Modelling job allocation where service duration is unknown

Nigel Thomas

University of Newcastle upon Tyne

Contents

- Motivation
- TAG
- Aims
- The basic model
- Some initial results
- Phase type distributions
- More results
- Conclusions

Motivation: Grid computing

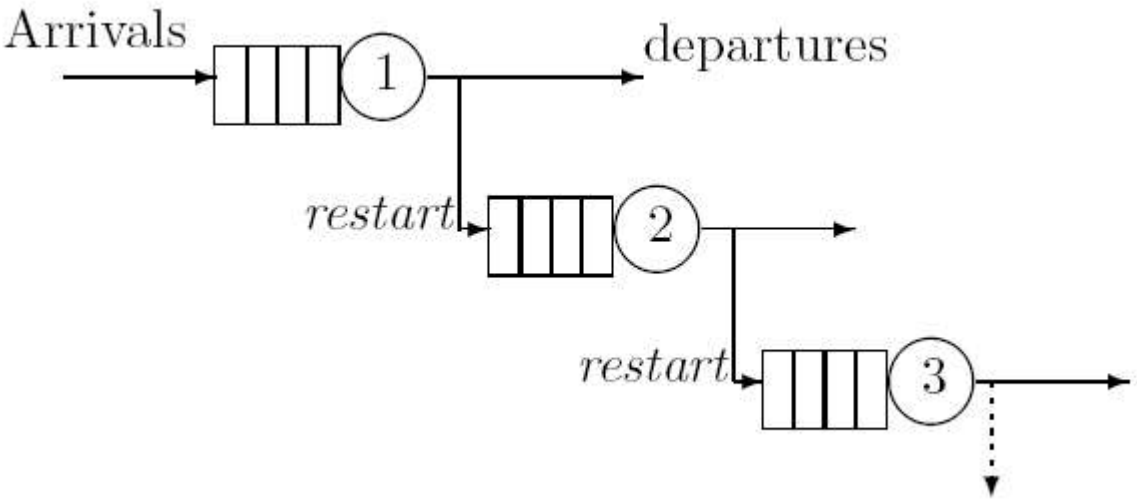
- The promise of grid computing is seamless integration of multiple resources from multiple domains.
  - Customers are promised the ability to automatically select the best resources for their task.
  - Providers are promised the ability to schedule tasks efficiently according to whatever criteria they specify.
- Currently the reality is somewhat different!
- This is due (in part) to the difficulty of finding the necessary information.
- Therefore an important research topic is determining the best strategies to employ in grid scheduling and brokerage with limited available information.

Motivation: fair use of resources

- Suppose there are *N* jobs waiting to be served, but the head job is much larger than all the others, then all the jobs must wait a long time.
- If we have a choice of servers/queues then we would like to allocate jobs queues according to their size to prevent (or reduce) this problem.
- However, estimating job duration is notoriously costly and unreliable, so what can we do?
  - Pull jobs from a central resource.
  - Assign jobs to the service centre with the shortest queue.
  - Assign jobs to service centres on a round robin basis.
  - Assign jobs to service centres randomly according to probabilities based on service capacity.

TAG (Mor Harchol-Balter, 2002)

- Seeks to reduce variance at each queue.
- Allows shorter jobs to proceed quickly, but delays longer jobs more.
- Employs a progressive filtering mechanism.
- Stops jobs that execute for too long and re-queues them at the subsequent node.



Aims of this paper

- To model the TAG scheme in PEPA.
- To investigate different service distributions (initially negative exponential and Cox).
- To investigate finite capacity queues, and hence consider job loss.

