FVRL: Fleming-Viot Reinforcement Learning for the efficient exploration of environments with sparse and rare rewards

Daniel Mastropietro, Szymon Majewski, Urtzi Ayesta, Matthieu Jonckheere
IRIT, Toulouse INP
École Polytechnique de Paris
CNRS, LAAS

STORE Seminar, Toulouse
April 6, 2022

Relevant keywords and concepts

- Fleming-Viot (is a Stochastic Process)

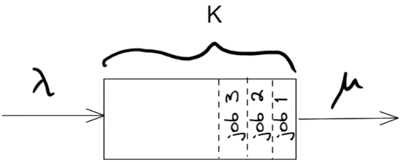
Relevant keywords and concepts

- Fleming-Viot (is a Stochastic Process)
- Reinforcement Learning (tackles Markov Decision Processes)

Relevant keywords and concepts

- Fleming-Viot (is a Stochastic Process)
- Reinforcement Learning (tackles Markov Decision Processes)
  - Exploration (impacts Learning)
  - Rewards (sparse and rare) (impacts Learning)

Motivation - The M/M/1/K queue system

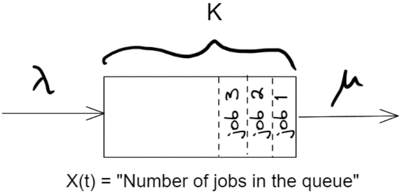


X(t) = "Number of jobs in the queue"

$$\lambda < \mu$$

- Simplest queue model - useful as **test bench**

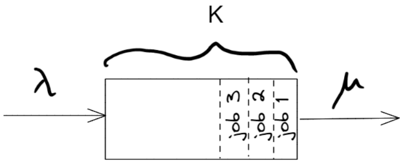
Motivation - The M/M/1/K queue system



$$\lambda < \mu$$

- Simplest queue model - useful as **test bench**
- Blocking is usually costly (i.e. when an incoming job cannot be accepted, e.g. when $X_t = K$)

Motivation - The M/M/1/K queue system

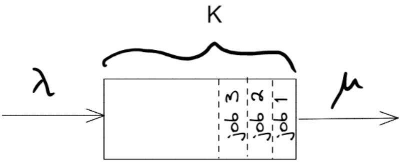


X(t) = "Number of jobs in the queue"

$$\lambda < \mu$$

- Simplest queue model - useful as **test bench**
- Blocking is usually costly (i.e. when an incoming job cannot be accepted, e.g. when $X_t = K$)
  - **sparse**: (K large) 1 out of $K+1$ states

Motivation - The M/M/1/K queue system

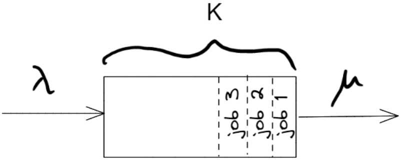


X(t) = "Number of jobs in the queue"

$$\lambda < \mu$$

- Simplest queue model - useful as **test bench**
- Blocking is usually costly (i.e. when an incoming job cannot be accepted, e.g. when $X_t = K$)
  - **sparse**: (K large) 1 out of $K+1$ states
  - **rare**: ($\lambda \ll \mu$ or K large)
    e.g. $K = 40, \lambda = 0.7, \mu = 1 \implies Pr(X_t = K) \sim 10^{-7}$

Motivation - The M/M/1/K queue system



X(t) = "Number of jobs in the queue"

$$\lambda < \mu$$

- Simplest queue model - useful as **test bench**
- Blocking is usually costly (i.e. when an incoming job cannot be accepted, e.g. when $X_t = K$)
  - **sparse**: (K large) 1 out of $K+1$ states
  - **rare**: ($\lambda \ll \mu$ or K large)
    e.g. $K = 40,\ \lambda = 0.7, \mu = 1 \implies Pr(X_t = K) \sim 10^{-7}$

Although rare, blocking can be very costly when it happens...

Outline

Environments with **sparse and rare rewards**

Fleming-Viot particle systems for **probability estimation**

FVRL: Fleming-Viot particle systems for **learning optimum policy**

Many environments give very few rewards - sparsity

In some, rewards are also rare

- Games (e.g. chess, go, ...)
  $\rightarrow$ normally **sparse** but *not* rare

Many environments give very few rewards - sparsity

In some, rewards are also rare

- Games (e.g. chess, go, ...)
  $\rightarrow$ normally **sparse** but *not* rare
- Industry (energy blackout, financial black swan, falling robot, ...)

Many environments give very few rewards - sparsity

In some, rewards are also rare

- Games (e.g. chess, go, ...)
  → normally **sparse** but *not* rare
- Industry (energy blackout, financial black swan, falling robot, ...)
  → **sparse** and **rare**

Many environments give very few rewards - sparsity

In some, rewards are also rare

- Games (e.g. chess, go, ...)
  $\rightarrow$ normally **sparse** but *not* rare
- Industry (energy blackout, financial black swan, falling robot, ...)
  $\rightarrow$ **sparse** and **rare**

- Commonly used to estimate prob. rare events:
  Importance Sampling
  $\rightarrow$ *Here we explore a completely different approach, using Fleming-Viot particle systems*

Normally we know a lot about environment structure

E.g. location of zero-reward states

- We can exploit structure knowledge to guide exploration

Normally we know a lot about environment structure

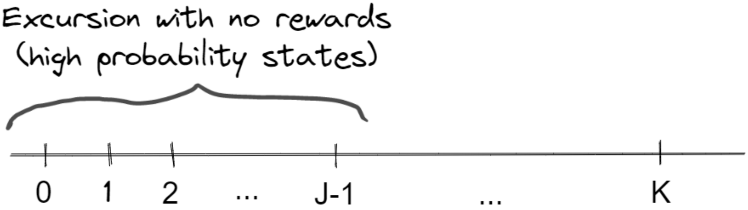
E.g. location of zero-reward states

- We can exploit structure knowledge to guide exploration
- Domain knowledge or prior exploration

