

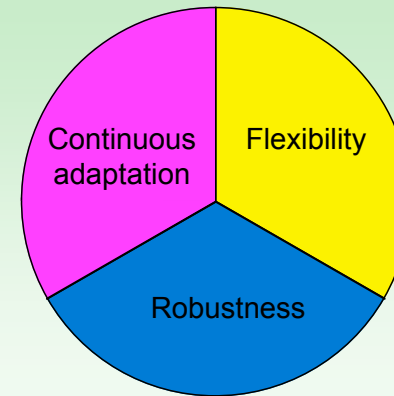
# Optimization in Dynamic Environments

GECCO Tutorial 2005

Jürgen Branke  
Institute AIFB, University of Karlsruhe  
Germany  
branke@aifb.uni-karlsruhe.de

3

## Three aspects in dynamic environments



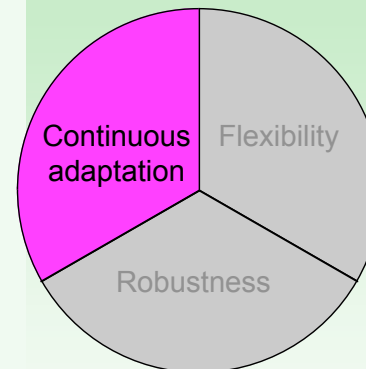
2

## Motivation

- Many real-world applications are dynamic
  - Scheduling
  - Control problems
  - Vehicle routing
  - Portfolio optimization
  - etc.
  - Also: Co-evolutionary approaches, model-refinement
- Current approaches
  - Ignore dynamics and re-optimize regularly
  - Use very simple control rules
- Large potential when dynamism is addressed explicitly
- Nature-inspired optimization algorithms seem particularly promising, as nature is a continuously changing environment

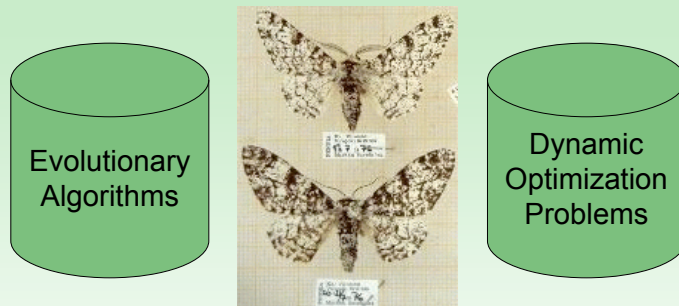
4

## Part I - Continuous Adaptation



- The problem of convergence
- Remedies
- Benchmarks  
(in particular: Moving Peaks)
- Additional aspects
  - Learning
  - Theory
- Other metaheuristics
  - Ant Colony Optimization
  - Particle Swarm Optimization

## Nature is able to adapt



## Possible Remedies

1. Restart after a change  
(only choice if changes are too severe)

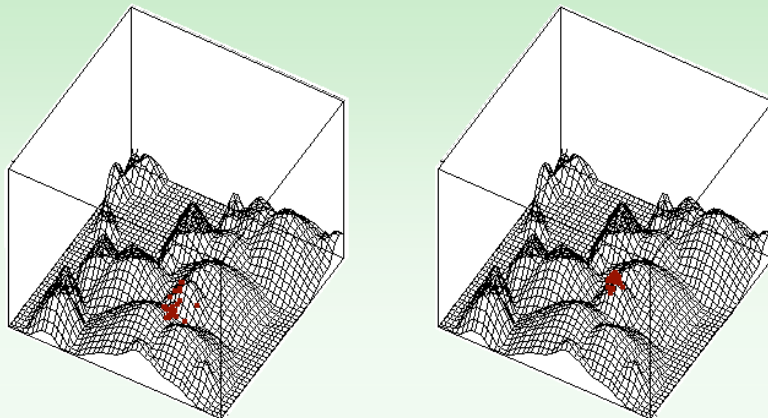
**But: Too slow**

2. Generate diversity after a change
  - Hypermutation [Cobb 1990]
  - Variable Local Search [Vavak et al. 1997]

**But: Randomization destroys information, only local search or similar to restart**

## The problem of convergence

For static optimization problems, convergence is desired.  
If the problem is dynamic, convergence is dangerous.



## Possible Remedies (2)

3. Maintain diversity throughout the run
  - Random Immigrants [Grefenstette 1992]
  - Sharing/Crowding [Andersen 1991, Cedeno & Vemuri 1997]
  - Thermodynamical GA [Mori et al. 1996]
  - Sentinels [Morrison 2004]

**But: Disturbs optimization process**

4. Memory-enhanced EAs

- **Implicit** memory [Goldberg & Smith 1987, Ng & Wong 1995, Lewis et al. 1998]
  - Redundant genetic representation (e.g. diploid)
  - EA is free to use additional memory
- **Explicit** memory [Ramsey & Grefenstette 1993, Trojanowski et al. 1997, Mori et al. 1997, Branke 1999]
  - Explicit rules which information to store in and retrieve from the memory

