TOPIC 1: ANYTIME BENCHMARKING / ROBUST RANKING IN MULTI- AND MANY OBJECTIVE OPTIMISATION

The Machine Learning and Optimisation (MALEO) Group, Paderborn University

Neha Singh[1] Saurabh Palve[1]

[1]Student, Department of Computer Science, Paderborn University, Germany

23rd Jun, 2025

Introduction

- Modern AI systems must solve complex optimization problems with many competing objectives.
- In such cases, we often ask: “Which solver is better?”
- But: Ranking solvers is hard — performance varies with time, tasks, and noise.
- Enter: Robust, Anytime Benchmarking – techniques that fairly compare solvers even in difficult, uncertain conditions.
- This talk explores how statistical resampling and machine learning offer smarter benchmarks.

Benchmarking: The Backbone of Solver Evaluation

- Benchmarking is the process of evaluating solvers across standardized problems.
- Helps identify strengths, weaknesses, and trade-offs.
- Essential for fair comparison, especially in multi-objective and anytime settings.
- Traditional methods rely on final performance or fixed budgets.
- But in real-world scenarios, performance evolves over time $\rightarrow$ need for anytime benchmarking.

Insights from Recent Research

Liefooghe et al. (2023)

- Introduced a feature-based view of multi/many-objective problems.
- Presented a novel approach using machine learning to analyze and predict the performance of various algorithms on distance-based multi- and many-objective optimization problems.

Fawcett et al. (2023)

- Proposed robust solver rankings using bootstrapping and statistical resampling.
- Focused on fairness and consistency in AI competitions

Rook et al. (2024)

- Extended robust ranking to multi-objective optimisation
- Emphasized:
  - Confidence intervals
  - Bootstrap-based score distributions

Results: Benchmarking based on budget

```
Results for Budget = 5000
```

	Budget	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
0	5000	IBEA	1.0	0.0	0.0	0.0
1	5000	MOEAD	0.0	1.0	0.0	0.0
2	5000	NSGAII	0.0	0.0	1.0	0.0
3	5000	Random	0.0	0.0	0.0	1.0

```
Results for Budget = 10000
```

	Budget	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
4	10000	IBEA	1.0	0.0	0.0	0.0
5	10000	MOEAD	0.0	0.0	1.0	0.0
6	10000	NSGAII	0.0	1.0	0.0	0.0
7	10000	Random	0.0	0.0	0.0	1.0

```
Results for Budget = 30000
```

	Budget	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
8	30000	IBEA	1.0	0.0	0.0	0.0
9	30000	MOEAD	0.0	0.0	0.0	1.0
10	30000	NSGAII	0.0	1.0	0.0	0.0
11	30000	Random	0.0	0.0	1.0	0.0

```
Results for Budget = 50000
```

	Budget	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
12	50000	IBEA	1.0	0.0	0.0	0.0
13	50000	MOEAD	0.0	0.0	0.0	1.0
14	50000	NSGAII	0.0	1.0	0.0	0.0
15	50000	Random	0.0	0.0	1.0	0.0

Budget Benchmarking- 5000, 10000, 15000, 20000

Results: Benchmarking based on features

```
Results for Feature = n_discon_ps, Group = High n_discon_ps
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
28	n_discon_ps	High n_discon_ps	IBEA	1.0	0.0	0.0	0.0
29	n_discon_ps	High n_discon_ps	MOEAD	0.0	0.0	1.0	0.0
30	n_discon_ps	High n_discon_ps	NSGAII	0.0	1.0	0.0	0.0
31	n_discon_ps	High n_discon_ps	Random	0.0	0.0	0.0	1.0

```
Results for Feature = n_discon_ps, Group = Low n_discon_ps
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
32	n_discon_ps	Low n_discon_ps	IBEA	1.0	0.0	0.000	0.000
33	n_discon_ps	Low n_discon_ps	MOEAD	0.0	0.0	0.957	0.043
34	n_discon_ps	Low n_discon_ps	NSGAII	0.0	1.0	0.000	0.000
35	n_discon_ps	Low n_discon_ps	Random	0.0	0.0	0.043	0.957

Benchmarking for "n_discon"

Results: Benchmarking based on features

Results for Feature = n_local_fronts, Group = High n_local_fronts

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
36	n_local_fronts	High n_local_fronts	IBEA	1.0	0.0	0.0	0.0
37	n_local_fronts	High n_local_fronts	MOEAD	0.0	0.0	0.0	1.0
38	n_local_fronts	High n_local_fronts	NSGAII	0.0	1.0	0.0	0.0
39	n_local_fronts	High n_local_fronts	Random	0.0	0.0	1.0	0.0

Results for Feature = n_local_fronts, Group = Low n_local_fronts

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
40	n_local_fronts	Low n_local_fronts	IBEA	1.0	0.0	0.0	0.0
41	n_local_fronts	Low n_local_fronts	MOEAD	0.0	0.0	1.0	0.0
42	n_local_fronts	Low n_local_fronts	NSGAII	0.0	1.0	0.0	0.0
43	n_local_fronts	Low n_local_fronts	Random	0.0	0.0	0.0	1.0