Normally we know a lot about environment structure

E.g. location of zero-reward states

- We can exploit structure knowledge to guide exploration
- Domain knowledge or prior exploration
  M/M/1/K queue: set of queue sizes where no blocking has happened



Fleming-Viot particle systems for probability estimation

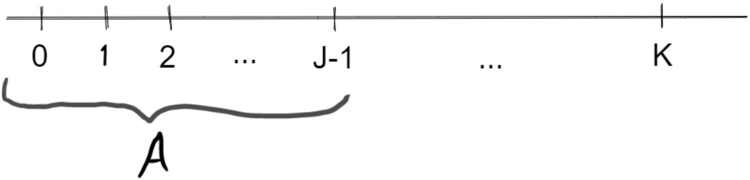
- FV based on Markov chain with **absorbing states**

Fleming-Viot particle systems for probability estimation

- FV based on Markov chain with **absorbing states**
- Proposed by Burdzy et al. in 1996 as genetic particle system to mimic evolution

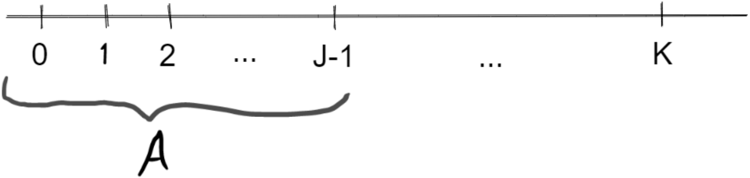
Fleming-Viot particle systems for probability estimation

- FV based on Markov chain with **absorbing states**
- Proposed by Burdzy et al. in 1996 as genetic particle system to mimic evolution
- Let $\mathcal{A}$: set of *known* zero-reward states



Fleming-Viot particle systems for probability estimation

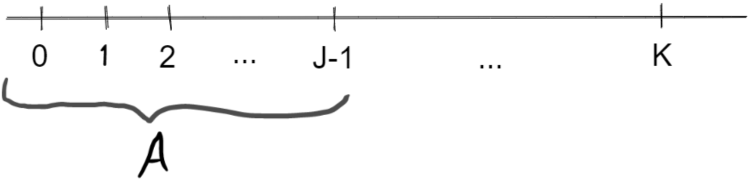
- FV based on Markov chain with **absorbing states**
- Proposed by Burdzy et al. in 1996 as genetic particle system to mimic evolution
- Let $\mathcal{A}$: set of *known* zero-reward states



- Absorbing state: $J-1$ (in general, the boundary of $\mathcal{A}$, "$\partial\mathcal{A}$")

Fleming-Viot particle systems for probability estimation

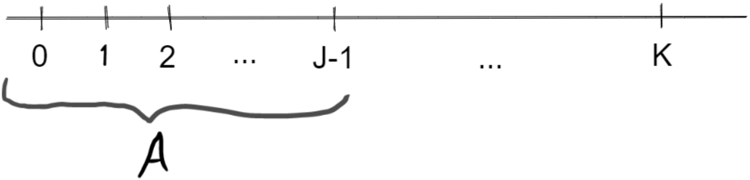
- FV based on Markov chain with **absorbing states**
- Proposed by Burdzy et al. in 1996 as genetic particle system to mimic evolution
- Let $\mathcal{A}$: set of *known* zero-reward states



- Absorbing state: $J-1$ (in general, the boundary of $\mathcal{A}$, "$\partial\mathcal{A}$")
- N particles evolve independently - same dynamics

Fleming-Viot particle systems for probability estimation

- FV based on Markov chain with **absorbing states**
- Proposed by Burdzy et al. in 1996 as genetic particle system to mimic evolution
- Let $\mathcal{A}$: set of *known* zero-reward states



- Absorbing state: $J-1$ (in general, the boundary of $\mathcal{A}$, "$\partial\mathcal{A}$")
- N particles evolve independently - same dynamics
- When **absorbed** $\rightarrow$ **reactivation** to one of other $N-1$ particles

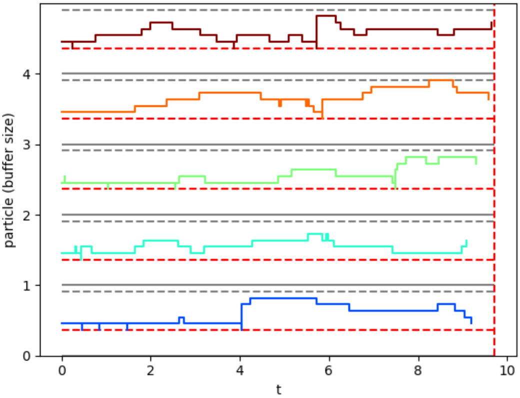
Fleming-Viot particle systems for probability estimation

The method pushes the system to be *closer* to rare states with non-zero rewards

Fleming-Viot particle systems for probability estimation

The method pushes the system to be *closer* to rare states with non-zero rewards

- FV dynamics example on $N = 5$ particles, $K = 10$, $J = 5$



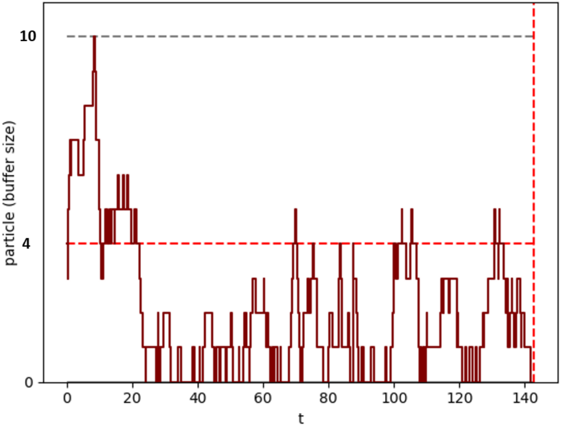
Fleming-Viot particle systems for probability estimation

...compared to standard Monte-Carlo...