**But: Only useful when optimum reappears at old location, Problem of convergence remains**

## Possible Remedies (3)

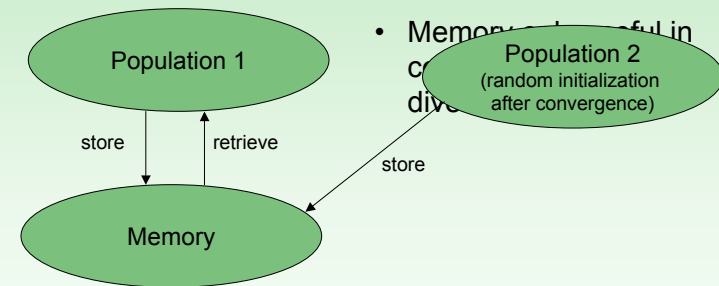
### 5. Multi-Population approaches

- Maintain different subpopulations on different peaks
  - adaptive memory
  - able to detect new optima
  - distance/similarity metric required
- Self-Organizing Scouts [Branke et al. 2000, Branke 2001]
- Multi-National EA [Ursem 2000]

Maintains useful diversity

## Memory/Search-Approach [Branke 1999]

- Explicit memorization of individuals
- Keep the better of the two most similar



➡ Sensible balance of exploration vs. exploitation

## Thermodynamical GA [Mori et al. 1996]

- Select next parent generation such that they are a good compromise between quality and diversity
- Select parents one by one such that the resulting (incomplete) parent generation minimizes

$$\min F = \langle E \rangle - TH$$

free energy (pointing to  $\langle E \rangle$ )  
 average population fitness (pointing to  $\langle E \rangle$ )  
 diversity (pointing to  $-TH$ )

- Requires to tune parameter T
- Computationally expensive

## Self Organizing Scouts (SOS) [Branke 2001]

- Idea: Collect information about search space
- Whenever a local optimum has been found
  - ➡ watch it with some scouts
- Base population should search for new peak
- Scouts should be able to track “their” peak

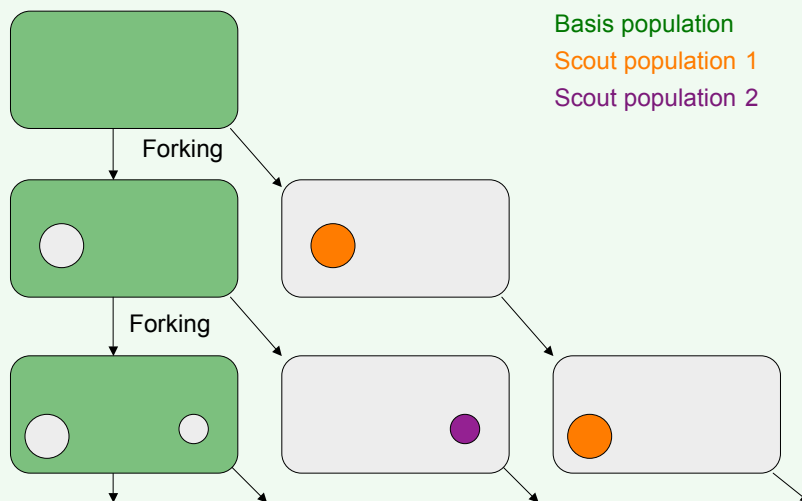
## How does it work, really?

- When a cluster is detected in basis population  
     ➡ **Forking**

## How does it work, really?

- When a cluster is detected in basis population  
     ➡ **Forking**
- Invalid individuals are replaced by random individuals  
     ➡ **Diversification**
- Best individual defines center ➡ **Tracking**
- Number of individuals in scout population depends on quality and trend ➡ **Efficiency**
- Size of the scout population's search space
  - Shrinks continuously
  - Is increased when two scout populations merge
 ➡ **Adaptation**

## Forking [Tsutsui et al. 1997]



## Typical benchmark problems

- Moving Peaks Benchmark [Branke 1999, Morrison & DeJong 1999]
- Dynamic knapsack problem, e.g. [Mori et al. 1996]
- Dynamic bit-matching, e.g. [Stanhope & Daida 1999, Droste 2003]
- Binary problem and changing XOR [Yang & Yao, to appear]
- Scheduling with new jobs arriving over time, e.g. [Mattfeld & Bierwirth 2004]
- Greenhouse control problem [Ursem et al. 2002]

Problem characteristics: [Branke 2001]

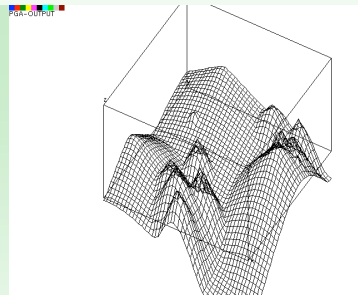
- Change severity
- Change frequency
- Predictability
- Cycle length / cycle accuracy

Measuring change characteristics [Branke et al. 2005]

