Modelling Fault Dependencies when Execution Time Budgets are Exceeded

David Griffin, Benjamin Lesage, Iain Bate,
Frank Soboczenski, Rob Davis

Introduction

- When we test out analyses, we make a lot of assumptions
  - We have representative data (Statistical methods)
  - A uniform distribution represents all possible inputs (Scheduling algorithms)
- Mostly, these assumptions are OK
  - Pragmatic assumptions which simplify our problems from impossible to doable

Introduction

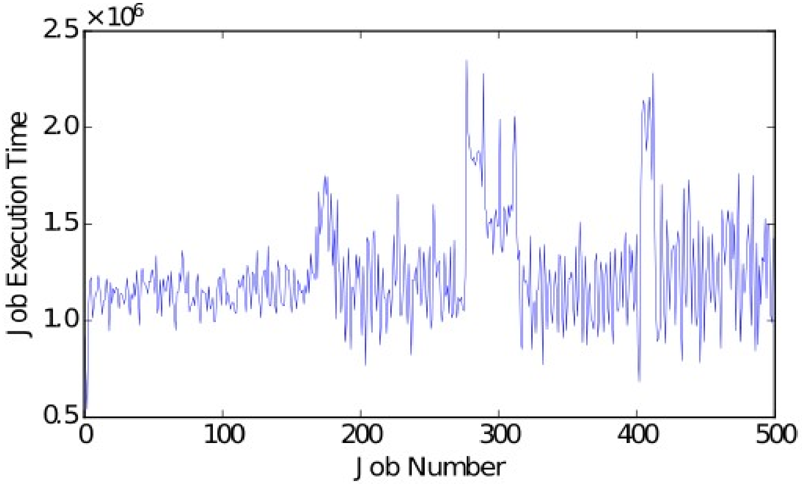
- But what about this one?
  - Job Execution Times are Independent
- It's made in a lot of places
  - Testing scheduling, early versions of MBPTA
- Problem:
  - It is completely unrealistic

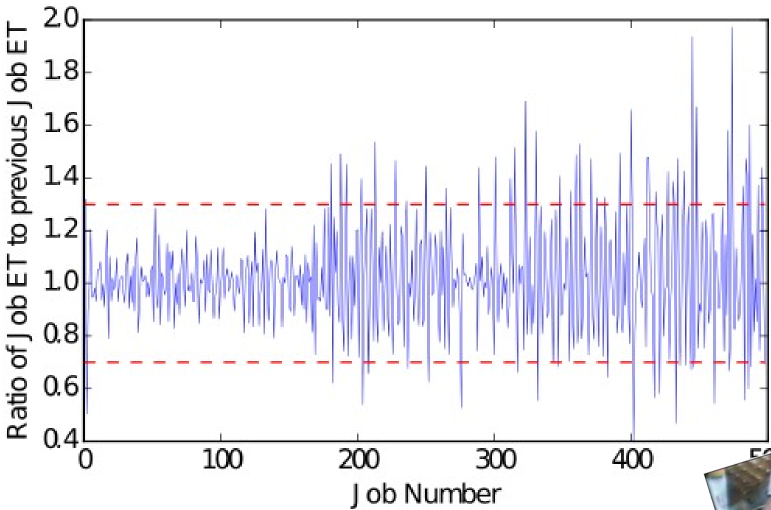
Dependencies of Job Execution Times

- Regardless of any intelligent processor design, the job still performs a task
- Presumably, the inputs to the job effect what the job will do
- So if there are dependencies in Job inputs, there are also dependencies in Job execution times
- Is there any realistic system where Job inputs are I.I.D random variables?

Dependencies of Job Execution Times

- FFMPEG Decoding video frames under Valgrind
  - FFMPEG provides a lot of 'real world' data
  - Valgrind instruction counts give exact instrumentation
- Inputs – Video frame data, current decoder state
- Very good indicator of job execution time is previous job execution time





RTS York

UNIVERSITY of York

Dependencies of Job Failures

- Given that Job execution times are dependent, Job failures due to budget overruns are also dependent
- Which means that they are definitely not I.I.D.

Is this a problem?

- Potentially – a lot of places make the assumption that execution times are independent
- For example, mixed criticality systems – how do these fare if overall the low-crit failure rate is 10^{-4}, but all those failures happen one after the other?
- Not something that has been tested currently

Problems

- Can't directly model failure durations
- Why?
  - Deadline overrun failures are rare events by definition
  - Won't get enough data
- Need to predict failure durations from data gathered in testing

Solution

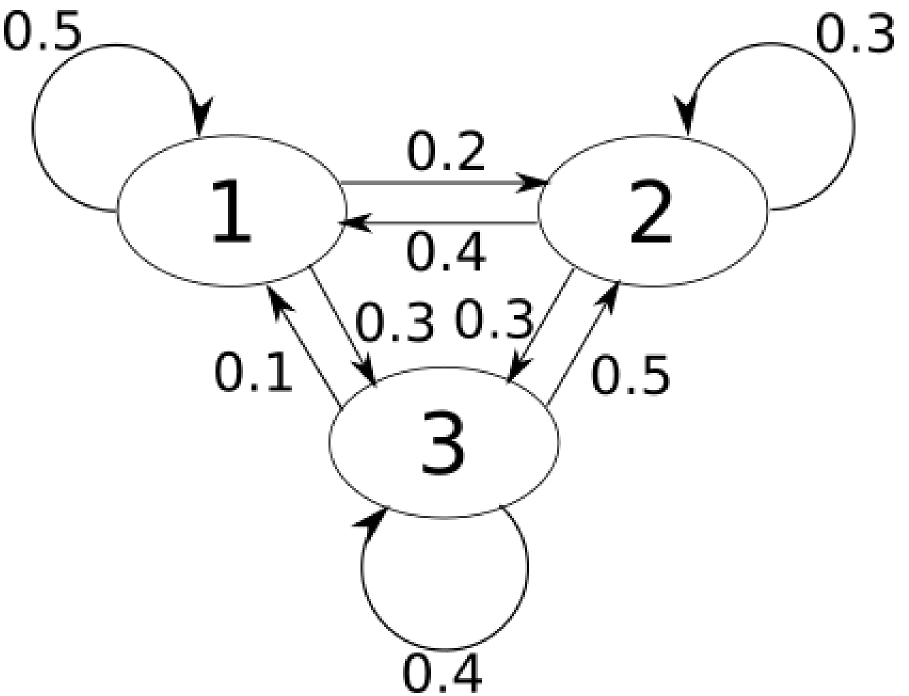
- Forecasting
- Take data from normal case, generate a model, use model to predict behaviour under extreme circumstances
  - Works around scarcity of actual data

