

# Extracting response times from fluid analysis of performance models

J. T. Bradley

R. Hayden

W. J. Knottenbelt

T. Suto

Department of Computer Science  
Imperial College London

SIPEW 2008, Darmstadt

# Response time suffers from state space explosion

- Quantitative analysis of response times important for many industrial systems e.g. “95 % of broadband connections established within 2s”
- Extraction of response times from underlying CTMC relies on uniformisation and then transient analysis
- Requires expansion of the underlying (CTMC's) state-space
- But even simple models suffer from *state space explosion*

# Can we adapt existing fluid analysis techniques?

- For example, continuous state space techniques for SPA using ODEs
  - Fairly well-developed
  - Generally quite accurate
  - Very cheap
- Allow approximation of transient expected 'component counts'
- Can we adapt them to extract response time measures inexpensively?

# PEPA

**Prefix**  $(\alpha, r).P$  — action  $\alpha$  happens at rate  $r$  before transitioning to component  $P$

**Choice**  $P + Q$  — allows either  $P$  or  $Q$  to occur, choice determined by races

**Cooperation**  $P \bowtie_L Q$  — For  $\alpha \in L$ , if  $P$  and  $Q$  both enable an  $\alpha$ -activity, only then can both make the  $\alpha$ -transition simultaneously at the rate of the slowest

**Hiding**  $P/L$  — Action types in set  $L$  become the hidden  $\tau$  action, which cannot be cooperated over

**Constant**  $A \stackrel{\text{def}}{=} P$  — Assigns the label  $A$  to component  $P$ , allows  $P \stackrel{\text{def}}{=} (\alpha, r).P$  etc.

# PEPA

**Prefix**  $(\alpha, r).P$  — action  $\alpha$  happens at rate  $r$  before transitioning to component  $P$

**Choice**  $P + Q$  — allows either  $P$  or  $Q$  to occur, choice determined by races

**Cooperation**  $P \bowtie_L Q$  — For  $\alpha \in L$ , if  $P$  and  $Q$  both enable an  $\alpha$ -activity, only then can both make the  $\alpha$ -transition simultaneously at the rate of the slowest

**Hiding**  $P/L$  — Action types in set  $L$  become the hidden  $\tau$  action, which cannot be cooperated over

**Constant**  $A \stackrel{\text{def}}{=} P$  — Assigns the label  $A$  to component  $P$ , allows  $P \stackrel{\text{def}}{=} (\alpha, r).P$  etc.

# PEPA

**Prefix**  $(\alpha, r).P$  — action  $\alpha$  happens at rate  $r$  before transitioning to component  $P$

**Choice**  $P + Q$  — allows either  $P$  or  $Q$  to occur, choice determined by races

**Cooperation**  $P \bowtie_L Q$  — For  $\alpha \in L$ , if  $P$  and  $Q$  both enable an  $\alpha$ -activity, only then can both make the  $\alpha$ -transition simultaneously at the rate of the slowest

**Hiding**  $P/L$  — Action types in set  $L$  become the hidden  $\tau$  action, which cannot be cooperated over

**Constant**  $A \stackrel{\text{def}}{=} P$  — Assigns the label  $A$  to component  $P$ , allows  $P \stackrel{\text{def}}{=} (\alpha, r).P$  etc.

# PEPA

**Prefix**  $(\alpha, r).P$  — action  $\alpha$  happens at rate  $r$  before transitioning to component  $P$

**Choice**  $P + Q$  — allows either  $P$  or  $Q$  to occur, choice determined by races

**Cooperation**  $P \bowtie_L Q$  — For  $\alpha \in L$ , if  $P$  and  $Q$  both enable an  $\alpha$ -activity, only then can both make the  $\alpha$ -transition simultaneously at the rate of the slowest

**Hiding**  $P/L$  — Action types in set  $L$  become the hidden  $\tau$  action, which cannot be cooperated over

**Constant**  $A \stackrel{def}{=} P$  — Assigns the label  $A$  to component  $P$ , allows  $P \stackrel{def}{=} (\alpha, r).P$  etc.

