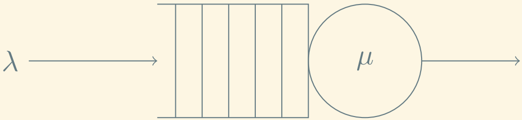
A 3-player game theoretic model of a choice between two queueing systems with strategic managerial decision making

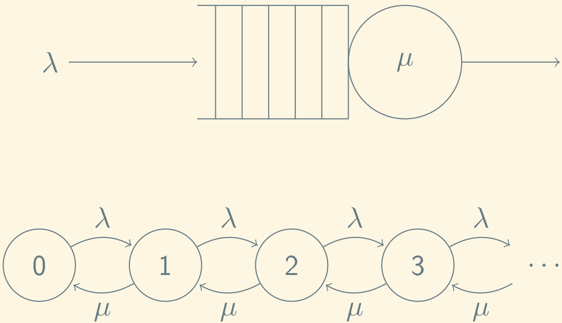
Michalis Panayides

Supervisors:
Dr. Vince Knight,
Prof. Paul Harper

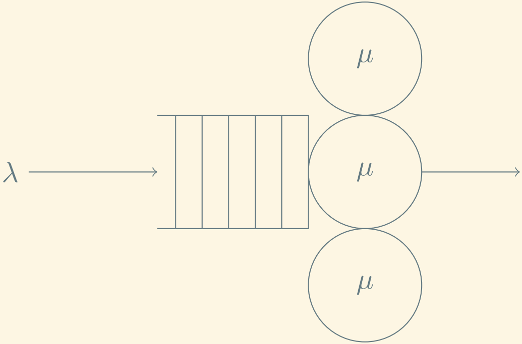
Queues - M/M/1



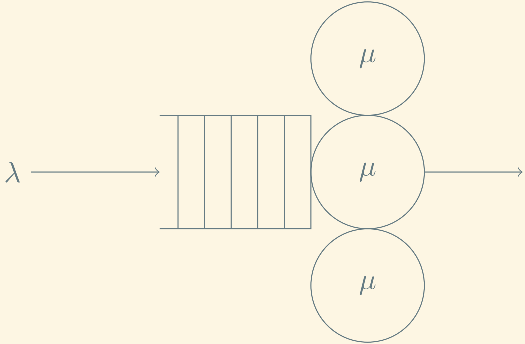
Queues - M/M/1

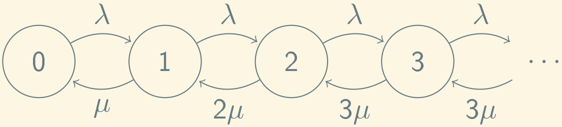


Queues - M/M/3



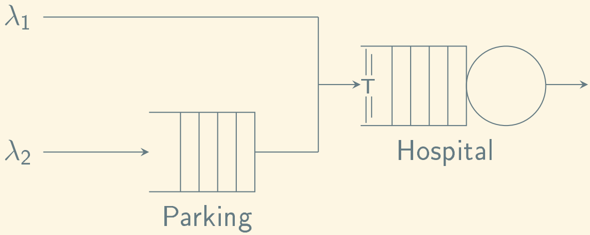
Queues - M/M/3





Queues - Custom network of queues

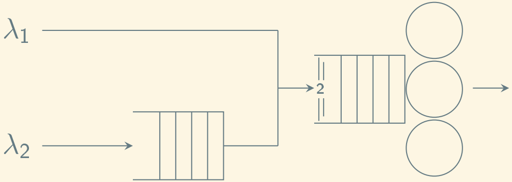
Queues - Custom network of queues

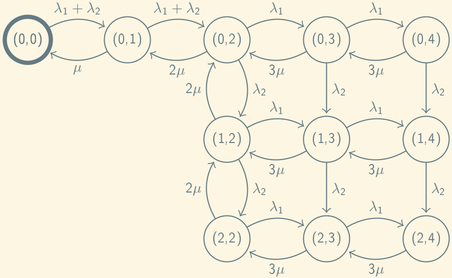


Parameters:

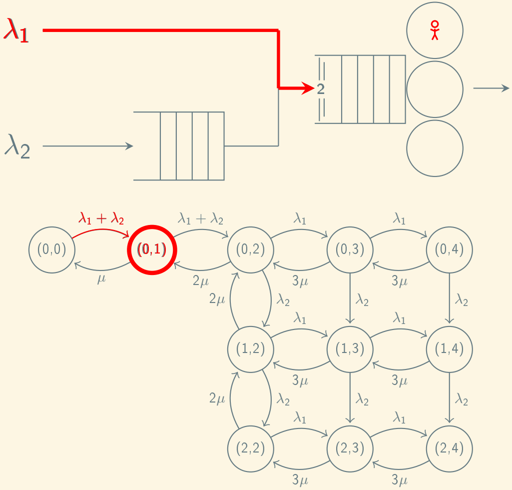
- λ_1: Arrival rate of type 1 individuals
- λ_2: Arrival rate of type 2 individuals
- μ: Service rate
- C: Number of servers
- T: Threshold

Markov Chain - Custom network

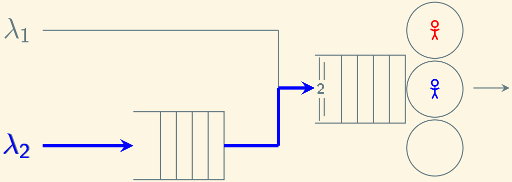


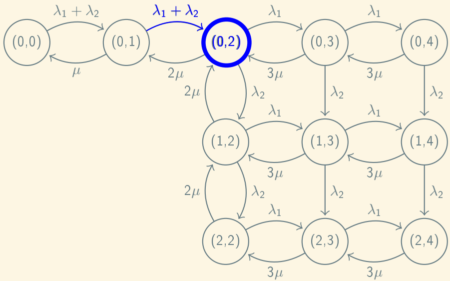


Markov Chain - Custom network

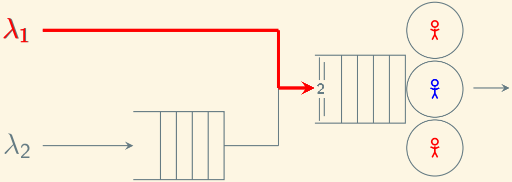


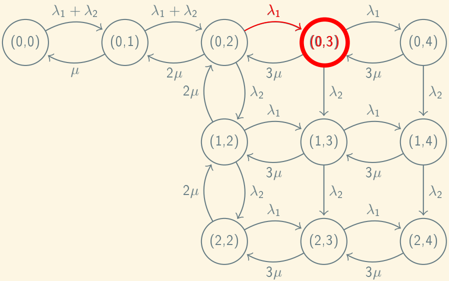
Markov Chain - Custom network



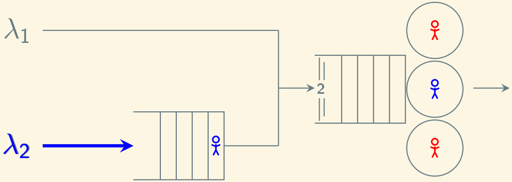


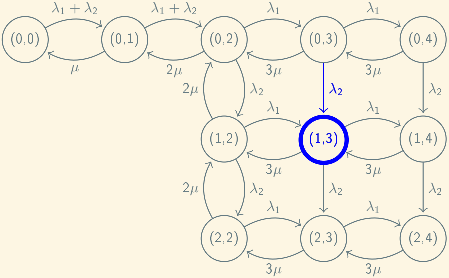
Markov Chain - Custom network



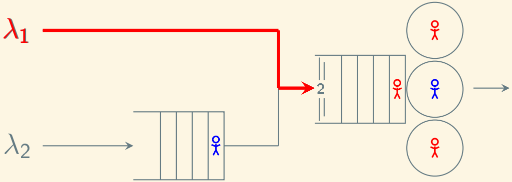


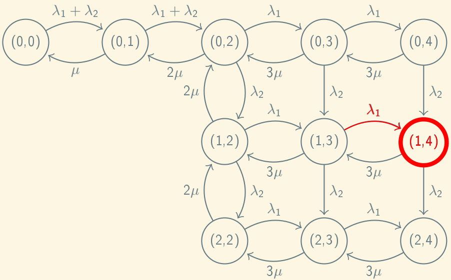
Markov Chain - Custom network



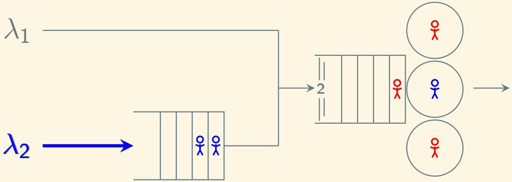