Solution (Part 2)

- How to do forecasting?
- For this case, using Model Extrapolation
- Given a set of models with the same structure, fit curves to the model parameters and extrapolate
- To get models, using Markov Chains and Lossy Compression

Markov Chain Models

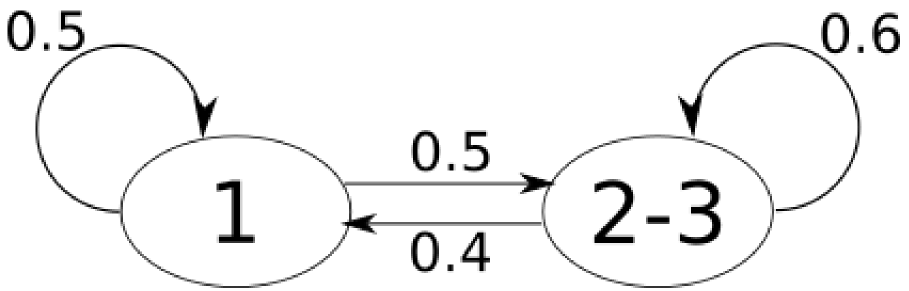
- Set of states corresponding to duration of overruns
- Each state has probability to change to any other state
- Simply trained on test data
  - Other ways to generate models being explored



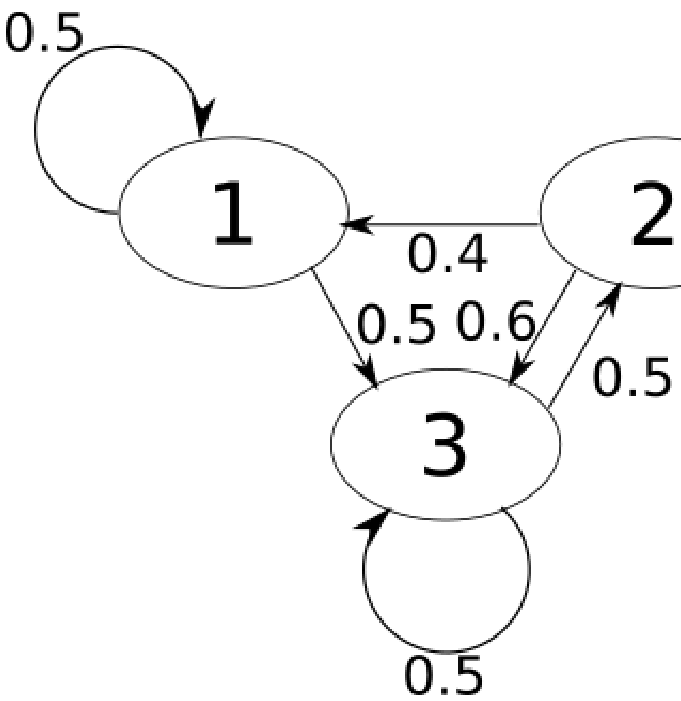
Compressed Markov Models

- Don't want to accept models which don't have enough data to back up each point
- Solution is to permit compressing the model
- Compression combines states/transitions which have a low amount of data behind each point
- Various methods of compression