Fleming-Viot particle systems for probability estimation

...compared to standard Monte-Carlo...

- MC dynamics (same number of events as FV case)



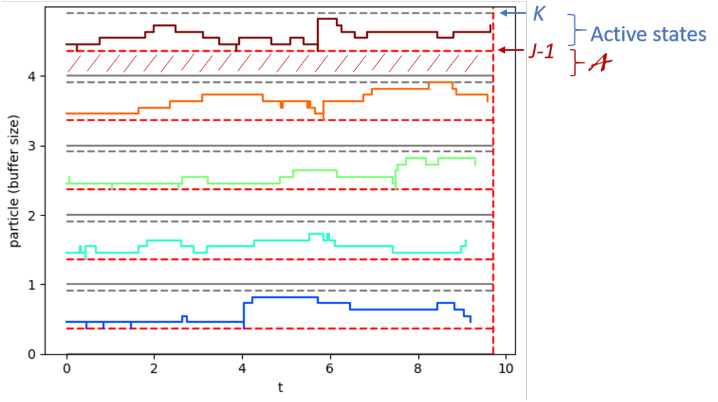
Fleming-Viot particle systems for probability estimation

How to choose J in FV?

Fleming-Viot particle systems for probability estimation

How to choose J in FV?

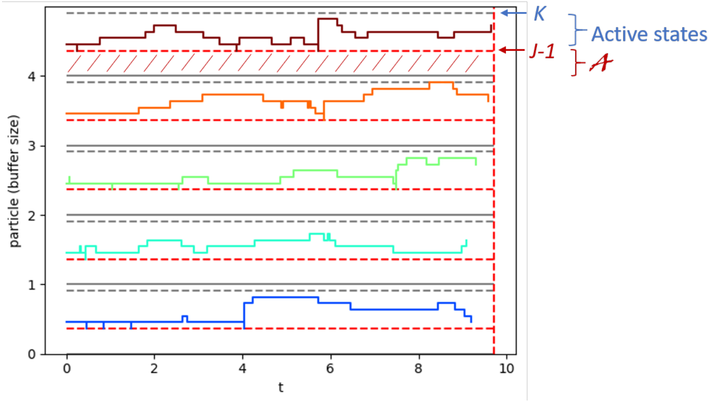
- FV dynamics example on $N = 5$ particles, $K = 10$, $J = 5$



Fleming-Viot particle systems for probability estimation

How to choose J in FV?

- FV dynamics example on $N = 5$ particles, $K = 10$, $J = 5$



First we need to know how to estimate probabilities using FV

Stationary probability estimation using FV

Assume **irreducible, aperiodic** Markov chain X_t, **renewal theory** gives us a characterization of the **stationary probability** of state x:

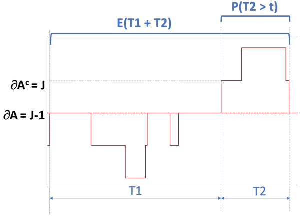
$$p(x) = \frac{\mathbb{E}^i\left(\int_0^{T_i} \mathbb{1}\{X_t = x\}dt\right)}{\mathbb{E}^i T_i} = \frac{\int_0^{\infty} \mathbb{E}^i \mathbb{1}\{X_t = x, t \le T_i\}dt}{\mathbb{E}^i T_i}$$

T_i is the random cycle time: state $i \to i$ (*any* chosen state i).

Stationary probability estimation using FV

Given knowledge of set $\mathcal{A}$ of zero-reward states, it's convenient to write, for $x \notin \mathcal{A}$:

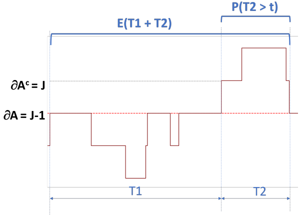
$$p(x) = \frac{\int_0^\infty \phi_t^{\partial \mathcal{A}^c}(x) P^{\partial \mathcal{A}^c}(T_2 > t) dt}{\mathbb{E}^{\partial \mathcal{A}}\left[T_1 + T_2\right]}$$



Stationary probability estimation using FV

Given knowledge of set $\mathcal{A}$ of zero-reward states, it's convenient to write, for $x \notin \mathcal{A}$:

$$p(x) = \frac{\int_0^\infty \phi_t^{\partial\mathcal{A}^c}(x) P^{\partial\mathcal{A}^c}(T_2 > t) dt}{\mathbb{E}^{\partial\mathcal{A}}\left[T_1 + T_2\right]}$$

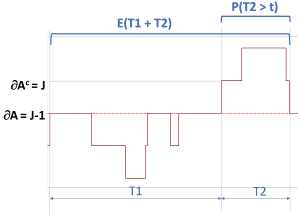


$\partial\mathcal{A}^c$ is the boundary of the complement set of $\mathcal{A}$,
$\phi_t^{\partial\mathcal{A}^c}(x) = P^{\partial\mathcal{A}^c}(X_t = x | T > t)$
T_1 is the time to hitting $\mathcal{A}^c$ starting at $\partial\mathcal{A}$,
T_2 is the time to absorption starting at $\partial\mathcal{A}^c$.

