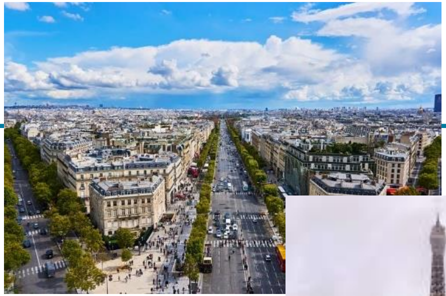
Dynamic Selection and Configuration of Optimization Algorithms:
From (Hyper-)Parameter Control to AutoML

Carola Doerr
CNRS research director at LIP6, Sorbonne Université, Paris, France
Homepage: https://webia.lip6.fr/~doerr/

My Background




- since 2013: CNRS researcher at Sorbonne Université in Paris
  - great research environment
  - great position: 100% research
    (student supervision + teaching as much as I like)
- 2008: “diplom” in Mathematics at the University of Kiel, Germany
- 2008-2009: Business Consultant with McKinsey & Company
  - focus: logistics (post, trains, container ships)
  - favorite projects: network optimization

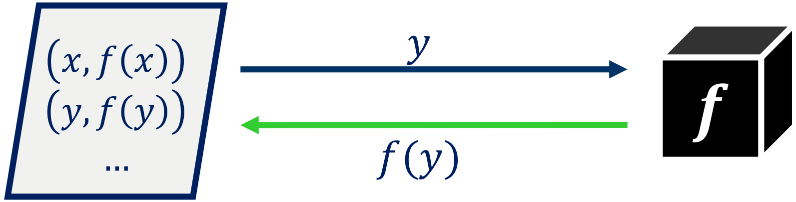
big surprise for me at the time: *heuristics and black-box optimization everywhere*

- 2010-2011: PhD in Computer Science at
  Max Planck Institute for Informatics and Saarland University
  in Saarbrücken
  - focus: theoretical aspects of Computer Science
    (performance guarantees and complexity statements for black-box optimization)

Black-Box Optimization

- **Optimization:** Given $f: \mathcal{S} \to \mathbb{R}$, find $x^* \in \mathcal{S}$ with $f(x^*)$ as large as possible

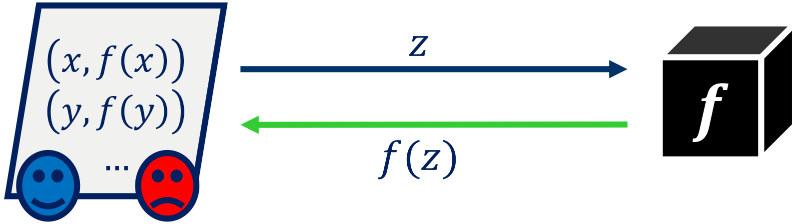
- **Black-Box** Optimization:



Evaluations:
- simulations
- physical experiments
- user study

Black-Box Optimization

- **Optimization:** Given $f: \mathcal{S} \to \mathbb{R}$, find $x^* \in \mathcal{S}$ with $f(x^*)$ as large as possible
- **Black-Box** Optimization:

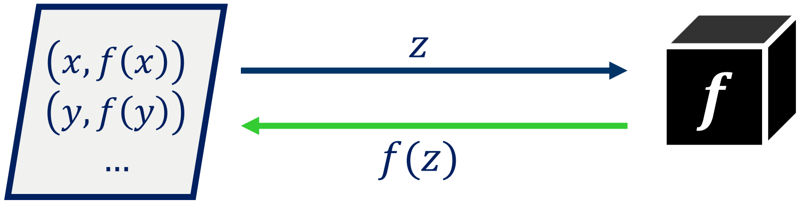


Evaluations:

- simulations
- physical experiments
- user study

- **Key objective:** find a good solution x^* for f using as few function evaluations as possible
  (yes, we often care only about *query complexity*, not CPU time or similar)
- **Applications:**
  - wherever simulations or experiments are needed to evaluate solution candidates (e.g., because we do not have an explicit model for the problem)
    (biology, engineering, machine learning, artificial intelligence, …)
  - no problem-specific algorithms available
    (lack of time, knowledge, other resources, …)
  - privacy concerns
    (e.g., cannot give full model to algorithm designer, only evaluations)

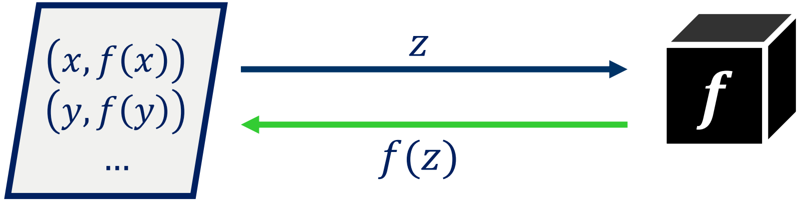
Black-box optimization algorithms



1. How many solution candidates to evaluate next?

2. How to generate them?

Black-box optimization algorithms



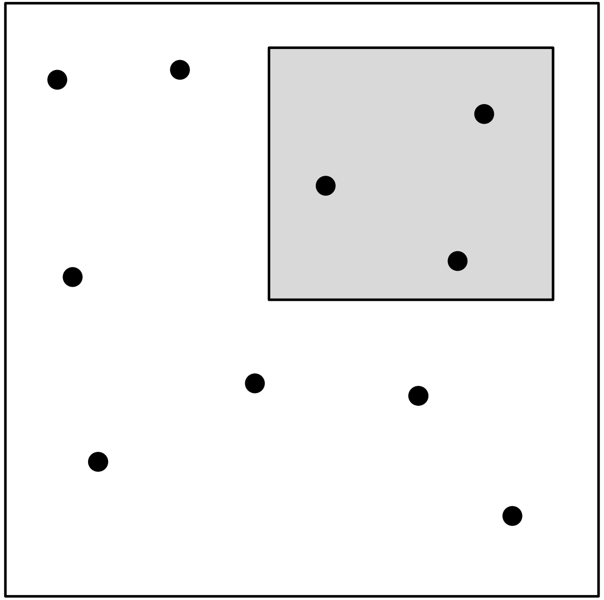
1. ~~How many solution candidates to evaluate next?~~
   (typically imposed by our resources)
2. How to generate them?