Compressed Markov Models

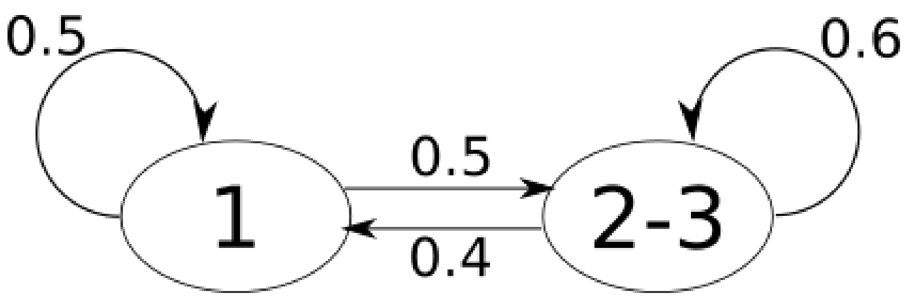


Combine States

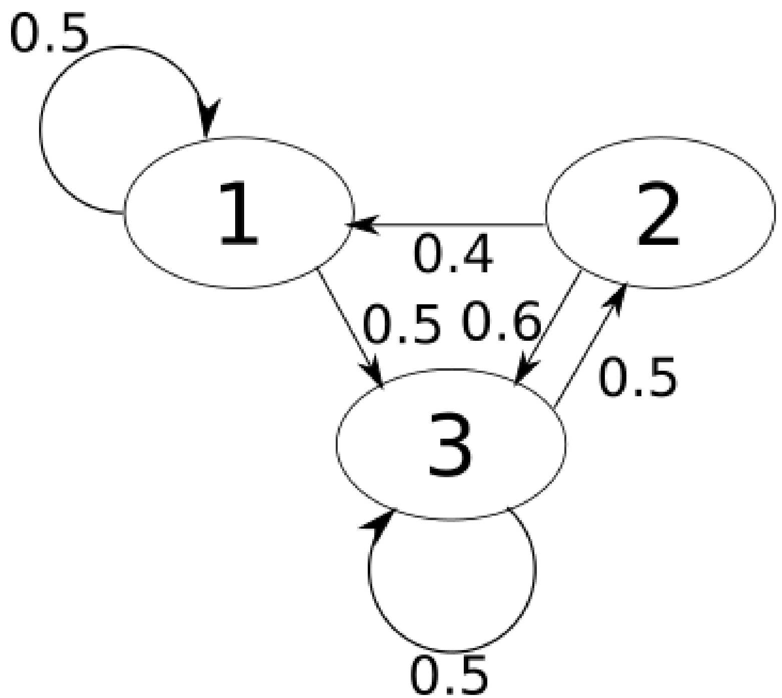


Combine Transitions

Compressed Markov Models



Combine States

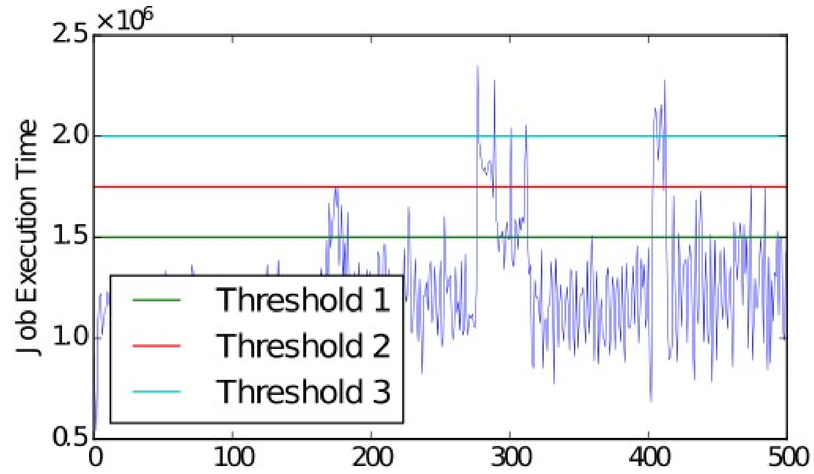


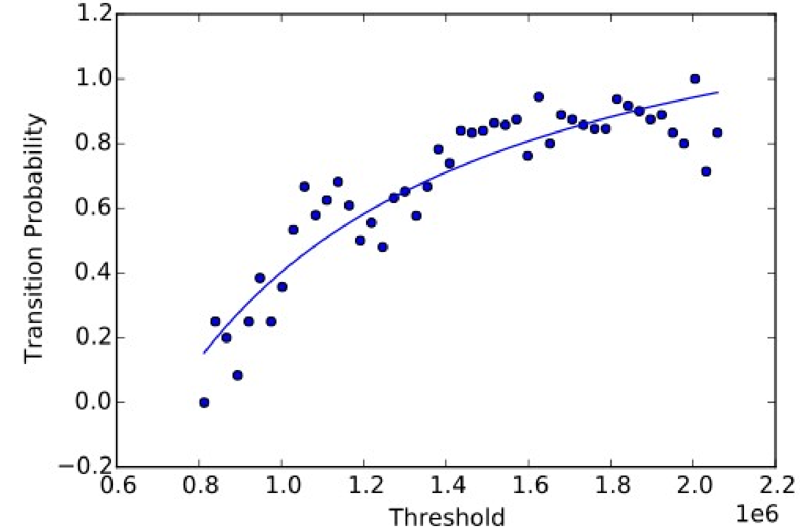
Combine Transitions

One catch: Will want the user to tell us what intervals we're Interested in

Forecasting

- Outline of method
  - Set multiple exceedance thresholds where there is sufficient data
  - Create models, using compression as needed
  - Find the most common shape of compressed model
  - Fit curves and extrapolate to the desired failure rate

