سیستم‌های چندعاملی

درس ۲۰

یادگیری چندعاملی

Multiagent Learning

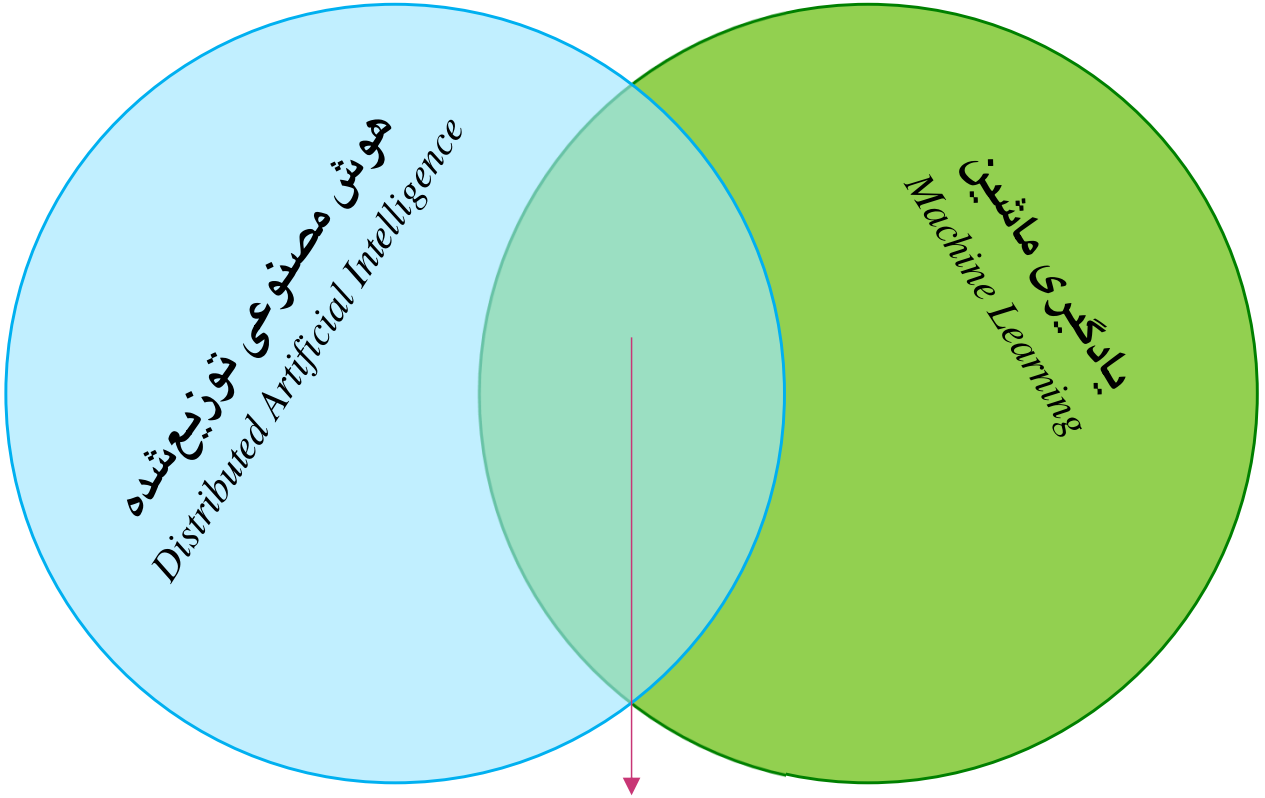
کاظم فولادی قلعه
دانشکده مهندسی، دانشکدگان فارابی
دانشگاه تهران

http://courses.fouladi.ir/mas

یادگیری چندعاملی

محل تقاطع هوش مصنوعی توزیع‌شده و یادگیری ماشین

MULTIAGENT LEARNING



* نیاز شدیدی وجود دارد که سیستم‌های چندعاملی به قابلیت‌های یادگیری مجهز شود.

* گسترش دید یادگیری ماشین در قالب یادگیری چندعاملی، شکل متفاوتی از یادگیری را متمایز از دیدگاه سنتی به یادگیری ایجاد می‌کند که منجر به تکنیک‌ها و الگوریتم‌های جدیدی برای یادگیری ماشین می‌شود.

سیستم‌های چندعاملی

یادگیری چندعاملی

۱

مشخصه‌های عمومی

یادگیری متمرکز دربرابر یادگیری نامتمرکز

یادگیری متمرکز *Centralized Learning*		یادگیری نامتمرکز *Decentralized Learning*
یادگیری مجزا *Isolated Learning*		یادگیری تعاملی *Interactive Learning*
یادگیری توسط یک عامل تنها صورت می‌گیرد. هیچ تعاملی با سایر عامل‌ها وجود ندارد.		عامل‌های متعدد در یک فرآیند یادگیری مشابه درگیر می‌شوند.
چندین عامل یادگیرنده‌ی متمرکز ممکن است سعی کنند اهداف متفاوت یا مشابهی را در یک زمان به‌دست آورند.		گروه‌های متعددی از عامل‌ها ممکن است سعی کنند اهداف متفاوت یا مشابهی را در یک زمان به‌دست آورند.

یک عامل واحد ممکن است در یک زمان
در چندین فرآیند یادگیری متمرکز/نامتمرکز شرکت نماید.

پارامترهای متفاوت‌کننده یادگیری چندعاملی

DIFFERENCING FEATURES

- درجه‌ی نامتمرکزسازی
  The Degree of Decentralization
- ویژگی‌های مختص اندرکنش
  Interaction-Specific Features
- ویژگی‌های مختص درگیری
  Involvement-Specific Features
- ویژگی‌های مختص هدف
  Goal-Specific Features
- روش یادگیری
  The Learning Method
- فیدبک یادگیری
  The Learning Feedback

پارامترهای متفاوت‌کننده یادگیری چندعاملی

درجه‌ی نامتمرکزسازی

THE DEGREE OF DECENTRALIZATION

درجه‌ی نامتمرکزسازی
The Degree of Decentralization

توزیع‌شدگی
Distributedness

موازی‌گرایی
Parallelism

ویژگی‌های مختص اندرکنش
Interaction-Specific Features

ویژگی‌های مختص درگیری
Involvement-Specific Features

ویژگی‌های مختص هدف
Goal-Specific Features

روش یادگیری
The Learning Method

فیدبک یادگیری
The Learning Feedback

پارامترهای متفاوت‌کننده یادگیری چندعاملی

ویژگی‌های مختص اندرکنش

INTERACTION-SPECIFIC FEATURES

طبقه‌بندی اندرکنش‌ها (تعامل‌ها)
برای تحقق یک فرآیند یادگیری نامتمرکز لازم است:

- **سطح اندرکنش**
  The level of interaction
- **دوام اندرکنش**
  The persistence of interaction
- **بسامد اندرکنش**
  The frequency of interaction
- **تغییرپذیری اندرکنش**
  The variability of interaction

درجه‌ی نامتمرکزسازی
The Degree of Decentralization
ویژگی‌های مختص اندرکنش
Interaction-Specific Features
ویژگی‌های مختص درگیری
Involvement-Specific Features
ویژگی‌های مختص هدف
Goal-Specific Features
روش یادگیری
The Learning Method
فیدبک یادگیری
The Learning Feedback

پارامترهای متفاوت‌کننده یادگیری چندعاملی

ویژگی‌های مختص درگیری

INVOLVEMENT-SPECIFIC FEATURES

ویژگی‌هایی که درگیری یک عامل
در فرآیند یادگیری را مشخص می‌کنند:

اهمیت درگیری
The relevance of involvement

نقش ایفا شده در حین درگیری
The role played during involvement

درجه‌ی نامتمرکزسازی
The Degree of Decentralization
ویژگی‌های مختص اندرکنش
Interaction-Specific Features
ویژگی‌های مختص درگیری
Involvement-Specific Features
ویژگی‌های مختص هدف
Goal-Specific Features
روش یادگیری
The Learning Method
فیدبک یادگیری
The Learning Feedback

پارامترهای متفاوت‌کننده یادگیری چندعاملی

ویژگی‌های مختص هدف

GOAL-SPECIFIC FEATURES

ویژگی‌هایی که هدف یادگیری را مشخص می‌کنند:

نوع بهبود اکتسابی توسط یادگیری
Type of improvement achieved by learning

سازگاری اهداف یادگیری عامل‌ها
Compatibility of learning goals of agents

درجه‌ی نامتمرکزسازی
The Degree of Decentralization
ویژگی‌های مختص اندرکنش
Interaction-Specific Features
ویژگی‌های مختص درگیری
Involvement-Specific Features
ویژگی‌های مختص هدف
Goal-Specific Features
روش یادگیری
The Learning Method
فیدبک یادگیری
The Learning Feedback

پارامترهای متفاوت‌کننده یادگیری چندعاملی

روش یادگیری

THE LEARNING METHOD

انواع مختلف روش‌های یادگیری
(بر اساس میزان تلاش لازم برای یادگیری):

یادگیری با تکرار
Rote learning

یادگیری از دستورالعمل توسط مشورت
Learning from instruction and by advice taking

یادگیری از مثال‌ها توسط تمرین
Learning from examples and by practice

یادگیری با قیاس
Learning by analogy

یادگیری با کشف
Learning by discovery

درجه‌ی نامتمرکزسازی
The Degree of Decentralization
ویژگی‌های مختص اندرکنش
Interaction-Specific Features
ویژگی‌های مختص درگیری
Involvement-Specific Features
ویژگی‌های مختص هدف
Goal-Specific Features
روش یادگیری
The Learning Method
فیدبک یادگیری
The Learning Feedback

پارامترهای متفاوت‌کننده یادگیری چندعاملی

فیدبک یادگیری

THE LEARNING FEEDBACK

فیدبک یادگیری،
سطح کارآیی حاصل‌شده تا کنون را نشان می‌دهد.