Benchmarking for "n_local_fronts"

Results: Benchmarking based on features

```
Results for Feature = n_obj, Group = High n_obj
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
8	n_obj	High n_obj	IBEA	1.0	0.0	0.000	0.000
9	n_obj	High n_obj	MOEAD	0.0	0.0	0.072	0.928
10	n_obj	High n_obj	NSGAII	0.0	1.0	0.000	0.000
11	n_obj	High n_obj	Random	0.0	0.0	0.928	0.072

```
Results for Feature = n_obj, Group = Low n_obj
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
12	n_obj	Low n_obj	IBEA	1.0	0.0	0.0	0.0
13	n_obj	Low n_obj	MOEAD	0.0	0.0	1.0	0.0
14	n_obj	Low n_obj	NSGAII	0.0	1.0	0.0	0.0
15	n_obj	Low n_obj	Random	0.0	0.0	0.0	1.0

Benchmarking for "n_obj"

Results: Benchmarking based on features

```
Results for Feature = n_resist_regions, Group = High n_resist_regions
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
44	n_resist_regions	High n_resist_regions	IBEA	1.0	0.0	0.0	0.0
45	n_resist_regions	High n_resist_regions	MOEAD	0.0	0.0	1.0	0.0
46	n_resist_regions	High n_resist_regions	NSGAII	0.0	1.0	0.0	0.0
47	n_resist_regions	High n_resist_regions	Random	0.0	0.0	0.0	1.0

```
Results for Feature = n_resist_regions, Group = Low n_resist_regions
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
48	n_resist_regions	Low n_resist_regions	IBEA	1.0	0.0	0.000	0.000
49	n_resist_regions	Low n_resist_regions	MOEAD	0.0	0.0	0.182	0.818
50	n_resist_regions	Low n_resist_regions	NSGAII	0.0	1.0	0.000	0.000
51	n_resist_regions	Low n_resist_regions	Random	0.0	0.0	0.818	0.182

Benchmarking for "n_resist_regions"

Results: Benchmarking based on features

```
Results for Feature = n_var, Group = High n_var
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
0	n_var	High n_var	IBEA	1.0	0.0	0.0	0.0
1	n_var	High n_var	MOEAD	0.0	0.0	1.0	0.0
2	n_var	High n_var	NSGAII	0.0	1.0	0.0	0.0
3	n_var	High n_var	Random	0.0	0.0	0.0	1.0

```
Results for Feature = n_var, Group = Low n_var
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
4	n_var	Low n_var	IBEA	1.0	0.0	0.0	0.0
5	n_var	Low n_var	MOEAD	0.0	0.0	0.0	1.0
6	n_var	Low n_var	NSGAII	0.0	1.0	0.0	0.0
7	n_var	Low n_var	Random	0.0	0.0	1.0	0.0

Benchmarking for "n_var"

Results: Benchmarking based on features

```
Results for Feature = nonident_ps, Group = High nonident_ps
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
16	nonident_ps	High nonident_ps	IBEA	1.0	0.0	0.0	0.0
17	nonident_ps	High nonident_ps	MOEAD	0.0	0.0	1.0	0.0
18	nonident_ps	High nonident_ps	NSGAII	0.0	1.0	0.0	0.0
19	nonident_ps	High nonident_ps	Random	0.0	0.0	0.0	1.0

```
Results for Feature = nonident_ps, Group = Low nonident_ps
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
20	nonident_ps	Low nonident_ps	IBEA	1.0	0.0	0.0	0.0
21	nonident_ps	Low nonident_ps	MOEAD	0.0	0.0	1.0	0.0
22	nonident_ps	Low nonident_ps	NSGAII	0.0	1.0	0.0	0.0
23	nonident_ps	Low nonident_ps	Random	0.0	0.0	0.0	1.0

```
Results for Feature = var_density, Group = Low var_density
```

	Feature	Group	Algorithm	Rank 1 Frequency	Rank 2 Frequency	Rank 3 Frequency	Rank 4 Frequency
24	var_density	Low var_density	IBEA	1.0	0.0	0.0	0.0
25	var_density	Low var_density	MOEAD	0.0	0.0	1.0	0.0
26	var_density	Low var_density	NSGAII	0.0	1.0	0.0	0.0
27	var_density	Low var_density	Random	0.0	0.0	0.0	1.0

Benchmarking for "non_ident_ps" and Low "var_density"

Take-home message

- IBEA consistently ranked 1st and NSGA-II 2nd across all settings. At higher budgets (e.g., 30,000–50,000), Random outperformed MOEA/D, showing strong late-stage performance.
- When analyzing by problem features, IBEA remained dominant, followed by NSGA-II, while Random outperformed MOEA/D on problems with many objectives and low n-resist regions.

Thank you! Questions?