Evolution Strategies: Basic Introduction

Thomas Bäck
Leiden Institute of Advanced Computer Science (LIACS)
Niels Bohrweg 1, 2333 CA Leiden, The Netherlands
baeck@liacs.nl

http://www.sigevo.org/gecco-2012/

GECCO'12 Companion, July 7–11, 2012, Philadelphia, PA, USA.
ACM 978-1-4503-1178-6/12/07.

1

Abstract

This tutorial gives a basic introduction to **evolution strategies**, a class of evolutionary algorithms. Key features such as mutation, recombination and selection operators are explained, and specifically the concept of **self-adaptation** of strategy parameters is introduced.

All algorithmic concepts are explained to a level of detail such that an implementation of basic evolution strategies is possible.

Some guidelines for utilization as well as some application examples are given.

Biographical Sketch

Thomas Bäck received his PhD in Computer Science from Dortmund University, Germany, in 1994, and then worked for the Informatik Centrum Dortmund (ICD) as department leader of the Center for Applied Systems Analysis, and later for divis digital solutions GmbH as President and Chief Executive Officer.

From 1996 – 2004, Thomas was associate professor of Computer Science at Leiden University, and since 2004 he is full Professor of Computer Science at Leiden University. From 2000 - 2009, Thomas was CEO of NuTech Solutions GmbH and CTO of NuTech Solutions, Inc., until November 2009. Thomas has ample experience in working with Fortune 1000 customers such as Air Liquide, BMW Group, Beiersdorf, Daimler, Corning, Inc., Ford of Europe, Honda, Johnson & Johnson, P&G, Symrise, Siemens, Unilever, and others.

Thomas Bäck has more than 200 publications on evolutionary computation, as well as a book on evolutionary algorithms, entitled *Evolutionary Algorithms: Theory and Practice*. He is editorial board member and associate editor of a number of journals on evolutionary and natural computation, and has served as program chair for the major conferences in evolutionary computation. He received the best dissertation award from the Gesellschaft für Informatik (GI) in 1995 and is an elected fellow of the International Society for Genetic and Evolutionary Computation for his contributions to the field.

He is co-editor of the Handbook of Evolutionary Computation and the Handbook of Natural Computing (Springer, 2011).

Agenda

- Introduction: Optimization and EAs
- Evolution Strategies
- Examples

A True Story …

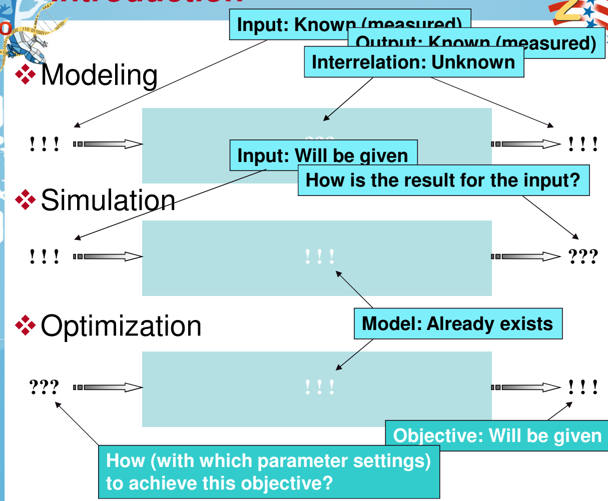
During my PhD

- Ran artificial test problems
- n=30 maximum dimensionality
- Evaluation took „no“ time
- No constraints
- Thought these were difficult

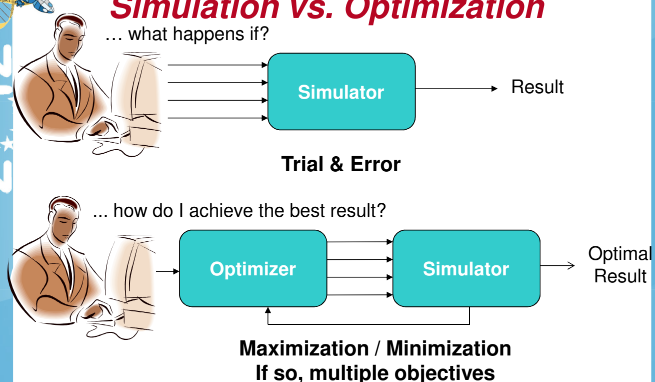
Now

- Real-world problems
- n=150, n=10,000
- Evaluation can take 20 hours
- 50 nonlinear constraints
- Tip of the iceberg

Introduction



Simulation vs. Optimization

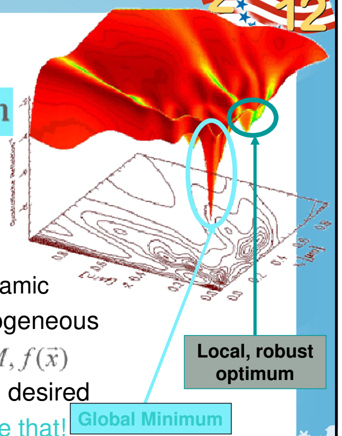


Introduction:

Optimization
Evolutionary Algorithms

Optimization

$f: M \to \Re, f(\vec{x}) \to \min$



- f : objective function
  - High-dimensional
  - Non-linear, multimodal
  - Discontinuous, noisy, dynamic
- $M \subseteq M_1 \times M_2 \times ... \times M_n$ heterogeneous
- Restrictions possible over $M, f(\vec{x})$
- Good local, robust optimum desired
- Realistic landscapes are like that!

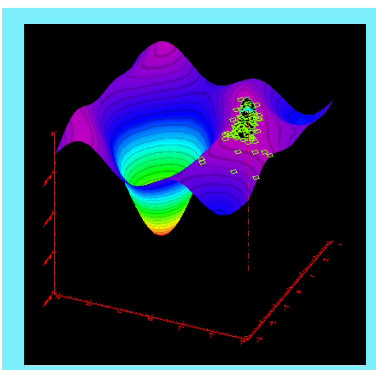
Optimization Creating Innovation

- Illustrative Example: Optimize Efficiency
  - Initial:

  - Evolution:

- 32% Improvement in Efficiency !

