Dynamic Analysis of Multiagent Q-learning with ε-greedy Exploration

Eduardo R. Gomes
Ryszard Kowalczyk

Intelligent Agent Technology - CS3 - FICT
egomes@groupwise.swin.edu.au

SWINBURNE
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

SWINBURNE
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Motivation

- Multiagent Learning (MAL) has become very active research area
- MAL-based systems are finding application in a wide variety of domains
- Tools to understand and model the expected dynamics are necessary

Motivation

> Multiagent Learning (MAL) has become very active research area
> MAL-based systems are finding application in a wide variety of domains
> Tools to understand and model the expected dynamics are necessary

Multiagent Q-learning with ε-greedy exploration

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Motivation

- Multiagent Learning (MAL) has become very active research area
- MAL-based systems are finding application in a wide variety of domains
- Tools to understand and model the expected dynamics are necessary

Multiagent Q-learning with ε-greedy exploration

- Classic algorithm
- It has been applied with success in several domains

Motivation

Q-learning

- Most studied Reinforcement Learning algorithm
- Strong theoretical support and convergence guarantees

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Motivation

Q-learning

- Most studied Reinforcement Learning algorithm
- Strong theoretical support and convergence guarantees
- ... only in the single-agent case

Motivation

Q-learning

- Most studied Reinforcement Learning algorithm
- Strong theoretical support and convergence guarantees
- ... only in the single-agent case

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Motivation

Q-learning

- Most studied Reinforcement Learning algorithm
- Strong theoretical support and convergence guarantees
- ... only in the single-agent case

Multiagent *Q*-learning

- Lack of theoretical support and convergence guarantees
- Very dynamic environment
- Co-adaptation effect
- Rewards and state transitions depend on the joint actions
- Very hard to obtain the dynamics

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

RL and Evolutionary Game Theory

RL and Evolutionary Game Theory

- Researchers have explored links between RL and EGT
- Same principles
  - Growth in one strategy's probability is directly proportional to its performance against the others
- Model of Multiagent *Q*-learning with Boltzmann exploration

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

RL and Evolutionary Game Theory

- Researchers have explored links between RL and EGT
- Same principles
  - Growth in one strategy's probability is directly proportional to its performance against the others
- Model of Multiagent Q-learning with Boltzmann exploration
- Cannot be applied because we have a semi-uniform distribution

SWINBURNE
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

RL and Evolutionary Game Theory

- Researchers have explored links between RL and EGT
- Same principles
  - Growth in one strategy's probability is directly proportional to its performance against the others
- Model of Multiagent *Q*-learning with Boltzmann exploration
- Cannot be applied because we have a semi-uniform distribution

$\varepsilon -$ *greedy* mechanism

- Selects the best action with probability $1 - \varepsilon$
- Selects a random action with probability ε

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Background

Multiagent Q-learning

- Each agent applies the standard Q-learning algorithm
- The agents learn independently
- Rewards and state transitions depend on their joint strategies

Background

Multiagent Q-learning

> Each agent applies the standard Q-learning algorithm
> The agents learn independently
> Rewards and state transitions depend on their joint strategies

> Each agent maintains a table of Q-values
  – $Q(s,i)$ represents how good it is to take action i at state s
> They update the Q-values as they gather experience in the environment
  $Q(s,i) = Q(s,i) + \alpha(r(s,i) + \gamma \max_{i'} Q(s',i') - Q(s,i))$
  – $r(s,i)$ is the reward for taking action i at state s
  – α is the learning rate
  – γ is the discount rate

Action-selection mechanism

Exploration - exploitation problem

- exploit actions known to be good
- explore new actions

ε-greedy

- chose the currently best action with probability $1 - \varepsilon$
- chose a random action with probability ε

Action-selection mechanism

Exploration - exploitation problem

- exploit actions known to be good
- explore new actions

ε-greedy

- chose the currently best action with probability $1-\varepsilon$
- chose a random action with probability ε

$$x(s,i) = \begin{cases} (1-\varepsilon)+(\varepsilon/n), & \text{if } Q(s,i) \text{ is currently the highest} \\ \varepsilon/n, & \text{otherwise} \end{cases}$$

