Imperial College London

LN: a Meta-Solver for Layered Queueing Network Analysis

Giuliano Casale, Yicheng Gao, Zifeng Niu, and Lulai Zhu

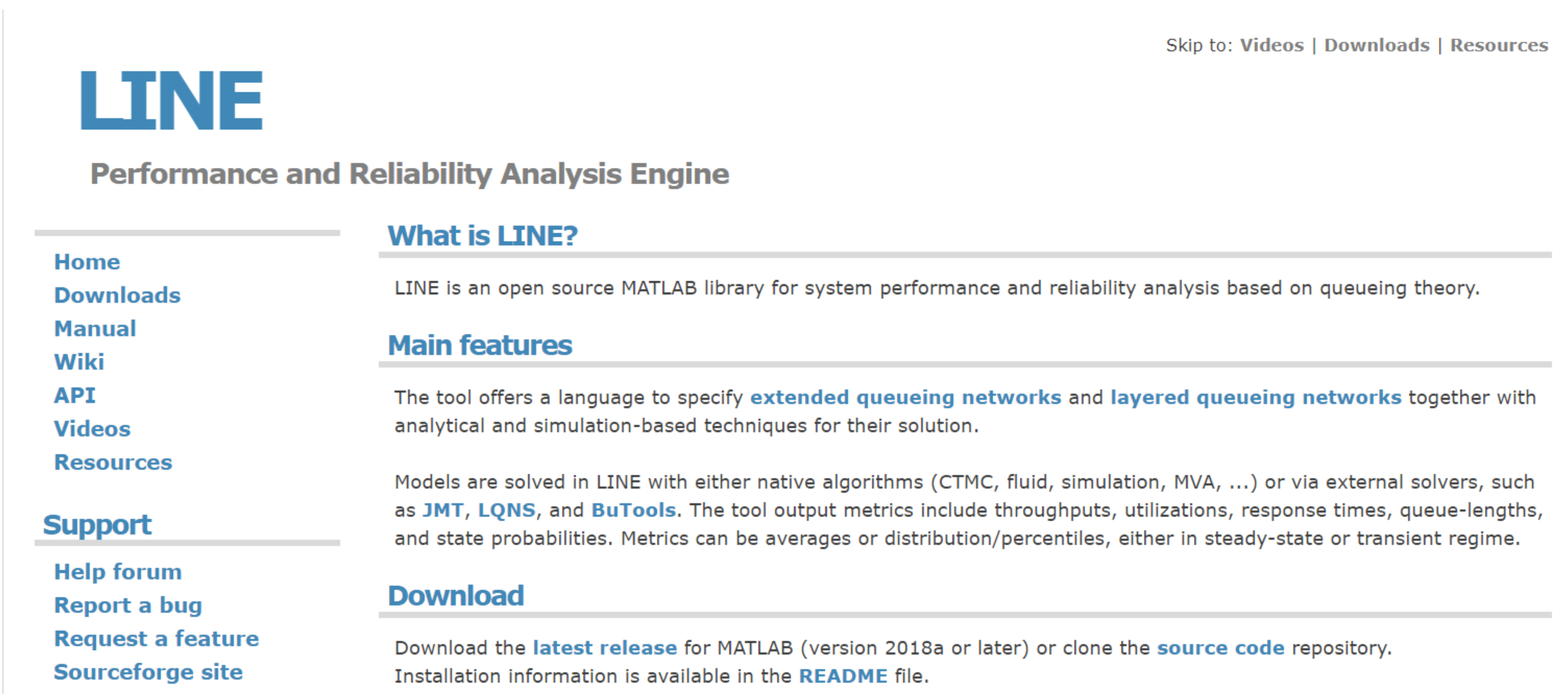
QORE Research Lab (qore.doc.ic.ac.uk)
Department of Computing
Imperial College London

14-Sep-2022, QEST, Warsaw, Poland

Imperial College London

LINE Solver (line-solver.sf.net)

- MATLAB/Java library for system performance and reliability analysis based on queueing theory
- Ver 2.0.0+ (BSD-3): multiple solution paradigms



Skip to: Videos | Downloads | Resources

LINE

Performance and Reliability Analysis Engine

Home
Downloads
Manual
Wiki
API
Videos
Resources

Support

Help forum
Report a bug
Request a feature
Sourceforge site

What is LINE?

LINE is an open source MATLAB library for system performance and reliability analysis based on queueing theory.

Main features

The tool offers a language to specify extended queueing networks and layered queueing networks together with analytical and simulation-based techniques for their solution.

Models are solved in LINE with either native algorithms (CTMC, fluid, simulation, MVA, ...) or via external solvers, such as JMT, LQNS, and BuTools. The tool output metrics include throughputs, utilizations, response times, queue-lengths, and state probabilities. Metrics can be averages or distribution/percentiles, either in steady-state or transient regime.

Download

Download the latest release for MATLAB (version 2018a or later) or clone the source code repository. Installation information is available in the README file.

Imperial College London

LINE: What's in it?

I. Object-oriented language to model extended and layered queueing networks (QNs / LQNs)

II. Several analysis paradigms: Fluid ODEs, MVA, Norm. Const., DES (JMT), SSA, CTMC, MAM, …

III. Seamless integration with JMT, LQNS, Q-MAM, BuTools, KPC-Toolbox, …

LINE in numbers:

- 70+ algorithms
- 40+ types of analyses
- 23 sched./routing strategies
- 125+ lang. classes
- 17 node types
- 7+ metrics

Rest of this talk

- Layered Queueing Networks (LQNs): theory & tools
- Advanced LQN analysis methods within LN
- Novel multichain QN solution algorithms used in LN

Rest of this talk

- Layered Queueing Networks (LQNs): theory & tools

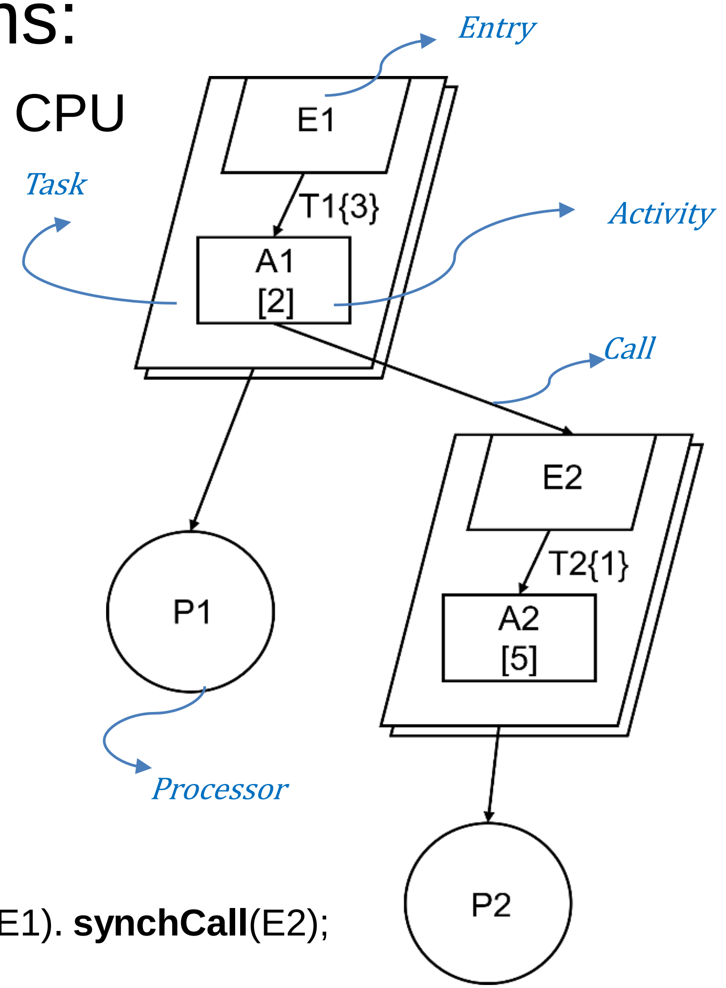
- Advanced LQN analysis methods within LN

