Learning Job Run-Times in HPC Systems

Valentin Reis

LIG, AMA Team.
Under supervision of: Eric Gaussier and Denis Trystram.

June 16, 2014

Resource Management in HPC Systems

High Performance Computing is:

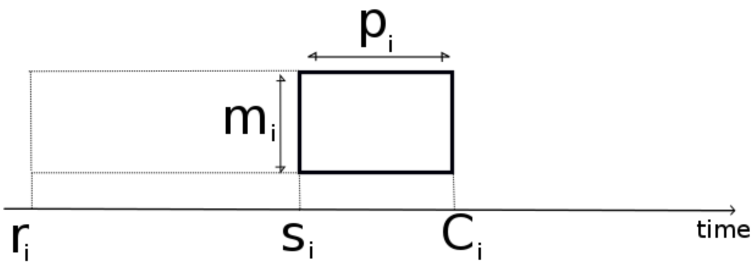
- Complex architectures.
- Uncertain data.

Jobs

This is a job:

Jobs

This is a job on a system:



Objectives

- $\sum_{i=0}^{i=n} C_i$ (Minsum)
- $\max_{i=0}^{i=n} C_i$ (Makespan)
- $\sum_{i=0}^{i=n} C_i - r_i$ (Flow Time)
- $\sum_{i=0}^{i=n} \frac{C_i - r_i}{C_i - \sigma(i)}$ (Sum Stretch)
- $\max_{i=0}^{i=n} \frac{C_i - r_i}{C_i - \sigma(i)}$ (Max Stretch)
- Many others
- And their combinations!

Objectives

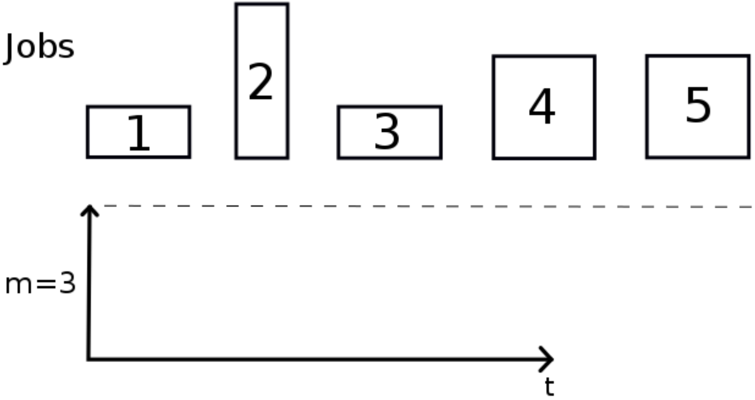
- $\sum_{i=0}^{i=n} C_i$ (Minsum)
- $\max_{i=0}^{i=n} C_i$ (Makespan)
- $\sum_{i=0}^{i=n} C_i - r_i$ (Flow Time)
- $\sum_{i=0}^{i=n} \frac{C_i - r_i}{C_i - \sigma(i)}$ (Sum Stretch)
- $\max_{i=0}^{i=n} \frac{C_i - r_i}{C_i - \sigma(i)}$ (Max Stretch)
- Many others
- And their combinations!

And: functional constraints. **no starvation**, please!

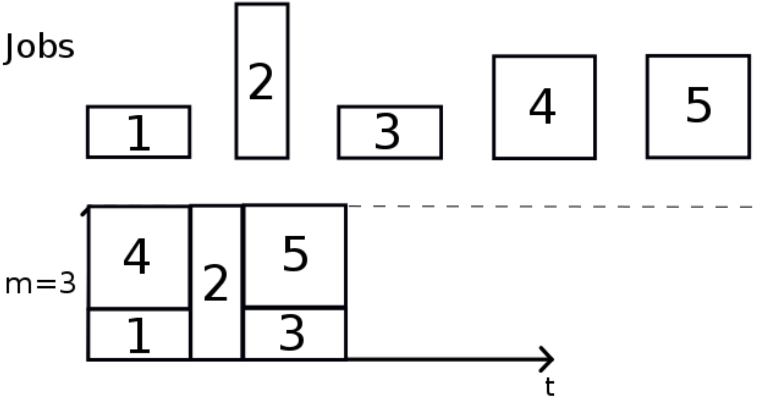
What problem is it?