Dynamic Optimization



- Dynamic Function
- 30-dimensional
- 3D-Projection

Classification of Optimization Algorithms

- Direct optimization algorithm: Evolutionary Algorithms

$f(\vec{x})$

- First order optimization algorithm: e.g., gradient method

$f(\vec{x}), \nabla f(\vec{x})$

- Second order optimization algorithm: e.g., Newton method

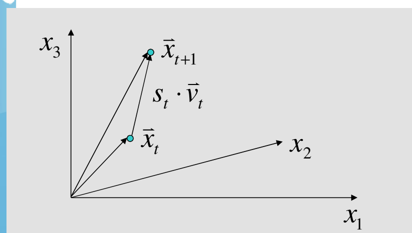
$f(\vec{x}), \nabla f(\vec{x}), \nabla^2 f(\vec{x})$

Iterative Optimization Methods

- General description:

$$\vec{x}_{t+1} = \vec{x}_t + s_t \cdot \vec{v}_t$$

New Point; Actual Point; Directional vector; Step size (scalar)



- At every Iteration:
  - Choose direction
  - Determine step size
- Direction:
  - Gradient
  - Random
- Step size:
  - 1-dim. optimization
  - Random
  - Self-adaptive

The Fundamental Challenge

- Global convergence with probability one:

$$\lim_{t\to\infty} \Pr(\vec{x}^* \in P(t)) = 1$$

- General, but for practical purposes useless
- Convergence velocity:

$$\varphi = E(f_{\max}(P(t+1)) - f_{\max}(P(t)))$$

- Local analysis only, specific (convex) functions

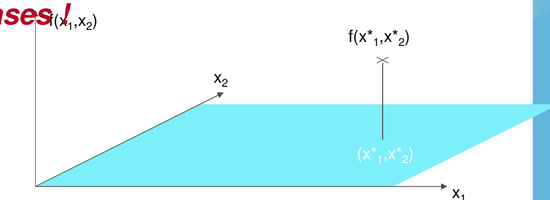
Theoretical Statements

- Global convergence (with probability 1):

$$\lim_{t\to\infty} \Pr(\vec{x}^* \in P(t)) = 1$$

- General statement (holds for all functions)
- Useless for practical situations:
  - Time plays a major role in practice
  - Not all objective functions are relevant in practice

An Infinite Number of Pathological Cases !



- NFL-Theorem:
  - All optimization algorithms perform equally well iff performance is averaged over all possible optimization problems.
- Fortunately: We are not Interested in „all possible problems“

Theoretical Statements

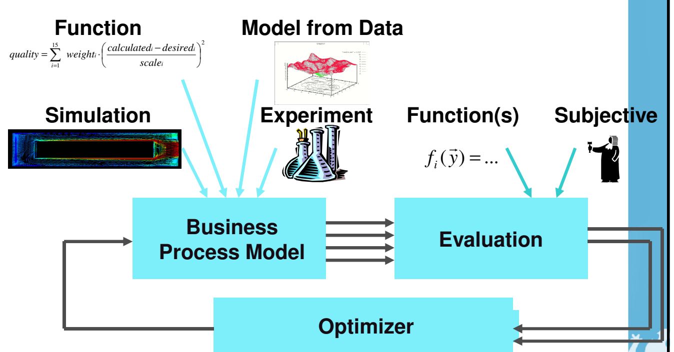
- Convergence velocity:

$$\varphi = E(f_{\max}(P(t+1)) - f_{\max}(P(t)))$$

- Very specific statements
  - Convex objective functions
  - Describes convergence in local optima
  - Very extensive analysis for Evolution Strategies

Evolution Strategies

Model-Optimization-Action



Evolutionary Algorithms Taxonomy

Evolutionary Algorithms

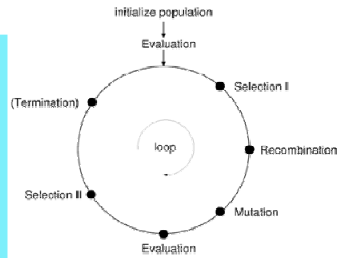
Evolution Strategies
- Mixed-integer capabilities
- Emphasis on mutation
- Self-adaptation
- Small population sizes
- Deterministic selection
- Developed in Germany
- Theory focused on convergence velocity

Genetic Algorithms
- Discrete representations
- Emphasis on crossover
- Constant parameters
- Larger population sizes
- Probabilistic selection
- Developed in USA
- Theory focused on schema processing

Other
- Evolutionary Prog
- Differential Evol.
- GP
- PSO
- EDA
- Real-coded Gas
- …

Generalized Evolutionary Algorithm

```
t := 0;
initialize(P(t));
evaluate(P(t));
while not terminate do
    P'(t) := mating_selection(P(t));
    P''(t) := variation(P'(t));
    evaluate(P''(t));
    P(t+1) := environmental_selection(P''(t) ∪ Q);
    t := t+1;
od
```



Evolution Strategy – Basics

- Mostly real-valued search space IR^n
  - also mixed-integer, discrete spaces
- Emphasis on mutation
  - n-dimensional normal distribution
  - expectation zero
- Different recombination operators
- Deterministic selection
  - (μ, λ)-selection: Deterioration possible
  - $(\mu+\lambda)$-selection: Only accepts improvements
- $\lambda >> \mu$, i.e.: Creation of offspring surplus
- Self-adaptation of strategy parameters.

Representation of search points

- Simple ES with 1/5 success rule:
  - Exogenous adaptation of step size σ
  - Mutation: $N(0, \sigma)$

$$\vec{a} = (x_1, \ldots, x_n)$$

- Self-adaptive ES with single step size:
  - One σ controls mutation for all x_i
  - Mutation: $N(0, \sigma)$

$$\vec{a} = ((x_1, \ldots, x_n), \sigma)$$

Representation of search points

- Self-adaptive ES with individual step sizes:
  - One individual σ_i per x_i
  - Mutation: $N_i(0, \sigma_i)$

$$\vec{a} = ((x_1, \ldots, x_n), (\sigma_1, \ldots, \sigma_n))$$

- Self-adaptive ES with correlated mutation:
  - Individual step sizes
  - One correlation angle per coordinate pair
  - Mutation according to covariance matrix: $N(\mathbf{0}, \mathbf{C})$

$$\vec{a} = ((x_1, \ldots, x_n), (\sigma_1, \ldots, \sigma_n), (\alpha_1, \ldots, \alpha_{n(n-1)/2}))$$

Evolution Strategy:

Algorithms
Mutation

Operators: Mutation – one σ

- Self-adaptive ES with one step size:
  - One σ controls mutation for all x_i
  - Mutation: N(0, σ)

$$\vec{a} = ((x_1,\ldots,x_n),\sigma)$$ (Individual before mutation)

$$\vec{a}' = ((x_1',\ldots,x_n'),\sigma')$$ (Individual after mutation)

$$\sigma' = \sigma \cdot \exp(\tau_0 \cdot N(0,1))$$ (1.: Mutation of step sizes)

$$x_i' = x_i + \sigma' \cdot N_i(0,1)$$ (2.: Mutation of objective variables)

Here the new σ' is used!