- Novel multichain QN solution algorithms used in LN

Layered Queueing Networks (LQNs)

Abstractions of layered systems:

- Processors: hardware resources, e.g., CPU
- Tasks: software resources
- Entries: job/service classes
- Activities: unit operations



LQNs in LINE:

model = LayeredNetwork('myModel');

P1 = Processor(model, 'P1', 1, SchedStrategy.PS);

T1 = Task(model, 'T1', 1, SchedStrategy.REF).on(P1);

E1 = Entry(model, 'E1').on(T1);

…

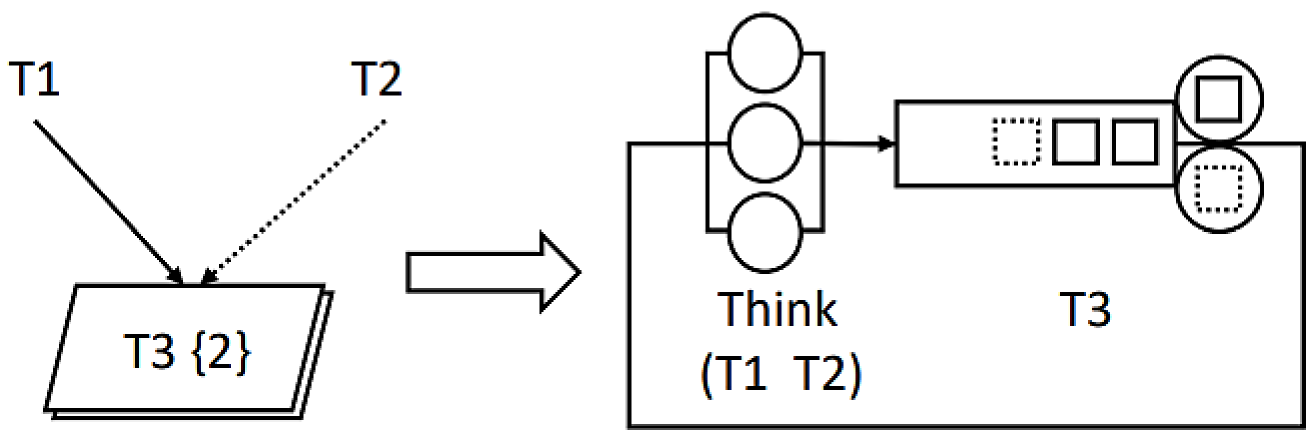
A1 = Activity(model, 'A1', Exp.fitMean(0.1)).on(T1).boundTo(E1). **synchCall**(E2);

Steady-state analysis

- LQNS [Franks et al., TSE'09]
  - Analyzes LQN layers using Linearizer and other AMVA/MVA algorithms.
  - Wrapper available in LINE to run LQNS.
- DiffLQN [Waizmann & Tribastone, ICPE'16]
  - A solver based on the mean-field fluid approximation theory developed in the context of PEPA models for scalable analysis of LQNs
- LN is a meta-solver, parametric in the layer solver:
  - Fluid, MVA, NC, JMT, SSA, CTMC, MAM, …
  - Broader focus than mean performance metrics

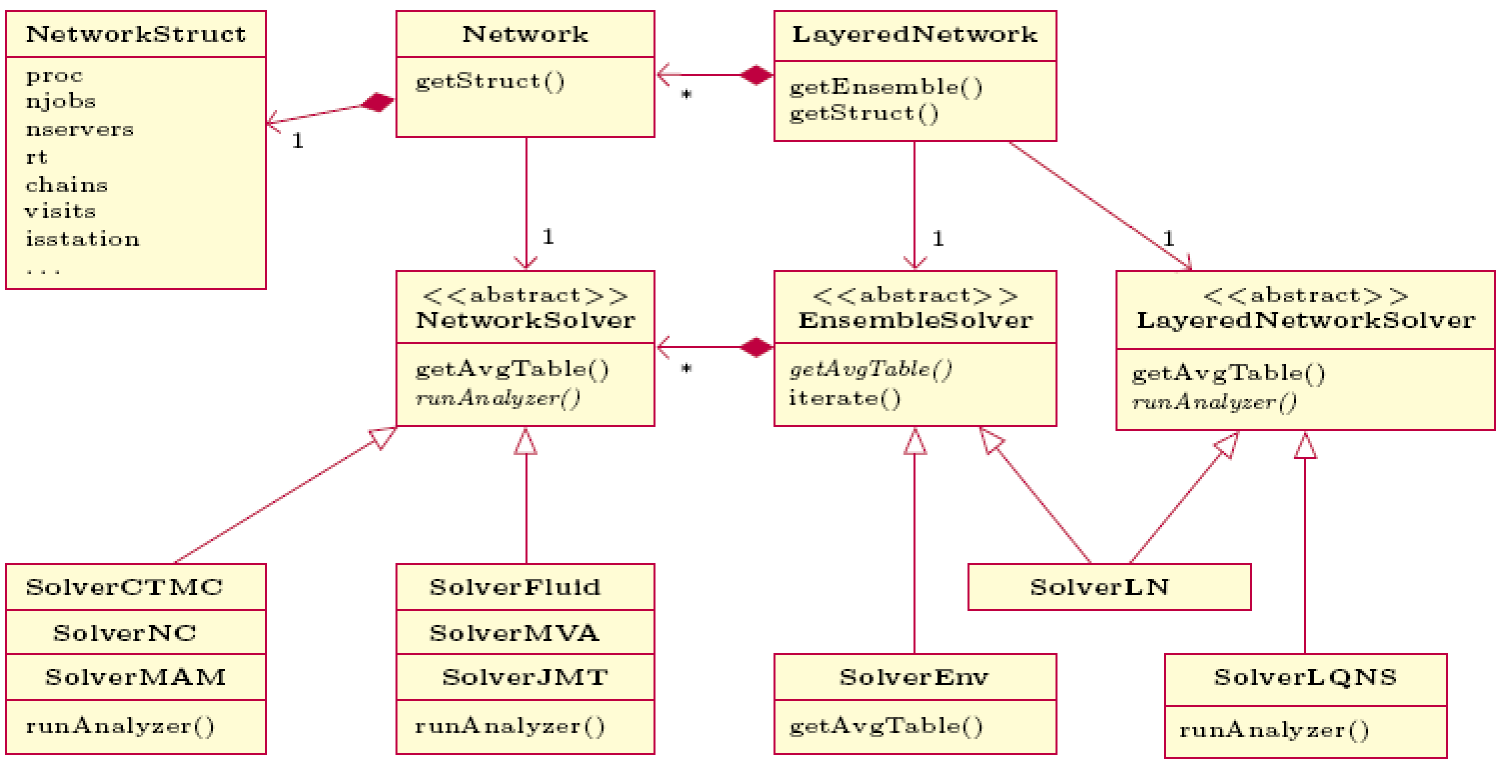
LQN Loose layering

- LN treats layers as composed by a single task:
  - Motivated by computational complexity
  - A layer includes a delay and *m* identical queues
  - We call this an homogeneous layer
- Proposed in SRVN models, LQN decomposition styles known to have marginal effect on accuracy