Modelling the algorithm

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Modelling the algorithm

> Build a continuous-time version of the Q-learning update rule

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Modelling the algorithm

- Build a continuous-time version of the Q-learning update rule
- Analyse the limits of this equation for the single-learner case

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Modelling the algorithm

- Build a continuous-time version of the Q-learning update rule
- Analyse the limits of this equation for the single-learner case
- Show how they change dynamically in the multi-learner case

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Modelling the algorithm

- Build a continuous-time version of the Q-learning update rule
- Analyse the limits of this equation for the single-learner case
- Show how they change dynamically in the multi-learner case
- Investigate how the ε-greedy affects the shape of the function

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Modelling the algorithm

- Build a continuous-time version of the Q-learning update rule
- Analyse the limits of this equation for the single-learner case
- Show how they change dynamically in the multi-learner case
- Investigate how the ε-greedy affects the shape of the function
- Develop a system of difference equations to obtain the expected behaviour of the agents

Notation

Notation

Single-state scenarios composed of 2 agents with 2 actions each

Notation

Single-state scenarios composed of 2 agents with 2 actions each

The reward functions can be described as payoff tables

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \qquad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Notation

Single-state scenarios composed of 2 agents with 2 actions each

The reward functions can be described as payoff tables

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \qquad B = \begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix}$$

Q-learning rule can be simplified to

$$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$$

Q_{a_i} is the Q-value of agent a for action i
r_{a_i} is the immediate reward that agent a receives for playing action i

Continuous-time version

$$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$$ *Q*-learning rule

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Continuous-time version

$$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$$ *Q*-learning rule

$$Q_{a_i}(k+1) = Q_{a_i}(k) + \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$$

Continuous-time version

$$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$$ *Q*-learning rule

$$Q_{a_i}(k+1) = Q_{a_i}(k) + \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$$

$$Q_{a_i}(k+1) - Q_{a_i}(k) = \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$$ discrete

SWINBURNE
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Continuous-time version

$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$ *Q*-learning rule

$Q_{a_i}(k+1) = Q_{a_i}(k) + \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$

$Q_{a_i}(k+1) - Q_{a_i}(k) = \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$ discrete

$Q_{a_i}(k+\Delta t) - Q_{a_i}(k) \approx \Delta t \times \alpha(r_{a_i}(k+\Delta t) - Q_{a_i}(k))$

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Continuous-time version

$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$ *Q*-learning rule

$Q_{a_i}(k+1) = Q_{a_i}(k) + \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$

$Q_{a_i}(k+1) - Q_{a_i}(k) = \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$ discrete

$Q_{a_i}(k+\Delta t) - Q_{a_i}(k) \approx \Delta t \times \alpha(r_{a_i}(k+\Delta t) - Q_{a_i}(k))$

$\lim_{\Delta t \to 0} \frac{Q_{a_i}(k+\Delta t) - Q_{a_i}(k)}{\Delta t} \approx \alpha(r_{a_i}(k) - Q_{a_i}(k))$

Continuous-time version

$Q_{a_i} \leftarrow Q_{a_i} + \alpha(r_{a_i} - Q_{a_i})$ Q-learning rule

$Q_{a_i}(k+1) = Q_{a_i}(k) + \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$

$Q_{a_i}(k+1) - Q_{a_i}(k) = \alpha(r_{a_i}(k+1) - Q_{a_i}(k))$ discrete

$Q_{a_i}(k+\Delta t) - Q_{a_i}(k) \approx \Delta t \times \alpha(r_{a_i}(k+\Delta t) - Q_{a_i}(k))$

$\lim_{\Delta t \to 0} \frac{Q_{a_i}(k+\Delta t) - Q_{a_i}(k)}{\Delta t} \approx \alpha(r_{a_i}(k) - Q_{a_i}(k))$