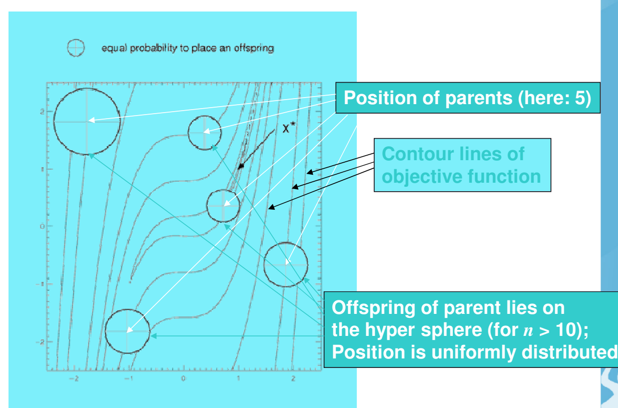
Operators: Mutation – one σ

- Thereby τ_0 is the so-called learning rate
  - Affects the speed of the σ-Adaptation
  - τ_0 bigger: faster but more imprecise
  - τ_0 smaller: slower but more precise
  - How to choose τ_0?
  - According to recommendation of Schwefel*:

$$\tau_0 = \frac{1}{\sqrt{n}}$$

*H.-P. Schwefel: Evolution and Optimum Seeking, Wiley, NY, 1995.

Operators: Mutation – one σ



Pros and Cons: One σ

- Advantages:
  - Simple adaptation mechanism
  - Self-adaptation usually fast and precise
- Disadvantages:
  - Bad adaptation in case of complicated contour lines
  - Bad adaptation in case of very differently scaled object variables
    - $-100 < x_i < 100$ and e.g. $-1 < x_j < 1$

Operators: Mutation – individual σ_i

- Self-adaptive ES with individual step sizes:
  - One σ_i per x_i
  - Mutation: $N_i(0, \sigma_i)$

$$\vec{a} = ((x_1,\ldots,x_n),(\sigma_1,\ldots,\sigma_n))$$
$$\vec{a}' = ((x_1',\ldots,x_n'),(\sigma_1',\ldots,\sigma_n'))$$
$$\sigma_i' = \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1))$$
$$x_i' = x_i + \sigma_i' \cdot N_i(0,1)$$

Individual before Mutation

Individual after Mutation

1.: Mutation of individual step sizes

2.: Mutation of object variables

The new individual σ_i' are used here!

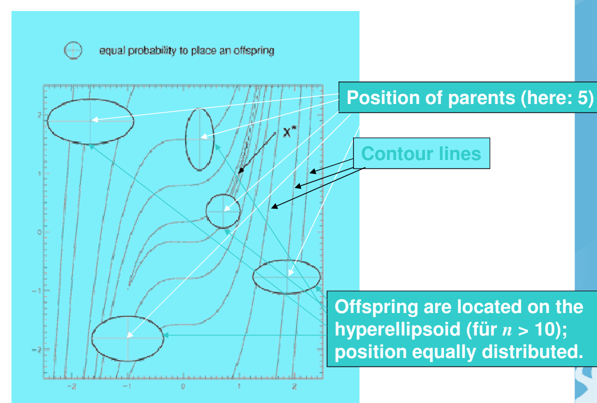
Operators: Mutation – individual σ_i

- τ, τ' are learning rates, again
  - τ': Global learning rate
  - N(0,1): Only one realisation
  - τ: local learning rate
  - $N_i(0,1)$: n realisations
  - Suggested by Schwefel*:

$$\tau' = \frac{1}{\sqrt{2n}} \qquad \tau = \frac{1}{\sqrt{2\sqrt{n}}}$$

*H.-P. Schwefel: Evolution and Optimum Seeking, Wiley, NY, 1995.

Operators: Mutation – individual σ_i



Pros and Cons: Individual σ_i

- Advantages:
  - Individual scaling of object variables
  - Increased global convergence reliability
- Disadvantages:
  - Slower convergence due to increased learning effort
  - No rotation of coordinate system possible
    - Required for badly conditioned objective function

Operators: Correlated Mutations

- Self-adaptive ES with correlated mutations:
  - Individual step sizes
  - One rotation angle for each pair of coordinates
  - Mutation according to covariance matrix: N(**0**, **C**)

$$\vec{a} = ((x_1,\ldots,x_n),(\sigma_1,\ldots,\sigma_n),(\alpha_1,\ldots,\alpha_{n(n-1)/2}))$$ ← Individual before

$$\vec{a}' = ((x_1',\ldots,x_n'),(\sigma_1',\ldots,\sigma_n'),(\alpha_1',\ldots,\alpha_{n(n-1)/2}'))$$ ← Individual after mutation

$$\sigma_i' = \sigma_i \cdot \exp(\tau' \cdot N(0,1) + \tau \cdot N_i(0,1))$$ ← 1.: Mutation of individual step sizes

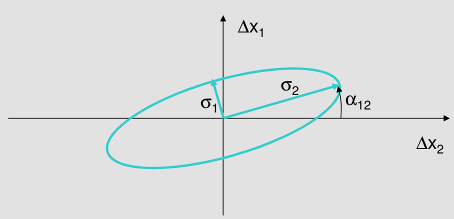
$$\alpha_j' = \alpha_j + \beta \cdot N_j(0,1)$$ ← 2.: Mutation of rotation angles

$$x_i' = x_i + \vec{N}(\vec{0}, C')$$ ← 3.: Mutation of object variables; New convariance matrix C‘ used here!

Operators: Correlated Mutations

- Interpretation of rotation angles α_{ij}
- Mapping onto convariances according to

$$c_{ij(i\neq j)} = \frac{1}{2}(\sigma_i^2 - \sigma_j^2)\tan(2\alpha_{ij})$$

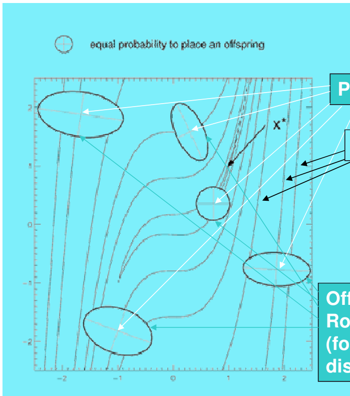


Operators: Correlated Mutation

- τ, τ‘, β are again learning rates
  - τ, τ‘ as before
  - β = 0,0873 (corresponds to 5 degree)
  - Out of boundary correction:

$$|\alpha_j'| > \pi \Rightarrow \alpha_j' \leftarrow \alpha_j' - 2\pi \cdot sign(\alpha_j')$$

Correlated Mutations for ES



Operators: Correlated Mutations

- How to create $\vec{N}(\vec{0}, C')$?
  - Multiplication of uncorrelated mutation vector with $n(n-1)/2$ rotational matrices

$$\vec{\sigma}_c = \prod_{i=1}^{n-1} \prod_{j=i+1}^{n} R(\alpha_{ij}) \cdot \vec{\sigma}_u$$

  - Generates only feasible (positiv definite) correlation matrices

Operators: Correlated Mutations

- Structur of rotation matrix

$$R(\alpha_{ij}) = \begin{pmatrix} 1 & & & & & & \\ & 1 & & & & & 0 \\ & & \cos(\alpha_{ij}) & & & -\sin(\alpha_{ij}) & \\ & & & 1 & & & \\ & & & & 1 & & \\ & & \sin(\alpha_{ij}) & & & \cos(\alpha_{ij}) & \\ & 0 & & & & & 1 \\ & & & & & & & 1 \end{pmatrix}$$