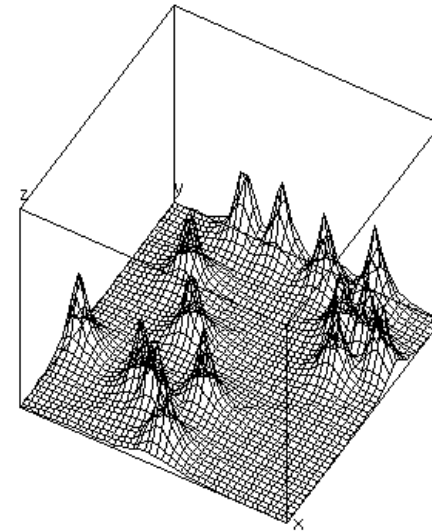
## Moving peaks benchmark [Branke 1999]

available at <http://www.aifb.uni-karlsruhe.de/~jbr/MovPeaks>

- Multi-modal environment characterised by moving peaks of varying widths and heights
- Small continuous changes in  $f$  can lead to discontinuous changes in  $x_{opt}$
- Parameters:
  - Change frequency
  - Number of peaks
  - Severity (length of shift vector, height and width)
  - Correlation of shifts
  - Number of dimensions
  - Shape of the peaks



## Demo: Self-Organizing Scouts



## Performance Measure: Offline Error

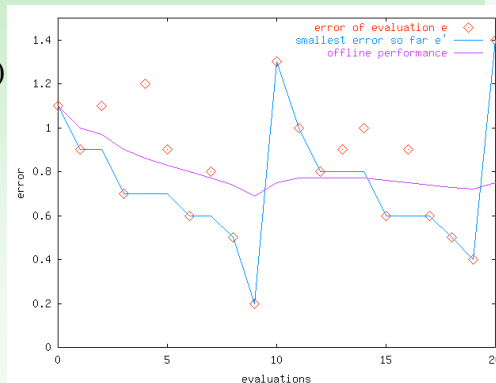
Difficulty: best solution found is not sufficient

→ Use modified offline error  $\varepsilon^*(T)$

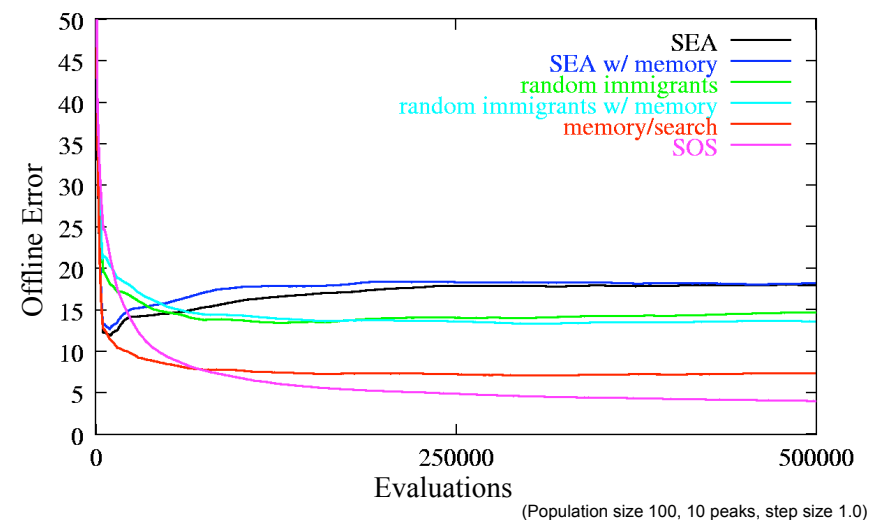
$$\varepsilon^*(T) = \frac{1}{T} \sum_{t=1}^T (opt_t - e'_t)$$

