Natural Computing

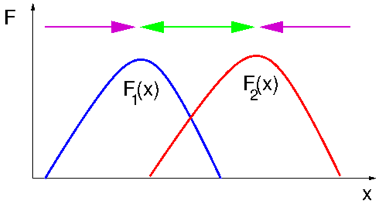
Lecture 7

Michael Herrmann
mherrman@inf.ed.ac.uk
phone: 0131 6 517177
Informatics Forum 1.42

11/10/2011

Multiobjective Optimisation by GAs, Evolution Strategies (ES) and Differential Evolution (DE)

Example: A machine is characterized by power and torque. A machine is better if – at equal torque – its power is higher.



Combination of fitness functions

$f(x) = |f_1(x)|^\alpha + |f_2(x)|^\alpha$

$f(x) = \alpha f_1(x) + (1-\alpha) f_2(x)$

How to set α?

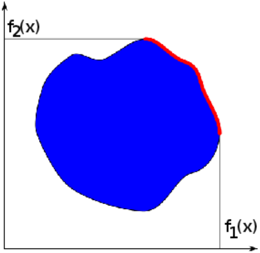
If α is not implied by the problem, any value in between the two maxima is equally good.

If a comparison between the two quantities is not possible, a set of solutions should be considered as optimal (Pareto-optimal).

How to optimise one criterion without loosing on other criteria?

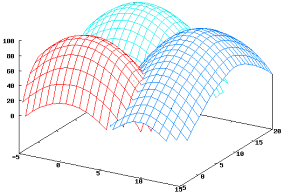
C. M. Fonseca & P. J. Fleming (1995) An Overview of Evolutionary Algorithms in Multiobjective Optimization. *Evolutionary Computation* **3**:1, 1-16.

Multiobjective Optimization

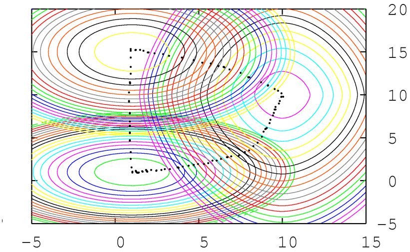


x^* is Pareto optimal for a class of fitness functions $\{f_i\}$ if there exists **no** $x \neq x^*$ with $f_i(x) \geq f_i(x^*)$ for all i

or, equivalently, x^* is not **dominated** by any other x : $\sim\exists x \succ x^*$ (more specifically $\sim\exists x \succ_{\{f_i\}} x^*$)



Example with three fitness functions



Same example: Pareto area spanned by maxima in a shape-dependent way

GA for Multiobjective Optimization

Benefits:

- Collective search required for sampling the Pareto set
- Non-connected Pareto sets are OK
- Incorporation of constraints in fitness function

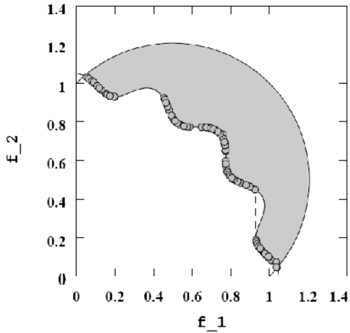
Problems:

- Selection of fit individuals?
- Elitism?
- Pareto-optimal diversity?
- Speed? (Pareto set can be high-dimensional)

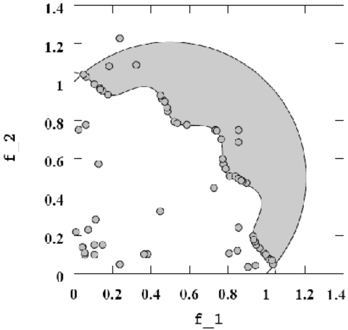
GA for Multiobjective Optimization

$f_1(x) = x_1,\ f_2(x) = x_2$ minimisation with constraints

$g_1(x)\colon\ x_1^2 + x_2^2 - \frac{1}{10}\cos\left(16\arctan\left(\frac{x}{y}\right)\right) \geq 1,\ g_2(x)\colon\ \left(x - \frac{1}{2}\right)^2 + \left(x_2 - \frac{1}{2}\right)^2 \leq \frac{1}{2}$