Operators: Correlated Mutations

- Implementation of correlated mutations

```
nq := n(n-1)/2;
for i:=1 to n do
   σu[i] := σ[i] * N_i(0,1);          } ← Generation of the uncorrelated mutation vector
for k:=1 to n-1 do
   n1 := n-k;
   n2 := n;
   for i:=1 to k do
        d1     := σu[n1]; d2:= σu[n2];
        σu[n2] := d1*sin(α[nq])+ d2*cos(α[nq]);
        σu[n1] := d1*cos(α[nq])- d2*sin(α[nq]);
        n2     := n2-1;
        nq     := nq-1;
   od
od
```

Rotations

Pros and Cons: Correlated Mutations

- Advantages:
  - Individual scaling of object variables
  - Rotation of coordinate system possible
  - Increased global convergence reliability
- Disadvantages:
  - Much slower convergence
  - Effort for mutations scales quadratically
  - Self-adaptation very inefficient

Operators: Mutation – Addendum

- Generating N(0,1)-distributed rnd numbers?

$$u = 2 \cdot U[0,1) - 1$$
$$v = 2 \cdot U[0,1) - 1$$
$$w = u^2 + v^2$$
$$\left.\begin{aligned} x_1 &= u \cdot \sqrt{\frac{-2\log(w)}{w}} \\ x_2 &= v \cdot \sqrt{\frac{-2\log(w)}{w}} \end{aligned}\right\} \text{If } w > 1$$

$$x_1, x_2 \sim N(0,1)$$

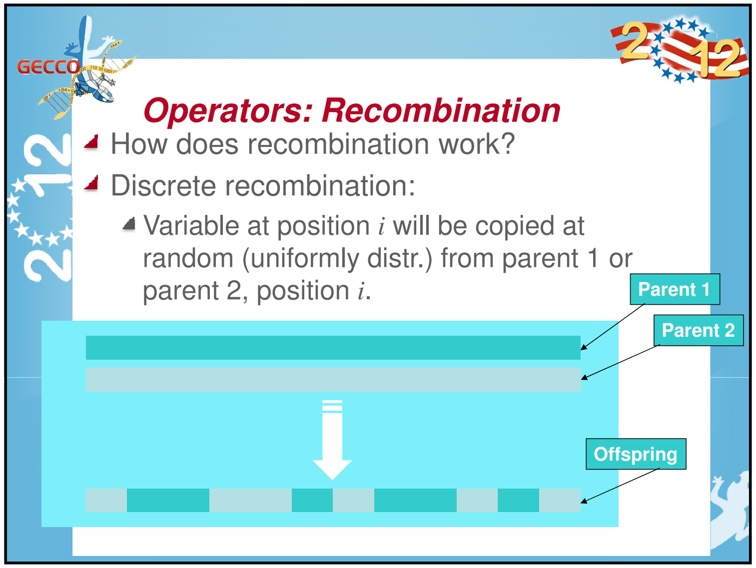
Evolution Strategy:

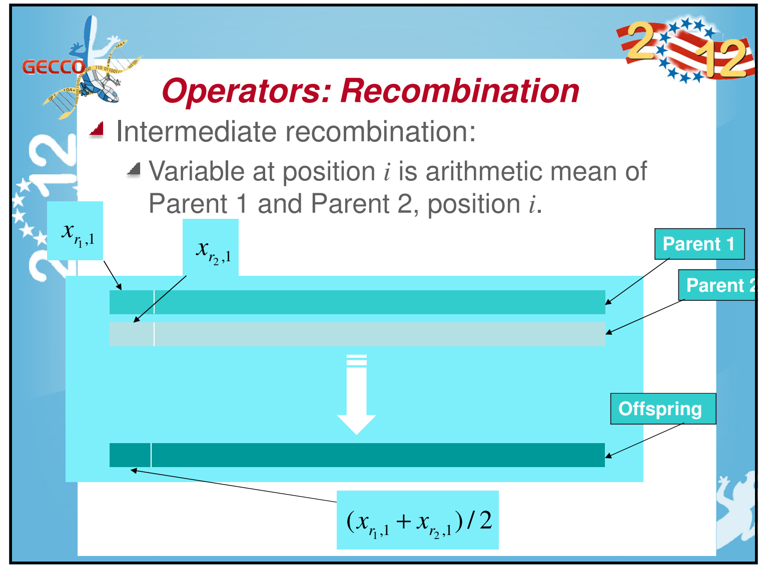
Algorithms Recombination

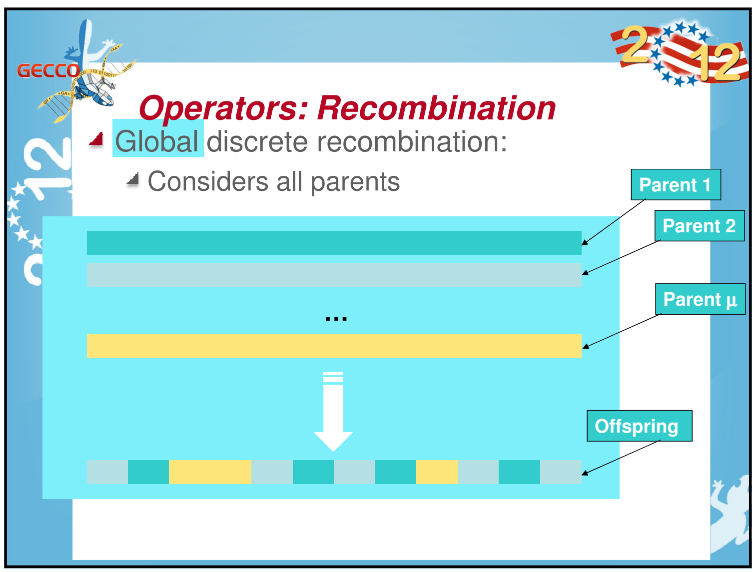
Operators: Recombination

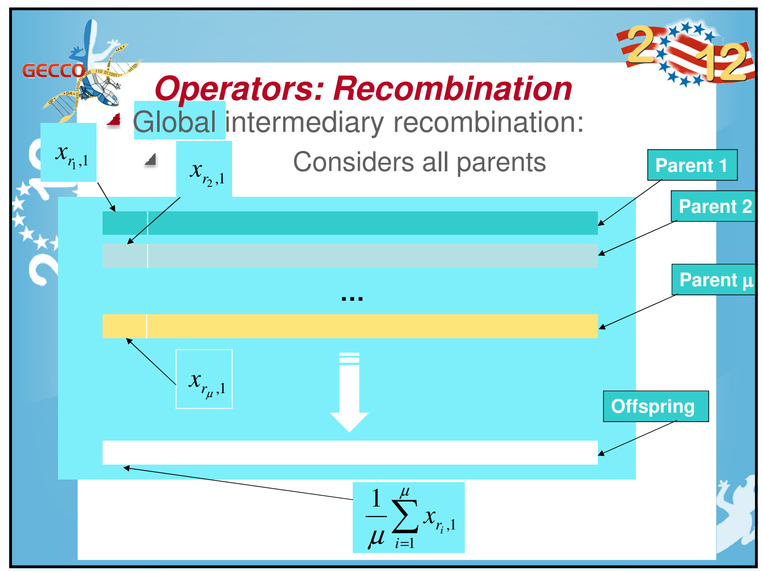
- Only for $\mu > 1$
- Directly after Selection
- Iteratively generates λ offspring:

```
for i:=1 to λ do
   choose recombinant r1 uniformly at random
       from parent_population;
   choose recombinant r2 <> r1 uniformly at random
       from parent population;
   offspring := recombine(r1,r2);
   add offspring to offspring_population;
od
```


