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

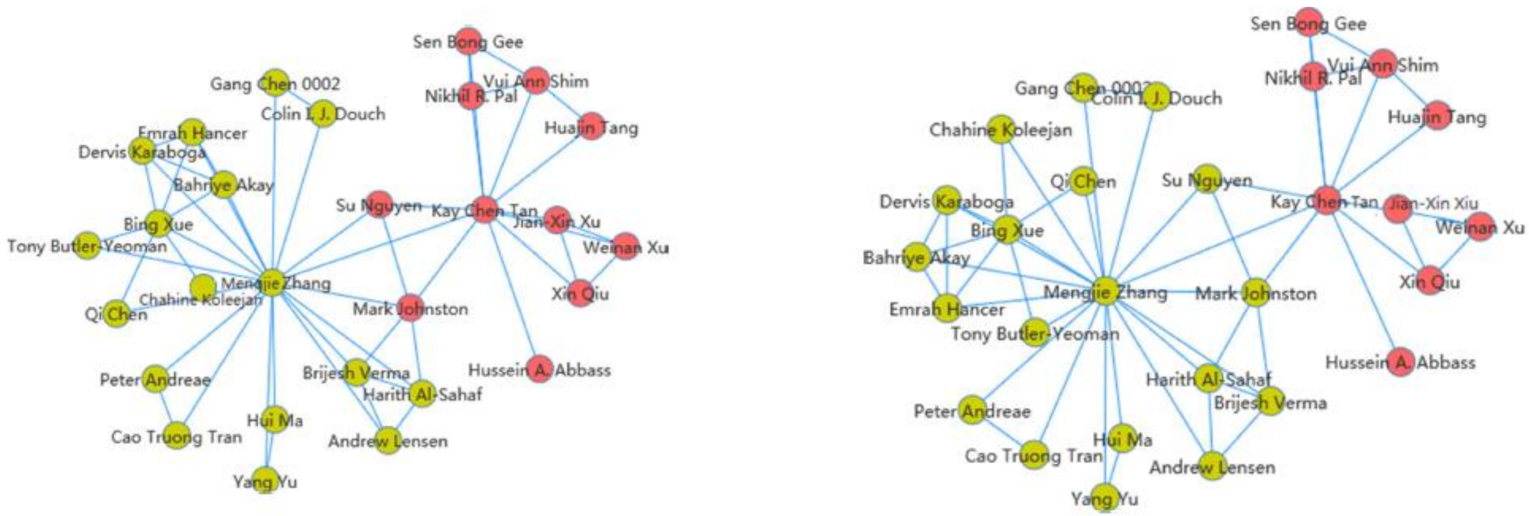
Sichuan University, Chengdu, CHINA

Day Eight of EIGHT, July 7, 2022

Case Study 8:
Dynamic Community Detection

- **Motivation:** The community detection in dynamic networks is essential for important applications such as social network analysis. Such detection requires simultaneous maximization of the clustering accuracy at the current time step while minimization of the clustering drift between two successive time steps.

- **Approach:** Knowledge from the previous step is obtained by extracting the intrapopulation consensus communities from the optimal population of the previous step. Subsequently, the intrapopulation consensus communities of the previous step obtained is voted by the population of the current time step during the evolutionary process. A subset of the consensus communities, which receives a high support rate, will be recognized as the interpopulation consensus communities of the previous and current steps. Interpopulation consensus communities are the knowledge that can be transferred from the previous to the current step



(a) (b)

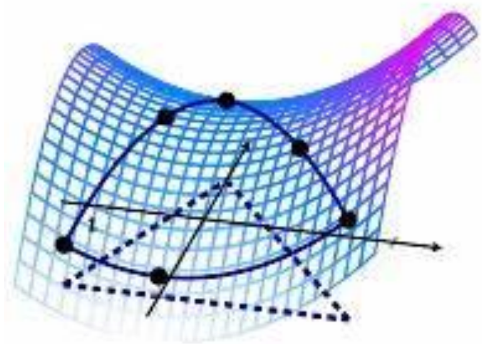
Fig. 11. Clustering results associated with Mengjie Zhang and Kay Chen Tan detected by the CCPSO in DBLP 2014 collaboration network. (a) Without transferring interpopulation consensus communities. (b) With transferring interpopulation consensus communities.

[redacted]

Linked in
skype
LIVEJOURNAL
foursquare
vimeo
You Tube
Insta
facebook
Blogger
my
twitter
flickr
tumblr.
Like
yelp

11. CONSTRAINT OPTIMIZATION IN EVOLUTIONARY ALGORITHMS

约束优化



Constrained Optimization

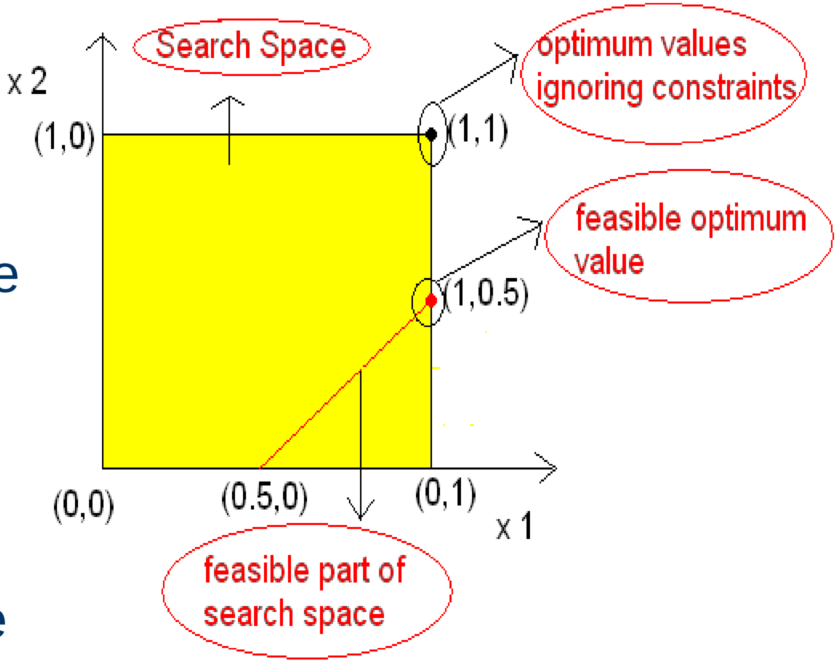
- Constraints on supplies of raw materials, manpower, and machines…
- None of the variables is available infinitely and they also have relationships among them which are constrained.
- *Difficulties are various limits on decision variables, constraints involved, interference among constraints, and the inter-relationship between the constraints and the objective functions.*
- Hence, we can safely say that almost all of the real-world optimization problems are constrained optimization problems.

An Illustrating Example

- Consider the optimization problem

$$\begin{aligned} &Maximize\ f(X) = x_1 + x_2 \\ &where,\ 0 \le x_1, x_2 \le 1 \end{aligned}$$

- Then the optimum solution will be at $x_1 = 1$ and $x_2 = 1$
- But, if an equality constraint is introduced such that $g(x) \equiv x_1 - x_2 = 0.5$
- The optimum solution in the presence of constraints would be $x_1 = 1\ and\ x_2 = 0.5$.
- Thus, the constraint play a very important role in optimization.



Need for Constraint Handling

- We saw from our previous example how the feasible search space of the problem diminished completely with the introduction of only one constraint.
- Considering a resolution of up to 2 decimal places in the discrete search space, there are only 50 feasible solutions among 10,000 possible solutions. This implies that feasible search space is only ***0.5%*** of the actual parameter space.
- For a search algorithm like the GA, this could cause extreme difficulty in even finding feasible solutions unless the search is directed to the feasible regions (GA has traditionally been designed as search algorithm for unconstrained optimization problems).

- Consider the numerical example,

$$\textit{unconstrained maxima} \Rightarrow f(X) = 2.0$$

$$\textit{constrained maxima} \Rightarrow f(X) = 1.5$$

- So obviously if the search is directed solely by the objective function, then GA will be stuck in the value of the unconstrained maxima of 2.0 and never find the maximum among feasible solutions, viz. 1.5.
- Hence the important issue in constrained optimization is the *simultaneous maintenance of feasibility among solutions while optimizing for the objective function*.

Problem Formulation

- The general constrained, continuous parameterized optimization problem is to find X so as to

$$Minimize\ f(X), \quad X = (x_1, x_2, ..., x_n)$$

- The domains of the variables are defined by their lower and upper bounds as,

$$l(i) \leq x_i \leq u(i), \quad 1 \leq i \leq n$$

- The constraints are defined by *q* inequality constraints and *m*-*q* equality constraints

$$g_j(X) < 0, \quad (j = 1, ..., q)$$

$$h_j(X) = 0, \quad (j = q+1, ..., m)$$

where the relationship holds

$$\boxed{F \Rightarrow \text{feasible space}}$$

$$S \Rightarrow \text{search space}$$

$$\mathrm{F} \subseteq \mathrm{S}$$

Constraint Violations

- Because we are dealing with both equality and inequality constraints, we can define the *constraint violation* as,

$$c_j(X) = \begin{cases} \max(0, g_j(X)), & \text{if } 1 \le j \le q \\ \max(0, (|h_j(X)| - \delta)), & \text{if } q+1 \le j \le m \end{cases}$$

  where $g_j(\mathrm{X})$ denotes the inequality constraints and $h_j(\mathrm{X})$ denotes the equality constraints.
- The value of δ is usually 0.001 or 0.0001 and is called the threshold value of the equality constraint.
- Usually, the equality constraint is converted to an inequality constraint with a small threshold value.
- The inequality constraint j that take the value of 0 at the global optimum, *i.e.* $g_j(X)=0$ are called the *active constraints*. All equality constraints are active ones.

Constraint Handling Approaches

1. Penalty function methods
2. Preference of feasible solutions over infeasible ones
3. Methods based on special operators
4. Methods based on decoders
5. Repair Algorithms
6. Methods based on multi-objective optimization

1. Penalty Function Methods

- Penalty function involve penalizing the individuals for violating any of the constraints. The penalty can range from completely rejecting any infeasible individuals to degrading their fitness values.
- There are different types of penalty functions like

  1.1 death penalty

  1.2 static penalty function

  1.3 dynamic penalty function

  1.4 self-adaptive penalty function

  an adaptive penalty formulation

  1.5 self-adaptive penalty function using co-evolution

  stochastic ranking scheme

1.1 Death Penalty

- The simplest constraint handling scheme; this involves completely ignoring any infeasible solutions. Whenever infeasible solutions are found, they will never be selected.

- This method is the simplest to implement but it may defer the convergence process especially in problems where feasible solutions are hard to come by. This method will *not exploit* any information from the infeasible individuals.

- The death penalty are suitable for problems with a large feasible region because it may be successful in finding feasible solutions.

1.2 Static Penalty Function

- In these methods a penalty term directly dependent on the constraint violation is added to objective function before evaluating the fitness of the individual.

- The fitness function for a minimization problem is defined as follows,

$$\boxed{\text{fitness}(X) = f(X) + \sum_{j=1}^{m} r_j c_j(X)}$$

  where $c_j(X)$ denotes the constraint violations defined before and r_j corresponds to the penalty coefficient for the *j*th constraint.

- The penalty coefficients should be decided, *heuristically*, based on the difficulty associated with satisfying each constraint.