- Offline Scheduling, a.k.a. strip packing?

Offline Scheduling



Offline Scheduling



What problem is it?

- Offline Scheduling, a.k.a. strip packing?

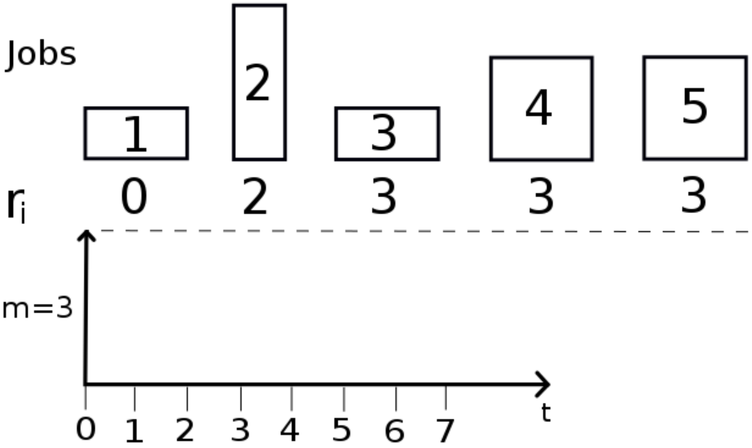
What problem is it?

- Offline Scheduling, a.k.a. strip packing? – no

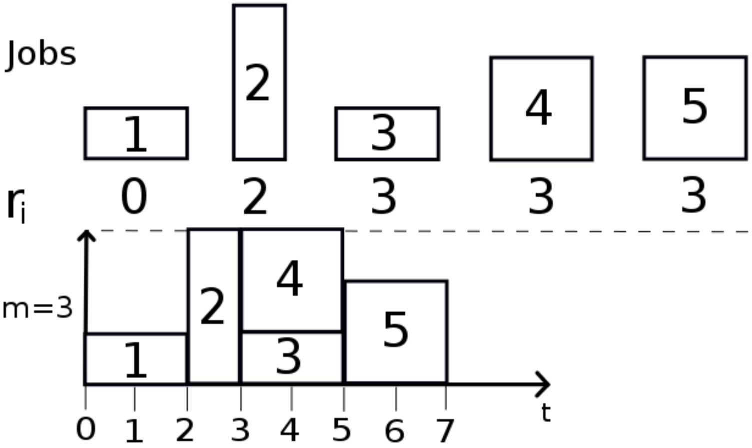
What problem is it?

- Offline Scheduling, a.k.a. strip packing? – no
- Online Scheduling?

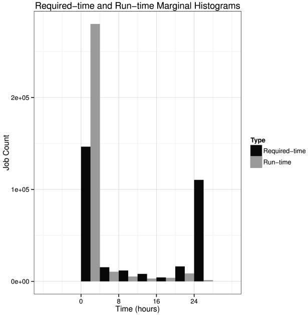
Online Scheduling



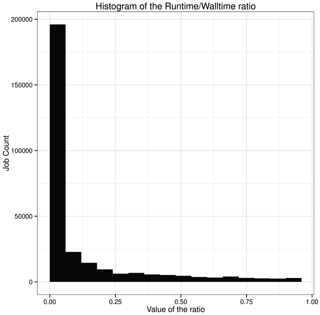
Online Scheduling



Uncertainty in Run-times



: Marginal



: Ratio

What problem is it?

- Offline Scheduling, a.k.a. strip packing? – no
- Online Scheduling?

What problem is it?