GECCO
2012
Operators: Recombination
How does recombination work?
Discrete recombination:
Variable at position i will be copied at random (uniformly distr.) from parent 1 or parent 2, position i.
Parent 1
Parent 2
Offspring


GECCO
2012
Operators: Recombination
Intermediate recombination:
Variable at position i is arithmetic mean of Parent 1 and Parent 2, position i.
$x_{r_1,1}$
$x_{r_2,1}$
Parent 1
Parent 2
Offspring
$(x_{r_1,1} + x_{r_2,1})/2$


GECCO
2012
Operators: Recombination
Global discrete recombination:
Considers all parents
Parent 1
Parent 2
...
Parent µ
Offspring


GECCO
2012
Operators: Recombination
Global intermediary recombination:
Considers all parents
$x_{r_1,1}$
$x_{r_2,1}$
Parent 1
Parent 2
...
Parent µ
$x_{r_\mu,1}$
Offspring
$\frac{1}{\mu}\sum_{i=1}^{\mu} x_{r_i,1}$

Evolution Strategy

Algorithms
Selection

Operators: (μ+λ)-Selection

= Actual solution candidate
= New solution candidate

- (μ+λ)-Selection means:
  - μ parents produce λ offspring by
    - (Recombination +)
    - Mutation
  - These μ+λ individuals will be considered together
  - The μ best out of μ+λ will be selected („survive“)
    - Deterministic selection
  - This method guarantees monotony
    - Deteriorations will never be accepted

Recombination may be left out
Mutation always exists!

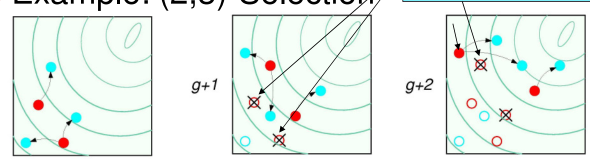
Operators: (μ,λ)-Selection

- (μ,λ)-Selection means:
  - μ parents produce λ >> μ offspring by
    - (Recombination +)
    - Mutation
  - λ offspring will be considered alone
  - The μ best out of λ offspring will be selected
    - Deterministic selection
  - The method doesn‘t guarantee monotony
    - Deteriorations are possible
    - The best objective function value in generation $t+1$ may be worse than the best in generation t.

Operators: Selection

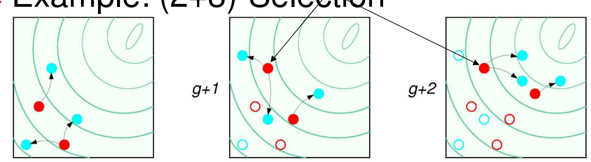
Parents don‘t survive …
Parents don‘t survive!
… but a worse offspring.

- Example: (2,3)-Selection

g g+1 g+2


… now this offspring survives.

- Example: (2+3)-Selection

g g+1 g+2


Operators: Selection

Exception!

- Possible occurrences of selection
  - (1+1)-ES: One parent, one offspring, 1/5-Rule
  - (1,λ)-ES: One Parent, λ offspring
    - Example: (1,10)-Strategy
    - One step size / n self-adaptive step sizes
    - Mutative step size control
    - Derandomized strategy
  - (μ,λ)-ES: μ > 1 parents, λ > μ offspring
    - Example: (2,15)-Strategy
    - Includes recombination
    - Can overcome local optima
  - (μ+λ)-strategies: elitist strategies

Evolution Strategy:

Self adaptation of step sizes

Self-adaptation

- No deterministic step size control!
- Rather: Evolution of step sizes
  - Biology: Repair enzymes, mutator-genes
- Why should this work at all?
  - Indirect coupling: step sizes – progress
  - Good step sizes improve individuals
  - Bad ones make them worse
  - This yields an indirect step size selection

Self-adaptation: Example

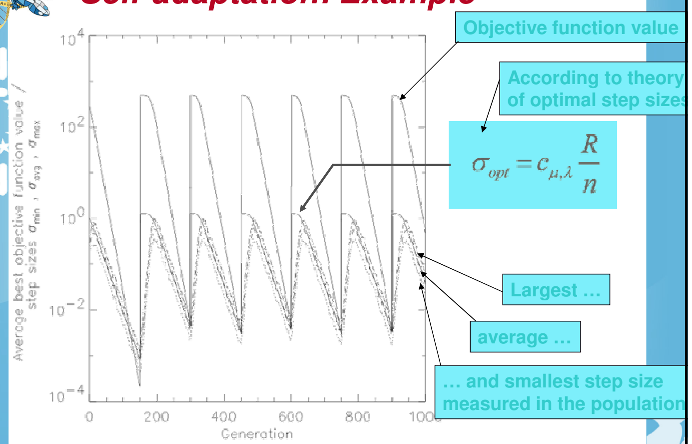
- How can we test this at all?
- Need to know optimal step size …
  - Only for very simple, convex objective functions
  - Here: Sphere model

$\vec{x}^*$: Optimum

$$f(\vec{x}) = \sum_{i=1}^{n} (x_i - x_i^*)^2$$

  - Dynamic sphere model
    - Optimum locations changes occasionally

Self-adaptation: Example



Self-adaptation

- Self-adaptation of one step size
  - Perfect adaptation
  - Learning time for back adaptation proportional n
  - Proofs only for convex functions
- Individual step sizes
  - Experiments by Schwefel
- Correlated mutations
  - Adaptation much slower

Evolution Strategy:

Derandomization

Derandomization

- Goals:
  - Fast convergence speed
  - Fast step size adaptation
  - Precise step size adaptation
  - Compromise convergence velocity – convergence reliability
- Idea: Realizations of $N(0, \sigma)$ are important!
  - Step sizes and realizations can be much different from each other
  - Accumulates information over time

Derandomzed (1,λ)-ES

- Current parent: $\vec{x}^g$ in generation g
- Mutation ($k=1,\ldots,\lambda$):

$$\vec{x}^g_{N_k} = \vec{x}^g + \delta^g \cdot \vec{\delta}^g_{scal} \cdot \vec{Z}_k$$

Offspring k

Global step size in generation g

Individual step sizes in generation g

$$\vec{Z} = (z_1,\ldots,z_n) \qquad z_i \sim N(0,1)$$

- Selection: Choice of best offspring

$$\vec{x}^{g+1} = \vec{x}^g_{N_{sel}}$$

Best of λ offspring in generation g

Derandomized (1,λ)-ES

- Accumulation of selected mutations:

$$\vec{Z}^g_A = (1-c)\cdot\vec{Z}^{g-1}_A + c\cdot\vec{Z}_{sel}$$

The particular mutation vector, which created the parent!

