Scalable Performance Analysis of Massively Parallel Stochastic Systems

Richard A. Hayden

rh@doc.ic.ac.uk

Department of Computing, Imperial College London, UK

November 22, 2012
Abstract client/server model

- Client
- Server

Population CTMC (aggregated) state space:

\[
\begin{align*}
X_C(t), \quad X_Cw(t), \quad X_Cp(t), \quad X_S(t), \quad X_Sp(t), \quad X_Sf(t) & \in \mathbb{Z}_6^+ \\
\end{align*}
\]

- State space grows exponentially as component types are added
- Explicit-state analysis techniques do not scale
- Simulation is also very costly for large populations
Abstract client/server model

Population CTMC (aggregated) state space:

\[
\left(X_{C}(t), X_{Cw}(t), X_{Cp}(t), X_{S}(t), X_{Sp}(t), X_{Sf}(t)\right) \in \mathbb{Z}^6^+
\]

- State space grows exponentially as component types are added
- Explicit-state analysis techniques do not scale
- Simulation is also very costly for large populations
Abstract client/server model

Population CTMC (aggregated) state space:

\[
(X_C(t), X_C_{\text{wait}}(t), X_C_{\text{proc}}(t), X_S(t), X_S_{\text{proc}}(t), X_S_{\text{fail}}(t)) \in \mathbb{Z}^6_+
\]

- State space grows exponentially as component types are added
- Explicit-state analysis techniques do not scale
- Simulation is also very costly for large populations
Abstract client/server model

Client

\[ \text{Client} \]

\[ \text{Client}_{\text{wait}} \]

\[ \text{Client}_{\text{proc}} \]

\[ \text{proc} \]

\[ \text{req} \]

\[ \text{res} \]

Server

\[ \text{Server} \]

\[ \text{Server}_{\text{proc}} \]

\[ \text{req} \]

\[ \text{res} \]

\[ \text{fail} \]

\[ \text{reset} \]

Population CTMC (aggregated) state space:

\[ (X_{\text{C}}(t), X_{\text{C}}_{\text{w}}(t), X_{\text{C}}_{\text{p}}(t), X_{\text{S}}(t), X_{\text{S}}_{\text{p}}(t), X_{\text{S}}_{\text{f}}(t))^T \in \mathbb{Z}^6_+ \]

- State space grows exponentially as component types are added.
- Explicit-state analysis techniques do not scale.
- Simulation is also very costly for large populations.
Abstract client/server model

Population CTMC (aggregated) state space:
\[
(X_C(t), X_C(w)(t), X_C(p)(t), X_S(t), X_S(p)(t), X_S(f)(t)) \in \mathbb{Z}_{+}^6
\]

State space grows exponentially as component types are added

Explicit-state analysis techniques do not scale

Simulation is also very costly for large populations
Abstract client/server model

Population CTMC (aggregated) state space:

\[
(X_C(t), X_{C\text{wait}}(t), X_{C\text{proc}}(t), X_S(t), X_{S\text{proc}}(t), X_{S\text{fail}}(t)) \in \mathbb{Z}^6_+
\]
Abstract client/server model

Population CTMC (aggregated) state space:
\[(X_{C}(t), X_{Cw}(t), X_{Cp}(t), X_{S}(t), X_{Sp}(t), X_{Sf}(t)) \in \mathbb{Z}^{6}_{+}\]

- State space grows exponentially as component types are added.
- Explicit-state analysis techniques do not scale.
- Simulation is also very costly for large populations.
Abstract client/server model

\[ \text{proc} \circ r_p \]

\[ \text{Client} \]

\[ \text{req} \]

\[ \text{Client}_{\text{wait}} \]

\[ \text{res} \]

\[ \text{Client}_{\text{proc}} \]

\[ N_C \]

\[ \text{Server} \]

\[ \text{req} \]

\[ \text{Server}_{\text{proc}} \]

\[ N_S \]

\[ \text{fail} \circ r_f \]

\[ \text{reset} \circ r_{\text{rst}} \]
Abstract client/server model

\[ (X_C(t), X_{Cw}(t), X_{Cp}(t), X_S(t), X_{Sp}(t), X_{Sf}(t)) \in \mathbb{Z}^6_+ \]

- State space grows exponentially as component types are added
- Explicit-state analysis techniques do not scale
- Simulation is also very costly for large populations

\[ \text{proc} \circ \circ \tau_p \]

\[ \text{Client} \]

\[ \text{Client}_{\text{wait}} \]

\[ \text{Client}_{\text{proc}} \]

\[ \text{proc} \circ \circ \tau \]

\[ \text{Server}_{\text{fail}} \]

\[ \text{Server}_{\text{proc}} \]

\[ \text{reset} \circ \circ \tau_{\text{rst}} \]

\[ \text{fail} \circ \circ \tau_f \]
Markovian dynamics depend on the aggregate rate of req/res-synchronisations, for example, for req:

\[ r_{rq \min}(X_C(t), X_S(t)) \]

Petri nets, queueing nets

\[ (r_{rq}/N_S)X_C(t)X_S(t) \]

mass-action, e.g. peer-to-peer nets
Abstract client/server model

Markovian dynamics depend on the aggregate rate of req/res-synchronisations, for example, for req:

\[ r_{rq} \min(X_C(t), X_S(t)) \] Petri nets, queueing nets

\[ (r_{rq}/N_S)X_C(t)X_S(t) \] mass-action, e.g. peer-to-peer nets
Abstract client/server model

- Population CTMC (aggregated) state space:

\[(X_C(t), X_{Cw}(t), X_{Cp}(t), X_S(t), X_{Sp}(t), X_{Sf}(t)) \in \mathbb{Z}_+^6\]
Abstract client/server model

- Population CTMC (aggregated) state space:

\[(X_C(t), X_{Cw}(t), X_{Cp}(t), X_S(t), X_{Sp}(t), X_{Sf}(t)) \in \mathbb{Z}_+^6\]

- State space grows exponentially as component types are added
Abstract client/server model

Population CTMC (aggregated) state space:

\[(X_C(t), X_{Cw}(t), X_{Cp}(t), X_S(t), X_{Sp}(t), X_{Sf}(t)) \in \mathbb{Z}_+^6\]

- State space grows exponentially as component types are added
- Explicit-state analysis techniques do not scale
Abstract client/server model

Population CTMC (aggregated) state space:

\[(X_C(t), X_{C_w}(t), X_{C_p}(t), X_S(t), X_{S_p}(t), X_{S_f}(t)) \in \mathbb{Z}_+^6\]

- State space grows exponentially as component types are added
- Explicit-state analysis techniques do not scale
- Simulation is also very costly for large populations
Mean-field/fluid analysis

- Alleviates the state-space explosion problem for Markov models of computer and communication systems

- Derives tractable systems of differential equations approximating statistics of number of components in each local state\(^{[1–3]}\)


Mean-field/fluid analysis

- Alleviates the state-space explosion problem for Markov models of computer and communication systems
- Derives tractable systems of differential equations approximating statistics of number of components in each local state
- These approaches are closely related to classical heavy-traffic analysis of queues\[^4\] and analysis of large-scale models in chemistry and biology\[^5\]


Mean-field/fluid analysis

- Alleviates the state-space explosion problem for Markov models of computer and communication systems

- Derives tractable systems of differential equations approximating statistics of number of components in each local state

- These approaches are closely related to classical heavy-traffic analysis of queues and analysis of large-scale models in chemistry and biology

- Key point: size of approximating system is independent of population size ⇒ scalability
Overview

Massively-parallel Markov models
- Population CTMCs
- e.g. stochastic Petri nets, queueing networks
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations
Overview

Massively-parallel Markov models

- **Population CTMCs**
- e.g. stochastic Petri nets, queueing networks

Differential equations

Passage times
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations

Passage times

SLA: \( \leq 6.5s \) w.p. \( \geq 90\% \)

Time, \( t \)

Probability

Faster

Slower

Rewards

Energy Consumption

Optimal SLA satisfaction

Both SLAs

Energy consumption
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations

- Passage times
- Rewards
Overview

Massively-parallel Markov models

▶ Population CTMCs
▶ e.g. stochastic Petri nets, queueing networks

Differential equations

Passage times

Rewards

![Graph showing energy consumption over time for faster and slower scenarios.](image-url)
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations

- Passage times
- Rewards

Optimal SLA satisfaction
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations

Passage times

Rewards

Optimal SLA satisfaction
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations

- Passage times
- Rewards

Optimal SLA satisfaction

![Graph showing energy consumption vs. number of slow and fast servers, with SLA H and SLA L regions marked.]
Overview

Massively-parallel Markov models

- Population CTMCs
- e.g. stochastic Petri nets, queueing networks

Differential equations

Passage times

Rewards

Optimal SLA satisfaction

SLA: \[ \leq 6.5 \text{ s w.p. } \geq 90\% \]

Energy consumption

Both SLAs

N\text{slow} N\text{fast}

N\text{slow} N\text{fast}
Client/server model fluid limit

\[ \text{Client} \]
proc \( r_p \)

\[ \text{Client}_\text{wait} \]
\( \text{req} \) \( \rightarrow \)
\( \text{res} \)

\[ \text{Client}_\text{proc} \]
\( \text{res} \) \( \rightarrow \)

\[ \text{Server} \]
\( \text{req} \) \( \rightarrow \)
\( \text{res} \)

\[ \text{Server}_{\text{fail}} \]
\( \text{reset} \circledast \ r_{rst} \)

\[ \text{reset} \circledast \ r_{rst} \)

\[ \text{fail} \circledast \ r_f \)

\[ \text{N}_C \]

\[ \text{N}_S \]

\[ r_q \min(X_C(t), X_S(t)) \]
\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]
Decouple the component count states (limiting independence assumption)
Decouple the component count states (limiting independence assumption)