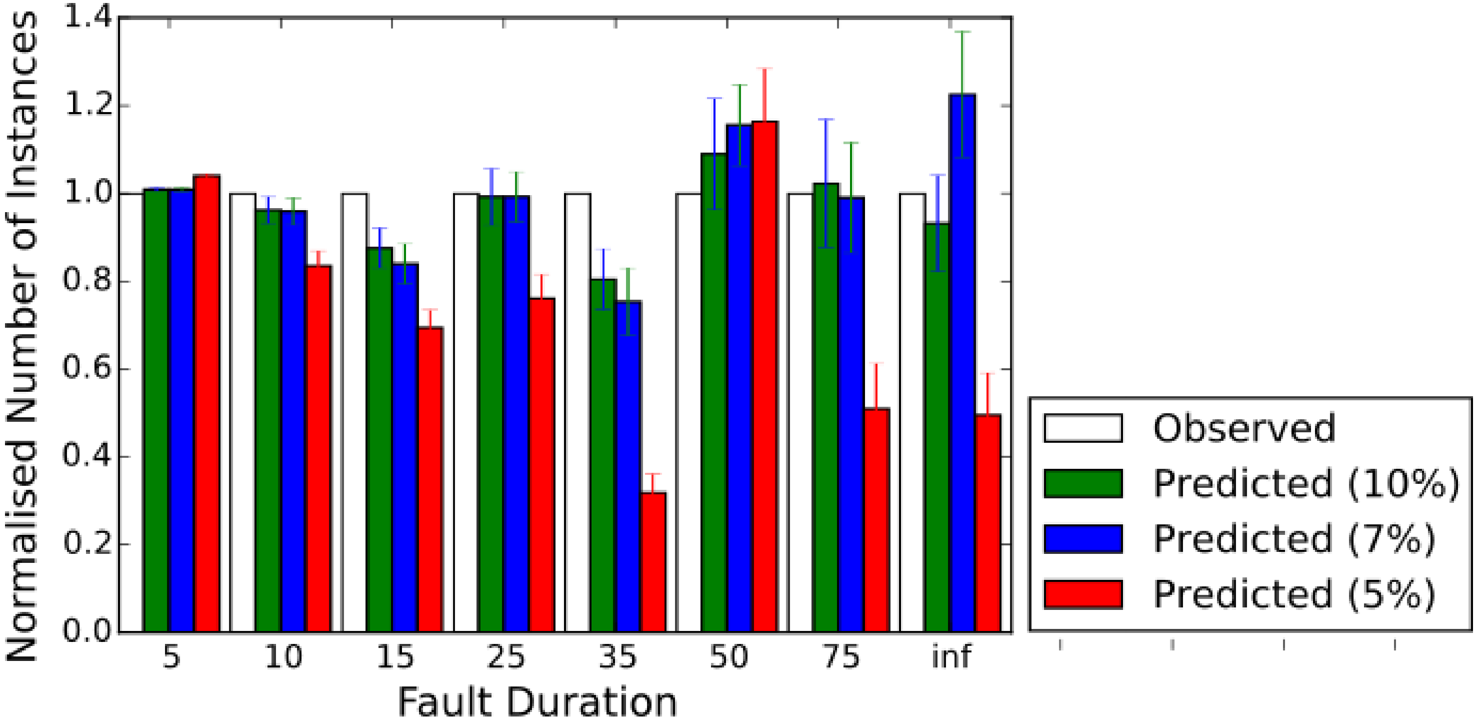




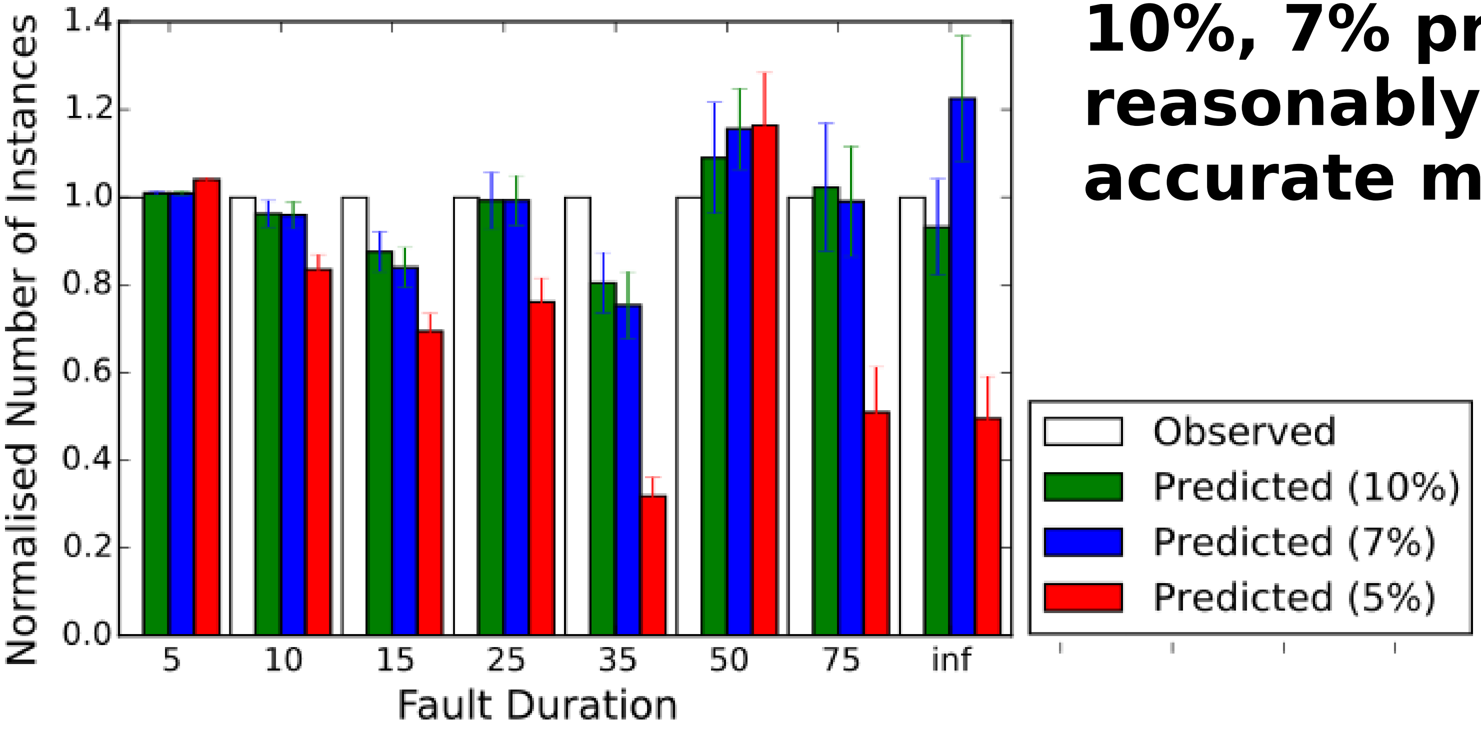
Evaluation

- Method
- Get a large amount of data from target system
  - FFMPEG decoding videos under Valgrind
- Create a forecast model based on a subset of the data
  - Subset of data does not have enough data to reliably model at desired confidence levels
- Compare results from forecast model with what actually happened