Stationary probability estimation using FV

Given knowledge of set $\mathcal{A}$ of zero-reward states, it's convenient to write, for $x \notin \mathcal{A}$:

$$p(x) = \frac{\int_0^\infty \phi_t^{\partial\mathcal{A}^c}(x) P^{\partial\mathcal{A}^c}(T_2 > t) dt}{\mathbb{E}^{\partial\mathcal{A}}\left[T_1 + T_2\right]}$$



$\partial\mathcal{A}^c$ is the boundary of the complement set of $\mathcal{A}$,
$\phi_t^{\partial\mathcal{A}^c}(x) = P^{\partial\mathcal{A}^c}(X_t = x | T > t)$
T_1 is the time to hitting $\mathcal{A}^c$ starting at $\partial\mathcal{A}$,
T_2 is the time to absorption starting at $\partial\mathcal{A}^c$.

Key fact ([Asselah et al.(2011)Asselah, Ferrari, and Groisman]):
"Empirical probability of x" $\rightarrow \phi_t^{\partial\mathcal{A}^c}(x)$ in L_1-norm when $N \rightarrow \infty$ as $\frac{1}{\sqrt{N}}$

Estimation of $p(x), x \notin \mathcal{A}$

Illustration on M/M/1/K queue system:

$$p(x) = \frac{\int_0^\infty \phi_t^J(x) P^J(T_2 > t) dt}{\mathbb{E}^{J-1}\left[T_1 + T_2\right]}$$

Estimation of $p(x), x \notin \mathcal{A}$

Illustration on M/M/1/K queue system:

$$p(x) = \frac{\int_0^\infty \phi_t^J(x) P^J(T_2 > t) dt}{\mathbb{E}^{J-1}\left[T_1 + T_2\right]}$$

- Each of three quantities estimated separately

Estimation of $p(x), x \notin \mathcal{A}$

Illustration on M/M/1/K queue system:

$$p(x) = \frac{\int_0^\infty \phi_t^J(x) P^J(T_2 > t) dt}{\mathbb{E}^{J-1}\left[T_1 + T_2\right]}$$

- Each of three quantities estimated separately
- Two simulations, *in order*:
  1. A single queue, run for sufficiently large time T to observe enough **re-absorption cycles** $\rightarrow \hat{\mathbb{E}}^{J-1}\left[T_1 + T_2\right], \hat{P}^J(T_2 > t)$ using moment estimators

Estimation of $p(x), x \notin \mathcal{A}$

Illustration on M/M/1/K queue system:

$$p(x) = \frac{\int_0^\infty \phi_t^J(x) P^J(T_2 > t) dt}{\mathbb{E}^{J-1}\left[T_1 + T_2\right]}$$

- Each of three quantities estimated separately
- Two simulations, *in order*:
  1. A single queue, run for sufficiently large time T to observe enough **re-absorption cycles**
     $\rightarrow \hat{\mathbb{E}}^{J-1}\left[T_1 + T_2\right], \hat{P}^J(T_2 > t)$ using moment estimators
  2. N queues $\sim$ Fleming-Viot process for as long as $\mathbf{maxT_2}$
     $\rightarrow \hat{\phi}_t^J(x)$ using empirical probability of state x at each t

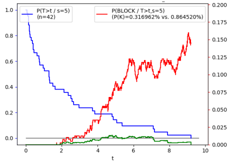
Estimation of $p(x), x \notin \mathcal{A}$

Illustration on M/M/1/K queue system:

$$p(x) = \frac{\int_0^\infty \phi_t^J(x) P^J(T_2 > t) dt}{\mathbb{E}^{J-1}\left[T_1 + T_2\right]}$$

- Each of three quantities estimated separately
- Two simulations, *in order*:
  1. A single queue, run for sufficiently large time T to observe enough **re-absorption cycles**
     $\rightarrow \hat{\mathbb{E}}^{J-1}\left[T_1 + T_2\right], \hat{P}^J(T_2 > t)$ using moment estimators
  2. N queues $\sim$ Fleming-Viot process for as long as $\mathbf{max T_2}$
     $\rightarrow \hat{\phi}_t^J(x)$ using empirical probability of state x at each t

Flavour of $\hat{P}^J(T_2 > t)$, $\hat{\phi}_t^J(x)$, $\hat{\phi}_t^J(x)\hat{P}^J(T_2 > t)$ as function of t



There is an estimation trade-off in J (size of $\mathcal{A}$)

- **Larger J:** Need larger simulation time T for proper estimation of denominator, $\mathbb{E}^{J-1}\left[T_1 + T_2\right], P^J(T_2 > t)$

There is an estimation trade-off in J (size of $\mathcal{A}$)

- **Larger J:** Need larger simulation time T for proper estimation of denominator, $\mathbb{E}^{J-1}\left[T_1 + T_2\right], P^J(T_2 > t)$
- **Smaller J:** Need larger number of particles N for proper estimation of numerator, $\phi_t^J(x)$

Results on the M/M/1/K queue system

Simulation setup

- Goal: Estimate blocking probability $p(K)$

Results on the M/M/1/K queue system

Simulation setup

- Goal: Estimate blocking probability $p(K)$
- Simulation setup
  - $\lambda = 0.7, \mu = 1$, different K's

Results on the M/M/1/K queue system

Simulation setup

- Goal: Estimate blocking probability $p(K)$
- Simulation setup
  - $\lambda = 0.7, \mu = 1$, different K's
  - $J = K/2$

Results on the M/M/1/K queue system

Simulation setup

- Goal: Estimate blocking probability $p(K)$
- Simulation setup
  - $\lambda = 0.7, \mu = 1$, different K's
  - $J = K/2$
  - Fixed simulation time T
  - Increasing number of particles N (analyze convergence)