یادگیری با نظارت (مربی)
Supervised learning (teacher)

یادگیری بی‌نظارت (مشاهده‌گر)
Unsupervised learning (observer)

یادگیری تقویتی (نقاد)
Reinforcement learning (critic)

درجه‌ی نامتمرکزسازی
The Degree of Decentralization

ویژگی‌های مختص اندرکنش
Interaction-Specific Features

ویژگی‌های مختص درگیری
Involvement-Specific Features

ویژگی‌های مختص هدف
Goal-Specific Features

روش یادگیری
The Learning Method

فیدبک یادگیری
The Learning Feedback

مسئله‌ی انتساب اعتبار

The Credit-Assignment Problem (CAP)

یک مسئله‌ی پایه در یادگیری ماشین چندعاملی:

مسئله‌ی انتساب اعتبار *The Credit-Assignment Problem*	انتساب مناسب فیدبک کارآیی کلی دریافت شده از محیط به تک‌تک عامل‌های حاضر در محیط

۱) انتساب اعتبار میان‌عاملی
Inter-Agent CAP

انتساب اعتبار یک تغییر کارآیی کلی
به کنش‌های خارجی عامل‌ها

مثلاً: کدام عامل موجب پیروزی تیم شده است؟

۲) انتساب اعتبار درون‌عاملی
Intra-Agent CAP

انتساب اعتبار یک کنش خاص خارجی عامل
به استنتاج‌ها و تصمیم‌های داخلی آن عامل

مثلاً: کدام جزء دانایی عامل مربوطه منجر به پیروزی شد؟

سیستم‌های چندعاملی

یادگیری چندعاملی

۲

یادگیری در بازی‌ها

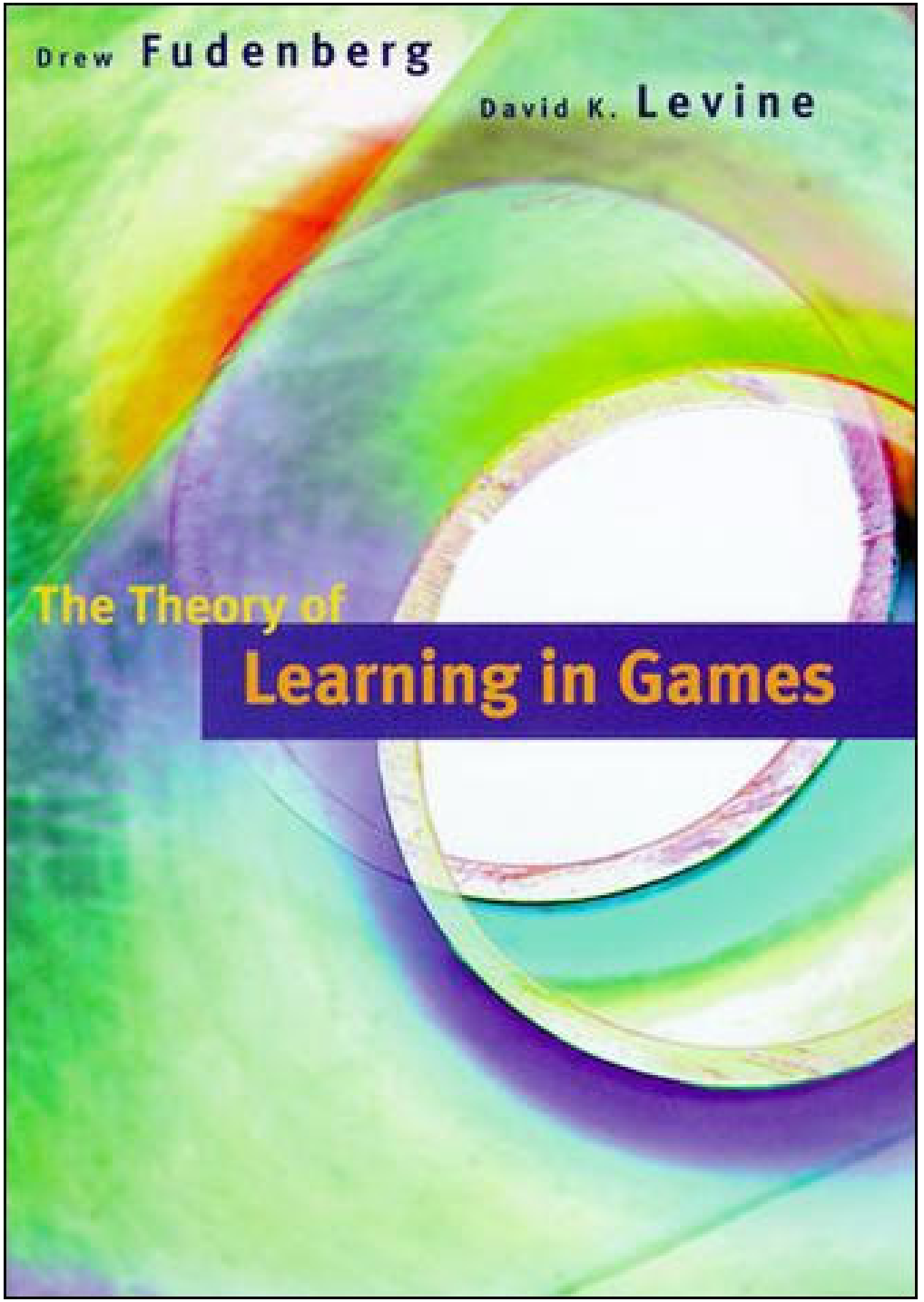
یادگیری در بازی‌ها

LEARNING IN GAMES

مسئله‌ی یادگیری در سیستم‌های چندعاملی می‌تواند در قالب
وادار کردن عامل‌ها به یادگیری چگونگی بازی (انتخاب کنش) در یک بازی دیده شود.

		j	
		c	d
i	a	0,0	5,1
	b	-1,6	1,5

مسئله‌ی یادگیری در بازی‌ها از دهه‌ی *1990* توسط ریاضی‌دان‌ها مطالعه شده است.


Drew Fudenberg
David K. Levine
The Theory of
Learning in Games

یادگیری در بازی‌ها

مدل یادگیری «بازی تصنعی»

FICTITIOUS PLAY

بازی تصنعی
Fictitious Play

هر عامل فرض می‌کند که دیگری یک استراتژی ثابت را بازی می‌کند.
از تاریخچه‌ی گذشته برای ساخت مدلی از عامل دیگر استفاده می‌شود:

عامل i تعداد دفعاتی که عامل j هر یک از استراتژی‌هایش s_j را در هر حالت بازی کرده است را می‌شمارد: $k_i(s_j)$.

$$k_i^t(s_j) = k_i^{t-1}(s_j) + \begin{cases} 1 & \text{if } s_j^{t-1} = s_j, \\ 0 & \text{if } s_j^{t-1} \neq s_j. \end{cases}$$

عامل i بر اساس این مدل، پیش‌بینی می‌کند.
i باور دارد که هر حالت s_j با احتمال زیر رخ می‌دهد:

$$\mathbf{Pr}_i^t[s_j] = \frac{k_i^t(s_j)}{\sum_{\tilde{s}_j \in S_j} k_i^t(\tilde{s}_j)}. \qquad s_j \in S_j$$

عامل i کنشی را انتخاب می‌کند که بالاترین امید سودمندی را دارد.

قضیه (تعادل نش، نقطه‌ی جذب برای بازی تصنعی است):
اگر s یک تعادل نش غالب باشد و t بار بازی شود، آن‌گاه در همه‌ی زمان‌های بزرگ‌تر از t هم بازی می‌شود.

یادگیری در بازی‌ها

مدل یادگیری «بازی تصنعی»: مثال

LEARNING IN GAMES

		j	
		c	d
i	a	0,0	1,2
	b	1,2	0,0

s_i	s_j	$k_i(c)$	$k_i(d)$	$\Pr_i[c]$	$\Pr_i[d]$
a	c	1	0	1	0
b	d	1	1	.5	.5
a	d	1	2	1/3	2/3
a	d	1	3	1/4	3/4
a	d	1	4	1/5	4/5

Example of fictitious play.
The matrix is shown above and the values at successive times,
each on a different row, are shown on the table above.
The first row corresponds to time 0.
Note that only i is using fictitious play, j plays the values as in the s_j column.
i's first two actions are stochastically chosen.

