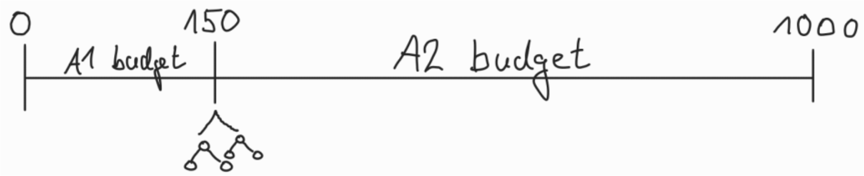
Towards Dynamic Switching in Per-Run Algorithm Selection

2nd Jul, 2025

RWTH AACHEN UNIVERSITY

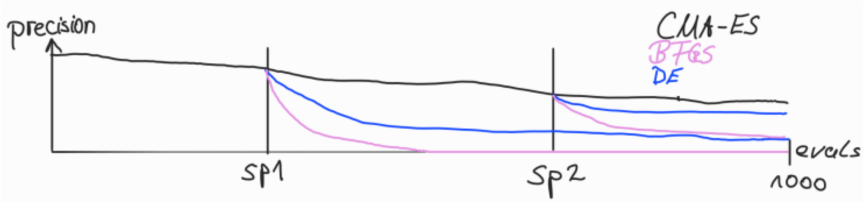
Outline

Why is Dynamic Switching Necessary?



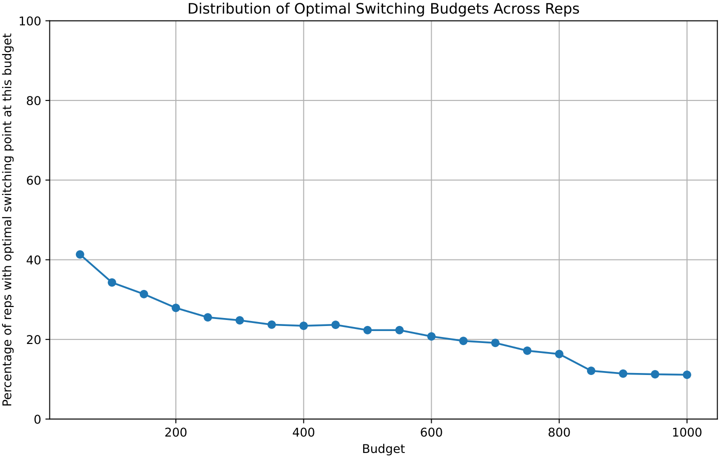
- Approach from Kostovska et al. 2022:
  1. After 150 evaluations, calculate ELA features from CMA-ES samples
  2. Random Forest regression models predict target precisions of the six $\mathcal{A}2$ algorithms
  3. Choose $\mathcal{A}2$ algorithm with lowest predicted precision
  4. Warm-start second algorithm
- $\mathcal{A}1$ budget is static!

Why is Dynamic Switching Necessary?



- Switching point 1 is optimal here

Why is Dynamic Switching Necessary?



Run-Specific Switching

- For each run, we try to find the switching points defined as above
- Selection process consists of two parts:

Switching Decision

- Let $s_1, \ldots, s_n$ be the considered switching points
- At switching point s_i, we predict the precision of the best algorithm at $s_i, s_{i+1}, \ldots, s_n$
- We switch if the predicted precision of the current switching point is the lowest

Algorithm Selection

- Predict the precisions of all six algorithms, switch to the algorithm with the lowest precision

Run-Specific Switching: Our setup

- Switching points: $8, 16, \ldots, 96, 100, 150, 200, \ldots, 1000$
- Machine learning input:
  - ELA features from CMA-ES samples
  - CMA-ES internal state during the last iteration
- Model training on first 5 instances of all BBOB functions, 20 runs each
- Evaluation on instances 6 and 7, 20 runs each
- Metric:

$$\frac{m_{SBS} - m_{selector}}{m_{SBS} - m_{VBS}}$$

Run-Specific Switching

Method	Ratio
static_B64	0.23748752678163765
selector_precision	0.07530683290316334
static_B16	0.035233300427877576
static_B8	-0.1955309433539329

Poor Performance

- Switching decision based on performance of best algorithm
  $\Rightarrow$ Switch too early
- ELA features do not capture enough run-specific information

Function-Specific Switching

Let $s_1, \ldots, s_n$ be the switching points, $a_1, \ldots, a_n$ be the algorithm selectors

1. Record the performances of $a_1, \ldots, a_n$ on first five instances
2. If a_i performs best on function f, define all s_l as the optimal switching point for all runs on that function for all $l \geq i$
3. Define binary switching models for each switching point
4. Train them on first five instances
5. Evaluation on instances 6 and 7

$\Rightarrow$ Switching decision is now function-specific and takes algorithm selector performance into account

Functions-Specific Switching

- Permutation test yields statistical significance over static selectors

Method	Ratio
selector_precision	0.33141463581869085
static_B64	0.23748758528445274
static_B80	0.2134653222051306
static_B96	0.20568772072759586
static_B56	0.20530246383816414
static_B48	0.20287766072778102
static_B350	0.16607201907197186
static_B72	0.1651166732074286
static_B250	0.1489128031965008
static_B150	0.1429869764045132
static_B400	0.13732275545744546
static_B300	0.13489707596408923
static_B100	0.13267790201574783
static_B550	0.1306764266537199
static_B600	0.11960515851248832
static_B450	0.11064745243999607
static_B200	0.10885630015605903
static_B88	0.09900772238554492
static_B650	0.09131779965438094
static_B500	0.0740486560703987
static_B16	0.03523337444839498
static_B700	0.034873681131498836
static_B32	-0.0003746523767353328
static_B40	-0.028241146351575108
static_B24	-0.08757819232985038
static_B750	-0.11635782198921293
static_B8	-0.1955308516283178
static_B800	-0.27190776390865
static_B850	-0.6489669848720225
static_B900	-0.9744155989214369
static_B950	-2.2033023379022225
static_B1000	-7.075054376573443

Conclusion

- Dynamic Switching leads to improvements over static switching
- Run-Specific Switching does not work
  - ELA features do not seem to capture run-specific information
  - Suboptimal performance of algorithm selectors across budgets
- Function-Specific Switching leads to significant improvements