- Offline Scheduling, a.k.a. strip packing? – no
- Online Scheduling? – no

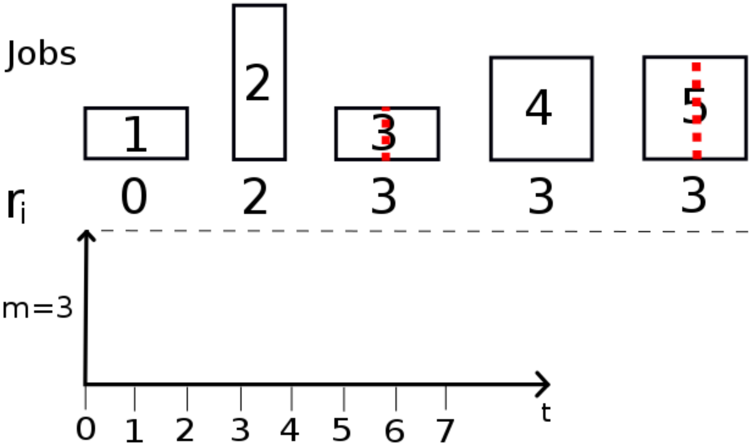
What problem is it?

- Offline Scheduling, a.k.a. strip packing? – no
- Online Scheduling? – no
- Online Scheduling under Uncertainty

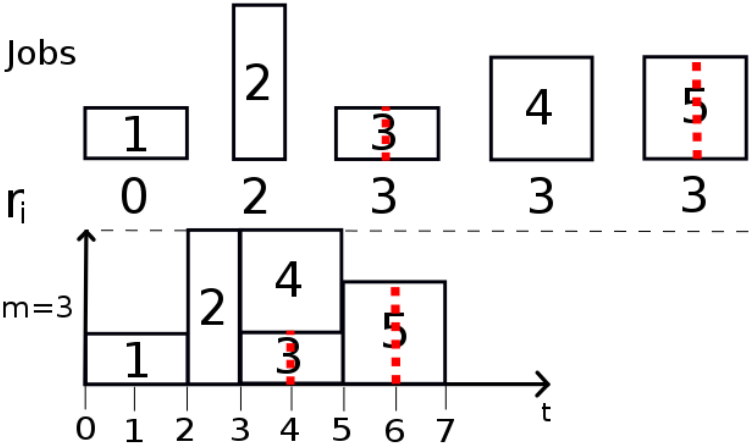
What problem is it?

- Offline Scheduling, a.k.a. strip packing? – no
- Online Scheduling? – no
- Online Scheduling under Uncertainty – yes!