Construct differential equations by balancing the aggregate rates of Markovian transitions
Decouple the component count states (limiting independence assumption)

Construct differential equations by balancing the aggregate rates of Markovian transitions

Deterministic approximations:

\[(\nu_C(t), \nu_{Cw}(t), \nu_{ Cp}(t), \nu_S(t), \nu_{Sp}(t), \nu_{Sf}(t)) \in \mathbb{R}^6_+\]
Client/server model fluid limit

\[
\frac{dv_C(t)}{dt} = \]

\[
\frac{dv_S(t)}{dt} = \]

\[
\text{Server}_{\text{fail}} \rightarrow \text{Server}_{\text{proc}} \]
Client/server model fluid limit

\[
\frac{dv_C(t)}{dt} = \frac{r_p v_{cp}(t)}{\text{aggregate rate}} - \frac{r_{rq} \min(v_C(t), v_S(t))}{\text{aggregate rate}}
\]

\[
\frac{dv_S(t)}{dt} = 
\]
Client/server model fluid limit

\[
\frac{dv_C(t)}{dt} = r_p v_{Cp}(t) - \left( r_{rq} \min(v_C(t), v_S(t)) \right)
\]

aggregate rate

\[Client_{proc} \rightarrow Client\]

\[
\frac{dv_S(t)}{dt} = r_{rst} v_{Sf}(t) + r_s \min(v_{Cw}(t), v_{Sp}(t)) - \left( r_{rq} \min(v_C(t), v_S(t)) + r_f v_S(t) \right)
\]

aggregate rate

\[Server_{fail} \rightarrow Server, Server_{proc} \rightarrow Server\]

\[Server \rightarrow Server_{proc}, Server \rightarrow Server_{fail}\]
Client/server model fluid limit

\[ \text{proc} @ r_p \]

\[ \text{Client}_{\text{wait}} \]

\[ \text{req} \]

\[ \text{Client}_{\text{proc}} \]

\[ \text{res} \]

\[ r_{rq} \min(X_C(t), X_S(t)) \]

\[ r_{rs} \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 10, \quad N_S = 6 \]

Graph showing the client/server model with fluid limit, including the following transitions:
- Client to Client\_wait
- Client\_wait to Client\_proc
- Server to Server\_proc
- Server\_proc to Server

The graphs illustrate the rescaled component count over time for both the client and server processes, with \( N_C = 10 \) and \( N_S = 6 \).
Client/server model fluid limit

\[ \text{proc} \@ r_p \]

\[ \begin{align*}
\text{Client} & \xrightarrow{\text{req}} \text{Client}_{\text{wait}} \\
\text{Client}_{\text{wait}} & \xrightarrow{\text{res}} \text{Client}_{\text{proc}}
\end{align*} \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{C_w}(t), X_{S_p}(t)) \]

\[ N_C = 50, N_S = 30 \]

\[ N_C = 50, N_S = 30 \]
Client/server model fluid limit

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]
\[ \text{propto} @ r_p \]
\[ \text{Server} \rightarrow \text{Server}_{\text{proc}} \rightarrow \text{Server}_{\text{fail}} \]
\[ \text{reset} @ r_{\text{rst}} \rightarrow \text{fail} @ r_f \]

\[ r_q \min(X_C(t), X_S(t)) \]
\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 100, N_S = 60 \]
Client/server model fluid limit

\[ N_C = 200, \quad N_S = 120 \]
Client/server model fluid limit

\[ \text{Client}_\text{wait} \xrightarrow{\text{proc} \ @ \ r_p} \ \text{Client}_\text{proc} \]

\[ \text{Server} \xrightarrow{\text{req} \ @ \ r_{req}} \ \text{Server}_\text{proc} \]

\[ \text{fail} \xrightarrow{\text{reset} \ @ \ r_{rst}} \ \text{Server}_\text{proc} \]

\[ r_{req} \min(X_C(t), X_S(t)) \]

\[ r_{rs} \min(X_{C_{\text{w}}}(t), X_{S_{\text{p}}}(t)) \]

\[ N_C = 500, N_S = 300 \]
Client/server model fluid limit

\[ \text{Server}_{\text{proc}} \]

\[ \text{Server}_{\text{fail}} \]

\[ \text{Client}_{\text{proc}} \]

\[ \text{Client}_{\text{wait}} \]

\[ \text{proc} \oplus r_p \]

\[ \text{req} \rightarrow \text{res} \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{C_w}(t), X_{S_p}(t)) \]

\[ N_C = 1000, N_S = 600 \]

\[ N_C \]

\[ N_S \]
Client/server model fluid limit

\[ \text{proc} @ r_p \]

\[ \text{Server}_{\text{fail}} \]

\[ \text{reset} @ r_{\text{rst}} \]

\[ \text{fail} @ r_f \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 5000, N_S = 3000 \]
Client/server model fluid limit

\[ N_C = 10000, \quad N_S = 6000 \]
Functional law of large numbers

Population CTMC (PCTMC) model:
- Consists of interacting objects with finite *local state spaces* $S$

Theorem: PCTMC FLLN

For $T, \epsilon > 0$

$$\lim_{N \to \infty} P\{\sup_{t \leq T} \|\frac{1}{N} X_N(t) - v(t)\| > \epsilon\} = 0$$
Functional law of large numbers

*Population CTMC (PCTMC) model:*

- Consists of interacting objects with finite *local state spaces* $S$
- *Global state space* $\mathcal{X} \subset \mathbb{Z}^{|S|}_+$, initial condition $x_0 \in \mathcal{X}$
Functional law of large numbers

Population CTMC (PCTMC) model:

- Consists of interacting objects with finite local state spaces $S$
- Global state space $\mathcal{X} \subset \mathbb{Z}^{|S|}_+$, initial condition $x_0 \in \mathcal{X}$
- Finite set of transitions $c \in \mathcal{C}$, $c = (L_c, r_c)$ for multiset $L_c \subset S \times S$ and $r_c \in \mathcal{X} \rightarrow \mathbb{R}_+$, where:
  
  $L_{c,s} := |\{(s', s) \in L_c\}| - |\{(s, s') \in L_c\}|$
Functional law of large numbers

Population CTMC (PCTMC) model:

- Consists of interacting objects with finite local state spaces $S$
- **Global state space** $X \subset \mathbb{Z}_+^{|S|}$, initial condition $x_0 \in X$
- Finite set of **transitions** $c \in C$, $c = (L_c, r_c)$ for multiset $L_c \subset S \times S$ and $r_c \in X \rightarrow \mathbb{R}_+$, where:
  $$L_{c,s} := \left| \left\{ (s', s) \in L_c \right\} \right| - \left| \left\{ (s, s') \in L_c \right\} \right|$$
- Sequence of such models indexed by $N$ with initial conditions $Nx_0$ and rate functions $r_c^N$ s.t. $r_c^N(x) \leq R(\|x\| + 1)$ for $R \in \mathbb{R}_+$ and $r_c(x) = (1/N)r_c^N(Nx)$ for locally Lipschitz $r_c$
Functional law of large numbers

Population CTMC (PCTMC) model:

- Consists of interacting objects with finite local state spaces \( S \)
- Global state space \( \mathcal{X} \subset \mathbb{Z}_+^{\lvert S \rvert} \), initial condition \( x_0 \in \mathcal{X} \)
- Finite set of transitions \( c \in \mathcal{C} \), \( c = (\mathcal{L}_c, r_c) \) for multiset \( \mathcal{L}_c \subset S \times S \) and \( r_c \in \mathcal{X} \rightarrow \mathbb{R}_+ \), where:
  \[
  L_{c,s} := \lvert \{(s', s) \in L_c \} \rvert - \lvert \{(s, s') \in L_c \} \rvert
  \]
- Sequence of such models indexed by \( N \) with initial conditions \( N x_0 \) and rate functions \( r_c^N \) s.t. \( r_c^N(x) \leq R(\|x\| + 1) \) for \( R \in \mathbb{R}_+ \) and \( r_c(x) = (1/N)r_c^N(Nx) \) for locally Lipschitz \( r_c \)
- Counting process \( X^N(t) \in \mathbb{Z}_+^{\lvert S \rvert} \) and ODE approximation:
  \[
  v(t) = x_0 + \int_0^t \sum_{c \in \mathcal{C}} \mathcal{L}_c r_c(v(s)) \, ds
  \]
Functional law of large numbers

**Population CTMC (PCTMC) model:**

- Consists of interacting objects with finite *local state spaces* $S$
- *Global state space* $X \subset \mathbb{Z}_+^{|S|}$, initial condition $x_0 \in X$
- Finite set of *transitions* $c \in C$, $c = (L_c, r_c)$ for multiset $L_c \subset S \times S$ and $r_c \in X \to \mathbb{R}_+$, where:
  $L_{c,s} := |\{(s', s) \in L_c\}| - |\{(s, s') \in L_c\}|
- Sequence of such models indexed by $N$ with initial conditions $N x_0$ and rate functions $r_c^N$ s.t. $r_c^N(x) \leq R(\|x\| + 1)$ for $R \in \mathbb{R}_+$ and $r_c(x) = (1/N) r_c^N(Nx)$ for locally Lipschitz $r_c$
- Counting process $X^N(t) \in \mathbb{Z}_+^{|S|}$ and ODE approximation:
  \[ v(t) = x_0 + \int_0^t \sum_{c \in C} L_c r_c(v(s)) \, ds \]

**Theorem: PCTMC FLLN**

For $T, \epsilon > 0$:

\[
\lim_{N \to \infty} \mathbb{P} \left\{ \sup_{t \leq T} \| (1/N)X^N(t) - v(t) \| > \epsilon \right\} = 0
\]

LLN in the stationary regime

- Convergence of PCTMC stationary measures $\mu^N$ to ODE fixed point $v^*$?