$f: \{1, \dots, 10\} \times \{1, \dots, 20\} \to \mathbb{R}$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
1																				
2																				
3																				
4																				
5																				
6																				
7																				
8																				
9																				
10																				

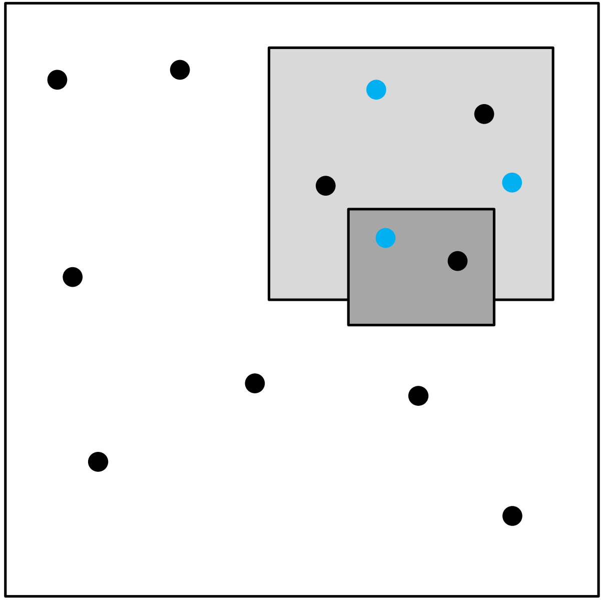
Dynamic Parameter Settings

- when optimizing black-box problems, we often want to
  - *explore* first: sample in different parts of the domain to obtain a feeling for the global structure and where to find promising regions
  - then *exploit*: focus on these most promising regions and *converge*



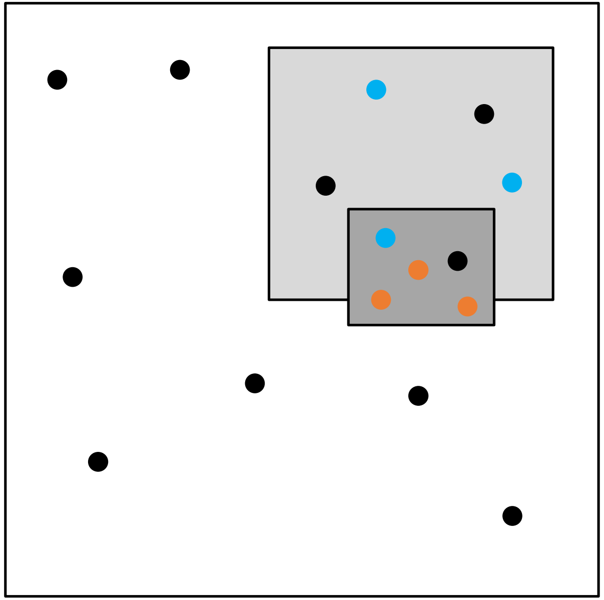
Dynamic Parameter Settings

- when optimizing black-box problems, we often want to
  - *explore* first: sample in different parts of the domain to obtain a feeling for the global structure and where to find promising regions
  - then *exploit*: focus on these most promising regions and *converge*



Dynamic Parameter Settings

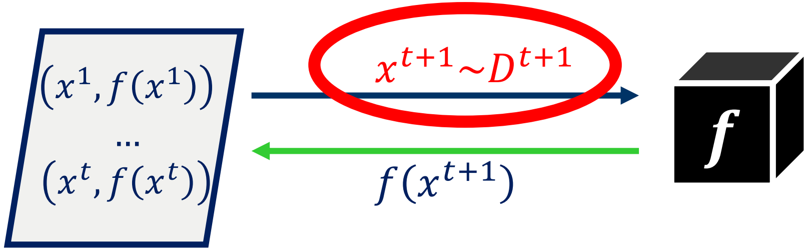
- when optimizing black-box problems, we often want to
  - *explore* first: sample in different parts of the domain to obtain a feeling for the global structure and where to find promising regions
  - then *exploit*: focus on these most promising regions and *converge*



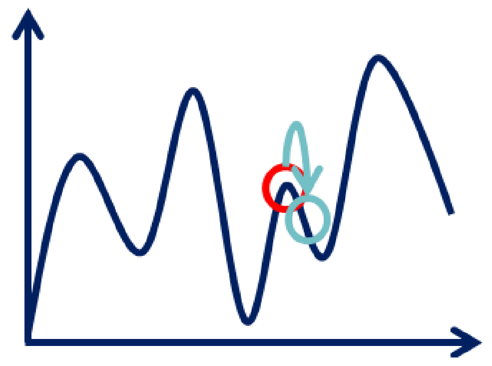
Key Goal:
automated, data-driven adjustment of the search

Dynamic Parameter Settings

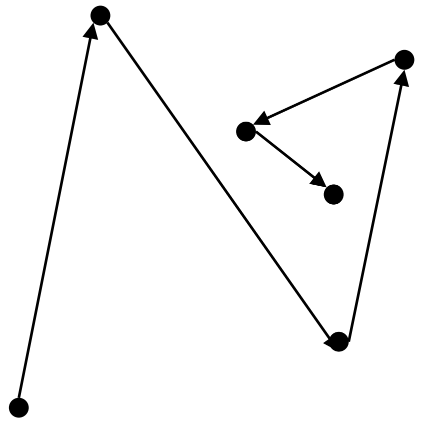
- when optimizing black-box problems, we often want to
  - *explore* first: sample in different parts of the domain to obtain a feeling for the global structure and where to find promising regions
  - then *exploit*: focus on these most promising regions and *converge*
- Formally, we ask for an automated adjustment of the distribution from which we sample the next solution candidate(s):



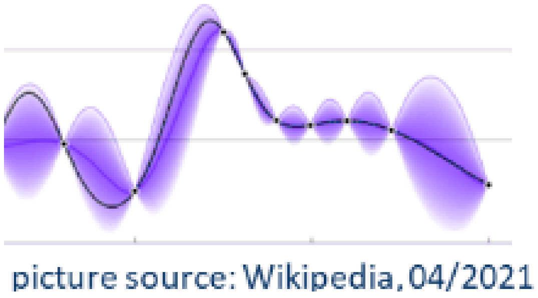
Example: Scheduled Parameter Selection



"Cooling" of Temperature in Simulated Annealing



Decreasing Mutation Rates in Evolutionary Algorithms

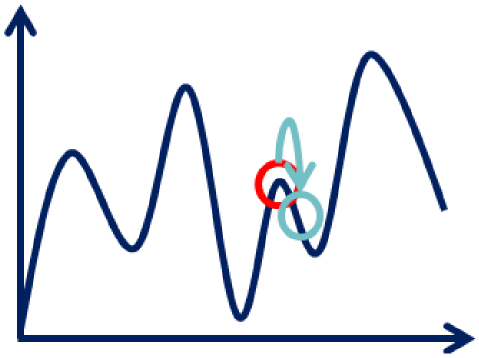


Choice of Acquisition Function in Bayesian Opt.