$\frac{dQ_{a_i}(k)}{dt} \approx \alpha(r_{a_i}(k) - Q_{a_i}(k))$ continuous

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Limit of the equation

$$\frac{dQ_{a_i}(k)}{dt} \approx \alpha(r_{a_i}(k) - Q_{a_i}(k))$$ continuous

Limit of the equation

$$\frac{dQ_{a_i}(k)}{dt} \approx \alpha(r_{a_i}(k) - Q_{a_i}(k))$$ continuous

$$Q_{a_i}(k) = Ce^{-\alpha t} + r_{a_i}$$ general solution

SWIN BUR NE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Limit of the equation

$$\frac{dQ_{a_i}(k)}{dt} \approx \alpha(r_{a_i}(k) - Q_{a_i}(k))$$ continuous

$$Q_{a_i}(k) = Ce^{-\alpha t} + r_{a_i}$$ general solution

$$\lim_{t\to\infty} Q_{a_i}(k) = \underbrace{\lim_{t\to\infty} Ce^{-\alpha t}}_{0} + \underbrace{\lim_{t\to\infty} r_{a_i}}_{r_{a_i}} = r_{a_i}$$

Non-learning adversary with pure strategy

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Q_{a_i} will monotonically increase or decrease towards r_{a_i}

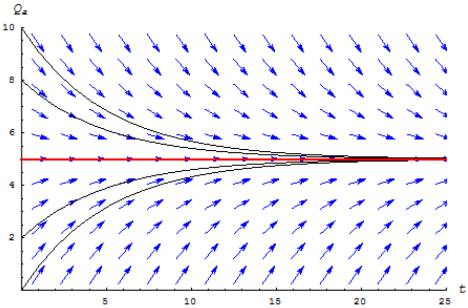
SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Non-learning adversary with pure strategy

Q_{a_i} will monotonically increase or decrease towards r_{a_i}

$\alpha = 0.2$ and $r_{a_i} = 5$; $Q_{a_i}(0) \in \{0, 2, 8, 10\}$



CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Non-learning adversary with mixed strategy

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Non-learning adversary with mixed strategy

r_{a_i} can be replaced by $E[r_{a_i}] = \sum_j a_{ij} y_j$

0.8	0.2
1	5
0	3

$$E[r_{a_1}] = (0.8 * 1) + (0.2 * 5) = 1.8$$
$$E[r_{a_2}] = (0.8 * 0) + (0.2 * 3) = 0.6$$

$$\frac{dQ_{a_i}(t)}{dt} \approx \alpha(E[r_{a_i}(t)] - Q_{a_i}(t))$$

Non-learning adversary with mixed strategy

r_{a_i} can be replaced by $E[r_{a_i}] = \sum_j a_{ij} y_j$

0.8	0.2
1	5
0	3

$$E[r_{a_1}] = (0.8 * 1) + (0.2 * 5) = 1.8$$
$$E[r_{a_2}] = (0.8 * 0) + (0.2 * 3) = 0.6$$

$$\frac{dQ_{a_i}(t)}{dt} \approx \alpha(E[r_{a_i}(t)] - Q_{a_i}(t))$$

$$Q_{a_i}(t) = Ce^{-\alpha t} + E[r_{a_i}]$$

$$\lim_{t\to\infty} Q_{a_i}(k) = \underbrace{\lim_{t\to\infty} Ce^{-\alpha t}}_{0} + \underbrace{\lim_{t\to\infty} E[r_{a_i}]}_{E[r_{a_i}]} = E[r_{a_i}]$$

Non-learning adversary with mixed strategy

r_{a_i} can be replaced by $E[r_{a_i}] = \sum_j a_{ij} y_j$

0.8	0.2
1	5
0	3

$$E[r_{a_1}] = (0.8 * 1) + (0.2 * 5) = 1.8$$
$$E[r_{a_2}] = (0.8 * 0) + (0.2 * 3) = 0.6$$

$$\frac{dQ_{a_i}(t)}{dt} \approx \alpha(E[r_{a_i}(t)] - Q_{a_i}(t))$$

$$Q_{a_i}(t) = Ce^{-\alpha t} + E[r_{a_i}]$$

$$\lim_{t\to\infty} Q_{a_i}(k) = \underbrace{\lim_{t\to\infty} Ce^{-\alpha t}}_{0} + \underbrace{\lim_{t\to\infty} E[r_{a_i}]}_{E[r_{a_i}]} = E[r_{a_i}]$$

then Q_{a_i} will move in expectation towards $E[r_{a_i}]$ in a monotonic fashion