Results on the M/M/1/K queue system

Simulation setup

- Goal: Estimate blocking probability $p(K)$
- Simulation setup
  - $\lambda = 0.7, \mu = 1$, different K's
  - $J = K/2$
  - Fixed simulation time T
  - Increasing number of particles N (analyze convergence)

Results on the M/M/1/K queue system

Simulation setup

- Goal: Estimate blocking probability $p(K)$
- Simulation setup
  - $\lambda = 0.7, \mu = 1$, different K's
  - $J = K/2$
  - Fixed simulation time T
  - Increasing number of particles N (analyze convergence)
- Fair comparison with vanilla Monte-Carlo (MC)

Results on the M/M/1/K queue system $(\lambda = 0.7, \mu = 1)$

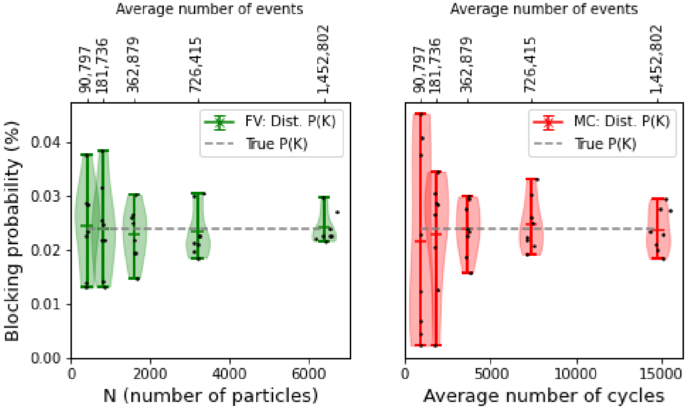
Medium-size capacity K - FV and MC give unbiased estimates

Results on the M/M/1/K queue system $(\lambda = 0.7, \mu = 1)$

Medium-size capacity K - FV and MC give unbiased estimates

Convergence with number of particles N - FV vs. MC

$\mathbf{K = 20, Pr(K) \sim 10^{-4}}$, $J = K/2$, on 8 replications per violin plot

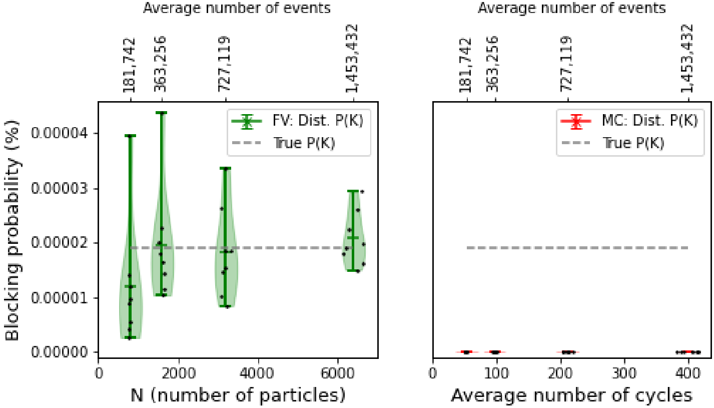


Results on the M/M/1/K queue system $(\lambda = 0.7, \mu = 1)$

Large capacity K - FV gives unbiased estimates while MC fails

Convergence with number of particles N - FV vs. MC

$\mathbf{K = 40, Pr(K) \sim 10^{-7}}$, $J = K/2$, on 8 replications per violin plot



FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners → what is the **optimum** K?

FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners $\rightarrow$ what is the **optimum** K?
  Examples:
  - Minimize holding cost $\rightarrow K^* = 0$

FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners $\rightarrow$ what is the **optimum** K?
  Examples:
  - Minimize holding cost $\rightarrow K^* = 0$
  - Minimize number of rejected jobs $\rightarrow K^* = \infty$

FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners $\rightarrow$ what is the **optimum** K?
  Examples:
  - Minimize holding cost $\rightarrow K^* = 0$
  - Minimize number of rejected jobs $\rightarrow K^* = \infty$

  We will illustrate the methodology on an objective with non-trivial optimum

FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners → what is the **optimum** K?
  Examples:
  - Minimize holding cost → $K^* = 0$
  - Minimize number of rejected jobs → $K^* = \infty$

  We will illustrate the methodology on an objective with non-trivial optimum
- The system can be cast as a Markov Decision Process (MDP)
  - **States** $x \in S$ and **Actions** $a \in A(x)$
  - **Rewards** for each state and action $(R(x, a))$
  - **Policy** of actions given the state: $\pi(a/x)$ (probability)

  States, actions and rewards are observed in time: X_t, A_t, R_t

FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners $\rightarrow$ what is the **optimum** K?
  Examples:
  - Minimize holding cost $\rightarrow K^* = 0$
  - Minimize number of rejected jobs $\rightarrow K^* = \infty$

  We will illustrate the methodology on an objective with non-trivial optimum
- The system can be cast as a Markov Decision Process (MDP)
  - **States** $x \in S$ and **Actions** $a \in A(x)$
  - **Rewards** for each state and action $(R(x, a))$
  - **Policy** of actions given the state: $\pi(a/x)$ (probability)

  States, actions and rewards are observed in time: X_t, A_t, R_t
- Goal: choose policy that optimises long-run reward
  $lim_{t\to\infty} \frac{1}{t} \sum_0^t \mathbb{E}^\pi[R_t]$

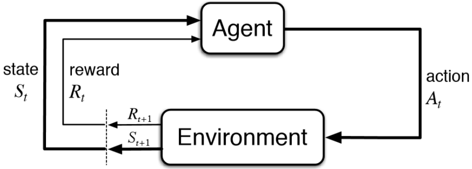
FVRL: Fleming-Viot particle systems for learning optimum policy