Online Scheduling under Uncertainty



Online Scheduling under Uncertainty



Optimization under Uncertainty

Optimization under Uncertainty

Robust Optimization

Optimization under Uncertainty

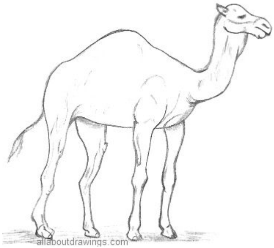
Robust Optimization

Stochastic Programming + Input Modeling

Optimization under Uncertainty

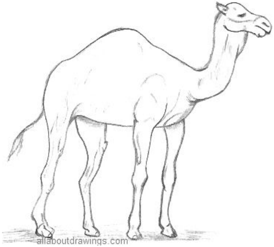
Robust Optimization

Stochastic Programming
$+$ Input Modeling

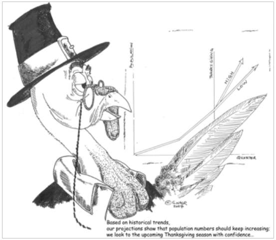


Optimization under Uncertainty

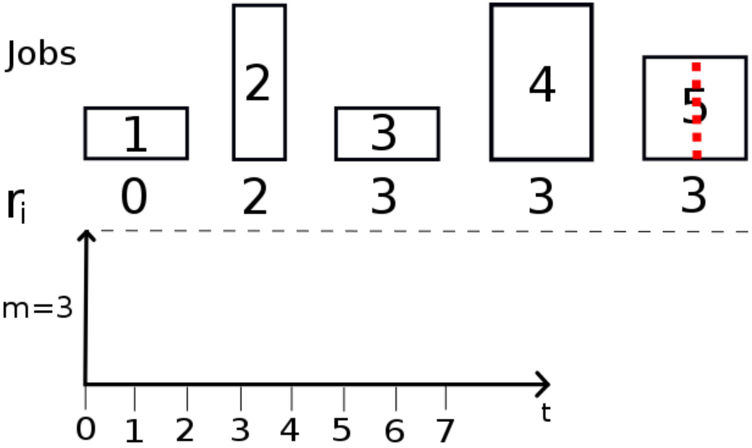
Robust Optimization



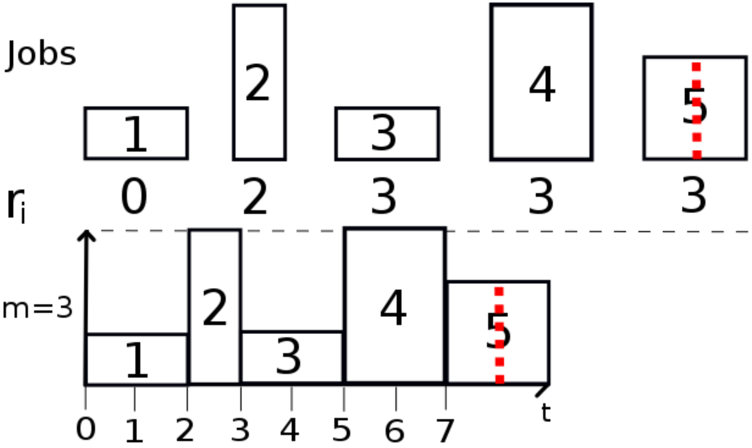
Stochastic Programming
+ Input Modeling



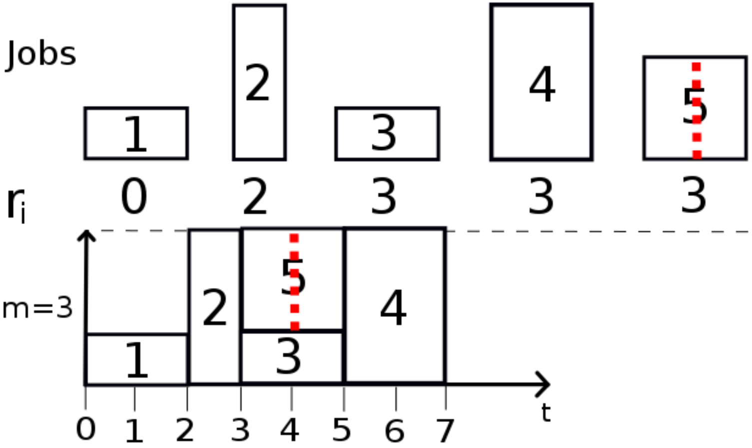
The FCFS algorithm



The FCFS algorithm

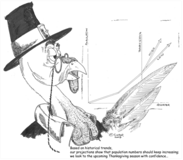


FCFS with EASY(conservative) backfilling



Runtime Prediction

Problem Statement: How to best Predict the Runtime of a job on a given system?



Nature of the Prediction

- Single-valued?
- Probability distribution?
- Confidence Interval?

Nature of the Prediction

- **Single-valued**
- Probability distribution
- Confidence Interval

Hypotheses on the Runtime

- Independent and identically distributed?
- Is it dependent on job characteristics?

Hypotheses on the Runtime

- Independent and identically distributed? **No.**
- Is it dependent on job characteristics? **Yes.**

State of the Art

A very popular predictor:

$$p_i^u = \frac{p_{i-1}^u + p_{i-2}^u}{2}$$

Baseline for evaluating our method.