# PEPA

**Prefix**  $(\alpha, r).P$  — action  $\alpha$  happens at rate  $r$  before transitioning to component  $P$

**Choice**  $P + Q$  — allows either  $P$  or  $Q$  to occur, choice determined by races

**Cooperation**  $P \bowtie_L Q$  — For  $\alpha \in L$ , if  $P$  and  $Q$  both enable an  $\alpha$ -activity, only then can both make the  $\alpha$ -transition simultaneously at the rate of the slowest

**Hiding**  $P/L$  — Action types in set  $L$  become the hidden  $\tau$  action, which cannot be cooperated over

**Constant**  $A \stackrel{\text{def}}{=} P$  — Assigns the label  $A$  to component  $P$ , allows  $P \stackrel{\text{def}}{=} (\alpha, r).P$  etc.

# PEPA

**Prefix**  $(\alpha, r).P$  — action  $\alpha$  happens at rate  $r$  before transitioning to component  $P$

**Choice**  $P + Q$  — allows either  $P$  or  $Q$  to occur, choice determined by races

**Cooperation**  $P \bowtie_L Q$  — For  $\alpha \in L$ , if  $P$  and  $Q$  both enable an  $\alpha$ -activity, only then can both make the  $\alpha$ -transition simultaneously at the rate of the slowest

**Hiding**  $P/L$  — Action types in set  $L$  become the hidden  $\tau$  action, which cannot be cooperated over

**Constant**  $A \stackrel{def}{=} P$  — Assigns the label  $A$  to component  $P$ , allows  $P \stackrel{def}{=} (\alpha, r).P$  etc.

# A not-so-simple SPA model (in PEPA)

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \boxtimes_{\{\text{task}_1\}} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

$2^{N_p+N_r}$  states

# A not-so-simple SPA model (in PEPA)

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \boxtimes_{\{\text{task}_1\}} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

$2^{N_p+N_r}$  states

# A not-so-simple SPA model (in PEPA)

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \boxtimes_{\{\text{task}_1\}} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

$2^{N_p+N_r}$  states

# A not-so-simple SPA model (in PEPA)

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \boxtimes_{\{\text{task}_1\}} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

$2^{N_p+N_r}$  states

# Aggregation is not a scalable solution

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \underset{\{\text{task}_1\}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

Only  $(N_p + 1) \times (N_r + 1)$  states if just count components in each state ...

... but e.g. 20 processors and resources, each with 10 derivative states:

10015005<sup>2</sup> (aggregate) states

# Aggregation is not a scalable solution

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \underset{\{\text{task}_1\}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

Only  $(N_p + 1) \times (N_r + 1)$  states if just count components in each state ...

... but e.g. 20 processors and resources, **each with 10 derivative states**:

$10015005^2$  (aggregate) states

# Fluid analysis with ODEs

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \overset{\{ \text{task}_1 \}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

- $P_0 \rightarrow P_1$  and  $R_0 \rightarrow R_1$  at rate  $\min(r_1 \times n_{P_0}, r_1 \times n_{R_0}) = r_1 \times \min(n_{P_0}, n_{R_0})$
- $P_1 \rightarrow P_0$  at rate  $r_2 \times n_{P_1}$
- $R_1 \rightarrow R_0$  at rate  $s \times n_{R_1}$
- Or:  $\frac{dn_{P_0}(t)}{dt} = -r_1 \min(n_{P_0}(t), n_{R_0}(t)) + r_2 n_{P_1}(t)$  etc.

## Fluid analysis with ODEs

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \overset{\{ \text{task}_1 \}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

- $P_0 \rightarrow P_1$  and  $R_0 \rightarrow R_1$  at rate  $\min(r_1 \times n_{P_0}, r_1 \times n_{R_0}) = r_1 \times \min(n_{P_0}, n_{R_0})$
- $P_1 \rightarrow P_0$  at rate  $r_2 \times n_{P_1}$
- $R_1 \rightarrow R_0$  at rate  $s \times n_{R_1}$
- Or:  $\frac{dn_{P_0}(t)}{dt} = -r_1 \min(n_{P_0}(t), n_{R_0}(t)) + r_2 n_{P_1}(t)$  etc.

# Fluid analysis with ODEs

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \underset{\{\text{task}_1\}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

- $P_0 \rightarrow P_1$  and  $R_0 \rightarrow R_1$  at rate  $\min(r_1 \times n_{P_0}, r_1 \times n_{R_0}) = r_1 \times \min(n_{P_0}, n_{R_0})$
- $P_1 \rightarrow P_0$  at rate  $r_2 \times n_{P_1}$
- $R_1 \rightarrow R_0$  at rate  $s \times n_{R_1}$
- Or:  $\frac{dn_{P_0}(t)}{dt} = -r_1 \min(n_{P_0}(t), n_{R_0}(t)) + r_2 n_{P_1}(t)$  etc.