$$e'_t = \max(e_t, e_{\tau+1}, \dots, e_t)$$

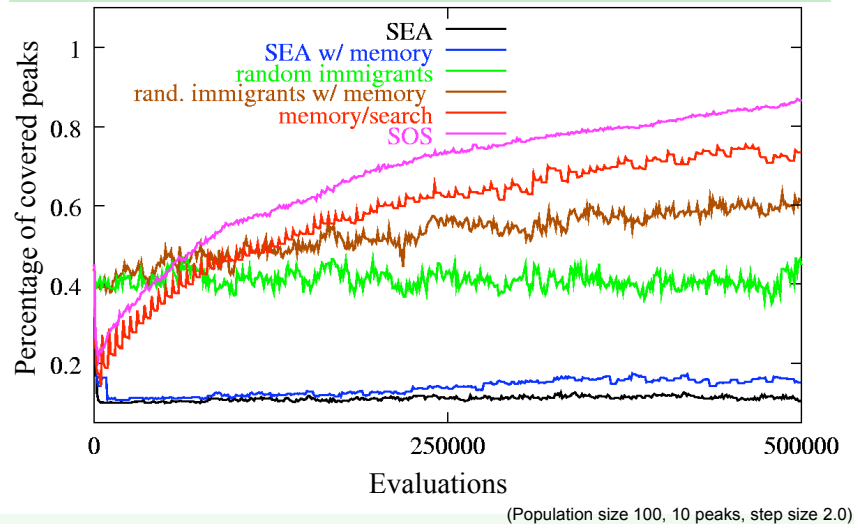
$\tau$  : time of last change



## Comparison of offline error



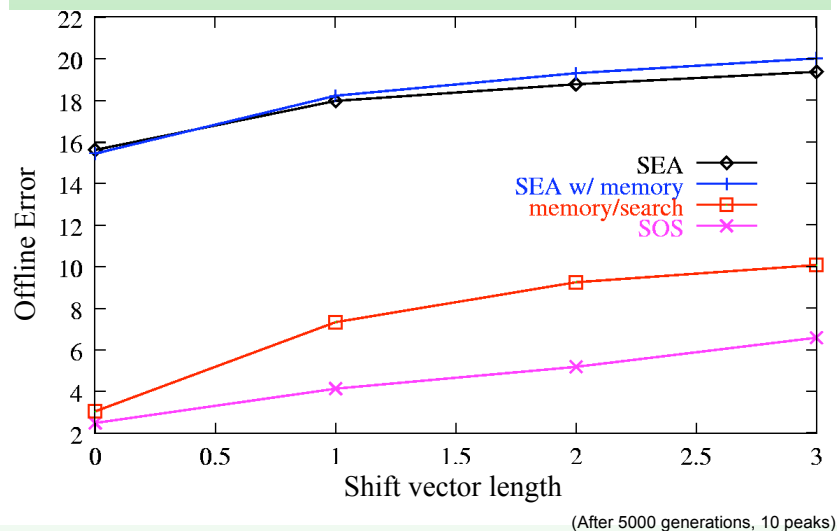
## Percentage of covered peaks



## Summary of Observations

- Standard EA gets stuck on single peak
- Diversity preservation slows down convergence
- Random immigrants introduce high diversity from the beginning, but benefit is limited
- Memory without diversity preservation is counterproductive
- Non-adaptive memory suffers significantly if peaks move
- Self-organizing scouts performs best

## Influence of step size



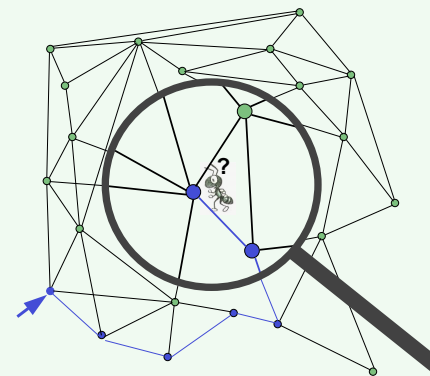
## Additional Aspects: Learning of change characteristics

- If the characteristics of a change can be learned, search can be biased accordingly. Examples
  - learn severity
  - learn direction
- Is standard ES self-adaptation sufficient?
  - Perhaps not, because Gaussian mutation is not appropriate [Weicker 2003]

## Additional Aspects: Learning vs. Evolution

- In nature, long-term adaptation is accomplished by evolution, while short-term adaptation is achieved by learning
- Lamarckian evolution performs better in static environments, Darwinian evolution is better in dynamic environments [Sasaki & Tokoro 1999]
- Hill-climbing (learning) may be better to follow a slowly moving peak.

## Other nature-inspired search heuristics: 1. Ant colony optimization (ACO)



Ants make decisions probabilistically, based on:

- Memory (e.g. no city may be visited twice)
- Heuristic (e.g. prefer nearby cities)
- Pheromones

In every iteration:

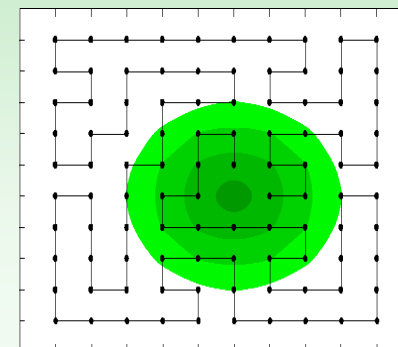
- Let  $m$  ants each construct a solution
- Ants that constructed good solutions may lay pheromone on their „decision path“
- Pheromones evaporate slowly

## Additional Aspects: Theoretical Results

- Infinite population GA for moving NIH problem [Ronnevinkel et al. 2001]
  - Identification of different phases depending on change frequency and mutation rate
  - Steady state and generational
- (1+1) ES for the dynamic bit-matching problem [Droste 2003]
  - Maximal change severity such that runtime (first passage time) is still polynomial
- $(\mu/\mu, \lambda)$  ES for continuously moving sphere [Arnold/Beyer 2002]
  - Distance to optimum and mutation step size in equilibrium
  - No loss of diversity?

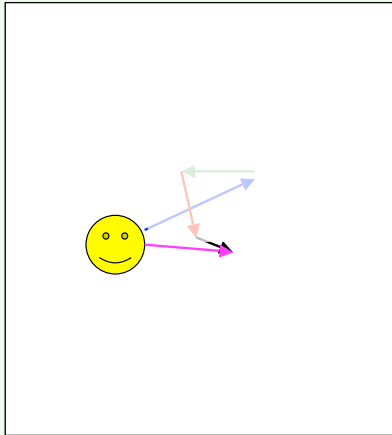
## ACO for Dynamic Problems [Guntsch et al. 2001]

- Introduce local variance where needed
- Heuristic repair of solutions



## Other nature-inspired search heuristics:

### 2. Particle Swarm Optimization



#### Swarm of particles

- Each particle
  - has velocity
  - keeps track of best visited solution
  - knows about best solution found so far
- In every iteration, each particle
  - adapts velocity, taking into account local and global best
  - moves according to new velocity
  - evaluates solution
  - updates local and global best

### Charged PSO

- Some of the particles are “charged”, i.e. repel each other
- Charged particles orbit nucleus similar to atom
- Neutral particles *exploit* and the charged particles *explore*
- Diversity is maintained and tracking is possible