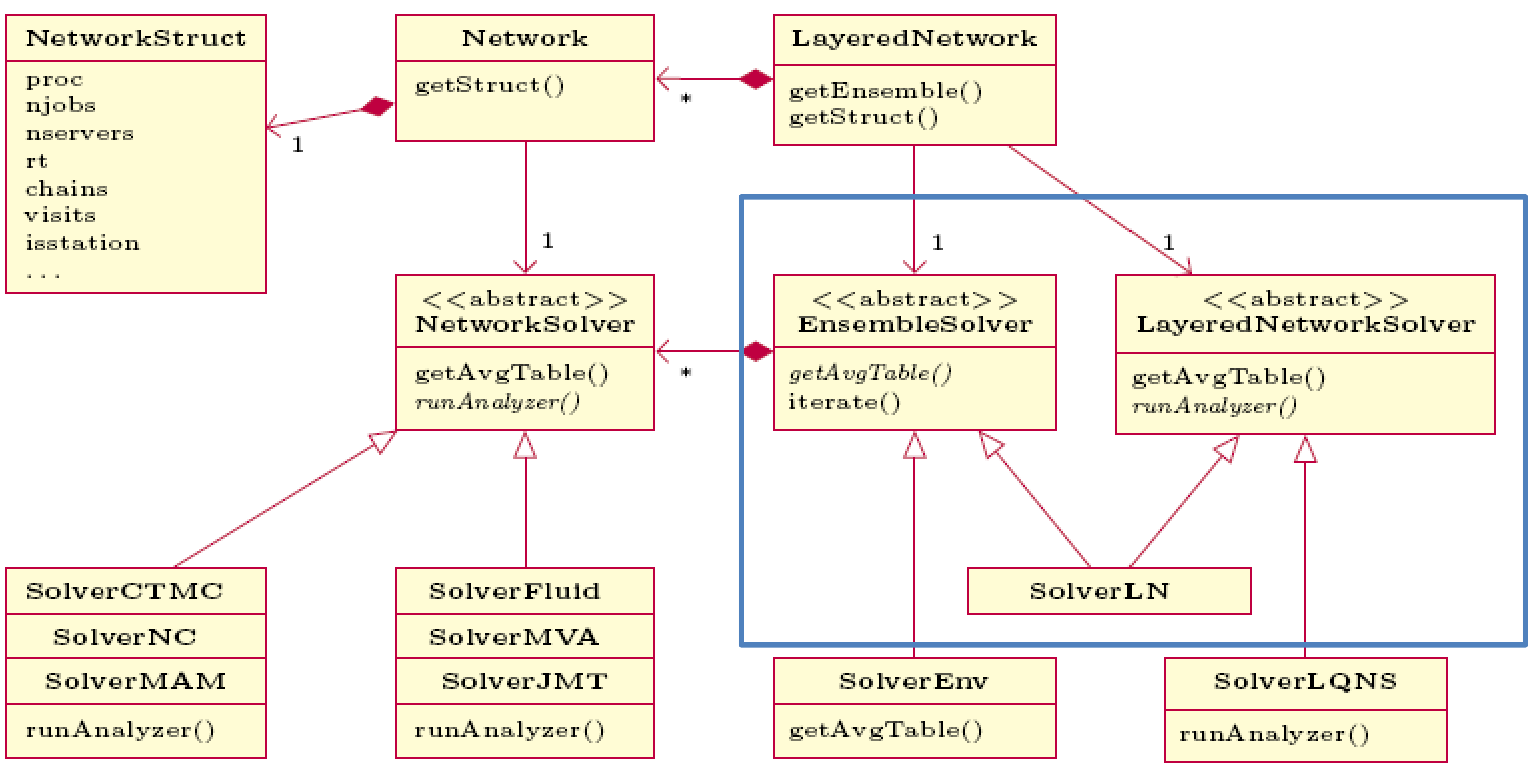
LINE Architecture Highlights

- Ensemble Solvers for collections of sub-models
  - Layers mapped into set of interacting QNs
  - QN mapped to static structure (faster in MATLAB)



LINE Architecture Highlights

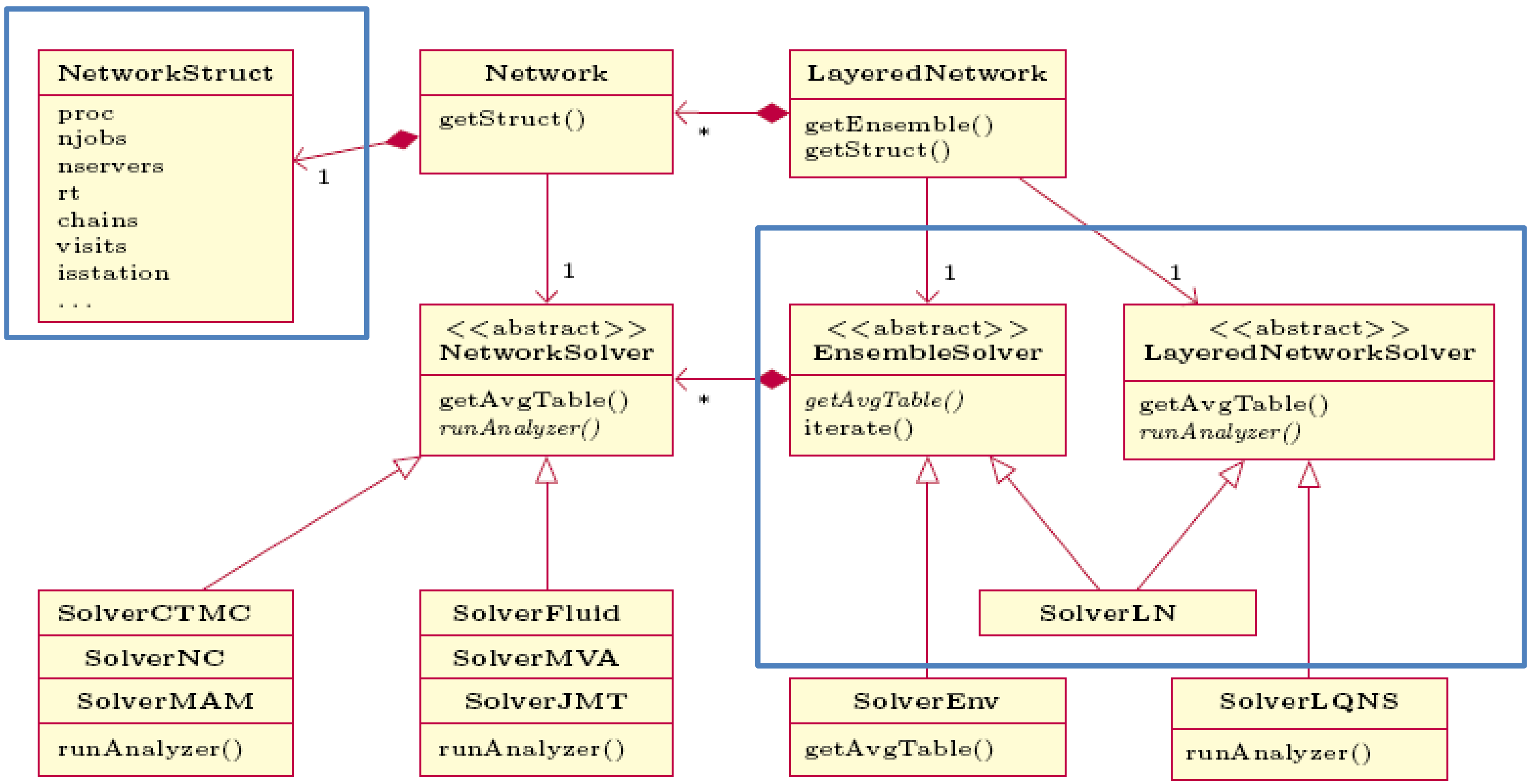
- Ensemble Solvers for collections of sub-models
  - Layers mapped into set of interacting QNs
  - QN mapped to static structure (faster in MATLAB)



Imperial College London

LINE Architecture Highlights

- Ensemble Solvers for collections of sub-models
  - Layers mapped into set of interacting QNs
  - QN mapped to static structure (faster in MATLAB)