# Fluid analysis with ODEs

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \overset{\{\text{task}_1\}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

- $P_0 \rightarrow P_1$  and  $R_0 \rightarrow R_1$  at rate  $\min(r_1 \times n_{P_0}, r_1 \times n_{R_0}) = r_1 \times \min(n_{P_0}, n_{R_0})$
- $P_1 \rightarrow P_0$  at rate  $r_2 \times n_{P_1}$
- $R_1 \rightarrow R_0$  at rate  $s \times n_{R_1}$
- Or:  $\frac{dn_{P_0}(t)}{dt} = -r_1 \min(n_{P_0}(t), n_{R_0}(t)) + r_2 n_{P_1}(t)$  etc.

# Fluid analysis with ODEs

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0$$

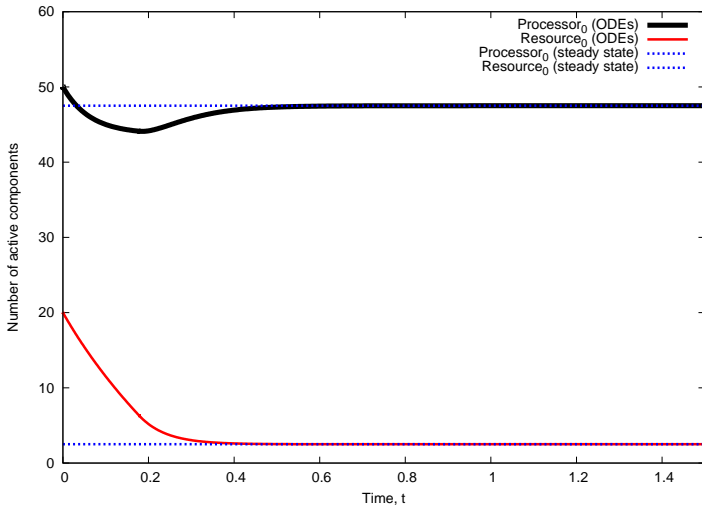
$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\underbrace{(P_0 \parallel \dots \parallel P_0)}_{n_{P_0}} \parallel \underbrace{(P_1 \parallel \dots \parallel P_1)}_{n_{P_1}} \overset{\{\text{task}_1\}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{n_{R_0}} \parallel \underbrace{(R_1 \parallel \dots \parallel R_1)}_{n_{R_1}}$$

- $P_0 \rightarrow P_1$  and  $R_0 \rightarrow R_1$  at rate  $\min(r_1 \times n_{P_0}, r_1 \times n_{R_0}) = r_1 \times \min(n_{P_0}, n_{R_0})$
- $P_1 \rightarrow P_0$  at rate  $r_2 \times n_{P_1}$
- $R_1 \rightarrow R_0$  at rate  $s \times n_{R_1}$
- Or:  $\frac{dn_{P_0}(t)}{dt} = -r_1 \min(n_{P_0}(t), n_{R_0}(t)) + r_2 n_{P_1}(t)$  etc.

# Fluid analysis with ODEs (example)



# Response time in a CTMC

Loosely, the response time from a source state to a set of given target states is the random variable:

*“the time taken to reach one of the target states from the source state”*

For example:

- The time taken to establish a broadband connection;
- In the context of our previous example, the time taken for all of the processors to complete their two tasks

# Response time as time to extinction

$$\begin{aligned} P_0 &\stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1 & R_0 &\stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1 \\ P_1 &\stackrel{\text{def}}{=} (\text{task}_2, r_2).P_0 & R_1 &\stackrel{\text{def}}{=} (\text{reset}, s).R_0 \end{aligned}$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \boxtimes_{\{\text{task}_1\}} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

response time for  $N_p$  processors to complete =  
time to extinction of all  $P_0$  and  $P_1$  components

# Response time as time to extinction

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).\text{Stop}$$

$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \boxtimes_{\{\text{task}_1\}} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

response time for  $N_p$  processors to complete =  
time to extinction of all  $P_0$  and  $P_1$  components