Theorem: PCTMC LLN in stationary regime

As $N \to \infty$, $\left(\frac{1}{N}\right)\mu^N \Rightarrow \delta v^*$ in distribution on $\mathbb{R}^{|S|}$ where $\delta v^*$ is the point mass at $v^*$. 
LLN in the stationary regime

- Convergence of PCTMC stationary measures $\mu^N$ to ODE fixed point $v^*$?
- Requires extra regularity conditions on the phase space of the differential equation

Theorem: PCTMC LLN in stationary regime
As $N \to \infty$, $\frac{1}{N} \mu^N \Rightarrow \delta_{v^*}$ in distribution on $\mathbb{R}^{|S|}$ where $\delta_{v^*}$ is the point mass at $v^*$.
LLN in the stationary regime

- Convergence of PCTMC stationary measures $\mu^N$ to ODE fixed point $v^*$?
- Requires extra regularity conditions on the phase space of the differential equation
- For example: if we have that $\lim_{t \to \infty} v(t) = v^*$ for all PCTMC initial conditions $x_0$, then:

\[
\frac{1}{N} \mu^N \Rightarrow \delta_{v^*}
\]
LLN in the stationary regime

- Convergence of PCTMC stationary measures $\mu^N$ to ODE fixed point $v^*$?
- Requires extra regularity conditions on the phase space of the differential equation
- For example: if we have that $\lim_{t \to \infty} v(t) = v^*$ for all PCTMC initial conditions $x_0$, then:

**Theorem: PCTMC LLN in stationary regime**

As $N \to \infty$, $(1/N)\mu^N \Rightarrow \delta_{v^*}$ in distribution on $\mathbb{R}^{|S|}$ where $\delta_{v^*}$ is the point mass at $v^*$


Client/server model first-moment approximation
Client/server model first-moment approximation

Alternatively, can interpret the same system of ODEs as approximations to the mean component counts:

\[
(E[X_C(t)], E[X_{Cw}(t)], E[X_{Cp}(t)], E[X_S(t)], E[X_{Sp}(t)], E[X_{Sf}(t)]) \in \mathbb{R}_+^6
\]
Client/server model first-moment approximation

Alternatively, can interpret the same system of ODEs as approximations to the mean component counts:

\[(\mathbb{E}[X_C(t)], \mathbb{E}[X_{Cw}(t)], \mathbb{E}[X_{Cp}(t)], \mathbb{E}[X_S(t)], \mathbb{E}[X_{Sp}(t)], \mathbb{E}[X_{Sf}(t)]) \in \mathbb{R}^6_+\]

Convergence as population size increases is much faster
By the Markov (memoryless) property, we have:

\[ P\{X_C(t + \delta t) - X_C(t) = 1 | X(t)\} = r_p X_{Cp}(t) \delta t + o(\delta t) \]
Client/server model first-moment approximation

By the Markov (memoryless) property, we have:

\[
\mathbb{P}\{X_C(t + \delta t) - X_C(t) = 1 \mid \mathbf{X}(t)\} = r_p X_{C_p}(t) \delta t + o(\delta t)
\]

\[
\mathbb{P}\{X_C(t + \delta t) - X_C(t) = -1 \mid \mathbf{X}(t)\} = r_q \min(X_C(t), X_S(t)) \delta t + o(\delta t)
\]
By the Markov (memoryless) property, we have:

\[
\mathbb{P}\{X_C(t + \delta t) - X_C(t) = 1 \mid X(t)\} = r_p X_{C_p}(t) \delta t + o(\delta t)
\]

\[
\mathbb{P}\{X_C(t + \delta t) - X_C(t) = -1 \mid X(t)\} = r_q \min(X_C(t), X_S(t)) \delta t + o(\delta t)
\]

Therefore:

\[
\frac{\mathbb{E}[X_C(t + \delta t) - X_C(t)]}{\delta t} = r_p \mathbb{E}[X_{C_p}(t)] - r_q \mathbb{E}[\min(X_C(t), X_S(t))] + \frac{o(\delta t)}{\delta t}
\]
By the Markov (memoryless) property, we have:

$$
\mathbb{P}\{X_C(t + \delta t) - X_C(t) = 1 | \mathbf{X}(t)\} = rpX_{Cp}(t)\delta t + o(\delta t)
$$

$$
\mathbb{P}\{X_C(t + \delta t) - X_C(t) = -1 | \mathbf{X}(t)\} = rq\min(X_C(t), X_S(t))\delta t + o(\delta t)
$$

Therefore:

$$
\frac{\mathbb{E}[X_C(t + \delta t) - X_C(t)]}{\delta t} = rp\mathbb{E}[X_{Cp}(t)] - rq\mathbb{E}[\min(X_C(t), X_S(t))] + \frac{o(\delta t)}{\delta t}
$$

And we recover the differential equation:

$$
\frac{d\mathbb{E}[X_C(t)]}{dt} = rp\mathbb{E}[X_{Cp}(t)] - rq\mathbb{E}[\min(X_C(t), X_S(t))]
$$
Client/server model first-moment approximation

\[ \frac{d\mathbb{E}[X_C(t)]}{dt} = r_p \mathbb{E}[X_{C_p}(t)] - r_q \mathbb{E}[\min(X_C(t), X_S(t))] \]
Client/server model first-moment approximation

\[
\frac{d\mathbb{E}[X_C(t)]}{dt} = r_p \mathbb{E}[X_{C_p}(t)] - r_q \mathbb{E}[\min(X_C(t), X_S(t))]\]
Client/server model first-moment approximation

\[ \frac{d\mathbb{E}[X_C(t)]}{dt} = r_p \mathbb{E}[X_{C_p}(t)] - r_q \mathbb{E}[\min(X_C(t), X_S(t))] \]
\[ \approx r_p \mathbb{E}[X_{C_p}(t)] - r_q \min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) \]
Client/server model first-moment approximation

\[
\frac{d\mathbb{E}[X_C(t)]}{dt} = r_p \mathbb{E}[X_{C_p}(t)] - r_q \mathbb{E}[\min(X_C(t), X_S(t))] \\
\approx r_p \mathbb{E}[X_{C_p}(t)] - r_q \min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) \\
\frac{dv_C(t)}{dt} = r_p v_{C_p}(t) - r_q \min(v_C(t), v_S(t)) \quad (v(t) \approx \mathbb{E}[X(t)])
\]
\[
\frac{d\mathbb{E}[X_S(t)]}{dt} = r_{st}\mathbb{E}[X_{S_f}(t)] + r_{rs}\mathbb{E}[^{\min}(X_{C_w}(t), X_{S_p}(t))] \\
- (r_{rq}\mathbb{E}[^{\min}(X_C(t), X_S(t))] + r_f\mathbb{E}[X_S(t)])
\]
Client/server model first-moment approximation

\[ \frac{d\mathbb{E}[X_S(t)]}{dt} = r_{rst}\mathbb{E}[X_{S_f}(t)] + r_{rs}\mathbb{E}[\min(X_{C_w}(t), X_{S_p}(t))] 
\]
\[ - (r_{rq}\mathbb{E}[\min(X_C(t), X_S(t))] + r_f\mathbb{E}[X_S(t)]) \]
Client/server model first-moment approximation

\[
\frac{d\mathbb{E}[X_S(t)]}{dt} = r_{st}\mathbb{E}[X_{S_f}(t)] + r_{rs}\mathbb{E}[\min(X_{C_w}(t), X_{S_p}(t))] \\
- (r_{rq}\mathbb{E}[\min(X_C(t), X_S(t))] + r_f\mathbb{E}[X_S(t)]) \\
\approx r_{st}\mathbb{E}[X_{S_f}(t)] + r_{rs}\min(\mathbb{E}[X_{C_w}(t)], \mathbb{E}[X_{S_p}(t)]) \\
- (r_{rq}\min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) + r_f\mathbb{E}[X_S(t)])
\]
Client/server model first-moment approximation

\[
\frac{d\mathbb{E}[X_S(t)]}{dt} = r_{rst}\mathbb{E}[X_{S_f}(t)] + r_{rs}\mathbb{E}[\min(X_{C_w}(t), X_{S_p}(t))] \\
- \left( r_{rq}\mathbb{E}[\min(X_C(t), X_S(t))] + r_f\mathbb{E}[X_S(t)] \right) \\
\approx r_{rst}\mathbb{E}[X_{S_f}(t)] + r_{rs}\min(\mathbb{E}[X_{C_w}(t)], \mathbb{E}[X_{S_p}(t)]) \\
- \left( r_{rq}\min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) + r_f\mathbb{E}[X_S(t)] \right)
\]

\[
\frac{dv_S(t)}{dt} = r_{rst}v_{S_f}(t) + r_{rs}\min(v_{C_w}(t), v_{S_p}(t)) \\
- \left( r_{rq}\min(v_C(t), v_S(t)) + r_f v_S(t) \right) \quad (v(t) \approx \mathbb{E}[X(t)])
\]
Client/server model first-moment approximation