1.3 Dynamic Penalty Function

- Unlike the static penalty, the value of the penalty in dynamic penalty method increases with the ***generations*** also.
- So, in the beginning, the infeasible solutions will face a lesser penalty than towards the end. This is possible because the generation number is used directly in the formulation of the dynamic fitness function, e.g.,

$$\text{fitness}(X) = f(X) + (C \times t)^{\alpha} \times SVC(\beta, X)$$

$$SVC(\beta, X) = \sum_{j=1}^{q} c_j^{\beta}(X) + \sum_{j=q+1}^{m} c_j(X)$$

$t - current\ generation\ number$

$\alpha, \beta, C - constants$

- A reasonable choice of parameters, $C = 0.5$, $\alpha = 2$ and $\beta = 2$.
- Experiment have shown that this method performs very well for quadratic problems.
- Because of the dynamic nature of the fitness function, the selective pressure in finding feasible solutions increases with the generations.
- Dynamic penalty function requires many parameters to be tuned simultaneously for good results to be obtained. This requires some information about the problem and hence this method is considered problem dependent.

Guidelines for Penalty Functions

- Penalties which are functions of the *distance from feasibility* perform better than those which are only functions of the number of violated constraints.
- For a problem having few constraints and a few feasible solutions, penalties which are solely functions of the violated constraints are not likely to produce any feasible solution.
- The more accurate the penalty is estimated, the better quality solution will be found.

1.4 Self-Adaptive Penalty Function

- Make use of feedback during evolution concerning the current population. In this way it is self adapting to the degree of constraint of the problem.
- Use a problem specific distance parameter, the “Near-Feasibility-Threshold” (*NFT*) for each constraint.
- The prominence of the *NFT* is that the penalty function will *encourage the GA to explore within the feasible region and the NFT-neighborhood of the feasible region and discourage search beyond the threshold*.
- Conceptually the *NFT* is the threshold distance from the feasible region at which the user would consider the search as valid.

- The primary problem-specific parameter can be varied using the following equation,

$$NFT_j = \frac{NFT_0}{1+\Lambda}$$

$$NFT_0 \Rightarrow \text{upper bound for } NFT$$

$$\Lambda \Rightarrow \text{dynamic search parameter}$$

- Notice that if $\Lambda = 0$, then the *NFT* remains static.

- Λ can be defined as a function of generation number or as a function of the feasibility of recent best solutions.

- The fitness of each solution is defined as,

$$\text{fitness}(X) = f(X) + \left(F_{\text{feasible}} - F_{\text{all}}\right)\sum_{j=1}^{m}\left(\frac{c_j(X)}{NFT_j(t)}\right)^{k_j}$$

$F_{\text{feasible}} \Rightarrow$ best known feasible objective function value at generation t

$F_{\text{all}} \Rightarrow$ best unpenalized objective function value at generation t

$NFT_j(t) \Rightarrow$ NFT corresponding to constraint j at generation t

$k_j \Rightarrow$ adjust the severity of the penalty for each constraint

- Hence, we can see that starting from an initial value of the *NFT*, the algorithm self-adapts to find feasible optimal solutions.
- The major difference with this method is that feasible solutions do not dominate infeasible solutions if they are nearly feasible. So, this can help in exploiting the solutions at the boundary of the constraint.

- This method is suitable especially for problems with active constraints. This is because the design of the algorithm favors a genetic search *on the boundary of the constraints* from both the feasible and infeasible regions.

- A drawback with this method is that if there is no feasible solution in the population, then there will be no penalty applied to any of the solutions. Hence the selective pressure is not directed towards finding feasible solutions.

An Adaptive Penalty Formulation

- An adaptive penalty function formulation was proposed to fully exploit information hidden in infeasible individuals- *finding feasible solutions before looking for optimal one*.
  - If there are few feasible individuals available, a larger amount of penalty will be added to infeasible individuals with a higher amount of constraint violation.
  - If there are sufficient number of feasible individuals present, them infeasible individuals with larger objective function values will be penalized more than infeasible individuals with smaller objective function values.
  - These two penalties will allow the algorithm to balance between *finding more feasible solutions* and searching for the optimum solution at anytime during the search process.
  - No parameter tuning is needed.

- Individuals with low constraint violation are preferred at the *beginning* of the search process, while individuals with low fitness values are preferred only at the *later* stages.
- normalized fitness $$\tilde{f}(\vec{x}) = \frac{f(\vec{x}) - f_{\min}}{f_{\max} - f_{\min}}$$
- the average, normalized constraint violation $$v(\vec{x}) = \frac{1}{m} \sum_{j=1}^{m} \frac{c_j(\vec{x})}{c_{\max,j}}$$
- modified fitness value

$$n(\vec{x}) = \begin{cases} v(\vec{x}), & \text{if } r_f = 0 \\ \sqrt{\tilde{f}(\vec{x})^2 + v(\vec{x})^2}, & \text{otherwise} \end{cases}$$

$$r_f = \frac{\text{number of feasible individuals in current population}}{\text{population size}}$$

$$n(\vec{x}) = \begin{cases} \boxed{v(\vec{x}), \qquad \text{if } r_f = 0} \\ \sqrt{\tilde{f}(\vec{x})^2 + v(\vec{x})^2}, & \text{otherwise} \end{cases}$$

- if there is no feasible individual in the current population, then the modified fitness value will be equal to the constraint violation of the individuals. An infeasible individual with smaller constraint violation will be considered better fit than another infeasible individual with higher constraint violation irrespective of their fitness values. It will help in approaching the feasible region very quickly.
- if there are one or more feasible solutions available,

“An Adaptive Penalty Formulation for Constrained Evolutionary Optimization,” by Tessema B. and Yen G.G., ***IEEE SMC A: Systems and Humans***, 39(3), 2009, pp. 565-578.

$$n(\vec{x}) = \begin{cases} v(\vec{x}), & \text{if } r_f = 0 \\ \sqrt{\tilde{f}(\vec{x})^2 + v(\vec{x})^2}, & \text{otherwise} \end{cases}$$

- For feasible individuals, the modified fitness *n*(*x*) is equal to $\tilde{f}(\vec{x})$ This implies that if we compare the modified fitness value of two feasible individuals, then the individual with smaller normalized fitness will have smaller modified fitness value.
- For infeasible individuals, the modified fitness value is the measure of the normalized fitness and the constraint violation. The individuals near the origin (in the $\tilde{f}(\vec{x})$- $v(\vec{x})$ space) would have lower modified fitness value than those farther away from the origin. Therefore, if we compare two infeasible individuals, the one that has both lower normalized fitness and lower constraint violation will be considered better fit.
- If we compare the modified fitness values of a feasible individual and an infeasible individual, then either one could have smaller modified fitness value. However, if the two individuals have the same normalized fitness, the feasible individual will have lower modified fitness value.

- In addition to the penalty imposed upon infeasible individuals this way, two other penalties will also be added.
    - to further reduce the fitness of infeasible individuals as the penalty imposed upon infeasible individuals by the modified fitness formulation is very small, and
    - to identify the best infeasible individuals in the current population by adding different amounts of penalty to each infeasible individual.

- The number of feasible individuals in the population is used to determine the amount of penalty imposed upon an infeasible individual. If there are *few feasible individuals* in the population, we would want infeasible individuals with ***lower constraint violation*** to be less penalized than those with higher constraint violation. If there are *many feasible individuals* in the population, we would favor infeasible individuals with ***lower normalized fitness*** to be less penalized.

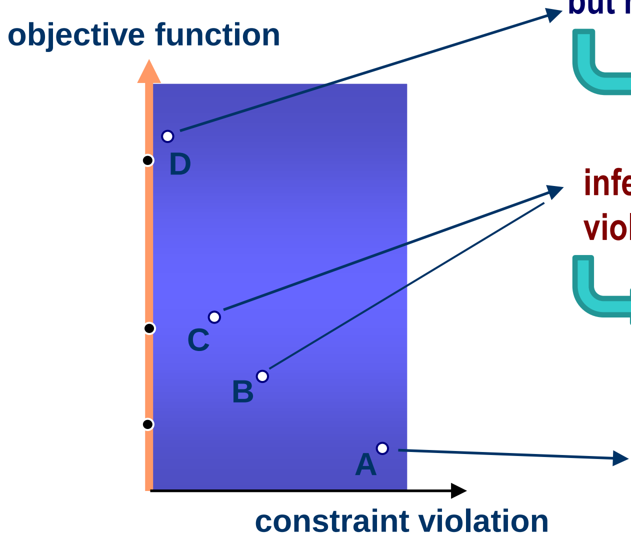
$$p(\vec{x}) = (1 - r_f)X(\vec{x}) + r_f Y(\vec{x})$$

where

$$X(\vec{x}) = \begin{cases} 0, & \text{if } r_f = 0 \\ v(\vec{x}), & \text{otherwise} \end{cases}$$

$$Y(\vec{x}) = \begin{cases} 0, & \text{for feasible individual} \\ \tilde{f}(\vec{x}), & \text{for infeasible individual.} \end{cases}$$

- From the penalty function defined, we can observe that if the feasibility ratio of the population is small (but not zero), then the first penalty ($X(x)$) will have more impact than the second penalty ($Y\ (x)$). The first penalty is formulated to have a large value for individuals with a large amount of constraint violation. Hence, in the case that there are few feasible individuals present in the population, infeasible individuals with higher constraint violation will be more penalized than those with lower constraint violation.
- On the other hand, if there are many feasible solutions in the population, the second penalty will have more effect than the first one. In this case, infeasible individuals with larger normalized fitness value will be more penalized than infeasible individuals with smaller normalized fitness.
- If there are no feasible individuals in the population, both penalties will be zero.



infeasible individual with low constraint violation
but high objective value

these types of infeasible individuals are needed
when number of feasible individuals is low

infeasible individuals with relatively low constraint
violation and low objective value

these types of infeasible individuals are needed during
most of the search process

infeasible individual with high constraint violation
but low objective value

these types of infeasible individuals are needed
when there are many feasible individuals

- In summary,
  - If there is no feasible individual in the current population, $n(x)$ will be equal to the constraint violation ($v(x)$) and $p(x)$ will be zero. In this case, the objective function value of these individuals will be totally disregarded, and all individuals will be given a fitness value based on their constraint violation alone. This will help us to find feasible individuals before we try to search for the optimum value.
  - If there are feasible individuals in the population, then $n(x)$ will mainly determine which individuals are better fit. An individual with lower modified fitness value will be better fit than an individual with higher one, or stated in a different way, individuals with both lower fitness value and lower constraint violation will be considered better fit than individuals that have higher fitness value or higher constraint violation or both.

- If two individuals have equal or very close modified fitness value, then the penalty value ($p(x)$) will determine which one is better. According to the penalty function, if the feasibility ratio (r_f) in the population is small, then the individual closer to the feasible space will be considered better fit. Otherwise, the individual with lower normalized fitness value will be better fit.
- If there is no infeasible individual in the population ($r_f = 1$), then individuals will be compared based on their normalized fitness value alone ($n(x) = \tilde{f}(x)$ and $p(x) = 0$).