NSGA-II
(nondominated sorting GA)



conventional algorithm
(also GA-style)

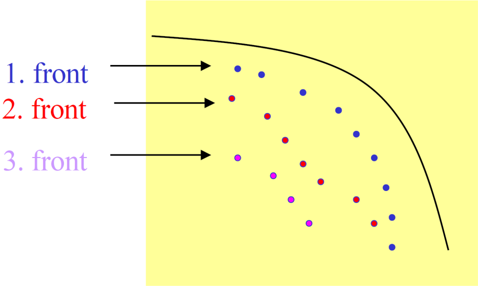
Kalyanmoy Deb, Amrit Pratap, Sameer Agarwal and T. Meyarivan (2000) A Fast Elitist Multi-Objective Genetic Algorithm: NSGA-II, IEEE Transact. Evolutionary Computation 6,182-197.

How does it work?

- Non-dominated-sorting genetic algorithm (NSGA)
- Selection by non-dominated sorting (M fitness functions)
- Preserving diversity along the non-dominated front
- Use two populations P and P' (each with N individuals)
- "being dominated by", denotes a partial order induced by a set of fitness functions

P'=find-nondomminated front(P)	
$P' = \{1\}$	include first member into P'
for each $p \in P \wedge p \notin P'$	take on solution at a time
$P' = P' \cup \{p\}$	temporarily include p into P'
for each $q \in P' \wedge q \neq p$	compare p to other members of P'
if $q \prec p$ then $P' = p' \setminus \{q\}$	if p dominates a member q of P' then delete q
else if $p \prec q$ then $P' = p' \setminus \{p\}$	if p is dominated by another member then do not include p in P'

Complexity per step: $O(MN^2)$



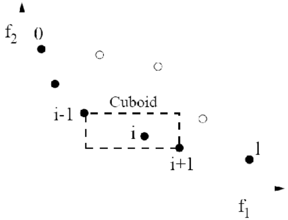
$\mathcal{F}$=fast-nondominated-sort(P); returns a set of nondominated fronts	
$i = 1$	i is the front counter
until $P \neq \emptyset$	temporarily include p into P'
$\mathcal{F}_i$=find-nondominated-front (P)	find the non-dominated front
$P = P \backslash \mathcal{F}_i$	remove nondominated solutions from P
$i = i + 1$	increment the front counter

Reserving density

New distance measure: first rank, then lowest density:

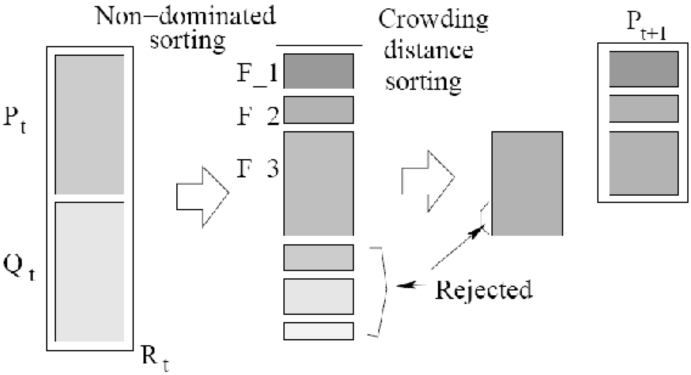
$i \succ_n j$ if $(i_{\text{rank}} < j_{\text{rank}})$ or

$((i_{\text{rank}} = j_{\text{rank}})$ and $i_{\text{dist}} > j_{\text{dist}})$



crowding-distance-assignment($\mathcal{I}$)	
$l = \{\mathcal{I}\}$	number of solutions in $\mathcal{I}$
for each i set $\mathcal{I}[i]_{\text{dist}} = 0$	initialise distance
for each objective m	temporarily include p into P'
$\mathcal{I} = \text{sort}(\mathcal{I}, m)$	sort using each objective value
$\mathcal{I}[1]_{\text{dist}} = \mathcal{I}[l]_{\text{dist}} = \infty$	so that boundary points are always selected
for $i = 2$ to $l - 1$	for all non-boundary points:
$\mathcal{I}[i]_{\text{dist}} = \mathcal{I}[i]_{\text{dist}} + (\mathcal{I}[i+1]_m - \mathcal{I}[i-1]_m)^2$	