- Also: weighted history of good mutation vectors!
- Initialization:

$$\vec{Z}^0_A = \vec{0}$$

- Weight factor: $c = \dfrac{1}{\sqrt{n}}$

Derandomized (1,λ)-ES

- Step size adaptation:

$$\delta^{g+1} = \delta^g \cdot \left(\exp\left(\frac{\left|\vec{Z}^g_A\right|}{\sqrt{n}\cdot\sqrt{\frac{c}{2-c}}}\right) - 1 + \frac{1}{5n}\right)^{\beta}$$

$$\vec{\delta}^{g+1}_{scal} = \vec{\delta}^g_{scal} \cdot \left(\frac{\left|\vec{Z}^g_A\right|}{\sqrt{\frac{c}{2-c}}} + 0.35\right)^{\beta_{scal}}$$

Norm of vector

Vektor of absolute values

Regulates adaptation speed and precision

Derandomized (1,λ)-ES

- Explanations:
  - Normalization of average variations in case of missing selection (no bias):

$$\sqrt{\frac{c}{2-c}}$$

  - Correction for small n: $1/(5n)$
  - Learning rates: $\beta = \sqrt{1/n}$, $\beta_{scal} = 1/n$

Evolution Strategy:

Rules of thumb

Some Theory Highlights

- Convergence velocity:

$$\varphi \sim 1/n$$

Problem dimensionality

- For (1,λ)-strategies:

$$\varphi \sim \ln \lambda$$

Speedup by λ is just logarithmic – more processors are only to a limited extend useful to increase φ.

- For (μ,λ)-strategis (discrete and intermediary recombination)

$$\varphi \sim \mu \ln \frac{\lambda}{\mu}$$

Genetic Repair Effect of recombination!

...

- For strategies with global intermediary recombination:

$$\lambda = 4 + \lfloor 3 \log n \rfloor$$
$$\mu = \lfloor \lambda / 2 \rfloor$$

n	λ	μ
10	10.91	5.45
20	12.99	6.49
30	14.20	7.10
40	15.07	7.53
50	15.74	7.87
60	16.28	8.14
70	16.75	8.37
80	17.15	8.57
90	17.50	8.75
100	17.82	8.91
110	18.10	9.05
120	18.36	9.18
130	18.60	9.30
140	18.82	9.41
150	19.03	9.52

- Good heuristic for (1,λ):

$$\lambda = 10$$

- General:

$$\lambda \approx 7\mu$$

Mixed-Integer Evolution Strategies

Mixed-Integer Evolution Strategy

❖ Generalized optimization problem:

$$f(r_1,\ldots,r_{n_r},z_1,\ldots,z_{n_z},d_1,\ldots,d_{n_d}) \rightarrow min$$

subject to:

$$r_i \in [r_i^{min}, r_i^{max}] \subset \mathbb{R},\ \ i=1,\ldots,n_r$$

$$z_i \in [z_i^{min}, z_i^{max}] \subset \mathbb{Z},\ \ i=1,\ldots,n_z$$

$$d_i \in D_i = \{d_{i,1},\ldots,d_{i,|D_i|}\}, i=1,\ldots,n_d$$

Mixed-Integer ES: Mutation

for $i = 1,\ldots,n_r$ **do**
  $s_i' \leftarrow s_i \exp(\tau_g N_g + \tau_l N(0,1))$
  $r_i' = r_i + N(0, s_i')$
end for
for $i = 1,\ldots,n_z$ **do**
  $q_i' \leftarrow q_i \exp(\tau_g N_g + \tau_l N(0,1))$
  $z_i' \leftarrow z_i + G(0, q_i')$
end for
$p_i' := 1/[1 + \frac{1-p_i}{p_i} * \exp(-\tau_l * N(0,1))]$
for $i \in \{1,\ldots,n_d\}$ **do**
  if $U(0,1) < p_i'$ **then**
    $d_i' \leftarrow$ uniformly randomly value from D_i
  end if
end for

Learning rates (global)

Learning rates (global)

Geometrical distribution

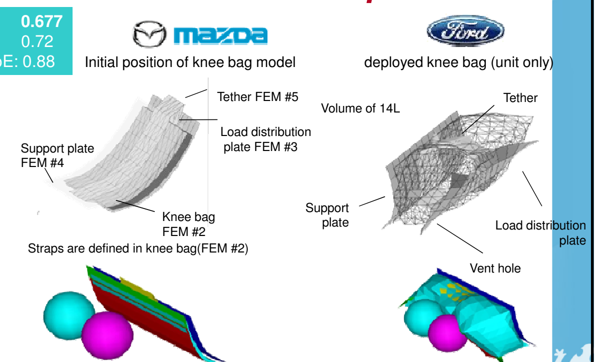
Mutation Probabilities

Some Application Examples

Mostly Engineering Problems

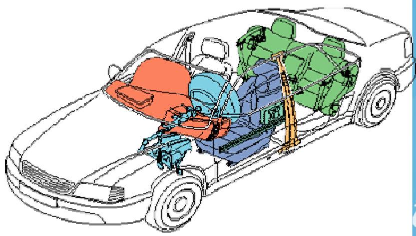
Examples I: Inflatable Knee Bolster Optimization

Low Cost ES: **0.677**
GA (Ford): 0.72
Hooke Jeeves DoE: 0.88



IKB: Previous Designs

# Variables	Characteristics	HIC	CG	Left foot load	Right foot load	$P_{Combined}$
4	Unconstrained	576,324	44,880	4935	3504	12,393
5	Unconstrained	384,389	41,460	4707	4704	8,758
9	Unconstrained	292,354	38,298	5573	5498	6,951
10	Constrained	305,900	39,042	6815	6850	7,289

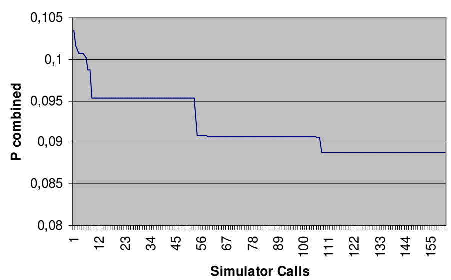


IKB: Problem Statement

- Objective: Min Ptotal   Subject to: Left   Femur load <= 7000
  Right   Femur load <= 7000