Results

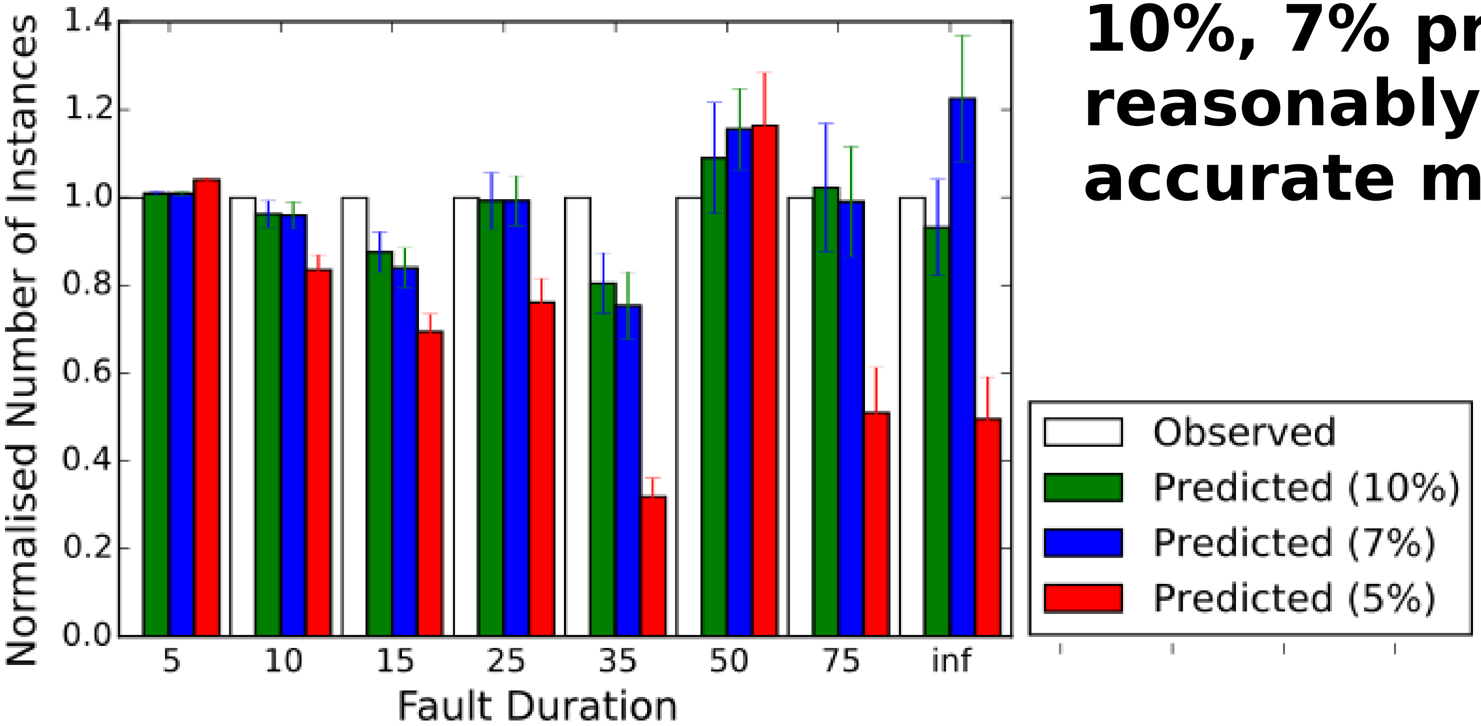

Normalised Number of Instances
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
5
10
15
25
35
50
75
inf
Fault Duration
Observed
Predicted (10%)
Predicted (7%)
Predicted (5%)

Results



10%, 7% produce reasonably accurate models

Results



10%, 7% produce reasonably accurate models

5% has very low accuracy, but first 5% of input data is not representative

RTS York

UNIVERSITY of York

DepET – A dependent execution time generator

- Have a way to generate exceedance durations at an arbitrary threshold
- DepET is an algorithm to utilise this to generate dependent execution times
- So as UUniFast generates a useful spread of realistic task utilisations, DepET generates realistic task execution times

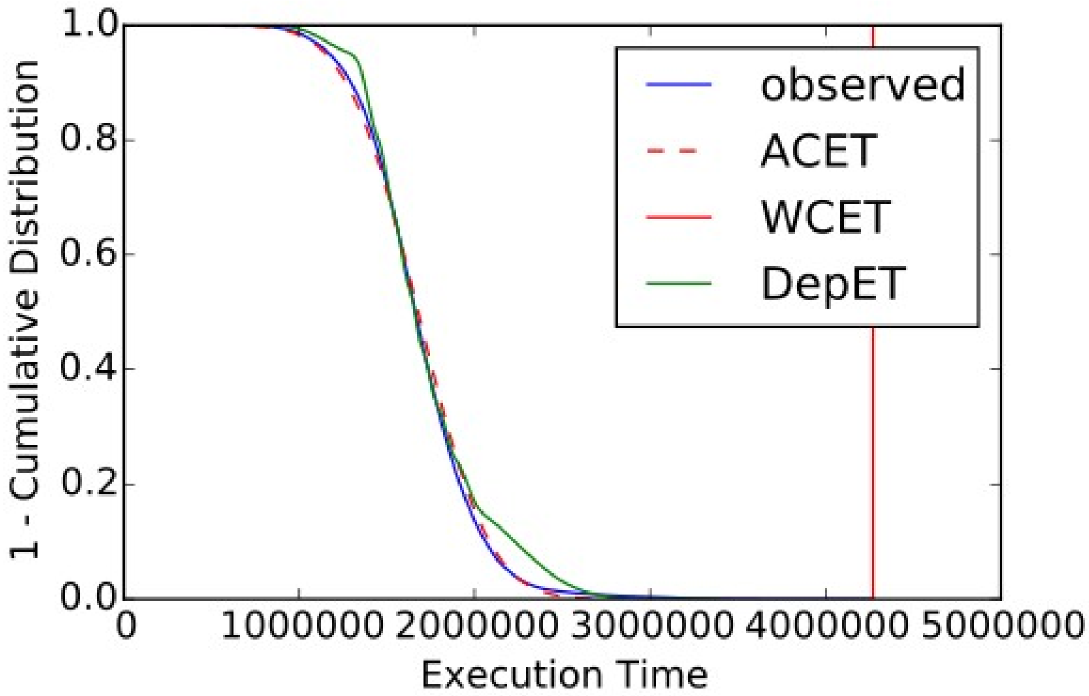
DepET Algorithm

- Divide execution time into a series of bands
- Each invocation has a probability of exceeding it's current band
- An exceedance model governs the duration of this exceedance
- Otherwise, randomly move about inside the band

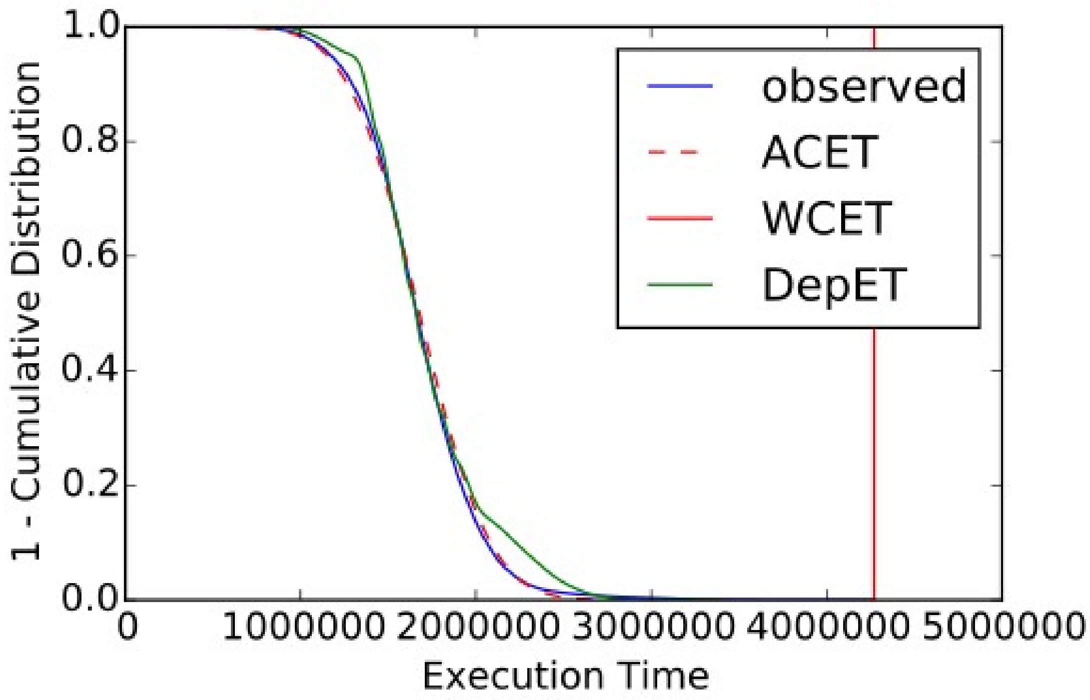
DepET Evaluation

- Compared against SimSO ACET method and observations
- SimSO ACET method implemented as normal distribution with parameters derived from training data
- Useful comparison as it attempts to be realistic
- Note: Other methods of execution time generation are few in number, and may not be trying to be realistic to compare against
  - e.g. SimSO WCET method

DepET Overall Distribution

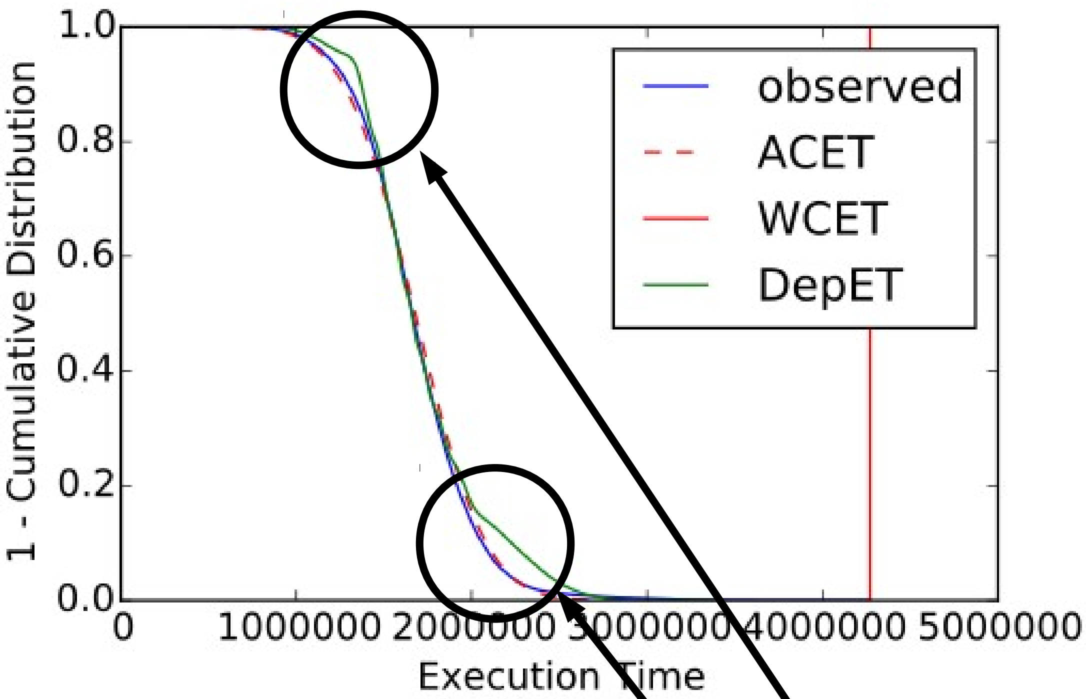


DepET Overall Distribution



Both methods give a good overall fit

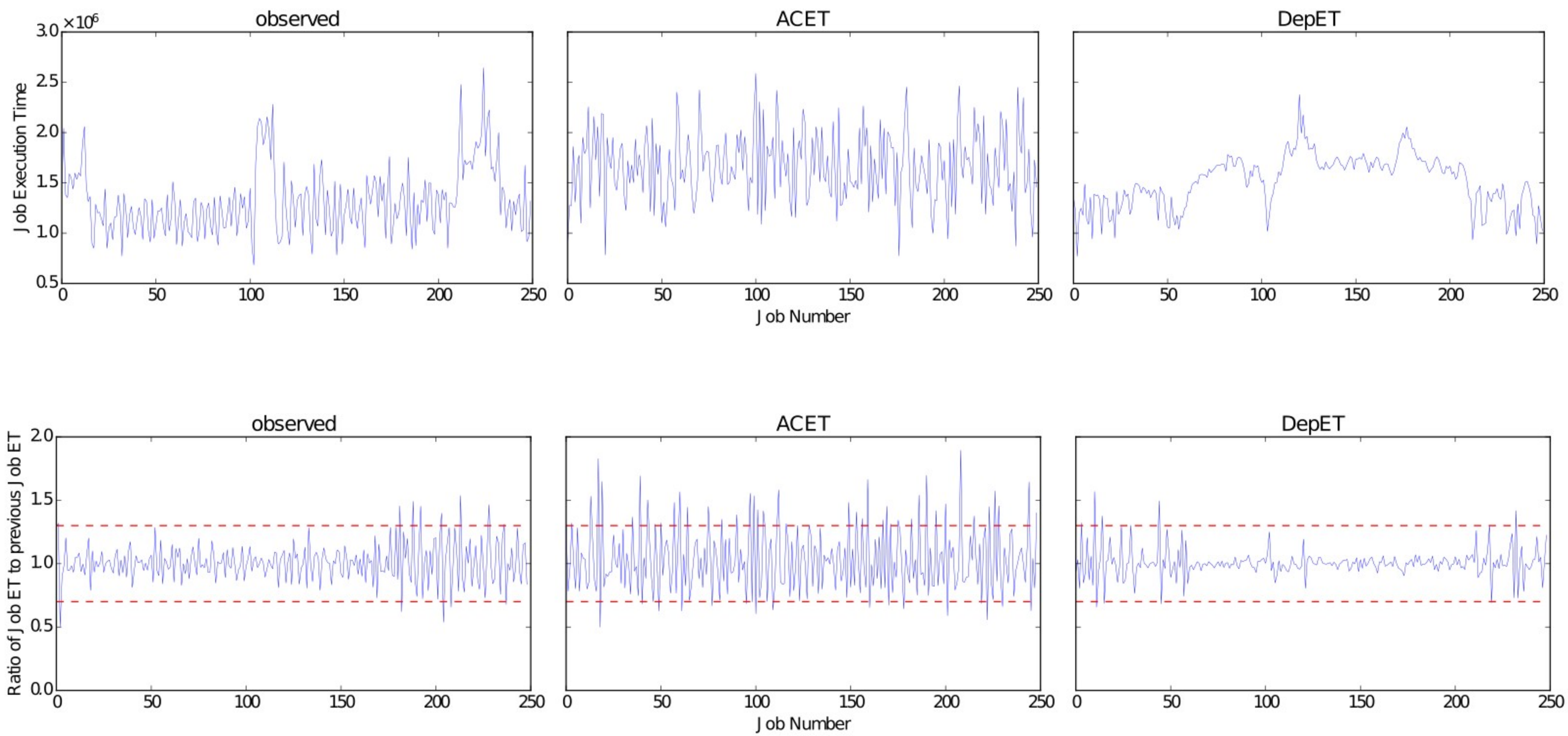
Evaluation - Overall Distribution



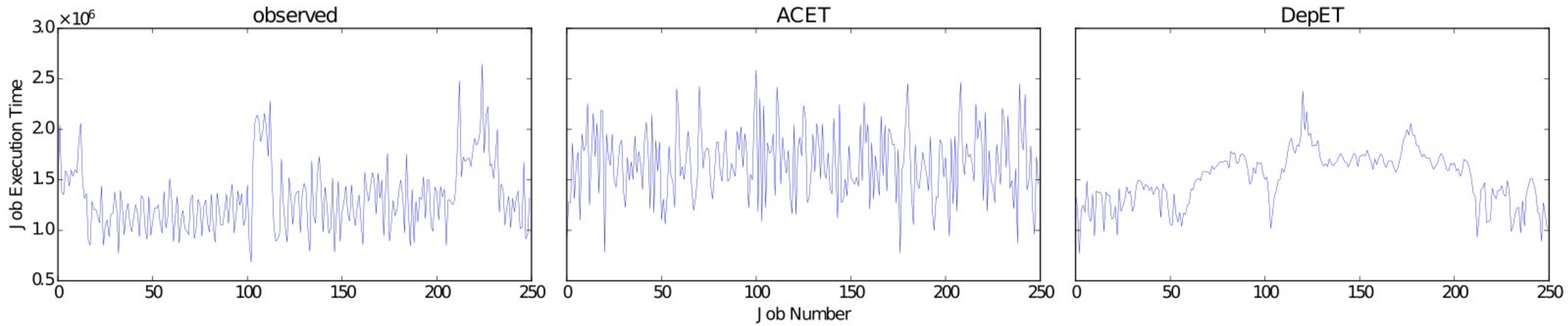
Both methods give a good overall fit

Although DepET has some inaccuracies due to compression

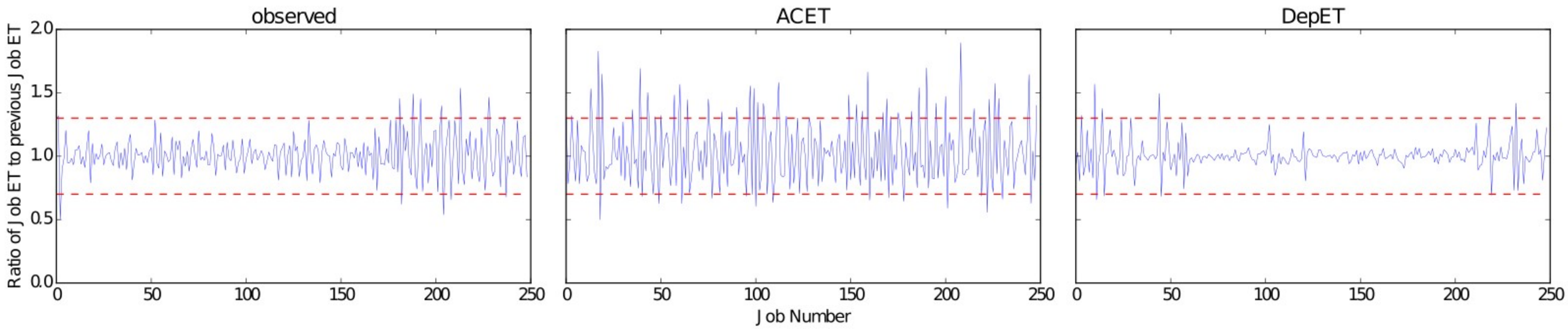
Evaluation - Dependencies



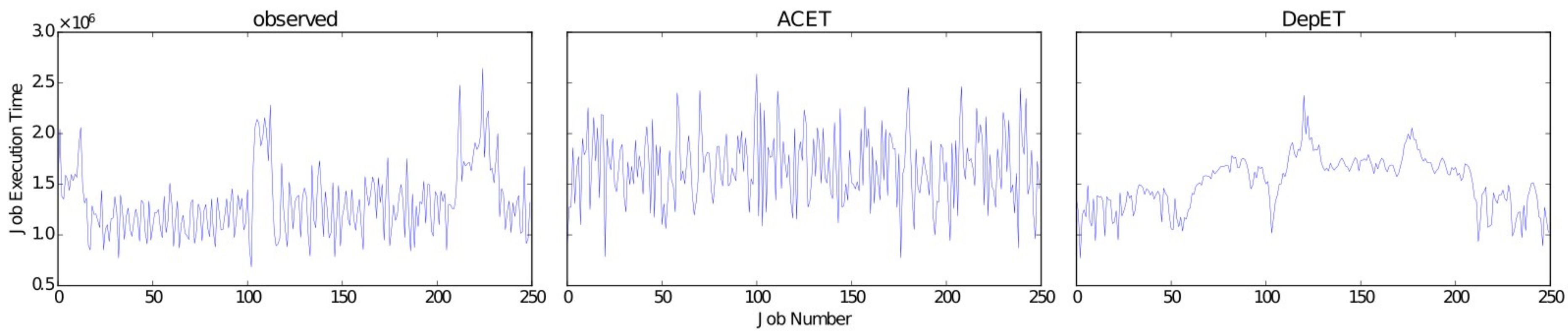
Evaluation - Dependencies



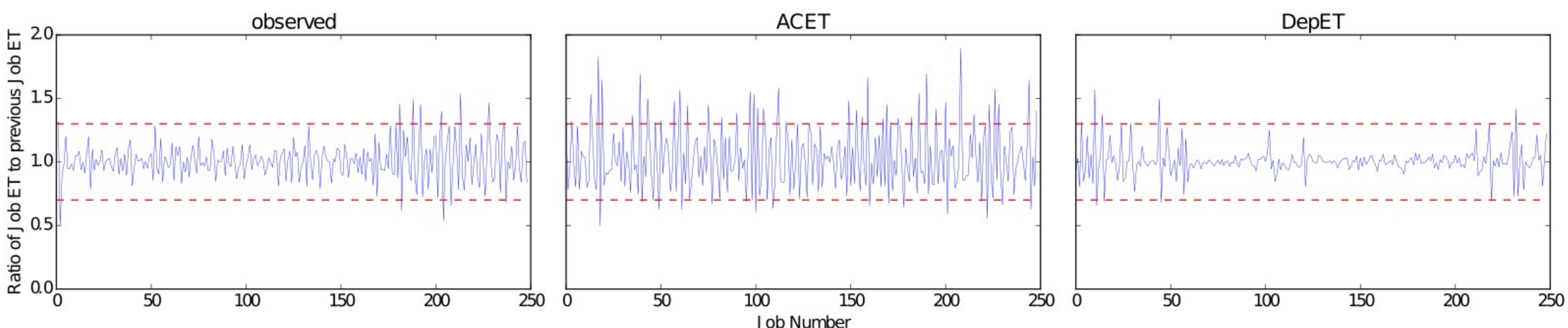
ACET has no dependencies



Evaluation - Dependencies



ACET has no dependencies



Observed and DepET have dependencies
In both, a good indicator of job execution time
Is previous job execution time

Conclusions

- Need to do better on dependencies between job execution times
  - Independence is not a realistic assumption
- Forecasting can be used to determine the expected duration of faults with reasonable accuracy
- Possible to use forecast models to generate dependent execution times using the new DepET algorithm

Any Questions?