Rest of this talk

- Layered Queueing Networks (LQNs): theory & tools
- Advanced LQN analysis methods within LN
- Novel multichain QN solution algorithms used in LN

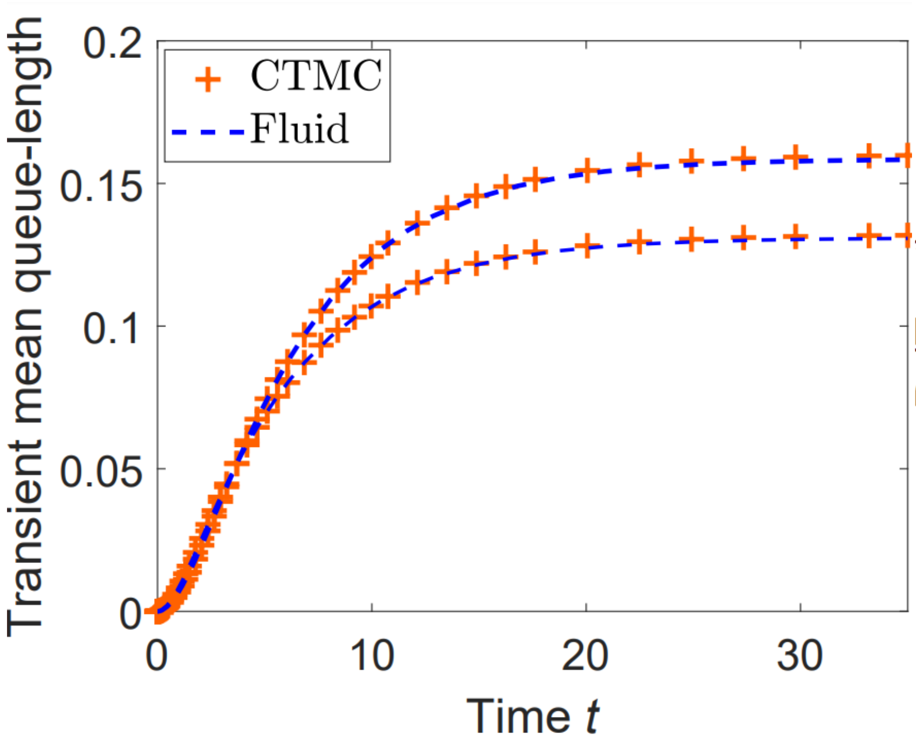
Rest of this talk

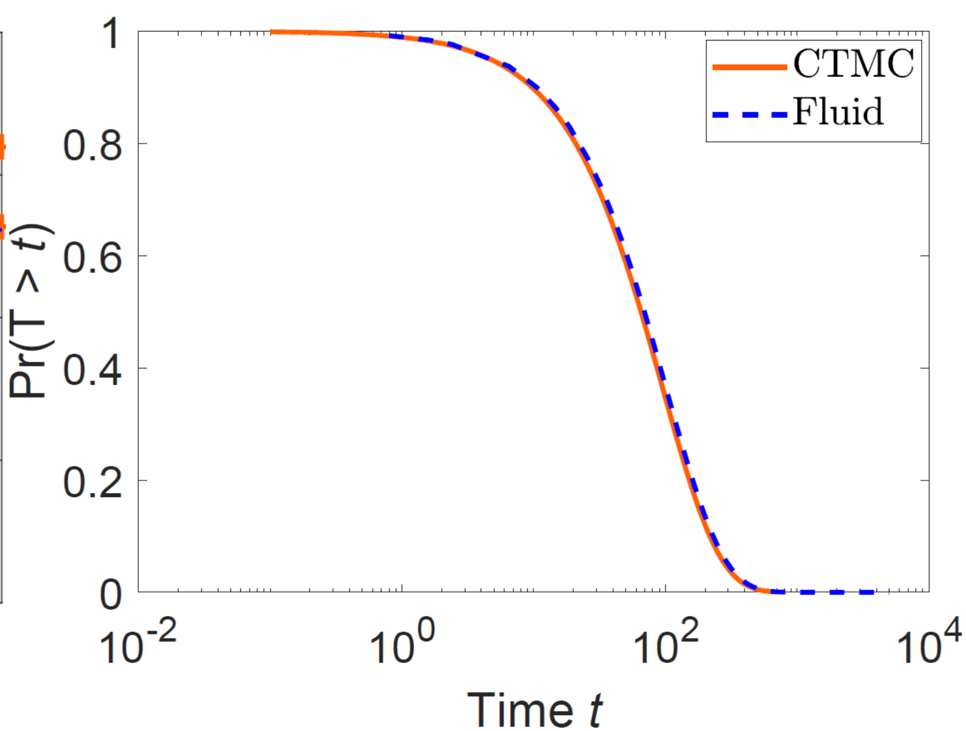
- Layered Queueing Networks (LQNs): theory & tools
- Advanced LQN analysis methods within LN
- Novel multichain QN solution algorithms used in LN

Imperial College London

Fluid transient approximation

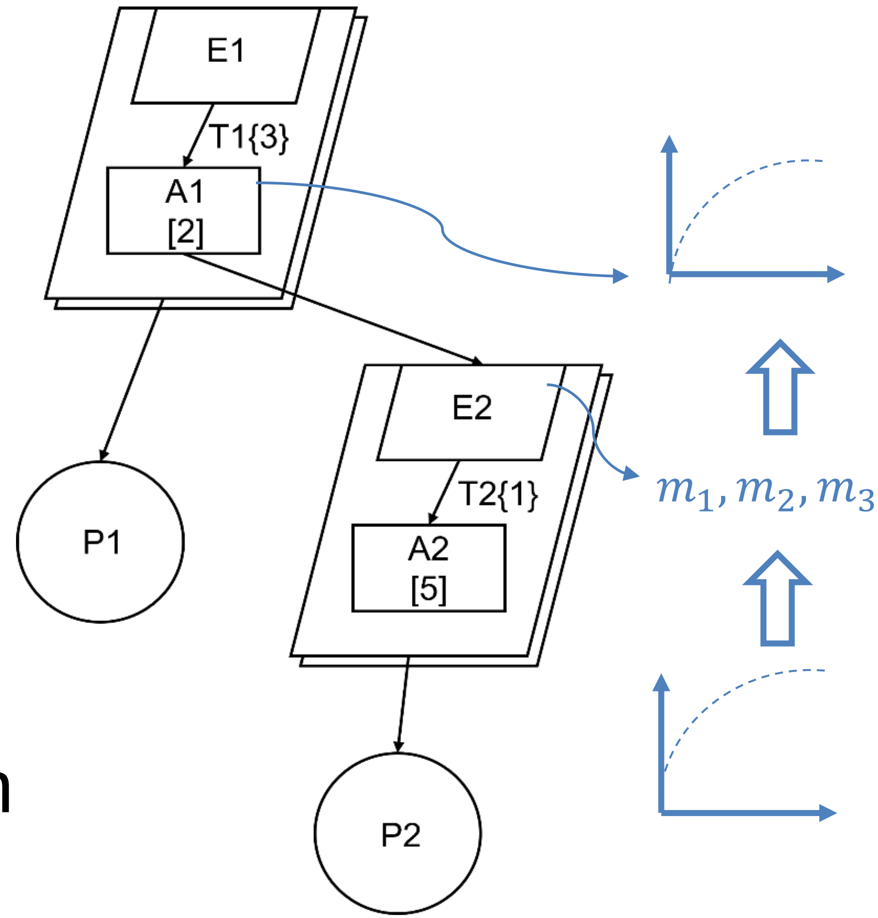
- Multi-chain transient analysis:
  - PS/DPS handled via Kurtz's theorem
  - FCFS approximated as PS using a hybrid Diffusion-M/G/k fixed-point iteration [WSC'20]