Learning adversary

Learning adversary

Adversary can change its strategy during the learning changing the expected rewards

Learning adversary

Adversary can change its strategy during the learning changing the expected rewards

0.8	0.2
1	5
0	3

$$E[r_{a_1}] = (0.8 * 1) + (0.2 * 5) = 1.8$$

Learning adversary

Adversary can change its strategy during the learning changing the expected rewards

0.8	0.2
1	5
0	3

$$E[r_{a_1}] = (0.8 * 1) + (0.2 * 5) = 1.8$$

0.2	0.8
1	5
0	3

$$E[r_{a_1}] = (0.2 * 1) + (0.8 * 5) = 4.2$$

Learning adversary

Adversary can change its strategy during the learning changing the expected rewards

0.8	0.2
1	5
0	3

$$E[r_{a_1}] = (0.8 * 1) + (0.2 * 5) = 1.8$$

0.2	0.8
1	5
0	3

$$E[r_{a_1}] = (0.2 * 1) + (0.8 * 5) = 4.2$$

Each time the expected reward changes, it changes the limits and direction fields

Learning adversary

Important to identify when the changes in the adversary's strategy will occur

Learning adversary

Important to identify when the changes in the adversary's strategy will occur

ε-greedy updates the strategy whenever a new action becomes the one with highest Q-value

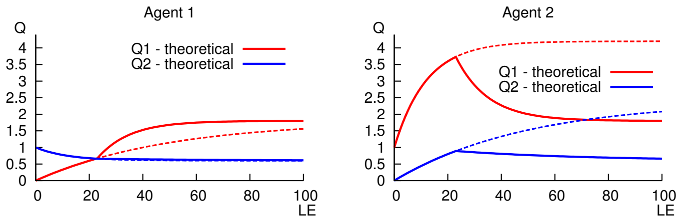
Need to find the intersection points in the adversary's functions

Learning adversary

Important to identify when the changes in the adversary's strategy will occur

ε-greedy updates the strategy whenever a new action becomes the one with highest Q-value

Need to find the intersection points in the adversary's functions



The effects of the ε-greedy

The effects of the ε-greedy

Actions have different probabilities (x_i) of being played

e.g. if $\varepsilon = 0.2 \quad \rightarrow \quad x$ = $[0.9, 0.1]$ or x = $[0.1, 0.9]$

they are updated at different *speeds*

The effects of the ε-greedy

Actions have different probabilities (x_i) of being played

e.g. if $\varepsilon = 0.2 \quad \rightarrow \quad x$ = $[0.9, 0.1]$ or x = $[0.1, 0.9]$

they are updated at different *speeds*

$$\frac{dQ_{a_i}(t)}{dt} \approx x_i(t)\alpha(E[r_{a_i}(t)] - Q_{a_i}(t))$$

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

The effects of the ε-greedy

Actions have different probabilities (x_i) of being played

e.g. if $\varepsilon = 0.2 \quad \rightarrow \quad x$ = $[0.9, 0.1]$ or x = $[0.1, 0.9]$

they are updated at different *speeds*

$$\frac{dQ_{a_i}(t)}{dt} \approx x_i(t)\alpha(E[r_{a_i}(t)] - Q_{a_i}(t))$$

$$Q_{a_i}(t) = Ce^{-x_i \alpha t} + E[r_{a_i}]$$

The effects of the ε-greedy

It does not change the limits of the equation

$$\lim_{t\to\infty} Q_{a_i}(t) = \underbrace{\lim_{t\to\infty} Ce^{-x_i \alpha t}}_{0} + \underbrace{\lim_{t\to\infty} E[r_{a_i}]}_{E[r_{a_i}]} = E[r_{a_i}]$$

