

Optimizing Machine Learning Workloads with Experiment Databases

Behrouz Derakhshan
behrouz.derakhshan@dfki.de



Database Systems and Information
Management Group
TU Berlin

<https://www.dima.tu-berlin.de/>



Intelligent Analytics for Massive Data
German Research Center for Artificial
Intelligence

<https://www.dfki.de/>

- Online data science platforms allow
 - Write solutions (typically via Jupyter Notebooks)
 - Share solutions
 - Learn from existing solutions
- Examples
 - Google Collaboratory
 - Kaggle
 - Online Education platforms

- Users incorporate existing solutions
- Repetition of (similar) operations

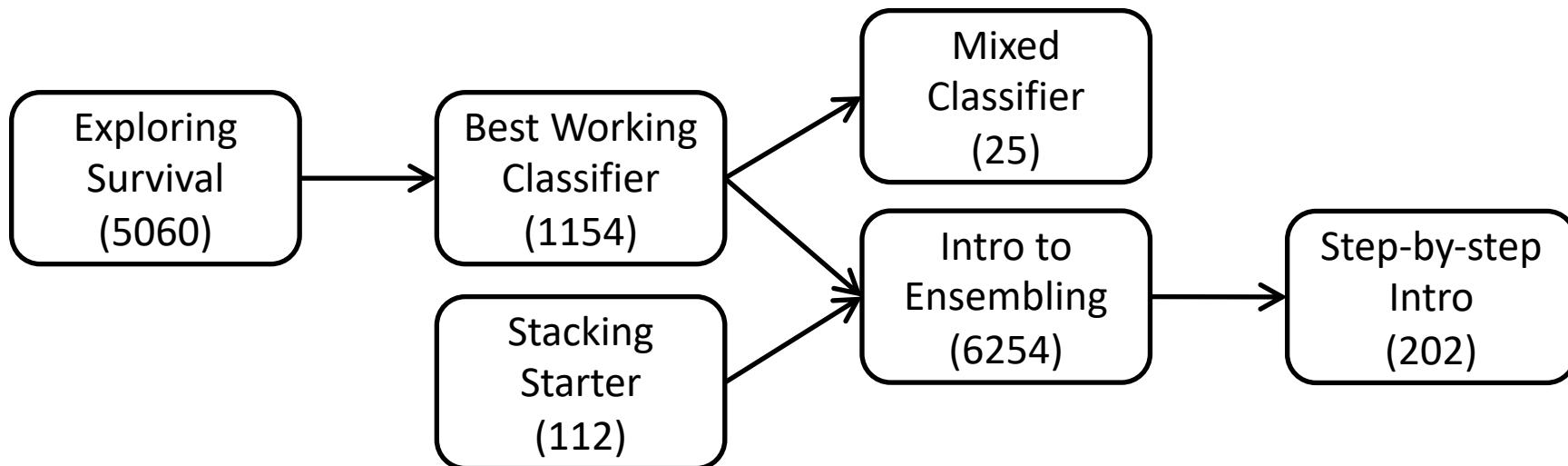
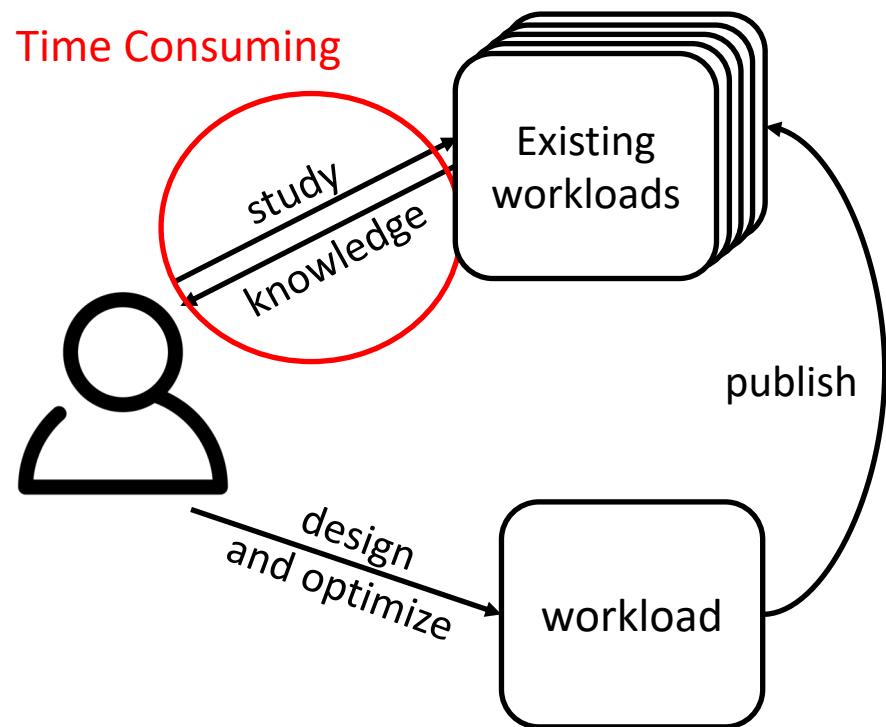


