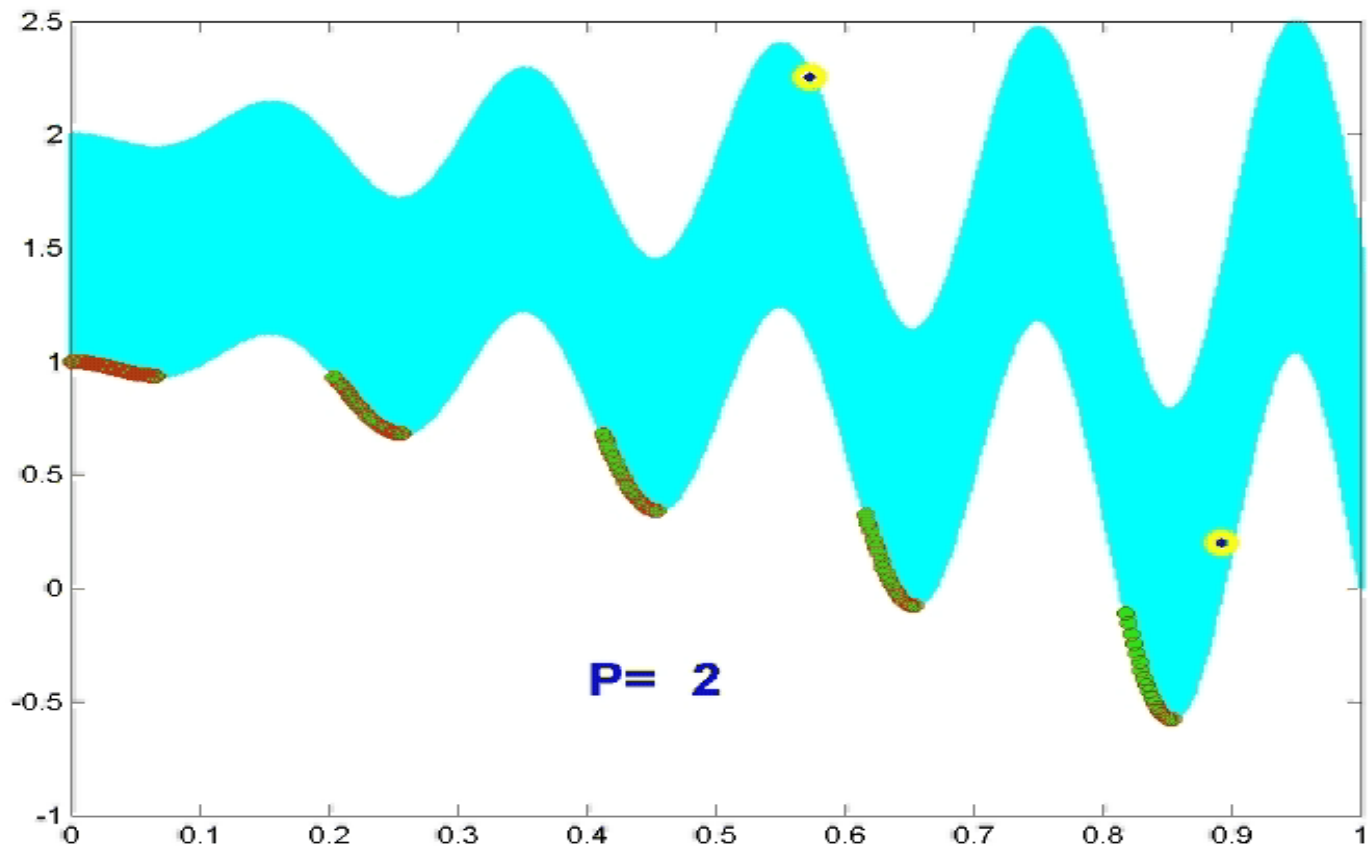
$$g(x) = 1 + 4 \left(\sum_{i=2}^5 x_i / 4 \right)^{0.25}, 0 \leq x_i \leq 1, i = 1, \dots, 5$$



