2025 MAS course

tatjanapetrov

May 2025

Each project should be reported in a two-page long report (without references). Depending on the chosen project, the report should contain all or most of the following sections:

(i) 'introduction' to the project topic, which will position the project topic and methodology in a wider research and application context, through high-level explanation and references to related works,

(ii) 'preliminaries and specific objectives'; the reader should be equipped with technical terminology and understand *what* specifically you aimed to do in the project,

(iii) 'methods', allowing the reader to understand *how* you approached the project goals,

(iv) 'results', describing the outcome of your project work,

(v) 'discussion and outlook', where you discuss your findings in a wider context, and potential future works.

1. Parameter Identification for DTMCs

Setting: Consider a two-state Discrete-Time Markov Chain (DTMC) modeling the switching behavior of a system between two conditions: A and B. The transition matrix of the system is:

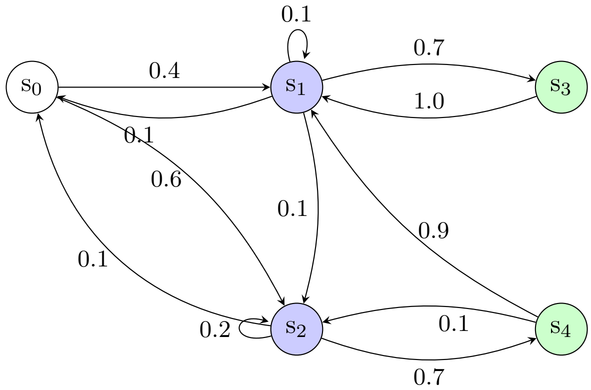
$$P = \begin{bmatrix} 1-p & p \\ q & 1-q \end{bmatrix}$$

We observe the system's evolution over time as a sequence of states, such as A $\rightarrow$ B $\rightarrow$ A $\rightarrow$ A $\rightarrow$ B. These observations arise from repeated runs or monitoring of the system.
Objectives:

- Given a finite number N of finite traces (of length n) produced by the two-state chain (e.g. for $n = 10$, a possible trace is $\sigma_1 = AABBAAAAAA$), estimate the parameters p and q using different parameter inference techniques, e.g. Maximum Likelihood Estimation. Illustrate the estimation process in detail, including the derivation of the likelihood function.
- Discuss how increasing the number of observations affects the quality and confidence of the parameter estimates.
- Evaluate your estimation method on synthetic data where true values of p and q are known.
- Discuss the general challenge of **parameter inference** for parametric DTMC's.

2. Model Reduction for DTMCs

Setting: Consider a Discrete-Time Markov Chain (DTMC) with five states. Assuming you can observe only state labels (depicted by different colors), and traces starting at s_0, states s_1 and s_2 are indistinguishable from each other, and the same holds for states s_3 and s_4. This opens the possibility of reducing the chain to only 3 states. *Lumpability* is a model reduction technique characterising when such reduction is possible.



Lumpable Pairs:

- **s_1 and s_2**
- **s_3 and s_4**

Objectives:

- How does the lumped process look like? Is the resulting process a Markov chain?
- Explain through the theory of **strong lumpability** the reduction of the 5-state model into a 3-state DTMC.
- Show that the lumped DTMC preserves the essential behavior of the original model in terms of long-run distributions or reachability properties.
- Discuss the conditions under which lumpability can be safely applied and how it scales to larger models.

3. Parameter Synthesis for CTMCs

Setting: Rates in CTMC are often uncertain or tunable, and one wishes to have guarantees that the system meets probabilistic performance requirements under rate uncertainty. Consider a CTMC with states A, B, C with a rate matrix:

$$Q = \begin{bmatrix} -r & r & 0 \\ s & -r-s & r \\ 0 & s & -s \end{bmatrix}$$

Can you synthesize values for the parameters r and s so that the probability of transitioning to B at time $t = 5$ exceeds a given threshold?
Objectives:

- Derive the time-dependent probability distribution of the system governed by the above CTMC.
- Formulate the synthesis goal as a constraint: $\Pr(\text{Reach state } C \mid t = 5) \geq 0.8$.
- Solve this constraint to identify a valid region of parameter values (r, s) satisfying the property.
- Implement a numerical or symbolic method to find such values.
- Discuss the uniqueness or multiplicity of solutions, and how the desired bound (e.g., 0.8) influences the feasible region.