یادگیری در بازی‌ها

مدل یادگیری «بازی تصنعی»: مثال (یک ماتریس بازی با یک چرخه‌ی بی‌نهایت)

LEARNING IN GAMES

		j	
		c	d
i	a	0,0	1,1
	b	1,1	0,0

s_i	s_j	$k_i(c)$	$k_i(d)$	$k_j(a)$	$k_j(b)$
		1	1.5	1	1.5
a	c	2	1.5	2	1.5
b	d	2	2.5	2	2.5
a	c	3	2.5	3	2.5
b	d	3	3.5	3	3.5

Example of fictitious play.
A game matrix with an infinite cycle.

یک استراتژی متداول برای رهایی از چرخه، کنش **تصادفی** است.
* وجود چرخه، بیانگر یک تعادل نش استراتژی مخلوط است.

یادگیری در بازی‌ها

مدل یادگیری «دینامیک تکراری»

REPLICATOR DYNAMICS

دینامیک تکراری
Replicator Dynamics

فرض می‌کنیم که کسری از عامل‌ها که استراتژی خاصی را بازی می‌کنند، متناسب با سودمندی دریافتی آن «تولید مثل» می‌شود.

$\phi^t(s)$ = تعداد عامل‌هایی که از استراتژی s در زمان t استفاده می‌کنند.
$\theta^t(s)$ = کسر عامل‌هایی که استراتژی s را در زمان t بازی می‌کنند:

$$\theta^t(s) = \frac{\phi^t(s)}{\sum_{s' \in S} \phi^t(s')}$$

$u^t(s)$ = امید سودمندی عاملی که استراتژی s را در زمان t بازی می‌کند:

$$u^t(s) = \sum_{s' \in S} \theta^t(s') u(s, s'),$$

که در آن $u(s,s')$ سودمندی حاصل از بازی عاملی با استراتژی s در مقابل عاملی با استراتژی s' است.

در این صورت نرخ «تولید مثل» هر عامل متناسب با این است که در گام قبلی چه‌قدر خوب عمل کرده است:

$$\phi^{t+1}(s) = \phi^t(s)(1 + u^t(s)).$$

یادگیری در بازی‌ها

مدل یادگیری «دینامیک تکراری»: قضیه‌ی تعادل نش

REPLICATOR DYNAMICS

قضیه (تعادل نش، حالت ماندگار است):
هر تعادل نش، یک حالت ماندگار برای دینامیک‌های تکراری است و برعکس.

زیرا در تعادل نش، همه‌ی استراتژی‌ها امید پی‌آف یکسانی دارند،
پس جمعیت‌های آنها با کسر یکسانی رشد می‌کند و
اندازه‌های نسبی آنها یکسان باقی می‌ماند.

ممکن است سیستم هرگز همگرا نشود.

یادگیری در بازی‌ها

مدل یادگیری «دینامیک تکراری»: استراتژی‌های پایدار تطوری

REPLICATOR DYNAMICS

استراتژی پایدار تطوری
Evolutionary Stable Strategy (ESS)

A **stable steady state** is one that, after suffering from a small perturbation, is pushed back to the same steady state by the system's dynamics.

یک *ESS* یک استراتژی تعادل است که می‌تواند بر حضور تعداد کوچکی از استراتژی‌های مهاجم پیروز شود:

- اگر تعداد کوچکی از عامل‌ها که برخی استراتژی‌های دیگر را اجرا می‌کنند، به بازی تهاجم کنند، نسبت به عامل‌هایی که *ESS* را بازی می‌کنند، سود کمتری به‌دست می‌آورند.
- **قضیه:** یک *ESS* حالت ماندگار پایدار مجانبی یک دینامیک تکراری است.
- اما عکس آن درست نیست: یک حالت پایدار لزوماً *ESS* نیست.
- *ESS* یک بهبود بیشتر برای مفهوم راه‌حل ارائه شده توسط دینامیک تکراری است.

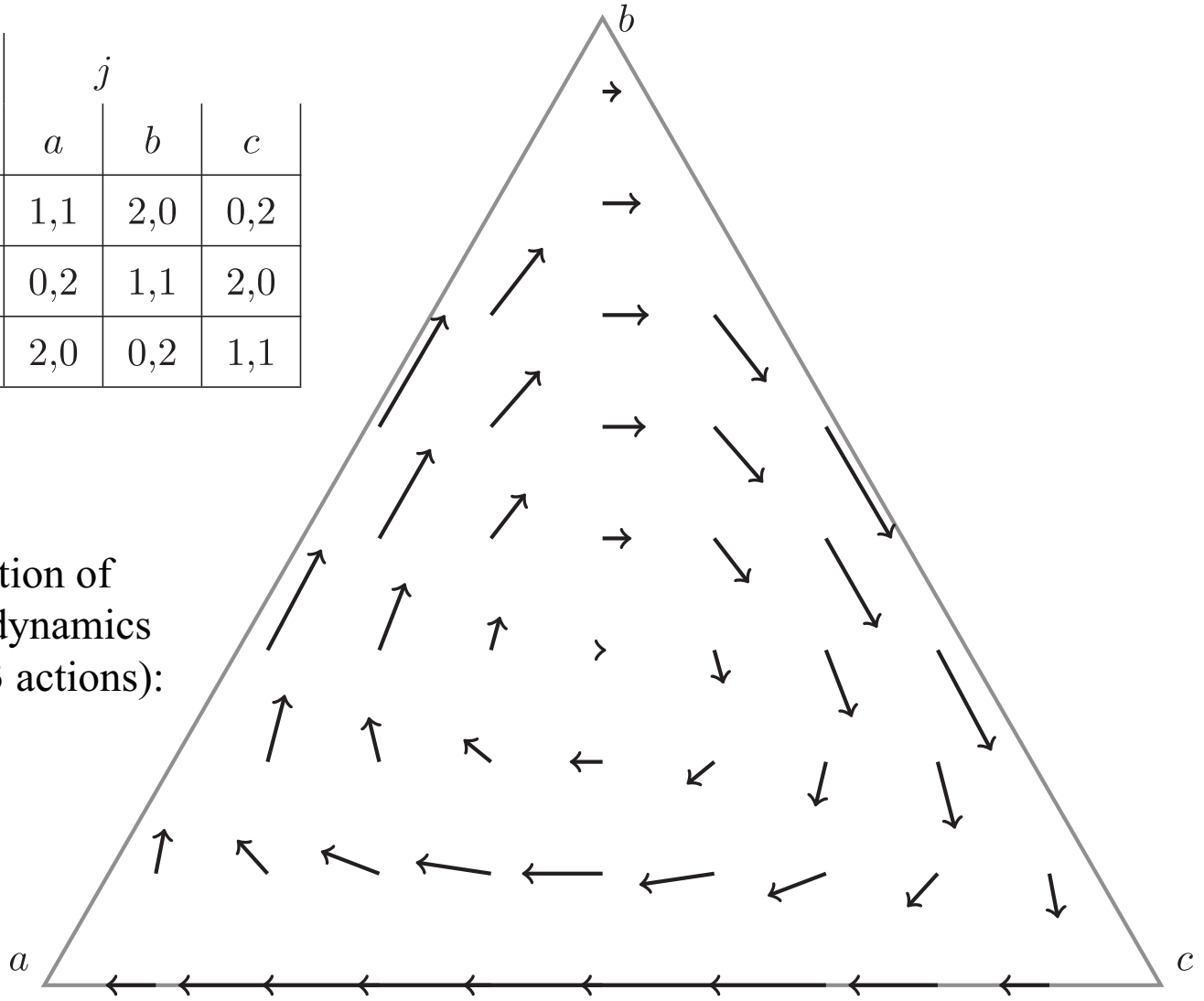
یادگیری در بازی‌ها

مدل یادگیری «دینامیک تکراری»: مثال

REPLICATOR DYNAMICS

		j		
		a	b	c
i	a	1,1	2,0	0,2
	b	0,2	1,1	2,0
	c	2,0	0,2	1,1

Simplex Plot:
Visualization of the evolution of populations in replicator dynamics of the above game (with 3 actions):