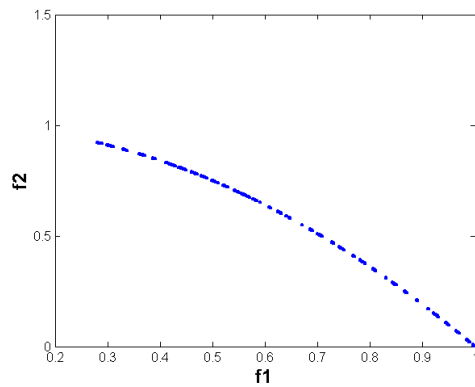
Demonstration: DMOEA for $F-4$



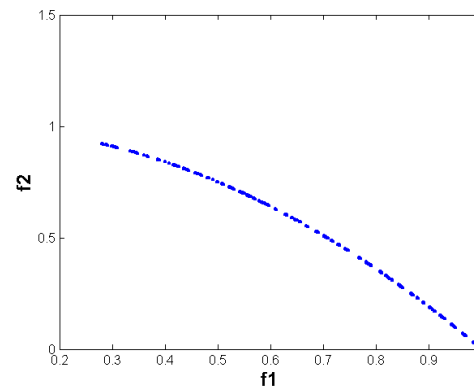
Demonstration: DMOEA



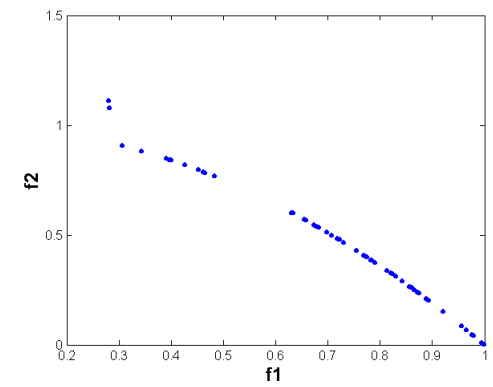
F-4: Resulted Pareto Fronts



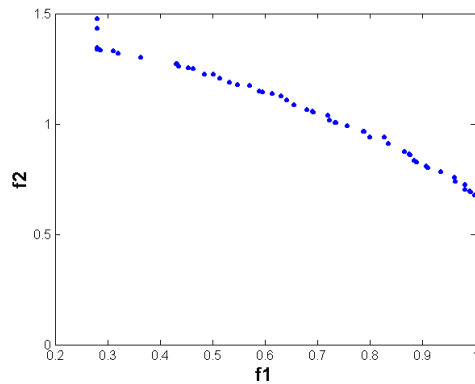
IMOE



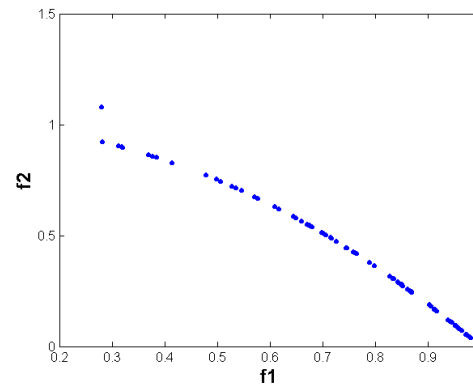
DMOEA



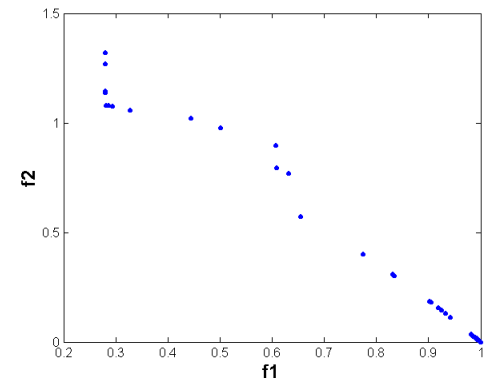
NSGA-II



PAES

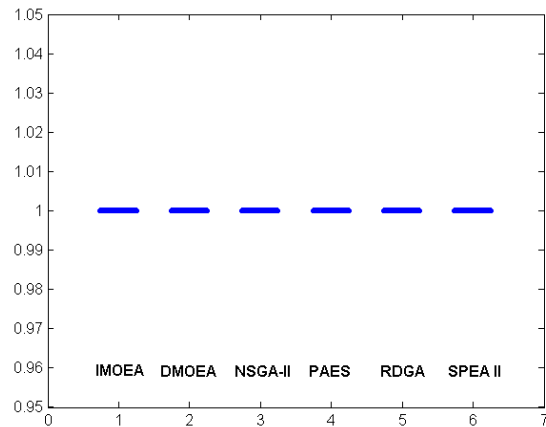


RDGA



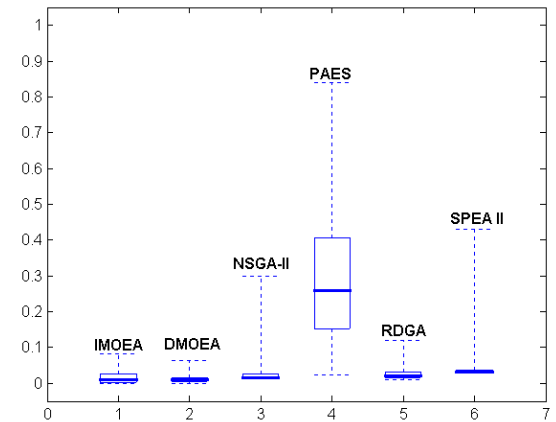
SPEA II

F-4: Rank, Density & Distance Measures

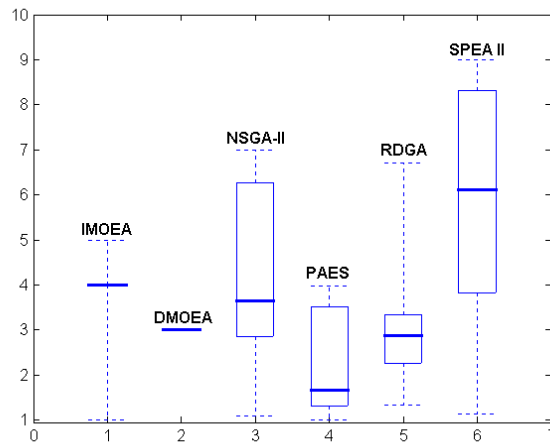


Rank

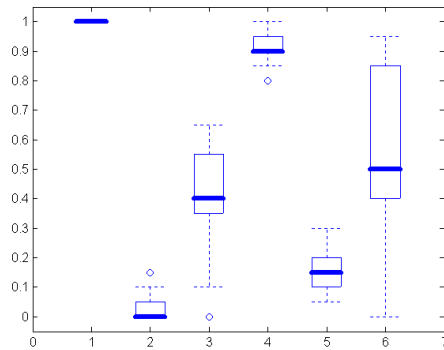
Density



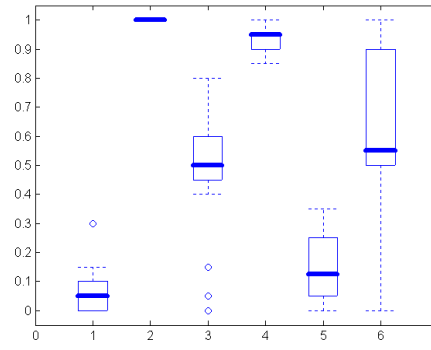
Distance



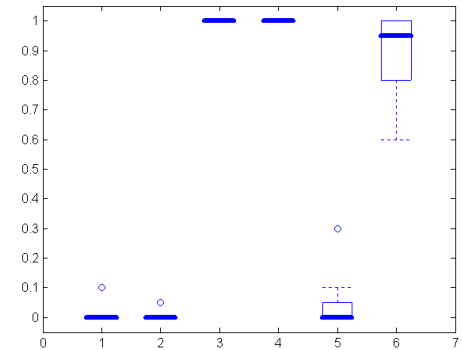
F-4: C Measure



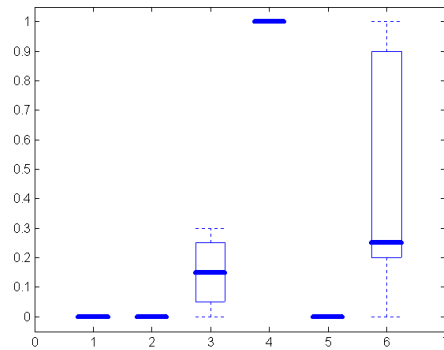
$C(X_1, X_{1-6})$



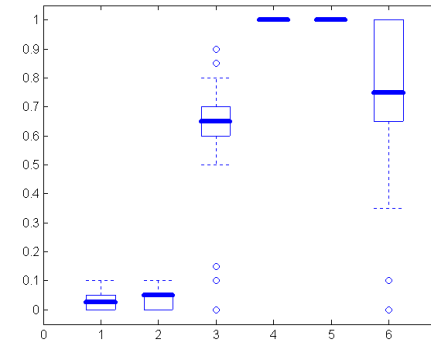
$C(X_2, X_{1-6})$



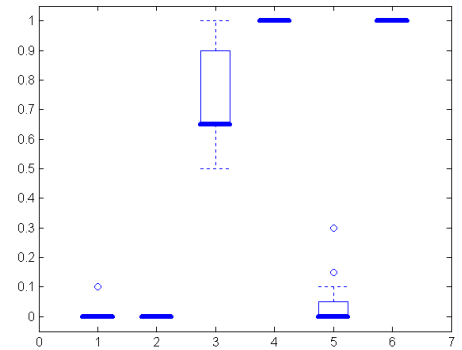
$C(X_3, X_{1-6})$



$C(X_4, X_{1-6})$



$C(X_5, X_{1-6})$



$C(X_6, X_{1-6})$

1.IMOEA; 2.DMOEA; 3.NSGA-II; 4.PAES; 5.RDGA; 6.SPEA II

F-6: High-dim Objective/Decision Spaces

- Deb (2002), one global and many local Pareto fronts.
- Minimize

$$f_1(x) = (1 + g(\mathbf{x})) \cos\left(\frac{\pi x_1}{2}\right) \cos\left(\frac{\pi x_2}{2}\right)$$

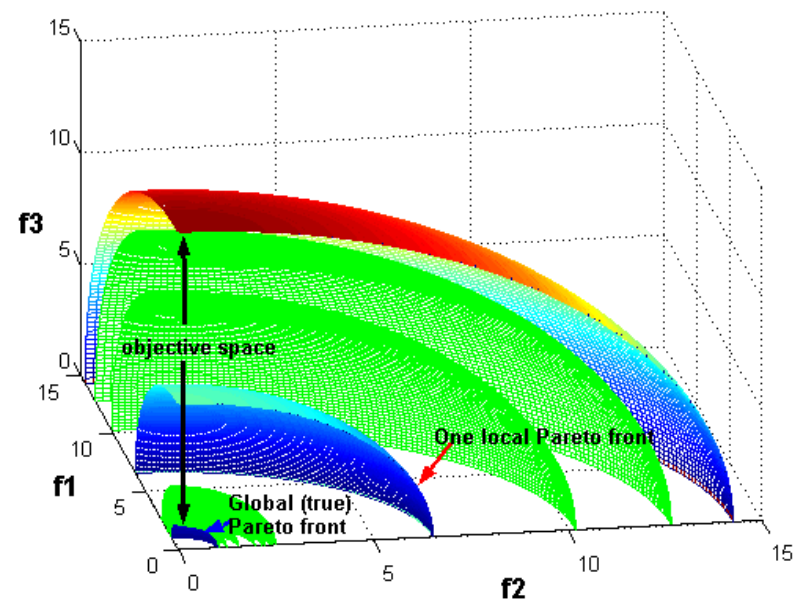
$$f_2(x) = (1 + g(\mathbf{x})) \cos\left(\frac{\pi x_1}{2}\right) \sin\left(\frac{\pi x_2}{2}\right)$$