**But:** difficult to control,  $N^2$  complexity

### Quantum PSO

- Quantum particles are randomized within a ball of radius  $r_{cloud}$  centered on the  $p_g$

### PSO for dynamic problems

- Re-initialization of local memory / replace with current location [Carlisle & Dozier 2000] Memory update
- Re-initialize part of the swarm population [Hu & Eberhart 2002] Discover new peaks
- Charged particles [Blackwell & Bentley 2002] Local diversity, tracking
- Hierarchical Swarms [Janson & Middendorf 2004] Local diversity, tracking
- Swarm species [Parrot & Li 2004] Remember several peaks
- Multi Swarms [Blackwell & Branke 2004] 
 Combination of ideas from
 
  - Charged PSO
  - Self-Organizing Scouts

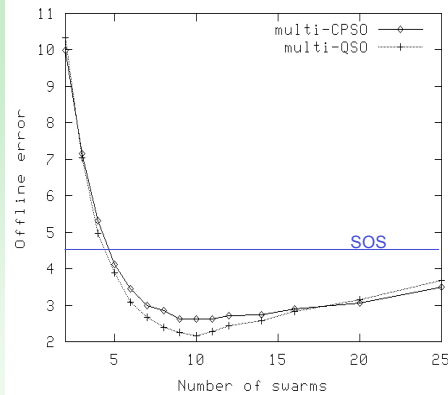
### Multi-Swarms

- More than one swarm
- Charged or quantum particles to allow tracking of peak
- Exclusion:
  - If global best of two swarms become too similar: competition
  - Swarm with lower fitness is randomized
  - Winner may continue optimization
- Anti-convergence:
  - When all swarms have converged, re-initialize worst swarm



## Results

- Single swarms (PSO, CPSO and QSO) are similar, and close to single population EA result
- QSO generally better than CPSO
- Best result for  $M = \text{number of peaks}$
- Better than SOS for  $5 \leq M \leq 25$



## General idea

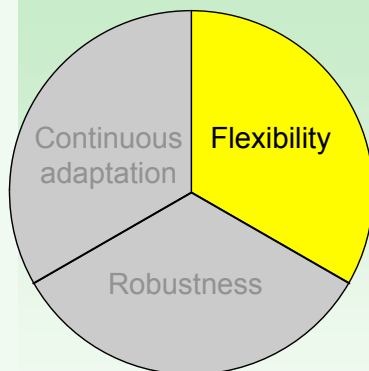
- Be prepared!
- Be flexible!

„If a problem requires *sequential decision making* under an *uncertain future*, and if the *decisions impact the future state* of the system, decision making should *anticipate future needs*. This means that an optimization algorithm should not just focus on the primary objective function, but should additionally try to *move the system into a flexible state*, i.e. a state that facilitates adaptation if necessary.“

[Branke&Mattfeld, to appear]

- Flexibility as secondary objective
- Easy to integrate into black box optimization heuristics

## Part II: Flexibility



- Motivation
- Challenges
- Example:  
Job shop scheduling

## Intuitive examples

- Portfolio optimization:  
Don't invest all your money long-term
- Transportation:  
Drive a route where additional customers are expected
- Manufacturing:  
Buy machines that can produce different products

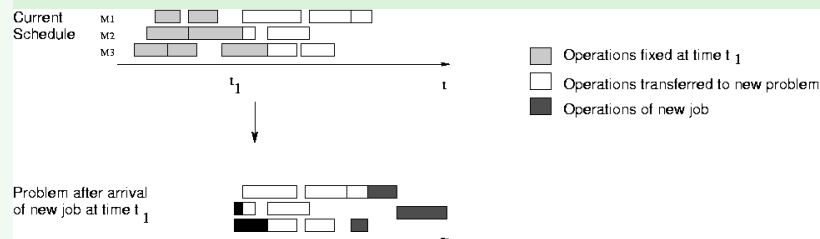
### Challenges:

1. What constitutes flexibility in the specific context?
2. How to integrate flexibility goal into the algorithm?

## Example: Minimum summed tardiness scheduling

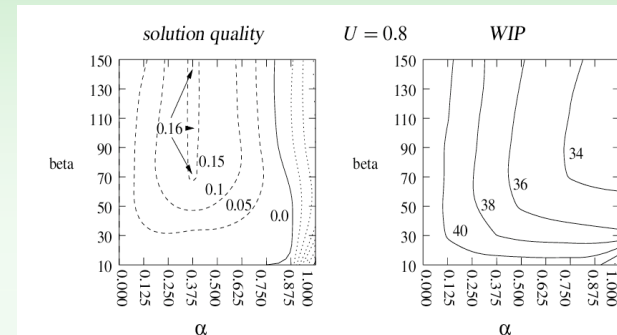
Problem:

- New jobs arrive dynamically and have to be integrated into the schedule
- Execute current best schedule until change occurs
- SEE [Branke & Mattfeld 2000, Branke & Mattfeld 2005]



## Result

- Reduction in tardiness objective:
  - Evolutionary algorithm: 14-22%
  - Biased random sampling: 15-18%
- Improvement relatively independent of parameter setting



What makes a schedule flexible?