Response time distribution analysis

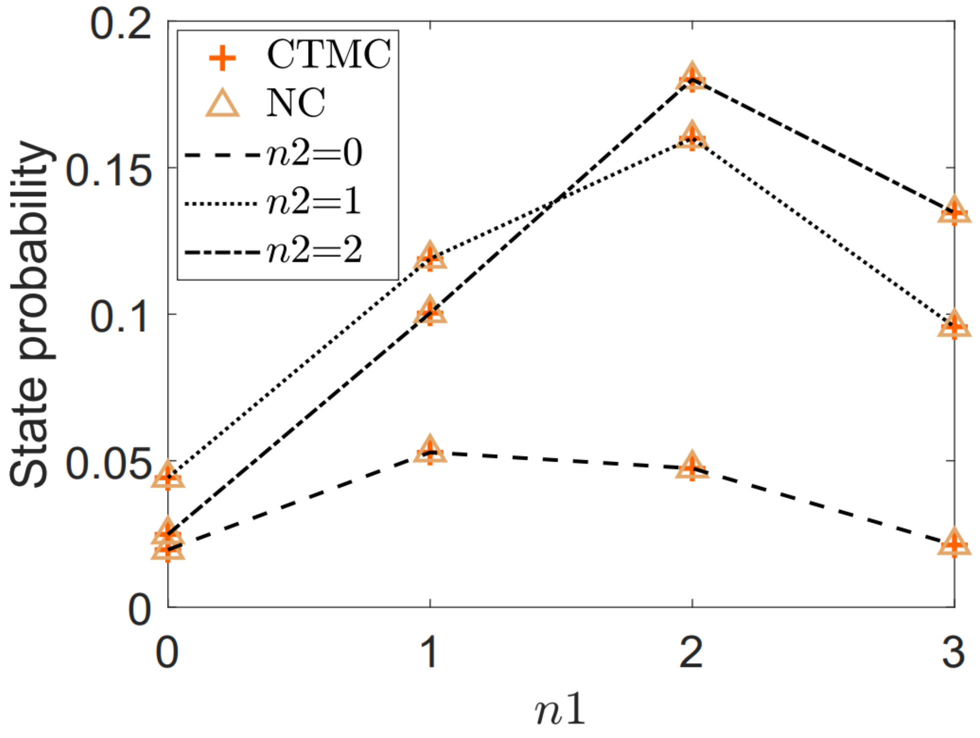
- LN obtains response time CDF in layers via Fluid, CTMC or JMT
- Recursive fitting of mean, variance, and skewness of resp.t. with APH models
- Performing convolutions on APHs to parameterize calling layers



Imperial College
London

Marginal state probabilities

- Normalizing constant based approximation of marginal / joint state probabilities in a layer

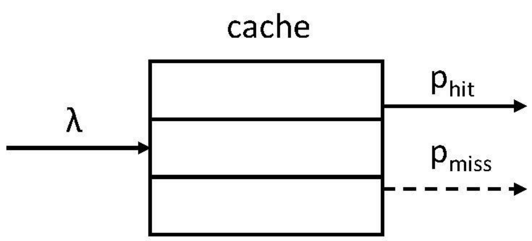


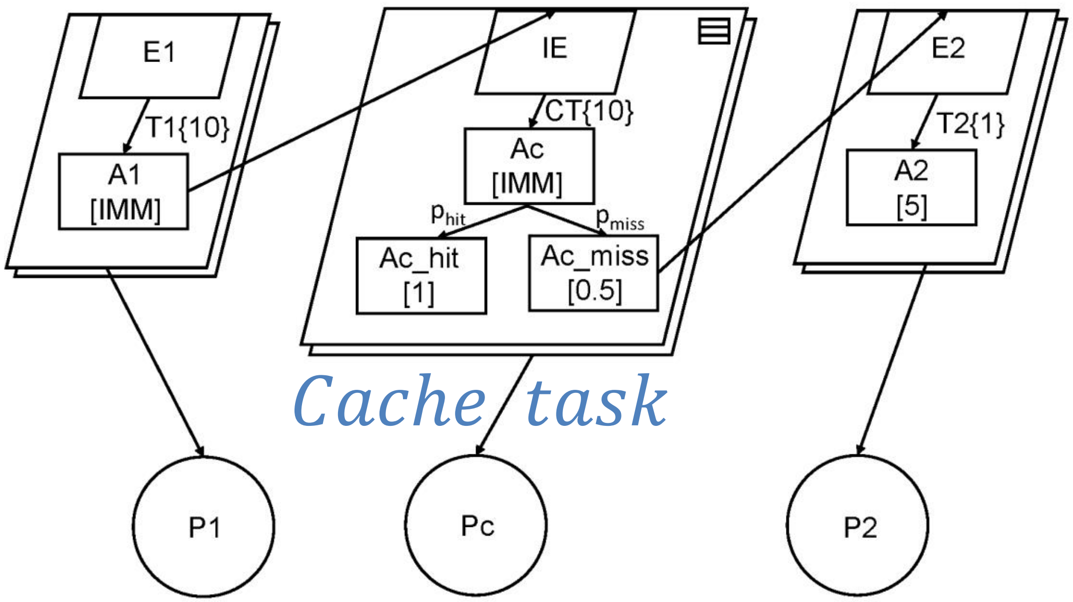
Imperial College London

LQN Extension: Layers with Caching

LN solver supports caching [IWQoS'21]:

- Caches in an LQN using specialized tasks/entries
- Cache miss/hit influence activity workflow
- Various replacement policies: FIFO, Random, LRU

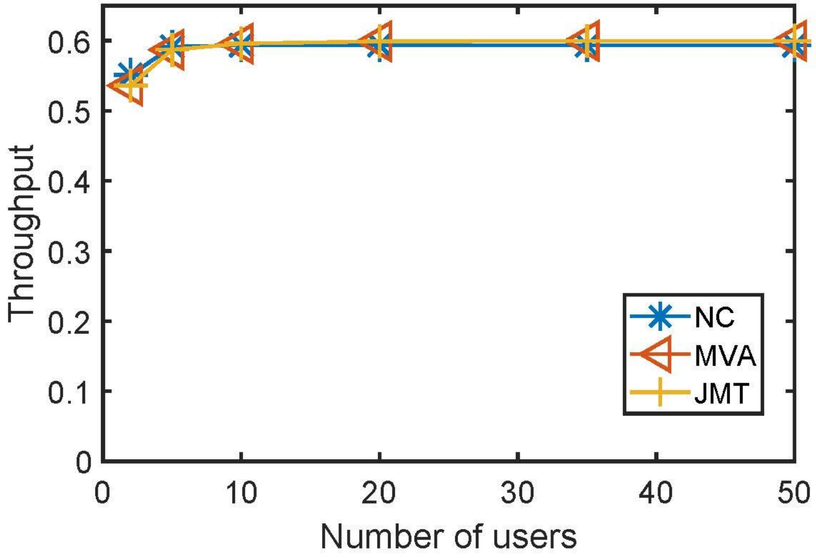




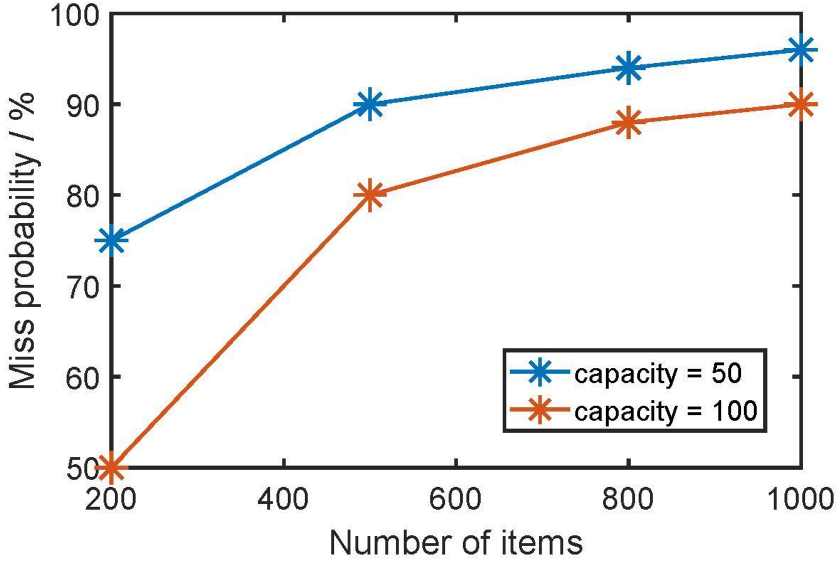
Imperial College London

Example: Caching Formalism

- Model: three-layer LQN model with caching
  - Uniform access popularity to items
  - Random replacement (RR) strategy
  - Analytical very close to simulation [ToN'21]