$$f_3(x) = (1 + g(\mathbf{x})) \sin\left(\frac{\pi x_1}{2}\right)$$

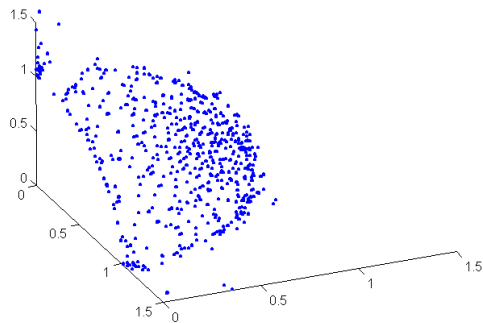
subject to

$$g(\mathbf{x}) = 12 + \sum_{i=1}^{12} (x_i - 0.5)^2 - \cos(20\pi(x_i - 0.5)),$$

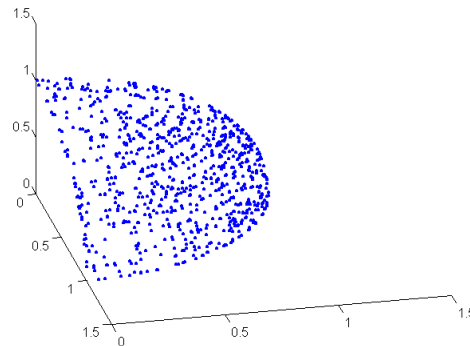
$$0 \leq x_i \leq 1, i = 1, \dots, 12$$



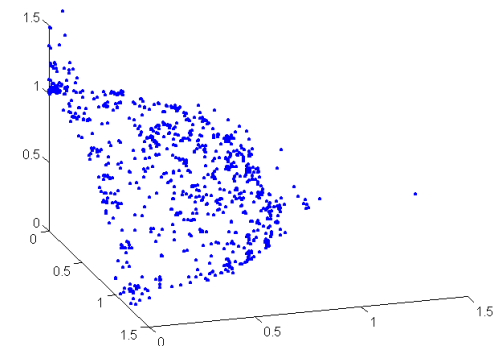
F-6: Resulted Pareto Fronts



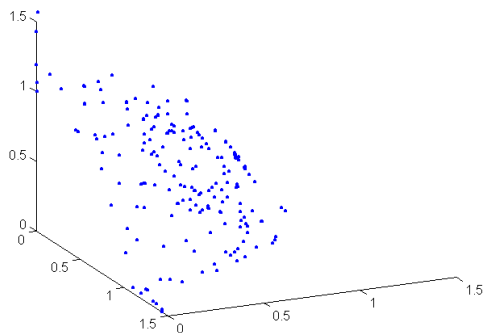
IMOE



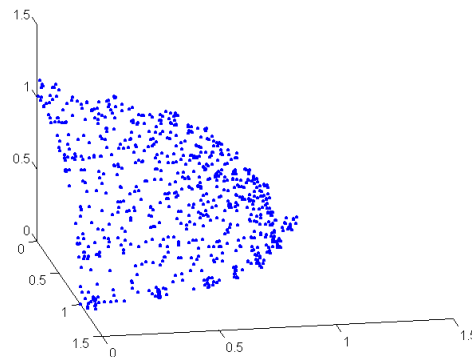
DMOEA



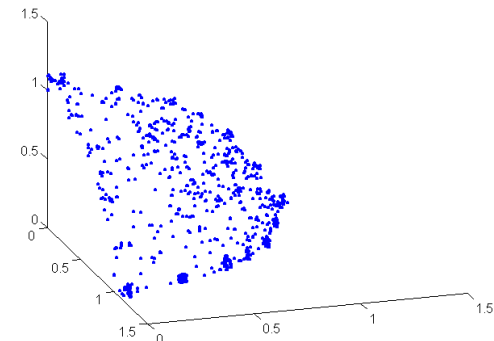
NSGA-II



PAES

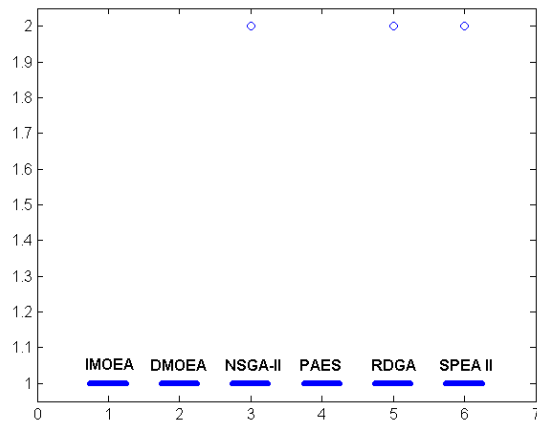


RDGA

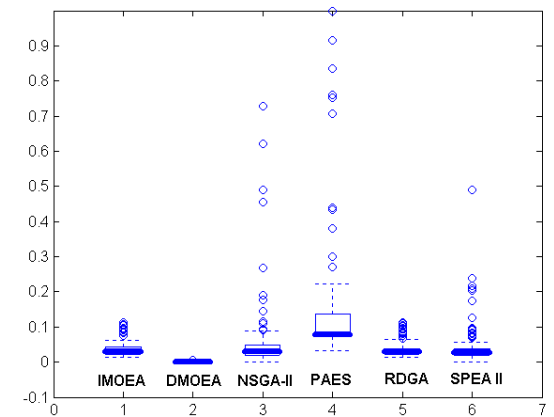


SPEA II

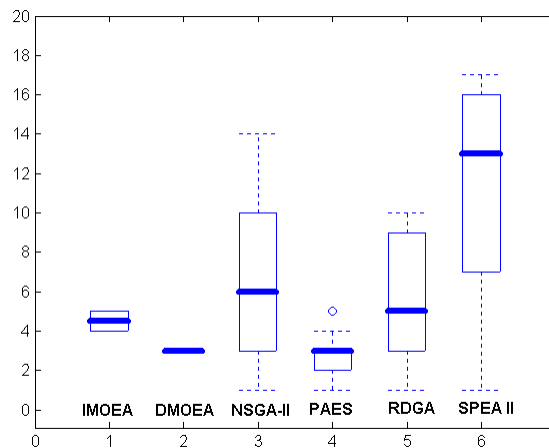
F-6: Rank, Density & Distance Measures



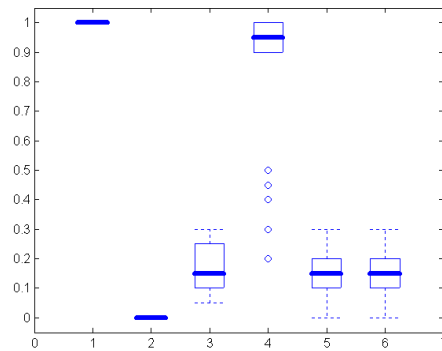
Rank



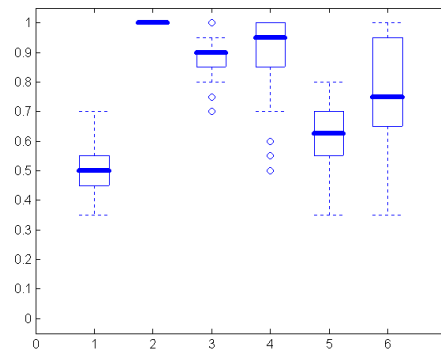
Distance



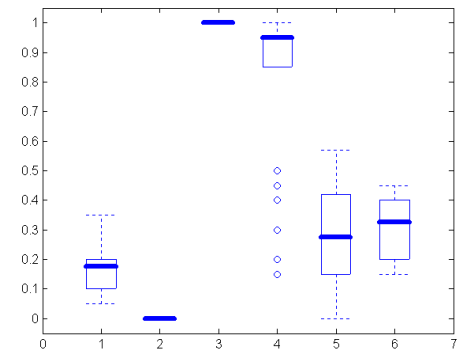
F-6: C Measure



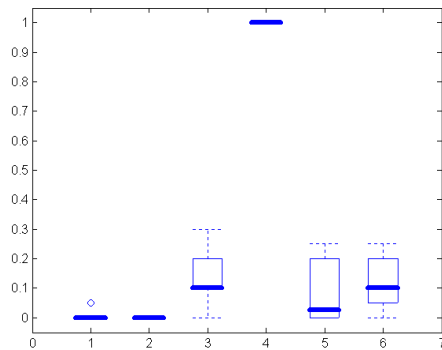
$C(X_1, X_{1-6})$



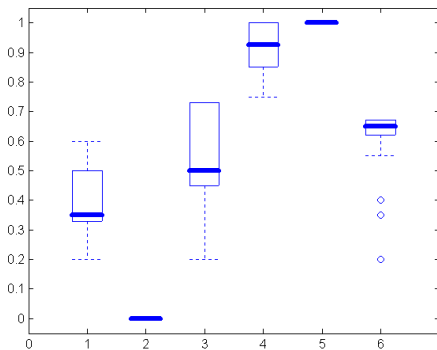
$C(X_2, X_{1-6})$



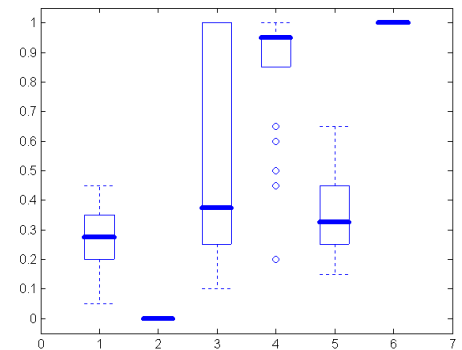
$C(X_3, X_{1-6})$



$C(X_4, X_{1-6})$



$C(X_5, X_{1-6})$



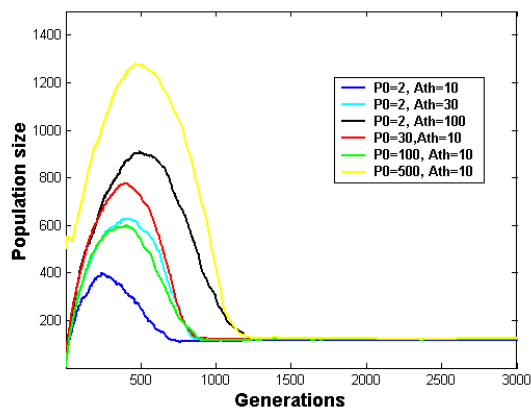
$C(X_6, X_{1-6})$

1.IMOEA; 2.DMOEA; 3.NSGA-II; 4.PAES; 5.RDGA; 6.SPEA II

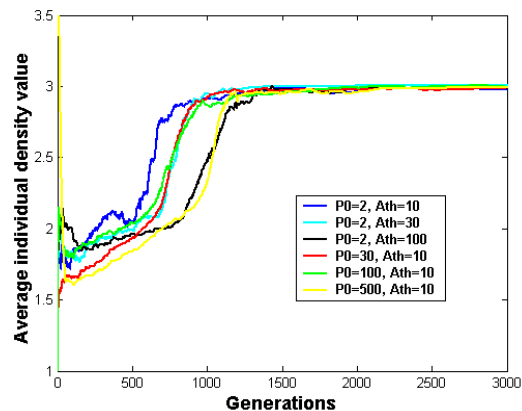
Robustness Study

Population size