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]
\[ \text{Server} \rightarrow \text{Server}_{\text{proc}} \]

\[ \text{proc } @ r_p \]
\[ \text{req} \rightarrow \text{res} \]
\[ r_q \min(X_C(t), X_S(t)) \]
\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 10, N_S = 6 \]
Client/server model first-moment approximation

\[
\text{Client} \xrightarrow{\text{req}} \text{Client}_{\text{wait}} \xrightarrow{\text{res}} \text{Client}_{\text{proc}}
\]

\[
\text{Server} \xrightarrow{\text{req}} \xrightarrow{\text{res}} \text{Server}_{\text{proc}} \xrightarrow{\text{fail}} \text{Server}_{\text{fail}}
\]

\[
\text{proc} \xrightarrow{r_p} \text{Client} \xrightarrow{r_q \min (X_C(t), X_S(t))} \text{Server} \xrightarrow{r_s \min (X_{Cw}(t), X_{Sp}(t))} \text{Server}_{\text{proc}}
\]

\[
N_C = 20, N_S = 12
\]
Client/server model first-moment approximation

\[ N_C = 50, \quad N_S = 30 \]
Client/server model first-moment approximation

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]

\[ \text{Server} \rightarrow \text{Server}_{\text{proc}} \]

\[ \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]

\[ \text{req} \rightarrow \text{res} \]

\[ r_p \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 100, N_S = 60 \]

\[ \text{Time, } t \]

\[ \text{Rescaled component count} \]

\[ \text{Client} \]

\[ \text{Client}_{\text{wait}} \]

\[ \text{Client}_{\text{proc}} \]

\[ \text{Server} \]

\[ \text{Server}_{\text{proc}} \]

\[ \text{Server}_{\text{fail}} \]

\[ \text{reset} \rightarrow \text{res} \]

\[ \text{fail} \rightarrow \text{reset} \]

\[ \text{9/30} \]
Client/server model first-moment approximation

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 500, N_S = 300 \]
Client/server model second-moment approximation

Client \( \rightarrow \) Client\(_{\text{wait}} \)
- \text{proc} \oplus \, r_p
- \text{req} \rightarrow \text{res}
  \begin{align*}
  r_q \min(X_C(t), X_S(t)) & \\
  r_s \min(X_{Cw}(t), X_{Sp}(t)) & \\
  \end{align*}

Server \( \rightarrow \) Server\(_{\text{proc}} \)
- \text{req} \rightarrow \text{res}
  \begin{align*}
  \end{align*}

Server\(_{\text{fail}} \rightarrow \) Server
- \text{reset} \oplus r_{rst}
- \text{fail} \oplus r_f

\begin{align*}
  N_C & \\
  N_S & \\
\end{align*}
This can be taken further and the system of ODEs extended to also approximate higher-order moments, e.g.:

\[ \mathbb{E}[X^2_C(t)] \quad \text{Var}[X^2_S(t)] \quad \mathbb{E}[X^3_{cw}(t)] \quad \text{Skew}[X_{sp}(t)] \]
This can be taken further and the system of ODEs extended to also approximate higher-order moments, e.g.:

\[
\mathbb{E}[X_C^2(t)] \quad \text{Var}[X_S^2(t)] \quad \mathbb{E}[X_{Cw}^3(t)] \quad \text{Skew}[X_{Sp}(t)]
\]

More detailed information about the distribution of the component counts
Client/server model second-moment approximation

Again by the Markov (memoryless) property, we have:

\[
P\{X_C^2(t + \delta t) - X_C^2(t) = 2X_C(t) + 1 | X(t)\} = r_p X_{C_p}(t)\delta t + o(\delta t)
\]
Again by the Markov (memoryless) property, we have:

\[
P\{X_C^2(t+\delta t) - X_C^2(t) = 2X_C(t) + 1 \mid X(t)\} = r_p X_{Cp}(t) \delta t + o(\delta t)
\]

\[
P\{X_C^2(t+\delta t) - X_C^2(t) = -2X_C(t) + 1 \mid X(t)\} = r_q \min(X_C(t), X_S(t)) \delta t + o(\delta t)
\]
Client/server model second-moment approximation

Again by the Markov (memoryless) property, we have:

\[
\mathbb{P}\{X_C^2(t + \delta t) - X_C^2(t) = 2X_C(t) + 1 \mid X(t)\} = r_p X_{Cp}(t)\delta t + o(\delta t)
\]

\[
\mathbb{P}\{X_C^2(t + \delta t) - X_C^2(t) = -2X_C(t) + 1 \mid X(t)\} = r_{rq} \min(X_C(t), X_S(t))\delta t + o(\delta t)
\]

\[
\frac{\mathbb{E}[X_C^2(t + \delta t) - X_C^2(t)]}{\delta t} = r_p \mathbb{E}[(2X_C(t) + 1)X_{Cp}(t)]
\]

\[
- r_{rq} \mathbb{E}[(2X_C(t) - 1) \min(X_C(t), X_S(t))] + \frac{o(\delta t)}{\delta t}
\]
Again by the Markov (memoryless) property, we have:

\[
P\{X_C^2(t + \delta t) - X_C^2(t) = 2X_C(t) + 1|X(t)\} = r_p X_C p(t) \delta t + o(\delta t)
\]

\[
P\{X_C^2(t + \delta t) - X_C^2(t) = -2X_C(t) + 1|X(t)\} = r_q \min(X_C(t), X_S(t)) \delta t + o(\delta t)
\]

\[
\frac{E[X_C^2(t + \delta t) - X_C^2(t)]}{\delta t} = r_p E[(2X_C(t) + 1)X_C p(t)]
\]

\[
- r_q E[(2X_C(t) - 1) \min(X_C(t), X_S(t))] + \frac{o(\delta t)}{\delta t}
\]

\[
\frac{dE[X_C^2(t)]}{dt} = 2r_p E[X_C(t)X_C p(t)] + r_p E[X_C p(t)]
\]

\[
- 2r_q E[X_C(t) \min(X_C(t), X_S(t))] + r_q E[\min(X_C(t), X_S(t))]
\]
Client/server model second-moment approximation

\[
\frac{d\mathbb{E}[X_C^2(t)]}{dt} = 2r_p\mathbb{E}[X_C(t)X_{cp}(t)] + r_p\mathbb{E}[X_{cp}(t)]
\]

\[
- 2r_q\mathbb{E}[X_C(t)\min(X_C(t), X_S(t))] + r_q\mathbb{E}[\min(X_C(t), X_S(t))]
\]
Client/server model second-moment approximation

\[ \frac{d\mathbb{E}[X_C(t)^2]}{dt} = 2r_p \mathbb{E}[X_C(t)X_{Cp}(t)] + r_p \mathbb{E}[X_{Cp}(t)] - 2r_{rq} \mathbb{E}[X_C(t) \min(X_C(t), X_S(t))] + r_{rq} \mathbb{E}[\min(X_C(t), X_S(t))] \]
Client/server model second-moment approximation

\[ \frac{d\mathbb{E}[X_C^2(t)]}{dt} = 2r_p \mathbb{E}[X_C(t)X_{cp}(t)] + r_p \mathbb{E}[X_{cp}(t)] \\
- 2r_q \mathbb{E}[X_C(t) \min(X_C(t), X_S(t))] + r_q \mathbb{E}[\min(X_C(t), X_S(t))] \\
\approx 2r_p \mathbb{E}[X_C(t)X_{cp}(t)] + r_p \mathbb{E}[X_{cp}(t)] \\
- 2r_q \min(\mathbb{E}[X_C^2(t)], \mathbb{E}[X_C(t)X_S(t)]) + r_q \min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) \]
Client/server model second-moment approximation

\[
\frac{d\mathbb{E}[X_C^2(t)]}{dt} = 2r_p\mathbb{E}[X_C(t)X_{C_p}(t)] + r_p\mathbb{E}[X_{C_p}(t)] - 2r_q\mathbb{E}[X_C(t)\min(X_C(t), X_S(t))] + r_q\mathbb{E}[\min(X_C(t), X_S(t))]
\approx 2r_p\mathbb{E}[X_C(t)X_{C_p}(t)] + r_p\mathbb{E}[X_{C_p}(t)] - 2r_q\min(\mathbb{E}[X_C^2(t)], \mathbb{E}[X_C(t)X_S(t)]) + r_q\min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)])
\]

\[
\frac{dv_{C_2}(t)}{dt} = 2r_pv_{C_{C_p}}(t) + r_pv_{C_p}(t) - 2r_q\min(v_{C_2}(t), v_{C_S}(t)) + r_q\min(v_C(t), v_S(t))
\]
Client/server model second-moment approximation

\[ \text{Server}_{\text{fail}} \rightarrow \text{reset} @ r_{\text{rst}} \rightarrow \text{fail} @ r_f \]

\[ \text{proc} @ r_p \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{req} \rightarrow \text{Client}_{\text{proc}} \rightarrow \text{res} \]

\[ \text{Server} \rightarrow \text{req} \rightarrow \text{Server}_{\text{proc}} \rightarrow \text{res} \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 50, \quad N_S = 30 \]
Client/server model second-moment approximation

\[ \begin{align*}
&\text{Client} \\
&\text{Client}_{\text{wait}} \\
&\text{Client}_{\text{proc}} \\
&\text{Server} \\
&\text{Server}_{\text{proc}} \\
\end{align*} \]

proc @ \( r_p \)