- As queue owners $\rightarrow$ what is the **optimum** K?
  Examples:
  - Minimize holding cost $\rightarrow K^* = 0$
  - Minimize number of rejected jobs $\rightarrow K^* = \infty$

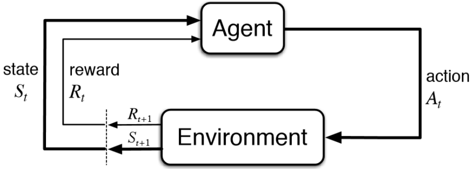
  We will illustrate the methodology on an objective with non-trivial optimum
- The system can be cast as a Markov Decision Process (MDP)
  - **States** $x \in S$ and **Actions** $a \in A(x)$
  - **Rewards** for each state and action $(R(x, a))$
  - **Policy** of actions given the state: $\pi(a/x)$ (probability)

  States, actions and rewards are observed in time: X_t, A_t, R_t
- Goal: choose policy that optimises long-run reward $lim_{t\rightarrow\infty} \frac{1}{t} \sum_0^t \mathbb{E}^\pi[R_t]$
- however... system parameters are normally unknown
  $\implies$ **reinforcement learning** comes into rescue

Overview of reinforcement learning (RL)



Overview of reinforcement learning (RL)



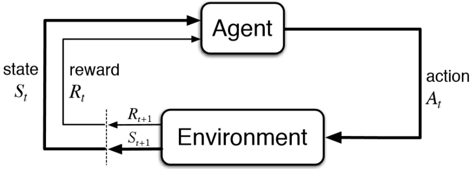
Given the rewards R_t and a **fixed policy** π, an underlying value for each state and action can be defined, as well as a value for each state:

$$Q^\pi(x,a) = \sum_{t=0}^{\infty} \mathbb{E}^\pi\left[(R_t - \overline{v}^\pi) \mid S_0 = x, A_0 = a\right]$$

$$v^\pi(x) = \sum_{t=0}^{\infty} \mathbb{E}^\pi\left[(R_t - \overline{v}^\pi) \mid S_0 = x\right]$$

where $\overline{v}^\pi := \sum_x p^\pi(x) \sum_a \pi(a|x) Q^\pi(x,a)$ (**average long-run reward**)

Overview of reinforcement learning (RL)



Given the rewards R_t and a **fixed policy** π, an underlying value for each state and action can be defined, as well as a value for each state:

$$Q^\pi(x,a) = \sum_{t=0}^{\infty} \mathbb{E}^\pi \left[(R_t - \overline{v}^\pi) \mid S_0 = x, A_0 = a \right]$$

$$v^\pi(x) = \sum_{t=0}^{\infty} \mathbb{E}^\pi \left[(R_t - \overline{v}^\pi) \mid S_0 = x \right]$$

where $\overline{v}^\pi := \sum_x p^\pi(x) \sum_a \pi(a|x) Q^\pi(x,a)$ (**average long-run reward**)

Goal: find policy that optimises the average long-run reward $\overline{v}^\pi$

Queue environment model

- **Actions:** 0 = **Block** incoming job; 1 = **Accept** incoming job

Queue environment model

- **Actions:** 0 = **Block** incoming job; 1 = **Accept** incoming job
- **Rewards:** modeled as blocking cost:

$$R(x,a) = B(1 + b^{x - x_{ref}})\mathbb{1}\{a = 0\}$$

Non-zero only when incoming job is **blocked**

B, b and x_{ref} are positive constants

Increasing function of blocking size x

Queue environment model

- **Actions:** $0 =$ **Block** incoming job; $1 =$ **Accept** incoming job
- **Rewards:** modeled as blocking cost:

$R(x, a) = B(1 + b^{x - x_{ref}})\mathbb{1}\{a = 0\}$	Non-zero only when incoming job is **blocked**
B, b and x_{ref} are positive constants	Increasing function of blocking size x

Sparse rewards... and **rare** if blocking state K is large.

Queue environment model

- **Actions:** $0 =$ **Block** incoming job; $1 =$ **Accept** incoming job
- **Rewards:** modeled as blocking cost:

$$R(x, a) = B(1 + b^{x - x_{ref}})\mathbb{1}\{a = 0\}$$

Non-zero only when incoming job is **blocked**

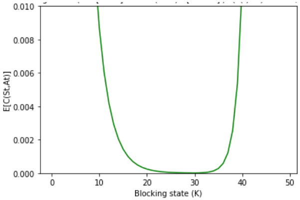
B, b and x_{ref} are positive constants

Increasing function of blocking size x

Sparse rewards... and **rare** if blocking state K is large.

Example of **Expected** cost vs. K

$$\mathbb{E}^{\pi}\left[R(X_t, A_t)\right] = 5(1 + 3^{K-30})p^{\pi}(K)$$



Policy gradient proposed to learn optimum K

Policy gradient theorem makes life eas(*ier*)

Policy gradient proposed to learn optimum K

Policy gradient theorem makes life eas(*ier*)

- Parameterised policy by θ: $\pi_\theta(a|x), \theta \in \mathbb{R}$

Policy gradient proposed to learn optimum K

Policy gradient theorem makes life eas(*ier*)

- Parameterised policy by θ: $\pi_\theta(a|x), \theta \in \mathbb{R}$
- Gradient descent to find minimum average cost, $\overline{v}_\theta^{\pi *}$
- Policy gradient theorem [Sutton et al.(2000)Sutton, McAllester, Singh, and Mansour]

$$\begin{aligned}\nabla_\theta \overline{v}_\theta^\pi &= \mathbb{E}^{\pi_\theta}[Q_\theta(X,a)\nabla_\theta \pi_\theta(a|X)] \\ &= \sum_{x \in \mathcal{S}} p^{\pi_\theta}(x) \sum_a Q_\theta(x,a)\nabla_\theta \pi_\theta(a|x)\end{aligned}$$