MVA and NC solutions



Miss ratio analysis

Imperial College London

Rest of this talk

- Layered Queueing Networks (LQNs): theory & tools
- Advanced LQN analysis methods within LN
- Novel multichain QN solution algorithms used in LN

Rest of this talk

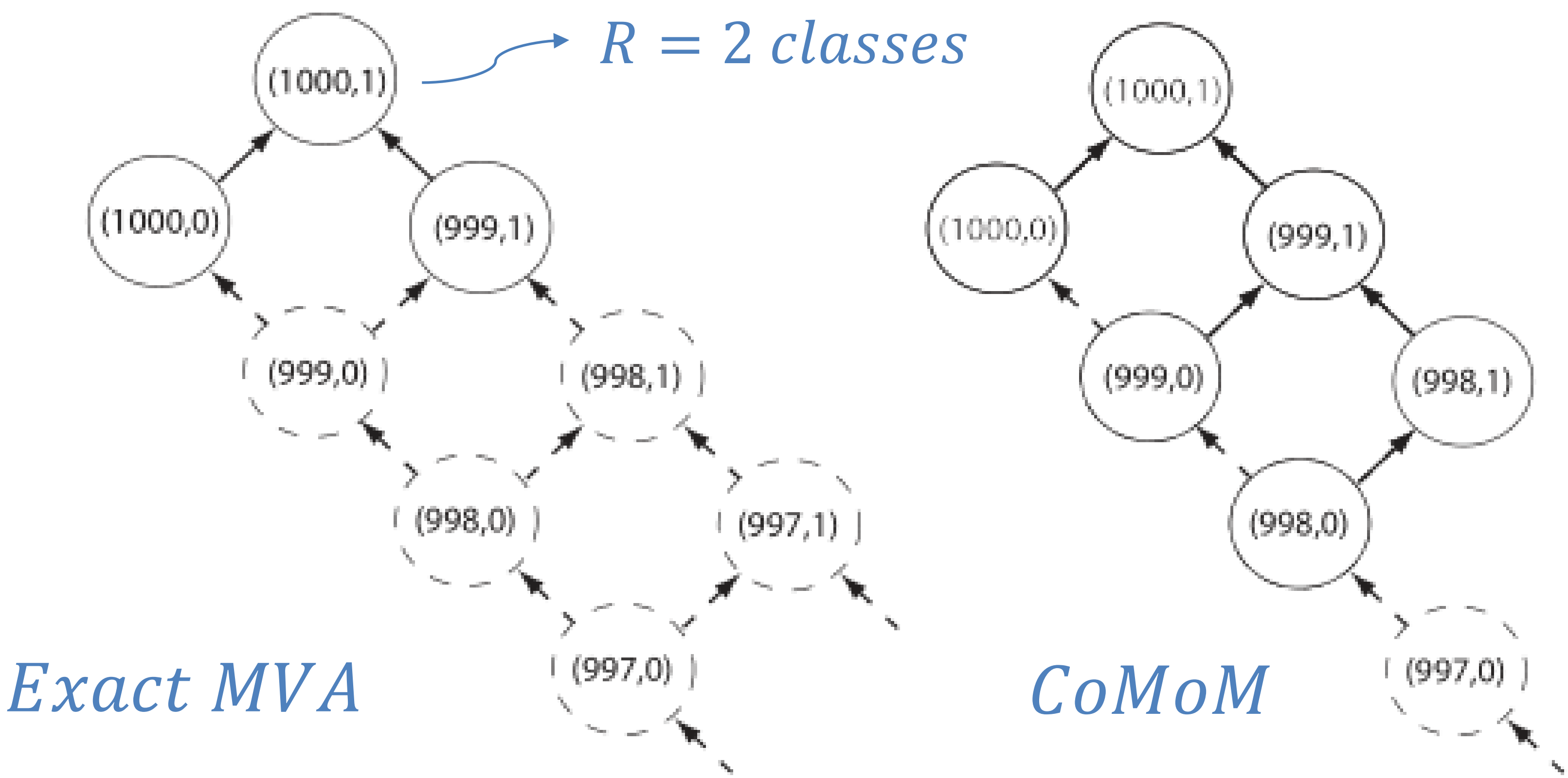
- Layered Queueing Networks (LQNs): theory & tools
- Advanced LQN analysis methods within LN
- Novel multichain QN solution algorithms used in LN

Imperial College London

Method of Moments

A subset of models is solved simultaneously:

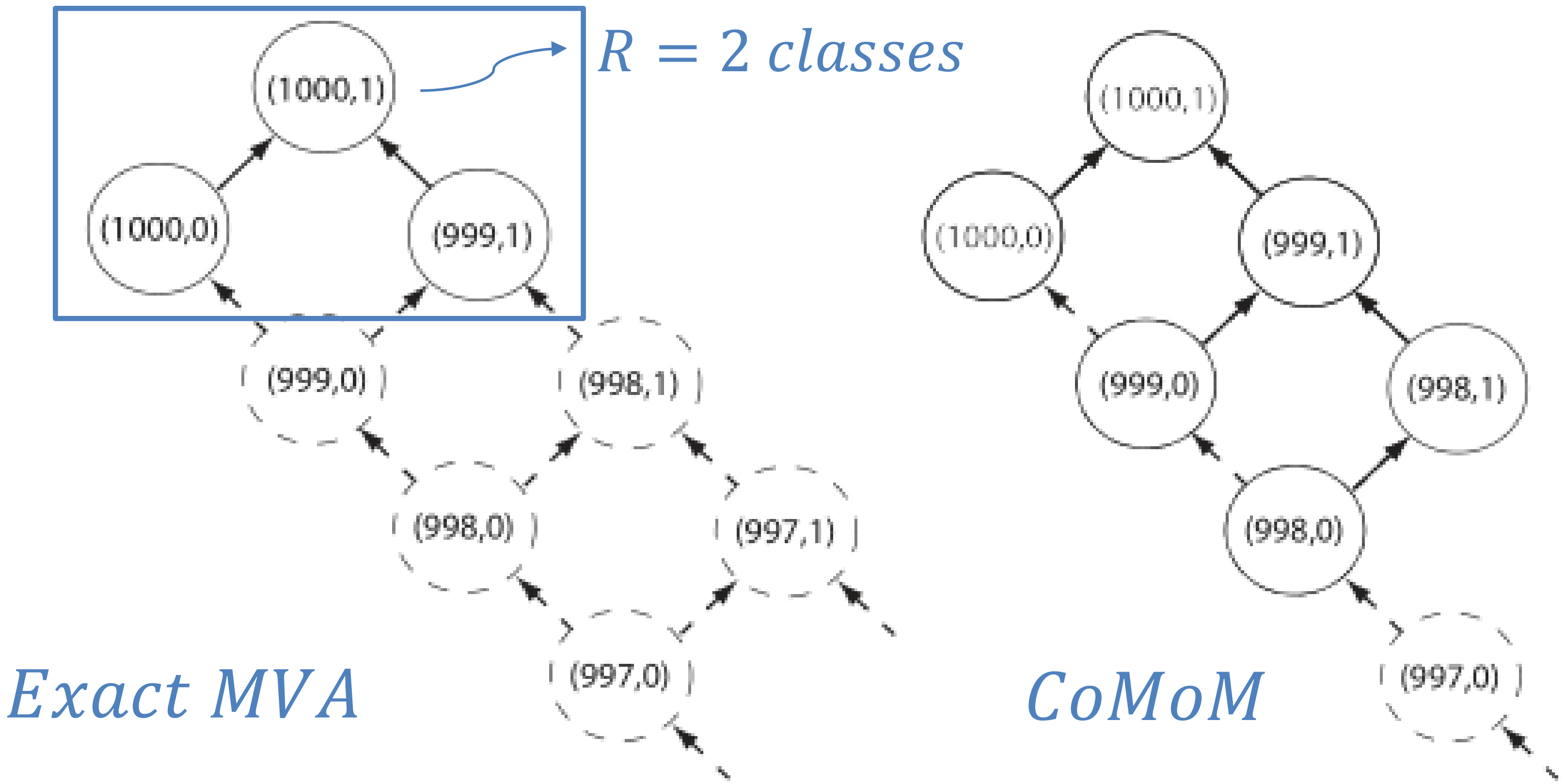
- $O(N \log N)$ with N jobs, *log N* due to stabilization
- Exact MVA is instead $O(N^R)$, with R classes