\( \text{req} \) \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{res} \rightarrow \text{Client}_{\text{proc}} \)

\( \text{req} \rightarrow \text{Server} \rightarrow \text{res} \rightarrow \text{Server}_{\text{proc}} \)

\( r_q \min(X_C(t), X_S(t)) \)

\( r_s \min(X_{Cw}(t), X_{Sp}(t)) \)

\( N_C = 100, N_S = 60 \)

\[ \begin{align*}
&\text{Rescaled component count s.d.} \\
&\text{Time, } t \\
&0 \quad 1 \quad 2 \quad 3 \quad 4 \quad 5 \\
&0 \quad 0.2 \quad 0.4 \\
\end{align*} \]
Client/server model second-moment approximation

\[ N_C = 200, \ N_S = 120 \]
Client/server model second-moment approximation

\[ \text{Client} \rightarrow \text{req} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{res} \rightarrow \text{Client}_{\text{proc}} \]

\[ \text{Server} \rightarrow \text{req} \rightarrow \text{Server}_{\text{proc}} \rightarrow \text{res} \rightarrow \text{Server}_{\text{fail}} \rightarrow \text{fail} \rightarrow \text{Server}_{\text{reset}} \]

\[ \text{req} \rightarrow r_{\text{req}} \min(X_C(t), X_S(t)) \rightarrow \text{res} \rightarrow r_{\text{res}} \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 500, N_S = 300 \]
Client/server model second-moment approximation

\[ \text{proc @ } r_p \]

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]

\[ \text{Server} \rightarrow \text{Server}_{\text{proc}} \rightarrow \text{Server}_{\text{fail}} \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{C_w}(t), X_{S_p}(t)) \]

\[ N_C = 1000, N_S = 600 \]
Client/server model second-moment approximation

\[ \text{proc} @ r_p \]

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]

\[ \text{req} \rightarrow \text{res} \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 10000, N_S = 6000 \]
Switch points

![Diagram of the switch points]

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]
\[ \text{Server}_{\text{fail}} \rightarrow \text{reset} @ r_{\text{rst}} \rightarrow \text{fail} @ r_{\text{f}} \]

\[ \text{Server} \rightarrow \text{req} \rightarrow \text{Server}_{\text{proc}} \]

\[ r_{\text{req}} \min(X_C(t), X_S(t)) \]
\[ r_{\text{res}} \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C = 100, \ N_S = 60 \]
Switch points

$N_C = 100, N_S = 60$
Switch points

\[ N_C = 100, \, N_S = 60 \]
Switch points

\[ \mathbb{E}[\min(X_C(t), X_S(t))] \approx \min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) \]
Switch points

\[
\mathbb{E}[\min(X_C(t), X_S(t))] \approx \min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)])
\]

- Most accurate when \(X_C(t)\) and \(X_S(t)\) are unlikely to be close
Switch points

\[ \mathbb{E}[\min(X_C(t), X_S(t))] \approx \min(\mathbb{E}[X_C(t)], \mathbb{E}[X_S(t)]) \]

- Most accurate when \( X_C(t) \) and \( X_S(t) \) are unlikely to be close
- However, often we wish to design a system so that they are likely to be close, e.g. quality and efficiency driven regime for queueing\(^8\)/resource-constrained models

Functional central limit theorem

*Same assumptions as before, plus:*

- Define $f(x) := \sum_{c \in C} L_c r_c(x) \, ds$ and assume $f$ is totally differentiable (w.r.t. $x$) at points $v(t)$, $t$-almost-everywhere
Functional central limit theorem

Same assumptions as before, plus:

- Define $f(x) := \sum_{c \in C} L_c r_c(x) \, ds$ and assume $f$ is totally differentiable (w.r.t. $x$) at points $v(t)$, $t$-almost-everywhere.

Theorem: PCTMC FCLT

As $N \to \infty$:

$$\frac{X_N(t) - Nv(t)}{\sqrt{N}} \Rightarrow E(t) \quad \text{in} \quad D_{\mathbb{R}^{|S|}}[0, \infty)$$

where $E(t)$ is the unique solution to:

$$E(t) = \int_0^t Df(v(s)) \cdot E(s) \, ds + \sum_{c \in C} B_c \left( \int_0^t r_c(v(s)) \, ds \right) L_c$$

and the $\{B_c(t)\}_{c \in C}$ are mutually independent standard Brownian motions.
Theorem: PCTMC FCLT

As $N \to \infty$:

$$\frac{X^N(t) - Nv(t)}{\sqrt{N}} \Rightarrow E(t) \quad \text{in } D_{\mathbb{R}|S|}[0, \infty)$$

where $E(t)$ is the unique solution to:

$$E(t) = \int_0^t Df(v(s)) \cdot E(s) \, ds + \sum_{c \in C} B_c \left( \int_0^t r_c(v(s)) \, ds \right) L_c$$

and the $\{B_c(t)\}_{c \in C}$ are mutually independent standard Brownian motions

- $E(t)$ is a Gaussian process

Functional central limit theorem

**Theorem: PCTMC FCLT**

As \( N \to \infty \):

\[
\frac{X^N(t) - Nv(t)}{\sqrt{N}} \Rightarrow E(t) \quad \text{in } D_{[0, \infty)}
\]

where \( E(t) \) is the unique solution to:

\[
E(t) = \int_0^t Df(v(s)) \cdot E(s) \, ds + \sum_{c \in C} B_c \left( \int_0^t r_c(v(s)) \, ds \right) L_c
\]

and the \( \{B_c(t)\}_{c \in C} \) are mutually independent standard Brownian motions

- \( E(t) \) is a Gaussian process
- Covariance matrix of \( E(t) \) satisfies exactly a system of differential equations very similar to those derived before

Improving the approximation

- As a result of the FCLT, rescaled component counts become approximately jointly normal
Improving the approximation

- As a result of the FCLT, rescaled component counts become approximately jointly normal
- Can thus compute approximation to $\mathbb{E}[\min(X_C(t), X_S(t))]$ by assuming that $(X_C(t), X_S(t))$ is jointly normal
Improving the approximation

- As a result of the FCLT, rescaled component counts become approximately jointly normal
- Can thus compute approximation to $\mathbb{E}[\min(X_C(t), X_S(t))]$ by assuming that $(X_C(t), X_S(t))$ is jointly normal

$$N_C = 100, \; N_S = 60$$
Improving the approximation

- As a result of the FCLT, rescaled component counts become approximately jointly normal
- Can thus compute approximation to $\mathbb{E}[\min(X_C(t), X_S(t))]$ by assuming that $(X_C(t), X_S(t))$ is jointly normal

$$N_C = 100, N_S = 60$$
Client/server model skewness approximation

\[ \text{proc} \odot r_p \]

\[ \text{Client} \]
\[ \text{Client}_{\text{wait}} \]
\[ \text{Client}_{\text{proc}} \]

\[ \text{req} \]
\[ \text{res} \]

\[ \text{Server}_{\text{fail}} \]
\[ \text{Server}_{\text{proc}} \]

\[ \text{reset} \odot r_{rst} \]
\[ \text{fail} \odot r_f \]

\[ r_q \min(X_C(t), X_S(t)) \]
\[ r_s \min(X_{Cw}(t), X_{Sp}(t)) \]

\[ N_C \]
\[ N_S \]
Client/server model skewness approximation

\[ \text{Client} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{Client}_{\text{proc}} \]
\[ \text{proc} @ r_p \]

\[ r_q \min(X_C(t), X_S(t)) \]

\[ \text{Server} \rightarrow \text{Server}_{\text{proc}} \rightarrow \text{Server}_{\text{fail}} \]
\[ \text{reset} @ r_{\text{rst}} \rightarrow \text{fail} @ r_f \]

\[ N_C = 100, N_S = 60 \]

\[ \text{Std. comp. count skewness} \]

0 1 2 3 4 5
0 0.5 1

Time, t

\[ N_C = 100, N_S = 60 \]

\[ \text{Std. comp. count skewness} \]

0 1 2 3 4 5
0 0.5 1

Time, t
Individual behaviour

In addition to aggregate (global) behaviour, this approach can also be used to reason about individual objects.
Individual behaviour

In addition to aggregate (global) behaviour, this approach can also be used to reason about individual objects

- Let $Y^N(t) \in S$ be the state of one object

Theorem

If $Y^N(0) \Rightarrow Y(0)$, then $(X^N(t), Y^N(t)) \Rightarrow (v(t), Y(t))$ as $N \to \infty$, where $Y(t)$ is a time-inhomogeneous CTMC with generator:

\[
Q_{ss'} = \sum_{c \in C} \left| \{(s, s') \in L_c \} \right| v_s r_c(v(t))
\]
Individual behaviour

In addition to aggregate (global) behaviour, this approach can also be used to reason about individual objects

- Let $Y^N(t) \in S$ be the state of one object
- State-dependent generator matrix for $Y^N(t)$:
\[
[Q^N(\mathbf{X}^N(t))]_{s,s'} = \sum_{c \in \mathcal{C}} \frac{|\{(s,s') \in L_c\}|}{X^N_s(t)} r_c(\mathbf{X}^N(t))
\]
Individual behaviour

In addition to aggregate (global) behaviour, this approach can also be used to reason about individual objects