سیستم‌های چندعاملی

یادگیری چندعاملی

۳

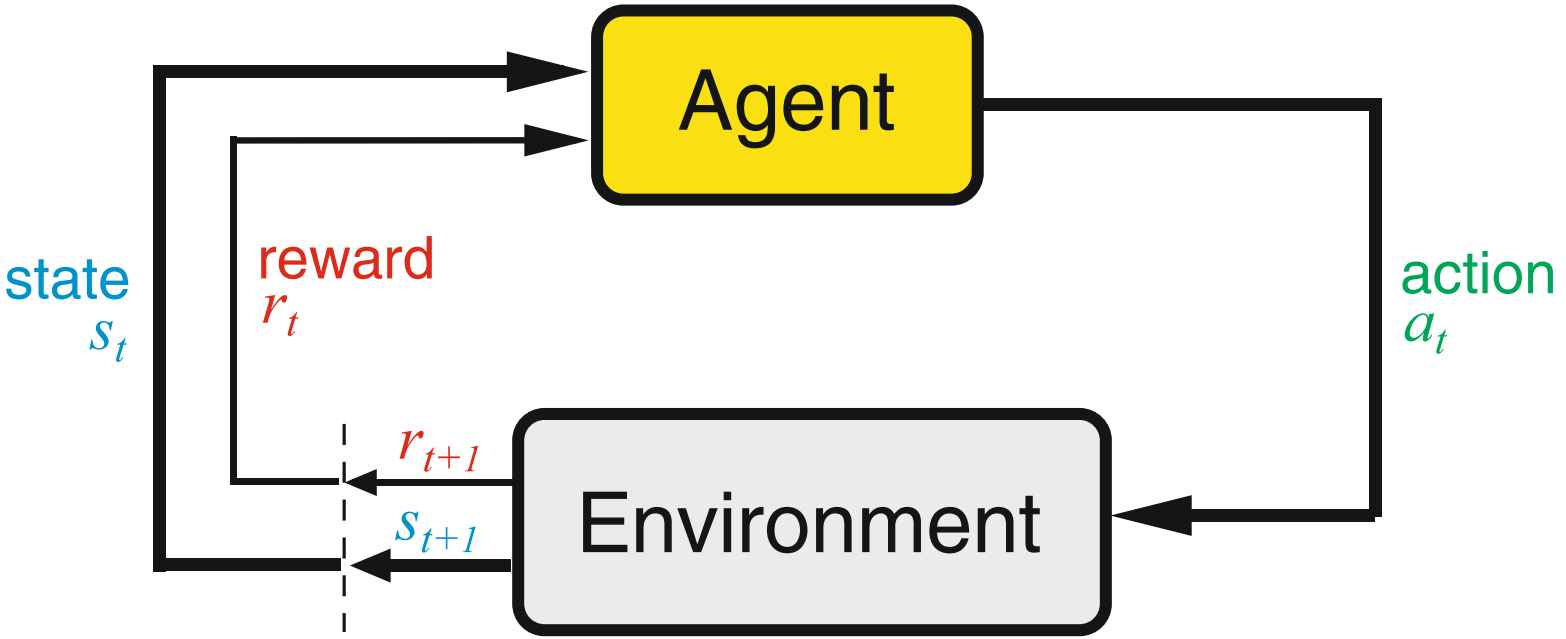
یادگیری تقویتی

یادگیری تقویتی

عامل و محیط

REINFORCEMENT LEARNING

- فیدبک اصلی دریافت شده توسط عامل: در قالب **پاداش‌ها**
- سودمندی عامل توسط تابع پاداش تعریف می‌شود.
- هدف: یادگیری چگونگی کنش به منظور ماکزیمم کردن امید پاداش‌ها



یادگیری تقویتی

REINFORCEMENT LEARNING

عامل در یک محیط MDP یا POMDP قرار دارد.
تنها فیدبک برای یادگیری: ادراک‌ها + پاداش‌ها

عامل باید یک **سیاست** را به یکی از شکل‌های زیر یاد بگیرد:

۱. **مدل گذر** *Transition Model* $T(s, a, s')$ + **تابع ارزش** *Value Function* $U(s)$

۲. **امید سودمندی** *Expected Utility* $Q(a, s)$

مقدار مورد انتظار سودمندی اگر کنش a را در حالت s انجام دهیم و سپس به طور بهینه کنش کنیم.

۳. **سیاست** *Policy* $\pi(s)$

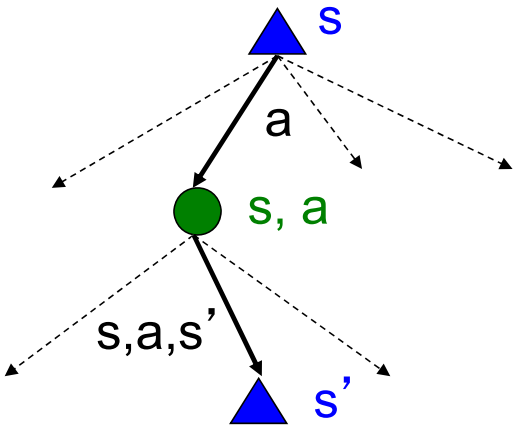
کنشی که در حالت s باید انجام شود.

یادگیری تقویتی

مسائل تصمیم‌گیری مارکوف

MDPs

مؤلفه‌های تعریف یک MDP				
کنش‌های عامل *Actions of Agent*	**حالت‌های محیط** *States of Environment*	**حالت آغازین** *Initial State*	**مدل گذر** *Transition Model*	**تابع پاداش** *Reward Function*
	States $s \in S$, actions $a \in A$	s_0	$T(s,a,s')$	$R(s)$ $R(s,a)$ $R(s,a,s')$



روش‌های راه‌حل		
تکرار ارزش *Value Iteration (VI)*	**تکرار سیاست** *Policy Iteration (PI)*	...

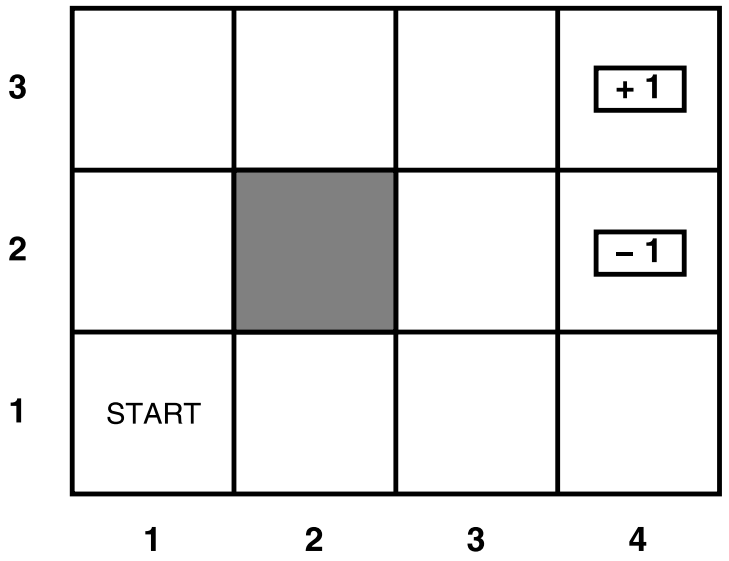
محدودیت‌ها:

* فضای حالت نباید زیاد بزرگ باشد.
* فرض شده است T و R (مدل محیط) معلوم است.

راه‌حل: روش‌های یادگیری تقویتی (Reinforcement Learning)

یادگیری تقویتی

مثال: دنیای ۴ در ۳



کنش‌ها = {Right, Left, Down, Up}

تابع پاداش

Reward function $R(s)$ (or $R(s,a)$, $R(s,a,s')$)

$$= \begin{cases} -0.04 & \text{(جریمه/پنالتی کوچک) برای حالت‌های ناپایانی} \\ \pm 1 & \text{برای حالت‌های پایانی} \end{cases}$$

$$(1,1)_{-.04} \to (1,2)_{-.04} \to (1,3)_{-.04} \to (1,2)_{-.04} \to (1,3)_{-.04} \to \cdots (4,3)_{+1}$$
$$(1,1)_{-.04} \to (1,2)_{-.04} \to (1,3)_{-.04} \to (2,3)_{-.04} \to (3,3)_{-.04} \to \cdots (4,3)_{+1}$$
$$(1,1)_{-.04} \to (2,1)_{-.04} \to (3,1)_{-.04} \to (3,2)_{-.04} \to (4,2)_{-1}.$$

یادگیری تقویتی

هدف

هدف : عامل باید سیاست خود را به‌گونه‌ای تغییر دهد که
در هر حالت **امید مجموع پاداش‌های آینده** را ماکزیمم کند.

در زمان t ، عامل سعی می‌کند تابعی از دنباله‌ی پاداش‌های آینده را ماکزیمم کند، مانند :

مجموع پاداش‌های آینده
Sum of Future Rewards

$$R_t + R_{t+1} + R_{t+2} + \cdots + R_T$$

برای وظایف مقطعی که در زمان T خاتمه می‌یابند.

مجموع پاداش‌های تخفیف‌یافته آینده
Sum of Discounted Future Rewards

$$R_t + \gamma R_{t+1} + \gamma^2 R_{t+2} + \cdots = \sum_{k=0} \gamma^k R_{t+k}$$

برای وظایف ادامه‌دار

نرخ تخفیف
Discount Rate

$$\gamma \in [0, 1]$$

یادگیری تقویتی

سودمندی‌های بهینه و سیاست‌های بهینه

Optimal Utilities and Optimal Policies

سودمندی بهینه‌ی یک حالت s ماکزیمم پاداش تخفیف‌یافته‌ی آینده است که می‌توان از آن حالت با دنبال کردن یک سیاست دریافت کرد:

$$U^*(s) = \max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi, s_0 = s\right]$$

$E[.]$ امید ریاضی نسبت به $P(s'|s,a)$ و $R(s_t)$ با داشتن سیاست منتخب π است.

سیاست بهینه
Optimal Policy

سیاستی است که عبارت فوق را ماکزیمم می‌کند: $\pi^*(s)$

ممکن است سیاست‌های بهینه‌ی $\pi^*(s)$ بسیاری برای یک وظیفه‌ی داده شده وجود داشته باشد، اما همه‌ی آنها مقدار بهینه‌ی $U^*(s)$ مشترک واحدی دارند.