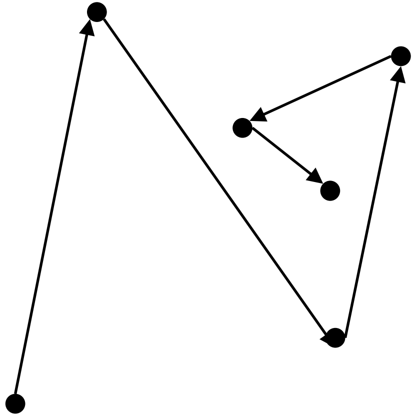
Can be quite efficient.
But:

- requires a well-designed schedule,

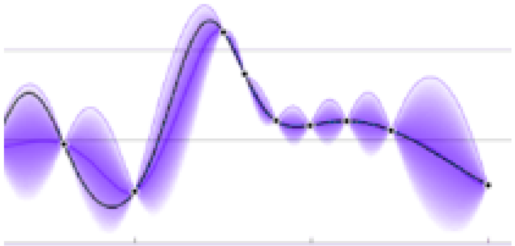
Example: Scheduled Parameter Selection



"Cooling" of Temperature in Simulated Annealing



Decreasing Mutation Rates in Evolutionary Algorithms



Choice of Acquisition Function in Bayesian Opt.

Can be quite efficient.
But:

- requires a well-designed schedule,
- suboptimal: does not take into account information obtained during the optimization process

Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

<u>The (1+1) Evolutionary Algorithm for maximizing $f\colon \{0,1\}^n \to \mathbb{R}$</u>

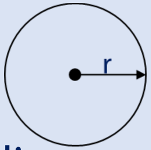
Initialization:

1. choose $x \in \{0,1\}^n$ uniformly at random (u.a.r.)

Optimization: in iteration $t = 1,2,\ldots$ do

1. $y \leftarrow \text{Mutation}(x, p)$ \\variation
2. If $f(y) \geq f(x)$
   replace x by y \\ selection

$y \leftarrow \text{Mutation}(x, p)$:
$y = x$, but we invert each bit with probability p
(equivalently: select search radius $r \sim \text{Bin}(n, p)$, then sample y at radius r around x)



Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

The (1+1) Evolutionary Algorithm with ``1-5th success rule"

If 1 out of 5 iterations is successful, then $p \leftarrow Ab^4$
Recommendation: set A, b such that $Ab^4 = 1$

Initialization:

1. choose $x \in \{0,1\}^n$ uniformly at random (u.a.r.)
2. initialize $p = p_{\text{init}}$

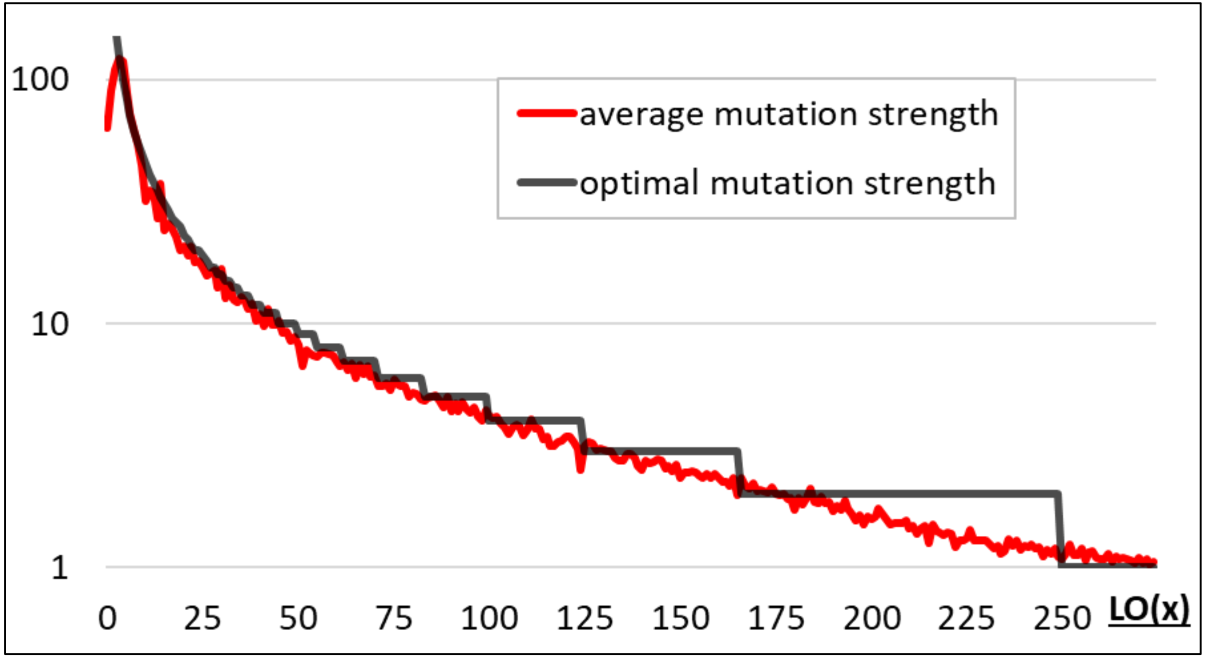
Optimization: in iteration $t = 1,2, \dots$ do

1. $y \leftarrow \text{Mutation}(x, p)$
2. If $f(y) \geq f(x)$
   replace x by y \\ selection
   replace p by Ap \\ parameter update
3. If $f(y) < f(x)$
   replace p by bp \\ parameter update

Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

The (1+1) Evolutionary Algorithm with ``1-5th success rule"



