technische universität dortmund

SFB 876 Providing Information by Resource-Constrained Data Analysis

Project A3

Methods for Efficient Resource Utilization in Machine Learning Algorithms

Prof. Dr. Peter Marwedel, Prof. Dr. Jörg Rahnenführer

Problem

Development of Methods for Algorithm Selection and Configuration under Resource Constraints

- Single evaluation of algorithms expensive (long runtimes)
- Large number of competing candidates of algorithms
- Each algorithm has specific hyperparameters to be tuned
- Exhaustive search not possible due to resource constraints
  - e.g. in embedded systems (heterogeneous architectures)

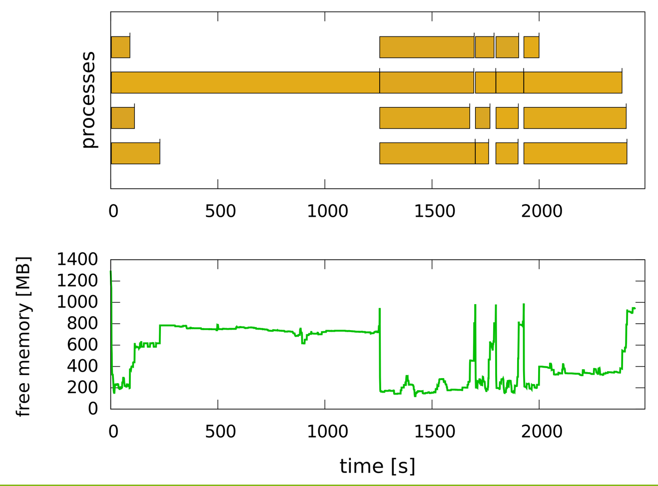
Expensive Black-Box Optimization

- Typical application: hyperparameter tuning for machine learning methods

Goal: Optimize the resource utilization and improve prediction quality

- Sustainable (open source) software required for reproducibility

Example: Resource Utilization Profile during Optimization



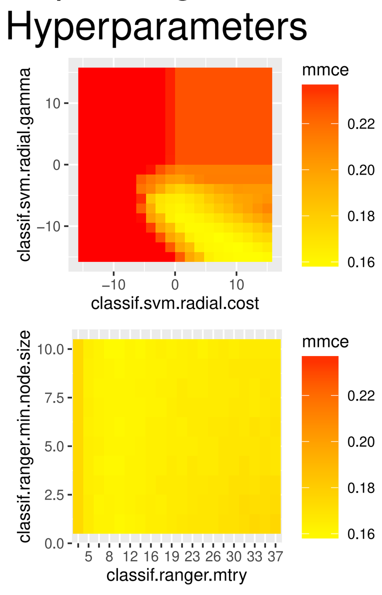
Hyperparameter optimization: SVM on 4 CPUs

Top: CPU usage, blank space between blocks indicates CPU idling due to heterogeneous runtimes of the SVM

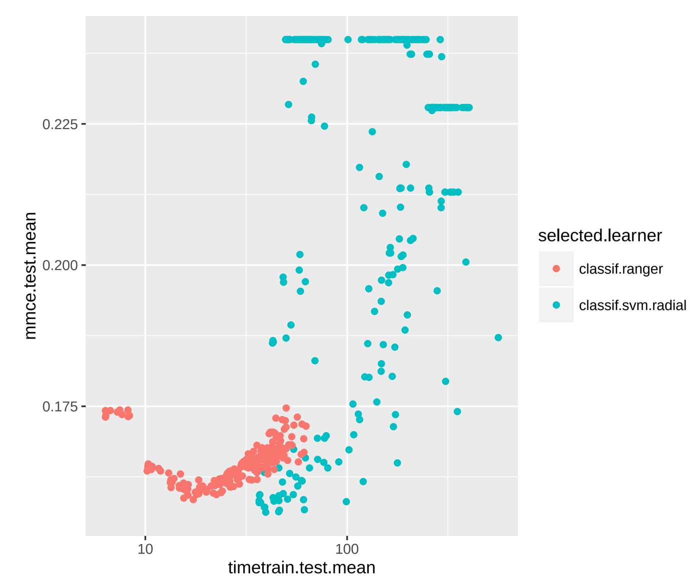
Bottom: Free main memory

Complexity of Hyperparameter Space: Prediction Quality versus Resource Utilization