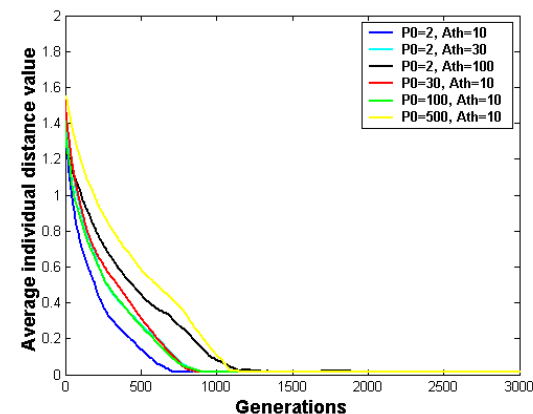
Function $F-4$:



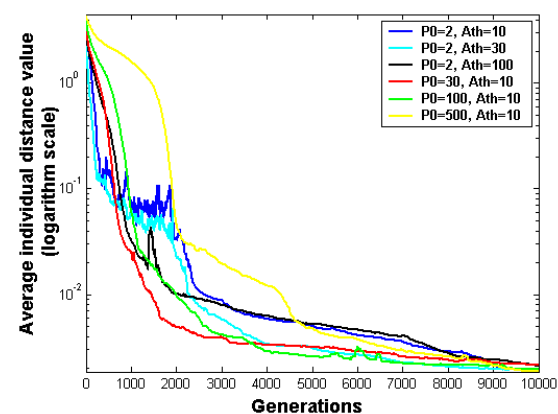
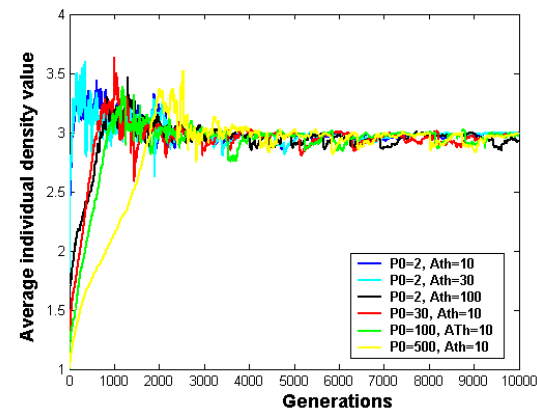
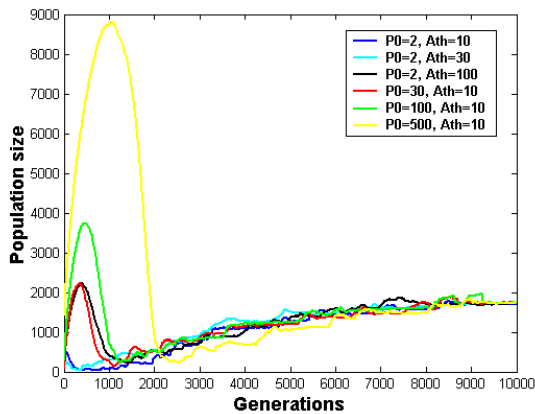
Density Value



Distance Value



Function $F-6$



Computational Complexity

- Efficiency of DMOEA— less time consuming comparing to the other advanced MOEAs.

	IMOEa	PAES	NSGA-II	RDGA	SPEA II	DMOEa (2,10)	DMOEa (2,30)	DMOEa (2,100)	DMOEa (30,10)	DMOEa (100,10)	DMOEa (500,10)
Time (min)	106	133	251	684	407	25	25	25	26	26	27

- DMOEa—potential in solving real-time complicated MOPs, which is still needed to be validated in future work

11- ϵ -MOEA

1) It is based on ϵ -dominance, which is defined as:

A feasible solution $x \in \Omega$ ϵ -dominates another solution $x' \in \Omega$, if and only if $f_i(x) - \epsilon \leq f_i(x'_i), \forall i \in \{1, 2, \dots, m\}$

2) ϵ -dominance mechanism generates a hypergrid in the objective space with multiple boxes:

The objective space is divided into hyperboxes by a size of ϵ

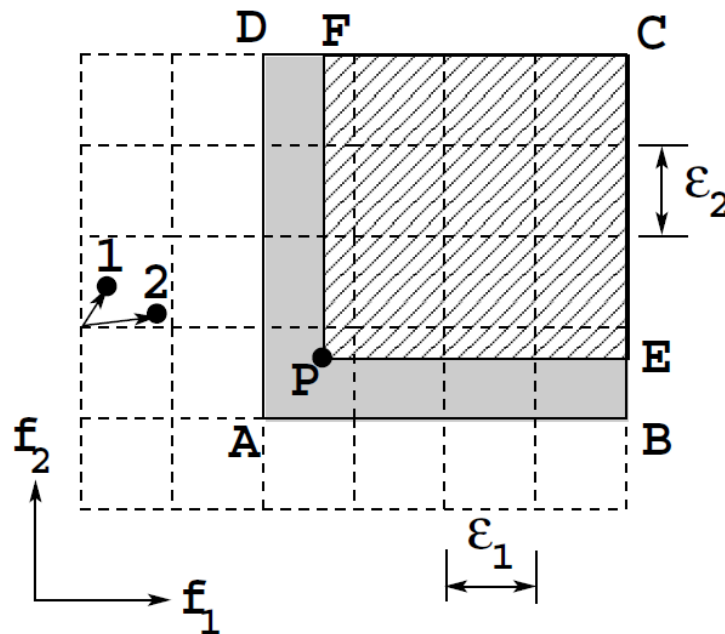
3) Each hyperbox is assigned at most a single point:

ϵ -dominance does not allow two solutions with a difference less than ϵ_i in the i -th objective to be non-dominated with each other

4) It provide a tradeoff among convergence, diversity, and computational time.