Method of Moments

A subset of models is solved simultaneously:

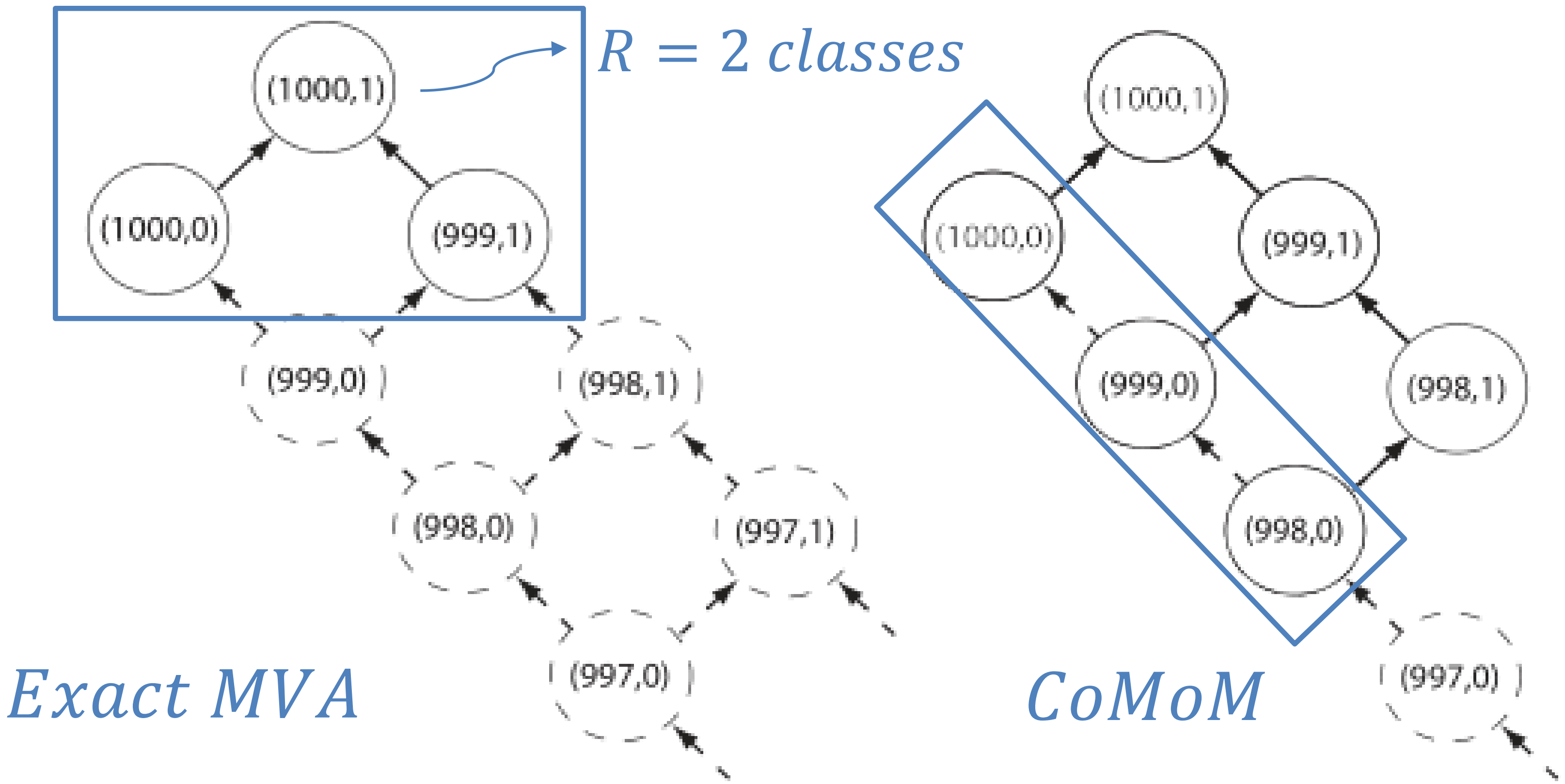
- O(N log N) with N jobs, *log N* due to stabilization
- Exact MVA is instead O(N^R), with R classes



Method of Moments

A subset of models is solved simultaneously:

- O(*N* log *N*) with *N* jobs, *log N* due to stabilization
- Exact MVA is instead O(N^R), with *R* classes