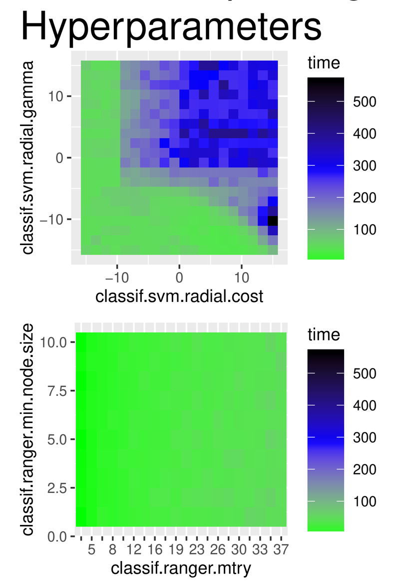
Misclassification Rate Depending on Hyperparameters



Misclassification Rate vs. Runtime



Runtime Depending on Hyperparameters



Complex search space

- SVM: good prediction quality only for specific hyperparameters
- RF: stable prediction quality

But: Best prediction quality reached by SVM with tuned hyperparameters

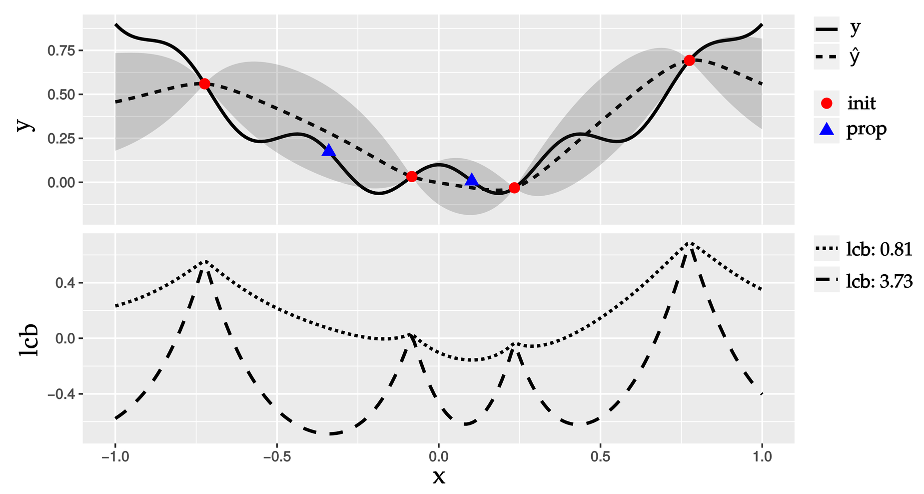
Heterogeneous runtimes

- SVM: runtime depends heavily on hyperparameters
- RF: homogeneous runtimes

Methodology

Model-Based Optimisation (MBO)

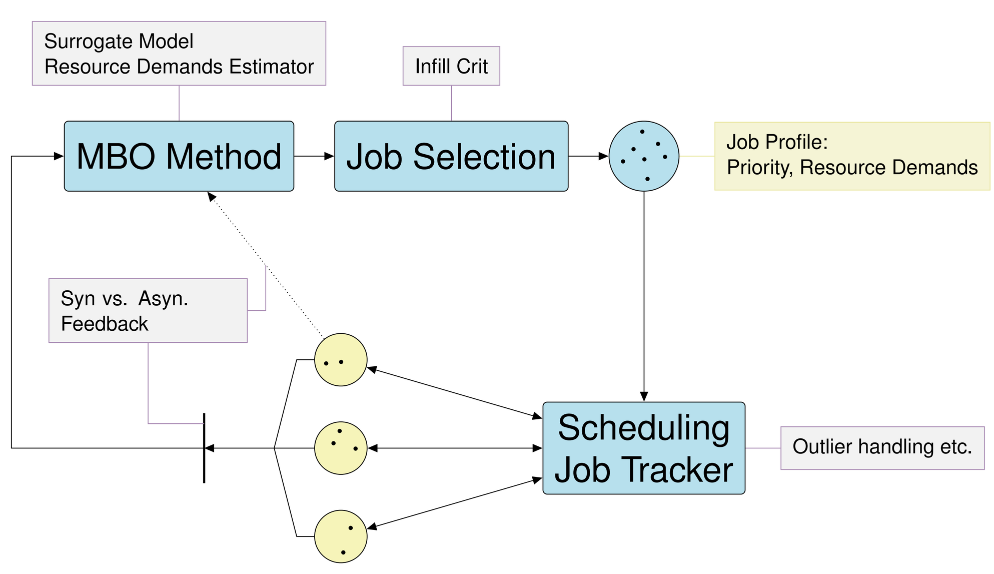
- Regression model as a surrogate to approximate relationship between x and $f(x)$
- Predictions of model help to move quickly to regions with promising prediction quality
- Infill criterion balances exploitation and exploration: $LCB(\mathbf{x}, \lambda) = \hat{\mu}(\mathbf{x}) - \lambda \cdot \hat{s}(\mathbf{x})$



Infill criterion with different weighting of uncertainty (different values for λ) leads to multiple independent proposals

Resource-Aware Model-Based Optimisation (RAMBO)

- RAMBO extends MBO to parallel systems with optimized resource utilization for heterogeneous runtimes
  - Resource Demands Estimator: Additional regression model to predict runtime
  - Job Selection: runtime estimates + execution priority $\rightarrow$ interaction between scheduling and infill
  - Scheduling: Controlled job selection to converge faster to optimum

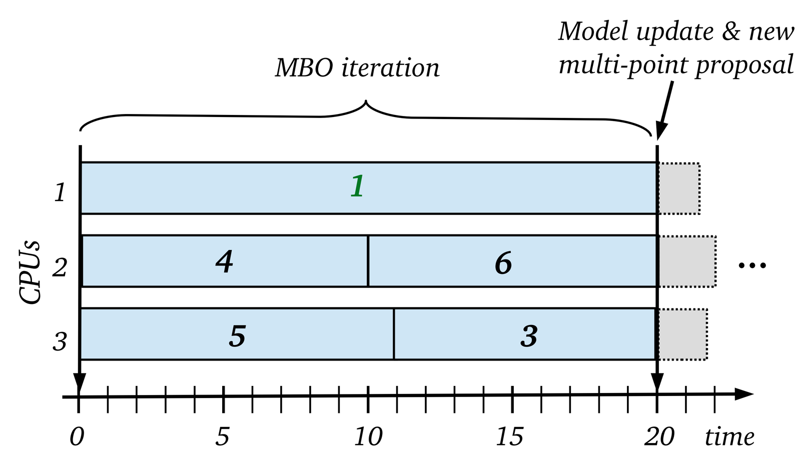


Scheduling Strategies for Efficient Resource Utilization

- Inputs for scheduling in each MBO iteration
  - p_j: Priority of jobs determined by infill criterion
  - $\hat{t}_j$: Estimated runtime of jobs
- Resource-aware strategy
  - Determine job with highest priority and set MBO iteration time bound to its runtime
  - Map most promising job to one CPU exclusively
  - Solve knapsack problem to maximize sum of priorities and map remaining jobs on CPUs
- Job priority refinement: based on hierarchical clustering to prefer jobs scattered across the search space

Set of job proposals

j	p_j	$\hat{t}_j$
1	10	20
2	8	~~25~~
3	7	9
4	5	10
5	3	11
6	2	10
7	1	2
8	6	~~21~~
9	4	~~24~~



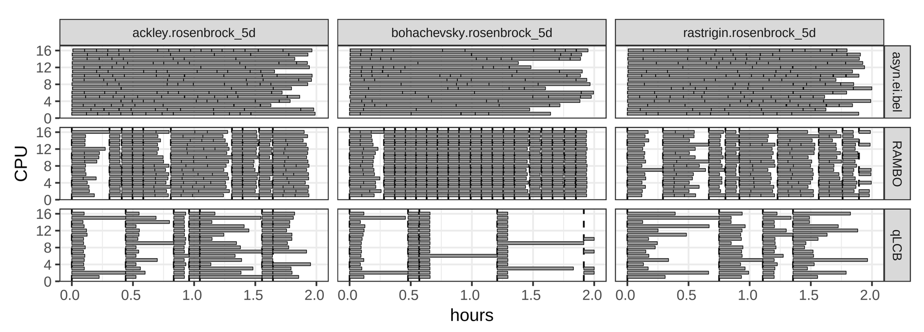
Results

RAMBO Uses Resources Efficiently

- Asynchronous strategy
  - `asyn.ei.bel`: Worker determines next evaluation immediately. Estimated results of running jobs are fed into the surrogate model
- Synchronous strategy with multipoint proposals
  - `RAMBO`: Central process creates set of scheduled jobs for all workers
  - `qLCB`: One proposal for each worker after evaluations are finished on all workers

Result:

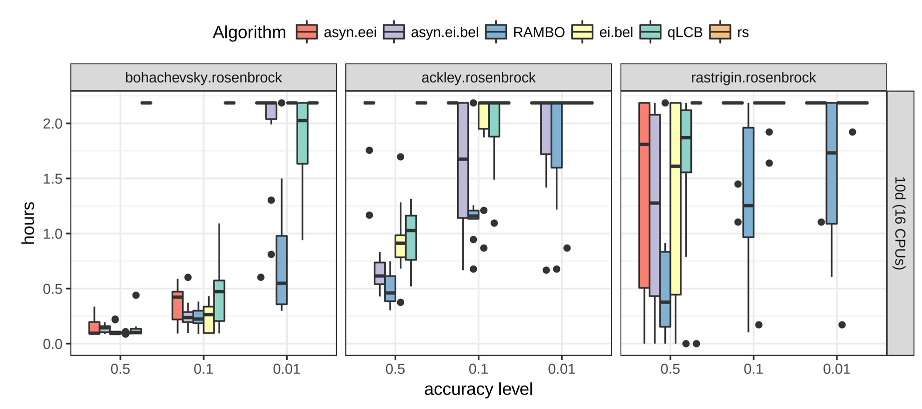
- `RAMBO` reduces idle time in comparison to `qLCB`
- `RAMBO` has similar resource utilization compared to `asyn.ei.bel`



Synthetic benchmarks on 16 CPUs

RAMBO Approaches Optimum Faster

- `RAMBO`: beneficial due to controlled exploration
- `asyn.eei`: Overhead of calculating the EEI via MC Simulation decreases performance
- `asyn.ei.bel`: Estimated results deteriorate performance
- `qLCB` + `ei.bel`: Less evaluations due to idling



Software:

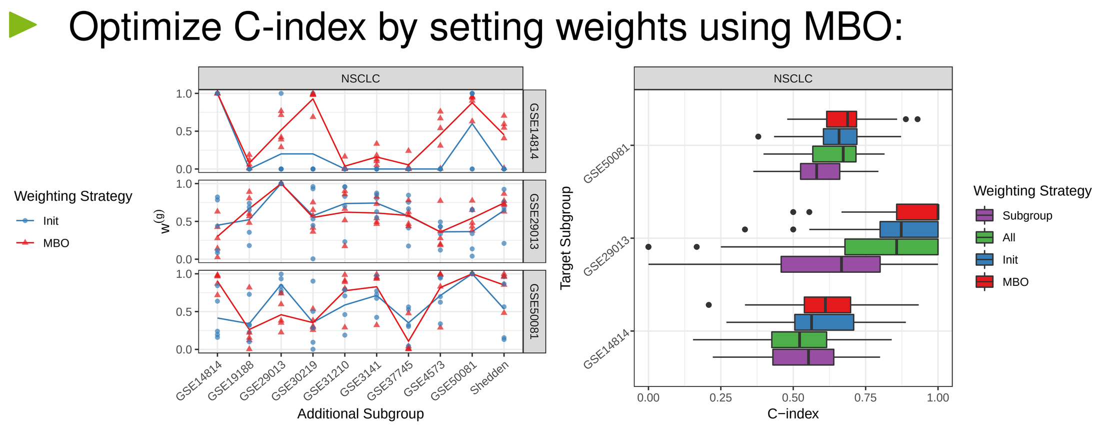
self-contained, well-documented, open-source packages, assuring reproducibility:

- mlr, mlrMBO, mlrMBO + RAMBO, mlrHyperopt, traceR

MBO Improves Prediction of Survival Times

Heterogeneous patient cohort (e.g. different clinical centers):

- Prediction of survival curves for subgroup
- Problem:
  - Subgroup model: small sample size
  - Pooled model: high heterogeneity between subgroups
- Solution:
  - Group-specific weights to select subgroups that improve prediction quality
  - Weighted version of partial log-likelihood
- Optimize C-index by setting weights using MBO:



Results:

- Subgroup 1: Few similar subgroups with large weight
- Subgroup 2+3: Most subgroups with medium weight
- MBO improves prediction quality by selecting suitable weights as hyperparameters

CS 12 computer science 12

fakultät statistik