Optimization in Dynamic Environments

GECCO Tutorial 2004

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

• Many real-world applications are dynamic
  – Scheduling
  – Control problems
  – Vehicle routing
  – Portfolio optimization
  – etc.

• 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
Three aspects in dynamic environments

- Continuous adaptation
- Flexibility
- Robustness
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

Evolutionary Algorithms

Dynamic Optimization Problems
The problem of convergence

For static optimization problems, convergence is desired. If the problem is dynamic, convergence is dangerous.
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
Possible Remedies (2)

3. Maintain diversity throughout the run
   - Random Immigrants [Grefenstette 1992]
   - Thermodynamical GA [Mori et al. 1996]

   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 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**
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

\[
\min F = \langle E \rangle \times TH
\]

- Requires to tune parameter \(T\)
- Computationally expensive
Memory/Search-Approach [Branke 1999]

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

Sensible balance of exploration vs. exploitation
Self Organizing Scouts (SOS) [Branke 2001]

- Idea: Collect information about search space
- Whenever a local optimum has been found watch it with some scouts
- Basis 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
Forking [Tsutsui et al. 1997]

Basis population
Scout population 1
Scout population 2
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
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]
- 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
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
Performance Measure: Offline Error

Difficulty: best solution found is not sufficient

Use modified offline error $\mathcal{E}^*(T)$

$$\mathcal{E}^*(T) = \frac{1}{T} \sum_{t=1}^{T} (opt_t - e_t)$$

$$e_t = \max(e_t, e_{t+1}, \ldots, e_T)$$

$\mathcal{T}$: time of last change
Demo: Self-Organizing Scouts
Comparison of offline error

Offline Error (Population size 100, 10 peaks, step size 1.0)
Percentage of covered peaks

Evaluations (Population size 100, 10 peaks, step size 2.0)
Influence of step size

(After 5000 generations, 10 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
**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.
Additional Aspects: Theoretical Results

- (1+1) ES for the dynamic bit-matching problem [Droste 2003]
  - Maximal change severity such that runtime (first passage time) is still polynomial

- (m/m,l) ES for continuously moving sphere [Arnold/Beyer 2002]
  - Distance to optimum and mutation step size in equilibrium
  - No loss of diversity?
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
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
PSO for dynamic problems

- Re-initialization of local memory / replace with current location [Carlisle & Dozier 2000]
- Re-initialize part of the swarm population [Hu & Eberhart 2002]
- Charged particles [Blackwell & Bentley 2002]
- Hierarchical Swarms [Janson & Middendorf 2004]
- Multi Swarms [Blackwell & Branke 2004]

Combination of ideas from
- Charged PSO
- Self-Organizing Scouts
Charged PSO

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

\textbf{But:} difficult to control, $N^2$ complexity

Quantum PSO

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

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

- Motivation
- Challenges
- Example: Job shop scheduling

Continuous adaptation

Robustness

Flexibility
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
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]
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 \( b \)

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

\[
\begin{align*}
  f_k &= (1 - a) \hat{T} + a \hat{P} \\
  \hat{T} &= \frac{T_k \min\{T_m\}}{\max\{T_m\} - \min\{T_m\}} \\
  \hat{P} &= \frac{P_k \min\{P_m\}}{\max\{P_m\} - \min\{P_m\}}
\end{align*}
\]
Result

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

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

\[
x \rightarrow x + \mathcal{D}\\
\mathcal{f}(x) \quad \mathcal{f}_{\text{eff}}(x) = E(\mathcal{f}(x + \mathcal{D})) = \int \mathcal{f}(x + \mathcal{D}) \mathcal{j}(\mathcal{D}) d\mathcal{D}
\]

\[\mathcal{j}(\mathcal{D}):]\text{ probability density function of } \mathcal{D}\]
- Effective fitness can be determined by Monte-Carlo integration
Effective Fitness - Example [Branke 2001]

\[ \text{gleichverteilt in } [-0.2, ..., 0.2] \]
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
Trade-off between solution quality and robustness

- Variance as second objective
- Evolutionary multi-objective optimization
- [Sendhoff & Jin 2000]
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 noise
    • Multiple sampling of promising areas
    • Population information may be used
Further readings

- EvoDOP repository and mailinglist
  http://www.aifb.uni-karlsruhe.de/~jbr/EvoDOP
- Workshop on Evolutionary Optimization in Stochastic and Dynamic Environments (EvoSTOC)
- Books:
- 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)
References


Questions