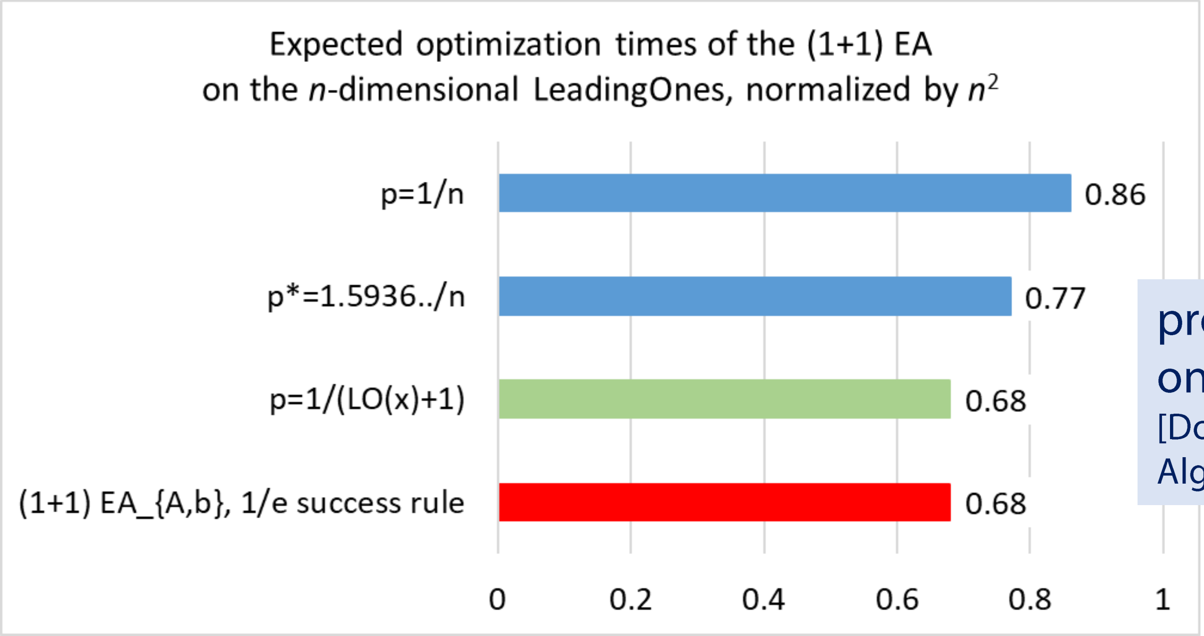
Results for LeadingOnes, $n = 500$,
zoom into $\mathrm{LO}(x) \leq 250$

Update strengths: $A = 2, b = 1/2$
from [Doerr, Wagner, GECCO 2018]

Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

The (1+1) Evolutionary Algorithm with ``1-5th success rule"

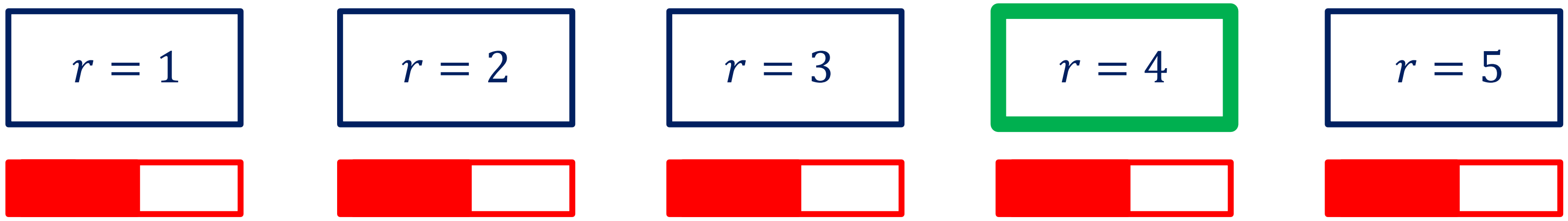


proven optimality on LeadingOnes
[Doerr, Doerr, Lengler. Algorithmica 2021]

Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

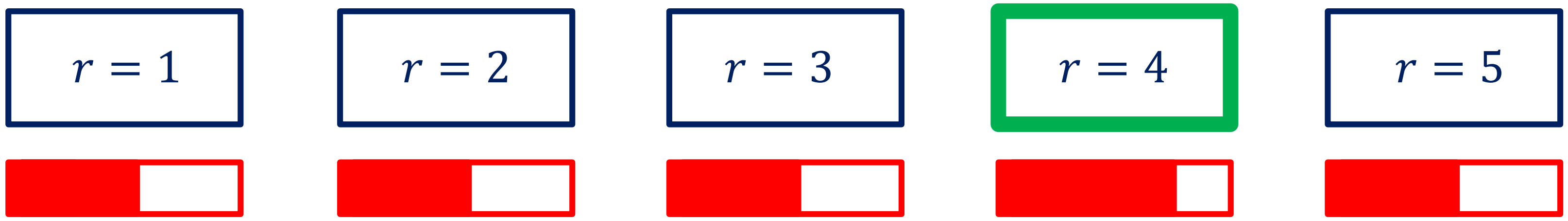
Dynamic Parameter Selection as a Multi-Armed Bandit Problem



Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

Dynamic Parameter Selection as a Multi-Armed Bandit Problem



ε-greedy reinforcement learning:

- w/probability ε chose random expert //exploration
- otherwise chose value with highest confidence //exploitation

After each step, update the confidence values

Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

Dynamic Parameter Selection as a Multi-Armed Bandit Problem

$r = 1$ $r = 2$ $r = 3$ $r = 4$ $r = 5$

ε-greedy reinforcement learning:

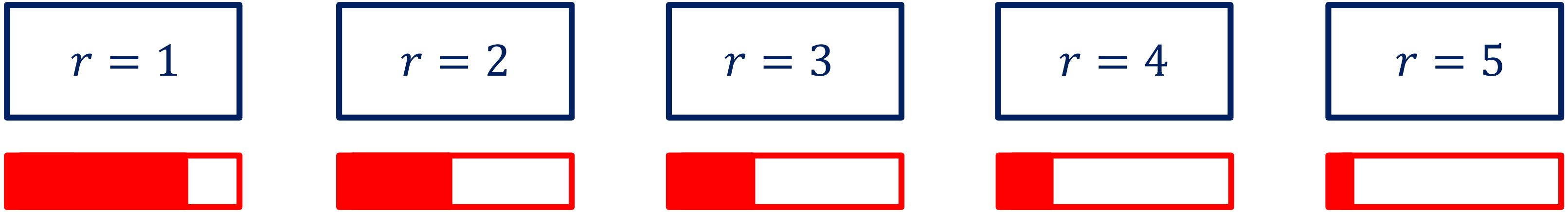
- w/probability ε chose random expert //exploration
- otherwise chose expert with highest confidence //exploitation

After each step, update the confidence values

Self-Adjusting Parameter Selection

Adjust the (hyper-)parameters during the run, but the update depends on the success of previous iterations

Dynamic Parameter Selection as a Multi-Armed Bandit Problem



Proven optimality on OneMax [Doerr, Doerr, Yang. PPSN 2016]

But: very sensitive with respect to its own hyperparameters!

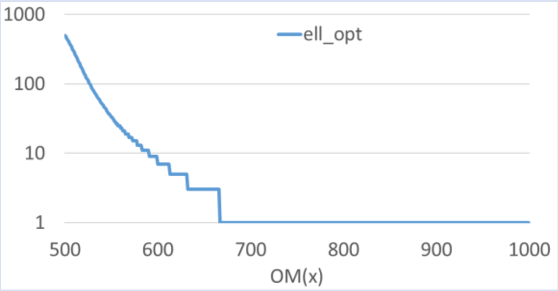
And: requires good correlation between

- progress in solution quality and

- progress in optimization behavior

→ this is not always the case!

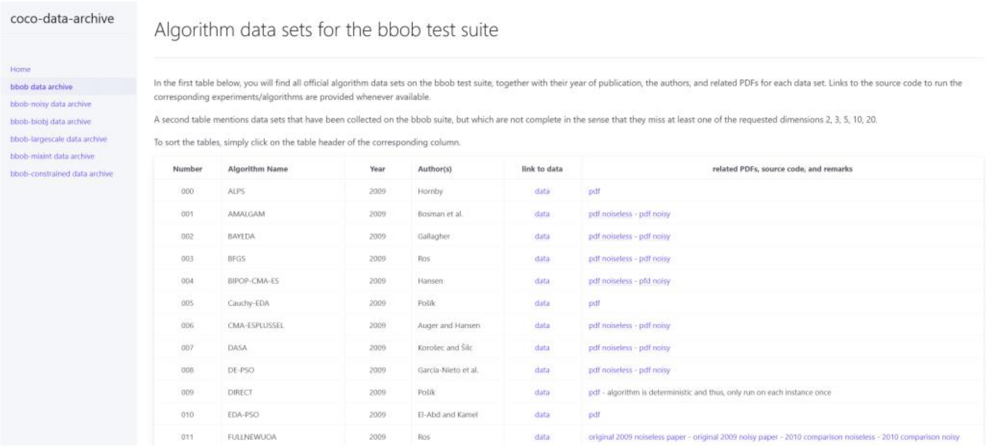
(which implies that it's hard to measure the reward of an action)



Dynamic Parameter Selection

So far: we only focused on our specific optimization problem

But:


coco-data-archive

- Home
- bbob data archive
- bbob-noisy data archive
- bbob-biobj data archive
- bbob-largescale data archive
- bbob-mixint data archive
- bbob-constrained data archive

Algorithm data sets for the bbob test suite

In the first table below, you will find all official algorithm data sets on the bbob test suite, together with their year of publication, the authors, and related PDFs for each data set. Links to the source code to run the corresponding experiments/algorithms are provided whenever available.

A second table mentions data sets that have been collected on the bbob suite, but which are not complete in the sense that they miss at least one of the requested dimensions 2, 3, 5, 10, 20.

To sort the tables, simply click on the table header of the corresponding column.

Number	Algorithm Name	Year	Author(s)	link to data	related PDFs, source code, and remarks
000	ALPS	2009	Hornby	data	pdf
001	AMALGAM	2009	Bosman et al.	data	pdf noiseless - pdf noisy
002	BAYEDA	2009	Gallagher	data	pdf noiseless - pdf noisy
003	BFGS	2009	Ros	data	pdf noiseless - pdf noisy
004	BIPOP-CMA-ES	2009	Hansen	data	pdf noiseless - pfd noisy
005	Cauchy-EDA	2009	Pošík	data	pdf
006	CMA-ESPLUSSEL	2009	Auger and Hansen	data	pdf noiseless - pdf noisy
007	DASA	2009	Korošec and Šilc	data	pdf noiseless - pdf noisy
008	DE-PSO	2009	García-Nieto et al.	data	pdf noiseless - pdf noisy
009	DIRECT	2009	Pošík	data	pdf - algorithm is deterministic and thus, only run on each instance once
010	EDA-PSO	2009	El-Abd and Kamel	data	pdf
011	FULLNEWUOA	2009	Ros	data	original 2009 noiseless paper - original 2009 noisy paper - 2010 comparison noiseless - 2010 comparison noisy

Nevergrad OpenML

We have a lot (A LOT!) of data from solving other problems and from benchmarking

How can we make use of this data to select parameters *on the fly*?

Dynamic Parameter Selection

My (our ?) dream:
automated configuration system that combines information from

- previous experiments
- current optimization process

to select hyperparameters for next iteration.

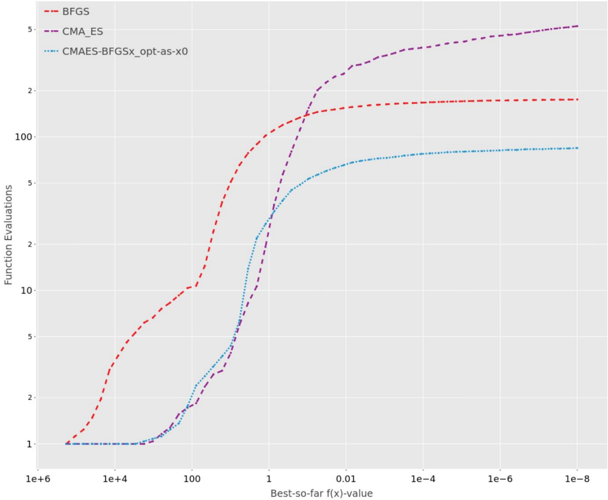
Dynamic Algorithm and Parameter Selection

My (our ?) dream:
automated configuration system that combines information from

- previous experiments
- current optimization process

to select an algorithm and its hyperparameters for next iteration.