Feasibility ratio
Linear inequality
Linear equality
Active Constraint

Prob.	n	Type of function	ρ	LI	NI	LE	NE	A
g01	13	Quadratic	0.0111%	9	0	0	0	6
g02	20	Nonlinear	99.9971%	1	1	0	0	1
g03	10	Nonlinear	0.0000%	0	0	0	1	1
g04	5	Quadratic	52.1230%	0	6	0	0	2
g05	4	Cubic	0.0000%	2	0	0	3	3
g06	2	Cubic	0.0066%	0	2	0	0	2
g07	10	Quadratic	0.0003%	3	5	0	0	6
g08	2	Nonlinear	0.8560%	0	2	0	0	0
g09	7	Nonlinear	0.5121%	0	4	0	0	2
g10	8	Linear	0.0010%	3	3	0	0	3
g11	2	Quadratic	0.0000%	0	0	0	1	1
g12	3	Quadratic	4.7713%	0	1	0	0	0
g13	5	Nonlinear	0.0000%	0	0	0	3	3
g14	10	Nonlinear	0.0000%	0	0	3	0	3
g15	3	Quadratic	0.0000%	0	0	1	1	2
g16	5	Nonlinear	0.0204%	4	34	0	0	4
g17	6	Nonlinear	0.0000%	0	0	0	4	4
g18	9	Quadratic	0.0000%	0	13	0	0	6
g19	15	Cubic	33.4761%	0	5	0	0	0
g21	7	Linear	0.0000%	0	1	0	5	6
g23	9	Linear	0.0000%	0	2	3	1	6
g24	2	Linear	79.6556%	0	2	0	0	2

Comparison of Median and Worst Results Found Using the Proposed Algorithm Against SMES [38], SR [39], and Alpha Simplex (αSIMPLEX) [23]

f	Optimal	Proposed Algorithm		SMES		SR		α SIMPLEX	
		Mean	Worst	Mean	Worst	Mean	Worst	Mean	Worst
g01	-15.000	-15.000	-15.000	-15.000	-15.000	-15.000	-15.000	-15.000	-15.000
g02	-0.803619	**-0.803518**	**-0.803511**	-0.785238	-0.751322	-0.781975	-0.726288	-0.784187	-0.754259
g03	-1.001	-1.001	-1.001	-1.000	-1.000	-1.000	-1.000	-1.001	-1.001
g04	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539	-30665.539
g05	5126.497	5127.5433	5130.041	5174.492	5304.167	5128.881	5142.472	**5126.497**	**5126.497**
g06	-6961.814	-6961.814	-6961.814	-6961.284	-6952.482	-6961.814	-6350.262	-6961.814	-6961.814
g07	24.306	24.306	24.306	24.475	24.843	24.374	24.642	24.306	24.306
g08	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825	-0.095825
g09	680.630	680.630	680.630	680.643	680.719	680.656	680.763	680.630	680.630
g10	7049.248	7077.6821	7189.5695	7253.047	7638.366	7559.192	8835.655	**7049.248**	**7049.248**
g11	0.750	0.750	0.750	0.750	0.750	0.750	0.750	0.750	0.750
g12	-1.000	-1.000	-1.000	-1.000	-1.000	-1.000	-1.000	-1.000	-1.000
g13	0.053942	**0.053942**	**0.053942**	0.166385	0.468294	0.067543	0.216915	0.066770	0.438803

1.5 Co-Evolutionary Penalty Function

- A method based on co-evolution was proposed which involved two different populations. The fitness function was determined as

$$\text{fitness}(X) = f(X) + \left(\omega_1 \times \text{coef}(X) + \omega_2 \times \text{viol}(X)\right)$$

$$\text{coef}(X) = \sum_{j=1}^{m} c_j(X)$$

$$\text{viol}(X) \Rightarrow \left|\text{violated constraints}\right|$$

- Hence the fitness scheme involves minimization of both the number of violated constraints as well as the magnitude of the constraint violations. w_1 and w_2 are the weighting coefficients.

- As part of the co-evolutionary algorithm, two populations *P1* and *P2* of size *M1* and *M2* respectively are used.
- The second of these populations (*P2*) encoded the set of weight combinations (w_1 and w_2) that would be used to compute the fitness of individuals in *P1* .
- Thus, one population was used to evolve the solutions while the other was used to evolve the penalty coefficients.

- Each individual A_j in *P2* is decoded and the weight combination is used to evolve *P1* for a certain number of generations (*Gmax1*) .
- The fitness of each individual B_k is computed keeping the penalty factors constant for every individual instance of *P1* corresponding to the individual A_j being processed.
- The drawback with this method is the need to define the population sizes (*M1* and *M2*) and the pre specified number of generations (*Gmax1*).
- At the same time diversity in the solutions may be naturally maintained because of the two populations evolved simultaneously.

Domination Relationship

- Consider the simple form of a static penalty function,

$$\text{fitness}(X) = f(X) + \sum_{j=1}^{m} r_j c_j(X)$$

then the fitness of the solution may be decided by its constraint violation or objective function depending on the value of r_j .
- If the value of r_j is too high, then the fitness of all the individuals will be determined by the constraint violation and if it is too low then the objective function value will basically decide the fitness of each individual.
- So the value of r_j, plays a very important role in obtaining feasible and optimal solutions in the search using GA's.

Stochastic Ranking Scheme

- To overcome the problem of choosing an optimal r_j the authors propose introducing a probability factor P_f which determines if the objective function value or the constraint violation value will determine the rank of each individual.

- The ranking method incorporated sees to it that feasible solutions are ranked based only on their objective function, while the probability factor P_f determines whether objective function or constraint violation should be used to rank infeasible individuals.

- In the experiments done a probability factor of 0.45 produced good results. This implies that the constraint violation should be used to rank infeasible individuals (55%) more often than objective function (45%).

- While the method produced best results for all of the problems tested, there was one fundamental flaw.

- The method did not produce feasible solution for all the runs especially for a particular problem- only 7 out of 30 runs could produce feasible solutions itself. This can be attributed to the selection scheme in which constraint violation does not dominate the objective function even while ranking infeasible solutions.

- Even while a value of $P_f = 0.45$ is chosen, there is no way to come up with this value empirically. It is possible that better results can be obtained by changing the value with respect to the difficulty of each problem.

2. Feasible Solution over Infeasible Ones

- Treated as bi-objective optimization problem- the first would be the original objective function while the second is the constraint violation.
- A selection scheme was proposed which always prefers feasible solutions over infeasible ones.
- Hence all solutions are selected based on penalty term domination.
- Among feasible solutions, selection was based on objective function domination.
- Here again the penalty coefficients r_j are used based on the relative difficulty of satisfying each constraint but the decision of penalty function domination or objective function domination is based on the feasibility of the individual.

- The first method that incorporated the explicit domination of feasible over infeasible solutions proposed the following scheme for minimization problems,
  - For feasible solutions, the objective value is scaled into (-∞,1) and
  - For infeasible solutions the constraint violation of the solution is mapped into (1, ∞).
- The modified objective function for fitness allocation is calculated as,

$$obj(X)=\begin{cases} S(f(X)), & \text{if } \sum_{j=1}^{m} c_j(X)=0 \\ 1+\mathrm{r}\times\sum_{j=1}^{m} c_j(X), & \text{if } \sum_{j=1}^{m} c_j(X)\neq 0 \end{cases}$$

where *S* is a function that maps *f* into the open interval of (-∞,1)

- This method has some interesting properties
  - *as long as feasible solutions are not found, the objective function will make no effect on the rank of the individual so the initial part of the search will be directed towards finding feasible solutions,*
  - *once there is a combination of feasible and infeasible solutions in the population then feasible solutions will be ranked ahead of all infeasible solutions*
  - *feasible solutions will be ranked based on their objective function values.*

- The major drawback which we could experience in this method is a lack of diversity operators either explicitly defined or as part of the selection scheme. This could cause problems especially in problems with disconnected feasible components in which case the GA may be stuck within one of the feasible components and never get a chance to explore.

Deb's Method

- The same idea of domination was extended in Deb' method which uses binary tournament selection to compare two individuals and a niching scheme to maintain diversity in the population. In addition, the penalty coefficients of the constraints are unity. Hence the fitness function is defined as,

$$\boxed{\mathrm{Obj}(X) = \begin{cases} f(X), & \text{if } X \text{ is feasible} \\ f_{\max} + \sum_{j=1}^{m} c_j(X), & \text{otherwise} \end{cases}}$$

- So, when comparing two individuals,
  - o *any feasible solution wins over any infeasible solutions,*
  - o *two feasible solutions are compared only based on their objective function values,*
  - o *two infeasible solutions are compared based on the amount of their constraint violations, and*
  - o two feasible solutions *i* and *j* are compared only if they are within a critical distance otherwise another solution *j* is checked.

3. Method Based on Special Operators

- Methods that use special operators can only be applied to particular class of problems, but they may be more efficient in this class.
- One of the examples of methods that use specialized operators is the GENOCOP, which is a method that can be used if the feasible space is convex.
- Hence it can be applied to all linear constrained problems and problems with nonlinear constraints that have convex feasible regions.

GENOCOP

- Genetic Algorithm for Numerical Optimization of Constrained Problems

- This method uses specialized operators that maintain the feasibility of the individuals, i.e., operators that are closed on the feasible part $\mathcal{F}$ of the search space.

- The method assumes *linear constraints* only and a *feasible starting point* (or feasible initial population).

- Linear equations are used to eliminate some variable; they are replaced as a linear combination of other variables. Linear inequalities are updated accordingly.

- Mutation is carried out so that the mutated solution is part of the feasible domain of the problem.
- Similarly for two feasible individuals *X* and *Y* and then $0 \le a \le 1$ will always give rise to a feasible solution.

$$\boxed{aX + (1-a)Y}$$

- Note that linear constraints imply convexity of the feasible search space, which usually requires a lesser effort than for non-convex and/or disjoint feasible space.
- The weakness of the method lies in its inability to search in nonconvex spaces.

4. Methods Based on Decoders

- In this case, a chromosome “gives instructions” on how to build a feasible solution.
- Each decoder imposes a relationship *T* between a feasible solution and a decoded solution.
- The following 2 conditions are usually satisfied for good decoders,
  - o the transformation *T* is computationally fast and
  - o it has locality feature such that small changes in decoded solution result in small changes to the solution itself.
- A homomorphous mapping was achieved between the *n*-dimensional search space and a feasible search space based on the following criteria
  - o for each solution *s* there must be a decoded feasible solution *d*.
  - o each decoded solution *d* must correspond to a feasible solution *s*.

- This method includes an additional problem-dependent parameter to partition the interval [0, 1] into subintervals of equal length such that the equation of each constraint has, at most one solution in each subinterval.
- The disadvantage with the homomorphous mapping is that it requires an initial feasible solution and that all infeasible solutions are rejected.
- Also, the locality feature is violated in non-convex search spaces

5. Repair Algorithms

- In repair algorithms, the idea is to make an infeasible solution feasible by repairing it.
- This is incorporated in the GENOCOP-III. The idea is to incorporate the original GENOCOP system that handles linear constraints only and extend it by maintaining two separate populations, where results in one population influence evaluation of individuals in the other population.
- The first population consists of points which satisfy the linear constraint of the problem: the feasibility of these points is maintained by specialized operators.

- The second population consists of feasible reference points. Since these reference points are already feasible, they are evaluated directly by the objective function, whereas search points are repaired for evaluation.

- This approach is helpful if it is easy to repair an infeasible solution to a feasible solution but otherwise the repair operators produce a strong bias in the search harming the evolutionary process.

6. Methods Based on MOP

- The main idea is to redefine the single-objective optimization of $f(X)$ as a *m+1* multiobjective optimization problem, where *m* is the total number of constraints.
- Any multiobjective technique can be applied to the above problem and minimized to drive the solutions to feasibility and optimality.
- An ideal solution would thus have $f_j(X)=0$ for $j=1,\ldots,m$ and lowest obtained objective function value $f(X)$.