- Flexibility = available machine capacity later in the schedule
- Secondary objective: avoid early idle time, penalty on idle time, linearly decreasing up to time  $\beta$

Integration into EA:

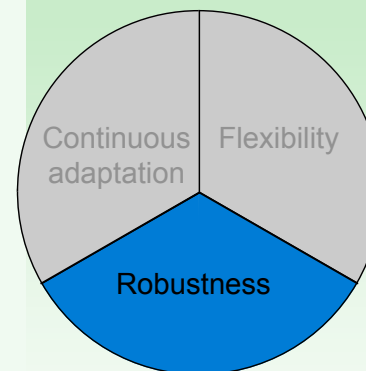
- Tardiness and idle time penalty normalized w.r.t. population max and min
- Fitness = linear combination of tardiness and idle time penalty

$$f_k = (1 - \alpha)\hat{T} + \alpha\hat{P}$$

$$\hat{T} = \frac{T_k - \min\{T_m\}}{\max\{T_m\} - \min\{T_m\}}$$

$$\hat{P} = \frac{P_k - \min\{P_m\}}{\max\{P_m\} - \min\{P_m\}}$$

## Part III: Robustness



- Robustness against
  - environmental changes
  - implementation noise
- Estimating the effective fitness
- Trade-off between fitness and robustness

## Two Variants

### 1. Environment changes, but adaptation is not possible

- Environment changes too quickly
- Adaptation too expensive
- Adaptation technically impossible
- Commitment long term

Solution needs to have high quality **even if environment changes**

### 2. Implementation of solution is prone to errors

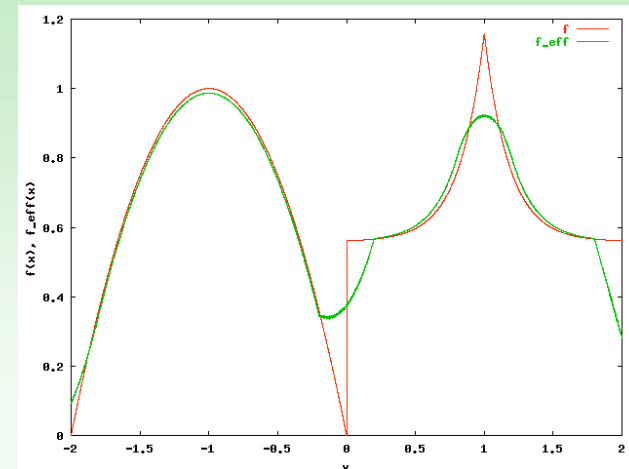
- Manufacturing tolerances
- Growth processes

Solution has to have high quality **even if modified slightly**

Both variants can be treated the same way.

## Effective Fitness - Example [Branke 2001]

$\delta$  uniformly distributed in  $[-0.2, \dots, 0.2]$



## Effective Fitness

- Given: probability distribution over different scenarios / deviations
- Goal (*effective* fitness):
  - optimize **expected value**
  - optimize worst case
  - ...

$$x \rightarrow x + \delta$$

$$f(x) \rightarrow f_{\text{eff}}(x) = E(f(x + \delta)) = \int_{-\infty}^{\infty} f(x + \delta) \varphi(\delta) d\delta$$

$\varphi(\delta)$ : probability density function of  $\delta$

- Effective fitness can be determined by Monte-Carlo integration

## Efficiently estimating expected values



Simply disturb individuals [Tsutsui & Ghosh 1997]



Multiple samples [Branke 1998]



Latin hypercube sampling  
[Loughlin & Ranjithan 1999, Branke 2001b]



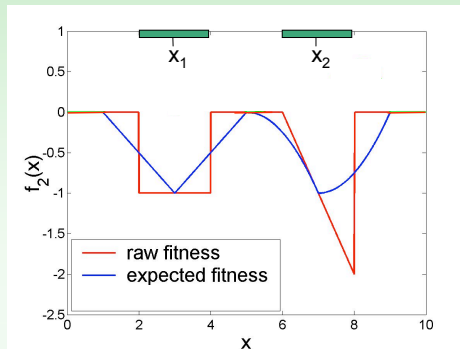
Use history of search [Branke 1998]



Use approximation models [Paenke et al., to appear]

## Trade-off between solution quality and robustness

- Variance as second objective
- Evolutionary multi-objective optimization  
[Ray 2002, Jin & Sendhoff 2003]



## Further readings

- EvoDOP repository and mailinglist  
<http://www.aifb.uni-karlsruhe.de/~jbr/EvoDOP>
- Workshop on Evolutionary Optimization in Stochastic and Dynamic Environments (EvoSTOC)  
[Every year in combination with EvoWorkshops](#)
- Books:
  - J. Branke: „Evolutionary Optimization in Dynamic Environments“, Kluwer, 2001
  - R. Morrison: „Designing Evolutionary Algorithms for Dynamic Environments“, Springer, 2004
  - K. Weicker: „Evolutionary Algorithms and Dynamic Optimization Problems“, Der Andere Verlag, 2003
- Journals:
  - Soft Computing Journal, special issue on dynamic optimization problems (to appear)
  - IEEE Transactions on Evolutionary Computation, special issue on optimization in uncertain environments (to appear)
  - Genetic Programming and Evolvable Machines, special issue on dynamic and uncertain environments (deadline: July 31)