یادگیری تقویتی

معادله‌ی بلمن

THE BELLMAN EQUATION

سودمندی بهینه‌ی یک حالت s ماکزیمم پاداش تخفیف‌یافته‌ی آینده است که
می‌توان از آن حالت با دنبال کردن یک سیاست دریافت کرد:

$$U^*(s) = \max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t) | \pi, s_0 = s\right]$$

برای محاسبه‌ی سودمندی بهینه، می‌توان از تعریف بازگشتی آن استفاده کرد **(معادله‌ی بلمن)**:

$$U^*(s) = R(s) + \gamma \max_{a} \sum_{s'} P(s'|s,a) U^*(s')$$

راه‌حل معادلات بلمن (برای هر حالت یک معادله) برای هر حالت، سودمندی بهینه‌ی هر حالت را تعریف می‌کند.

تعریف بازگشتی مشابهی برای مقادیر Q برقرار است:

$$Q^*(s,a) = R(s) + \gamma \sum_{s'} P(s'|s,a) \max_{a'} Q^*(s',a')$$

یادگیری تقویتی

سیاست‌های حریصانه و انتخاب کنش

THE BELLMAN EQUATION

سیاست بهینه، بر اساس مقادیر سودمندی بهینه، به‌صورت **حریصانه** انتخاب می‌شود:

$$\pi^*(s) = \arg\max_a \sum_{s'} P(s'|s,a)U^*(s')$$

$$= \arg\max_a Q^*(s,a)$$

وقتی از $Q^*(s,a)$ استفاده می‌کنیم، دیگر نیازی به مدل گذر $P(s'|s,a)$ نداریم.

یادگیری تقویتی

تکرار ارزش

VALUE ITERATION

تکرار ارزش یک روش ساده برای محاسبه‌ی سودمندی‌های بهینه در یک MDP است،
وقتی که مدل گذر معلوم باشد.

با مقادیر سودمندی تصادفی $U(s)$ برای هر حالت شروع می‌کنیم و
سپس به‌صورت تکراری قاعده‌ی زیر را اعمال می‌کنیم:

$$U(s) \leftarrow R(s) + \gamma \max_a \sum_{s'} P(s'|s,a)U(s')$$

این قاعده برای هر حالت s به مقدار بهینه‌ی $U^*(s)$ همگرا می‌شود.

تکرار ارزش، معادله‌ی بهینگی بلمن را به یک قاعده‌ی به‌هنگام‌سازی بازگشتی تبدیل می‌کند.

یادگیری تقویتی

یادگیری Q

Q-Learning

یادگیری Q، یک روش یادگیری تقویتی رها از مدل است (به مدل گذر نیاز ندارد).

مقادیر $Q(s,a)$ برای هر (s,a) مقداردهی اولیه می‌شود و سپس تکرار:

$$Q(s,a) \leftarrow (1-\alpha)Q(s,a) + \alpha[R + \gamma \max_{a'} Q(s',a')]$$

برای «تجربه»ی (s,R,a,s') با نرخ یادگیری α انجام می‌شود.

یادگیری Q به مقادیر بهینه‌ی $Q^*(s,a)$ همگرا می‌شود اگر α به‌کندی کاهش یابد.

یادگیری تقویتی

یادگیری Q : اکتشاف

Q-Learning: Exploration

همگرایی یادگیری Q، مستقل از یک سیاست **اکتشاف** به‌خصوص است.

گزینه‌های متداول برای سیاست اکتشاف:

سیاست ε –حریصانه
ε-Greedy Policy

در هر حالت s،
یک کنش تصادفی با احتمال ϵ انتخاب می‌شود
و کنش بهینه a با احتمال $1 - \epsilon$ انتخاب می‌شود.
(برای ϵ کوچک که $0 < \epsilon < 1$)

سیاست توزیع بولتزمن
Boltzmann Distribution Policy

در هر حالت s،
یک کنش a با احتمال زیر (بر اساس توزیع بولتزمن) انتخاب می‌شود:

$$p(a|s) = \frac{\exp(Q(s,a)/\tau)}{\sum_{a'} \exp(Q(s,a')/\tau)}$$

(برای پارامتر دمای τ کاهشی)

یادگیری تقویتی چندعاملی

MULTIAGENT REINFORCEMENT LEARNING

مفروضات

۱	**عامل‌ها** *Agents*	$n > 1$ عامل در دنیا وجود دارد.
۲	**حالت‌ها** *States*	هر حالت $s \in S$ برای همه‌ی عامل‌ها **مشاهده‌پذیر کامل** است.
۳	**بازی‌های استراتژیک محلی** *Local Strategic Game*	هر حالت $s \in S$ یک بازی استراتژیک محلی G_s را با مقادیر پی‌آف متناظر تعریف می‌کند.
۴	**مدل گذر اتفاقی** *Stochastic Transition Model*	یک مدل گذر اتفاقی $p(s'\|s,a)$ داریم که در آن a **کنش توأم** عامل‌هاست.

هدف: محاسبه‌ی یک **سیاست توأم بهینه** (optimal joint action) است

$$\pi^*(s) = (\pi_i^*(s))$$

که پاداش تخفیف‌یافته‌ی آینده را ماکزیمم می‌کند.

چالش در عامل‌های همکار: تضمین اینکه سیاست‌های بهینه‌ی فردی $\pi_i^*(s)$ **هماهنگ‌شده** هستند.

یادگیری تقویتی چندعاملی

یادگیری مستقل

INDEPENDENT LEARNING

یک روی‌کرد این است که اجازه بدهیم هر عامل، یادگیری Q را **به‌طور مستقل** از سایرین اجرا کند.

☹ در این حالت با سایر عامل‌ها به‌مثابه بخشی از یک محیط **پویا** برخورد می‌شود و صراحتاً مدل نمی‌شوند.

☹ مدل گذر $p(s'|s,a_i)$ در این صورت **غیرایستان** (non-stationary) [متغیر با زمان] است.
(زیرا سایر عامل‌ها نیز در حال یادگیری هستند.)

☹ همگرایی یادگیری Q در اینجا دیگر تضمین نمی‌شود.

☺ با این وجود، این روش در عمل استفاده می‌شود و موفقیت‌هایی از آن گزارش شده است.

یادگیری تقویتی چندعاملی

یادگیری کنش مشترک

JOINT ACTION LEARNING

نتایج بهتر در صورتی حاصل می‌شوند که عامل‌ها تلاش کنند یکدیگر را **مدل** کنند.

هر عامل یک تابع ارزش–کنش $Q^{(i)}(s,a)$ را برای همه‌ی جفت‌های **حالت** و **کنش مشترک** نگهداری می‌کند.
در این صورت، یادگیری Q می‌شود:

$$Q^{(i)}(s,a) \leftarrow (1-\alpha)Q^{(i)}(s,a) + \alpha[R + \gamma \max_{a'} Q^{(i)}(s',a')]$$

ملاحظات		
بازنمایی *Representation*	**بهینه‌سازی** *Optimization*	**اکتشاف** *Exploration*
چگونه $Q^{(i)}(s,a)$ را بازنمایی کنیم؟	چگونه $\max_{a'} Q^{(i)}(s',a')$ را محاسبه کنیم؟	چگونه کنش‌های اکتشاف a را انتخاب کنیم؟

یادگیری تقویتی چندعاملی

یادگیری کنش مشترک: بازنمایی

JOINT ACTION LEARNING: REPRESENTATION

بازنمایی
Representation

چگونه
$Q^{(i)}(s,a)$
را بازنمایی کنیم؟

ساده‌ترین گزینه، استفاده از بازنمایی **جدولی** است:
$Q^{(i)}(s,a)$
یک ماتریس است که تعداد درایه‌های آن برابر با تعداد جفت‌های حالت و کنش توأم است.

در این صورت، محاسبه‌ی $\max_{a'} Q^{(i)}(s',a')$ به یک حلقه‌ی for نیاز دارد.

اگر تعداد زیادی عامل وجود داشت،
می‌توان از یک **گراف هماهنگی** استفاده کرد:
در این حالت فرض می‌کنیم:
$Q(s,a) = \sum_j Q_j(s,a_j)$
که در آن a_j کنش توأم زیرمجموعه‌ای از عامل‌هاست.

در این صورت، محاسبه‌ی $\max_{a'} Q^{(i)}(s',a')$ می‌تواند
با **الگوریتم حذف متغیر** انجام شود.

یادگیری تقویتی چندعاملی

یادگیری کنش مشترک: اکتشاف

JOINT ACTION LEARNING: EXPLORATION

اکتشاف
Exploration

چگونه
کنش‌های اکتشاف a
را انتخاب کنیم؟

برای سادگی، فرض می‌کنیم که همه‌ی عامل‌ها پاداش دقیقاً یکسانی می‌گیرند.
در این صورت،
هر عامل می‌تواند یک کنش توأم اکتشافی a را
با توجه به یک توزیع بولتزمن بر روی کنش‌های توأم انتخاب کند.

⇓

لازم است هر عامل کنش توأم یکسانی را نمونه‌برداری کند.

⇓

در نتیجه، هر عامل یادگیری–Q را بر روی کنش‌های توأم
به‌طور یکسان و **به‌صورت موازی** اجرا می‌کند.