Figure 1: Kaggle's notebook relation¹

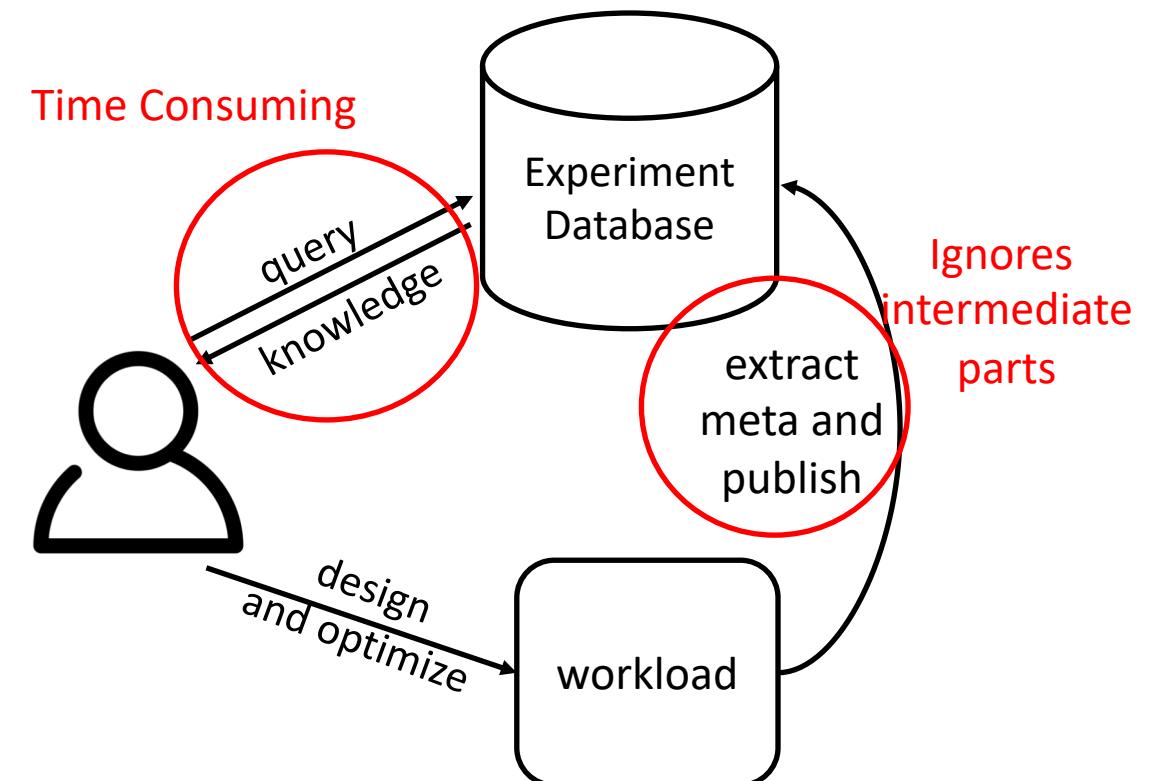
1. <https://www.kaggle.com/c/titanic>

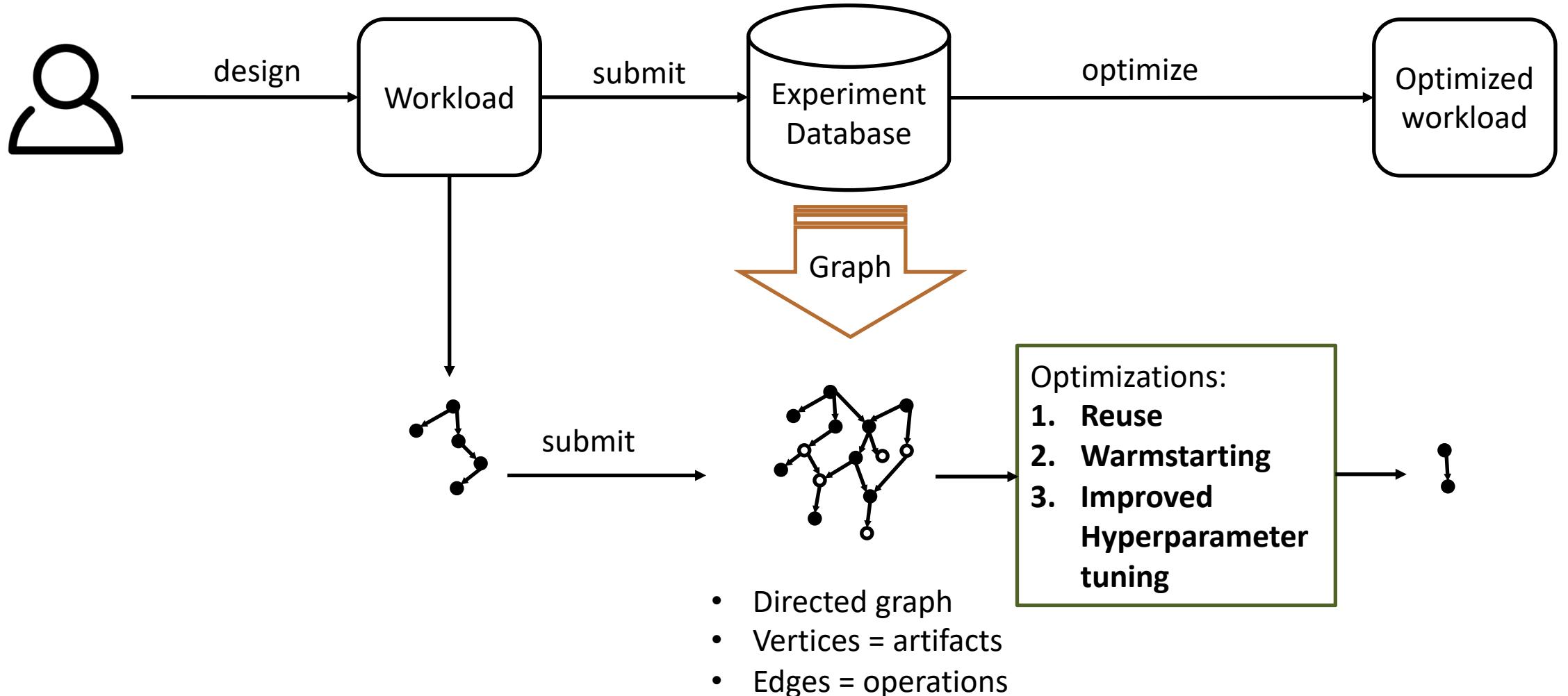
Efficient **design** and **execution** of machine learning (and data science) workloads in collaborative environments

Learn by reading



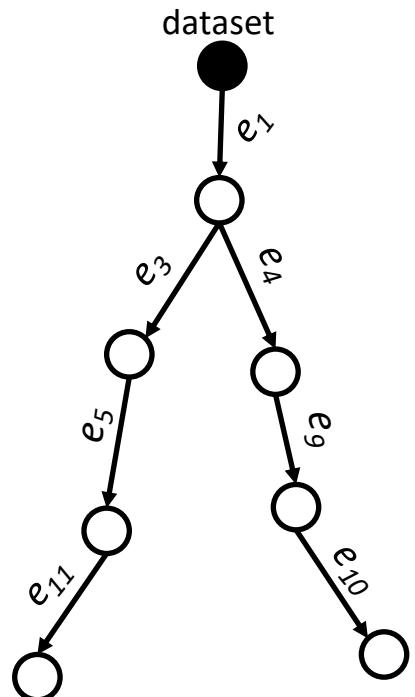
Through Experiment Databases (e.g., OpenML)



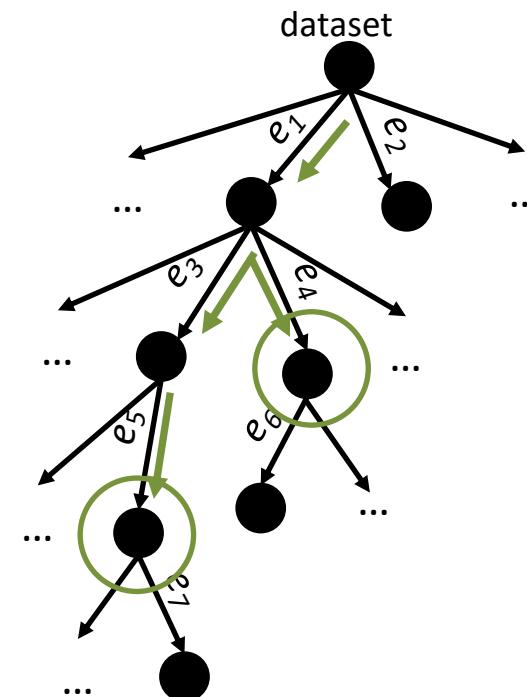


- Edges are uniquely identified
- For new workloads, traverse the Experiment Graph

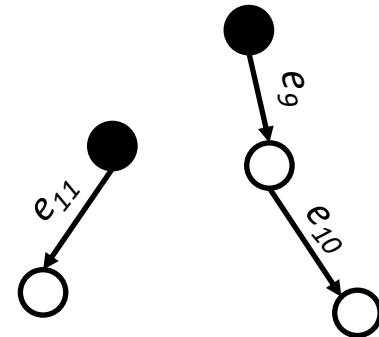
Workload



Experiment Graph



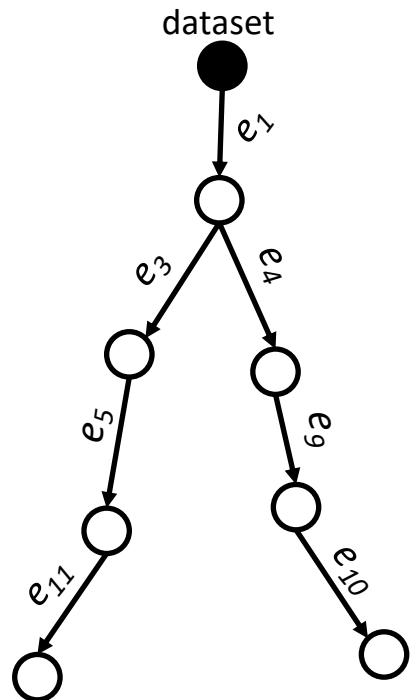
Optimized workload



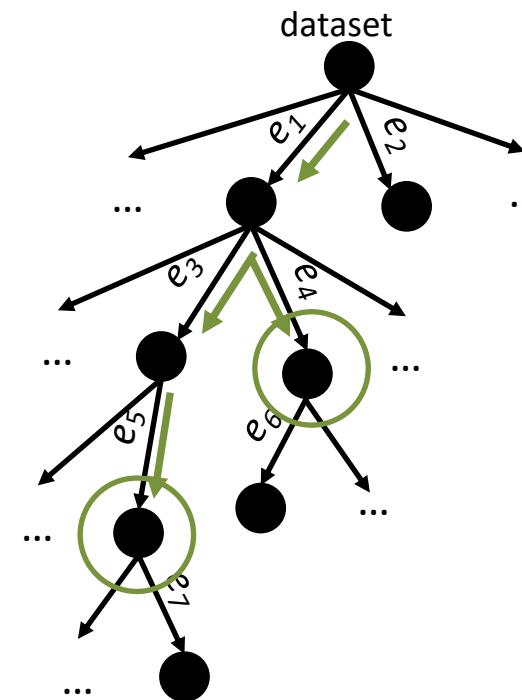
Improves total execution time

- Warmstart a training operations with the best model from the same model group
- **Model group:** set of all models (of same type) trained on one artifact