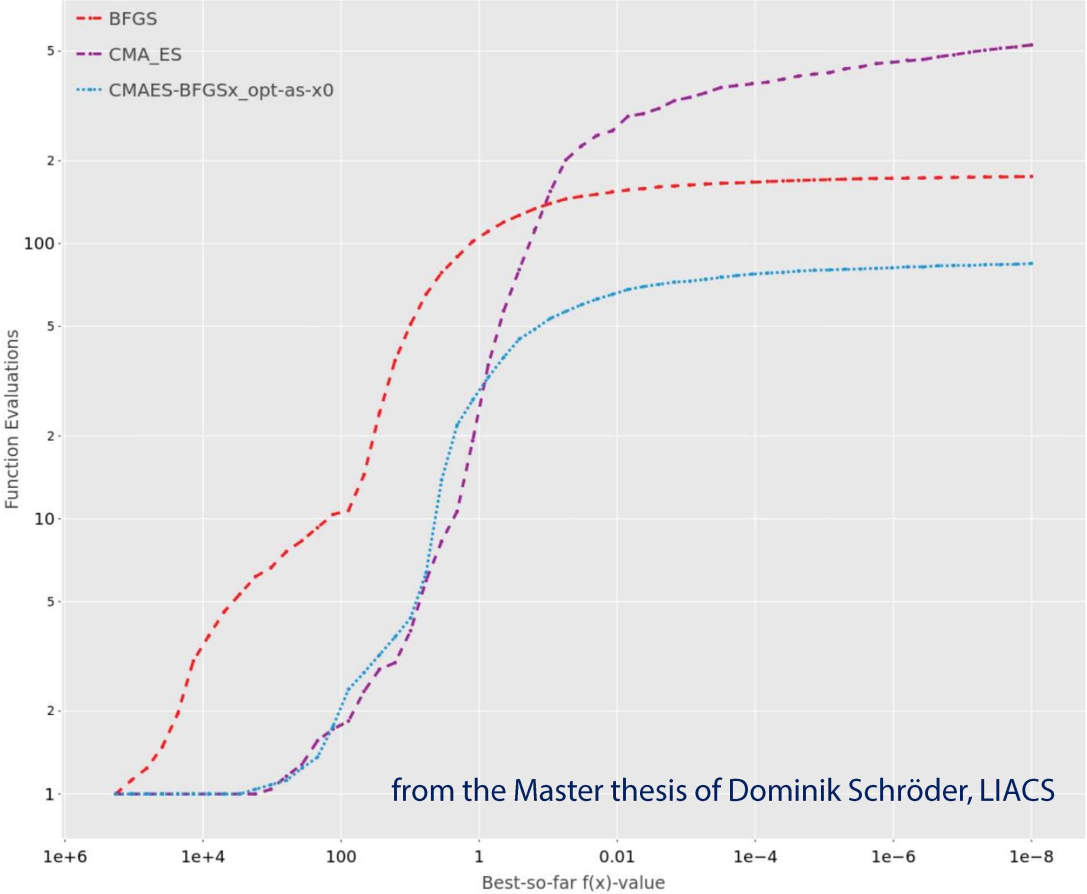
What we have discussed for hyperparameters also holds for algorithms: one algorithm may be good for certain parts of the problem, another one better suited for other parts.



Dynamic Algorithm and Parameter Selection

My (our ?) drea
automated cor
- previous
- current
to select an alg

What we have discussed fo
also holds for algorithms: c
good for certain parts of th
one better suited for other



Dynamic Algorithm and Parameter Selection

My (our ?) dream:
automated configuration system that combines information from

- previous experiments
- current optimization process

to select <u>an algorithm</u> and its hyperparameters for next iteration.

Dynamic Algorithm Configuration (André Biedenkapp, H. Furkan Bozkurt, Theresa Eimer, Frank Hutter, Marius Lindauer. ECAI'20)*:
Find a policy $\pi\colon \mathcal{I} \times \mathcal{S} \rightarrow \Theta$ that assigns to each (instance, state) pair an algorithm and its configuration such that the expected cost $\int_{\mathcal{I}} p(i) c(i, \pi)\, \mathrm{d}i$ is minimized

* see also *Automated Dynamic Algorithm Configuration* by Steven Adriaensen, André Biedenkapp, Gresa Shala, Noor Awad, Theresa Eimer, Marius Lindauer, Frank Hutter. arXiv 2022