The model

- A 2 node system modelled in PEPA.
- The deterministic timeout is modelled with an Erlang distribution.
- Queues are modelled along conventional state based lines (queue-centric).
- The second queue employs an Erlang repeat service followed by the exponential remainder.

$$Node_1 \underset{\{timeout\}}{\bowtie} Node_2$$

Node 1

$$
\begin{aligned}
Queue1_0 &\stackrel{def}{=} (arrival, a).Queue1_1 \\
Queue1_i &\stackrel{def}{=} (arrival, a).Queue1_{i+1} + (service1, \top).Queue1_{i-1} \\
&\quad +(timeout, \top).Queue1_{i-1} + (tick1, \top).Queue1_i \ , 1 \leq i < K_1 \\
Queue1_{K_1} &\stackrel{def}{=} (timeout, \top).Queue1_{K_1-1} + (service1, s).Queue1_{K_1-1} \\
&\quad +(tick1, \top).Queue1_{K_1}
\end{aligned}
$$

$$
Server1 \stackrel{def}{=} (service1, s).Server1
$$

$$
\begin{aligned}
Time1_0 &\stackrel{def}{=} (timeout, t).Time1_n + (service1, \top).Time1_n \\
Time1_i &\stackrel{def}{=} (tick1, t).Time1_{i-1} + (service1, \top).Time1_n \ , 1 \leq i \leq n
\end{aligned}
$$

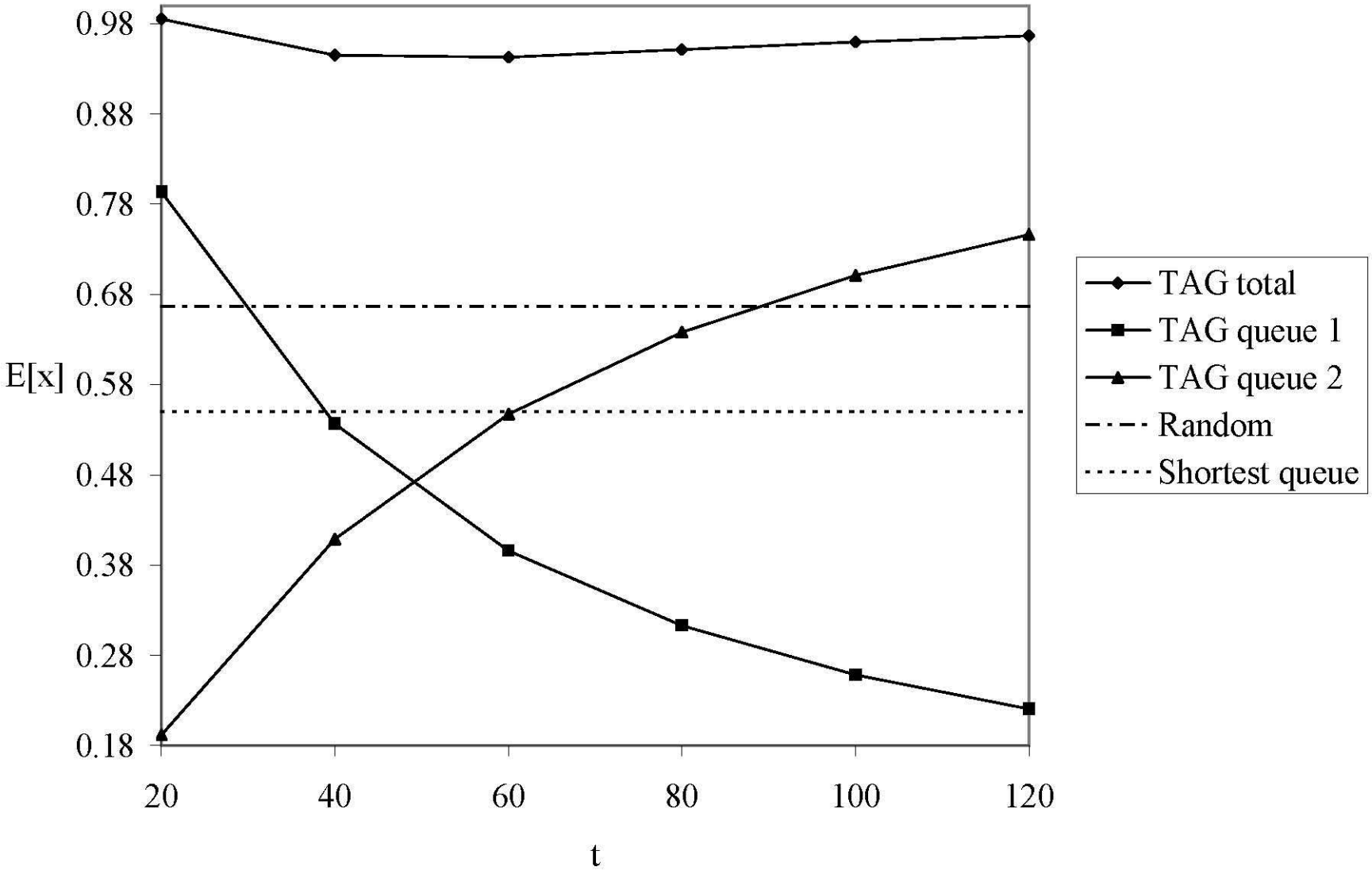
$$
Node_1 \stackrel{def}{=} Server1 \underset{\{service1\}}{\bowtie} Queue1_0 \left\{ \underset{service1, tick1}{\overset{timeout,}{\bowtie}} \right\} Time1_n
$$