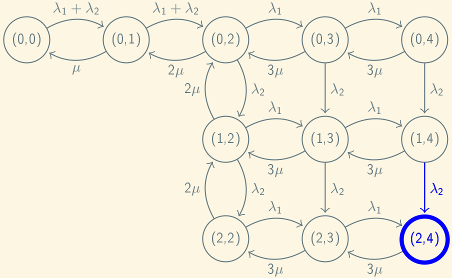
Markov Chain - Custom network



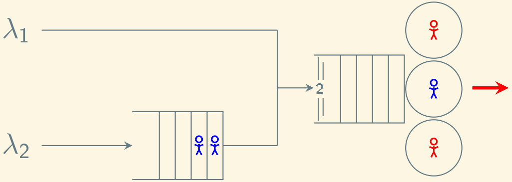


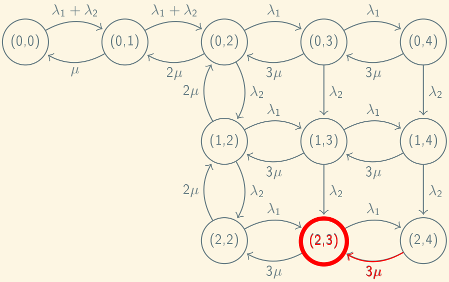
Markov Chain - Custom network



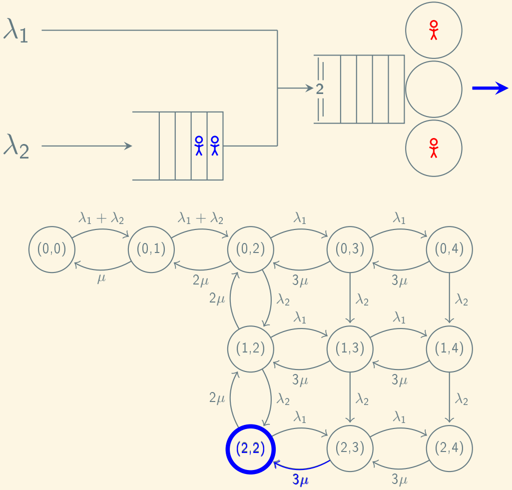


Markov Chain - Custom network

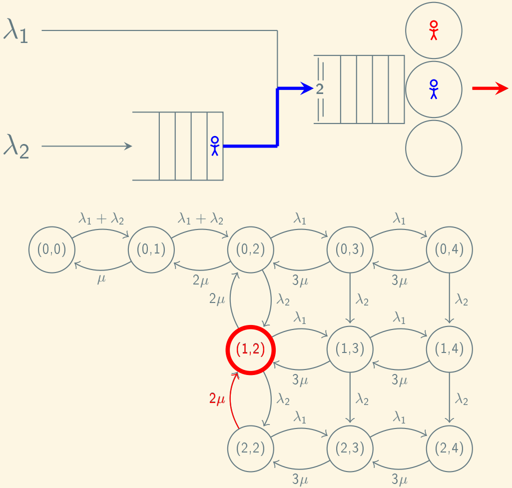




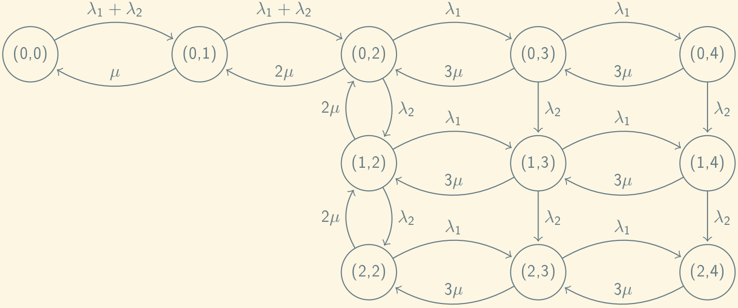
Markov Chain - Custom network



Markov Chain - Custom network



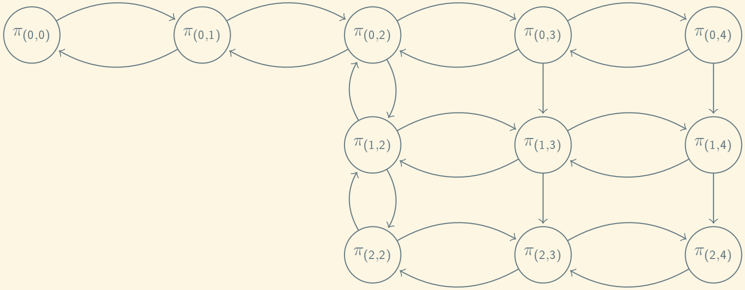
Steady state probabilities - Generator matrix



From \To	**(0,0)**	**(0,1)**	**(0,2)**		**(2,3)**	**(2,4)**
(0,0)	$-\lambda_1 - \lambda_2$	$\lambda_1 + \lambda_2$	0	$\dots$	0	0
(0,1)	μ	$-\mu - \lambda_1 - \lambda_2$	$\lambda_1 + \lambda_2$	$\dots$	0	0