- Let $Y^N(t) \in S$ be the state of one object
- State-dependent generator matrix for $Y^N(t)$:

$$[Q^N(X^N(t))]_{s,s'} = \sum_{c \in C} \frac{|\{(s,s') \in L_c\}|}{\chi^N_s(t)} r_c(X^N(t))$$

**Theorem**

If $Y^N(0) \Rightarrow Y(0)$, then $(X^N(t), Y^N(t)) \Rightarrow (v(t), Y(t))$ as $N \to \infty$, where $Y(t)$ is a time-inhomogeneous CTMC with generator:

$$[Q(t)]_{s,s'} = \sum_{c \in C} \frac{|\{(s,s') \in L_c\}|}{\nu_s(t)} r_c(v(t))$$
Individual behaviour

In addition to aggregate (global) behaviour, this approach can also be used to reason about individual objects

- Let $Y^N(t) \in S$ be the state of one object
- State-dependent generator matrix for $Y^N(t)$:

$$[Q^N(X^N(t))]_{s,s'} = \sum_{c \in C} \frac{\| (s,s') \in L_c \|}{X^N_s(t)} r_c(X^N(t))$$

**Theorem**

If $Y^N(0) \Rightarrow Y(0)$, then $(X^N(t), Y^N(t)) \Rightarrow (v(t), Y(t))$ as $N \rightarrow \infty$, where $Y(t)$ is a time-inhomogeneous CTMC with generator:

$$[Q(t)]_{s,s'} = \sum_{c \in C} \frac{\| (s, s') \in L_c \|}{v_s(t)} r_c(v(t))$$

- Or under steady state assumptions, generator is time-homogeneous:

$$[Q]_{s,s'} = \sum_{c \in C} \frac{\| (s, s') \in L_c \|}{v^*_s} r_c(v^*)$$
Individually passage time

How long does it take a single client to make a request, receive a response and process it?
Individual passage time

How long does it take a single client to make a request, receive a response and process it?
How long does it take a single client to make a request, receive a response and process it?
**Individual passage time**

\[
T := \inf \{ t \geq 0 : C(t) = \text{Client'} \}, \text{ given that } C(0) = \text{Client}
\]
Individual passage time example
Individual passage time example

$N_C = 10, N_S = 6$
Individual passage time example

$N_C = 20, N_S = 12$
Individual passage time example

$N_C = 50, N_S = 30$
Individual passage time example

$N_C = 100$, $N_S = 60$
Individual passage time example

$N_C = 200, N_S = 120$
Accumulated reward measures

- Cost, energy, heat, ...
- Constant rate
Accumulated reward measures

- Cost, energy, heat, ...
- Constant rate

\[ \text{Server} \quad \int_{t_0}^{t} X_s(u) \, du + \int_{t_0}^{t} X_{sp}(u) \, du \]

\[ \text{Server}_{proc} \quad \uparrow \]
Accumulated reward measures

- Cost, energy, heat, …
- Constant rate

\[
\text{Total energy}(t) = \int_0^t X_S(u) \, du + \int_0^t X_{Sp}(u) \, du
\]
Accumulated reward measures

- Cost, energy, heat, ...  
- Constant rate

\[
\text{total energy}(t) = r_S \int_0^t X_S(u) \, du + r_{Sp} \int_0^t X_{Sp}(u) \, du
\]
Accumulated reward measures

- Cost, energy, heat, ...
- Constant rate

\[
\text{total energy}(t) = \int_0^t \! X_S(u) \, du + \int_0^t \! X_{Sp}(u) \, du
\]
Accumulated reward measures

- Cost, energy, heat, ... 
- Constant rate

\[
\text{Total energy}(t) = r_S \int_0^t X_S(u) \, du + r_{Sp} \int_0^t X_{Sp}(u) \, du
\]
Accumulated reward measures

- Cost, energy, heat, ...
- Constant rate

![Graph showing accumulated rewards over time]

Total energy at time $t$ is given by:

$$\text{total energy}(t) = \int_0^t r_S(u) \, du + \int_0^t r_{Sp}(u) \, du$$
Accumulated reward measures

- Cost, energy, heat, ...
- Constant rate

\[
\text{total energy}(t) = r_S \int_0^t X_S(u) \, du + r_{Sp} \int_0^t X_{Sp}(u) \, du
\]
Accumulated reward measures

- Cost, energy, heat, ...
- Constant rate

\[
\text{total energy}(t) = r_s \int_0^t X_S(u) \, du + r_{sp} \int_0^t X_{Sp}(u) \, du
\]
Moment approximations of accumulated rewards

- Explicit state analysis and simulation are also very costly for accumulated rewards
Moment approximations of accumulated rewards

- Explicit state analysis and simulation are also very costly for accumulated rewards

- Can **extend** the ODE system for component count moments with ODEs for moments of accumulated counts:[9]

---

Moment approximations of accumulated rewards

- Explicit state analysis and simulation are also very costly for accumulated rewards.

- Can extend the ODE system for component count moments with ODEs for moments of accumulated counts:

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_{S_p}(u) \, du \right] = \cdots
\]
Moment approximations of accumulated rewards

- Explicit state analysis and simulation are also very costly for accumulated rewards

- Can extend the ODE system for component count moments with ODEs for moments of accumulated counts:

\[
\frac{d}{dt} E \left[ \int_0^t X_{S_p}(u) \, du \right] = \cdots
\]

- Can be used, for example, to derive completion time distributions
Moment approximations of accumulated rewards

- Explicit state analysis and simulation are also very costly for accumulated rewards

- Can extend the ODE system for component count moments with ODEs for moments of accumulated counts:

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_{S_p}(u) \, du \int_0^t X_S(u) \, du \right] = \cdots
\]

- Can be used, for example, to derive completion time distributions
Moment approximations of accumulated rewards

First-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] =
\]

Second-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = 2 \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right] + \cdots + \mathbb{E} \left[ X_S^2(t) \right]
\]
Moment approximations of accumulated rewards

First-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]
Moment approximations of accumulated rewards

**First-order moments**

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]
Moment approximations of accumulated rewards

**First-order moments**

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]

**Second-order moments**

\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = \ldots
\]
Moment approximations of accumulated rewards

**First-order moments**

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]

**Second-order moments**

\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = 2\mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right]
\]

First-order moments

\[
\frac{d}{dt} \mathbb{E}[X_S(t)] = \cdots
\]
Moment approximations of accumulated rewards

First-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]

Second-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = 2 \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right]
\]

Combined moments

\[
\frac{d}{dt} \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]
Moment approximations of accumulated rewards

First-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]

Second-order moments

\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = 2 \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right]
\]

Combined moments

\[
\frac{d}{dt} \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right] = \mathbb{E} \left[ X_C(t) \int_0^t X_S(u) \, du \right] + \cdots + \mathbb{E}[X_S^2(t)]
\]
Moment approximations of accumulated rewards

First-order moments
\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]

Second-order moments
\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = 2 \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right]
\]

Combined moments
\[
\frac{d}{dt} \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right] = \mathbb{E} \left[ X_C(t) \int_0^t X_S(u) \, du \right] + \cdots + \mathbb{E}[X_S^2(t)]
\]

First-order moments
\[
\frac{d}{dt} \mathbb{E}[X_S(t)] = \cdots
\]

Second-order moments
\[
\frac{d}{dt} \mathbb{E}[X_S(t)X_{S_p}(t)] = \cdots
\]
Moment approximations of accumulated rewards

First-order moments
\[
\frac{d}{dt} \mathbb{E} \left[ \int_0^t X_S(u) \, du \right] = \mathbb{E}[X_S(t)]
\]

Second-order moments
\[
\frac{d}{dt} \mathbb{E} \left[ \left( \int_0^t X_S(u) \, du \right)^2 \right] = 2 \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right]
\]

Combined moments
\[
\frac{d}{dt} \mathbb{E} \left[ X_S(t) \int_0^t X_S(u) \, du \right] = \mathbb{E} \left[ X_C(t) \int_0^t X_S(u) \, du \right] + \cdots + \mathbb{E}[X_S^2(t)]
\]
Trade-off between energy and performance

![Diagram showing the trade-off between energy and performance]
Trade-off between energy and performance

---

**Diagram:**

- **Client**
  - req → **Client\_wait**
  - res → **Client\_proc**

- **Server**
  - req → **Server\_proc**
  - res → **Server\_fail**

- **N\_C**
- **N\_S**

**Graphs:**

1. **Client serviced before t**
   - Time, t: 0, 2, 4, 6, 8, 10
   - Probability: 0, 0.6, 0.8, 1

2. **E[total energy(t)]**
   - Time, t: 0, 2, 4, 6
   - Energy: 0, 50, 100, 150

---

**Text:**

- Client wait
- Client proc
- Server
- Server proc
- Server fail
- Proc

---
Trade-off between energy and performance

Client

\[ \text{Client}_{\text{wait}} \]

\[ \text{Client}_{\text{proc}} \]

\[ \text{Server}_{\text{fail}} \]

\[ \text{Server}_{\text{proc}} \]

\[ n_C \]

\[ n_S \]

\begin{align*}
\text{Probability} & \quad \text{Time, } t \\
0 & \quad 0 \\
0.6 & \quad 2 \\
0.8 & \quad 4 \\
1 & \quad 6 \\
\end{align*}

\begin{align*}
\text{SLA} & \quad 7s \\
\geq & \quad 0.99 \\
\end{align*}

\begin{align*}
\text{Energy} & \quad \mathbb{E}[\text{total energy}(t)] \\
0 & \quad 0 \\
50 & \quad 2 \\
100 & \quad 4 \\
150 & \quad 6 \\
\end{align*}
Trade-off between energy and performance