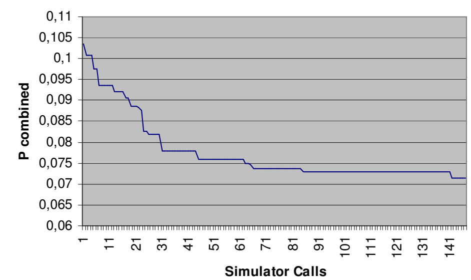
Design Variable	Description	Base Design 1	Base Design 2	GA (Yan Fu)
dx	IKB center offset x	0	0	0,01
dz	IKB center offset y	0	0	-0,01
rcdex	KB venting area ratio	1	1	2
massrat	KB mass inflow ratio	1	1	1,5
rcdexd	DB venting area ratio	1	1	2,5
Dmassratf	DB high output mass inflow ratio	1	1	1,1
Dmassratl	DB low output mass inflow ratio	1	1	1
dbfire	DB firing time	0	0	-0,003
dstraprat	DB strap length ratio	1	1	1,5
emr	Load of load limiter (N)	3000	3000	2000
Performance Response	**Description**			
NCAP_HIC_50	HIC	590	555.711	305,9
NCAP_CG_50	CG	47	47.133	39,04
NCAP_FMLL_50	Left foot load	760	6079	6815
NCAP_FMRL_50	Right foot load	900	5766	6850
P combined (Quality)		13.693	13.276	7,289

IKB Results I: Hooke-Jeeves



Quality: 8.888   Simulations: 160

IKB Results II: (1+1)-ES



Quality: 7.142   Simulations: 122

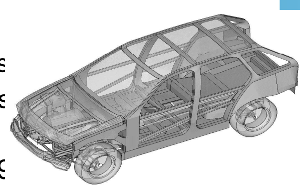
Engineering Optimization

Safety Optimization – Pilot Study

- Aim: Identification of most appropriate Optimization Algorithm for realistic example!
- Optimizations for 3 test cases and 14 algorithms were performed (28 x 10 = 280 shots)
  - Body MDO Crash / Statics / Dynamics
  - MCO B-Pillar
  - MCO Shape of Engine Mount
- NuTech's ES performed significantly better than Monte-Carlo-scheme, GA, and Simulated Annealing
- Results confirmed by statistical hypothesis testing

MDO Crash / Statics / Dynamics

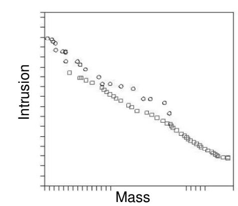
- Minimization of body mass
- Finite element mesh
  - Crash ~ 130.000 elements
  - NVH ~ 90.000 elements
- Independent parameters: Thickness of each unit: 109
- Constraints: 18



Algorithm	Avg. reduction (kg)	Max. reduction (kg)	Min. reduction (kg)
Best so far	-6.6	-8.3	-3.3
Our ES	-9.0	-13.4	-6.3

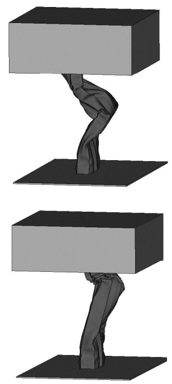
MCO B-Pillar – Side Crash

- Minimization of mass & displacement
- Finite element mesh
  - ~ 300.000 elements
- Independent parameters: Thickness of 10 units
- Constraints: 0
- ES successfully developed Pareto front



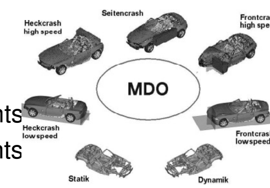
MCO Shape of Engine Mount

- Mass minimal shape with axial load > 90 kN
- Finite element mesh
  - ~ 5000 elements
- Independent parameters: 9 geometry variables
- Dependent parameters: 7
- Constraints: 3
- ES optimized mount
  - less weight than mount optimized with best so far method
  - geometrically better deformation

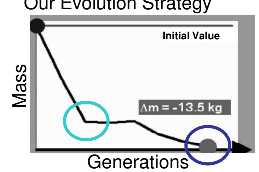


Safety Optimization – Example

- Production Run !
- Minimization of body mass
- Finite element mesh
  - Crash ~ 1.000.000 elements
  - NVH ~ 300.000 elements
- Independent parameters:
  - Thickness of each unit: 136
- Constraints: 47, resulting from various loading cases
- 180 (10 x 18) shots ~ 12 days
- No statistical evaluation due to problem complexity

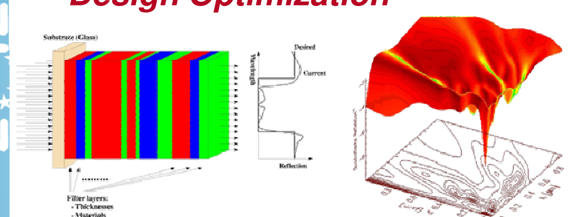


Safety Optimization – Example of use



- 13,5 kg weight reduction by NuTech's ES
- Beats best so far method significantly
- Typically faster convergence velocity of ES
  ~ 45% less time (~ 3 days saving) for comparable quality needed
- Still potential of improvements after 180 shots.
- Reduction of development time from 5 to 2 weeks allows for process integration

Optical Coatings: Design Optimization



- Nonlinear mixed-integer problem, variable dimensionality.
- Minimize deviation from desired reflection behaviour.
- Excellent synthesis method; robust and reliable results.

Dielectric Filter Design Problem

CORNING

Client:

Corning, Inc.,
Corning, NY

- Dielectric filter design.
- n=40 layers assumed.
- Layer thicknesses xi in [0.01, 10.0].
- Quality function: Sum of quadratic penalty terms.

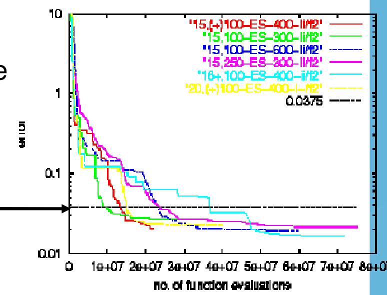
$$quality = \sum_{i=1}^{15} weight \cdot \left(\frac{calculated - desired}{scale}\right)^2 \rightarrow \min$$

- Penalty terms = 0 iff constraints satisfied.

Results: Overview of Runs

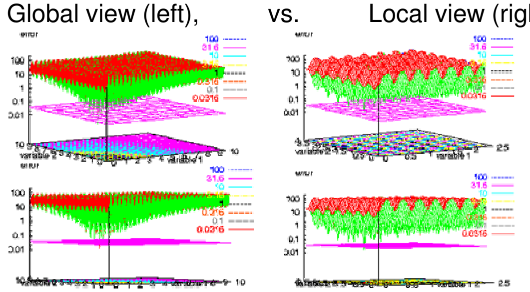
- Factor 2 in quality.
- Factor 10 in effort.
- Reliable, repeatable results.

