Simple On-the-Fly Parameter Selection

Carola Doerr  
CNRS and Sorbonne University, Paris, France

Markus Wagner  
University of Adelaide, Australia

Presentation at GECCO 2018
Carola Doerr, Markus Wagner: Simple On-the-Fly Parameter Selection Mechanisms for Two Classical Discrete Black-Box Optimization Benchmark Problems
The Parameter Selection Problem

- Evolutionary algorithms and related iterative optimization heuristics are parametrized algorithms
  - Example: \((\mu + \lambda)\) EAs

```
Initialization of the population:
Sample search points \(X = \{x^1, ..., x^\mu\}\)
```

```
Variation:
Create \(\lambda\) offspring by mutating and recombining search points from \(X\)
```

```
Selection:
Update population \(X\) by selecting \(\mu\) individuals
```

- Parameters:
  - Memory size \(\mu\)
  - Offspring population size \(\lambda\)
  - Crossover rate
  - Mutation rate, search radius, etc
  - Selective pressure

How shall I set these parameters to get a well-performing EA?
Parameter Tuning vs. Parameter Control

- **Parameter Tuning:**
  - Initial set of experiments
  - Deduce reasonable parameter settings

  → Does not have to be done manually, but a number of powerful, ready-to-use tools available: irace, SPOT, ParamILS, SMAC, GGA,...

- **Parameter Control:**
  - 2 main differences:
    - Parameters are set *while* optimizing
    - Parameters *change over time*:
      
      Key motivation: different parameter values can be optimal in different stages of an optimization process
Goals of Parameter Control

- to **identify** good parameter values “on the fly”
- to **track** good parameter values when they change during the optimization process
Parameter Control

- Example: LeadingOnes: $\text{LO}(110110101010) = 2$
- Randomized Local search: flip $k$ bits, keep the better of parent and offspring

$$k_{\text{opt}}(x) = \left\lfloor \frac{1}{\text{LO}(x) + 1} \right\rfloor$$
Parameter Control

- Example: LeadingOnes: LO(110110101010)=2
- Randomized Local search: flip $k$ bits, keep the better of parent and offspring
- $n=1000$
Parameter Control

- Example: LeadingOnes: LO(110110101010) → 2
- Randomized Local search: flip bits, keep the better of parent and offspring
- n=1000

![Graph showing parameter control and optimization time]

How can I find/predict such a dependence???

22% smaller optimization time
Good News: You Don’t Have to!

- Easy mechanisms which find close-to-optimal parameter values automatically:

![Graph showing mutation strength vs. LO(x)]

- Optimal mutation strength
- Avg. mutation strength of adaptive EA
Good News: You Don’t Have to!

With close-to-optimal performance:

- optimal mutation strength
- Avg. mutation strength of adaptive EA
- Avg. hitting time of dynamic (1+1) EA
- Avg. hitting time of best static RLS
- Avg. hitting time of best dynamic RLS
Good News: You Don’t Have to!

- Running time for update strengths $A = 2, b = 1/2$
  
  ![Expected optimization times on n-dimensional LeadingOnes, normalized by n^2](chart)

- around 20.5% performance gain over the $(1+1) \text{EA}_{>0}$ with static mutation rate $p = 1/n$
- 14% performance gain over RLS
- larger gains possible for other combinations of $A$ and $b$
Success-Based Multiplicative Update Rule

**Algorithm 2**: The $(1 + 1) \text{EA}_\alpha$ with update strengths $A$ and $b$ and initial mutation rate $p_0 \in \left[1/n^2, 1/2\right]$ for the maximization of a pseudo-Boolean function $f : \{0, 1\}^n \rightarrow \mathbb{R}$

1. **Initialization**: Sample $x \in \{0, 1\}^n$ uniformly at random and compute $f(x)$;
2. Set $p = p_0$;
3. **Optimization**: for $t = 1, 2, 3, \ldots$ do
   4. Create offspring $y$ through standard bit mutation with mutation probability $p$
   5. if $f(y) \geq f(x)$ then
      6. $x \leftarrow y$ and $p \leftarrow \min\{A \cdot p, 1/2\}$
   else
      7. $p \leftarrow \max\{b \cdot p, 1/n^2\}$

A $> 1$

b $< 1$
Success-Based Multiplicative Update Rule

Algorithm 2: The \((1+1)\) EA with update strengths \(A\) and \(b\) and initial mutation rate \(p_0 \in [1/n^2, 1/2]\) for the maximization of a pseudo-Boolean function \(f : \{0, 1\}^n \rightarrow \mathbb{R}\)