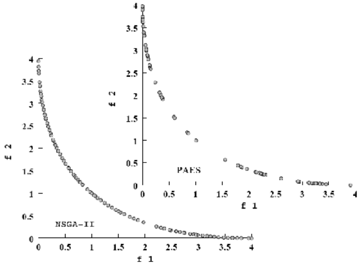
NSGA-II: Main Loop



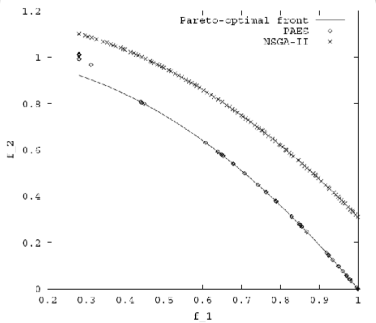
NSGA-II: Main Loop

$R_t = P_t \cup Q_t$	combine parents and children
$\mathcal{F}$=fast-nondominated-sort(R_t)	$\mathcal{F} = (\mathcal{F}_1, \mathcal{F}_2, \dots)$, all nondominated fronts of R_t
$P_{t+1} = \emptyset$ and $i = 1$	
until $\|P_{t+1}\| + \|\mathcal{F}_i\| \leq N$	till the parent population is filled
crowding-distance-assignment($\mathcal{F}_i$)	calculated crowing distance in $\mathcal{F}_i$
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i$	include the ith front into parent population
$i = i + 1$	check next front for inclusion
sort($\mathcal{F}_i, \prec_n$)	take part of the following front
$P_{t+1} = P_{t+1} \cup \mathcal{F}_i\,[1 : (N - \|P_{t+1}\|)]$	choose the first $(N - \|P_{t+1}\|)$ elements of $\mathcal{F}_i$
Q_{t+1} =make-new-pop(P_{t+1})	use selection, crossover and mutation to create a new population Q_{t+1} (standard GA)
$t = t + 1$	increment the generation counter

Performance



$$f_1(x) = x^2$$
$$f_2(x) = (x-2)^2$$



$$f_1(x) = x^2$$
$$f_2(x) = g(x)\left(1 - \sqrt{x_1/g(x)}\right)$$
$$g(x) = 1 + 10(n-1) + \sum_{i=2}^{n}\left(x_i^2 - 10\cos(4\pi x_i)\right)$$

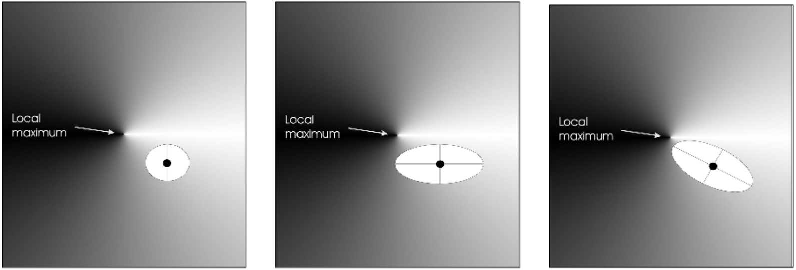
Left: Performance similar, NSGA-II has better distribution. Right: Even spread of the solution is a further goal that may compromise Pareto optimality of NSGA-II. (optimality is towards down and left)

For comparison: (1 parent, 1 child) Pareto-Archived Evolution Strategy (PAES) by Knowles and Corne (1999)

- Natural problem-dependent representation for search and optimisation (without “genetic” encoding)
- Individuals are vectors of real numbers which describe current solutions of the problem
- Recombination by exchange or averaging of components (but is often not used in ES)
- Mutation in continuous steps with adaptation of the mutation rate to account for different scales and correlations of the components
- Selection by fitness from various parent sets
- Variations of the algorithm: Elitism, islands, adaptation of parameters, ...