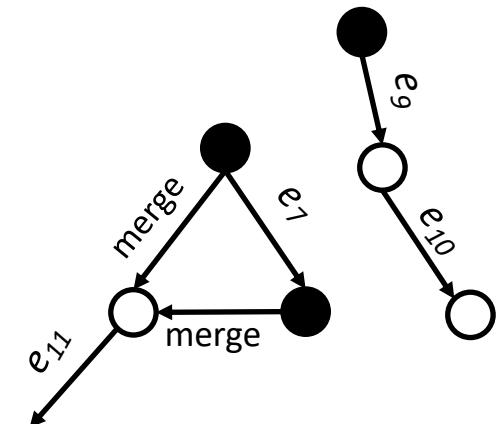
Workload



Experiment Graph



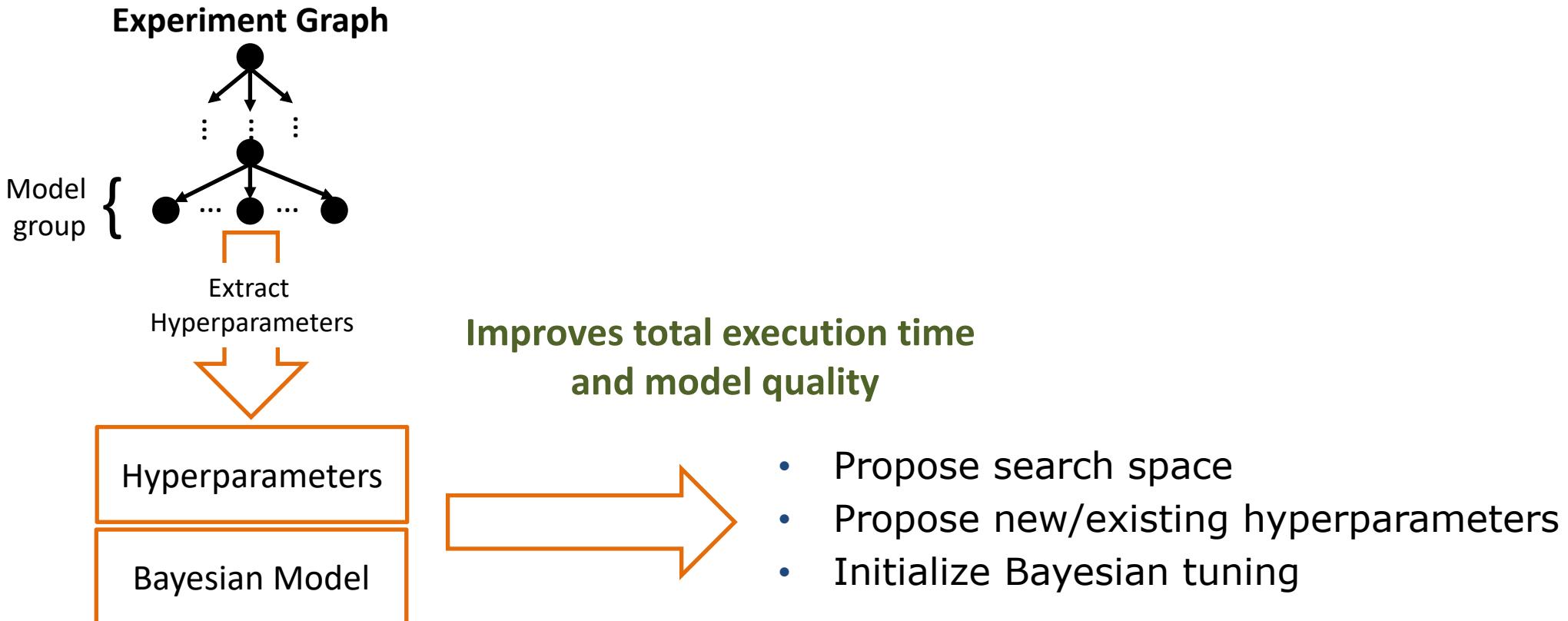
Optimized workload



Improves total execution time
and *model quality*

e_7 and e_{11} are both model training operations

- Challenges in Hyperparameter tuning
 - Search space definition (grid, random, and Bayesian tuning)
 - Requires several initial runs until promising models are trained (Bayesian tuning)

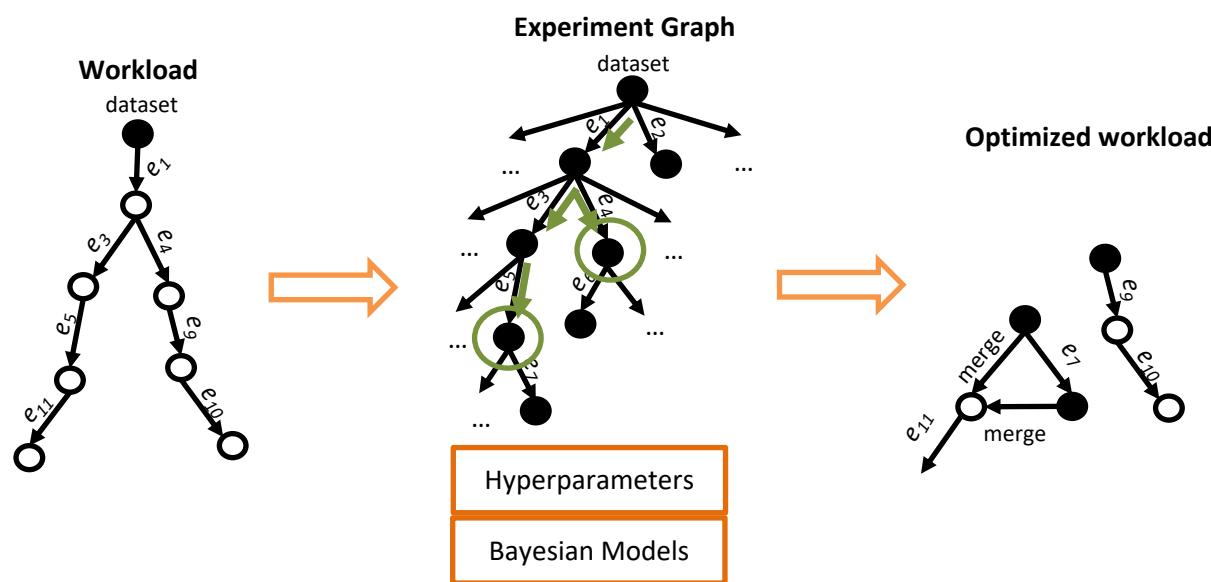
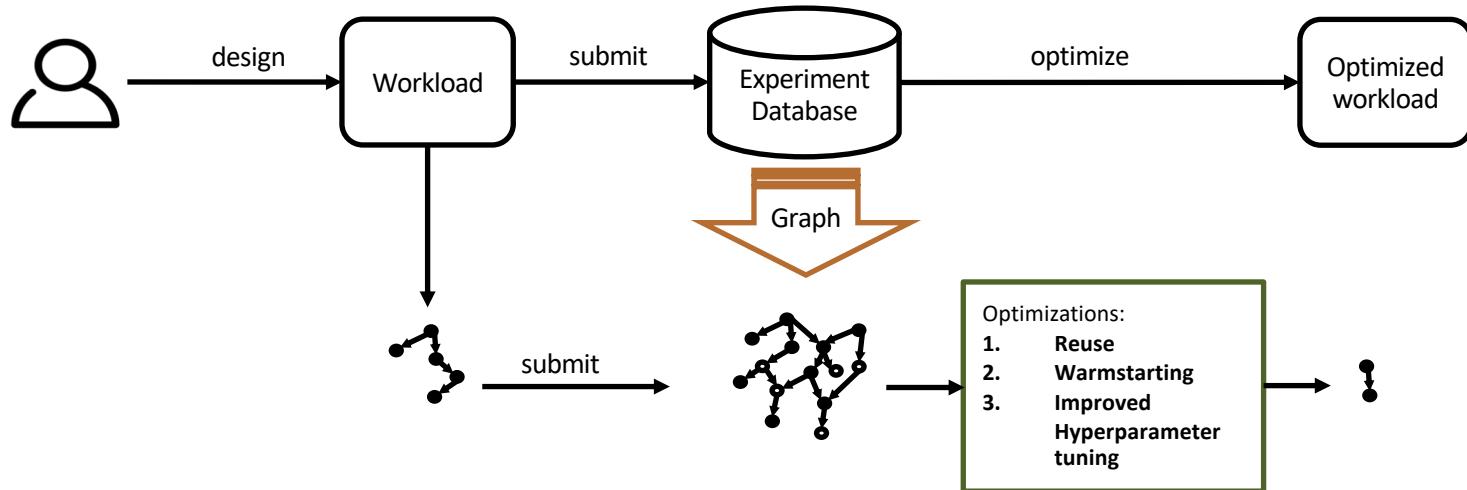


Implementation

- Promising initial experiment results
- Prototype of Experiment Graph with Pandas dataframes and scikit-learn
- Experiments on Kaggle use case
 - Home Credit Default Risk (In progress)
 - Another competition

Questions

- Single node with a large memory or a cluster
- Are these experiments adequate
 - OpenML
 - 2 Kaggle competitions