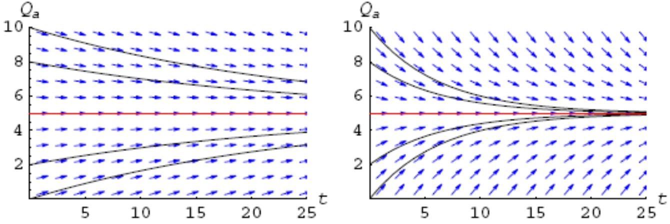
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

The effects of the ε-greedy

It does not change the limits of the equation

$$\lim_{t\to\infty} Q_{a_i}(t) = \underbrace{\lim_{t\to\infty} Ce^{-x_i\alpha t}}_{0} + \underbrace{\lim_{t\to\infty} E[r_{a_i}]}_{E[r_{a_i}]} = E[r_{a_i}]$$

But changes the shape of the function and associated direction field



Summary of the analysis (roughly speaking)

Summary of the analysis (roughly speaking)

Expected Rewards

are the values to wich the Q-values will converge to

Summary of the analysis (roughly speaking)

Expected Rewards

are the values to wich the Q-values will converge to

Speeds

determine the paths that the Q-values will follow to get there

Summary of the analysis (roughly speaking)

Expected Rewards

are the values to wich the Q-values will converge to

Speeds

determine the paths that the Q-values will follow to get there

Intersection points

define if the Q-values will ever get there

System of difference equations

System of difference equations

A and B payoff tables	X and Y strategy vectors	Q_a and Q_b Q-values vectors

System of difference equations

A and B payoff tables	X and Y strategy vectors	Q_a and Q_b Q-values vectors

$$Q_{a_i}(t+1) = Q_{a_i}(t) + x_i(t)\alpha(\sum_j a_{ij} y_j(t) - Q_{a_i}(t))$$

$$Q_{b_i}(t+1) = Q_{b_i}(t) + y_i(t)\alpha(\sum_j b_{ij} x_j(t) - Q_{b_i}(t))$$

$$x_i(t) = \begin{cases} (1-\varepsilon) + (\varepsilon/n), & \text{if } Q_{a_i}(t) \text{ is currently the highest} \\ \varepsilon/n, & \text{otherwise} \end{cases}$$

$$y_i(t) = \begin{cases} (1-\varepsilon) + (\varepsilon/n), & \text{if } Q_{b_i}(t) \text{ is currently the highest} \\ \varepsilon/n, & \text{otherwise} \end{cases}$$

Prisoner's Dilemma

$$A = \begin{bmatrix} 1 & 5 \\ 0 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 5 & 3 \end{bmatrix}$$

Prisoner's Dilemma

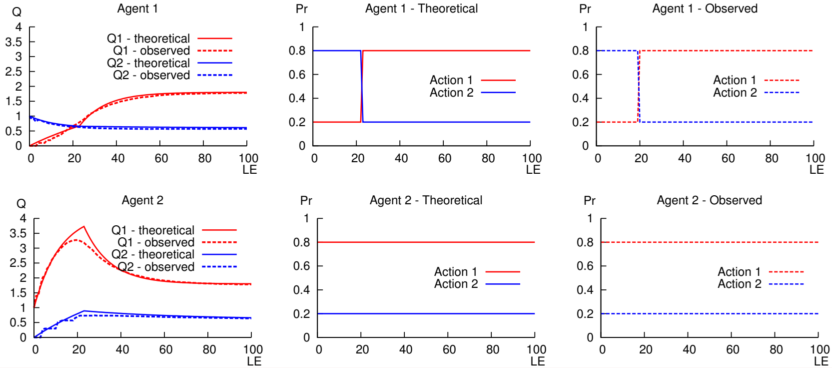
$$A = \begin{bmatrix} 1 & 5 \\ 0 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 5 & 3 \end{bmatrix}$$

$$Q_a = [0,1],\ Q_b = [1,0],\ \alpha = 0.1,\ \varepsilon = 0.4$$
$$X = [0.2, 0.8],\ Y = [0.8, 0.2].$$