Policy gradient proposed to learn optimum K

Proposed parameterised policy is a linear step function

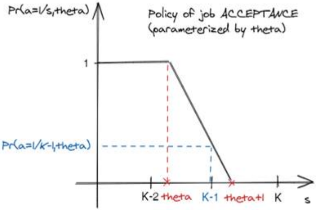
Policy gradient proposed to learn optimum K

Proposed parameterised policy is a linear step function

- Linear-step parameterised policy [Massaro et al.(2019)Massaro, Pellegrini, and Maggi]

Non-deterministic policy

$$\pi_\theta(a = 1|x) = \begin{cases} 1 & \text{if } x \leq \theta, \\ x - \theta + 1 & \text{if } \theta < x < \theta + 1 \\ 0 & \text{if } x \geq \theta + 1 \end{cases}$$



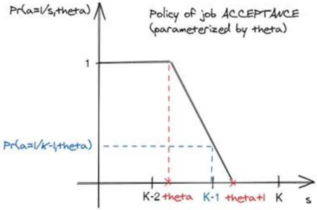
Policy gradient proposed to learn optimum K

Proposed parameterised policy is a linear step function

- Linear-step parameterised policy [Massaro et al.(2019)Massaro, Pellegrini, and Maggi]

Non-deterministic policy

$$\pi_\theta(a = 1|x) = \begin{cases} 1 & \text{if } x \leq \theta, \\ x - \theta + 1 & \text{if } \theta < x < \theta + 1 \\ 0 & \text{if } x \geq \theta + 1 \end{cases}$$



- Gradient becomes

$$\frac{\partial v_\theta^\pi}{\partial \theta} = p_\theta^\pi(K-1)\left[Q_\theta(K-1,1) - Q_\theta(K-1,0)\right]$$

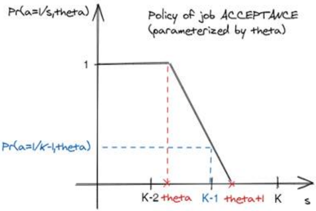
Policy gradient proposed to learn optimum K

Proposed parameterised policy is a linear step function

- Linear-step parameterised policy [Massaro et al.(2019)Massaro, Pellegrini, and Maggi]

Non-deterministic policy

$$\pi_\theta(a=1|x) = \begin{cases} 1 & \text{if } x \leq \theta, \\ x-\theta+1 & \text{if } \theta < x < \theta+1 \\ 0 & \text{if } x \geq \theta+1 \end{cases}$$



- Gradient becomes

$$\frac{\partial v_\theta^\pi}{\partial \theta} = \underbrace{p_\theta^\pi(K-1)}_{\text{Estimated with FV}} \underbrace{[Q_\theta(K-1,1) - Q_\theta(K-1,0)]}_{\text{Estimated with standard Monte-Carlo}}$$

FVRL results on the M/M/1/K queue system

Learning setup

FVRL results on the M/M/1/K queue system

Learning setup

- Initial guess of parameter: $\theta_0 > 0$

FVRL results on the M/M/1/K queue system

Learning setup

- Initial guess of parameter: $\theta_0 > 0$
- At each learning step:
  - FV estimation of $p_\theta^\pi(K)$ with appropriate simulation time T and N particles

FVRL results on the M/M/1/K queue system

Learning setup

- Initial guess of parameter: $\theta_0 > 0$
- At each learning step:
  - FV estimation of $p^{\pi}_{\theta}(K)$ with appropriate simulation time T and N particles
  - Estimation of Q difference until trajectories cross (a.s.)

FVRL results on the M/M/1/K queue system

Learning setup

- Initial guess of parameter: $\theta_0 > 0$
- At each learning step:
  - FV estimation of $p_\theta^\pi(K)$ with appropriate simulation time T and N particles
  - Estimation of Q difference until trajectories cross (a.s.)
  - θ updated by gradient descent

$$\theta \leftarrow \theta - \alpha \frac{\partial \widehat{v_\theta^\pi}}{\partial \theta}$$

FVRL results on the M/M/1/K queue system

Learning setup

- Initial guess of parameter: $\theta_0 > 0$
- At each learning step:
  - FV estimation of $p_\theta^\pi(K)$ with appropriate simulation time T and N particles
  - Estimation of Q difference until trajectories cross (a.s.)
  - θ updated by gradient descent

$$\theta \leftarrow \theta - \alpha \frac{\partial \widehat{v_\theta^\pi}}{\partial \theta}$$

- Fair comparison with vanilla Monte-Carlo (MC)

FVRL results on the M/M/1/K queue system

FVRL is faster than Monte-Carlo-based learning for moderate optimum K value

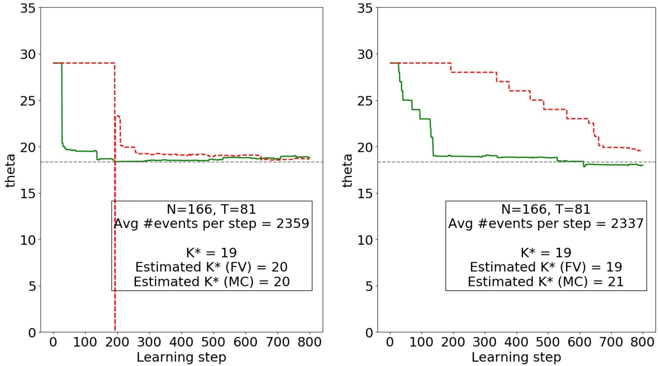
FVRL results on the M/M/1/K queue system