1. **Initialization:** Sample \(x \in \{0, 1\}^n\) uniformly at random and compute \(f(x)\);
2. Set \(p = p_0\);
3. **Optimization:** for \(t = 1, 2, 3, \ldots\) do
   - Sample \(\ell\) from \(\text{Bin}_{>0}(n, p)\);
   - \(y \leftarrow \text{flip}_\ell(x)\);
   - evaluate \(f(y)\);
   - if \(f(y) \geq f(x)\) then
     - \(x \leftarrow y\) and \(p \leftarrow \min\{A \cdot p, 1/2\}\)
   - else
     - \(p \leftarrow \max\{b \cdot p, 1/n^2\}\)

Standard bit mutation, condition to flip at least one bit

\[A > 1\]
\[b < 1\]
LeadingOnes

- Average optimization time for different combinations of $A$ and $b$ (101 independent runs)

- For comparison: RLS needs $n^2/2$ iterations (=0.5 and =3.125 above), $(1+1)\ \text{EA}_{\geq 0}$ needs 0.54 and $3.4 \times 10^4$ iterations, respectively
LeadingOnes

- Average optimization time for different combinations of $A$ and $b$ (101 independent runs)

For comparison: RLS needs $n^2/2$ iterations (=0.5, =3.125, 1.25 above), (1+1) EA$_{>0}$ needs 0.54, $3.4 \times 10^4$, and $1.35 \times 10^5$ iterations, respectively
LeadingOnes

- Average optimization time for different combinations of $A$ and $b$ (101 independent runs)

  For comparison: RLS needs $n^2/2$ iterations (=1.25*10^5 for $n=500$), (1+1) EA$_{>0}$ needs 1.35*10^5 iterations, respectively

Carola Doerr and Markus Wagner: Simple On-the-Fly Parameter Selection
1/5-th Success Rules

- 1/5-th success rule:
  - originally from continuous optimization [Rechenberg, Devroye, Schumer/Steiglitz]
  - (1+1) ES optimizing sphere $f(x) = \sum x_i^2$
  - When success rate > 1/5: increase search radius
    When success rate < 1/5: decrease search radius

- In discrete optimization, e.g.,
  [Kern/Müller/Hansen/Büche/Ocenasek/Koumoutsakos04, Auger09]:
  - When success rate $\approx 1/5$, parameter value should be stable
  - In our algorithm:
    
    $$
    \text{If } f(y) \geq f(x): p \leftarrow \min\left\{Ap, \frac{1}{2}\right\} \\
    \text{else } p \leftarrow \max\{bp, 1/n^2\}
    $$

  - $A = \left(\frac{1}{b}\right)^{1/4}$ since $Ab^4 = 1$
  - $b = 1/A^4$
1/5-th Success Rules

- 1/5-th success rule:
  - originally from continuous optimization [Rechenberg, Devroye, Schumer/Steiglitz]
  - (1+1) ES optimizing sphere \( f(x) = \sum x_i^2 \)
  - When success rate > 1/5: increase search radius
  - When success rate < 1/5: decrease search radius

- In discrete optimization, e.g., [Kern/Müller/Hansen/Büche/Ocenasek/Koumoutsakos04, Auger09]:
  - When success rate \( \approx 1/5 \), parameter value should be stable

In our algorithm:

- If \( f(y) \geq \frac{1}{5} \):
  - \( A = \left( \frac{1}{b} \right)^{1/4} \)
- else
  - \( b = 1/A^4 \)
Results for the 1/5-th Success Rule

- LO, $n=500$, 100 independent runs
- RLS performance: 125,000 iterations
1:x Success Rules

- A priori no reason to restrict ourselves to a 1:5 success ratio
- We can also try different success rules
Average Optimization Times of 1:x Rules

- LO, n=500, 100 independent runs
- RLS performance: 125,000 iterations
Overall Performance Summary

- 50% of all configurations with $1 < A \leq 2.5$ and $0.4 \leq b < 1$ are better than RLS by at least 13%
Results for OneMax