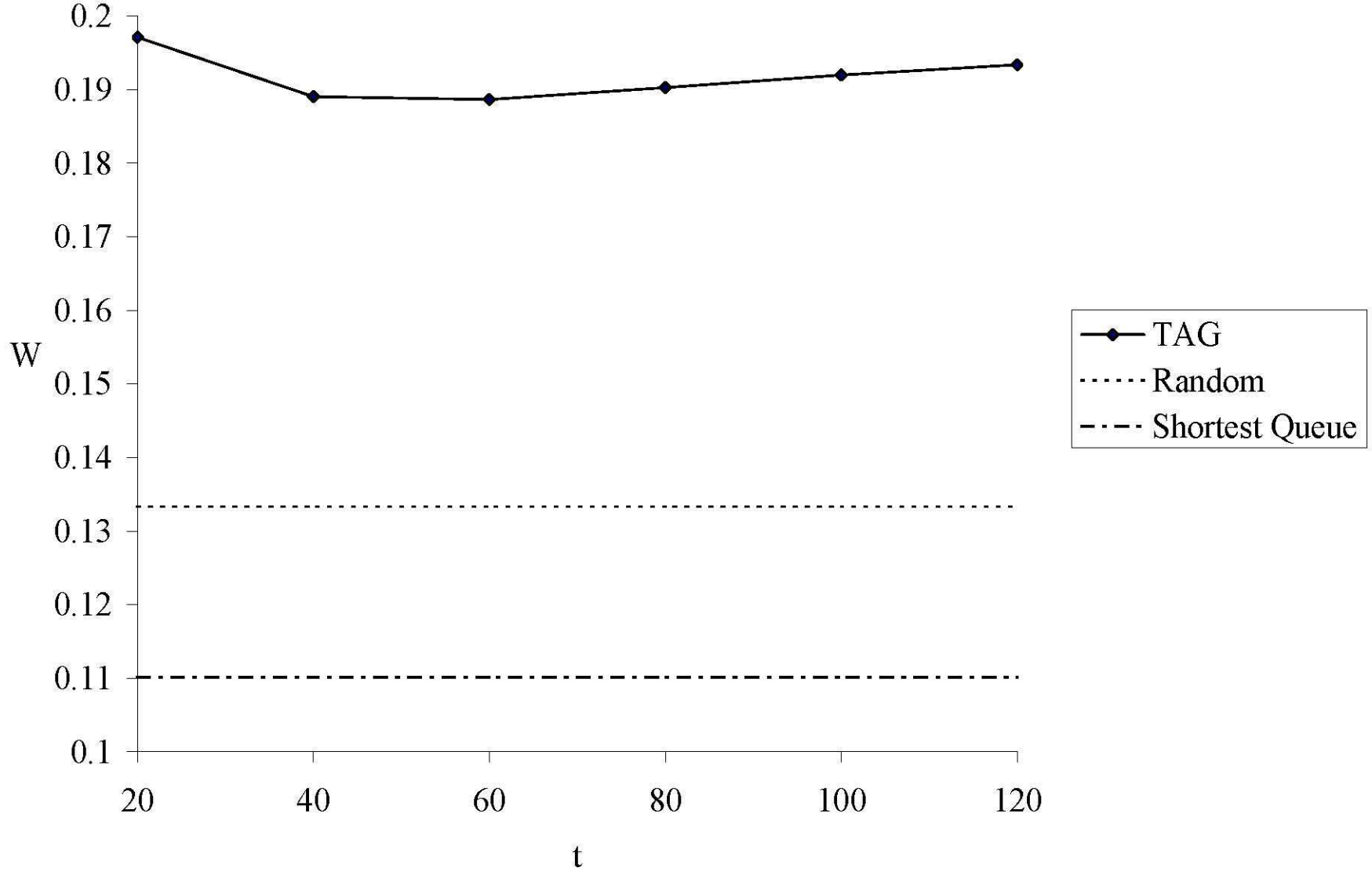
Node 2

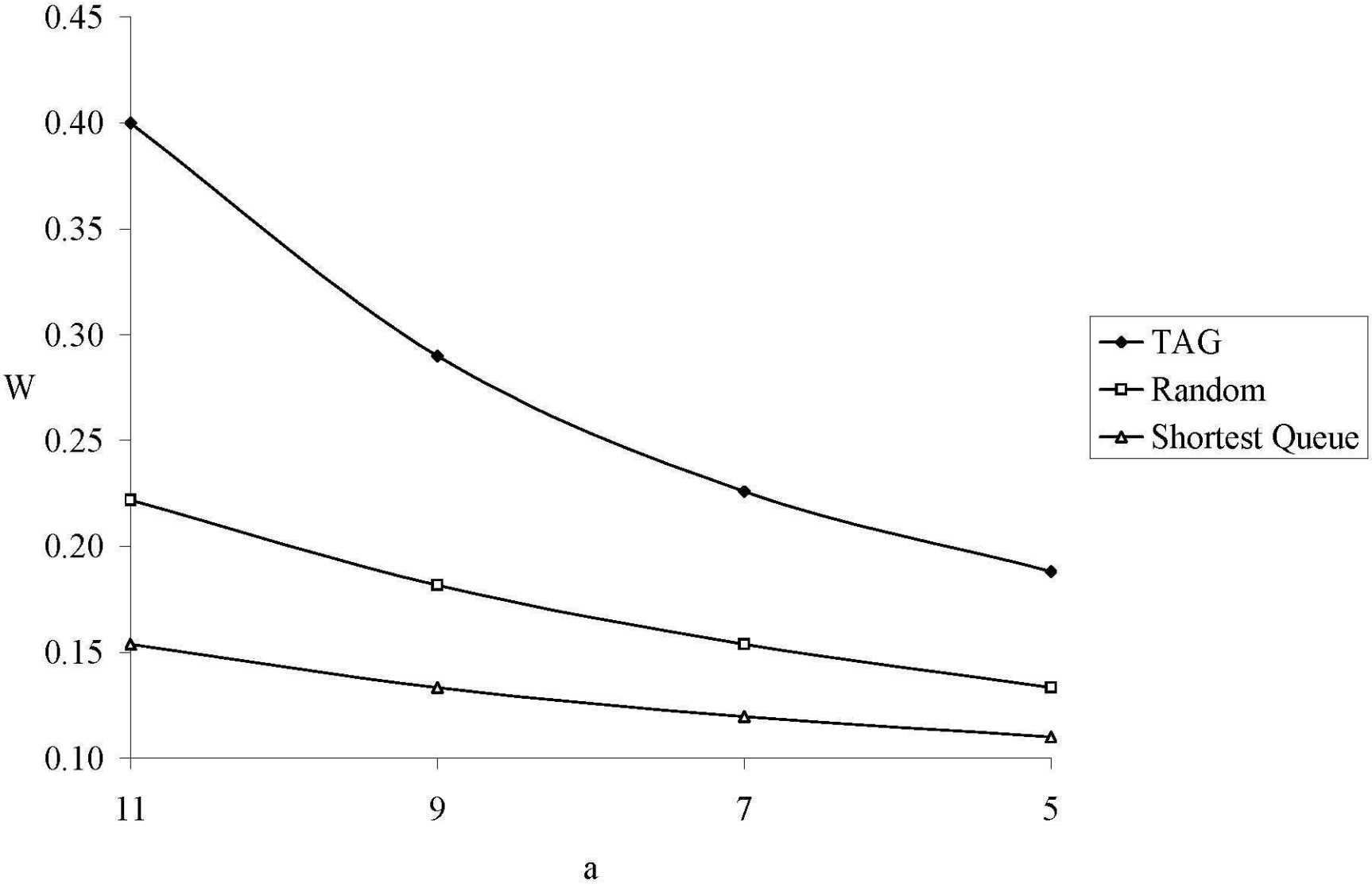
$$
\begin{aligned}
Queue2_0 &\stackrel{def}{=} (timeout, \top).Queue2_1 \\
Queue2_i &\stackrel{def}{=} (timeout, \top).Queue2_{i+1} + (repeatservice, \top).Q2_i \\
&\quad +(tick2, \top).Queue2_i \ , 1 \leq i < K_2 \\
Q2_i &\stackrel{def}{=} (timeout, \top).Q2_{i+1} + (service2, s).Queue2_{i-1} \\
&\quad +(tick2, \top).Q2_i \ , 1 \leq i < K_2 \\
Queue2_{K_2} &\stackrel{def}{=} (timeout, \top).Queue2_{K_2} + (repeatservice, \top).Q2_{K_2} \\
&\quad +(tick2, \top).Queue2_{K_2} \\
Q2_{K_2} &\stackrel{def}{=} (timeout, \top).Q2_{K_2} + (service2, s).Queue2_{K_2-1} \\
&\quad +(tick2, \top).Q2_{K_2} \\
Server2 &\stackrel{def}{=} (service2, s).Server2 \\
Time2_0 &\stackrel{def}{=} (repeatservice, t).(service2, \top).Time2_n \\
Time2_i &\stackrel{def}{=} (tick2, t).Time2_{i-1} \ , 1 \leq i \leq n
\end{aligned}
$$