“Evaluating the ϵ -domination based multi-objective evolutionary algorithm for a quick computation of Pareto-optimal solutions,” by K. Deb, M. Mohan and S. Mishra, *Evolutionary Computation*, 13(4), 2005, pp. 501-525.

ε -dominance



- ε -dominance:
Point P dominates
the entire region
ABCD
- Pareto dominance:
Point P dominates
only the region
PECF

Archive Acceptance Procedure

- 1) Every solution in the archive is assigned an identification array $B = (B_1, B_2, \dots, B_M)^T$, M is the total number of objectives:

$$B_j(f) = \begin{cases} \lfloor (f_j - f_j^{\min}) / \varepsilon_j \rfloor & \text{for minimizing } f_j \\ \lceil (f_j - f_j^{\min}) / \varepsilon_j \rceil & \text{for maximizing } f_j \end{cases}$$

f_j^{\min} is the minimum possible value of the j -th objective and ε_j is the allowable tolerance in the j -th objective.

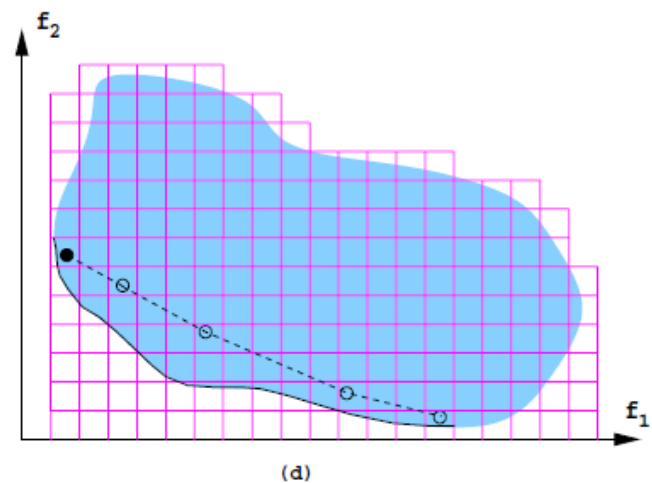
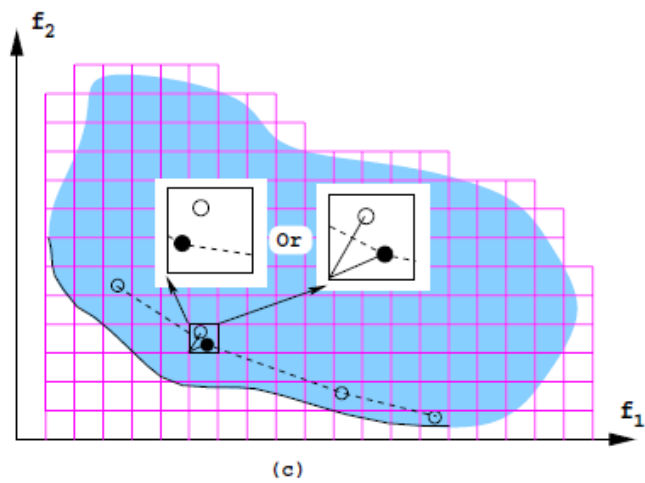
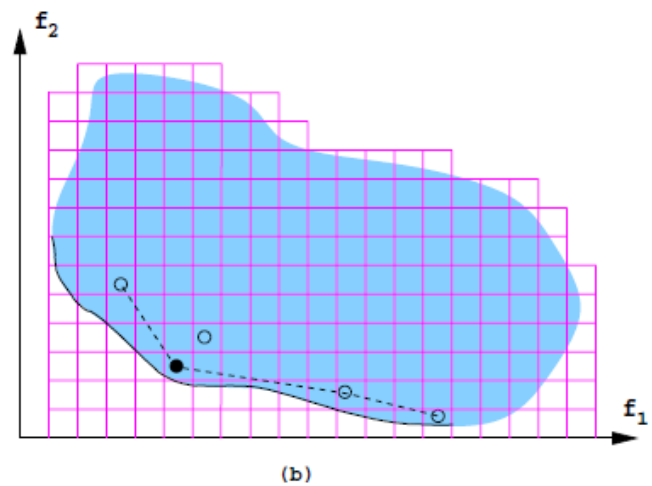
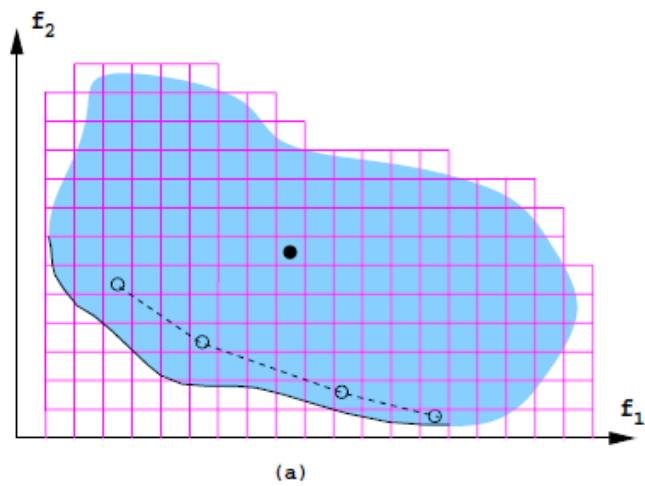
- 2) For offspring c and each archive member a :

Case1: If B^c dominates B^a , c is included in $A(t)$

Case2: If B^c is dominated by B^a , c is rejected by $A(t)$

Case3: If B^c is non-dominated with B^a and both c and a are in the same grid, when c is closed to the B vector, c is included in $A(t)$

Case4: If B^c is non-dominated with B^a and both c and a are not in the same grid, c is included in $A(t)$



General Framework

Input

- N : the size of population P ; a MOP and stopping criteria

Output

- $A(t)$: archive population

Step 1: Initialization

- Randomly initialize a population $P(0)$. The non-dominated solutions of $P(0)$ are copied to an archive population $A(0)$. Set the iteration counter $t = 0$.

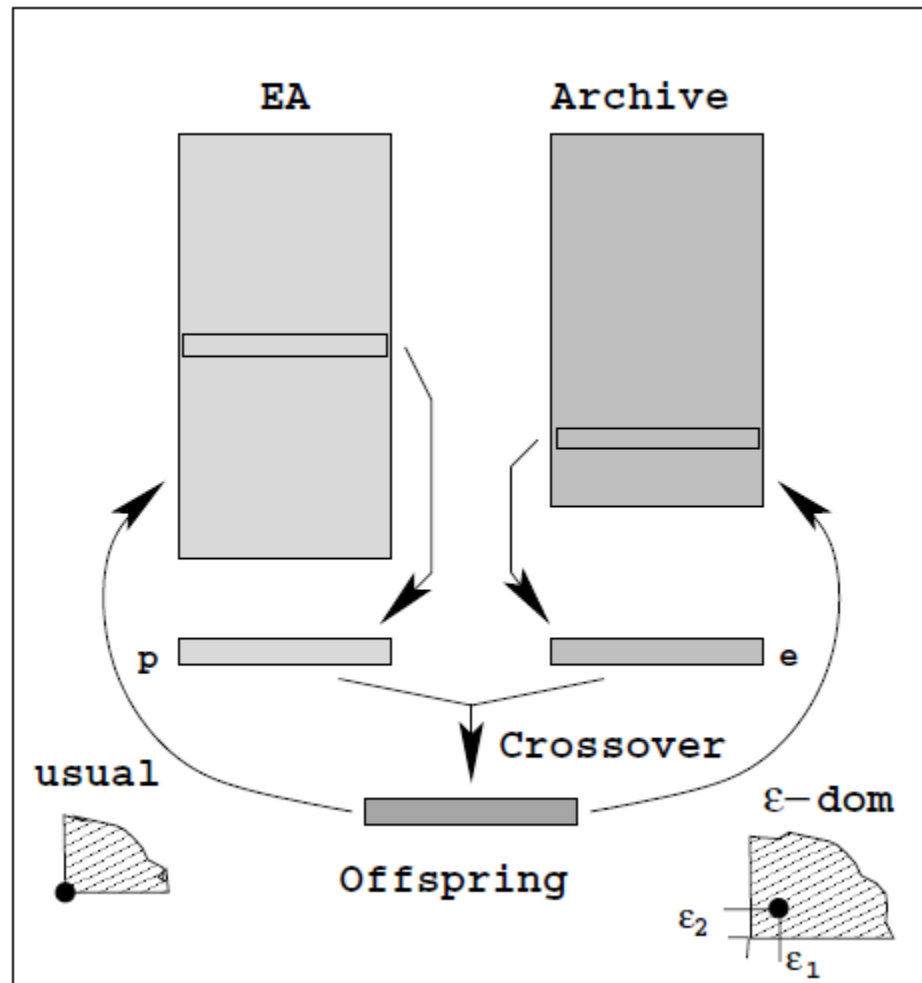
Step 2: Generating offspring

- One solution p is chosen from the population $P(t)$ by selection based on Pareto-dominance.
- One solution e is chosen from the population $A(t)$ randomly
- One offspring solutions c is created using p and e