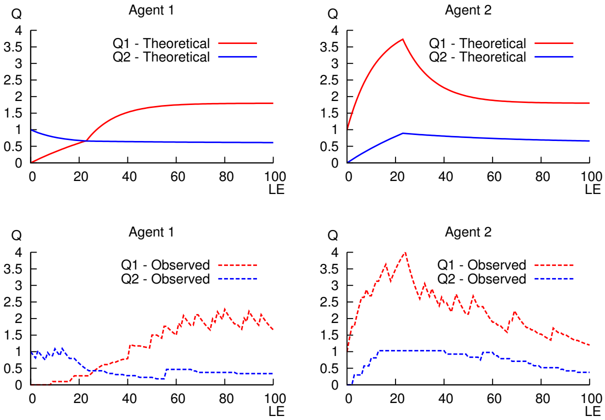
Prisoner's Dilemma

$$A = \begin{bmatrix} 1 & 5 \\ 0 & 3 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 5 & 3 \end{bmatrix}$$

$$Q_a = [0,1],\ Q_b = [1,0],\ \alpha = 0.1,\ \varepsilon = 0.4$$
$$X = [0.2, 0.8],\ Y = [0.8, 0.2].$$



Prisoner’s Dilemma



CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Battle of the Sexes

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

Battle of the Sexes

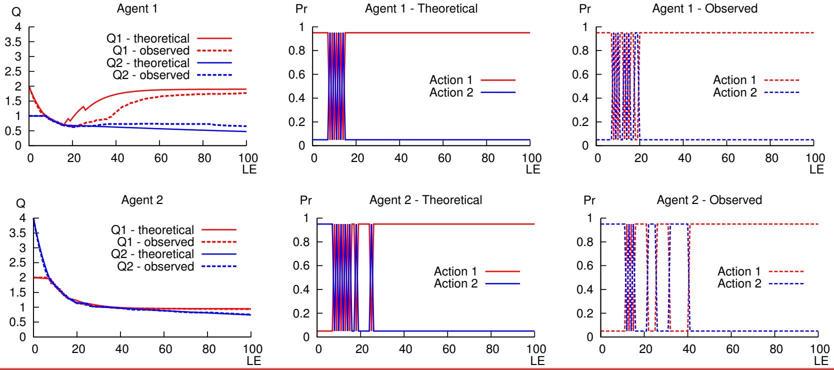
$$A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

$$Q_a = [2, 1],\ Q_b = [2, 4],\ \alpha = 0.1,\ \varepsilon = 0.1$$
$$X = [0.95, 0.05],\ Y = [0.05, 0.95].$$

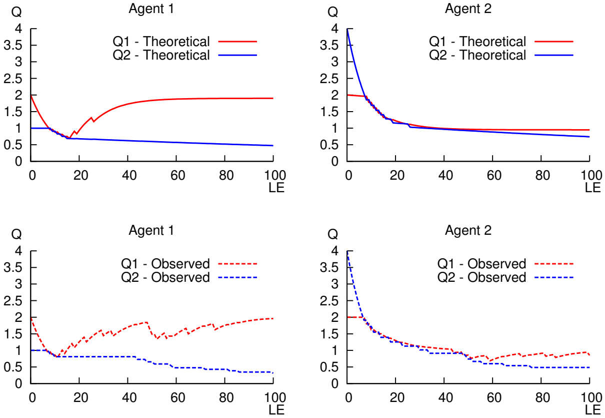
Battle of the Sexes

$$A = \begin{bmatrix} 2 & 0 \\ 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

$$Q_a = [2,1],\ Q_b = [2,4],\ \alpha = 0.1,\ \varepsilon = 0.1$$
$$X = [0.95, 0.05],\ Y = [0.05, 0.95].$$