Benchmark

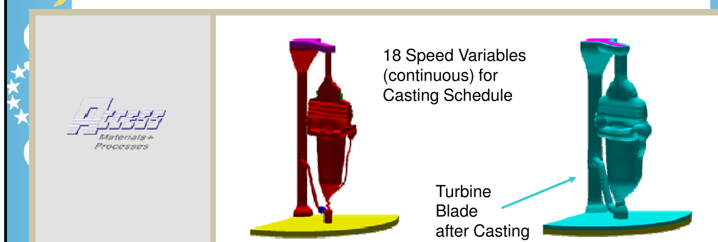


Problem Topology Analysis: An Attempt

- Grid evaluation for 2 variables.
- Close to the optimum (from vector of quality 0.0199).
- Global view (left), vs. Local view (right).



Bridgman Casting Process

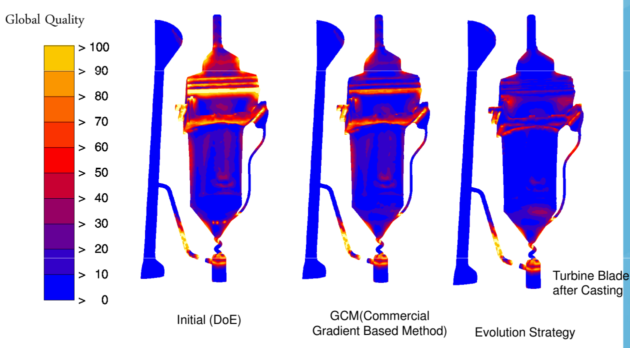


- FE mesh of 1/3 geometry: 98.610 nodes, 357.300 tetrahedrons, 92.830 radiation surfaces

⇒ large problem:

- run time varies: 16 h 30 min to 32 h (SGI, Origin, R12000, 400 MHz)
- at each run: 38,3 GB of view factors (49 positions) are treated!

Bridgman Casting Process



Quality Comparison of the Initial and Optimized Configurations

Traffic Light Control

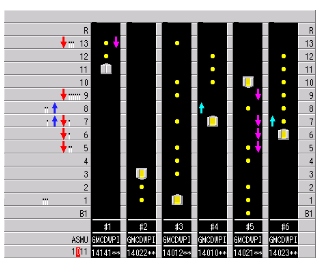
Ministerie van Verkeer en Waterstaat

- Generates green times for next switching schedule.
- Minimization of total delay / number of stops.
- Better results (3 – 5%) / higher flexibility than with traditional controllers.
- Dynamic optimization, depending on actual traffic (measured by control loops).

- **Client:**
  Dutch Ministry of Traffic
  Rotterdam, NL

Elevator Control

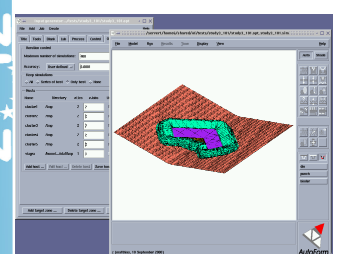
FUJITEC



- Minimization of passenger waiting times.
- Better results (3 – 5%) / higher flexibility than with traditional controllers.
- Dynamic optimization, depending on actual traffic.

- **Client:**
  Fujitec Co. Ltd., Osaka, Japan

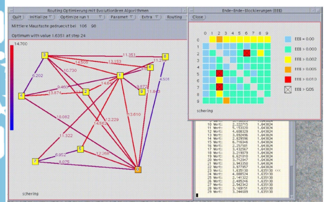
Metal Stamping Process



- Minimization of defects in the produced parts.
- Optimization on geometric parameters and forces.
- Fast algorithm; finds very good results.

- **Client:**
  AutoForm Engineering GmbH, Dortmund

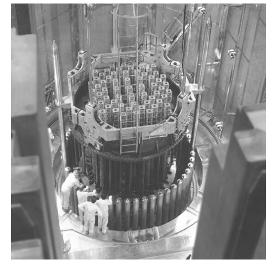
Network Routing



- Minimization of end-to-end-blockings under service constraints.
- Optimization of routing tables for existing, hard-wired networks.
- 10%-1000% improvement.

- **Client:**
  SIEMENS AG, München

Nuclear Reactor Refueling



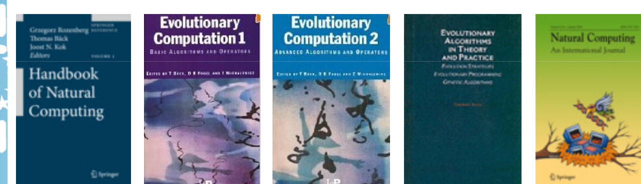
- Minimization of total costs.
- Creates new fuel assembly reload patterns.
- Clear improvements (1%-5%) of existing expert solutions.
- Huge cost saving.

- **Client:**
  SIEMENS AG, München

Business Issues

- ❖Supply Chain Optimization
- ❖Scheduling & Timetabling
- ❖Product Development, R&D
- ❖Management Decision Making, e.g., project portfolio optimization
- ❖Optimization of Marketing Strategies; Channel allocation
- ❖Multicriteria Optimization (cost / quality)
- ❖... And many others

Exciting Literature ...



Leiden Institute of Advanced Computer Science (LIACS)

- See www.liacs.nl and http://natcomp.liacs.nl
- Masters in
  - Comp. Science
  - ICT in Business
  - Media Technology
- Elected „Best Comp. Sci. Study“ by students.
- Excellent job opportunities for our students.
- Research education with an eye on business.

LIACS Research

Algorithms
Prof. J.N. Kok, Prof. T. Bäck
- Novel Algorithms
- Data Mining
- Natural Computing
- Applications
  - Drug Design
  - Medicine
  - Engineering
  - Logistics
  - Physics

Technology and Innovation Management
Prof. B. Katzy
- Coevolution of Technology and Social Structures
- Entrepreneurship
- Innovation Management

Core Computer Technologies
Prof. H. Wijshoff, Prof. E. Deprettere
- Embedded Systems
- Parallel / Distributed Computing
- Compiler Technology
- Data Mining

Synergies & Collaboration

Imagery and Media
Dr. M. Lew, Dr. F. Verbeek
- Computer Vision and Audio/Video
- Bioimaging
- Multimedia Search
- Internet Technology
- Computer Graphics

Foundations of Software Technology
Prof. F. Arbab, Prof. J.N. Kok
- Software Systems
- Embedded Systems
- Service Composition
- Multicore Systems
- Formal Methods
- Coordination / Concurrency

Leiden Institute of Advanced Computer Science

Literature

- H.-P. Schwefel: *Evolution and Optimum Seeking*, Wiley, NY, 1995.
- I. Rechenberg: *Evolutionsstrategie 94*, frommann-holzboog, Stuttgart, 1994.
- Th. Bäck: *Evolutionary Algorithms in Theory and Practice*, Oxford University Press, NY, 1996.
- Th. Bäck, D.B. Fogel, Z. Michalewicz (Hrsg.): *Handbook of Evolutionary Computation, Vols. 1,2*, Institute of Physics Publishing, 2000.