Step 3: Archive selection

- Solutions c is included in $P(t)$ if it is non-dominated with others in $P(t)$
- Solutions c is included in $A(t)$ using archive-acceptance procedure
- Output $A(t)$ if stopping criteria is satisfied; otherwise, go back to step 2.

Steady State EA



DTLZ1

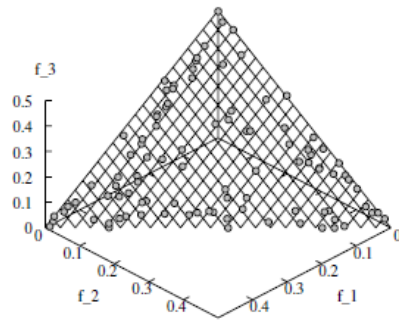


Figure 9: NSGA-II distribution on DTLZ1.

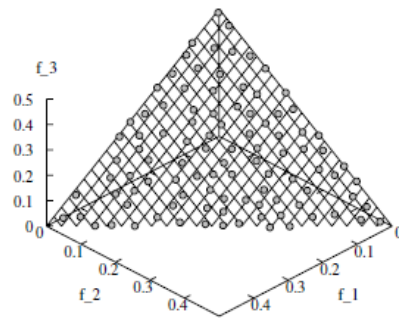


Figure 10: C-NSGA-II distribution on DTLZ1.

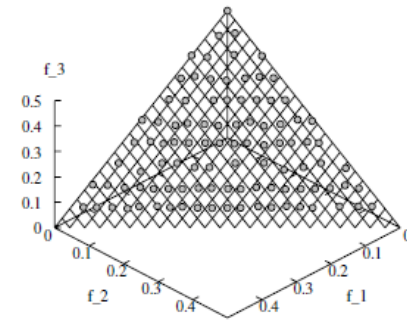


Figure 11: ϵ -MOEA distribution on DTLZ1.

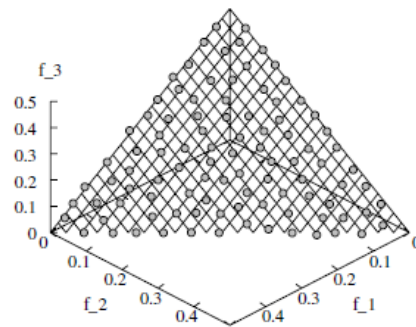


Figure 12: SPEA2 distribution on DTLZ1.

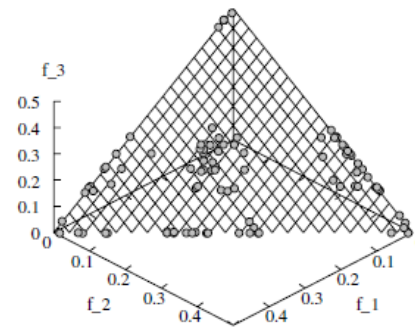


Figure 13: PESA distribution on DTLZ1.

DTLZ2

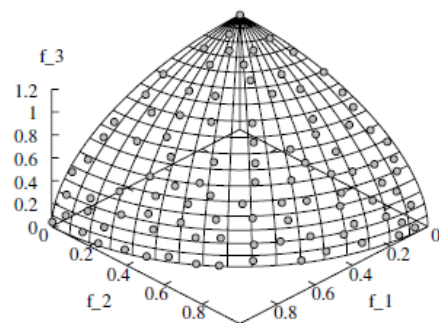


Figure 15: C-NSGA-II distribution on DTLZ2.

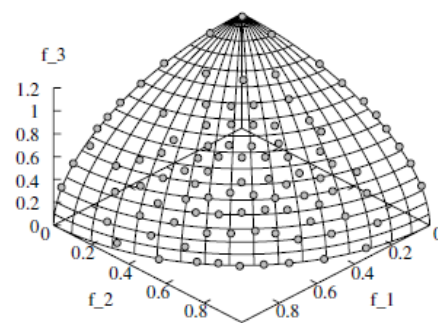


Figure 16: ϵ -MOEA distribution on DTLZ2.

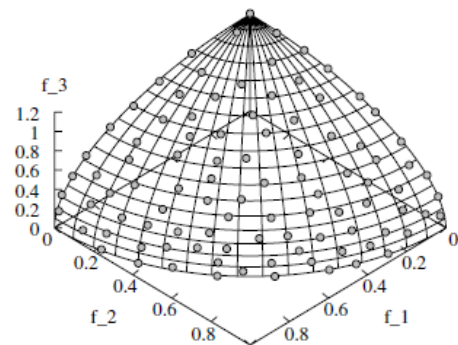


Figure 17: SPEA2 distribution on DTLZ2.

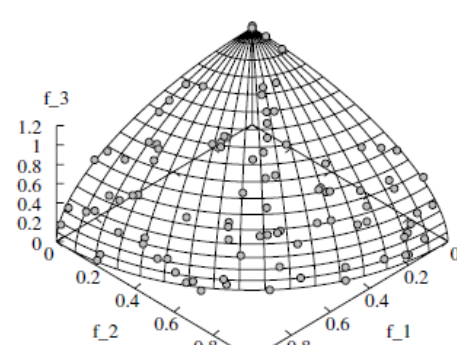


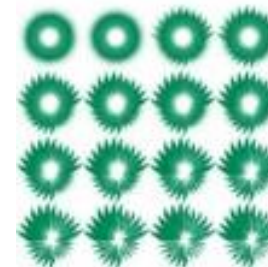
Figure 18: PESA distribution on DTLZ2.

Discussions of ε -MOEA

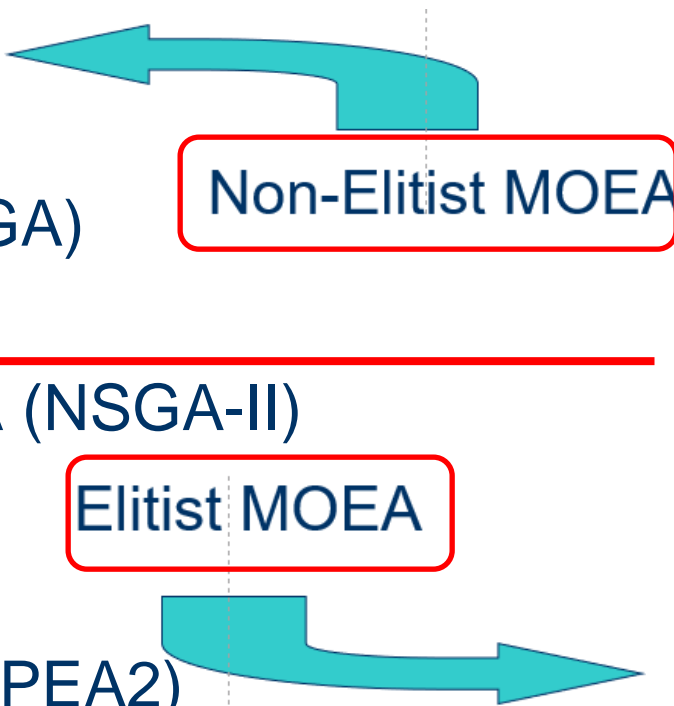
- The use of ε -dominance has two advantages:
 - reducing the cardinality of Pareto-optimal region
 - keeping two solutions away from each other with the minimum distance ε_i in the i th objective
- ε -MOEA generates a well-distributed and well-converged approximation front with a small computational time
- ε -MOEA is the first interactive MOEA with a decision-maker providing tradeoff among convergence, diversity and computational time. ε can be chosen by decision-maker according to his/her preference.