Client

\[\text{req} \rightarrow \text{Client}_{\text{wait}} \rightarrow \text{res} \rightarrow \text{Client}_{\text{proc}} \rightarrow N_C\]

Server

\[\text{req} \rightarrow \text{Server}_{\text{proc}} \rightarrow \text{req} \rightarrow \text{Server}_{\text{fail}} \rightarrow \text{reset} \rightarrow \text{Server}_{\text{sleep}} \rightarrow \text{sleep/wakeup} \]

\[\text{SLA} 7s \geq 0.99\]

\[\text{Client serviced before } t\]

\[\text{Energy } E[\text{total energy}(t)]\]

\[\text{Energy } E[\text{total energy}(t)]\]

\[\text{Energy } E[\text{total energy}(t)]\]
Trade-off between energy and performance

- Client
  - req
  - Client\textsubscript{wait}
  - res
  - Client\textsubscript{proc}
  - N\textsubscript{C}

- Server\textsubscript{fail}
  - reset
  - Server\textsubscript{sleep}

- Server
  - req
  - Server\textsubscript{proc}
  - N\textsubscript{S}

\textbf{Probability}

- Client serviced before \( t \)
- SLA 7s \( \geq 0.99 \)

\textbf{Energy}

- \( E[\text{total energy}(t)] \)

\textbf{Time, } t

- 0 2 4 6 8 10
- 0.6 0.7 0.8 0.9 1.0

- 0 50 100 150
- 0 2 4 6
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations

- Number of servers, $N_S$
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate

Minimise energy consumption while satisfying SLAs
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate

Minimise energy consumption while satisfying SLAs

Individual passage-time SLA:
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate

Minimise energy consumption while satisfying SLAs

Individual passage-time SLA: clients must finish in at most $7s \geq 99\%$ of the time
Trade-off between energy and performance

**Scalable analysis** allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate

**Minimise** energy consumption while satisfying SLAs

Individual passage-time SLA: clients must finish in at most 7s $\geq 99\%$ of the time
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations
- Number of servers, $N_S$
- Sleep/wakeup rate

Minimise energy consumption while satisfying SLAs

Individual passage-time SLA: clients must finish in at most 7s ≥ 99% of the time
Trade-off between energy and performance

Scalable analysis allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate

Minimise energy consumption while satisfying SLAs

Individual passage-time SLA: clients must finish in at most 7s

$\geq 99.5\%$ of the time
Trade-off between energy and performance

**Scalable analysis** allows exploration of many configurations

- Number of servers, $N_S$
- Sleep/wakeup rate

**Minimise** energy consumption while **satisfying** SLAs

![3D graph showing the trade-off between sleep rate and number of servers on energy consumption with SLA met at certain conditions.]

Individual passage-time SLA: clients must finish in at most 7s $\geq 99.5\%$ of the time
Extensions: generally-timed transitions
Software update model with deterministic timeouts

\[ \dot{v}_c(t) = -\rho v_c(t) - \beta N v_c(t) v_a(t) + \lambda v_e(t) - 1 \{ t \geq \gamma \} \lambda v_e(t - \gamma) \]

Rate of determ. clocks starting at \( t - \gamma \) exp\( ( -\int_{t}^{t-\gamma} \beta v_a(s) N ds ) \) exp\( ( -\rho \gamma ) \)

Prob. that timeout occurs before node updated or went off

Software update model with deterministic timeouts

\[ \dot{v}_c(t) = \]

Software update model with deterministic timeouts

\[ \dot{v}_c(t) = -\rho v_c(t) - \frac{\beta}{N} v_c(t)v_a(t) + \lambda v_e(t) \]

Software update model with deterministic timeouts

\[
\dot{v}_c(t) = -\rho v_c(t) - \frac{\beta}{N} v_c(t) v_a(t) + \lambda v_e(t)
\]

Rate of determ.
clocks starting at \(t-\gamma\)

\[
- \mathbf{1}_{\{t \geq \gamma\}} \lambda v_e(t-\gamma) \exp \left( - \int_{t-\gamma}^{t} \frac{\beta v_a(s)}{N} \, ds \right) \exp(-\rho \gamma)
\]

Prob. that timeout occurs before node updated or went off

Software update model with deterministic timeouts

For some classes of models which admit a representation in terms of delayed Poisson processes, a similar proof can be carried out.

Otherwise it seems necessary to use representations which become infinite dimensional in the limit.
Extensions: mixed discrete–continuous local state space
Nodes are in one of two discrete states: *active* (a) or *idle* (i)
Nodes are in one of two discrete states: *active* (a) or *idle* (i)

- Idle nodes are awaiting stimulus, active nodes are exchanging collected data with a neighbour
Wireless sensor network model

Nodes are in one of two discrete states: active (\textit{a}) or idle (\textit{i})

- Idle nodes are awaiting stimulus, active nodes are exchanging collected data with a neighbour
- Each has own battery, drains at a state-dependent rate

\[ 1_{\{B_t > 0\}} \lambda \]

\[ B_t \]
Nodes are in one of two discrete states: \emph{active} (a) or \emph{idle} (i).

Idle nodes are awaiting stimulus, active nodes are exchanging collected data with a neighbour.

Each has own battery, drains at a state-dependent rate.

Threshold control — wireless radios operate at two different power levels: $0 < B_t \leq B^*$ (\emph{low}) or $B_t > B^*$ (\emph{high}).
Wireless sensor network model

Nodes are in one of two discrete states: active (a) or idle (i)
Idle nodes are awaiting stimulus, active nodes are exchanging collected data with a neighbour
Each has own battery, drains at a state-dependent rate
Threshold control — wireless radios operate at two different power levels: \(0 < B_t \leq B^*\) (low) or \(B_t > B^*\) (high)

\[
r(A_l(t), A_h(t)) := (1_{0 < B_t \leq B^*} \beta_l + 1_{B_t > B^*} \beta_h) \frac{A_l(t) + A_h(t) - 1}{N}
\]
Wireless sensor network model

- Nodes are in one of two discrete states: active (a) or idle (i)
- Idle nodes are awaiting stimulus, active nodes are exchanging collected data with a neighbour
- Each has own battery, drains at a state-dependent rate
- Threshold control — wireless radios operate at two different power levels: \(0 < B_t \leq B^*\) (low) or \(B_t > B^*\) (high)

\[
\begin{align*}
\text{Presumably } \beta_l &\leq \beta_h, \quad \tau_l \leq \tau_h \\
\end{align*}
\]
Wireless sensor network model

Nodes are in one of two discrete states: active (a) or idle (i).
- Idle nodes are awaiting stimulus, active nodes are exchanging collected data with a neighbour.
- Each has own battery, drains at a state-dependent rate.
- Threshold control — wireless radios operate at two different power levels: $0 < B_t \leq B^*$ (low) or $B_t > B^*$ (high).

$$r(A_l(t), A_h(t)) := 1_{0 < B_t \leq B^*} \beta_l \frac{\tau_l (A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{B_t > B^*} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N}$$

Presumably $\beta_l \leq \beta_h$, $\tau_l \leq \tau_h$ and $\gamma_h \leq \gamma_l \leq \gamma_i \leq 0$. 
Mean-field PDEs

\[
r(A_l(t), A_h(t)) := 1_{0 < B_t \leq B^*} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{B_t > B^*} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N}
\]

\[
\frac{dB_t}{dt} = \begin{cases} 
\gamma_i : \text{state } i, 0 < B_t \\
\gamma_l : \text{state } a, 0 < B_t \leq B^* \\
\gamma_h : \text{state } a, B_t > B^* \\
0 : \text{otherwise}
\end{cases}
\]

\[
\implies F_a(t, z), F_i(t, z): \text{proportion of nodes in } a \text{ or } i, \text{ battery } \leq z
\]

Mean-field PDEs

\[
\begin{aligned}
\frac{dB_t}{dt} &= \\
&= \begin{cases}
\gamma_i : \text{state i, } 0 < B_t \\
\gamma_l : \text{state a, } 0 < B_t \leq B^* \\
\gamma_h : \text{state a, } B_t > B^* \\
0 : \text{otherwise}
\end{cases}
\end{aligned}
\]

\[
r(A_l(t), A_h(t)) := 1\{0 < B_t \leq B^*\} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1\{B_t > B^*\} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N}
\]

\[
1\{B_t > 0\} \lambda \quad \text{r}(A_l(t), A_h(t))
\]

\[
\Rightarrow \quad f_a(t, z) := \frac{\partial}{\partial z} F_a(t, z), \quad f_i(t, z) := \frac{\partial}{\partial z} F_i(t, z) \text{ for } z \in (0, 1]
\]