## Conclusion Part I-III

- Very interesting and active research area
- Still in its infancy
- Nature inspired optimization has a lot to offer
  - Continuous adaptation
  - Multi-objective optimization
    - Flexibility
    - Robustness
    - Change-cost
  - Ability to cope with uncertainty (Robustness)
    - Multiple sampling of promising areas
    - Population information may be used

## References

- [Andersen 1991] H. C. Andersen. An investigation into genetic algorithms, and the relationship between speciation and the tracking of optima in dynamic functions. Honours thesis, Queensland University of Technology, Brisbane, Australia
- [Arnold & Beyer 2002] D. V. Arnold and H.-G. Beyer. Random Dynamics Optimum Tracking with Evolution Strategies. In J.J. Merelo, P. Adamidis, H.-G. Beyer, J.L. Fernández-Villacañas, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature*, pages 3-12, Springer.
- [Blackwell & Bentley 2002] T. M. Blackwell, and P. J. Bentley, P.J.: Dynamic search with charged swarms. *Genetic and Evolutionary Computation Conference*, pages 19-26
- [Blackwell & Branke 2004] T. M. Blackwell, and J. Branke. Multi-swarm optimization in dynamic environments. In: G. Raidl et al. (eds), *Applications of Evolutionary Computing*. Springer, LNCS 3005, pages 489-500
- [Branke 1999] J. Branke. Memory enhanced evolutionary algorithms for changing optimization problems. In *Congress on Evolutionary Computation CEC99*, Volume 3, pages 1875-1882. IEEE
- [Branke et al. 2000] J. Branke, T. Kaußler, C. Schmidt, and H. Schmeck. A multi-population approach to dynamic optimization problems. In *Adaptive Computing in Design and Manufacturing*. Springer
- [Branke 2001] J. Branke. *Evolutionary Optimization in Dynamic Environments*. Kluwer
- [Branke 2001b] J. Branke. Reducing the sampling variance when searching for robust solutions. In: *Genetic and Evolutionary Computation Conference*, L. Spector et al. (eds.), Morgan Kaufmann, pages 235-242
- [Branke & Mattfeld 2000] J. Branke and D. Mattfeld. Anticipation in dynamic optimization: The scheduling case. In: *Parallel Problem Solving from Nature*, Springer, pages 253-262
- [Branke & Mattfeld 2005] J. Branke and D. Mattfeld. Anticipation and flexibility in dynamic scheduling. In: *International Journal of Production Research*, to appear
- [Branke et al. 2005] J. Branke, E. Salihoglu, S. Uyar. Towards an analysis of dynamic environments. In: *Genetic and Evolutionary Computation Conference*, ACM, 2005

- [Carlisle & Dozier 2000] Carlisle, A. and Dozier, G.: Adapting Particle Swarm Optimization to Dynamic Environments. *Int. Conference on Artificial Intelligence*, pages 429-434
- [Cedeno & Vemuri 1997] W. Cedeno and V. R. Vemuri. On the use of niching for dynamic landscapes. In *International Conference on Evolutionary Computation*. IEEE.
- [Cobb 1990] H. G. Cobb. An investigation into the use of hypermutation as an adaptive operator in genetic algorithms having continuous, time-dependent nonstationary environments. Technical Report AIC-90-001, Naval Research Laboratory, Washington, USA.
- [Droste 2003] S. Droste. Analysis of the (1+1) EA for a dynamically bitwise changing OneMax. In E. Cantu-Paz, editor, *Genetic and Evolutionary Computation Conference*, volume 2723 of *LNCS*, pages 909-921. Springer
- [Goldberg & Smith 1987] D. E. Goldberg and R. E. Smith. Nonstationary function optimization using genetic algorithms with dominance and diploidy. In J. J. Grefenstette, editor, *Second International Conference on Genetic Algorithms*, pages 59-68. Lawrence Erlbaum Associates, 1987
- [Grefenstette 1992] J. Grefenstette. Genetic algorithms for changing environments. In R. Maenner and B. Manderick, editors, *Parallel Problem Solving from Nature 2*, pages 137-144. North Holland
- [Guntsch et al. 2001] M. Guntsch, M. Middendorf, and H. Schmeck. An Ant Colony Optimization Approach to Dynamic TSP. In: L. Spector et al. (eds.) *Genetic and Evolutionary Computation Conference*, San Francisco, CA: Morgan Kaufmann pages 860-867.
- [Hu & Eberhart 2002] X. Hu, and R. C. Eberhart. Adaptive particle swarm optimisation: detection and response to dynamic systems. *Congress on Evolutionary Computation*. Pages 1666-1670
- [Janson & Middendorf 2004] T. S. Janson and M. Middendorf. A hierarchical particle swarm optimizer for dynamic optimization problems. In: G. Raidl et al. (eds), *Applications of Evolutionary Computing*. Springer, LNCS 3005, pages 513-524
- [Jin & Sendhoff 2003] Y. Jin and B. Sendhoff. Trade-off between optimality and robustness: An evolutionary multiobjective approach. In: *Intl. Conference on Evolutionary Multi-criterion Optimization*, LNCS 2632, Springer, pages 237-251
- [Jin & Branke 2005] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments - A survey. In: *IEEE Transactions on Evolutionary Computation*, to appear
- [Lewis et al. 1998] J. Lewis, E. Hart, and G. Ritchie. A comparison of dominance mechanisms and simple mutation on non-stationary problems. In A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, *Parallel Problem Solving from Nature*, number 1498 in *LNCS*, pages 139-148. Springer

- [Loughlin & Ranjithan 1999] D. H. Loughlin and S. Ranjithan. Chance-constrained genetic algorithms. In: Banzhaf et al. (eds.), *Genetic and Evolutionary Computation Conference*, Morgan Kaufmann, pages 369-376
- [Mattfeld & Bierwirth 2004] D. C. Mattfeld and C. Bierwirth. An efficient genetic algorithm for job shop scheduling with tardiness objectives. *European Journal of Operational Research* 155(3), pages 616-630
- [Mori et al. 1996] N. Mori, H. Kita, and Y. Nishikawa. Adaptation to a changing environment by means of the thermodynamical genetic algorithm. In H.-M. Voigt, editor, *Parallel Problem Solving from Nature*, number 1141 in *LNCS*, pages 513-522. Springer
- [Mori et al. 1997] N. Mori, S. Imanishi, H. Kita, and Y. Nishikawa. Adaptation to changing environments by means of the memory based thermodynamical genetic algorithm. In T. Bäck, editor, *Seventh International Conference on Genetic Algorithms*, pages 299-306. Morgan Kaufmann.
- [Morrison 2004] R. W. Morrison. Designing Evolutionary Algorithms for Dynamic Environments. Springer
- [Morrison & DeJong 1999] R. W. Morrison and K. A. DeJong. A test problem generator for non-stationary environments. In *Congress on Evolutionary Computation*, volume 3, pages 2047-2053. IEEE
- [Ng & Wong 1995] K. P. Ng and K. C. Wong. A new diploid scheme and dominance change mechanism for non-stationary function optimization. In *Sixth International Conference on Genetic Algorithms*, pages 159-166. Morgan Kaufmann
- [Parrott & Li 2004] D. Parrott and X. Li. A particle swarm model for tracking multiple peaks in a dynamic environment using speciation. In: *Congress on Evolutionary Computation*, pages 98-103. IEEE
- [Ramsey & Grefenstette 1993] C. L. Ramsey and J. J. Grefenstette. Case-based initialization of genetic algorithms. In S. Forrest, editor, *Fifth International Conference on Genetic Algorithms*, pages 84-91. Morgan Kaufmann.
- [Ray 2002] T. Ray. Constrained robust optimal design using a multi-objective evolutionary algorithm. In: *Congress on Evolutionary Computation*, IEEE, pages 419-424
- [Ronnwinkler et al. 2001] C. Ronnwinkler and C. O. Wilke and T. Martinetz. Genetic algorithms in time-dependent environments. In: L. Kallel et al. (eds.), *Theoretical Aspects of Evolutionary Computing*. Springer
- [Sasaki & Tokoro 1999] T. Sasaki and M. Tokoro. Evolving learnable neural networks under changing environments with various rates of inheritance of acquired characters: Comparison between darwinian and lamarckian evolution. *Artificial Life*, 5(3):203-223

- [Stanhope & Daida 1999] S. A. Stanhope and J. M. Daida. Genetic algorithm fitness dynamics in a changing environment. In *Congress on Evolutionary Computation*, volume 3, pages 1851-1858. IEEE
- [Tsutsui & Ghosh 1997] S. Tsutsui and A. Ghosh. Genetic algorithms with a robust solution searching scheme. *IEEE Transactions on Evolutionary Computation* 1(3): 201-208
- [Tsutsui et al. 1997] S. Tsutsui, Y. Fjimoto, and A. Ghosh. Forking genetic algorithms: Gas with serach space division schemes. *Evolutionary Computation* 5(1), pages 61-80
- [Trojanowski et al. 1997] K. Trojanowski, Z. Michalewicz, and Jing Xiao. Adding memory to the evolutionary planner/navigator. In *IEEE Intl. Conference on Evolutionary Computation*, pages 483-487.
- [Ursem 2000] R. K. Ursem. Multinational GA optimization techniques in dynamic environments. In D. Whitley, D. Goldberg, E. Cantu-Paz, L. Spector, I. Parmee, and H.-G. Beyer, editors, *Genetic and Evolutionary Computation Conference*, pages 19-26. Morgan Kaufmann
- [Ursem et al. 2002] R. K. Ursem, T. Krink, M.T. Jensen, and Z. Michalewicz: Analysis and modeling of control tasks in dynamic systems. In: *IEEE Transactions on Evolutionary Computation*, 6(4), Pates 378-389
- [Vavak et al. 1997] F. Vavak, K. Jukes, and T. C. Fogarty. Adaptive combustion balancing in multiple burner boiler using a genetic algorithm with variable range of local search. In T. Bäck, editor, *Seventh International Conference on Genetic Algorithms*, pages 719-726. Morgan Kaufmann.
- [Yang & Yao 2005] S. Yang and X. Yao. Experimental study on population-based incremental learning algorithms for dynamic optimization problems. In: *Soft Computing Journal*, to appear
- [Weicker 2003] K. Weicker. „Evolutionary Algorithms and Dynamic Optimization Problems“, Der Andere Verlag

# Questions

?