- **Contributions:**
 - A Framework for optimization of machine learning and data science workloads
 - *Materialization strategy*
 - Reuse
 - Warmstarting
 - Improved Hyperparameter tuning
- Planned submission:
 - VLDB 2020, May or July Submission
- Publications since last Offsite:
 - Continuous Deployment paper (offsite 2017) accepted for publication in EDBT 2019

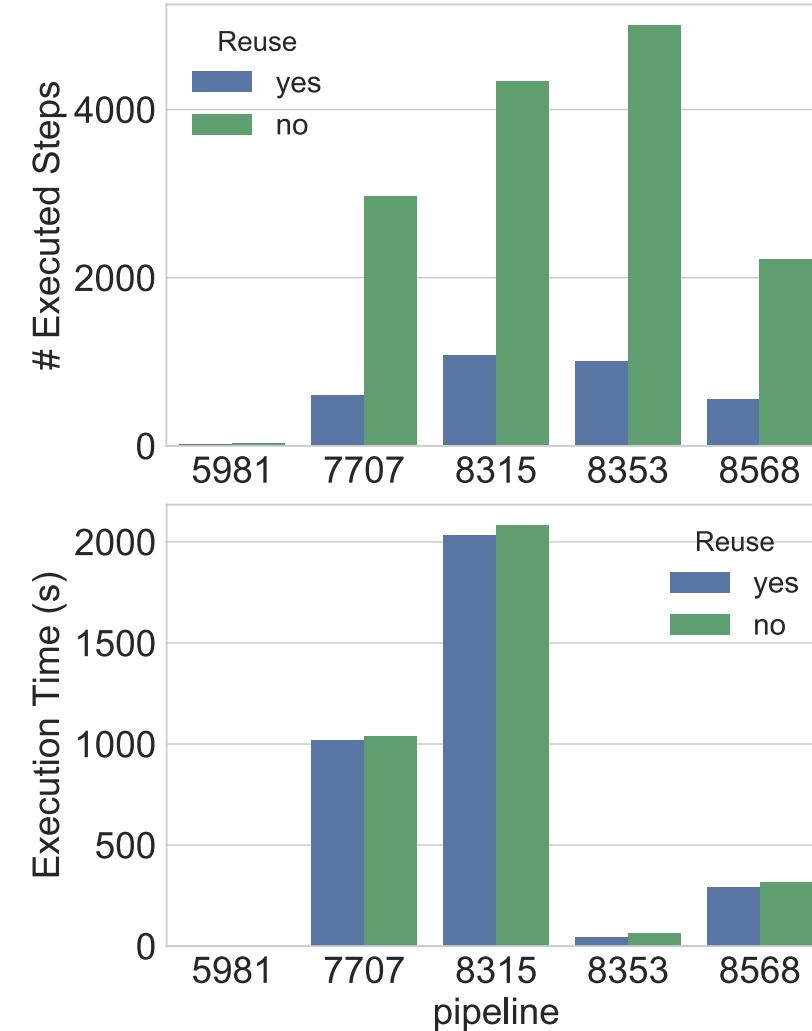
1. <https://data.world>
2. <https://colab.research.google.com>
3. <https://www.kaggle.com/>
4. <https://www.coursera.org/>
5. Vanschoren, Joaquin, et al. "OpenML: networked science in machine learning." *ACM SIGKDD Explorations Newsletter* 15.2 (2014): 49-60.
6. Vartak, Manasi, et al. "Model DB: a system for machine learning model management." *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*. ACM, 2016.
7. Hutter, Frank, Holger H. Hoos, and Kevin Leyton-Brown. "Sequential model-based optimization for general algorithm configuration." *International Conference on Learning and Intelligent Optimization*. Springer, Berlin, Heidelberg, 2011.
8. Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms In Advances in neural information processing systems, pages 2951–2959, 2012.
9. Bergstra, James, and Yoshua Bengio. "Random search for hyper-parameter optimization." *Journal of Machine Learning Research* 13.Feb (2012): 281-305.
10. Vartak, Manasi, et al. "MISTIQUE: A System to Store and Query Model Intermediates for Model Diagnosis." *Proceedings of the 2018 International Conference on Management of Data*. ACM, 2018.
11. Souvik Bhattacherjee, Amit Chavan, Silu Huang, Amol Deshpande, and Aditya Parameswaran. Principles of dataset versioning: Exploring the recreation/storage tradeo. *Proceedings of the VLDB Endowment*, 8(12):1346–1357, 2015.

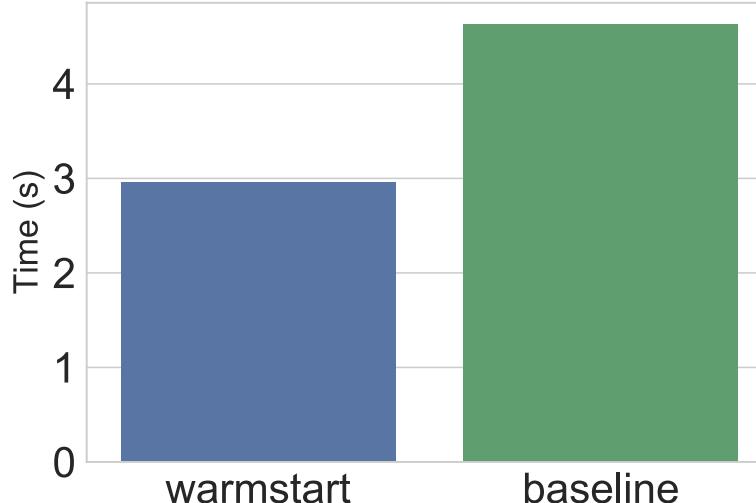
- Optimizations are limited to a task
- A **task** defines the goal of the machine learning or data science work
- Examples of tasks and workloads:
 - A competition on the [Kaggle](#),
 - An assignment in [Coursera](#)
- Each task is associated with one connected component in the experiment graph

OpenML Pipelines

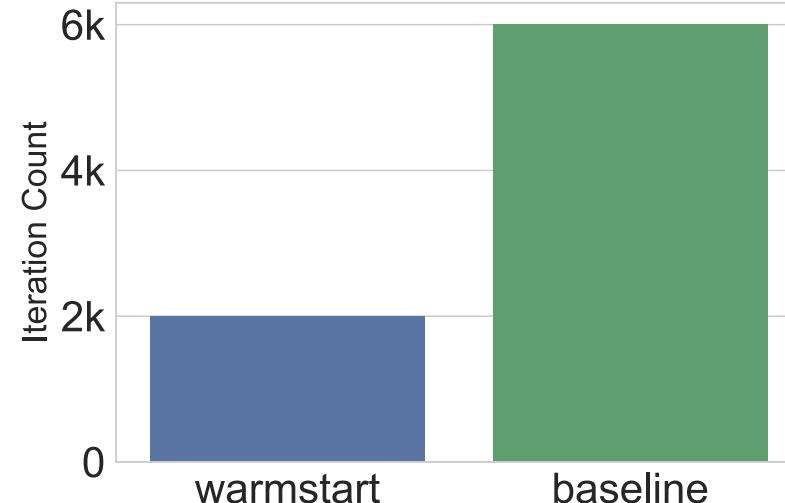
id	operations	#exec
5981	Impute → Scale → logistic regression	11
7707	Impute → Onehot encode → Scale → Variance Threshold → SVM	594
8315	Impute → Onehot encode → Variance Threshold → Random Forest	1084
8353	Impute → Onehot encode → Variance Threshold → SVM Forest	1000
8568	Impute → Onehot encode → Variance Threshold → Random Forest	555

Reuse Optimization

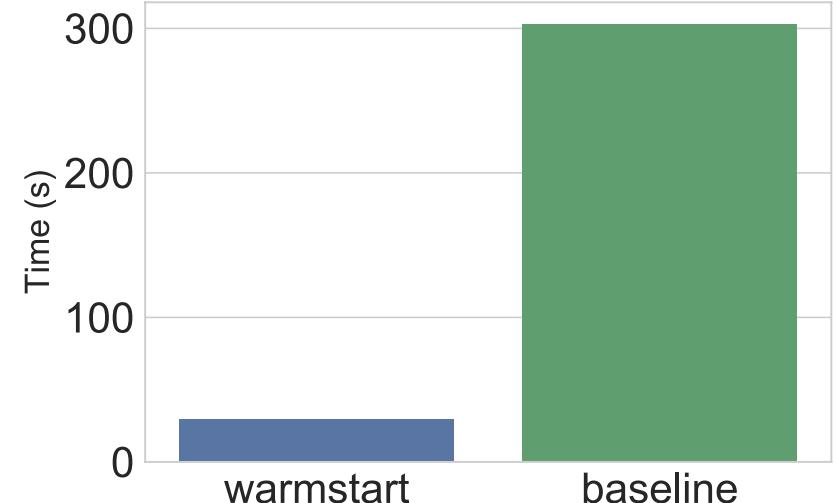




(a) Total training time (#5981)



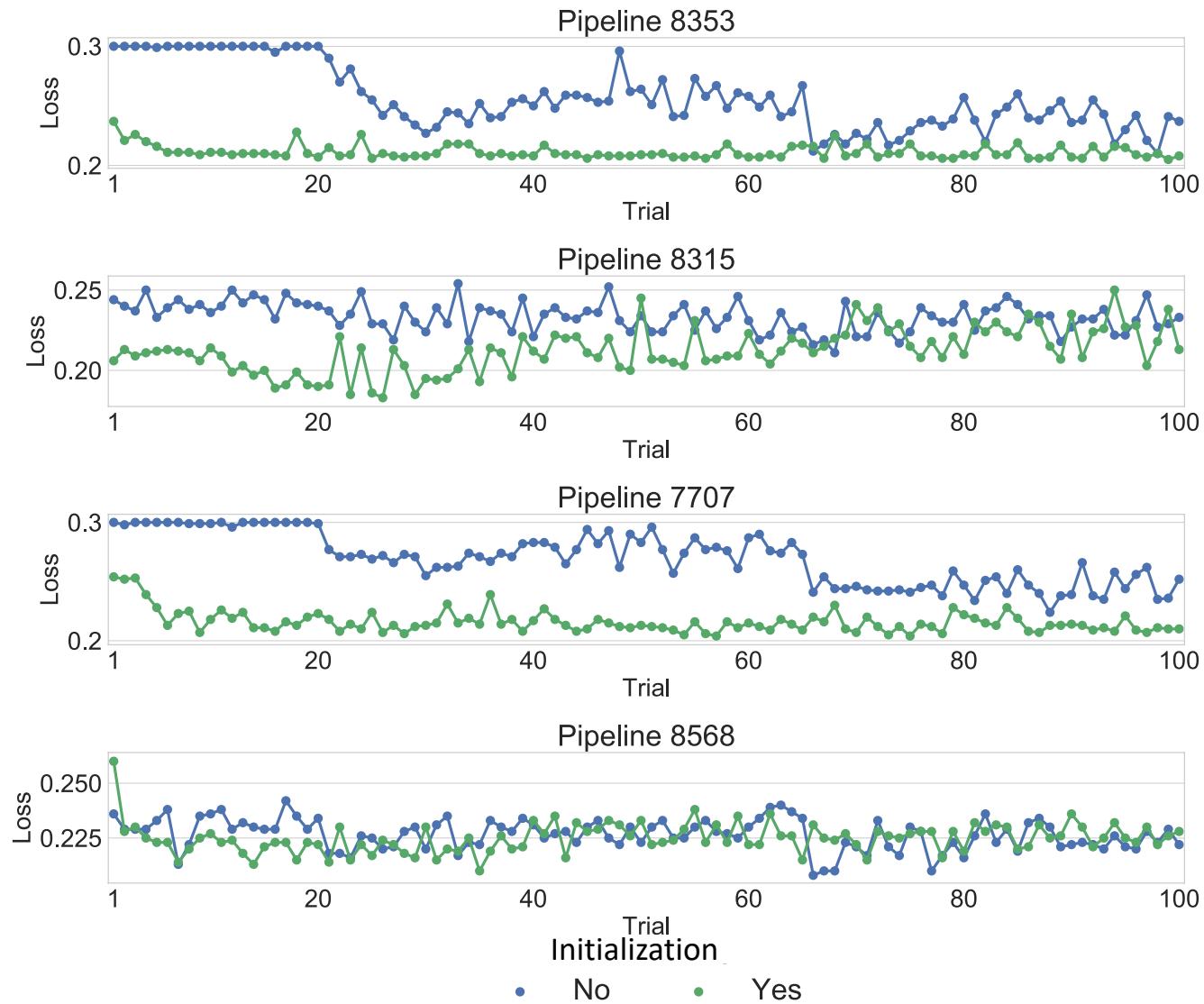
(a) Total iteration count (#5981)



(a) Total training time (#8568)

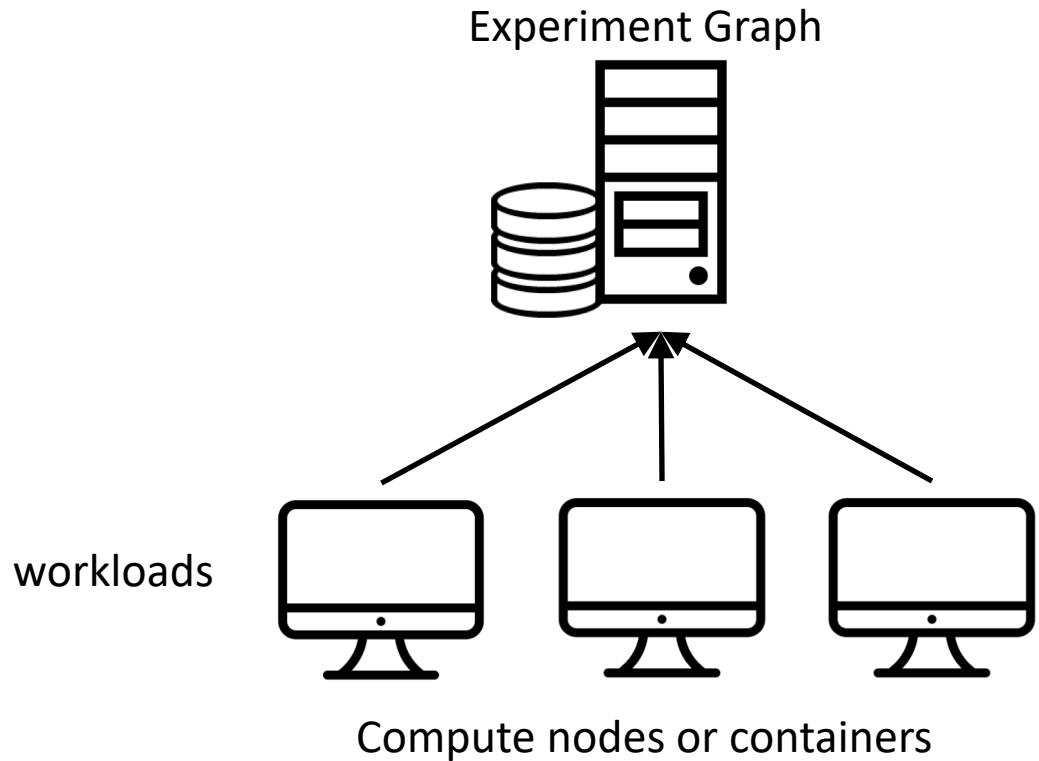
Effect of the warm-starting optimization on the total training time and iteration count. In (a) and (b) we train 11 logistic regression models and in (c) we train 555 random forest model from the configurations that exist in the experiment data.

- Initialize Bayesian search with values from the experiment database

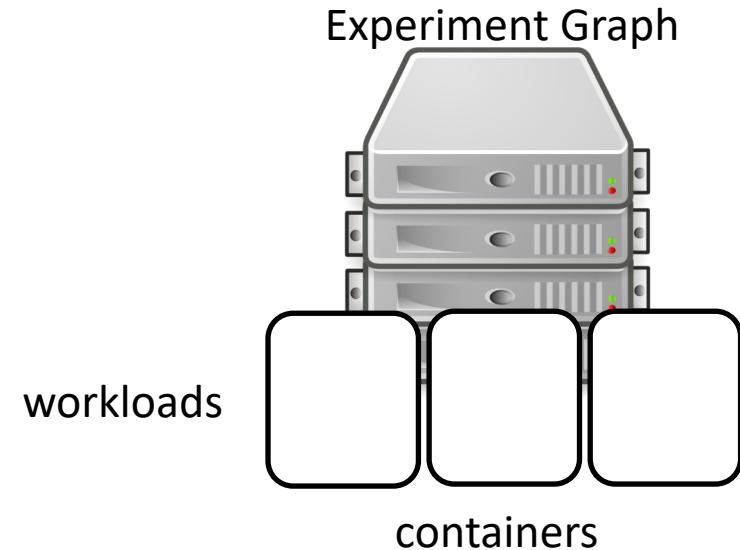


- The figure shows the loss value gained for a Bayesian tuning process with budget of 100 (with/without initialization)

Remote Experiment Graph



Single Node



Example script

```

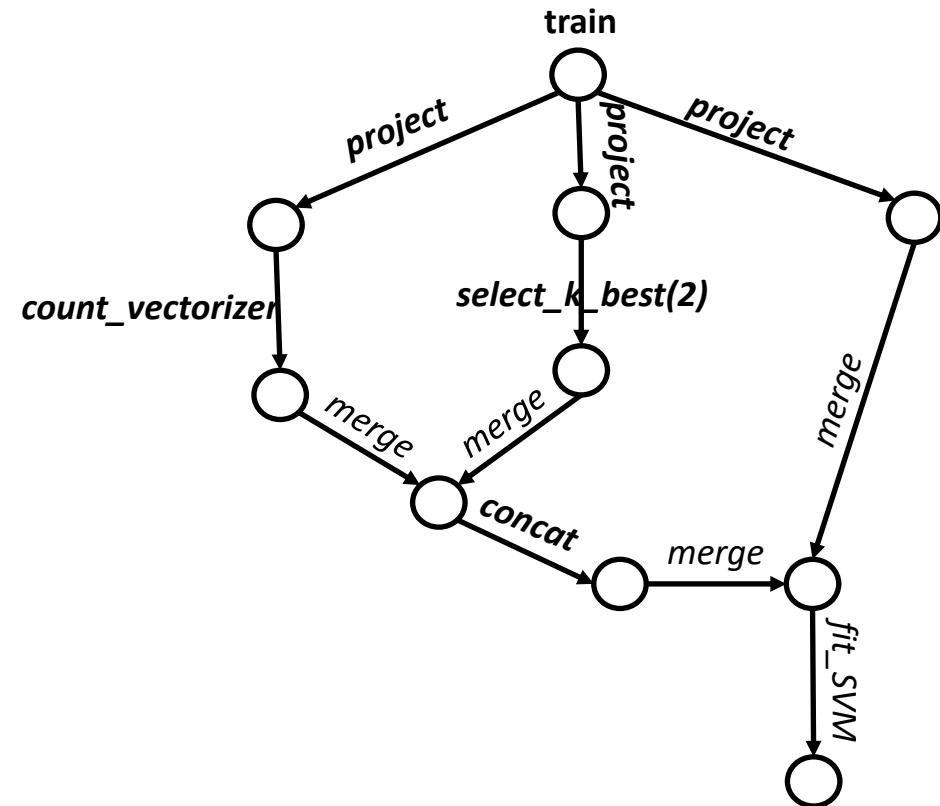
import numpy as np
import pandas as pd

from sklearn import svm
from sklearn.feature_selection import SelectKBest
from sklearn.feature_extraction.text import CountVectorizer

train = pd.read_csv('../input/train.csv')
print train.columns # [ad_description,ts,u_id,price,y]
vectorizer = CountVectorizer()
count_vectorized = vectorizer.fit_transform(train['ad_description'])
selector = SelectKBest(k=2)
top_features = selector.fit_transform(train[['ts','u_id','price']],
                                      train['y'])
X = pd.concat([count_vectorized,top_features], axis = 1)
model = svm.SVC()
model.fit(X, train['y'])

```

Graph Representation



merge operation:

- Logical operation with a run time of 0 that combines two nodes

- Problem statement: given a size budget, T (representing the total storage capacity), materialize the artifacts that result in the smallest weighted recreation cost of the experiment graph $\mathcal{G}(\mathcal{V}, \mathcal{E})$, where the weighted recreation cost is:

$$w(\mathcal{G}) = \sum_{e \in \{e' \in \mathcal{E} \mid dest(e') \notin \mathcal{MV}\}} e.f \times e.t$$

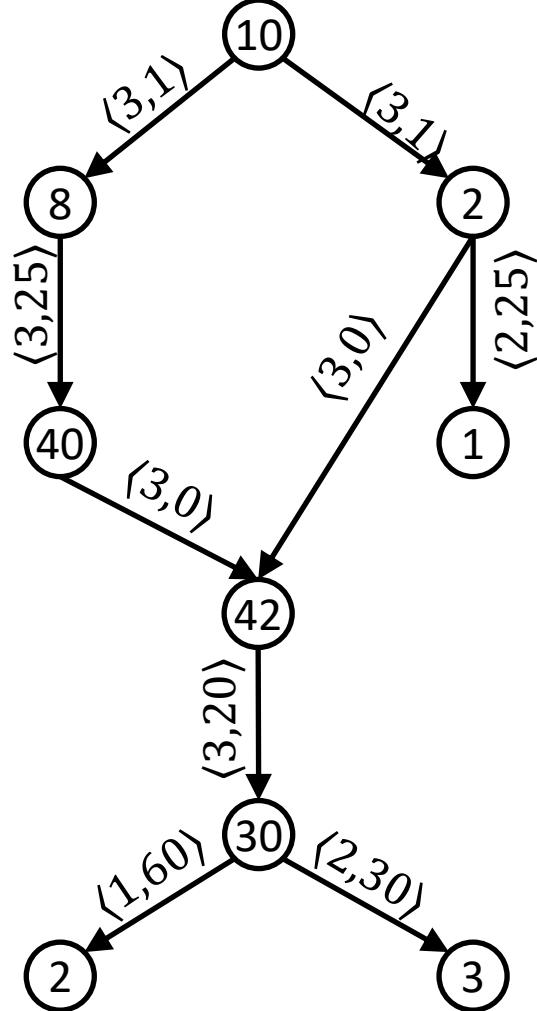
- Where \mathcal{MV} represents the set of materialized artifacts
- Each edge e is labeled with $\langle f, t \rangle$, representing the frequency and the average execution time of the operation.
- Vertices are labeled with $\langle s \rangle$, representing the size of the artifact

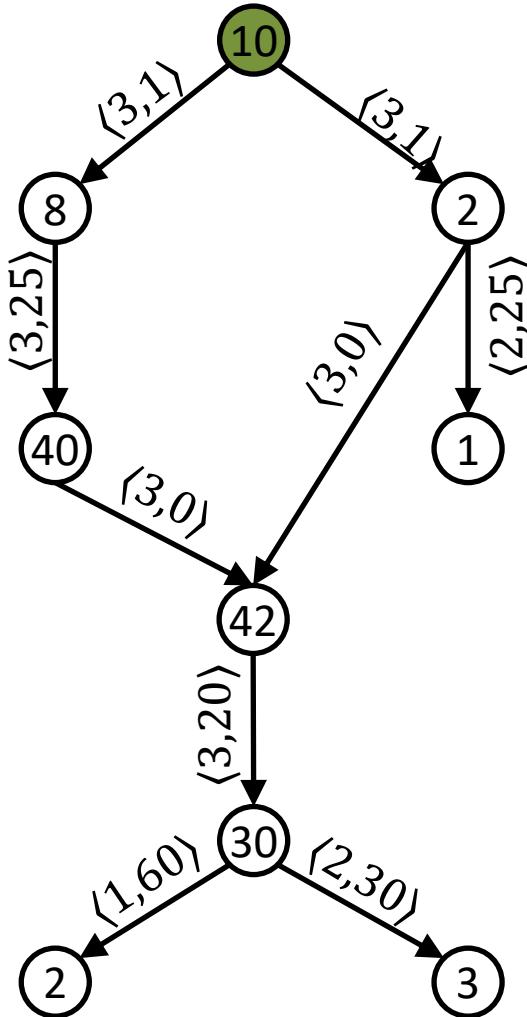
- Materialize the root node
- While storage limit is not reached:
 - Find the vertex v , with maximum value of $\frac{\rho(\mathcal{G}, v)}{v.s}$

$$v^* = \operatorname{argmax}_{v \in \mathcal{V}} \frac{\rho(\mathcal{G}, v)}{v.s}$$

Where,

$$\rho(\mathcal{G}, v) = \alpha(\mathcal{G}, v) \times \sum_{e \in \text{path}(\mathcal{G}, v_0, v)} e.t$$



$T = 10$ $\max \text{ size} = 55$ 

$$T = 10$$

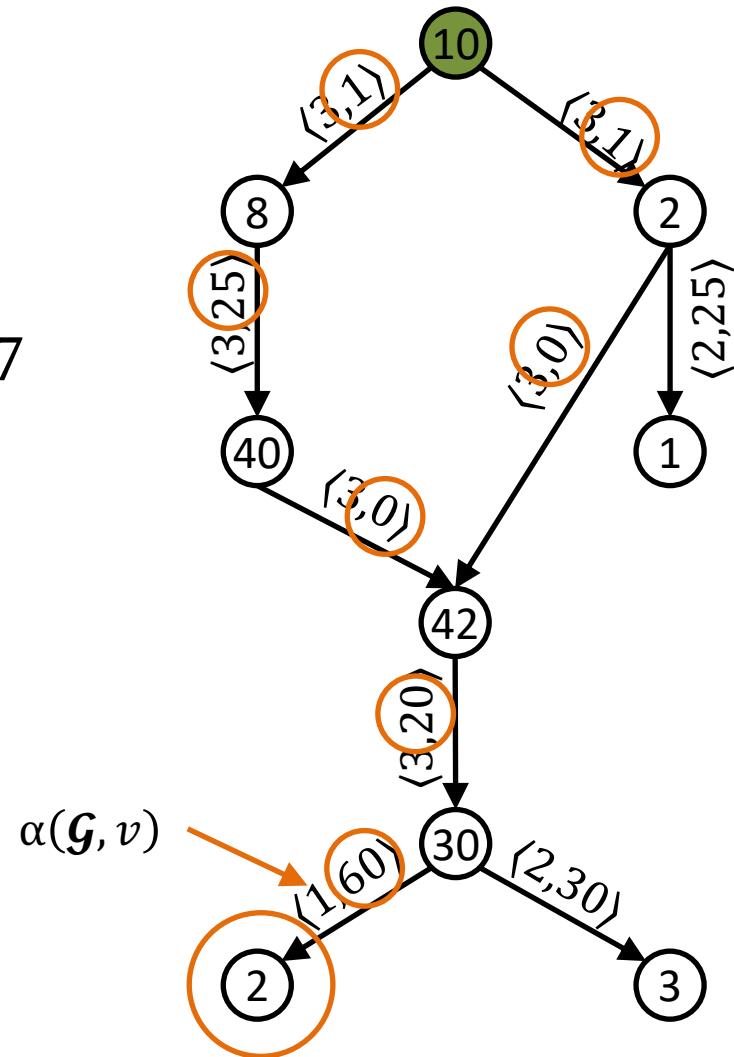
$$\max size = 55$$

$$v.s = 2$$

$$T + v.s < \max size$$

$$\rho(\mathcal{G}, v) = 1 \times (1 + 1 + 25 + 0 + 0 + 20 + 60) = 107$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 53.5$$



$$T = 12$$

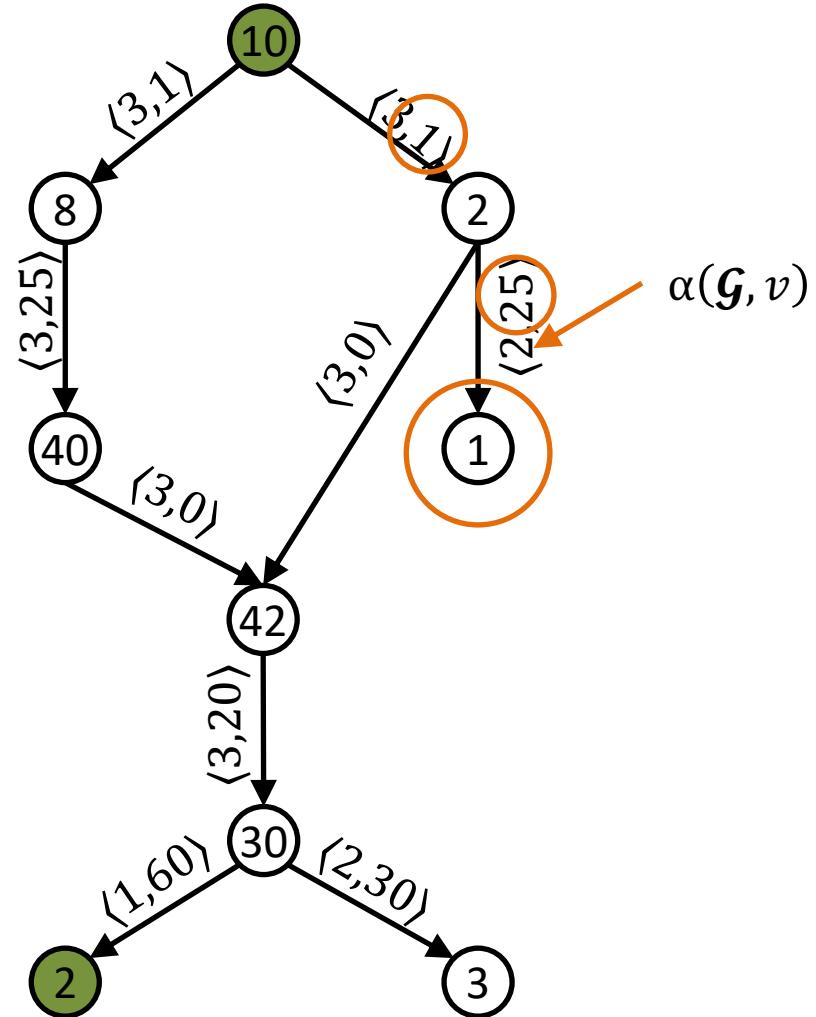
$$\max size = 55$$

$$v.s = 1$$

$$T + v.s < \max size$$

$$\rho(\mathcal{G}, v) = 2 \times (1 + 25) = 52$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 52$$



$$T = 13$$

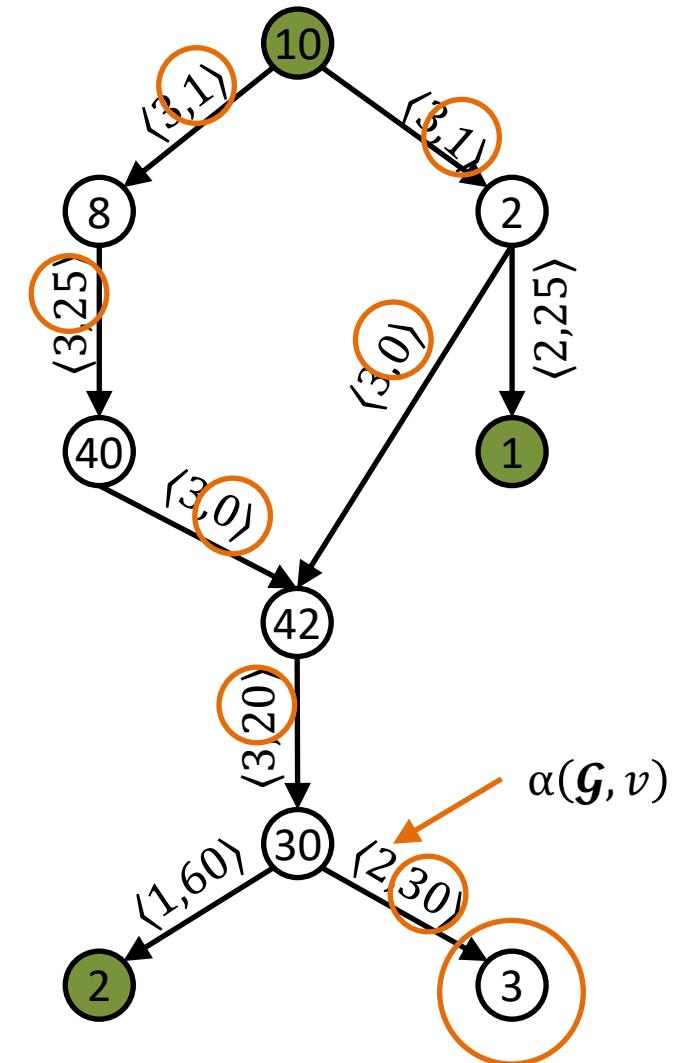
$$\max size = 55$$

$$v.s = 3$$

$$T + v.s < \max size$$

$$\rho(\mathcal{G}, v) = 2 \times (1 + 1 + 25 + 0 + 20 + 30) = 154$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 51.3$$