Dynamic Algorithm and Parameter Selection

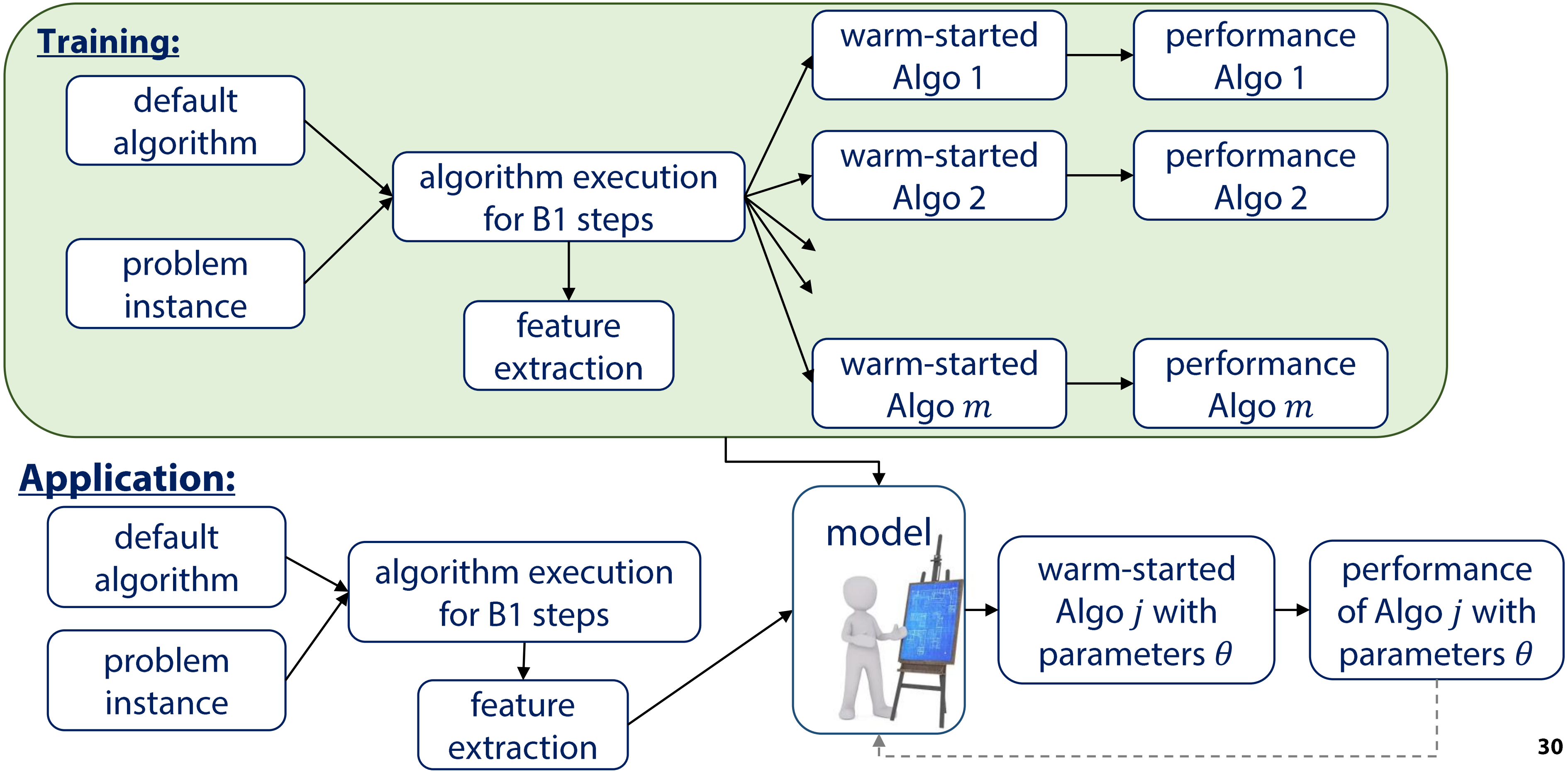
need: ML <u>and</u> optimization expertise

ML	**Opt.**
Which models to use, how to train them, how to generalize, etc.	which algorithms to combine, which components to tune, how to warmstart, etc.

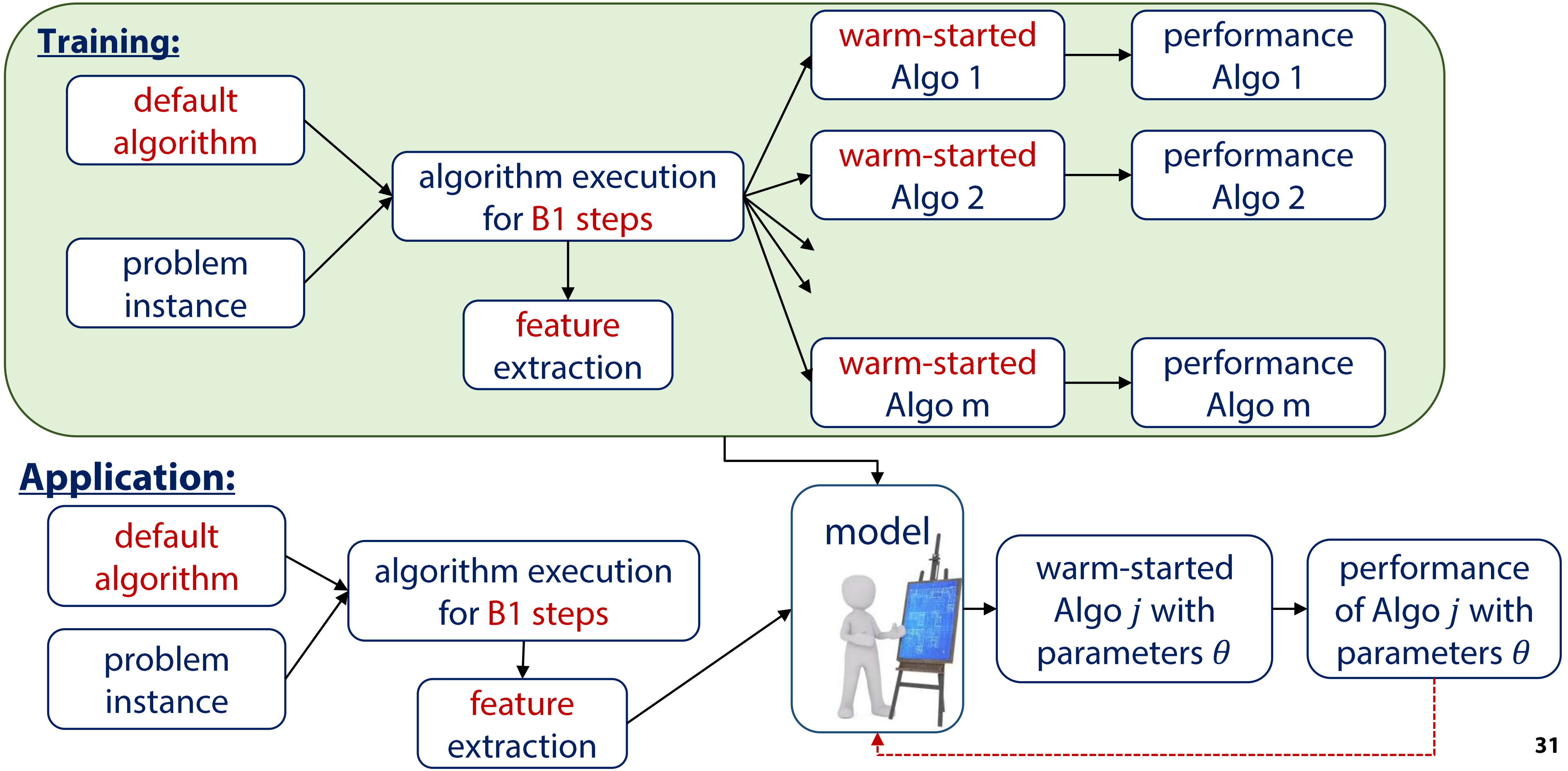
Key Challenges for DAC

1. From the whole universe of algorithms, which ones to preselect?
2. For each algorithm, which components to configure?
3. How to warmstart algorithms?
4. How to describe the state of an algorithm?
5. How to characterize the instances and how to measure similarity?
6. How to generate diverse training instances?
7. How to assign rewards/loss/total cost to individual actions?
8. …