(0,2)	0	2μ	$-2\mu - \lambda_1 - \lambda_2$	$\dots$	0	0
	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
(2,3)	0	0	0	$\dots$	$-\lambda_1 - 3\mu$	λ_1
(2,4)	0	0	0	$\dots$	3μ	-3μ

Steady state probabilities - Generator matrix (Q)

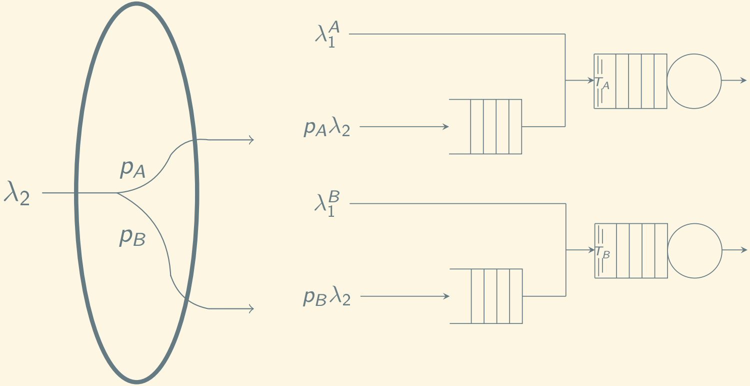
$$\pi = \begin{bmatrix} \pi_{(0,0)} & \pi_{(0,1)} & \pi_{(0,2)} & \dots & \pi_{(2,3)} & \pi_{(2,4)} \end{bmatrix}, \qquad \sum \pi_{(u,v)} = 1$$



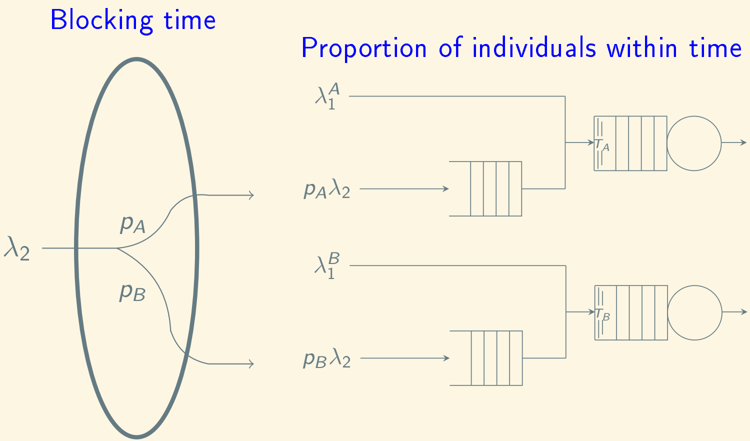
$$\frac{d\pi}{dt} = \pi Q = 0$$

Game - Definition

Game - Players and objectives



Game - Players and objectives



Performance Measures - Blocking time

$$B = \frac{\sum_{(u,v)\in S_A^{(2)}} \pi_{(u,v)} \; b(u,v)}{\sum_{(u,v)\in S_A^{(2)}} \pi_{(u,v)}}$$

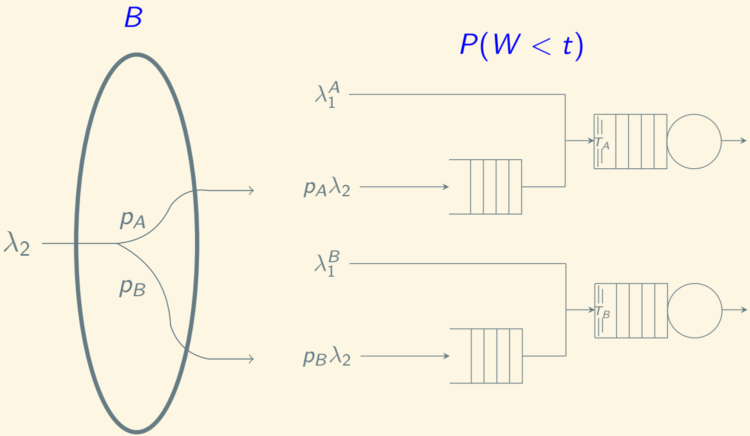
Performance Measures - Proportion within time

$$P(W < t) = \frac{\lambda_1 P_{L'_1}}{\lambda_2 P_{L'_2} + \lambda_1 P_{L'_1}} P(W^{(1)} < t) + \frac{\lambda_2 P_{L'_2}}{\lambda_2 P_{L'_2} + \lambda_1 P_{L'_1}} P(W^{(2)} < t)$$

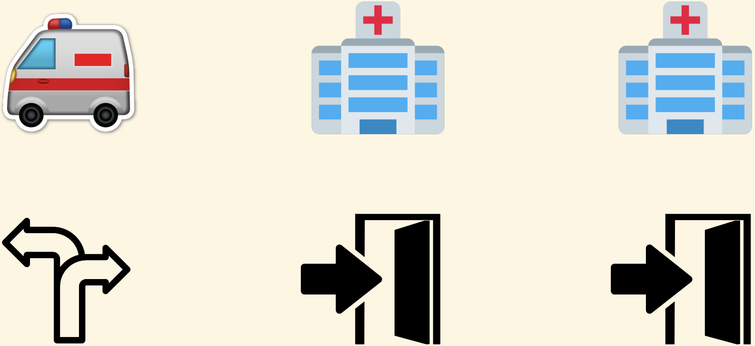
$$P(W^{(1)} < t) = \frac{\sum_{(u,v) \in S_A^{(1)}} P(W_{(u,v)}^{(1)} < t) \pi_{u,v}}{\sum_{(u,v) \in S_A^{(1)}} \pi_{u,v}}$$

$$P(W^{(2)} < t) = \frac{\sum_{(u,v) \in S_A^{(2)}} P(W_{(u,v)}^{(2)} < t) \pi_{u,v}}{\sum_{(u,v) \in S_A^{(2)}} \pi_{u,v}}$$

Game - Players and objectives



Game - Strategies



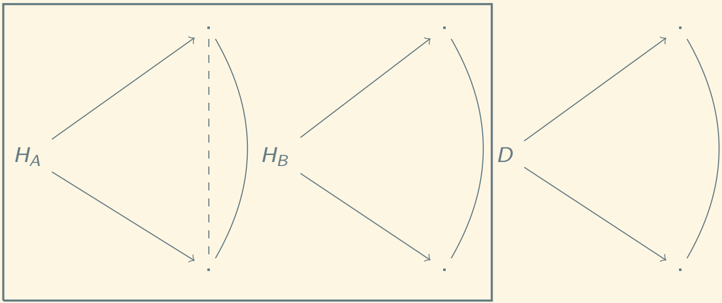
$p_A, p_B \in [0, 1]$

$p_A + p_B = 1$

$T_A \in [1, N_A]$

$T_B \in [1, N_B]$

Game - Formulation



Game - Payoff matrices

$$A = \begin{pmatrix} U^A_{1,1} & U^A_{1,2} & \cdots & U^A_{1,N_B} \\ U^A_{2,1} & U^A_{2,2} & \cdots & U^A_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ U^A_{N_A,1} & U^A_{N_A,2} & \cdots & U^A_{N_A,N_B} \end{pmatrix}, \quad B = \begin{pmatrix} U^B_{1,1} & U^B_{1,2} & \cdots & U^B_{1,N_B} \\ U^B_{2,1} & U^B_{2,2} & \cdots & U^B_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ U^B_{N_A,1} & U^B_{N_A,2} & \cdots & U^B_{N_A,N_B} \end{pmatrix}$$

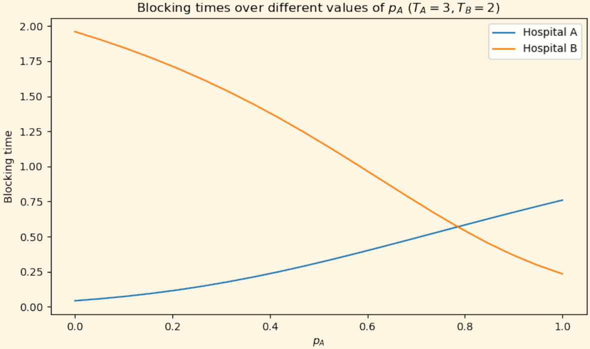
$$R = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,N_B} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_A,1} & p_{N_A,2} & \cdots & p_{N_A,N_B} \end{pmatrix}$$

Ambulance's Decision

$$R = \begin{pmatrix} p_{1,1} & p_{1,2} & \cdots & p_{1,N_B} \\ p_{2,1} & p_{2,2} & \cdots & p_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ p_{N_A,1} & p_{N_A,2} & \cdots & p_{N_A,N_B} \end{pmatrix}$$

Ambulance's Decision

$$\begin{matrix} T_A = 3 \\ T_B = 2 \end{matrix} \rightarrow \begin{pmatrix} - & - & - & - \\ - & - & - & - \\ - & \text{x} & - & - \\ - & - & - & - \end{pmatrix} \rightarrow B_A(p_A) = B_B(1 - p_A)$$



Hospitals' Decision

$$A = \begin{pmatrix} U^A_{1,1} & U^A_{1,2} & \cdots & U^A_{1,N_B} \\ U^A_{2,1} & U^A_{2,2} & \cdots & U^A_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ U^A_{N_A,1} & U^A_{N_A,2} & \cdots & U^A_{N_A,N_B} \end{pmatrix}, \quad B = \begin{pmatrix} U^B_{1,1} & U^B_{1,2} & \cdots & U^B_{1,N_B} \\ U^B_{2,1} & U^B_{2,2} & \cdots & U^B_{2,N_B} \\ \vdots & \vdots & \ddots & \vdots \\ U^B_{N_A,1} & U^B_{N_A,2} & \cdots & U^B_{N_A,N_B} \end{pmatrix}$$

$$U^{(i)}_{T_A,T_B} = 1 - \left[\left(P(X^{(i)} < t) - 0.95 \right)^2 \right]$$

Nash Equilibrium

$$R = \begin{pmatrix} 0.5 & 0.1 & 0 & 0 \\ 0.9 & 0.5 & 0.2 & 0 \\ 1 & 0.8 & 0.5 & 0.3 \\ 1 & 1 & 0.7 & 0.5 \end{pmatrix}$$

$$A = \begin{pmatrix} 0.99998394 & 0.99998394 & 0.99998394 & 0.99998394 \\ 0.99998955 & 0.99998848 & 0.99998649 & 0.9999845 \\ 0.99999952 & 0.9999987 & 0.99999596 & 0.99999199 \\ 0.99994372 & 0.99995113 & 0.99998603 & 0.99999911 \end{pmatrix}$$

$$B = \begin{pmatrix} 0.99998394 & 0.99998955 & 0.99999952 & 0.99994372 \\ 0.99998394 & 0.99998848 & 0.9999987 & 0.99995113 \\ 0.99998394 & 0.99998649 & 0.99999596 & 0.99998603 \\ 0.99998394 & 0.9999845 & 0.99999199 & 0.99999911 \end{pmatrix}$$

Nash Equilibria: $\dfrac{A}{(0, 0, 0.4, 0.6)} \quad \dfrac{B}{(0, 0, 0.4, 0.6)}$

Asymmetric Replicator Dynamics

$$\frac{dx}{dt}_i = x_i((f_x)_i - \phi_x), \qquad \text{for all } i$$

$$\frac{dy}{dt}_i = y_i((f_y)_i - \phi_y), \qquad \text{for all } i$$

Inefficiency measure

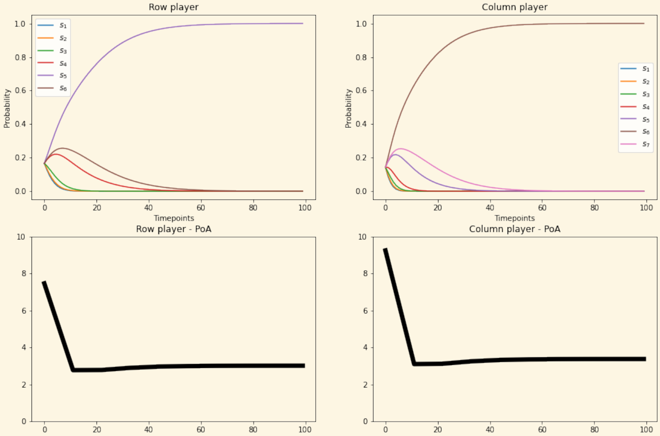
$$PoA = \frac{\max_{s \in E} Cost(s)}{\min_{s \in S} Cost(S)}$$

Inefficiency measure

$$PoA = \frac{\max_{s \in E} Cost(s)}{\min_{s \in S} Cost(S)}$$

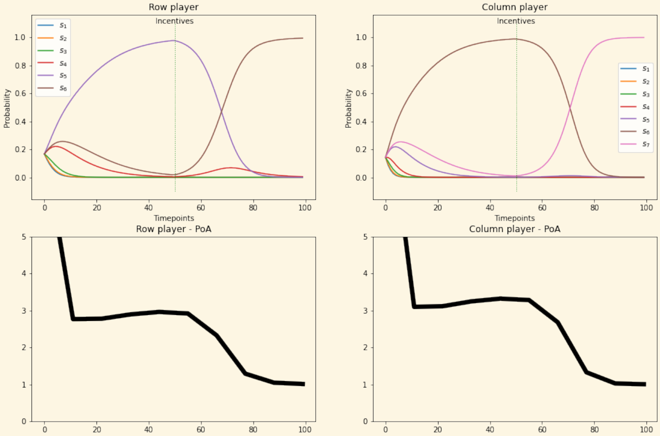
$$PoA_A(s_r) = \frac{Cost(s_r)}{\min_{s \in S} Cost(S)}, \qquad PoA_B(s_c) = \frac{Cost(s_c)}{\min_{s \in S} Cost(S)}$$

Learning algorithms - Asymmetric replicator dynamics



Inefficiencies can be learned and emerge naturally

Learning algorithms - Asymmetric replicator dynamics



Targeted incentivisation of behaviours can help escape learned inefficiencies

Thank you!

"Inefficiencies can be learned and emerge naturally"

"Targeted incentivisation of behaviours can help escape learned inefficiencies"

$ pip install ambulance_game

`https://github.com/11michalis11/AmbulanceDecisionGame`

PanayidesM@cardiff.ac.uk

@Michalis_Pan

@11michalis11