⇓

در این صورت، با کل سیستم چندعاملی،
به‌صورت مؤثر در قالب یک سیستم تک‌عاملی «بزرگ» برخورد می‌شود.

یادگیری چندعاملی

۴

یادگیری درباره و از سایر عامل‌ها

یادگیری درباره و از سایر عامل‌ها

LEARNING ABOUT AND FROM OTHER AGENTS

عامل‌ها یاد می‌گیرند که کارآیی انفرادی خودشان را بهبود دهند.

عامل‌ها با پیش‌بینی رفتار سایر عامل‌ها
(ترجیحات، استراتژی‌ها، قصدها و . . .)
بهتر می‌توانند بر روی فرصت‌های موجود سرمایه‌گذاری کنند.

یادگیری درباره و از سایر عامل‌ها

یادگیری نقش‌های سازمانی

LEARNING ORGANIZATIONAL ROLES

فرض می‌کنیم عامل‌ها قابلیت ایفای یکی از چندین نقش را در یک موقعیت داشته باشند.

عامل‌ها نیاز دارند انتساب نقش‌ها را یاد بگیرند تا به‌طور مؤثر یکدیگر را کامل کنند.

چهارچوب UPC، نقش پذیرفته شده در یک موقعیت به‌خصوص را تخمین می‌زند.

U	**سودمندی** *Utility*	وضعیت سود حالت نهایی، اگر عامل نقش داده شده را در وضعیت فعلی بپذیرد.
P	**احتمال** *Probability*	شانس رسیدن به یک حالت پایانی موفق (با داشتن: نقش / وضعیت)
C	**هزینه** *Cost*	هزینه‌ی محاسباتی تحمیل شده‌ی برای یک نقش در یک حالت
	استعداد *Potential*	قابل استفاده بودن یک نقش در یک حالت

UPC Framework

یادگیری درباره و از سایر عامل‌ها

یادگیری نقش‌های سازمانی: چهارچوب نظری

LEARNING ORGANIZATIONAL ROLES

S_k : مجموعه‌ی وضعیت‌ها برای عامل k

R_k : مجموعه‌ی نقش‌ها برای عامل k

$\Downarrow$

هر عامل $|R_k| \cdot |S_k|$ بردار UPC را ایجاد و نگهداری می‌کند.

در حین فرآیند یادگیری:

$$\Pr(r) = \frac{f(U_{rs}, P_{rs}, C_{rs}, Potential_{rs})}{\sum_{j \in R_k} f(U_{js}, P_{js}, C_{js}, Potential_{js})}$$

تابع f با ترکیب مؤلفه‌ها، به یک نقش نمره می‌دهد.

پس از پایان مرحله‌ی یادگیری، نقشی که باید در وضعیت S ایفا شود، می‌شود:

$$r = \arg\max_{j \in R_k} f(U_{js}, P_{js}, C_{js}, Potential_{js})$$

مقادیر UPC با استفاده از یادگیری تقویتی یاد گرفته می‌شوند.

تخمین‌های UPC پس از n به‌روزرسانی: $\hat{U}^n_{rs}, \hat{P}^n_{rs}, \hat{Potential}^n_{rs}$

یادگیری درباره و از سایر عامل‌ها

یادگیری نقش‌های سازمانی: چهارچوب نظری: به‌روزرسانی سودمندی

LEARNING ORGANIZATIONAL ROLES: UPDATING THE UTILITY

S: مجموعه‌ی وضعیت‌هایی که بین زمان پذیرش نقش r در وضعیت s
و رسیدن به یک حالت نهایی F با سودمندی U_F با آنها مواجه می‌شویم.

$\Downarrow$

مقادیر سودمندی برای همه‌ی نقش‌های انتخاب شده در هر وضعیت در S به‌روزرسانی می‌شود:

در حین فرآیند یادگیری:

$$\hat{U}_{rs}^{n+1} \leftarrow (1-\alpha)\cdot\hat{U}_{rs}^{n} + \alpha\cdot U_F$$

یادگیری درباره و از سایر عامل‌ها

یادگیری نقش‌های سازمانی: چهارچوب نظری: به‌روزرسانی احتمال

LEARNING ORGANIZATIONAL ROLES: UPDATING THE PROBABILITY

تابع $O: S \rightarrow [0,1]$:

اگر حالت داده شده موفقیت‌آمیز باشد، ۱ برمی‌گرداند وگرنه صفر.

$\Downarrow$

قاعده‌ی به‌روزرسانی برای احتمال می‌شود:

در حین فرآیند یادگیری:

$$\hat{P}_{rs}^{n+1} \leftarrow (1-\alpha) \cdot \hat{P}_{rs}^{n} + \alpha \cdot O(F)$$

یادگیری درباره و از سایر عامل‌ها

یادگیری نقش‌های سازمانی: چهارچوب نظری: به‌روزرسانی استعداد

LEARNING ORGANIZATIONAL ROLES: UPDATING THE POTENTIAL

$Conf(S)$: اگر در مسیر به‌سوی حالت نهایی، تداخل‌ها تشخیص داده شوند
و از طریق تبادل اطلاعات رفع شوند، ۱ برمی‌گرداند وگرنه صفر.

⇓

قاعده‌ی به‌روزرسانی برای استعداد می‌شود:

در حین فرآیند یادگیری:

$$\hat{P}otential_{rs}^{n+1} \leftarrow (1-\alpha)\cdot \hat{P}otential_{rs}^{n} + \alpha \cdot Conf(S)$$

یادگیری درباره و از سایر عامل‌ها

یادگیری نقش‌های سازمانی: مثال: بازی فوتبال رباتیک

LEARNING ORGANIZATIONAL ROLES: EXAMPLE: ROBOTIC SOCCER GAME

بیشتر پیاده‌سازی‌های تیم‌های فوتبال رباتیک،
از روی‌کرد یادگیری نقش‌های سازمانی استفاده می‌کنند.

استفاده از متدولوژی یادگیری لایه‌ای:

- **مهارت‌های سطح پایین** (مانند: شوت کردن توپ)
- **تصمیم‌گیری سطح بالا** (مانند: به چه کسی پاس داده شود؟)

یادگیری به‌منظور بهره‌برداری از حریف

روی‌کرد مبتنی بر مدل

LEARNING TO EXPLOIT AN OPPONENT: MODEL-BASED APPROACH

روی‌کرد غالب در هوش مصنوعی برای توسعه‌ی استراتژی‌های بازی: **الگوریتم می‌نیماکس**
(فرض می‌کند حریف بدترین حرکت برای ما را انجام می‌دهد.)

یک **مدل دقیق از حریف** می‌تواند برای توسعه‌ی استراتژی‌های بهتر استفاده شود.

مشکل اصلی یادگیری تقویتی، سرعت پایین همگرایی آن است.
روی‌کردهای مبتنی بر مدل سعی می‌کنند تعداد مثال اندرکنش با محیط برای یادگیری را کاهش دهند.

روی‌کرد مبتنی بر مدل: تقسیم فرآیند یادگیری به دو مرحله‌ی جداگانه

استخراج مدل دیگران	استفاده از مدل
استخراج یک مدل از سایر عامل‌ها بر اساس تجربه‌ی گذشته	استفاده از مدل یادگیری شده برای طراحی استراتژی تعامل مؤثر برای آینده

یادگیری به‌منظور بهره‌برداری از حریف

روی‌کرد مبتنی بر مدل: استخراج یک استراتژی بهترین–پاسخ

LEARNING TO EXPLOIT AN OPPONENT: MODEL-BASED APPROACH: INFERRING A BEST-RESPONSE STRATEGY

مدل حریف را در قالب یک DFA بازنمایی می‌کنیم.

قضیه: با داشتن یک مدل حریف در قالب DFA،
یک DFA بهترین–پاسخ وجود دارد که در زمان چندجمله‌ای بر حسب اندازه‌ی DFA حریف قابل محاسبه است.

- The US-L* algorithm infers a DFA that is consistent with the sample of the opponent's behavior
- The US-L* algorithm extends the model according to the three guiding principles:
  - **Consistency:** The new model must be consistent with the give sample
  - **Compactness:** A smaller model is better
  - **Stability**: Should be similar to the previous model as much as possible

سیستم‌های چندعاملی

یادگیری چندعاملی

۵

یادگیری و ارتباطات

کاهش ارتباطات به‌وسیله‌ی یادگیری

REDUCING COMMUNICATION BY LEARNING

یادگیری یکی از روش‌های کاهش بار ارتباطات میان عامل‌هاست.

کاهش ارتباطات به‌وسیله‌ی یادگیری

مثال: شبکه‌ی پیمانی

REDUCING COMMUNICATION BY LEARNING: CONTRACT-NET

در شبکه‌ی پیمانی،
پخش همگانی (Broadcasting) «اعلان وظیفه» لازم است.
با افزایش تعداد مدیران یا وظیفه‌ها، به مشکل مقیاس‌پذیری برمی‌خوریم.

یک مکانیسم منعطف مبتنی بر یادگیری با نام **یادگیری آدرس گیرنده** داریم

- عامل‌ها را قادر می‌کند که در مورد توانایی‌های حل وظیفه‌ی سایر عامل‌ها دانایی به‌دست آورند.
- وظایف می‌توانند مستقیم‌تر نسبت داده شوند.