$$\alpha(\mathcal{G}, v)$$

$$T = 16$$

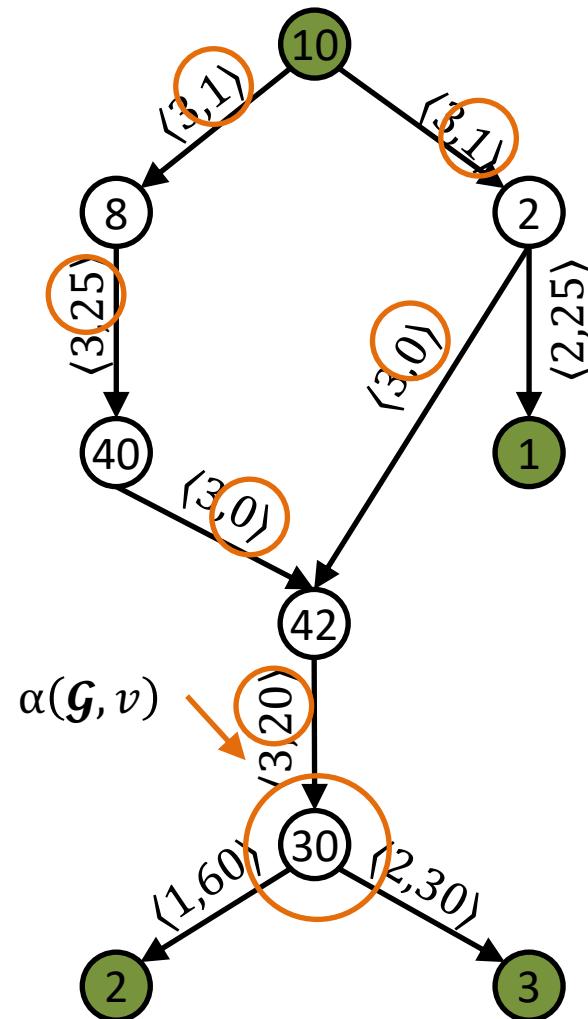
$$\max size = 55$$

$$v.s = 30$$

$$T + v.s < \max size$$

$$\rho(\mathcal{G}, v) = 3 \times (1 + 1 + 25 + 0 + 0 + 20) = 141$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 4.7$$



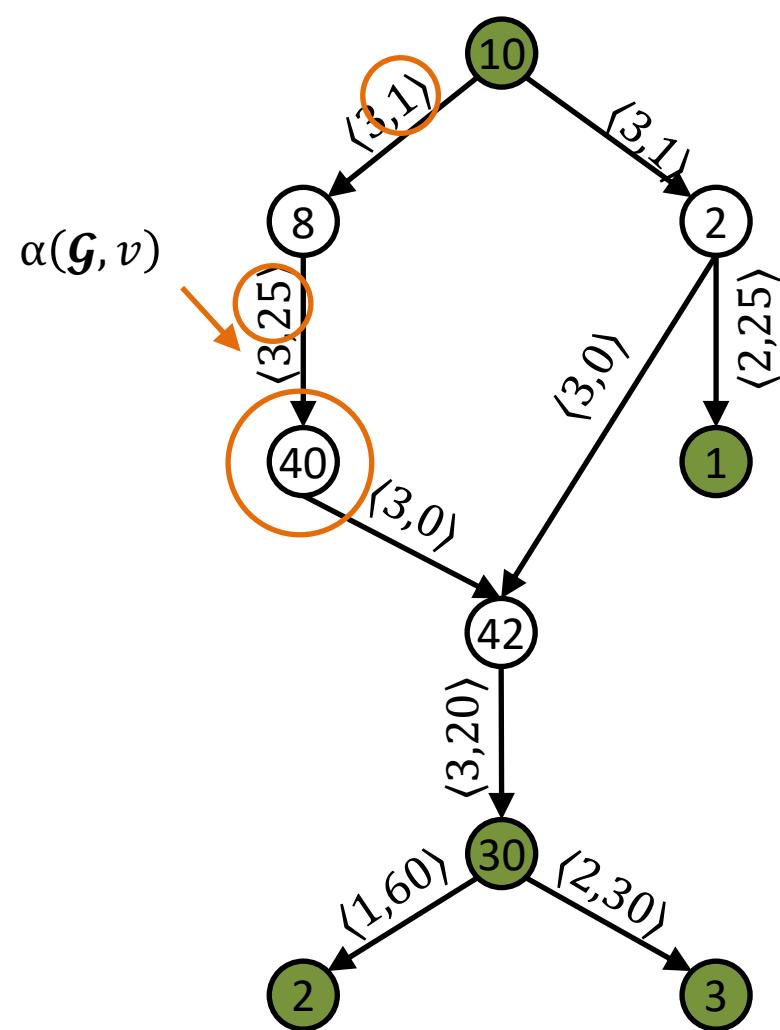
$$T = 46$$

$$v.s = 40$$

$$\cancel{T + v.s < max\ size}$$

$$\rho(\mathcal{G}, v) = 3 \times (1 + 25) = 78$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 1.95$$



$$T = 46$$

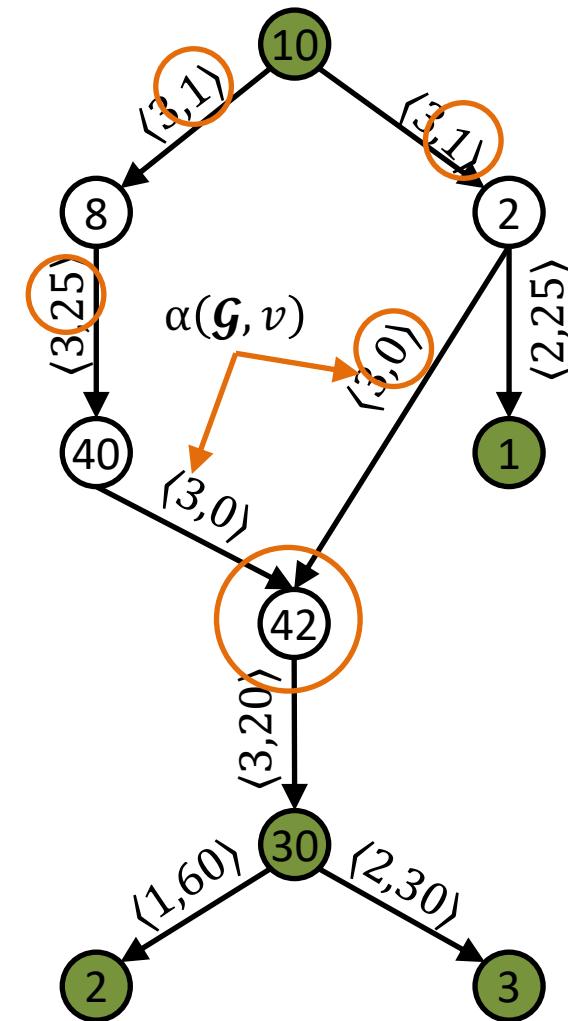
$$\max size = 55$$

$$v.s = 42$$

$$\cancel{T + v.s < \max size}$$

$$\rho(\mathcal{G}, v) = 3 \times (1 + 1 + 25 + 0 + 0) = 81$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 1.93$$



$$T = 46$$

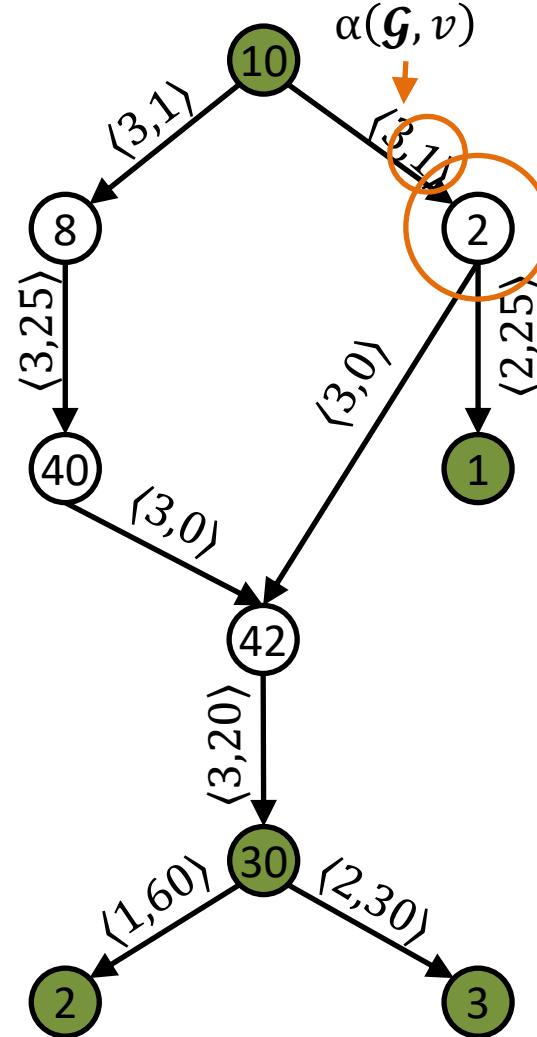
$$\max size = 55$$

$$v.s = 2$$

$$T + v.s < \max size$$

$$\rho(\mathcal{G}, v) = 3 \times (1) = 3$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 1.5$$



$$T = 48$$

$$v.s = 8$$

$$\cancel{T + v.s < max\ size}$$

$$\rho(\mathcal{G}, v) = 3 \times (1) = 3$$

$$\frac{\rho(\mathcal{G}, v)}{v.s} = 0.375$$

