HyperSpace

Library: https://github.com/yngtodd/hyperspace
Jumana's: https://github.com/jdakka/hyperspace-RCT
Use-Case Doc

Background

- Machine learning methods require the user to define *a priori* a set of parameters that maximize the usefulness of the learning (hyperparameters) [6].
- Hyperparameters are set manually, via rules-of-thumb or by testing a set of hyperparameters on a predefined grid [6].
    - Why this is important:
        - Impracticality of hand-tuning hyperparameters when there are many hyperparameters
        - Reproducibility
- Hyperparameter optimizations with different approaches [1]:
    - Grid search "*parameter sweeping*" [7]
    - **Random search** [8]
    - **Sequential Model-Based Optimization** (SMBO) [4]
    - Gradient-based optimization [8]
    - Evolutionary optimization [2]

Definitions

Hyperparameter configuration: a snapshot of a combination of hyperparameters that need to be evaluated at time_x.

Machine learning methods rely on minimizing an objective function

- **Objective function**: A function that measures how well a machine learning method is able to predict the expected outcome [2]
  - Common method of finding minimum is **stochastic gradient descent (SGD)**
  - Objective functions are computationally expensive to evaluate [1]

Validation protocol: evaluates the machine learning method in terms of predicting the expected outcome, typically measured as **test accuracy** (cross-validation).

Bayesian SMBO

- Class of optimization algorithms used when the optimization function like SGD is expensive to evaluate [1].
- SMBO evaluates the performance of the model using an optimization function that has more “intelligence” than SGD.
- SMBO uses Bayesian optimization to model the conditional probability $p(y|ƛ)$
  - y (test accuracy)
  - ƛ (hyperparameter configuration) [3]

- SMBO requirements:
  - Machine learning method of interest
  - Train/test datasets
  - Validation protocol (cross-validation)
  - Define upper and lower bounds for each hyperparameter
  - Define the “intelligent” optimization function → Gaussian process with guided sampling

Parallel Bayesian SMBO

- Same as previous slide:
  - Define the SMBO requirements:
    - Model/network of interest
    - Train/test datasets
    - Validation protocol (cross-validation)
    - Define upper and lower bounds for each hyperparameter
    - Define the optimization function (Gaussian process)

- Parallel version creates **combinations of hyperparameters** using overlapping boundaries between hyperparameters

- In HyperSpace, these combinations are referred to as **hyperspaces** [1]
  - Parallel Bayesian SMBO run multiple optimizations in **parallel** where each optimization explores a hyperspace

SMBO Algorithms

- **Spearmint** (Bayesian) [2]
  - parallelism across the global search space
  - Single optimization
  - Runs one Gaussian process with Monte Carlo estimates

- **HyperSpace** (Bayesian) [1]
  - parallelism across “hyperparameter search space” i.e., hyperspace [1]
  - Runs bag-of-tasks of optimizations where each task runs the Gaussian process that explores a hyperspace

- **HyperSpace** has demonstrated faster convergence than **Spearmint** by exploiting multiple hyperparameter configurations in parallel [1]

Pseudo-code HyperSpace (Step 1)

Algorithm 1 HyperSpace

Input: Hyperparameter intervals, subinterval length α, overlap length ϕ
Output: Optimization over 2^H hyperspaces (H = number of hyperparameters)

```
for each hyperparameter interval do
    if hyperparameter interval > α then
        Partition interval into two equal subintervals with overlap φ
    else
        No split
    end if
end for
Combine all possible subintervals to form hyperspaces
for each hyperspace in PARALLEL do
    Run optimization
end for
```

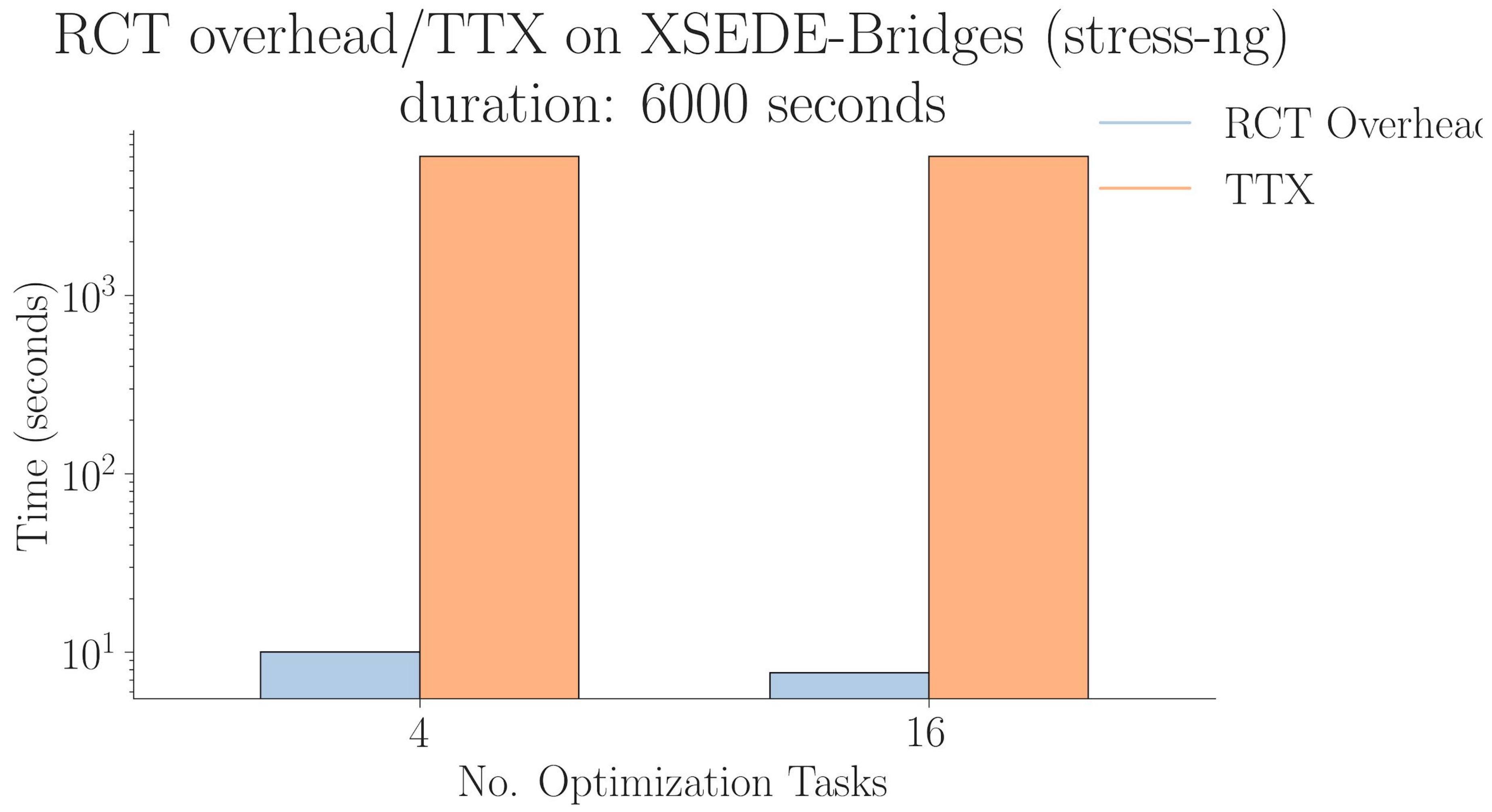
Execution of HyperSpace - Step 2

- HyperSpaces use Scikit-Optimize

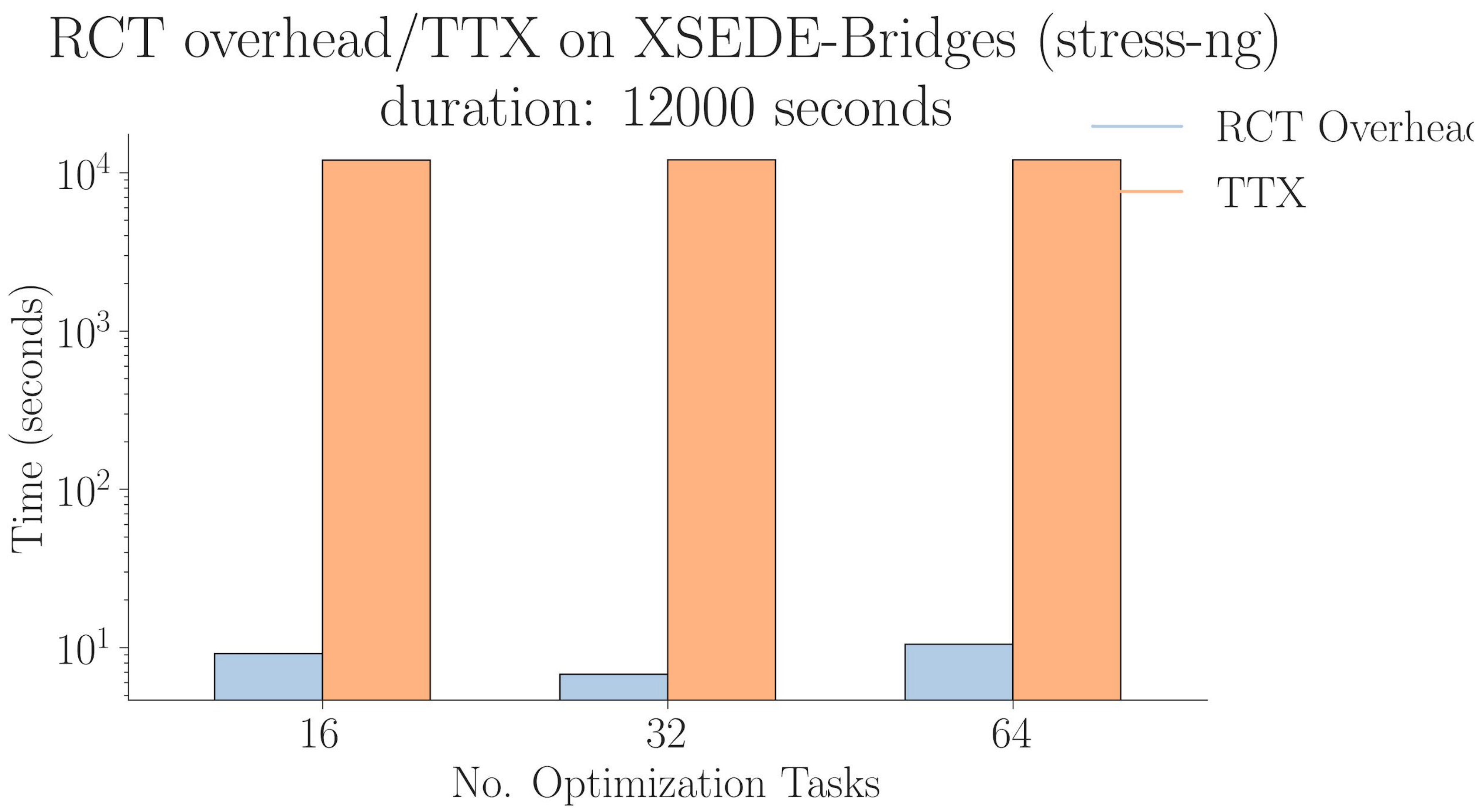
- *From previous slide:* bag-of-tasks are executed with mpi4py
    - Limitations of mpi4py:
        - Each rank contains the same number of resources but hyperspaces have non-uniform resource requirements

- Number of tasks depends on the number of hyperparameters for the model:
    - HyperSpaces = 2^H where H is the number of hyperparameters
    - Avg. num of hyperparameters ~ 7-8 but depending on model can go up to 12
    - Each optimization runs for N-iterations, where N is ~100
    - We are looking at supporting 2^8 concurrent tasks, but upper-bound can be as high as 2^12
    - Avg. duration of each optimization: between ~3 hours to several days

- Each task requires 1 or more nodes, depending on the size of the input data



RCT overhead/TTX on XSEDE-Bridges (stress-ng)
duration: 6000 seconds
RCT Overhead
TTX
Time (seconds)
10^3
10^2
10^1
4
16
No. Optimization Tasks



RCT overhead/TTX on XSEDE-Bridges (stress-ng)
duration: 12000 seconds
RCT Overhead
TTX
Time (seconds)
10^1
10^2
10^3
10^4
16
32
64
No. Optimization Tasks

References

[1] Young et al., “HyperSpace: Distributed Bayesian Hyperparameter Optimization”

[2] Bergstra et al., “Algorithms for Hyperparameter Optimization” NIPS 2011

[3] Li et al., “Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization”, JMLR 2018

[4] Hutter et al., “Sequential Model-Based Optimization for General Algorithm Configuration”, ACM 2011

[5] Feurer et al., “Using Meta-Learning to Initialize Bayesian Optimization of Hyperparameters”, ECAI Workshop on Meta-Learning and Algorithm Selection, 2014

[6] Claesen et al., “Hyperparameter Search in Machine Learning”

[7] Chin-Wei Hsu et al. “A Practical Guide to Support Vector Classification”, Technical Report, NTU, 2010

[8] Ziyu et al. “Bayesian Optimization in a Billion Dimensions via Random Embeddings”, JAIR, 2018