Battle of the Sexes



SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

A game with no equilibrium

$$A = \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix}$$

A game with no equilibrium

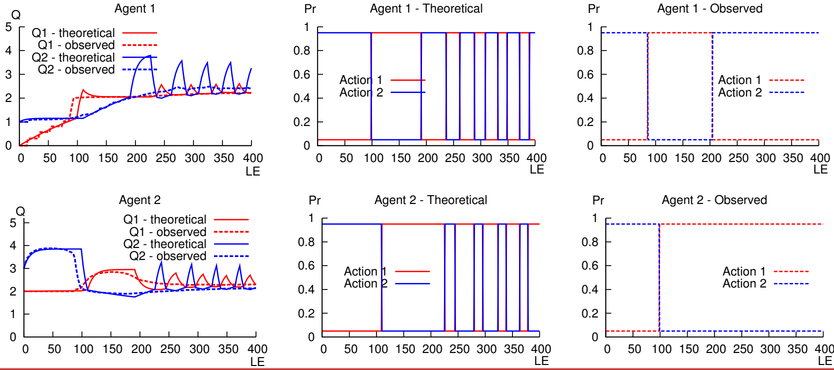
$$A = \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix}$$

$$Q_a = [0, 1],\ Q_b = [2, 3],\ \alpha = 0.1,\ \varepsilon = 0.1$$
$$X = [0.05, 0.95],\ Y = [0.05, 0.95].$$

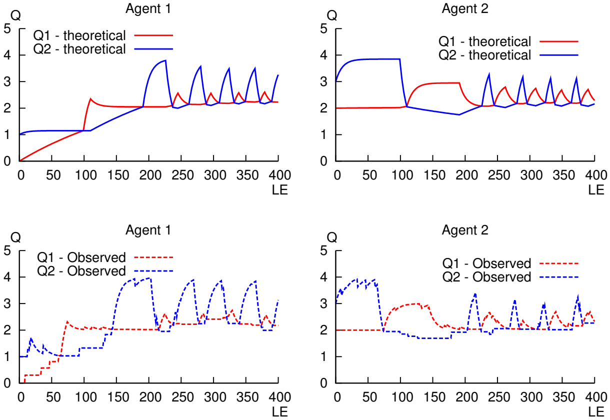
A game with no equilibrium

$$A = \begin{bmatrix} 2 & 3 \\ 4 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix}$$

$$Q_a = [0,1],\ Q_b = [2,3],\ \alpha = 0.1,\ \varepsilon = 0.1$$
$$X = [0.05, 0.95],\ Y = [0.05, 0.95].$$



A game with no equilibrium



SWINBURNE
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Conclusions

- Presented a model for the dynamics of Multiagent Q-learning with ε-greedy exploration
  - Studied a continuous-time version of the Q-learning update rule
  - Investigated how the presence of other agents and the ε-greedy mechanism affect it

SWIN BUR NE
CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Conclusions

- Presented a model for the dynamics of Multiagent Q-learning with ε-greedy exploration
  - Studied a continuous-time version of the Q-learning update rule
  - Investigated how the presence of other agents and the ε-greedy mechanism affect it
- Defined a system of difference equations
  - Model the expected evolution of the Q-values
  - Derive the expected behaviour from the Q-values

SWINBURNE

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

Conclusions

- Presented a model for the dynamics of Multiagent Q-learning with ε-greedy exploration
  - Studied a continuous-time version of the Q-learning update rule
  - Investigated how the presence of other agents and the ε-greedy mechanism affect it
- Defined a system of difference equations
  - Model the expected evolution of the Q-values
  - Derive the expected behaviour from the Q-values
- The evaluation of the model in typical games has shown its feasibility

Future Works

CENTRE FOR COMPLEX SOFTWARE SYSTEMS AND SERVICES

- Extend the model to multi-state scenarios
- Develop techniques for the visualization of the agents' behaviour

Dynamic Analysis of Multiagent Q-learning with ε-greedy Exploration

Eduardo R. Gomes
Ryszard Kowalczyk

Intelligent Agent Technology - CS3 - FICT
egomes@groupwise.swin.edu.au