FVRL is faster than Monte-Carlo-based learning for moderate optimum K value

Moderate K^* - Fleming-Viot vs. Monte-Carlo

$$K^* = 19, K_0 = 30, J = 0.3K$$

"free" learning | clipped learning to ± 1



FVRL results on the M/M/1/K queue system

FVRL converges while Monte-Carlo-based learning fails for large optimum K value

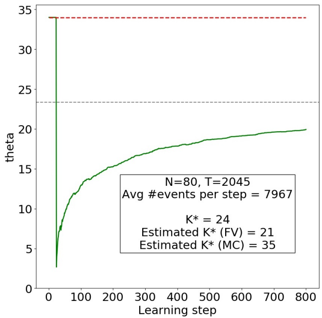
FVRL results on the M/M/1/K queue system

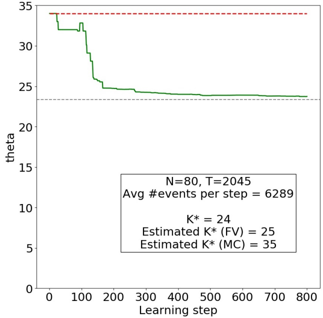
FVRL converges while Monte-Carlo-based learning fails for large optimum K value

Large K^* - Fleming-Viot vs. Monte-Carlo

$$K^* = 24, K_0 = 35, J = 0.5K$$

"free" learning clipped learning to ± 1





Conclusions

- Results show Fleming-Viot approach is able to successfully
  - estimate very small probabilities ($\sim 10^{-7}$)
  - learn optimum parameterised policy with RL

Conclusions

- Results show Fleming-Viot approach is able to successfully
  - estimate very small probabilities ($\sim 10^{-7}$)
  - learn optimum parameterised policy with RL

while Monte-Carlo fails.

Conclusions

- Results show Fleming-Viot approach is able to successfully
  - estimate very small probabilities ($\sim 10^{-7}$)
  - learn optimum parameterised policy with RL

  while Monte-Carlo fails.
- In FVRL estimation accuracy is not as crucial as in probability estimation problem

Conclusions

- Results show Fleming-Viot approach is able to successfully
  - estimate very small probabilities ($\sim 10^{-7}$)
  - learn optimum parameterised policy with RL

  while Monte-Carlo fails.
- In FVRL estimation accuracy is not as crucial as in probability estimation problem
- Fleming-Viot can be applied when:
  - Simulation is possible or historical data is available
  - The environment is stationary

Conclusions

- Results show Fleming-Viot approach is able to successfully
  - estimate very small probabilities ($\sim 10^{-7}$)
  - learn optimum parameterised policy with RL

  while Monte-Carlo fails.
- In FVRL estimation accuracy is not as crucial as in probability estimation problem
- Fleming-Viot can be applied when:
  - Simulation is possible or historical data is available
  - The environment is stationary

Next steps

- Prove that Fleming-Viot is faster than Monte-Carlo

Next steps

- Prove that Fleming-Viot is faster than Monte-Carlo
- Characterise optimum J in terms of estimation accuracy and algorithm efficiency

Next steps

- Prove that Fleming-Viot is faster than Monte-Carlo
- Characterise optimum J in terms of estimation accuracy and algorithm efficiency
- Consider dynamic J as more knowledge is acquired about zero-reward states

Next steps

- Prove that Fleming-Viot is faster than Monte-Carlo
- Characterise optimum J in terms of estimation accuracy and algorithm efficiency
- Consider dynamic J as more knowledge is acquired about zero-reward states
- Reactivate absorbed particles more intelligently, by weighting particle with the value of the state they are in

Next steps

- Prove that Fleming-Viot is faster than Monte-Carlo
- Characterise optimum J in terms of estimation accuracy and algorithm efficiency
- Consider dynamic J as more knowledge is acquired about zero-reward states
- Reactivate absorbed particles more intelligently, by weighting particle with the value of the state they are in
- Apply Fleming-Viot to other classical RL problems (e.g. labyrinth, mountain car, etc.)

Next steps

- Prove that Fleming-Viot is faster than Monte-Carlo
- Characterise optimum J in terms of estimation accuracy and algorithm efficiency
- Consider dynamic J as more knowledge is acquired about zero-reward states
- Reactivate absorbed particles more intelligently, by weighting particle with the value of the state they are in
- Apply Fleming-Viot to other classical RL problems (e.g. labyrinth, mountain car, etc.)

Thank you for your attention!

This presentation is based on the paper:

Mastropietro, D., Majewski S., Ayesta U., Jonckheere M.
"Boosting reinforcement learning with sparse and rare rewards using Fleming-Viot particle systems"
submitted to ICML 2022

partially supported by CIMI grant ANR-11-LABX-0040.

Asselah, A., Ferrari, P., and Groisman, P.
Quasistationary distributions and Fleming-Viot processes in finite spaces.
J. Appl. Probab., 48(2):322–332, 2011.
ISSN 0021-9002.
doi: 10.1239/jap.
URL http://dx.doi.org/10.1239/jap/1308662630.

Massaro, A., Pellegrini, F. D., and Maggi, L.
Optimal trunk-reservation by policy learning.
In *IEEE INFOCOM 2019*, apr 2019.
doi: 10.1109/infocom.2019.8737552.

Sutton, R. S., McAllester, D. A., Singh, S. P., and Mansour, Y.
Policy gradient methods for reinforcement learning with function approximation.

In *Advances in neural information processing systems*, pp. 1057–1063, 2000.