# Response time as time to extinction

$$P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1$$

$$P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).\text{Stop}$$

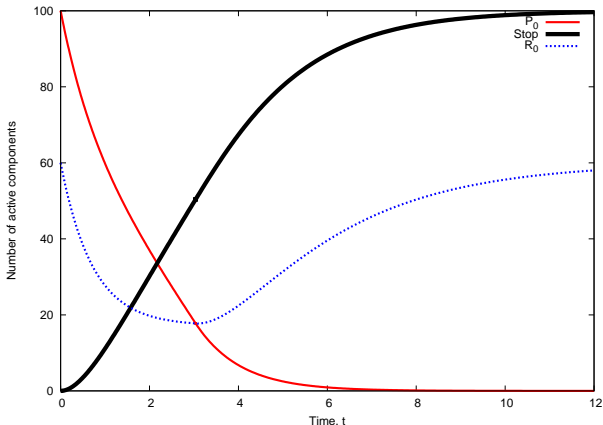
$$R_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).R_1$$

$$R_1 \stackrel{\text{def}}{=} (\text{reset}, s).R_0$$

$$\text{System} \stackrel{\text{def}}{=} \underbrace{(P_0 \parallel \dots \parallel P_0)}_{N_p} \underset{\{\text{task}_1\}}{\boxtimes} \underbrace{(R_0 \parallel \dots \parallel R_0)}_{N_r}$$

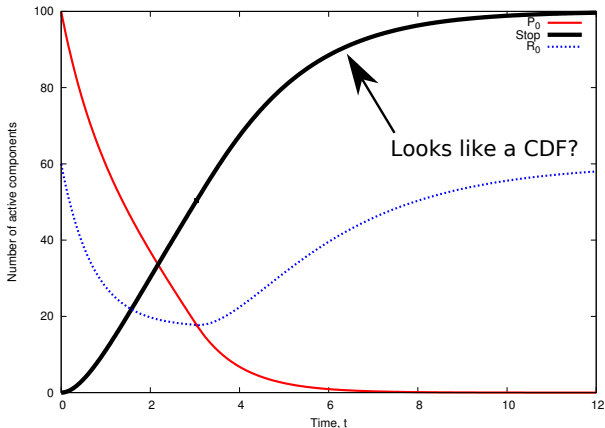
response time for  $N_p$  processors to complete =  
time to extinction of all  $P_0$  and  $P_1$  components

# Response time as time to extinction (example)



But where do we say  $P_0 + P_1$  has reached 0 (i.e. equiv. **Stop** has reached  $N_p (= 100)$ )?

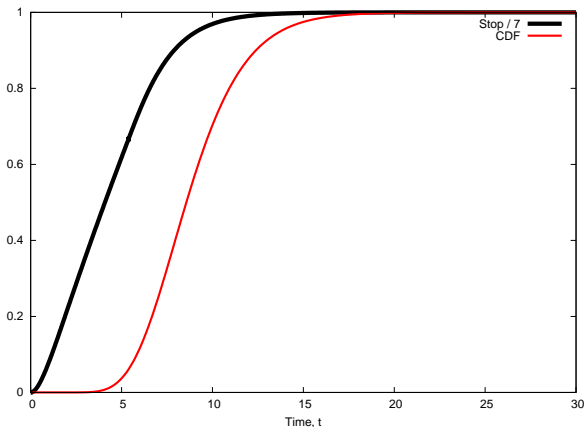
# Response time as time to extinction (example)



But where do we say  $P_0 + P_1$  has reached 0 (i.e. equiv. **Stop** has reached  $N_p (= 100)$ )?

# Response time as time to extinction (example)

Scale by  $N_p (= 7)$  and compare to actual CDF:



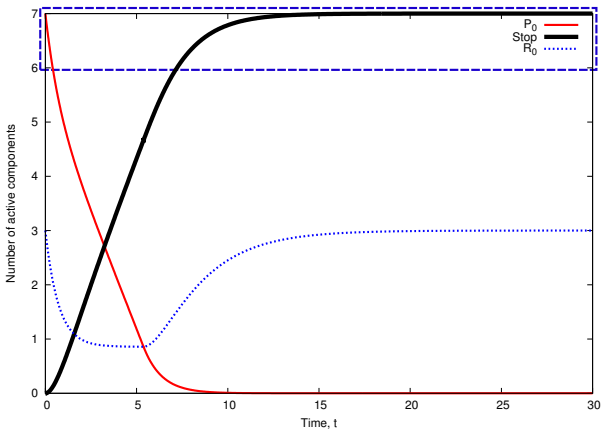
# Fluid response time lower bound

We wouldn't really expect this to be a good approximation in general . . .