COMOGA Method

- Solutions are initially ranked based on their non-dominated rank of the constraint violations and separately on their objective function values.
- The vector evaluated genetic algorithm (VEGA) type approach was used where in a proportion of the population was selected based on the constraint violation rank while the others are chosen based on their objective value rank.
- A drawback of the method is the high computational complexity involved in this method to find the non-dominated rank of the constraint violation increases with the number of constraints. In order to overcome this particular problem another method was proposed.

Coello's Method

- Here, non-domination ranking is not used but the population is divided into various subpopulations like the traditional VEGA.
- So, there are *m+1* subpopulations in a problem with *m* constraints. The first subpopulation is used to optimize for the objective function value only.
- If a solution does not violate the constraint corresponding to the subpopulation but is infeasible, then the subpopulation will minimize the total number of violations.
- Finally, once a solution becomes feasible, then it will be merged with the first subpopulation and look to minimize the objective function.

Summary

- No single method has been proven to be good for all types of problem (i.e., *No Free Lunch* theorem).
- Static penalty method is the most popular one.
- Dynamic penalty method outperforms static penalty method but has too many parameters to tune.
- Remained to be an open area of research.

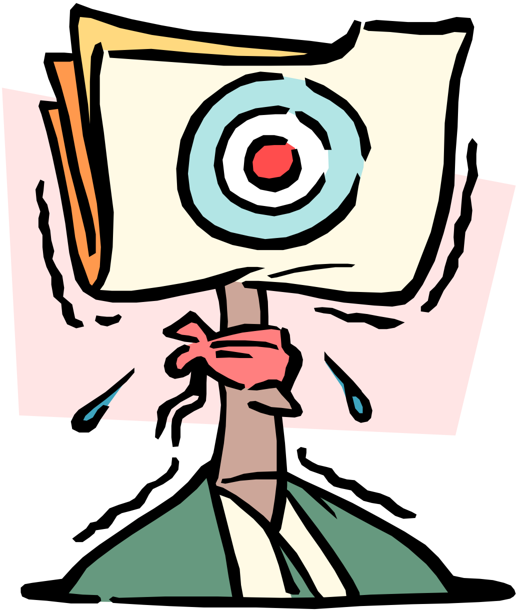
Critical Message Conveyed

Solving

Constrained Optimization

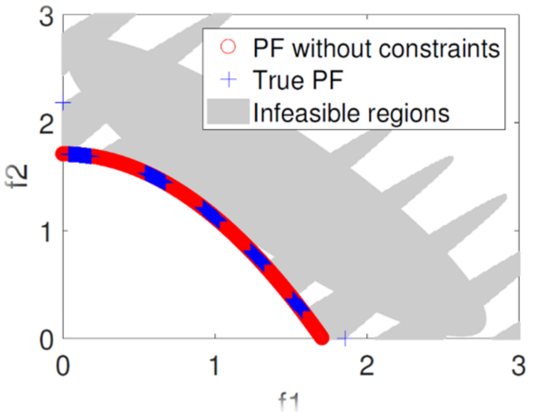
practically is one step closer to address real-world complications.

Q&A



12. Constraint Optimization in Multi-Objective Evolutionary Algorithms

约束多目标优化



Constrained MOEAs

$$
\begin{aligned}
\text{Optimize} \quad & f_i(x) = f_i(x_1, x_2, \ldots, x_n),\ i = 1,2,\ldots,p \\
\text{subject to} \quad & g_i(x) = g_i(x_1, x_2, \ldots, x_n) < 0,\ i = 1,\ldots,q \\
& h_i(x) = h_i(x_1, x_2, \ldots, x_n) = 0,\ i = q+1,\ldots,m \\
& x_j^{\min} \le x_j \le x_j^{\max},\ j = 1,2,\ldots,n \\
& x \in F \subseteq S \subseteq R^n
\end{aligned}
$$

- Challenges in constrained multi-objective optimization
  - various limits on the decision variables,
  - the constraints involved,
  - the interference among constraints
  - the interrelationship between the constraints and the objective functions
  - simultaneous optimization of a set of competing objectives

Proposed CMOEA

- extends the algorithm proposed by *Tessema and Yen* into the multi-objective cases
- uses modified objective function values for checking dominance in the population
- The modified objective value has two components
  - distance measure
  - adaptive penalty function

"Constraint handling in multi-objective evolutionary optimization," Woldesenbet Y.G., Yen G.G. and Tessema B.G., ***IEEE Transactions on Evolutionary Computation***, 13(3), 2009, pp. 514-525.

- Calculate sum of constraint violation as: $v(x) = \frac{1}{m} \sum_{j=1}^{m} \frac{c_j(x)}{c_{\max}^j}$

  where

$$c_j(x) = \begin{cases} \max(0, g_j(x)) & j = 1, \ldots, q \\ \max(0, |h_j(x)| - \delta) & j = q+1, \ldots, m \end{cases}$$

$$c_{\max}^j = \max_x c_j(x)$$

- Find $f_{\min}^i = \min_x f_i(x)$ and $f_{\max}^i = \max_x f_i(x)$
- Calculate *normalized* fitness as follows: $\tilde{f}_i(x) = \frac{f_i(x) - f_{\min}^i}{f_{\max}^i - f_{\min}^i}$
- Finally calculate ***distance*** as:

$$d_i(x) = \begin{cases} v(x), & \text{if } r_f = 0 \\ \sqrt{\tilde{f}_i(x)^2 + v(x)^2}, & \text{otherwise} \end{cases}$$

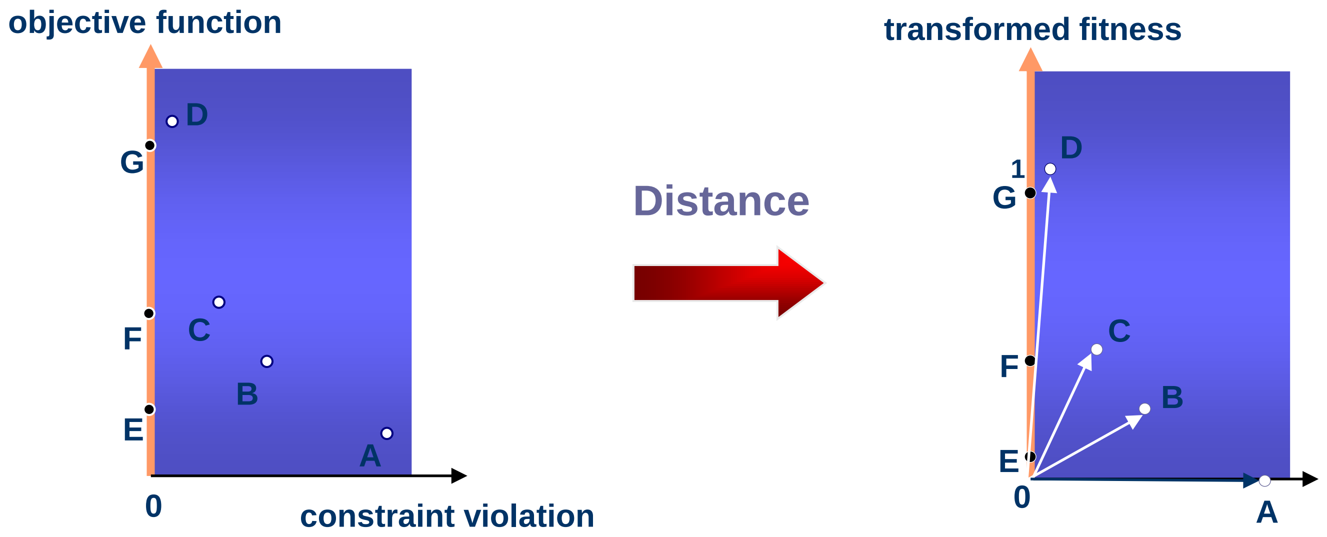
where

$$r_f = \frac{\text{number of feasible individuals in current population}}{\text{population size}}$$

[redacted]

1) For a feasible individual x, the distance value in a given objective function dimension i is equal to $\tilde{f}_i(x)$. Hence, those feasible individuals with smaller objective function values will have smaller distance values in that given dimension.
2) For infeasible individuals, the distance value has two components: the objective function value and the constraint violation. Hence, individuals closer to the origin in the $\tilde{f}_i(x) - v(x)$ space would have lower distance value in that objective function dimension than those farther away from the origin.
3) If we compare the distance values of infeasible and feasible individuals, then either one may have a smaller value. However, if the two individuals have similar objective function values, then the feasible individual will have a smaller distance value in the corresponding objective function dimension.

- Distance Calculation



- Penalty formulation $$p_i(x) = (1 - r_f)X_i(x) + r_f Y_i(x)$$

where

$$X_i(x) = \begin{cases} 0, & \text{if } r_f = 0 \\ v(x), & \text{otherwise} \end{cases}$$

$$Y_i(x) = \begin{cases} 0, & \text{if } x \text{ is a feasible individual} \\ \tilde{f}_i(x), & \text{if } x \text{ is an infeasible individual} \end{cases}.$$

- Major purposes:
  - To further reduce the fitness of infeasible individuals.
  - To identify the best infeasible individuals in the population
- Modified objective function formulation

$$F_i(x) = d_i(x) + p_i(x).$$

$$p_i(x) = (1 - r_f)X_i(x) + r_f Y_i(x)$$

$$X_i(x) = \begin{cases} 0, & \text{if } r_f = 0 \\ v(x), & \text{otherwise} \end{cases}$$

$$Y_i(x) = \begin{cases} 0\,, & \text{if } x \text{ is a feasible individual} \\ \tilde{f}_i(x)\,, & \text{if } x \text{ is an infeasible individual} \end{cases}.$$

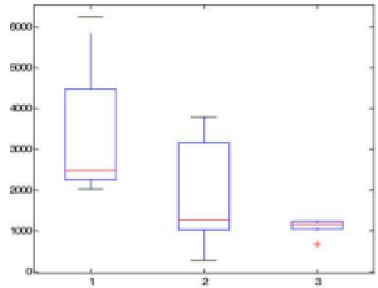
In Summary

- If the feasibility ratio of the population is small (but not zero), then the first penalty ($X_i(x)$) will have more impact than the second penalty($Y_i\,(x)$). The first penalty is formulated to have larger value for individuals with large amount of constraint violation. Hence, in the case when there are few feasible individuals present in the population, infeasible individuals with higher constraint violation will be more penalized than those with lower constraint violation.
- On the other hand, if there are many feasible individuals in the population. The second penalty will have more effect than the first. In this case, infeasible individuals with larger objective function value will be more penalized than infeasible individuals with smaller objective function value.
- Additionally, if there is no feasible individual in the population, both penalty terms will be zero.
- The two components of the penalty function allow the algorithm to switch between finding more feasible solutions and finding better optimal

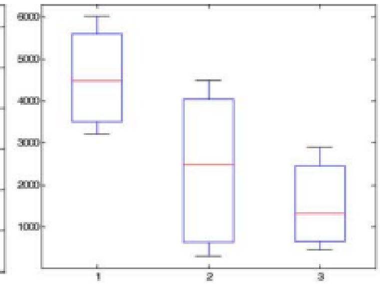
Benchmark Problems

- Fourteen constrained multi-objective test problems that represent a diverse set of functions that resemble real world problems:
  - Linear, nonlinear, quadratic objective functions and constraints
  - different feasible regions