$$
Node_2 \stackrel{def}{=} Server2 \underset{\{service2\}}{\bowtie} Queue2_0 \overset{\bowtie}{\underset{\left\{\substack{repeatservice,\\ service2,tick2}\right\}}{}} Time2_n
$$

Initial results (1/3)




0.2
0.19
0.18
0.17
0.16
0.15
0.14
0.13
0.12
0.11
0.1
W
20
40
60
80
100
120
t
TAG
Random
Shortest Queue


0.45
0.40
0.35
0.30
0.25
0.20
0.15
0.10
W
11
9
7
5
a
TAG
Random
Shortest Queue

Phase type distributions

- Consist of multiple exponentials, e.g.
  - Erlang
  - Hyper-exponential

$$StartH_2 \stackrel{def}{=} (branch, \alpha \times fast).(phase1, \lambda_1).End + (branch, (1-\alpha) \times fast).(phase2, \lambda_2).End$$

  - Cox

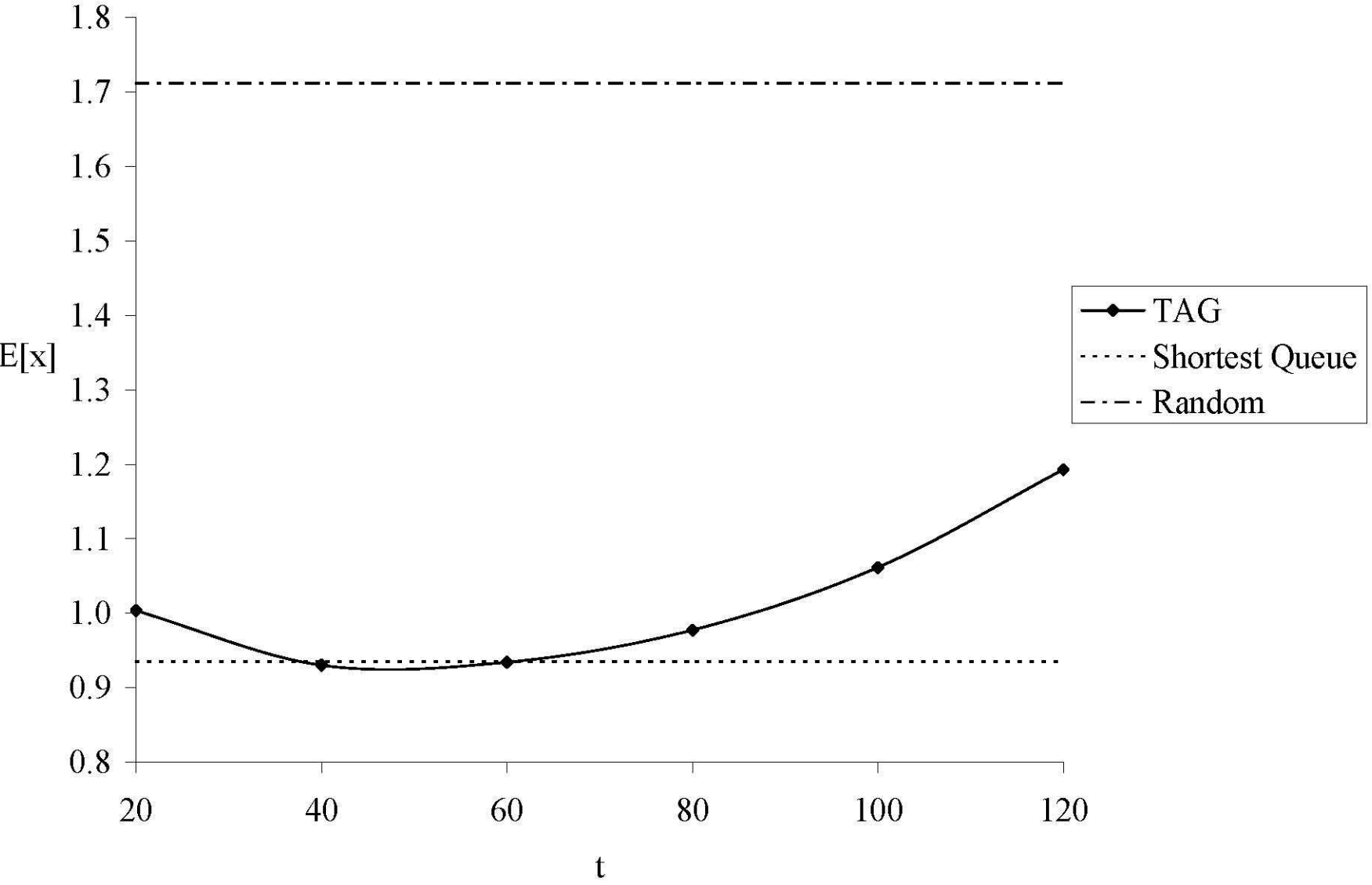
$$StartC_2 \stackrel{def}{=} (phase1, \alpha\lambda_1).End + (phase1, (1-\alpha)\lambda_1).(phase2, \lambda_2).End$$