Case Study: ELA-based Algorithm Selection



Case Study: ELA-based Algorithm Selection



Case Study: DAC using RL (DDQN)

Theory-inspired Parameter Control Benchmarks for Dynamic Algorithm Configuration

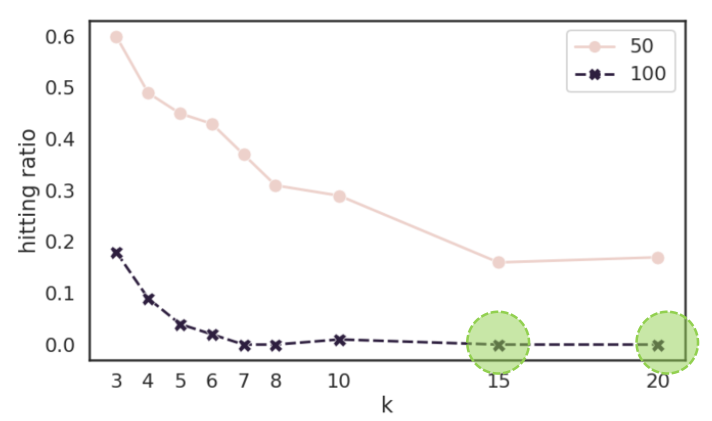
André Biedenkapp
University of Freiburg
Freiburg, Germany

Nguyen Dang
University of St Andrews
St Andrews, United Kingdom

Martin S. Krejca
Sorbonne Université, CNRS, LIP6
Paris, France

Frank Hutter
University of Freiburg, Germany
Bosch Center for Artificial Intelligence

Carola Doerr
Sorbonne Université, CNRS, LIP6
Paris, France

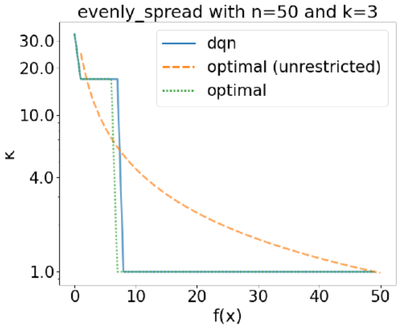


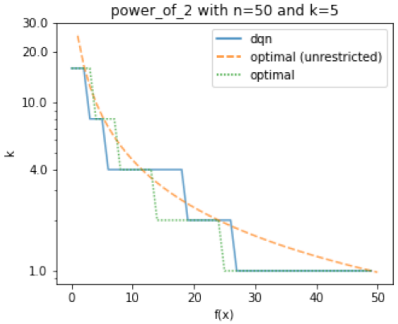
$n \in \{50, 100\}$ and $k \in \{3, 4, 5, 6, 7, 8, 10, 15, 20\}$

Benchmark Generator:
for given problem size and portfolio of actions,

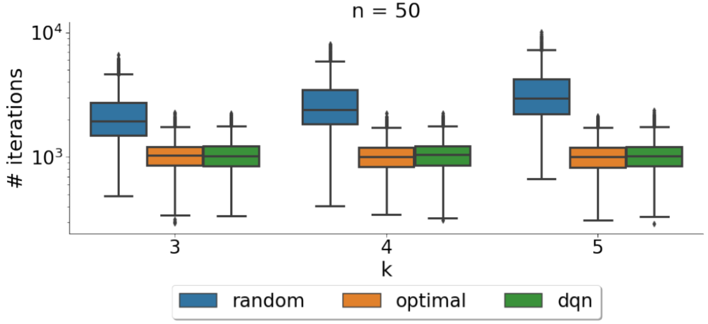
1 2 4 8

we know the optimal policy (theoretical results)
and can compare them to the output of agents trained by RL:





Example DDQN best learnt policies vs optimal policies

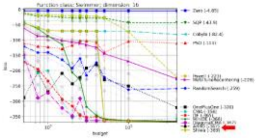


Performance of DDQN learnt policies vs optimal/random policies

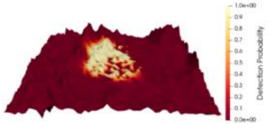
Combining theory, benchmarking, and practice

Theory

Benchmarking



Practice



different results and approaches help us get a more complete understanding

different way of looking at things,
different questions,
different inspiration,…

Our dream team ☺

Carola Doerr

Anja Jankovic

Elena Raponi

Alexis Robbes

Koen van der Blom

Martin Krejca
(now at École Polytechnique)

Mara Santarelli

Océane Fourquet

François Clement

Maria Laura Santoni

Duri Janett

Universiteit Leiden

UNIVERSIDADE D COIMBRA

Meta

THALES

AutoML.org Freiburg-Hannover

École Polytechnique IP PARIS

HRI Honda Research Institute

INSTITUT PASTEUR

ETH zürich

TUM TECHNISCHE UNIVERSITÄT MÜNCHEN

Our team's main research objectives

1. develop efficient black-box optimization algorithms

2. understand which algorithms to choose for which settings

depends on
- problem characteristics
- resources (#of evaluations, CPU hours, parallelization)

3. make this knowledge and the algorithms available to practitioners

IOHProfiler Nevergrad

Dynamic Algorithm and Parameter Selection

Beautiful dream: automated configuration system that combines information from

- previous experiments
- current optimization process

to select an algorithm and its hyperparameters for next iteration.

A lot remains to be done!

1. From the whole universe of algorithms, which ones to preselect?
2. For each algorithm, which components to configure?
3. How to warmstart algorithms?
4. How to describe the state of an algorithm?
5. How to characterize the instances and how to measure similarity?
6. How to generate diverse training instances?
7. How to assign rewards/loss/total cost to individual actions?
8. …

Thank you for your attention