- The median response time is unlikely to be close to the expected time for (only) half of the processors to be absorbed
- In fact, we can show it is always a *lower bound modulo errors in the ODE approximation*
- Not particularly useful!
- **Better idea (?)**: look at the absorption period for just the last component

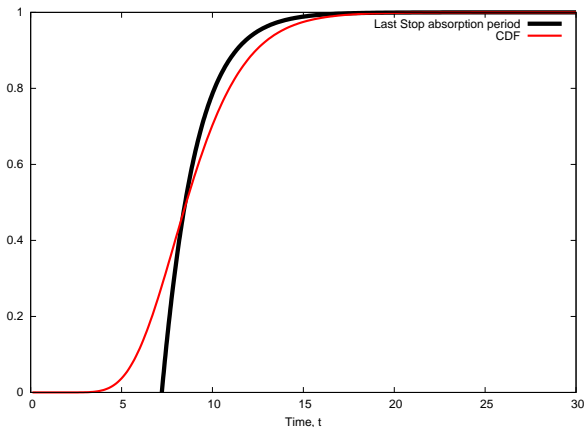
# Fluid response time upper bound

Consider just the absorption period for the last component:



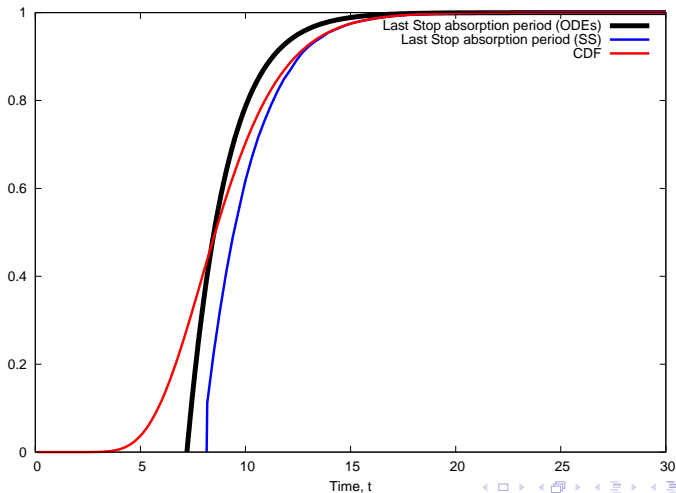
# Fluid response time upper bound

Scale by  $N_p$  and compare to CDF:



# Fluid response time upper bound

Factor out the error from the ODE approximation:



# Fluid response time upper bound

- We can show that the last absorption period is always an *upper bound modulo errors in the ODE approximation*
- Other types of (more useful) response time measures can be dealt with similarly, e.g. regenerating processors, by component duplication:

$$\begin{array}{ll} P_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P_1 & P'_0 \stackrel{\text{def}}{=} (\text{task}_1, r_1).P'_1 \\ P_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P'_0 & P'_1 \stackrel{\text{def}}{=} (\text{task}_2, r_2).P'_0 \end{array}$$

- One then looks at fluid absorption of the  $P'_0$  and  $P'_1$  components instead of **Stop**

## Case study: hospital model

- More detailed model of a hospital A & E department
- Interested in response time measure for all patients to pass through
- 5 patients, 3 nurses and 2 doctors
  - 95% quantile: CTMC: 5.45 hours, fluid upper bound: 5.48 hours
- **100 patients**, 3 nurses and 2 doctors  $> 10^{100}$  states
  - 95% quantile: fluid upper bound: 15.1 hours

# Conclusions

- Intuitively can think of certain response times as times to component extinction, which can be approximated (in some sense) using fluid analysis techniques
- We have presented two possible senses, provably lower and upper bounds resp., up to errors in the ODE approximation
- Accuracy of these techniques is thus dependent on the accuracy of the original ODE approximation to the transient expected component counts

## Future work

- More often than not, response time measures are specified in terms of actions
- For example, stochastic probes
- In many scenarios these can be bounded above by component extinction-specified response times
- For less sharply-peaked distributions of the component count, might be able to tighten bound by considering absorption period for the last  $n > 1$  components

# Questions?

# Bounding proofs

Let  $X$  be the response time r.v. of interest and assume  $\mathbb{P}\{X \leq t\} = p$ . Then:

$$\begin{aligned}\mathbb{E}[P_0(t)] &\geq 0 \times (1 - p) + N_p \times p \\ &= N_p \times p\end{aligned}$$

Furthermore:

$$\begin{aligned}\mathbb{E}[P_0(t)] &\leq (N_p - 1) \times (1 - p) + N_p \times p \\ &= (N_p - 1) + p\end{aligned}$$