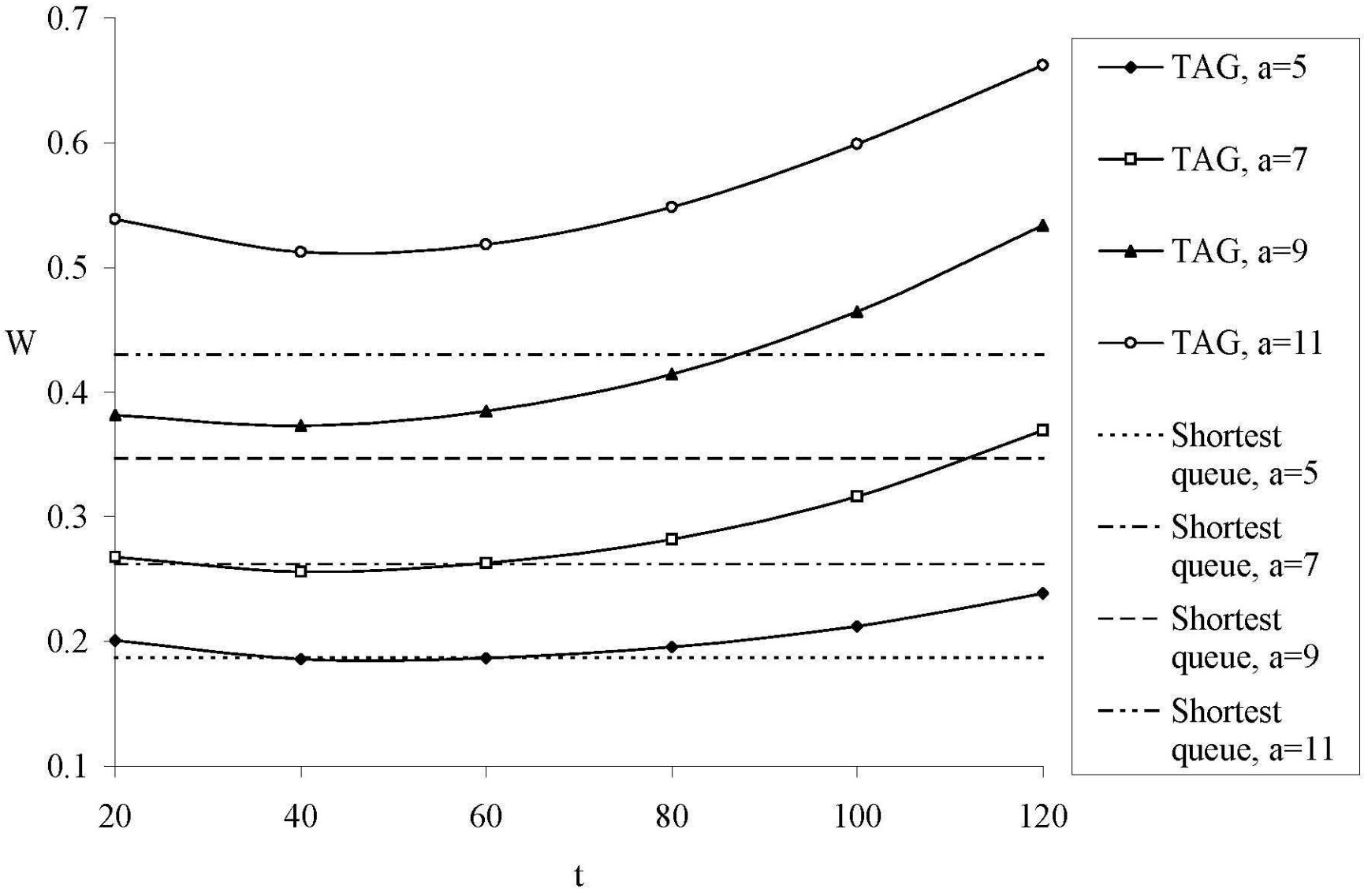
Adding Cox distributed service

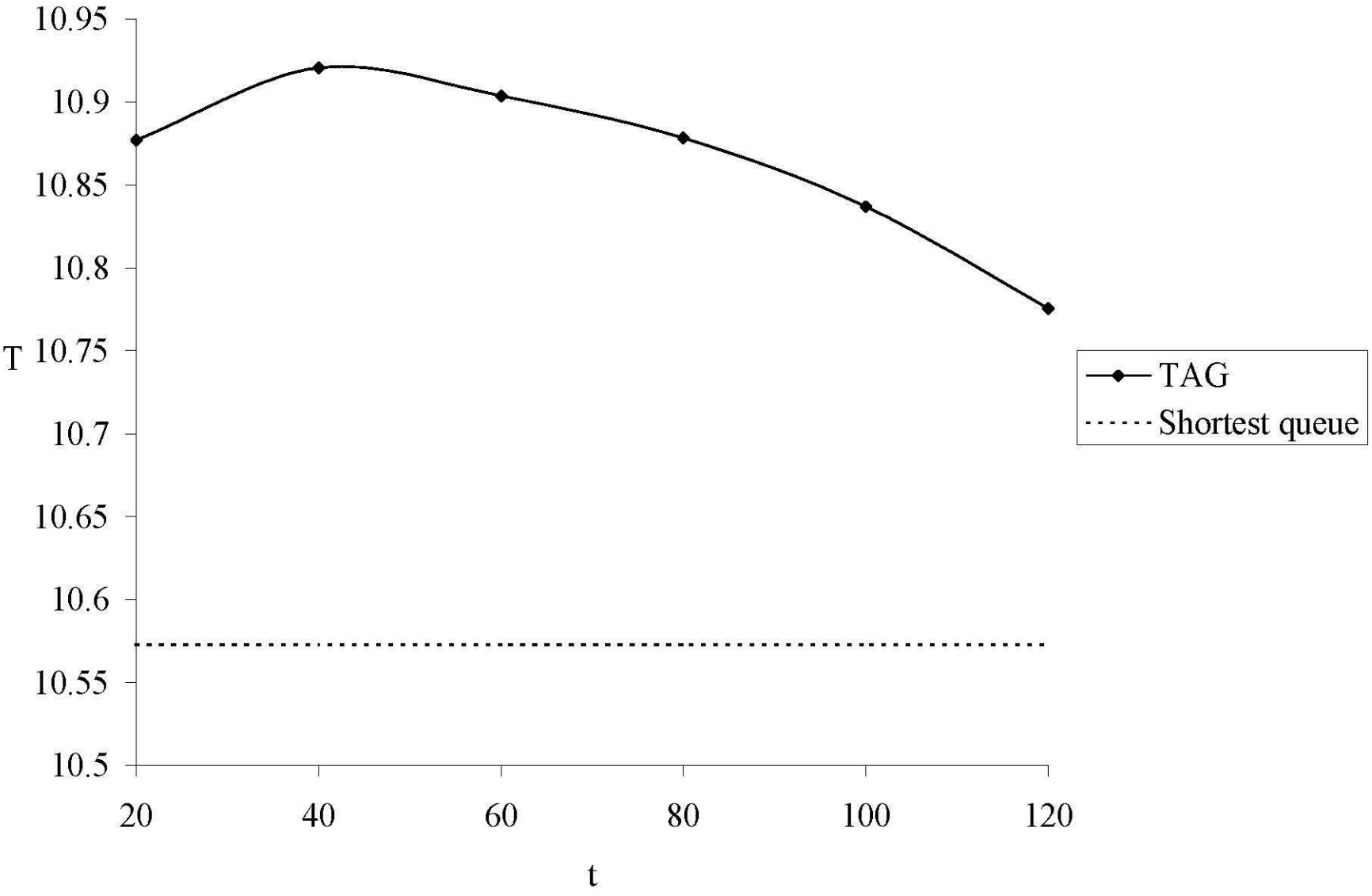
$$
\begin{aligned}
Server1 &\stackrel{def}{=} (service1, \alpha s_1).Server1 + (phase1, (1-\alpha)s_1).Server1' \\
&\quad + (timeout, \top).Server1 \\
Server1' &\stackrel{def}{=} (service1, s_2).Server1 + (timeout, \top).Server1 \\
Server2 &\stackrel{def}{=} (branch, p \times fast).(service2, s_2).Server2 \\
&\quad + (branch, (1-p) \times fast).Server2' \\
Server2' &\stackrel{def}{=} (service2, \alpha s_1).Server2 \\
&\quad + (phaseone, (1-\alpha)s_1).(service2, s_2).Server2
\end{aligned}
$$

- Actions *phase1* and *branch* must be added to the cooperation sets and queue components to prevent them firing when the queue is empty.
- The *timeout* action is used to reset the service at node 1 if it is interrupted.
- *p* is the result of the race between *timeout* and *phase1* conditional on *service1* having not occurred.

More results (1/3)




0.7
0.6
0.5
0.4
0.3
0.2
0.1
W
20
40
60
80
100
120
t
TAG, a=5
TAG, a=7
TAG, a=9
TAG, a=11
Shortest queue, a=5
Shortest queue, a=7
Shortest queue, a=9
Shortest queue, a=11


10.95
10.9
10.85
10.8
10.75
10.7
10.65
10.6
10.55
10.5
T
20
40
60
80
100
120
t
TAG
Shortest queue

Conclusions

- This paper has only scratched the surface of this problem, much remains to be done.
- It has been shown that TAG has some potential benefit for finite systems.
- More experimentation is required with the Cox distribution, particularly in varying α.
- A simulator has been produced (by Alex Liu) which will be used for comparison and to assess the degree of error introduced by the approximations.

Thanks for your time!