Efforts in Enhancing EAs for MOPs

- Modifying the *fitness assignment*
Lu&Yen, IEEE TEVC, 7(4), 2003, 325-343
- Enhancing the *convergence*
Leong&Yen, IEEE TSMCb, 38(5), 2008, 1270-1293
- Preserving the *diversity*
Daneshyari&Yen, IEEE TSMCa, 42(2), 2012, 475-490
- Managing the *population*
Yen&Lu, IEEE TEVC, 7(3), 2003, 253-274
- *Constraints and uncertainty handling*
Venkatraman&Yen, IEEE TEVC, 9(4), 2005, 424-435
Tessema&Yen, IEEE TSMCa, 39(3), 2009, 565-578
Woldesenbet&Yen, IEEE TEVC, 13(3), 2009, 500-513



List of MOEAs Studied

1. Vector Evaluated GA (VEGA)
 2. Weight Based GA (WBGA)
 3. Multiple Objective GA (F-MOGA)
 4. Non-Dominated Sorting GA (NSGA)
 5. Niche Pareto GA (NPGA)
-
6. Elitist Non-dominated Sorting GA (NSGA-II)
 7. Clustered NSGA-II
 8. Strength Pareto EA (SPEA)
 9. Advanced Strength Pareto EA (SPEA2)
 10. Dynamic Population MOEA (DMOEA)
 11. e-MOEA
- 
- The diagram illustrates the classification of the listed MOEAs. A red box labeled "Non-Elitist MOEA" is connected by a teal arrow pointing left to items 1 through 5. Another red box labeled "Elitist MOEA" is connected by a teal arrow pointing right to items 6 through 11. A horizontal red line separates the two groups.
- Non-Elitist MOEA
- Elitist MOEA

Critical Message Conveyed

The Design of *an Effective MOEA*
rests on four ingredients:

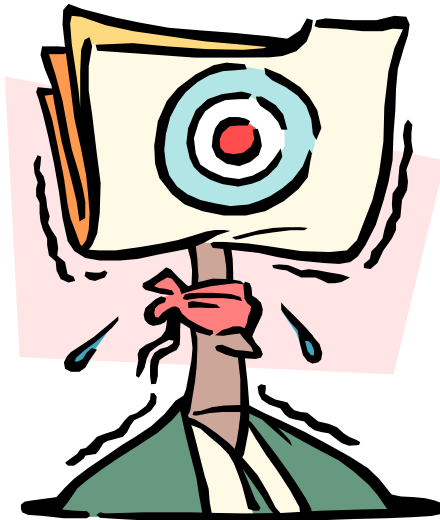
Convergence Promotion

Diversity Preservation

Elite Archive

Population Management

Q&A





PlatEMO

Evolutionary multi-objective optimization platform

Developed by BIMK (Institute of Bioinspired Intelligence and Mining Knowledge) of Anhui University and NICE (Nature Inspired Computing and Engineering Group) of University of Surrey



UNIVERSITY OF
SURREY

- 162 open source evolutionary algorithms
- 345 open source benchmark problems
- Powerful GUI for performing experiments in parallel
- Generating results in the format of Excel or LaTeX table by one-click operation
- State-of-the-art algorithms will be included continuously



Download **PlatEMO** source code from
<https://github.com/BIMK/PlatEMO>

The screenshot shows the GitHub repository page for BIMK/PlatEMO. The repository is public and has 43 watches, 619 stars, and 282 forks. The 'Code' button is highlighted with a red box, and the 'Download ZIP' option in the dropdown menu is also highlighted with a red box. The repository name is BIMK / PlatEMO. The file list shows: DestinyMy Update README.md, Doc (Update releasenote.md), PlatEMO (Update GUI.m), .gitignore (update gitignore), and README.md (Update README.md). The README.md file is selected, showing the PlatEMO logo and the text 'PlatEMO'. The right sidebar shows the 'About' section with the description 'Evolutionary multi-objective optimization platform' and tags 'matlab', 'evolutionary-algorithms', and 'multi-objective-optimization'. The 'Releases' section shows 'PlatEMO v3.3 (2021/8/14)' as the latest release. The 'Packages' section shows 'No packages published'. The 'Contributors' section shows 7 contributors.



PlatEMO-master

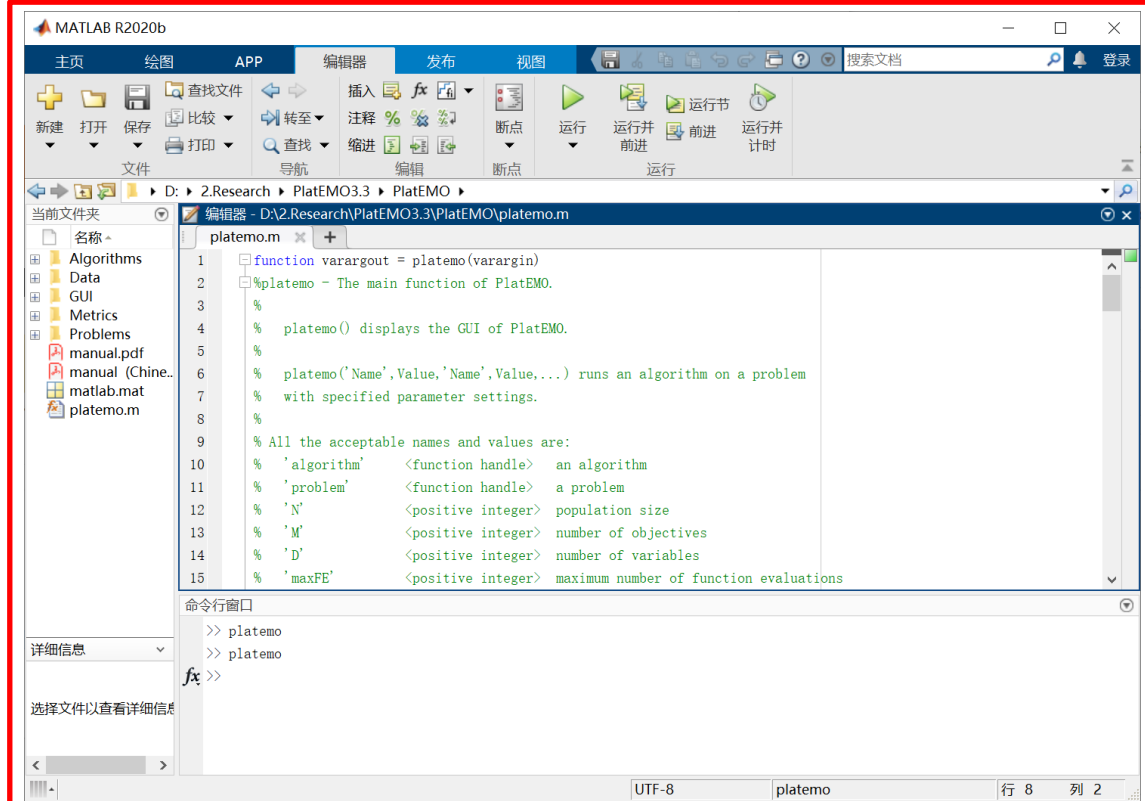


PlatEMO-master

> PlatEMO-master > PlatEMO-master

名称

- Algorithms
- GUI
- Metrics
- Problems
- manual (Chinese)
- manual
- platemo.m



Unzip

Run **platemo.m** in **MATLAB R2020b**

Test Module

PlatEMO v3.3

Modules Support

Test Module Application Module Experiment Module

Algorithm selection

Number of objectives
single multi many

Encoding scheme
real binary permutation

Special difficulties
large constrained expensive
multimodal sparse preference

Algorithms
91 / 16

Problems
129 / 345

Parameter setting

NSGAI ZDT3

N 100
M
D
maxFE 10000

Result display

Result selection
runtime


1. Choose a MOEA


2. Choose a MOP


Start Stop

PlatEMO v3.3

ModulesSupport

Test Module

Application Module

Experiment Module

Algorithm selection

Number of objectives

singlemulti**many**

Encoding scheme

realbinarypermutation

Special difficulties

largeconstrainedexpensive

multimodalsparsepreference

Algorithms

91 / 163

NSGAII

Problems

129 / 345

ZDT3

Parameter setting

NSGAII

ZDT3

N

100

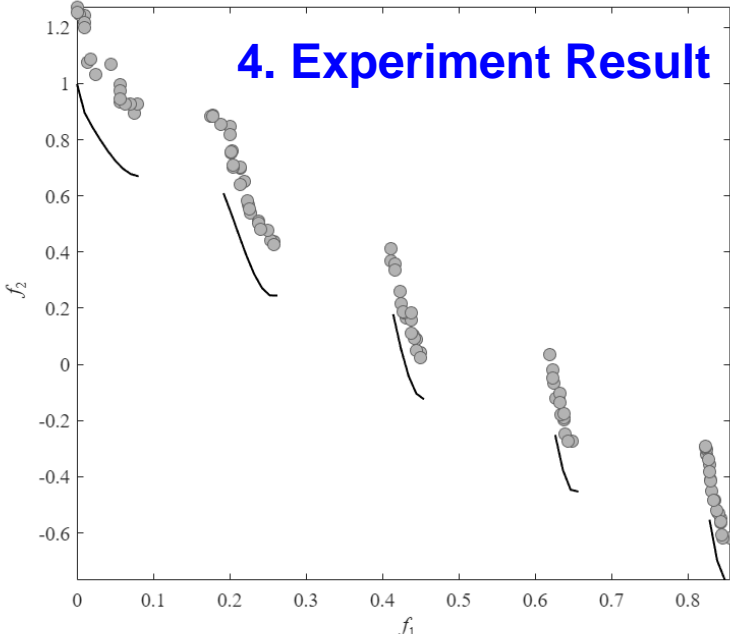
M

D

maxFE

10000

Result display



0%25%4700 evaluations

StartStop

Result selection

NSGAII on ZDT3

runtime0.1449s

<Algorithm: NSGAII>

<Problem: ZDT3>

N: 100

M:

D:

maxFE: 10000

4. Experiment Result

3. Start testing the performance

Experiment Module

PlatEMO v3.3

Modules Support

Test Module Application Module **Experiment Module**

Algorithm selection Parameter setting Result display

Number of objectives
single multi **many**

Encoding scheme
real binary permutation

Special difficulties
large constrained expensive
multimodal sparse preference

Algorithms 66 / 165
MaOEA
MyODEMR
NMPSO
NSGAIISDR
NSGAIIconflict
NSGAI

Problems 55 / 345
SMOP3
SMOP4
SMOP5
SMOP6
SMOP7
SMOP8

Number of runs 30
Number of results 1
File path ... D:\2.Research\PlatEMO

Parameter setting

NSGAI

GrEA

div

MOEAD

type 1

NMPSO

Result display

N M D FE IGD Mean (STD) Rank sum test Highlight the best

Parallel execution

Start Stop

1. Choose competing MOEAs

PlatEMO v3.3

Modules Support

Test Module Application Module **Experiment Module**

Algorithm selection

Number of objectives: single multi many

Encoding scheme: real binary permutation

Special difficulties: large constrained expensive multimodal sparse preference

Algorithms: 66 / 16

Problems: 55 / 34

Number of runs: 30

Number of results: 1

File path: D:\2.Research\PlatEMO

Parameter setting

NSGAIII

GrEA

div

MOEAD

type 1

NMPSO

WFG1

WFG2

WFG3

N 100

M

D

maxFE 10000

K

Result display

N M D FE IGD Mean (STD) Rank sum test Highlight the best

Parallel execution

Start Stop

2. Choose test MOPs

PlatEMO v3.3

ModulesSupport

Test Module

Application Module

Experiment Module

Algorithm selection

Parameter setting

Result display

Number of objectives

single

multi

many

Encoding scheme

real

binary

permutation

Special difficulties

large

constrained

expensive

multimodal

sparse

preference

Algorithms

66 / 163

MyODEMR

NMPSO

NSGAIISDR

NSGAIIconflict

NSGAI

Problems

55 / 345

SMOP8

WFG1

WFG2

WFG3

WFG4

WFG5

Number of runs

2

Number of results

1

File path

...

D:\2.Research\PlatEI

NSGAI

GrEA

div

MOEA

type

1

NMPSO

WFG1

WFG2

WFG3

N

100

M

D

maxFE

10000

K

Result display

N

M

D

FE

IGD

Mean (STD)

Rank sum test

Highlight the best

	N	M	D	NSGAI	GrEA	MOEA	NMPSO
WFG1	100	3	12	7.2881e-1 (1.15e-1) =	5.0535e-1 (1.52e-4) =	7.3442e-1 (6.22e-2) =	1.1214e+0 (4.55e-2)
WFG2	100	3	12	1.6982e-1 (4.80e-3) =	2.2594e-1 (1.70e-2) =	4.7808e-1 (2.65e-1) =	3.7626e-1 (1.99e-3)
WFG3	100	3	12	1.7931e-1 (2.42e-2) =	9.6625e-2 (2.47e-3) =	2.5833e-1 (3.24e-2) =	9.3469e-2 (1.19e-3)
+/-/=				0/0/3	0/0/3	0/0/3	

4. Comparison results

3. Start comparison experiments

Parallel execution

Start

Stop



PlatEMO

More detail in [Manual.pdf](#)
(User Manual Version 3.3)



PlatEMO

*Evolutionary Multi-Objective
Optimization Platform*

User Manual 3.3

BIMK Group
August 14, 2021

Contents

I. Quick Start.....	1
II. Using PlatEMO without GUI.....	2
A. Solving Benchmark Problems.....	2
B. Solving User-Defined Problems.....	3
C. Collecting the Results.....	5
III. Using PlatEMO with GUI.....	7
A. Functions of Test Module.....	7
B. Functions of Application Module.....	7
C. Functions of Experiment Module.....	8
D. Labels of Algorithms and Problems.....	9
IV. Extending PlatEMO.....	11
A. ALGORITHM Class.....	11
B. PROBLEM Class.....	13
C. SOLUTION Class.....	17
D. Whole Procedure of One Run.....	18
E. Metric Function.....	19
V. List of Algorithms.....	20
VI. List of Problems.....	26

- Due: next Monday on 7/11/22, before noon

Homework #5

Problem 1: Develop the NSGA-II [1] or SPEA2 [2] multiobjective evolutionary algorithms to solve five benchmark problems given in ZDT suite [3] (specifically, ZDT1, ZDT2, ZDT3, ZDT4 and ZDT6). First follow closely the algorithm presented by the contributing authors.

Prepare a 3-page document in the following order.

1. Summary: overview the MOEAs considered
2. Description: detail the algorithm design in the context of fitness assignment, diversity maintenance and other ingredients involved in the recipes
3. Simulation: present the simulation findings (non-dominated front) and *statistical* results
4. Observations: document the pros and cons of the algorithm
5. Potential Improvement: provide your thought to advance the proposed concept from what you have design in Problem 2
6. Conclusion: draw relevant conclusions
7. Appendix: codes developed

References:

Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II):

K. Deb, S. Agrawal, A. Pratap and T. Meyarivan, "A fast elitist non-dominated sorting genetic algorithm for multi-objective optimization: NSGA-II," *IEEE Transactions on Evolutionary Computation*, 6, pp. 182-197, 2002.

Improved Strength Pareto Evolutionary Algorithm (SPEA2):

E. Zitzler, M. Laumanns and L. Thiele, "SPEA2: improving the Strength Pareto Evolutionary Algorithm," *Technical Report TIK-Report 103*, Swiss Federal Institute of Technology, 2001.

ZDT Test Suite:

E. Zitzler, K. Deb and L. Thiele, "Comparison of multiobjective evolutionary algorithms: empirical results," *Evolutionary Computation*, 8(2), 2000, pp. 173-195.