4. Emulating CTMCs with Deep Neural Networks

Setting: Simulation of Continuous-Time Markov Chains (CTMCs) is a common method for analyzing stochastic systems in biology, epidemiology, and network protocols. However, simulating long trajectories or computing transition probabilities over time can be computationally expensive. An alternative is to learn a neural network emulator that approximates the CTMC's future state distribution, conditioned on the current state and time. This project considers an SIR (Susceptible–Infected–Recovered) CTMC model and its emulation using deep learning.

Objectives:

- Generate simulation traces from a simple SIR CTMC using standard stochastic simulation algorithms (e.g., Gillespie's SSA).
- Preprocess the data to produce supervised learning examples: current state + time as input, next-state distribution as output.
- Design and train a neural network model to approximate the time-dependent transition kernel of the CTMC.
- Evaluate the model by comparing the learned distribution to the ground truth (obtained via simulations or numerical solution of the master equation).
- Discuss the benefits and limitations of neural emulators for CTMCs, including generalization, interpretability, and performance.

5. Chemical Reaction Networks and Population Limits

Setting: Chemical Reaction Networks (CRNs) are used to model interacting species in biology, epidemiology, and chemistry. These systems can be described stochastically via CTMCs or deterministically using systems of Ordinary Differential Equations (ODEs). The stochastic behavior becomes increasingly deterministic as the population size increases — a phenomenon known as the *law of large numbers* in population processes. This project focuses on analyzing the behavior of a standard SIR (Susceptible-Infected-Recovered) model represented as a CRN at varying population sizes.

Objectives:

- Model the SIR system as a CRN with three species and a set of reactions with associated rates.
- Simulate the system stochastically using Gillespie's algorithm for different population sizes: $N = 10, 100, 1000$.
- Compute the deterministic ODE limit using the mean-field equations, and compare its solution with the average behavior of the stochastic simulations.
- Analyze the variability of outcomes at small population sizes and the convergence toward the ODE as $N \to \infty$.
- Discuss the implications of population size on modeling accuracy and the trade-offs between stochastic and deterministic representations.

6. Scheduling Problem with Random Arrivals

Setting: Scheduling problems are central to computing systems, where jobs (or tasks) must be assigned to limited resources over time. The goal is often to optimize metrics such as average waiting time, latency[1], or system throughput. The complexity of the scheduling problem varies depending on system assumptions, such as the number of servers, whether preemption is allowed, and whether job durations and arrival times are known in advance.

Objectives:

[1] **Latency** refers to the total time a job spends in the system from its arrival to its completion. It includes both the waiting time and the service time. Minimizing latency is crucial in systems where responsiveness is key, such as in real-time processing or user-facing services.

- Implement a simulation of a task processing system where tasks arrive according to a Poisson process and are distributed to three processors.
- Assume each task has a service time drawn from an exponential or another specified distribution.
- Compare at least two scheduling strategies — such as **First-In-First-Out (FIFO)** and **Shortest Job First (SJF)** — in terms of throughput, average waiting time, and load balance.
- Evaluate the strategies under different load conditions (e.g., low vs. high arrival rate).
- (optional) Model the scheduling problem as an MDP.

7. AI Teacher using Reinforcement Learning

Setting: Reinforcement Learning (RL) is increasingly used to adaptively guide learning processes in educational technology. Imagine an AI teacher tasked with helping a student master two mathematical skills: arithmetic and algebra. At each step, the teacher must choose one of three assignment types — arithmetic-only, algebra-only, or mixed — based on the student's performance history. The goal is to learn a teaching policy that accelerates the student's mastery of both skills. This setup can be modeled as a Markov Decision Process (MDP) or extended to a multi-armed bandit framework with evolving reward distributions.
Objectives:

- Formulate the AI teaching task as an RL problem where the teacher is an agent interacting with a student model.
- Implement a simulation environment where the student's skill level evolves probabilistically based on the type of assignments received.
- Use standard RL algorithms (e.g., Q-learning, policy gradients) to learn a policy that minimizes the expected time until the student reaches 90% mastery in both skills.
- Compare the performance of different strategies (e.g., greedy, epsilon-greedy, RL-based) in terms of learning efficiency.
- Discuss implications for adaptive education systems and potential ethical considerations of automated teaching.