Mean-field PDEs

\[
\begin{align*}
\lambda_1 \gamma_l &:= 1_{\{0 < B_t \leq B^*\}} \beta_l \gamma_l [B_t - 1] + 1_{\{B_t > B^*\}} \beta_h \gamma_h [B_t - 1] \\
\lambda_2 \gamma_l &:= 1_{\{B_t > 0\}} \lambda_1 \gamma_l \\
\lambda_3 \gamma_l &:= \frac{\partial}{\partial z} \mathcal{F}_l(t, z)
\end{align*}
\]

\[
\begin{align*}
\frac{dB_t}{dt} &= \begin{cases} 
\gamma_l : \text{state } l, 0 < B_t \\
\gamma_l : \text{state } l, 0 < B_t \leq B^* \\
\gamma_h : \text{state } h, B_t > B^* \\
0 : \text{otherwise}
\end{cases}
\end{align*}
\]

\[
\begin{align*}
f_a(t, z) &:= \frac{\partial}{\partial z} \mathcal{F}_a(t, z), f_i(t, z) := \frac{\partial}{\partial z} \mathcal{F}_i(t, z) \text{ for } z \in (0, 1)
\end{align*}
\]

\[
\begin{align*}
f_a(t + \delta t, z) &\approx f_a(t, z) + (f_a(t, z) + [1_{\{z \leq B^*\}} \gamma_l + 1_{\{z > B^*\}} \gamma_h] \delta t) - f_a(t, z) \\
&\quad + \lambda \delta t f_i(t, z)
\end{align*}
\]

\[
\begin{align*}
&\quad + (1_{\{z \leq B^*\}} \beta_l + 1_{\{z > B^*\}} \beta_h) \delta t f_a(t, z) \left( \tau_l \int_0^{B^*} f_a(t, v) \, dv + \tau_h \int_{B^*}^1 f_a(t, v) \, dv \right) + o(\delta t)
\end{align*}
\]

Discharging of batteries in \([t, t+\delta t]\)

Discrete transitions \(i \rightarrow a\) in \([t, t+\delta t]\)

Discrete transitions \(a \rightarrow i\) in \([t, t+\delta t]\)
Mean-field PDEs

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_l (A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \]

\[ \frac{dB_t}{dt} = \begin{cases} \gamma_i : \text{state } i, \ 0 < B_t \\ \gamma_l : \text{state } a, \ 0 < B_t \leq B^* \\ \gamma_h : \text{state } a, \ B_t > B^* \\ 0 : \text{otherwise} \end{cases} \]

\[ f_a(t, z) := \frac{\partial}{\partial z} F_a(t, z), \ f_i(t, z) := \frac{\partial}{\partial z} F_i(t, z) \text{ for } z \in (0, 1] \]

\[ \frac{\partial f_a(t, z)}{\partial t} - (1_{\{z \leq B^*\}} \gamma_l + 1_{\{z > B^*\}} \gamma_h) \frac{\partial f_a(t, z)}{\partial z} = \]

\[ \lambda f_i(t, z) - (1_{\{z \leq B^*\}} \beta_l + 1_{\{z > B^*\}} \beta_h) f_a(t, z) \left( \tau_l \int_0^{B^*} f_a(t, v) \, dv + \tau_h \int_{B^*}^1 f_a(t, v) \, dv \right) \]

Mean-field PDEs

\[
\begin{align*}
\frac{dB_t}{dt} &= \begin{cases} 
\gamma_i & : \text{state } i, \ 0 < B_t \\
\gamma_l & : \text{state } a, \ 0 < B_t \leq B^* \\
\gamma_h & : \text{state } a, \ B_t > B^* \\
0 & : \text{otherwise}
\end{cases} \\
1_{\{B_t > 0\}} \lambda & \quad r(A_l(t), A_h(t)) \\
1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} & + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N}
\end{align*}
\]

- \( e_a(t), e_i(t) \): proportion of nodes in a or i with empty battery

\[
e_a(t + \delta t) \approx e_a(t) + \int_0^{\gamma_l \delta t} f_a(t, \nu) \, d\nu + o(\delta t)
\]

Discharging of batteries in \([t, t + \delta t]\)

Mean-field PDEs

\[
\begin{align*}
    r(A_l(t), A_h(t)) &:= 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \\
    dB_t &\quad = \begin{cases} 
    \gamma_i & : \text{state } i, \ 0 < B_t \\
    \gamma_i & : \text{state } a, \ 0 < B_t \leq B^* \\
    \gamma_h & : \text{state } a, \ B_t > B^* \\
    0 & : \text{otherwise}
    \end{cases}
\end{align*}
\]

\[\begin{align*}
    de_a(t), \ e_i(t) : \text{proportion of nodes in } a \text{ or } i \text{ with empty battery}
\end{align*}\]

\[
\frac{de_a(t)}{dt} = \gamma_i f_a(t, 0)
\]
Mean-field PDEs

\[
\begin{align*}
r(A_l(t), A_h(t)) & := 1\{0 < B_t \leq B^*\} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1\{B_t > B^*\} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \\
1\{B_t > 0\} \lambda & \quad r(A_l(t), A_h(t))
\end{align*}
\]

- System of two non-linear partial (functional) differential equations with ordinary differential equations capturing the mass at zero

Mean-field PDEs

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\gamma_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\gamma_h(A_l(t) + \tau_h(A_h(t) - 1)}{N} \]

\[ dB_t = \begin{cases} 
\gamma_i : \text{state } i, 0 < B_t \\
\gamma_l : \text{state } a, 0 < B_t \leq B^* \\
\gamma_h : \text{state } a, B_t > B^* \\
0 : \text{otherwise}
\end{cases} \]

- System of two non-linear partial (functional) differential equations with ordinary differential equations capturing the mass at zero
- Specify initial conditions at \( t = 0 \) and also boundary conditions \( f_a(t, 1) = f_i(t, 1) = 0 \) for \( t > 0 \)

Example solutions

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \]

\[ 1_{\{B_t > 0\}} \lambda r(A_l(t), A_h(t)) \]

\[ \frac{dB_t}{dt} = \begin{cases} 
\gamma_i & \text{state i, } 0 < B_t \\
\gamma_l & \text{state a, } 0 < B_t \leq B^* \\
\gamma_h & \text{state a, } B_t > B^* \\
0 & \text{otherwise}
\end{cases} \]
Example solutions

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_A(t) - 1 + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_A(t) + \tau_h (A_h(t) - 1)}{N} \]

\[ 1_{\{B_t > 0\}} \lambda r(A_l(t), A_h(t)) \]

\[ dB_t = \begin{cases} 
\gamma_i : \text{state } i, \ 0 < B_t \\
\gamma_l : \text{state } a, \ 0 < B_t \leq B^* \\
\gamma_h : \text{state } a, \ B_t > B^* \\
0 : \text{otherwise} 
\end{cases} \]

\[ N = 10 \]
Example solutions

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \]

\[ dB_t = \begin{cases} \gamma_i : \text{state i, } 0 < B_t \\ \gamma_l : \text{state a, } 0 < B_t \leq B^* \\ \gamma_h : \text{state a, } B_t > B^* \\ 0 : \text{otherwise} \end{cases} \]

\[ N = 20 \]
Example solutions

\[ r(A_l(t), A_h(t)) := 1_{0 < B_t \leq B^*} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{B_t > B^*} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \]

\[
1_{B_t > 0} \lambda | r(A_l(t), A_h(t)) |
\]

\[
\frac{dB_t}{dt} = \begin{cases} 
\gamma_i : \text{state i}, 0 < B_t \\
\gamma_l : \text{state a}, 0 < B_t \leq B^* \\
\gamma_h : \text{state a}, B_t > B^* \\
0 : \text{otherwise}
\end{cases}
\]

\[ N = 100 \]
Example solutions

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \beta_l \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \beta_h \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \]

\[ 1_{\{B_t > 0\}} \lambda \]

\[ dB_t = \begin{cases} \gamma_i : \text{state } i, \ 0 < B_t \\ \gamma_l : \text{state } a, \ 0 < B_t \leq B^* \\ \gamma_h : \text{state } a, \ B_t > B^* \\ 0 : \text{otherwise} \end{cases} \]

\[ N = 1000 \]
Example solutions

\[ r(A_l(t), A_h(t)) := 1_{\{0 < B_t \leq B^*\}} \frac{\tau_l(A_l(t) - 1) + \tau_h A_h(t)}{N} + 1_{\{B_t > B^*\}} \frac{\tau_l A_l(t) + \tau_h (A_h(t) - 1)}{N} \]

\[ 1_{\{B_t > 0\}} \lambda \]

\[ \gamma_i : \text{state } i, 0 < B_t \]
\[ \gamma_l : \text{state } a, 0 < B_t \leq B^* \]
\[ \gamma_h : \text{state } a, B_t > B^* \]
\[ 0 : \text{otherwise} \]

Compared to single trace

<table>
<thead>
<tr>
<th>Pop. size</th>
<th>Avg. error</th>
<th>Max. error</th>
</tr>
</thead>
<tbody>
<tr>
<td>N = 10</td>
<td>0.1425</td>
<td>0.3540</td>
</tr>
<tr>
<td>N = 100</td>
<td>0.0264</td>
<td>0.1212</td>
</tr>
<tr>
<td>N = 1000</td>
<td>0.0138</td>
<td>0.0345</td>
</tr>
<tr>
<td>N = 10000</td>
<td>0.0042</td>
<td>0.0336</td>
</tr>
</tbody>
</table>

Compared to mean of 1000 traces

<table>
<thead>
<tr>
<th>Pop. size</th>
<th>Avg. error</th>
<th>Max. error</th>
</tr>
</thead>
<tbody>
<tr>
<td>N = 10</td>
<td>0.0307</td>
<td>0.0577</td>
</tr>
<tr>
<td>N = 100</td>
<td>0.0042</td>
<td>0.0234</td>
</tr>
<tr>
<td>N = 1000</td>
<td>Simulation too costly</td>
<td></td>
</tr>
<tr>
<td>N = 10000</td>
<td>Simulation too costly</td>
<td></td>
</tr>
</tbody>
</table>
Thank you!

... and also to Anton Stefanek and Jeremy Bradley who were involved in this research