1964: Ingo Rechenberg; Hans-Paul Schwefel

Multidimensional Mutations in ES



Generation of offspring: $y = x + \mathcal{N}(0, C')$

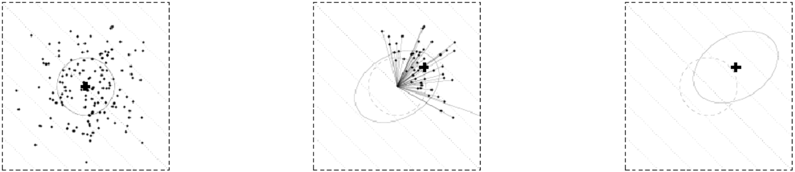
x stands for the vector $(x_1, \ldots, x_L)^\top$ describing a parent

C' is the covariance matrix C after mutation of the σ values where

- $C = \text{diag}(\sigma, ..., \sigma)$ for homogeneous uncorreleted mutations,
- $C = \text{diag}(\sigma_1, ..., \sigma_L)$ for scaled axes or
- $C = (C_{ij})$ for correlated mutations

A.E. Eiben and J.E. Smith, Introduction to Evolutionary Computing, 2008.

Multidimensional Mutations in ES



Off-spring vectors for parent m: $x_i := m + z_i, \quad z_i \sim \mathcal{N}(0, C)$

Select λ best [i.e. $(1, \lambda)$ - ES, see below]

Correlations among successful offspring: $Z := \frac{1}{\lambda}\Sigma_i z_i z_i^\top$

Update correlations: $C := (1 - \varepsilon)C + \varepsilon Z$

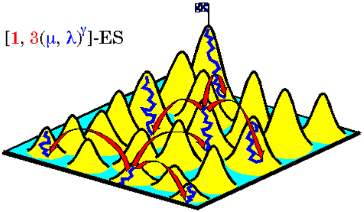
New state vector by averaging: $m := m + \frac{1}{\lambda}\Sigma_i z_i$

Smoothes fitness fluctuations; or: $m =$best

Heuristic 1/5 rule: If less than 1/5 of the children are better than their parents then decrease size of mutations

Nested Evolution Strategy

- Hills are not independently distributed (hills of hills)
- Find a local maximum as a start state
- Generate 3 offspring populations (founder populations) that then evolve in isolation
- Local hill-climbing (if convergent: increase diversity of offspring populations)
- Select only highest population
- Walking process from peak to peak within an "ordered hill scenery" named Meta-Evolution
- Takes the role of crossover in GA



http://www.bionik.tu-berlin.de/intseit2/xs2mulmo.html

- (μ, λ): From μ parents λ children (mutants) are generated. Selection only from the set of the λ children
- $(\mu + \lambda)$: Same as above, but selection from the set of μ parents plus λ children
- $(\mu', \lambda'(\mu, \lambda)^{\gamma})$: Hierarchical (nested) variant: From μ' parent sub-populations, λ' child-populations are generated. Then the children are isolated for γ generations where each time λ children are created (total population is $\lambda\lambda'$) and μ are selected. Then the best μ' subpopulations are selected and become parents for the new cycle of again γ generations
- Analogous: $(\mu' + \lambda'(\mu, \lambda)^{\gamma})$, $(\mu' + \lambda'(\mu + \lambda)^{\gamma})$, $(\mu', \lambda'(\mu + \lambda)^{\gamma})$

From Genetic Algorithms to Genetic Programming

- GA and GP are closely related fields
- Many of the empirical results discovered in one field apply to the other field, e.g. maintaining high diversity in a population improves performance
- GAs use a fixed-length linear representation GP uses a variable-size tree representation (variable size up to some bounds)
- Representations and genetic operators of GA and GP appear different (ultimately they are populations of bit strings in the computer's memory)
- An important difference lies in the interpretation of the representation: 1-to-1 mapping between the description of an object and the object itself (GA) or a many-to-1 mapping (GP)
- No-Free-Lunch theorem is valid for 1-to-1 mappings but not for many-to-1 mappings

Woodward (2003)

No-Free-Lunch Theorems

- Statement:
  - Averaged over all problems
  - for any performance metric related to number of distinct data points
  - all black-box algorithms will display the same performance
- Implications
  - If a new black box algorithm is good for one problem $\rightarrow$ it is probably poor for another one
  - There are as many deceptive as easy fitness functions (in large problems)
  - Makes sense not to use "black-box algorithms"
- Ongoing work showing counterexamples (given specific constraints or universes of problems or in co-evolutionary algorithms with self-play)