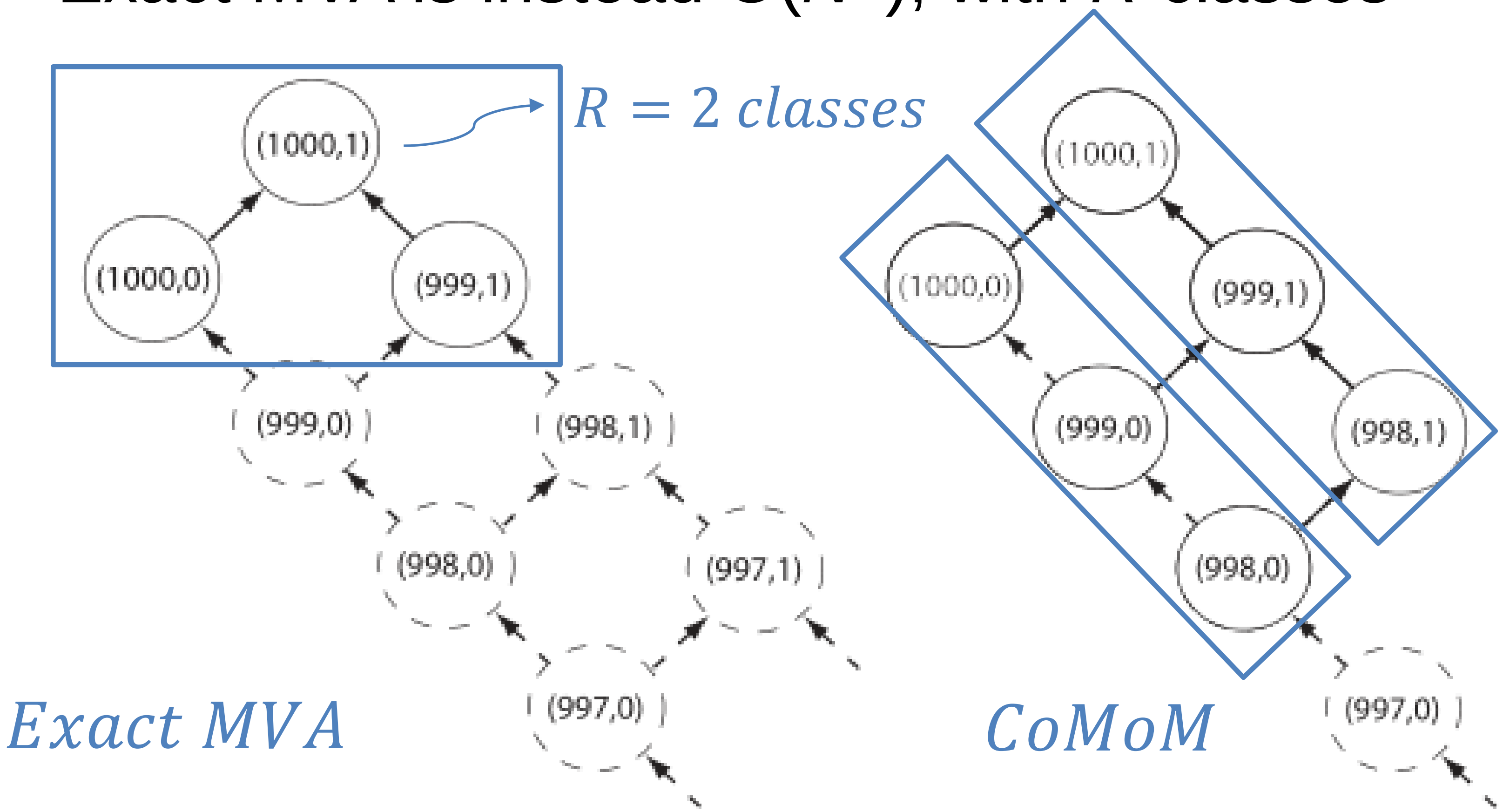


Imperial College London

Method of Moments

A subset of models is solved simultaneously:

- O(N log N) with N jobs, *log N* due to stabilization
- Exact MVA is instead O(N^R), with R classes



Imperial College London

Enhanced CoMoM for LQN layers

Homogenous QN with *m* queues, *R* classes, *N* jobs

CoMoM Basis:

$$\Lambda(N) = \left[g(m+1, N) \quad g(m, N) \right]^T$$

$$g(m, N) = \left[G(m, N)\ G(m, N-1_1) \cdots G(m, N-1_{R-1}) \right]$$

Enhanced CoMoM recurrence relation:

$$\Lambda(N) = (\boldsymbol{F}_{1,R} + N_R^{-1} \boldsymbol{F}_{2,R}) \Lambda(N - 1_R)$$

Explicit formula, i.e., no need for a system of linear eq.

Cf. paper for details and extensions to marginal prob.

Enhanced CoMoM Results

Enhanced CoMoM is observed to give stable results.

Classes	*Total jobs*	*Method*	*Runtime [s]*
8	40	Convolution	0.0033
8	40	CoMoM (original)	0.0047
8	40	CoMoM (enhanced)	0.0014
8	400	Convolution	1.4201
8	400	CoMoM (original)	1.1433
8	400	CoMoM (enhanced)	0.0016
8	4000	Convolution	Memory exhausted
8	4000	CoMoM (original)	Timeout
8	4000	CoMoM (enhanced)	0.0017
8	10^6	Convolution	Memory exhausted
8	10^6	CoMoM (original)	Timeout
8	10^6	CoMoM (enhanced)	0.2591

Integral Form Approximations

Integral form for homogeneous QNs:

$$G(m,N) = \frac{1}{(m-1)!\prod_{r=1}^{R} N_r!}\int_{u=0}^{+\infty} e^{-u} u^{m-1} \prod_{r=1}^{R} (Z_r + D_r u)^{N_r} du$$

We apply in LN Gaussian quadrature methods:

- Gauss-Laguerre (uses Laguerre polynomial roots)

$$\int_{x=0}^{\infty} e^{-x} f(x) dx \approx \sum_{k=1}^{K} w_k f(x_k) \qquad L_K(x) = \sum_{j=0}^{K} \binom{K}{i} \frac{(-1)^j}{j!} x^j$$

- Gauss-Legendre, similar but for finite domains

Asymptotically, LN uses the logistic expansion [Cas17].

Imperial College London

Integral Form Results

Classes	*Total jobs*	*Method*	*Rel. error [%]*	*Runtime [s]*
8	40	MATLAB's integral	0.0000	0.0006
8	40	Gauss-Legendre	0.0000	0.0004
8	40	Gauss-Laguerre	0.0000	0.0010
8	40	Logistic expansion	-0.1249	0.0012
8	400	MATLAB's integral	0.0144	0.0005
8	400	Gauss-Legendre	-0.0001	0.0006
8	400	Gauss-Laguerre	-0.0001	0.0010
8	400	Logistic expansion	0.0033	0.0013
8	4000	MATLAB's integral	Unstable	0.0008
8	4000	Gauss-Legendre	-0.0006	0.0021
8	4000	Gauss-Laguerre	-0.0006	0.0010
8	4000	Logistic expansion	0.0003	0.0013
8	10^6	MATLAB's integral	Unstable	0.0008
8	10^6	Gauss-Legendre	-0.0592	0.0095
8	10^6	Gauss-Laguerre	0.2508	0.0011
8	10^6	Logistic expansion	0.0000	0.0013

Some lessons learned

- QN approximations mostly very accurate and fast
  - Overheads of sw architecture and programming language often larger than QN/LQN solution time

- LQN a good way for integrating different stochastic formalisms:
  - Inherently suitable for decomposition/aggregation
  - High-level concepts clear to sw&system engineers
  - Many tools for model-to-model transformations
  - Little research beyond traditional MVA

Imperial College London

Future Work

Ongoing work:

- Java migration
- Fluid for mixed models [Ruuskanen et al., PEVA 2021]
- Fork & join approximations (available in LINE 2.0.23)

Long-term extensions:

- Impatience: retrial, balking, reneging
- Control methods (e.g., reinforcement learning based)
- AI/ML methods for inference on QNs

Imperial College London

Bibliography

- [Franks et al., TSE’09] Franks, G., Al-Omari, T., Woodside, M., Das, O., Derisavi, S.: Enhanced modelling and solution of layered queueing networks. IEEE Trans. Softw. Eng. 35(2), 148–161 (2009)
- [IWQoS’21] Gao, Y., Casale, G.: JCSP: Joint caching and service placement for edge computing systems. In: Proc. of IEEE/ACM IWQoS. IEEE (2022)
- [Ruuskanen et al., PEVA 2021] Ruuskanen, J., Berner, T., Arzen, K.E., Cervin, A.: Improving the mean-field fluid model of processor sharing queueing networks for dynamic performance models in cloud computing. PEVA 151 – IFIP Performance 2021 special issue, 102231 (2021).
- [WSC’20] Casale, G.: Integrated performance evaluation of extended queueing network mod els with LINE. In: Proc. of WSC, IEEE (2020)
- [ToN’21] Casale, G., Gast, N.: Performance analysis methods for list-based caches with non- uniform access. IEEE/ACM ToN 29(2), 651–664 (2021)
- [Waizmann & Tribastone, ICPE’16] Waizmann, T., Tribastone, M.: DiffLQN: Differential equation analysis of layered queuing networks. In: Compendium of ICPE. pp. 63–68. ACM (2016)

QNs with Class Switching (multi-chain)

- Jobs circulate among stations switching class
- Chain = subsets of reachable classes for a job type
- Each LQN client follows a workflow (activity graph) in one-to-one mapping with an ergodic chain