- از **استدلال مبتنی بر مورد**، برای اکتساب و اصلاح دانایی استفاده می‌شود.

بهبود یادگیری به‌وسیله‌ی ارتباطات

IMPROVING LEARNING BY COMMUNICATION

دو صورت از **بهبود یادگیری به‌وسیله‌ی ارتباطات**	
یادگیری بر اساس ارتباطات سطح پایین *Learning Based on Low-Level Communication*	**یادگیری بر اساس ارتباطات سطح بالا** *Learning Based on High-Level Communication*
برای مثال: تبادل اطلاعات گمشده	برای مثال: توضیح متقابل

بهبود یادگیری به‌وسیله‌ی ارتباطات

مثال: دامنه‌ی شکارچی–شکار

EXAMPLE: PREDATOR-PREY DOMAIN

- شکارچی‌ها Q-learner هستند.
- هر شکارچی ادراک بینایی محدودی دارد.
- تبادل اطلاعات سنسوری: ارتباطات سطح پایین
- آزمایش‌ها نشان می‌دهد که این نتایج یادگیری را به‌وضوح بهتر می‌کند.

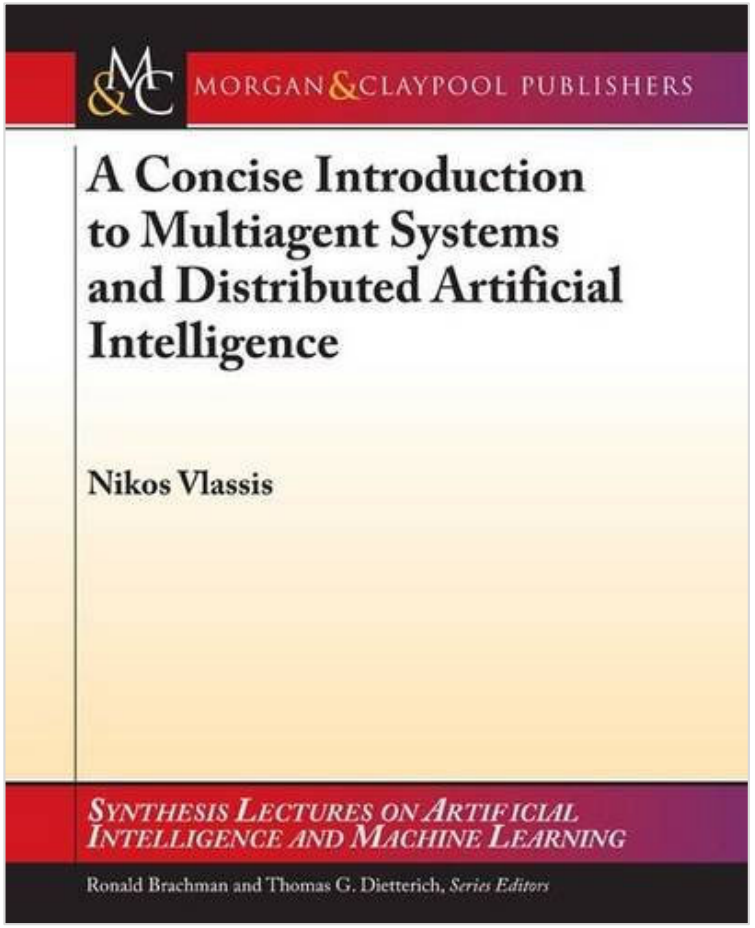
سیستم‌های چندعاملی

یادگیری چندعاملی

۶

منابع

منبع اصلی



Nikos Vlassis,
A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence,
Morgan & Claypool, 2007.
Chapter 7

CHAPTER 7

Learning

In this chapter we briefly address the issue of learning, in particular reinforcement learning which allows agents to learn from delayed rewards. We outline existing techniques for single-agent systems, and show how they can be extended in the multiagent case.

7.1 REINFORCEMENT LEARNING

Reinforcement learning is a generic name given to a family of techniques in which an agent tries to learn a task by directly interacting with the environment. The method has its roots in the study of animal behavior under the influence of external stimuli (Thorndike, 1898). In the last two decades, reinforcement learning has been extensively studied in artificial intelligence, where the emphasis is on how agents can improve their performance in a given task by perception and trial-and-error. The field of single-agent reinforcement learning is mature, with well-understood theoretical results and many practical techniques (Bertsekas and Tsitsiklis, 1996, Sutton and Barto, 1998).

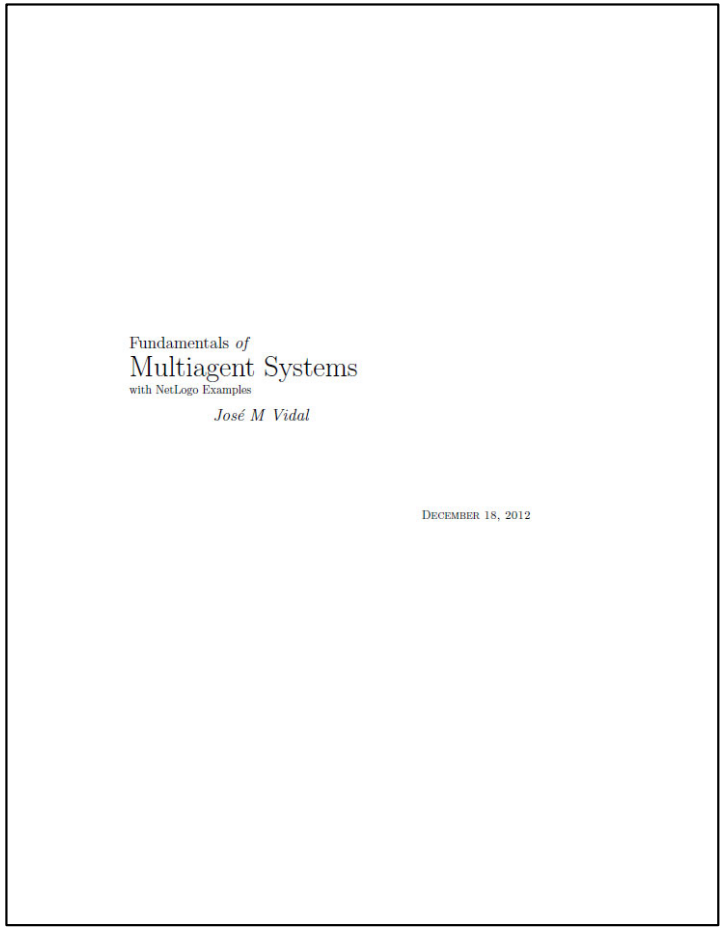
On the other hand, **multiagent reinforcement learning**, where several agents are simultaneously learning by interacting with the environment and with each other, is still an active area of research, with a mix of positive and negative results. The main difficulty in extending reinforcement learning to multiagent systems is that the dynamics of concurrently learning systems can be very complicated, which calls for different approaches to modeling and analysis than those used in single-agent systems.

In this chapter we will outline the theory and some standard algorithms for single-agent reinforcement learning, and then briefly discuss their multiagent extensions. We must unavoidably be laconic as the literature on the topic has grown large; the reader is referred to the book of Greenwald (2007) for a more detailed treatment.

7.2 MARKOV DECISION PROCESSES

In Chapter 2 we described a generic utility-based framework that allows an agent to behave optimally under conditions of uncertainty. In this section we describe a framework that allows an agent to *learn* optimal policies in a variety of tasks.