8. Consensus for robotic swarms

Setting: Consider the cross-inhibition model specified as a CRN with the following reactions:

$$\begin{aligned}
X + Y &\xrightarrow{q_X} X + U \\
X + Y &\xrightarrow{q_Y} U + Y \\
X + U &\xrightarrow{q_X} X + X \\
Y + U &\xrightarrow{q_Y} Y + Y \\
X + Z_Y &\xrightarrow{q_Y} U + Z_Y \\
U + Z_Y &\xrightarrow{q_Y} Y + Z_Y \\
Y + Z_X &\xrightarrow{q_X} U + Z_X \\
U + Z_X &\xrightarrow{q_X} X + Z_X,
\end{aligned}$$

and the consensus property expressed as

$$F^{<=t}(G^{<=h}([X + Z_X > min_m \wedge [X - Y > d]] \vee [Y + Z_Y > m \wedge [Y - X > d]])),$$

where t, h, m, d are parameters. In words, the formula specifies that a group, before time t, reaches majority in favor of one decision, so that the total number of agents for the voted decision is larger than the majority parameter $min_m = (m/100) \cdot N$, the absolute difference between agent groups is larger than distance d, and the group stays in such state for at least h time units.
Objectives:

- Find the probability that the system finds consensus with parameters $m = 50$, $d = 10$, $t = 35$, $h = 40$, if there are total of $N = 100$ agents, out of which twenty are zealots ($Z_X = Z_Y = 10$), and initially there are 40 agents of type X and 40 agents of type Y.
- Explore how the probability change when varying values of $Z_X = Z_Y$
- Explore how group size N affects consensus probability

References

[1] Durrett, R. (2010). *Probability: Theory and Examples.* Cambridge University Press.

[2] Norris, J. R. (1998). *Markov Chains.* Cambridge University Press.

[3] Rabiner, L. R. (1989). A tutorial on hidden Markov models. *IEEE Proceedings.*

[4] Buchholz, P. (1994). Exact and Ordinary Lumpability. *J. Applied Probability.*

[5] Kemeny, J. G., Snell, J. L. (1976). *Finite Markov Chains.* Springer.

[6] Hillston, J. (1996). *A Compositional Approach to Performance Modelling.* Cambridge University Press.

[7] Baier, C., Katoen, J.-P. (2008). *Principles of Model Checking.* MIT Press.

[8] Hahn, E. M., Hermanns, H., Zhang, L. (2011). Probabilistic reachability for parametric Markov models. *STTT*, 13(1).

[9] Zuliani, P., Platzer, A., Clarke, E. (2013). Bayesian statistical model checking. *FMSD.*

[10] Hermanns, H. (2002). *Interactive Markov Chains.* Springer.

[11] Andova, S., Hermanns, H., Katoen, J.-P. (2003). Discrete-time rewards model-checking. *FMSD.*

[12] Gillespie, D. T. (1977). Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* 81(25).

[13] Jiang, F., et al. (2021). Neural Jump Stochastic Differential Equations. *NeurIPS.*

[14] Wang, Y., Li, L. (2020). A survey of learning for continuous-time stochastic processes. *arXiv preprint* arXiv:2007.10610.

[15] Kurtz, T. G. (1970). ODE limits of jump processes. *J. Applied Probability.*

[16] Anderson, D. F., Kurtz, T. G. (2011). CTMCs for CRNs. In *Design and Analysis of Biomolecular Circuits.* Springer.

[17] Kleinrock, L. (1975). *Queueing Systems, Volume 1.* Wiley.

[18] Mitzenmacher, M., Upfal, E. (2005). *Probability and Computing.* Cambridge University Press.

[19] Harchol-Balter, M. (2013). *Performance Modeling and Design of Computer Systems.* Cambridge University Press.

[20] Rafferty, A. N., et al. (2016). Faster teaching via POMDP planning. *Cognitive Science*, 40(6).

[21] Clement, B., et al. (2015). Multi-armed bandits for ITS. *JEDM*, 7(2).

[22] Sutton, R. S., Barto, A. G. (2018). *Reinforcement Learning: An Introduction.* MIT Press.