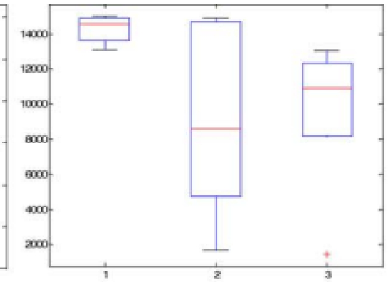
Function name	Objective functions	Decision dimensions	Feasibility ratio (ρ)	Constraints				
				Inequality	Equality	Linear	Non-linear	Active
BNH	2	2	93.61%	2	0	0	2	0
SRN	2	2	16.18%	2	0	1	1	0
OSY	2	6	3.25%	6	0	4	2	3
TNK	2	2	5.09%	2	0	0	2	1
CTP1	2	2	99.58%	2	0	0	2	1
CTP2	2	2	78.65%	1	0	0	1	1
CTP3	2	2	76.85%	1	0	0	1	1
CTP4	2	2	58.17%	1	0	0	1	1
CTP5	2	2	77.54%	1	0	0	1	1
CTP6	2	2	0.40%	1	0	0	1	1
CTP7	2	2	36.68%	1	0	0	1	0
CTP8	2	2	17.83%	2	0	0	2	1
CONSTR	2	2	52.52%	2	0	2	0	1
Welded Beam	2	4	18.67%	4	0	1	3	0



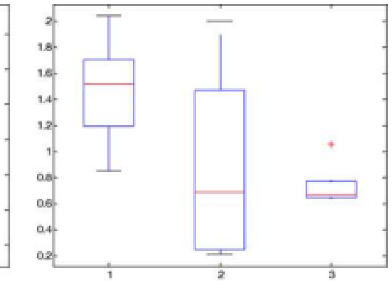
I_H values for BNH



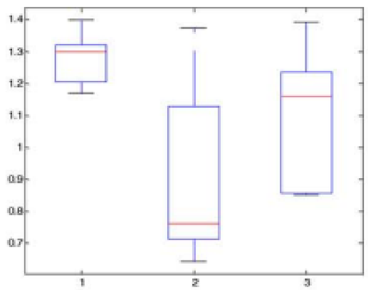
I_H values for SRN



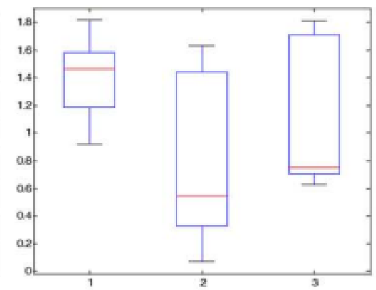
I_H values for OSY



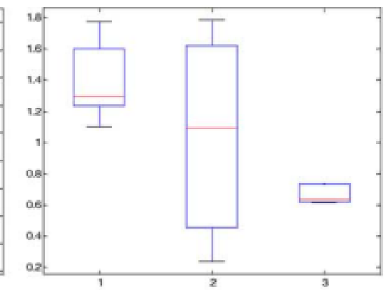
I_H values for TNK



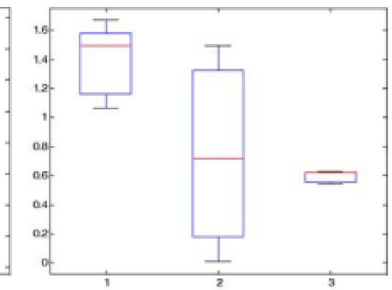
I_H values for CTP1



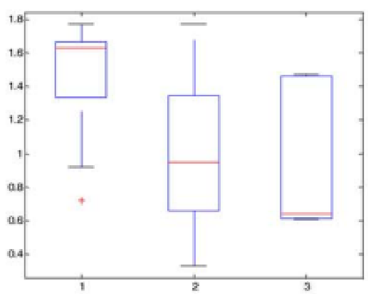
I_H values for CTP2



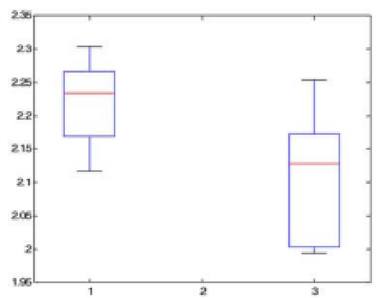
I_H values for CTP3



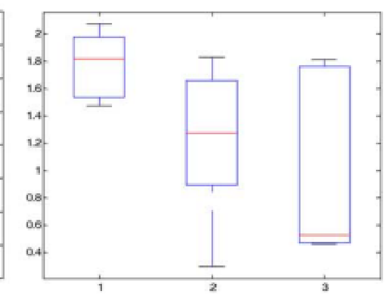
I_H values for CTP4



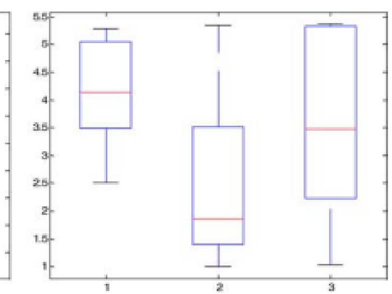
I_H values for CTP5



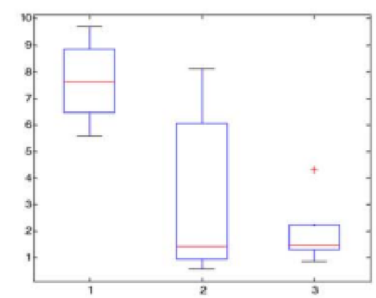
I_H values for CTP6



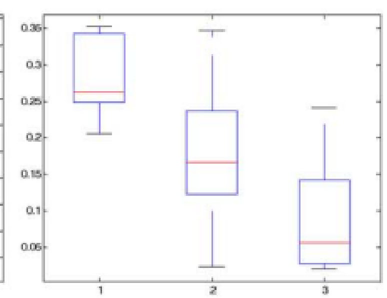
I_H values for CTP7



I_H values for CTP8



I_H values for CONSTR



I_H values for Welded Beam

- If few feasible individuals are present, then those infeasible individuals with higher constraint violation are penalized more than those with lower constraint violations.
- On the other hand, if a sufficient number of feasible individuals exists, then those infeasible individuals with worse objective values are penalized more than those with better objective values.
- However, if the number of feasible individuals is in the middle of the two extreme, then the individual with lower constraint violation and better objective function is less penalized.
- The two components of the penalty function allow the algorithm to switch between feasibility and optimality.
- Since priority is initially given to finding feasible individuals before searching for optimal solutions, the algorithm is capable of finding feasible solutions when the feasible space is very small compared to the search space.

1) If there is no feasible individual in the current population, each $d_i(x)$ will be equal to the constraint violation $(v(x))$, and each $p_i(x)$ term will be zero. In this case, the objective values of the individuals will be totally disregarded, and all individuals will be compared based on their constraint violation only. This will help us find feasible individuals before looking for optimal solutions.
2) If there are feasible individuals in the population, then individuals with both low objective function values and low constraint violation values will dominate individuals with high objective function values or high constraint violation or both.
3) If two individuals have equal or very close distance values, then the penalty term $(p_i(x))$ determines the dominant individual. According to our penalty formulation, if the feasibility ratio (r_f) in the population is small, then the individual closer to the feasible space will be dominant. On the other hand, the individual with smaller objective function values will be dominant. Otherwise, the two individuals will be nondominant solutions.
4) If there is no infeasible individual in the population $(r_f = 1)$, then individuals will be compared based on their objective function values alone.

Performance Indexes

- Convergence metric:- obtained by calculating the smallest normalized Euclidean distance between the non-dominated set and the true Pareto-front
  - $$\gamma = \frac{\sum_{k \in P^*} L_k}{|P^*|} \quad \text{where} \quad L_k = \min_{j \in T} \sqrt{\sum_{i=1}^{p} \left(\frac{f_i(k) - f_i(j)}{f_i^{\max} - f_i^{\min}} \right)^2}$$
- Diversity Metric:- The diversity metric measures the extent of spread achieved among the obtained solutions

$$\Delta = \frac{L_f + L_l + \sum_{k=1}^{N-1} \left| L_i - \overline{L} \right|}{L_f + L_l + (N-1)\overline{L}}$$

Convergence Metric

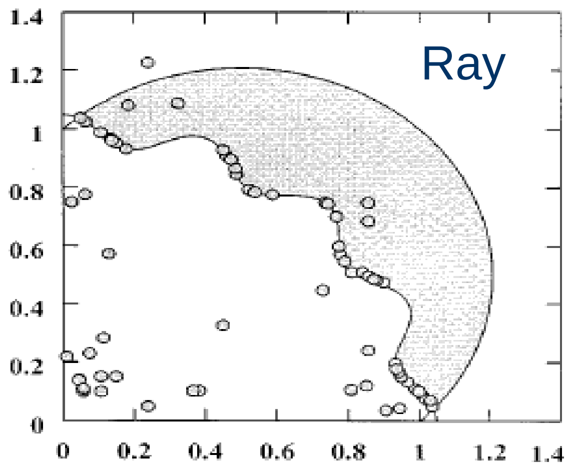
	OSY	BNH	CTP1	CTP2	CTP3	CTP4	CTP5
Avg	0.0056	0.0038	0.0014	9.3e-4	0.0074	0.0259	0.0017
Var	9.7e-6	2.5e-5	3.5e-5	4.3e-4	3.4e-4	4.1e-4	3.6e-4

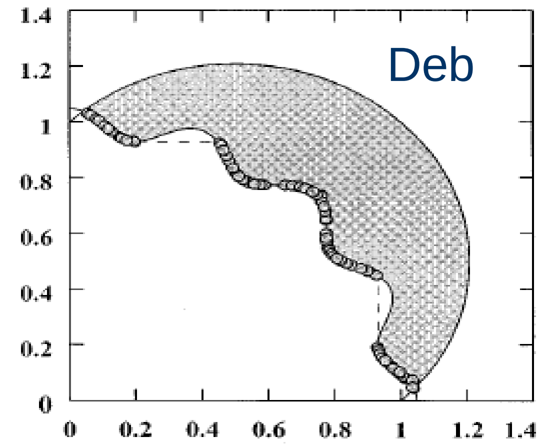
	CTP6	CTP7	CTP8	CONSTR	SRN	TNK
Avg	0.0039	9.6e-4	7.6e-4	0.0049	0.0016	0.0053
Var	4.2e-4	4.6e-4	4.9e-4	3.3e-5	3.1e-5	2.9e-4

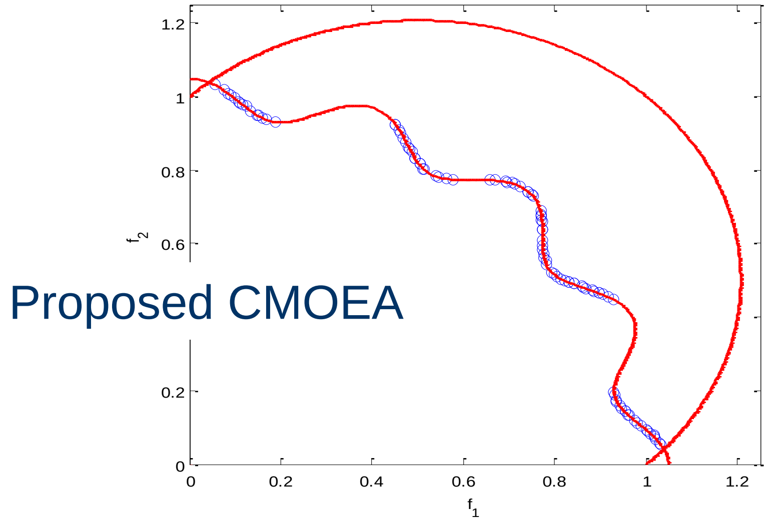
Divergence Metric

	OSY	BNH	CTP1	CTP2	CTP3	CTP4	CTP5
Avg	0.6879	0.6162	0.5827	0.6389	0.9032	0.8993	0.9182
Var	0.0422	0.0502	0.0459	0.0479	0.0497	0.0508	0.0517
	CTP6	**CTP7**	**CTP8**	**CONSTR**	**SRN**	**TNK**	
Avg	0.5632	0.6602	0.6368	0.6841	0.5792	0.7461	
Var	0.0513	0.0534	0.0595	0.0224	0.0381	0.0529	

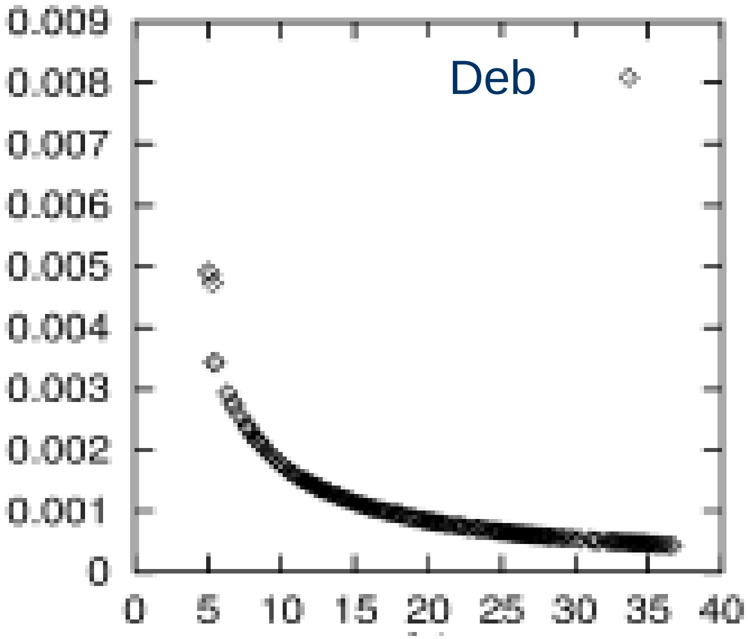
TNK Test Problem

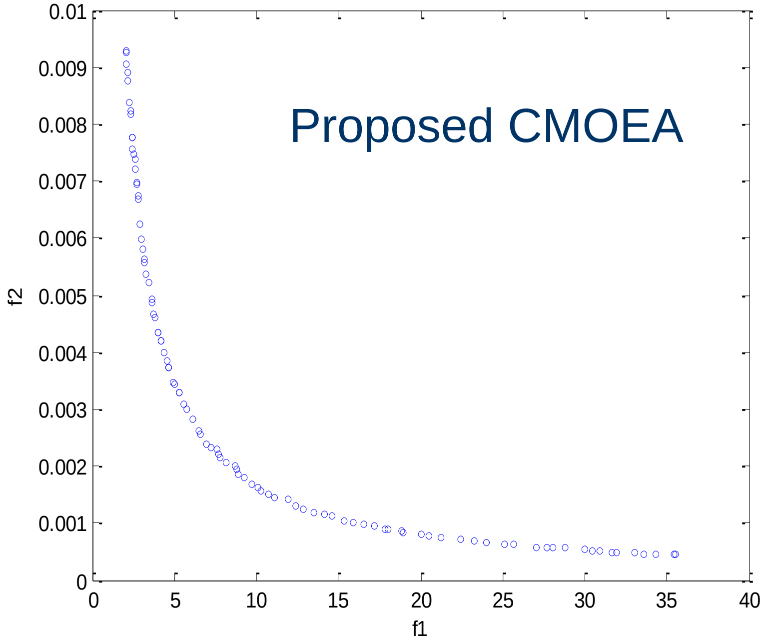






Welded Beam Test Problem





Summary

- Constrained multi-objective EA
  - Easy to implement
  - No parameter tuning in the constraint handling design
  - Better exploration of the information hidden in infeasible individuals
  - Shown to provide better results

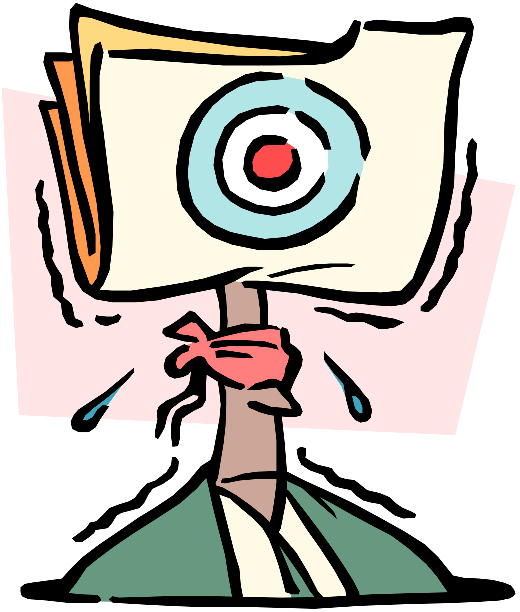
Critical Message Conveyed

An effective and efficient

Constrained Optimization in MOEAs

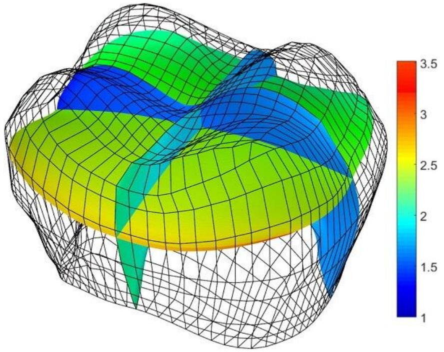
presents an opportunity in solving Real-World Problems.

Q&A



13. Soft Constraint Optimization in Evolutionary Algorithms

软约束优化

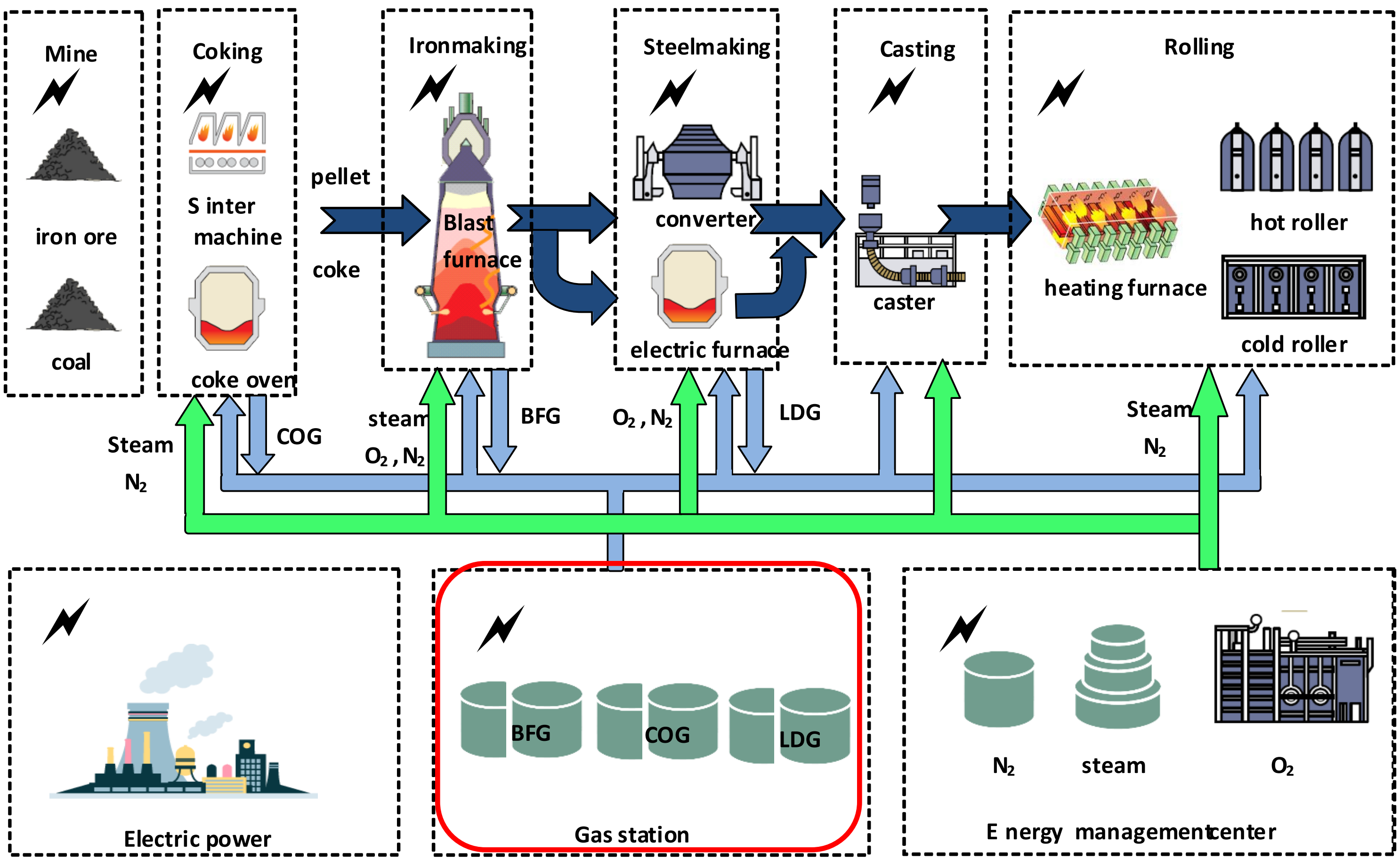


Motivations

- In China, the price of primary energy has risen up to ***2.5 times*** than that of 10 years ago. Expectedly, ***reducing energy cost*** in energy-intensive industry has been an important means to control total production cost.
- In iron and steel plant, energy refers to materials including electricity, gas, oxygen, steam, and so on that guarantee smooth production and accounts for 25-30% of the total production cost.
- One of the effective means to realize energy saving is through ***optimal scheduling and distributing energy*** to guarantee safe and economical energy supply.
- This problem is challenging because energy consumption, emission and regeneration occur simultaneously.
- The different color denotes the flow of a certain type of energy. The ‘up’ arrows refer to energy consumption, while the ‘down’ arrows denote energy regeneration.

BAOSTEEL

- ***Energy Distribution in an Iron & Steel Factory***



- Real-world investigations show that current, practical energy distribution plan is completed by often relaxing some negotiable constraints to find an ***acceptable*** plan in a ***timely*** manner.
- In the <u>amount-related constraints</u>, the violation with respect to higher price energy such as electricity will result in worse effect on the cost-based objectives than the lower price energy such as water because of the huge difference in consumption quantity.
- In the <u>holding capacity constraints</u> with respect to gas-like coke oven gas (COG), Linz-Donawitz gas (LDG), blast furnace gas (BFG), and oxygen that are transported through pipelines, constraints relaxation is not allowed because higher pressure than the designed specifications will lead to explosion.

- In the demand constraints, the violation in continuous process such as blast furnace ironmaking is not allowed because breaking off the blast furnace will lead to a serious production accident, while marginal violation of demand constraints in batch pro-duction such as rolling processes can be tolerated.

"Soft constraint handling for a real-world multi-objective energy distribution problem," Zhang Y., Yen G.G. and Tang L., ***International Journal on Production Research***, 58(19), 2020, pp. 6061-6077.

Objective

- Objective-

  To propose a framework for solving highly constrained real-world optimization problems and providing practical alternatives for decision makers.

- In an iron and steel plant, gas is one of the most important energy resources.
- There are three kinds of gas, COG, BFG, and LDG, which are generated along with the production in the process of coking, ironmaking, and steelmaking, respectively.
- The goal of gas distribution problem is to minimize the total energy cost by determining the amount of energy supply for all processes, considering constraints in energy supply, holding capacity, regeneration, and conversion. All constraints regarding these operations should be considered, resulting a highly COP.

To improve the energy utilization level, the goal of energy distribution problem is to minimize the total energy cost by determining the amount of energy supply for all processes, considering energy supply, holding capacity, regeneration, and conversion.

Objective functions to be minimized:

$$f_1 = \sum_{i=1}^{I} c_0 x_i$$ gas consumption cost

$$f_2 = c_1 z$$ gas purchase cost

$$f_3 = c_2 w$$ emission cost

$$f_4 = \max\{0, h(H + \sum_{i=1}^{I} y_i + z - \sum_{i=1}^{I} x_i - w)\}$$ gas inventory holding cost

Decision variables involved:

x_i	The amount of energy allocated to process i
y_i	The amount of secondary energy generated at process i
z	The purchase of energy
w	The emission of energy

Possible Disturbances:

sensor measurements

model uncertainties

fluctuation of energy market

Constraints Considered

$$\sum_{i=1}^{I} x_i + w \le \sum_{i} y_i + z + \left(H - H^0\right)$$

gas supply capacity constraint
regenerated secondary energy, the purchased energy, and the decrease of storage

$$x_i \ge \alpha_i^0 p_i, \ i = 1, 2, \ldots, I$$

lower bound gas consumption ratio constraint

$$x_i \le \alpha_i^1 p_i, \ i = 1, 2, \ldots, I$$

upper bound gas consumption ratio constraint

$$H + \sum_{i=1}^{I} y_i + z - \sum_{i=1}^{I} x_i - w \ge H^0$$

lower bound gas pipeline inventory capacity (pressure) constraint

$$H + \sum_{i=1}^{I} y_i + z - \sum_{i=1}^{I} x_i - w \le H^1$$

upper bound gas pipeline pressure constraint

$$y_i \ge \theta_i^0 p_i, \ i = 1, 2, \ldots, I$$

lower bound gas regeneration capacity constraint

$$y_i \le \theta_i^1 p_i, i = 1, 2, \ldots, I$$

upper bound gas regeneration capacity constraint

$$z \le Z, \textit{with } z = \sum_{i=1}^{I} x_i - \sum_{i=1}^{I} y_i - \left(H - H^0\right)$$

additional purchase constraint

$$w \le W, \textit{with } w = \sum_{i=1}^{I} y_i + \left(H^1 - H\right) - \sum_{i=1}^{I} x_i$$

gas emission limit constraint

$$x_i, y_i, z, w \ge 0, \ i = 1, 2, \ldots, I$$

nonnegative value fields

- The violation for constraint (1), the gas supply capacity constraint

$$G_1 = \max\left\{0, \sum_i x_i + w - \sum_i y_i - z - (H - H^0)\right\}$$

- The violation for combined constraints (2) and (3), gas consumption ratio constraints

$$G_2 = \sum_i \left(\max\{0, \alpha_i^0 p_i - x_i\} + \max\{0, x_i - \alpha_i^1 p_i\}\right)$$

- The violation for joint constraints (4) and (5), gas pipeline inventory capacity (pressure)

$$G_3 = \max\left\{0, \sum_i x_i + w + H^0 - H - \sum y_i - z\right\}$$
$$+ \max\left\{0, H + \sum_i y_i + z - \sum_i x_i - w - H^1\right\}$$

- Constraints (6) and (7) denote the gas regeneration capacity constraints. Under a normal production, the regeneration ratios of COG, BFG, and LDG remain at a relatively stable level. Considering the dynamic features in the practical production, the ratio often varies within an interval instead of being a fixed value. Because the amount of secondary energy relates to the production yield and the technological level of the corresponding process, these constraints can be viewed as ***soft*** ones.

$$G_4 = \sum_i (\max\{0, \theta_i^0 p_i - y_i\} + \max\{0, y_i - \theta_i^1 p_i\})$$

- Constraint (8) is an additional purchase constraint. As a major energy customer, an iron and steel plant usually signs a contract with energy suppliers, specifying the amount of energy required for the upcoming production cycle. The energy purchase beyond this contract is called additional purchase, which will be subjected to higher price than that specified in the contract. Additional energy purchase occurs when demand cannot be satisfied by storage and regeneration. In practice, there is a ***tolerance*** beyond the prespecified amount of additional purchase but will lead to a serious penalty.

$$G_5 = \max \left\{ 0, \sum_i x_i - \sum_i y_i - (H - H^0) - Z \right\}$$

- Constraint (9) describes gas emission limit. Emission occurs when the storage reaches the upper bound of pipeline pressure. Occasional emission is unavoidable especially under dynamic situation. Usually, there is a limit on the emission for each kind of energy. In practice, to ensure safety, emission beyond this limit by a tolerable range is permitted but will be penalized.

$$G_6 = \max\left\{0, \sum_i y_i + (H^1 - H) - \sum_i x_i - W\right\}$$

Recall: Constrained MOEAs

$$
\begin{array}{ll}
\text{Minimise} & f(x) = [f_1(x), f_2(x), \ldots, f_k(x),]^T \\
\text{subject to} & g_j(x) \leq 0 \; j = 1, 2, \ldots, l \\
 & h_j(x) = 0 \; j = l+1, l+2, \ldots, m \\
 & x_i^{\min} \leq x_i \leq x_i^{\max} \; i = 1, 2, \ldots, n
\end{array}
$$

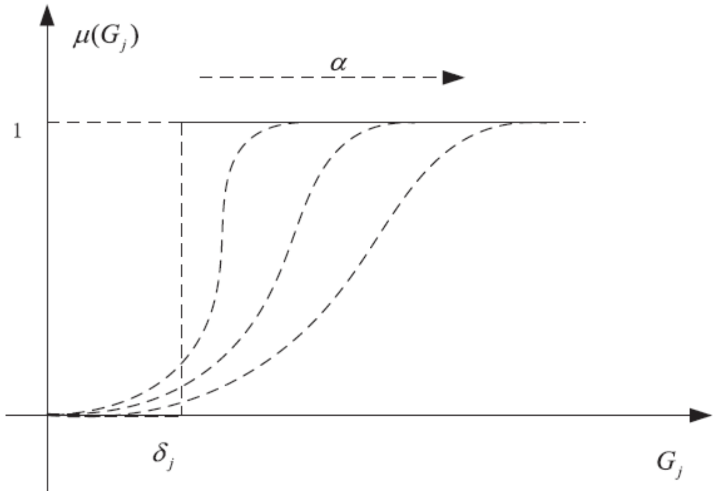
Violation on Constraint *j*

$$
G_j = \begin{cases} \max(0, g_j(x)) \text{ for } j = 1, 2, \ldots, l \\ \max(0, |h_j(x)| - \delta) \text{ for } j = l+1, l+2, \ldots, m \end{cases}
$$

Soft Constraint Violation

- To express the degree of violation of a soft constraint, a Gaussian function is used to quantify the satisfaction degree of constraint violation:

$$\mu(G_j) = \begin{cases} 0 \; G_j \leq \delta_j \\ 1 - e^{-(\frac{G_j - \delta_j}{\alpha})} \; G_j > \delta_j \end{cases}$$



Pseudocode

```
Set tolerance level β
For each p ∈ P , compute its fitness and constraint violation c_p
For each q ∈ P ,  q ≠ p , compute its fitness  and satisfaction degree c_q
If ( c_p = 0 )
  | If ( c_q = 0 )
  |  | get p based on fitness comparison
  | Else
  |    if ( c_q ≤ β )
  |     | get p based on fitness comparison
  |    Else
  |       p p q  //p dominates q
Else
   If ( c_p ≤ β )
    | If ( c_q ≤ β )
    |  | get p based on fitness comparison
    | Else
    |    p p q
   else  if ( c_q ≤ β ) then q p p //q dominates p
         else get p based on violation comparison
```

- Between two feasible solutions, the one with the better fitness value wins.
- If one solution is feasible and the other one is infeasible when the overall violation degree of the infeasible solution is less than a predefined tolerance level β, the solution with the better fitness value wins. Otherwise, the feasible solution wins.
- When both solutions are infeasible,
    - If both the overall violation are less than a predefined tolerance level β, the one with the better fitness value wins. In the case of non-dominated fitness value the solution smaller violation wins;
    - If the overall violation degrees of one solution is less than β while the other is greater than β, the one with smaller violation value wins;
    - If both the overall violations are greater than β, the one with smaller value of violations is preferred. In the case of equal violation degree, the solution with better fitness wins.

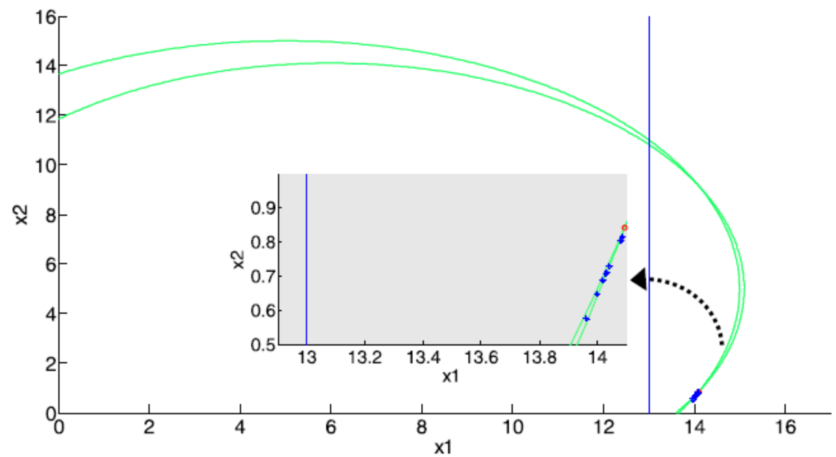
NSGA-II with Soft Constraints

Benchmark function, g6 (Michalewicz and Schoenauer 1996),

$$
\begin{aligned}
\text{Minimize} \quad & f(x) = (x_1 - 10)^3 + (x_2 - 20)^3 \\
\text{Subject to} \quad & g_1(x) = (x_1 - 5)^2 + (x_2 - 5)^2 - 100 \geq 0 \\
& g_2(x) = -(x_1 - 6)^2 - (x_2 - 5)^2 + 82.81 \geq 0 \\
& x_1 \in [13, 100], x_2 \in [0, 100]
\end{aligned}
$$

- It can be easily observed that in some cases, objective improvement can be achieved with only about one-tenth of constraint violation (e.g. solution (14.03915, 0.72594) acquires 1.9018% improvement at the cost of merely 0.1295% constraints violation).

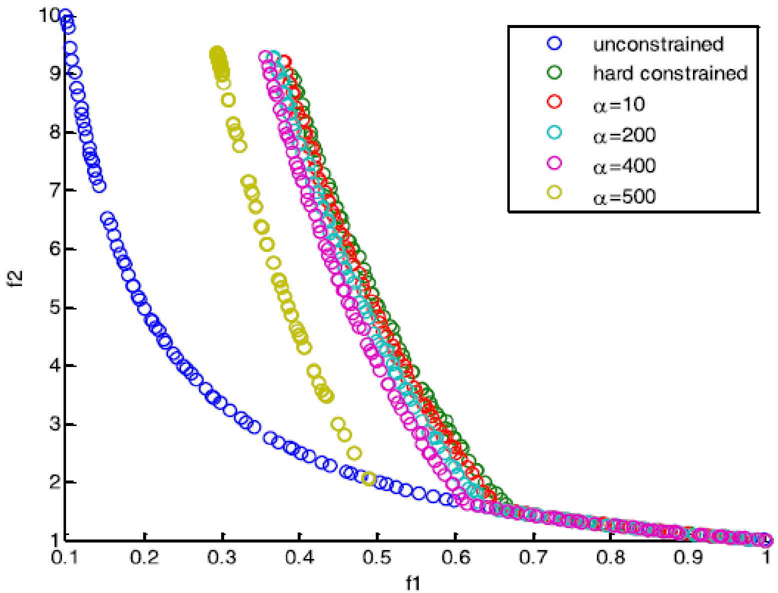
	x_1	x_2	$f(x)$	Overall penalty
Feasible	14.09500	0.84296	− 6961.81490	0.0000
Infeasible	14.03915	0.72594	− 7094.21094 (1.9018%)	0.11170 (0.1295%)
	14.03863	0.72667	− 7093.42236 (1.8904%)	0.11274 (0.1275%)
	14.07684	0.80787	− 7001.42627 (0.5690%)	0.03631 (0.0377%)
	14.07900	0.81529	− 6993.12109 (0.4497%)	0.03200 (0.0937%)
	14.07603	0.80674	− 7002.71289 (0.5875%)	0.03794 (0.0475%)
	14.06692	0.79146	− 7020.07617 (0.8369%)	0.05616 (0.1067%)
	14.06910	0.80664	− 7003.17432 (0.5941%)	0.05180 (0.3065%)
	14.01431	0.68990	− 7135.66211 (2.4972%)	0.16139 (0.1700%)

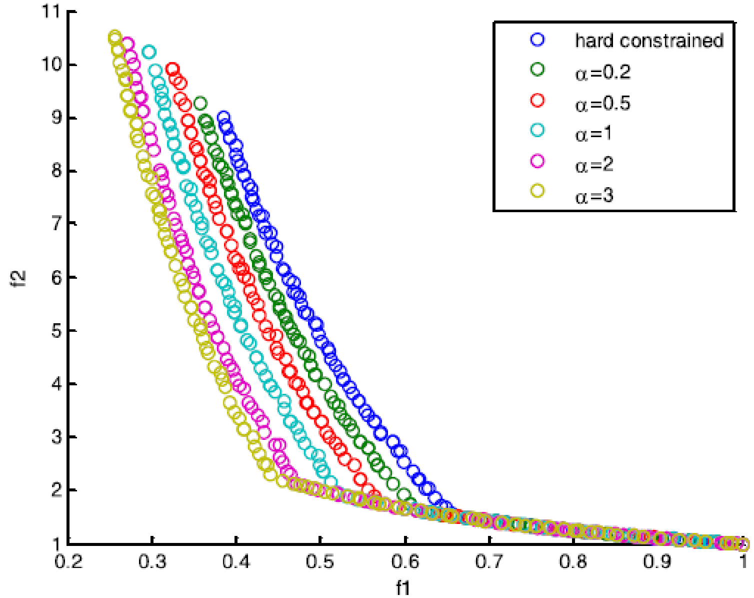


Benchmark Test Functions

CONSTER (Deb et al. 2002)

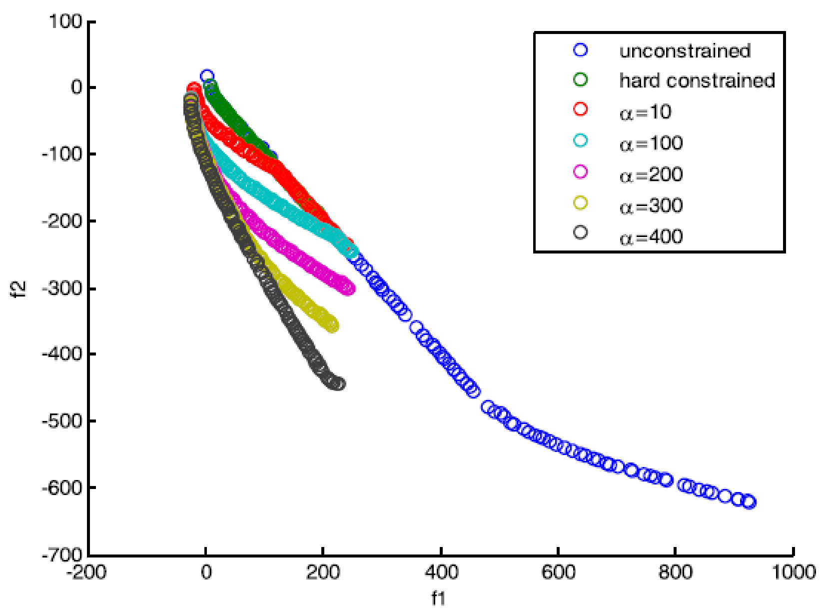
$$\begin{aligned} \text{Minimize} \quad & f_1(x) = x_1,\ f_2(x) = (1 + x_2)/x_1 \\ \text{subject to} \quad & g_1(x) = x_2 + 9x_1 \geq 6 \\ & g_2(x) = -x_2 + 9x_1 \geq 1 \\ & x_1 \in [0.1, 1.0], x_2 \in [0, 5] \end{aligned}$$

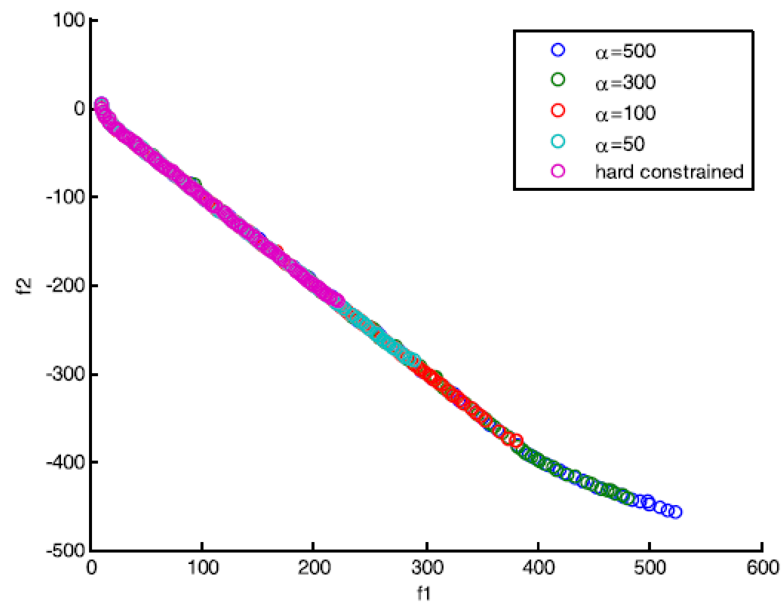




SRN (Deb, 2001)

$$\begin{array}{ll} \text{Minimise} & f_1(x) = (x_1 - 2)^2 + (x_2 - 1)^2 + 2,\ f_2(x) = 9x_1 - (x_2 - 1)^2 \\ \text{subject to} & g_1(x) = {x_1}^2 + {x_2}^2 \leq 225 \\ & g_2(x) = x_1 - 3x_2 \leq -10 \\ & x_1 \in [-20, 20], x_2 \in [-20, 20] \end{array}$$





- Considering the complex constraints imposed upon the gas distribution model, feasibility ratio test is first implemented by randomly sampling 1,000,000 data points and checking the feasibility. The test shows the percentage of feasible solutions is approximately ***4%*** for the problem at hand.

	f_1	f_2	f_3	f_4
Hard-constrained	184,904.2	0.000316	0.000004	0.00000
	184,902.0	0.000198	0.000125	0.00000
	184,901.6	0.000203	0.000125	0.00000
	179,591.3	0.012635	3656.074	0.00000
Soft-constrained	178,223.1	0.027014	2062.292	377.6878
	176,524.6	0.190921	1433.072	556.6623
	175,470.0	605.5200	0.021669	146.0952
	172,615.2	0.405488	6.338696	1364.969
	172,544.3	28.12651	7.413556	1010.731
	171,557.2	0.055235	4894.456	0.00000
	169,928.8	0.001742	3144.724	159.9545
	169,623.7	0.029134	568.1262	767.3773
	167,897.4	0.001049	2703.362	865.9019
	167,146.8	0.003139	3605.884	113.9985
	156,861.1	828.5714	1675.780	302.6916
	156,428.7	604.0353	1144.983	398.4023
	153,452.2	0.468527	2949.239	654.1028
	153,111.5	543.0526	1066.617	882.8607
	148,242.1	176.8478	44.63483	1689.571
	149,701.6	609.1605	10758.08	0.00000

- Recall that f_1 is the energy consumption cost, which is the direct component of production cost in real world. The hard-constrained results in Table show that in order to satisfy all the constraints in the model, f_1 is high in value. After the soft constraints are relaxed marginally, a lot of trade-off solutions have been obtained. Since the *pi* value is the production quantity per hour and *c*0, *c*1, and *c*2 are RMB price of gas in China, then from solution (184904.2, 0.000316, 0.000004, and 0.00000) to solution (179591.3, 0.012635, 3656.074, and 0.00000), the reduction of energy cost will be 39,763 RMB per day, which is a considerable energy saving for the iron and steel plant under study. When the constraints are further relaxed, the daily reduction of energy cost will increase accordingly.

Summary

- Under normal production, the energy allocation plan that well balances four objectives may be preferred.
- When rush orders arrive, the increase of energy consumption and additional purchase are usually unavoidable, the decision maker can choose the Pareto solution with tolerable relaxation on related constraints.
- When the inventory holding level is approaching the critical value, based on the safety consideration, the solutions with minimal inventory holding cost may be more welcomed by the decision maker.
- When dealing with some heavy polluted waste gas, the emission requirement of which will be more rigid than others, then the allocation plan with lower emission cost will be a preferential candidate solution.

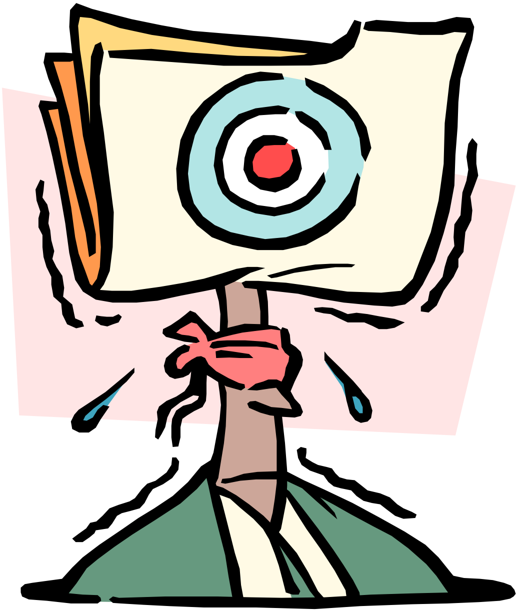
Critical Message Conveyed

A better understanding on
Real-World Complications presents

a great research opportunity, such as

Soft-Constraint Handling

Q&A



Future Research Directions

- Real-time implementation based on GPUs/FPGAs
- Computational constrained implementations via knee
- Theory-guided Machine Learning/AI/CI
- Data-driven Evolutionary Optimization
- Large-scale, Dynamic & Robust Optimization
- Continuous learning with reasoning cause-and-effect ability
- Real-world Applications…

The End