Average Runtime on OneMax for Different Dimensions

<table>
<thead>
<tr>
<th>Dimension n</th>
<th>RLS</th>
<th>RLS_opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>445</td>
<td>436</td>
</tr>
<tr>
<td>500</td>
<td>3,050</td>
<td>2,974</td>
</tr>
<tr>
<td>1000</td>
<td>6,871</td>
<td>6,690</td>
</tr>
<tr>
<td>2000</td>
<td>14,809</td>
<td>14,722</td>
</tr>
<tr>
<td>3000</td>
<td>23,814</td>
<td>23,507</td>
</tr>
</tbody>
</table>

The graph shows the average optimization time for different dimensions using RLS and RLS_opt. As the dimension increases, the average runtime also increases for both methods, with RLS_opt generally showing a lower runtime compared to RLS.
Results for OneMax

Average Runtime on OneMax for Different Dimensions

<table>
<thead>
<tr>
<th>Dimension n</th>
<th>(1+1) EA_(&gt;0)</th>
<th>RLS</th>
<th>RLS_opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>679</td>
<td>445</td>
<td>436</td>
</tr>
<tr>
<td>500</td>
<td>4,756</td>
<td>3,050</td>
<td>2,974</td>
</tr>
<tr>
<td>1000</td>
<td>10,574</td>
<td>6,871</td>
<td>6,690</td>
</tr>
<tr>
<td>2000</td>
<td>24,352</td>
<td>14,809</td>
<td>14,722</td>
</tr>
<tr>
<td>3000</td>
<td>37,256</td>
<td>23,814</td>
<td>23,507</td>
</tr>
</tbody>
</table>
Results for OneMax

Average Runtime on OneMax for Different Dimensions

<table>
<thead>
<tr>
<th>Dimension n</th>
<th>(1+1) EA_&gt;0</th>
<th>A=1,1, b=0,66</th>
<th>A=1,2, b=0,85</th>
<th>A=1,3, b=0,75</th>
<th>A=2,0, b=0,5</th>
<th>RLS</th>
<th>RLS_opt</th>
</tr>
</thead>
<tbody>
<tr>
<td>100</td>
<td>679</td>
<td>447</td>
<td>450</td>
<td>450</td>
<td>455</td>
<td>445</td>
<td>436</td>
</tr>
<tr>
<td>500</td>
<td>4,756</td>
<td>3,039</td>
<td>3,059</td>
<td>3,033</td>
<td>3,013</td>
<td>3,050</td>
<td>2,974</td>
</tr>
<tr>
<td>1000</td>
<td>10,574</td>
<td>6,749</td>
<td>6,751</td>
<td>6,801</td>
<td>6,753</td>
<td>6,871</td>
<td>6,690</td>
</tr>
<tr>
<td>2000</td>
<td>24,352</td>
<td>15,134</td>
<td>14,801</td>
<td>14,974</td>
<td>14,613</td>
<td>14,809</td>
<td>14,722</td>
</tr>
<tr>
<td>3000</td>
<td>37,256</td>
<td>23,726</td>
<td>23,558</td>
<td>23,715</td>
<td>23,417</td>
<td>23,814</td>
<td>23,507</td>
</tr>
</tbody>
</table>
Heatmaps for OneMax

(a) OneMax with $n = 500$

(b) OneMax with $n = 1500$
$\gamma \% \text{ Configs better than } (1 + 1)^{EA_{>0}} \text{ by at least } x \%$

Even better results if we restrict to configurations with $1 < A \leq 2.5$ and $0.4 \leq b < 1$
1:x Rules, OneMax, $n=5000$, 100 independent runs

Carola Doerr and Markus Wagner: Simple On-the-Fly Parameter Selection
Next Steps

- **Theoretical performance guarantees** for the adaptive $(1+1)$ EA$_{\alpha}$
- Comparison with other adaptation schemes, e.g.,
  - Adaptive Pursuit [Thierens 05]
  - UCB algorithms from Machine Learning [Da Costa, Fialho, Schoenauer, Sebag 08-11]
  - $\varepsilon$-greedy algorithm from [Doerr, Doerr, Yang 16]
- Performance on other test functions
  - Real-world problems?
    - you are all cordially invited to collaborate on this!
- Want to know more about dynamic parameter choices?
  - confer the tutorial slides (available on my homepage)
Acknowledgments

- We thank Eduardo Carvalho Pinto for providing his implementation of the (1+1) EA and his contributions to a preliminary experimentation with the multiplicative parameter control mechanism.

- Our work was supported by a public grant as part of the Investissement d’avenir project, reference ANR-11-LABX-0056-LMH, LabEx LMH, and by the Australian Research Council project DE160100850.