Our Approach

Machine Learning: Regression problem.

Our Approach

Machine Learning: Regression problem.
Build vectors containing:

Our Approach

Machine Learning: Regression problem.
Build vectors containing:

- Job characteristics (e.g., required time and nodes)

Our Approach

Machine Learning: Regression problem.
Build vectors containing:

- Job characteristics (e.g., required time and nodes)
- Status of the system (e.g., day of the week)

Our Approach

Machine Learning: Regression problem.
Build vectors containing:

- Job characteristics (e.g., required time and nodes)
- Status of the system (e.g., day of the week)
- Characteristics of the last jobs of this user (e.g., the baseline)

Our Approach

Machine Learning: Regression problem.
Build vectors containing:

- Job characteristics (e.g., required time and nodes)
- Status of the system (e.g., day of the week)
- Characteristics of the last jobs of this user (e.g., the baseline)

Perform Regression using the Random Forest algorithm.

Random Forests

- Training:
  - Randomly partition the training data.
  - Learn a decision tree on each subset.

Decision Tree Learning

$$X = \begin{pmatrix} x \\ y \end{pmatrix}, (\alpha, \beta, a, b, c) \in \mathbb{R}^5$$

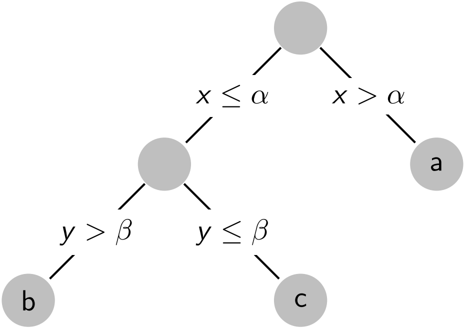


Figure: A decision tree for predicting the value of X.

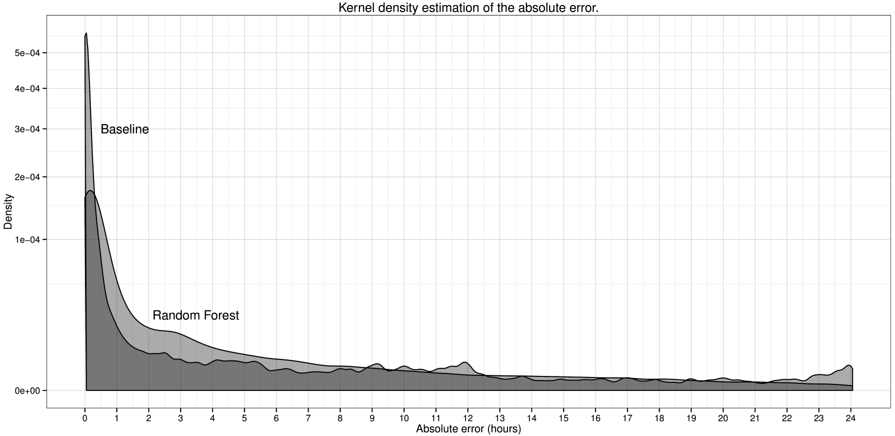
Random Forests

- Training:
  - Randomly partition the training data.
  - Learn a decision tree on each subset.
- Predict a value by averaging the results from all the decision trees.

Experiments

On the CURIE log:
The features are extracted with the SimPy discrete event simulation package.
The RF is trained on the CURIE log using the first 80% of the jobs, using the Scikit-Learn package.

Results: Absolute Error Distribution



Results: metrics

Measure	Baseline	Random Forest
MSE	1.98497515×10^8	1.58202218×10^8
MAE	4.680836×10^3	5.551237×10^3
Standard Error	5.312812×10^1	4.512378×10^1

Perspectives

- Online Algorithm
- More datasets, more features.
- Sensitivity analysis, objective/cost functions.

Thank you!