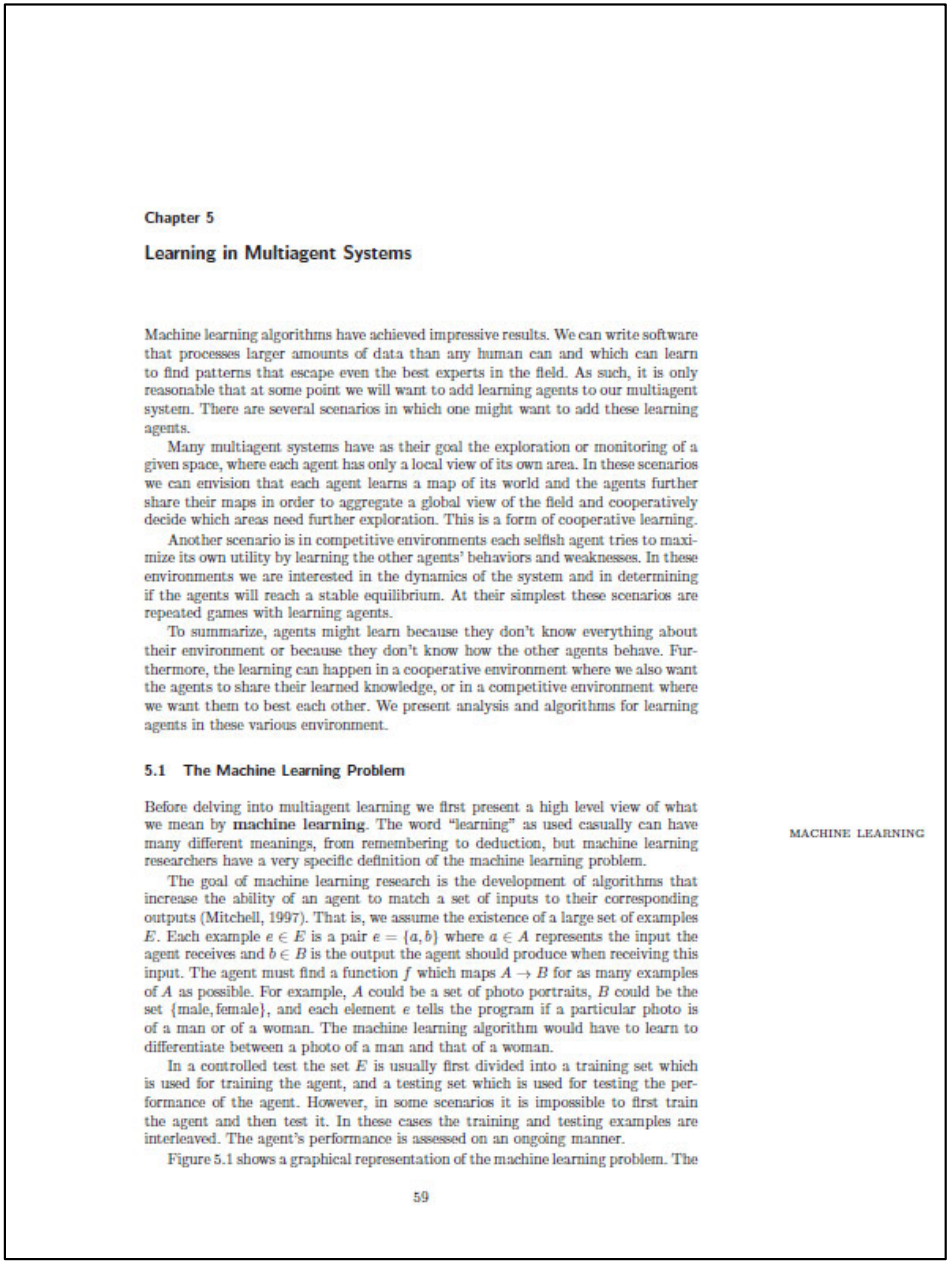
منبع کمکی


Fundamentals *of*
Multiagent Systems
with NetLogo Examples

José M Vidal

December 18, 2012

José M Vidal,
Fundamentals of Multiagent Systems with NetLogo Examples,
Unpublished, 2012.
Chapter 5


Chapter 5

Learning in Multiagent Systems

Machine learning algorithms have achieved impressive results. We can write software that processes larger amounts of data than any human can and which can learn to find patterns that escape even the best experts in the field. As such, it is only reasonable that at some point we will want to add learning agents to our multiagent system. There are several scenarios in which one might want to add these learning agents.

Many multiagent systems have as their goal the exploration or monitoring of a given space, where each agent has only a local view of its own area. In these scenarios we can envision that each agent learns a map of its world and the agents further share their maps in order to aggregate a global view of the field and cooperatively decide which areas need further exploration. This is a form of cooperative learning.

Another scenario is in competitive environments each selfish agent tries to maximize its own utility by learning the other agents' behaviors and weaknesses. In these environments we are interested in the dynamics of the system and in determining if the agents will reach a stable equilibrium. At their simplest these scenarios are repeated games with learning agents.

To summarize, agents might learn because they don't know everything about their environment or because they don't know how the other agents behave. Furthermore, the learning can happen in a cooperative environment where we also want the agents to share their learned knowledge, or in a competitive environment where we want them to best each other. We present analysis and algorithms for learning agents in these various environment.

5.1 The Machine Learning Problem

MACHINE LEARNING

Before delving into multiagent learning we first present a high level view of what we mean by **machine learning**. The word "learning" as used casually can have many different meanings, from remembering to deduction, but machine learning researchers have a very specific definition of the machine learning problem.

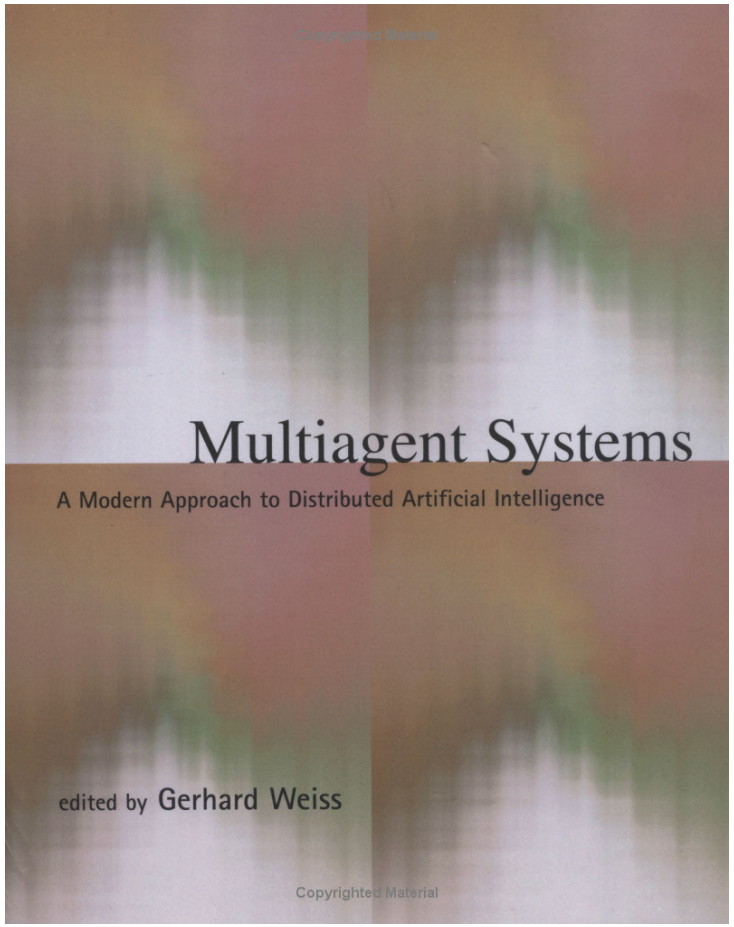
The goal of machine learning research is the development of algorithms that increase the ability of an agent to match a set of inputs to their corresponding outputs (Mitchell, 1997). That is, we assume the existence of a large set of examples E. Each example $e \in E$ is a pair $e = \{a, b\}$ where $a \in A$ represents the input the agent receives and $b \in B$ is the output the agent should produce when receiving this input. The agent must find a function f which maps $A \rightarrow B$ for as many examples of A as possible. For example, A could be a set of photo portraits, B could be the set $\{\text{male}, \text{female}\}$, and each element e tells the program if a particular photo is of a man or of a woman. The machine learning algorithm would have to learn to differentiate between a photo of a man and that of a woman.

In a controlled test the set E is usually first divided into a training set which is used for training the agent, and a testing set which is used for testing the performance of the agent. However, in some scenarios it is impossible to first train the agent and then test it. In these cases the training and testing examples are interleaved. The agent's performance is assessed on an ongoing manner.

Figure 5.1 shows a graphical representation of the machine learning problem. The

59

منبع کمکی



Gerhard Weiss (ed.),
Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence,
MIT Press, 1999.
Chapter 6

6 Learning in Multiagent Systems

Sandip Sen and Gerhard Weiss

6.1 Introduction

Learning and intelligence are intimately related to each other. It is usually agreed that a system capable of learning deserves to be called intelligent; and conversely, a system being considered as intelligent is, among other things, usually expected to be able to learn. Learning always has to do with the self-improvement of future behavior based on past experience. More precisely, according to the standard artificial intelligence (AI) point of view learning can be informally defined as follows:

> *The acquisition of new knowledge and motor and cognitive skills and the incorporation of the acquired knowledge and skills in future system activities, provided that this acquisition and incorporation is conducted by the system itself and leads to an improvement in its performance.*

This definition also serves as a basis for this chapter. Machine learning (ML), as one of the core fields of AI, is concerned with the computational aspects of learning in natural as well as technical systems. It is beyond the scope and intention of this chapter to offer an introduction to the broad and well developed field of ML. Instead, it introduces the reader into learning in multiagent systems and, with that, into a subfield of both ML and distributed AI (DAI). The chapter is written such that it can be understood without requiring familiarity with ML.

The intersection of DAI and ML constitutes a young but important area of research and application. The DAI and the ML communities largely ignored this area for a long time (there are exceptions on both sides, but they just prove the rule). On the one hand, work in DAI was mainly concerned with multiagent systems whose structural organization and functional behavior typically were determined in detail and therefore were more or less fixed. On the other hand, work in ML primarily dealt with learning as a centralized and isolated process that occurs in intelligent stand-alone systems. In the past this mutual ignorance of DAI and ML has disappeared, and today the area of learning in multiagent systems receives broad and steadily increasing attention. This is also reflected by the growing number of publications in this area; see [23, 24, 43, 45, 64, 66, 68] for collections of papers related to learning in multiagent systems. There are two major reasons for this attention, both showing the importance of bringing DAI and ML together: