بسم الله الرحمن الرحیم

سیستم‌های چندعاملی

درس ۱۶

طراحی مکانیسم

Mechanism Design

کاظم فولادی قلعه
دانشکده مهندسی، دانشکدگان فارابی
دانشگاه تهران

http://courses.fouladi.ir/mas

عامل‌های منفعت‌طلب شخصی

SELF-INTERESTED AGENTS

در سیستم‌های چندعاملی **همکارانه**،
فرض می‌شود که هر عامل مطابق آنچه
توسط طراح پروتکل / الگوریتم به وی دستور داده می‌شود، رفتار می‌کند.

اما در حالت عامل‌های **منفعت‌طلب شخصی**،
توسعه‌ی یک پروتکل خیلی ساده نیست:

- **اول**، باید برای یک عامل انگیزه ایجاد کنیم که در جمع شرکت کند!
- **دوم**، ممکن است یک عامل سعی کند که این پروتکل را دستکاری نماید!

طراحی مکانیسم

MECHANISM DESIGN

طراحی مکانیسم، توسعه‌ی پروتکل‌هایی است که برای عامل‌های دخیل در آن،
غیر قابل دستکاری و
به طور انفرادی رسیونال است.

طراحی مکانیسم

نظریه‌ی بازی تعمیم‌یافته – نظریه‌ی بازی معکوس

GENERALIZED GAME THEORY – INVERSE GAME THEORY

نظریه‌ی بازی
Game Theory

در نظریه‌ی بازی، می‌خواهیم برآمد یک بازی را تعیین کنیم.

طراحی مکانیسم
Mechanism Design

نظریه‌ی بازی معکوس
Inverse Game Theory

نظریه‌ی بازی تعمیم‌یافته
Generalized Game Theory

در طراحی مکانیسم، می‌خواهیم یک بازی را **طراحی** کنیم که منجر به یک برآمد مطلوب شود.

طراحی مکانیسم

فرضیات

MECHANISM DESIGN

۱ عامل‌ها *Agents*

$n > 1$ عامل در دنیا وجود دارد.

۲ برآمدها *Outcomes*

مجموعه‌ی برآمدها O که هر برآمد متعلق به آن است $o \in O$

۳ تابع ارزش‌گذاری *Valuation Function*

هر عامل i یک تابع ارزش‌گذاری دارد: $v_i(o, \theta_i)$

تابع ارزش‌گذاری ترجیحات عامل را در مورد برآمدها مشخص می‌کند.

تابع ارزش‌گذاری هر عامل مشخص است، اما پارامتر آن θ_i مجهول است.

۴ نوع عامل *Agent's Types*

پارامتر θ_i نوع عامل i نام دارد.

پارامتر θ_i (نوع عامل) تنها برای عامل i آشکار است.

تابع‌های انتخاب اجتماعی

Social Choice Functions

در طراحی مکانیسم، می‌خواهیم یک **تابع انتخاب اجتماعی** را پیاده‌سازی کنیم:

$$o = f(\theta), \text{ with } \theta = (\theta_i) \text{ the profile of types.}$$

به‌عنوان یک نمونه‌ی نوعی:

$$f(\theta) = \arg\max_{o \in \mathcal{O}} \sum_{i=1}^{n} \nu_i(o, \theta_i)$$

اگر θ را بدانیم، محاسبه‌ی $f(\theta)$ ساده می‌شود.
اما همان طور که گفتیم، مقدار θ را نمی‌دانیم.

شاید بتوانیم محترمانه از هر عامل بپرسیم که نوع خود (θ_i) را گزارش دهد.

مسئله‌ی طراحی مکانیسم (محاسباتی)

The (Computational) Mechanism Design Problem

طراحی مکانیسم محاسباتی،
توسعه‌ی الگوریتم‌های کارآمد برای مسائل بهینه‌سازی است که در آنها برخی پارامترهای تابع هدف در کنترل عامل‌های منفعت‌طلب شخصی است که برای راه‌حل‌های متفاوت، ترجیحات متفاوتی دارند.

بنابراین، چالش در طراحی مکانیسم،
هدایت عامل‌ها به سوی انتخاب برآمد مطلوب o توسط خودشان است!

$$o = f(\theta)$$

پرداخت‌ها و پی‌آف‌ها

PAYMENTS AND PAYOFFS

برای ایجاد انگیزه در عامل i برای شرکت در یک مکانیزم:
وقتی برآمد o اتفاق بیفتد، **پرداخت** $p_i(o)$ را به وی انجام می‌دهیم.

$$u_i(o, \theta_i) = v_i(o, \theta_i) + p_i(o)$$

پی‌آف *Payoff* = **ارزش‌گذاری** *Valuation* + **پرداخت** *Payment*

هر عامل، مراقب ماکزیمم کردن **سود** خودش است.
و این وابسته است به:

۱) ارزش‌گذاری اصلی وی برای یک برآمد

۲) پرداختی که وی از سوی ما دریافت می‌کند.

بازی‌های استراتژیک و تابع‌های برآمد

Strategic Games and Outcome Functions

بر روی مکانیسم‌هایی تمرکز می‌کنیم که خروجی آنها به‌فرم **بازی استراتژیک** است.

$$\mathcal{M} = (A_i, g, p)$$

A_i	**کنش‌ها** *Actions*	مجموعه‌ی کنش‌های موجود برای عامل i
g	**تابع برآمد** *Outcome Function*	نگاشت کنش توأم a به یک برآمد $o \in \mathcal{O}$ $g(a) = o$
p	**تابع پرداخت** *Payment Function*	برای هر عامل یک تابع پرداخت وجود دارد. $p = (p_i(o))$

وقتی عامل‌ها این بازی را انجام می‌دهند،
یک کنش توأم در **تعادل** را انتخاب می‌کنند:

$$a^* = (a_i^*(\theta_i))$$

می‌خواهیم که هر کنش $a_i^*(\theta_i)$ یک کنش **غالب *(dominant)*** باشد:

$$u_i(a_{-i}, a_i^*(\theta_i)) \geq u_i(a_{-i}, a_i), \text{ for any } (a_{-i}, a_i).$$

مسئله‌ی طراحی مکانیسم (محاسباتی)

با خروجی بازی استراتژیک

THE (COMPUTATIONAL) MECHANISM DESIGN PROBLEM

طراحی مکانیسم محاسباتی با خروجی بازی استراتژیک:
با داشتن مجموعه‌ای از برآمدها $o \in \mathcal{O}$،
پروفایلی از توابع ارزش‌گذاری $v_i(o,\theta_i)$ با پارامتر θ_i و یک تابع گزینش اجتماعی $f(\theta)$،
مجموعه‌های کنش مناسب A_i، تابع برآمد $g(a) = o$، و یک تابع پرداخت $p(o)$،
به‌گونه‌ای که
$\theta = (\theta_i)$، و برای توابع پی‌آف $u_i(o,\theta_i) = v_i(o,\theta_i) + p_i(o)$،
$g\left(a^*(\theta)\right) = f(\theta)$ برقرار می‌شود.
که در آن $a^*(\theta) = (a_i^*(\theta_i))$ یک راه‌حل تعادل کنش‌های غالب بازی استراتژیک $\mathcal{M} = (A_i,g,p)$ است.

در این صورت، می‌گوییم که مکانیسم $\mathcal{M}$
تابع گزینش اجتماعی f را در کنش‌های غالب **پیاده‌سازی** می‌کند.

مسئله‌ی طراحی مکانیسم (محاسباتی)

مثال: حراج دومین قیمت

SECOND-PRICE AUCTION

n عامل و **یک** قلم کالا (مثلاً یک منبع در یک شبکه‌ی کامپیوتری) وجود دارد.

می‌خواهیم این کالا را به عاملی نسبت بدهیم که **بیشترین ارزش‌گذاری** را برای آن دارد.

ارزش‌گذاری واقعی هر عامل (نوع آن عامل) θ_i را نمی‌دانیم.

یک برآمد $o \in \{1,2, \dots ,n\}$ اندیس عاملی است که کالا به آن تخصیص می‌یابد.

ارزش‌گذاری عامل i با نوع θ_i:

$$\nu_i(o, \theta_i) = \theta_i \text{ if } o = i \text{ and } 0 \text{ otherwise.}$$

تابع گزینش اجتماعی

$$f(\theta_1, \dots, \theta_n) = \arg\max_i \{\theta_i\}.$$

پرداخت‌ها در یک مکانیسم باید چگونه تنظیم شود تا این مسئله حل شود؟

مسئله‌ی طراحی مکانیسم (محاسباتی)

مثال: حراج دومین قیمت: راه‌حل

SECOND-PRICE AUCTION

اگر داشته باشیم:

$$p_i(o) = 0 \text{ for all } i$$

آن‌گاه مکانیسم

$$\mathcal{M}_1 = (A_i, g, 0)$$

که f را پیاده‌سازی می‌کند، همیشه **رسیونال انفرادی** است.

اگر داشته باشیم:

$$p_i = -\max_{j \neq i}\{\theta_j\}$$

تابع پی‌آف هر عامل می‌شود:

$$u_i(o, \theta_i) = \theta_i - \max_{j \neq i}\{\theta_j\} \text{ if } o = i \text{ and } 0 \text{ otherwise.}$$

آن‌گاه مکانیسم

$$\mathcal{M}_2 = (A_i, g, p)$$

که f را پیاده‌سازی می‌کند، همیشه **رسیونال انفرادی** است.

اصل فاش‌سازی

THE REVELATION PRINCIPLE

آیا لازم است که فضای همه‌ی سه‌تایی‌های

$$\mathcal{M} = (A_i, g, p)$$

را برای یافتن مکانیسمی که f را پیاده‌سازی می‌کند، جستجو کنیم؟

اصل فاش‌سازی به این پرسش پاسخ می‌دهد.

اصل فاش‌سازی

THE REVELATION PRINCIPLE

اصل فاش‌سازی

The Revelation Principle

اگر یک تابع گزینش اجتماعی f در کنش‌های غالب توسط مکانیسم

$$\mathcal{M} = (A_i, g, p)$$

قابل پیاده‌سازی باشد، آن‌گاه f در کنش‌های غالب توسط مکانیسم

$$\mathcal{M}' = (\Theta_i, f, p)$$

نیز قابل پیاده‌سازی است که در آن

از هر عامل خواسته می‌شود که به‌طور ساده نوع خودش را گزارش دهد.

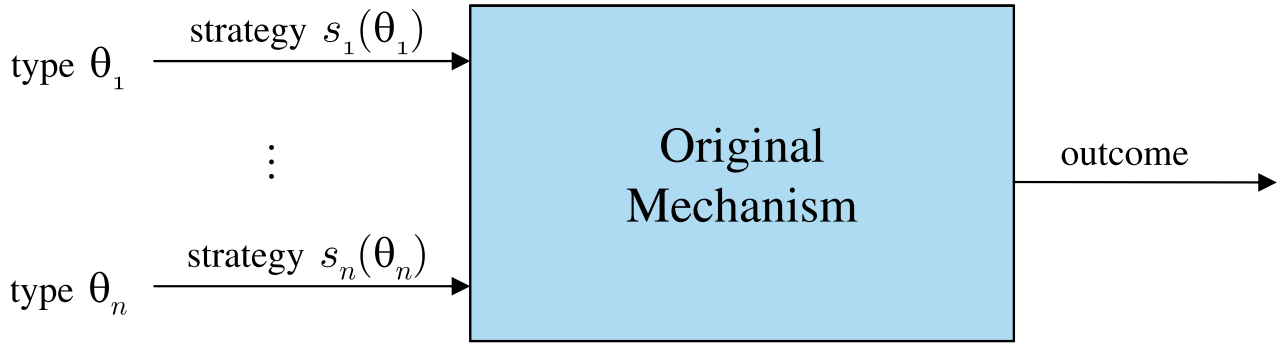
به‌علاوه،

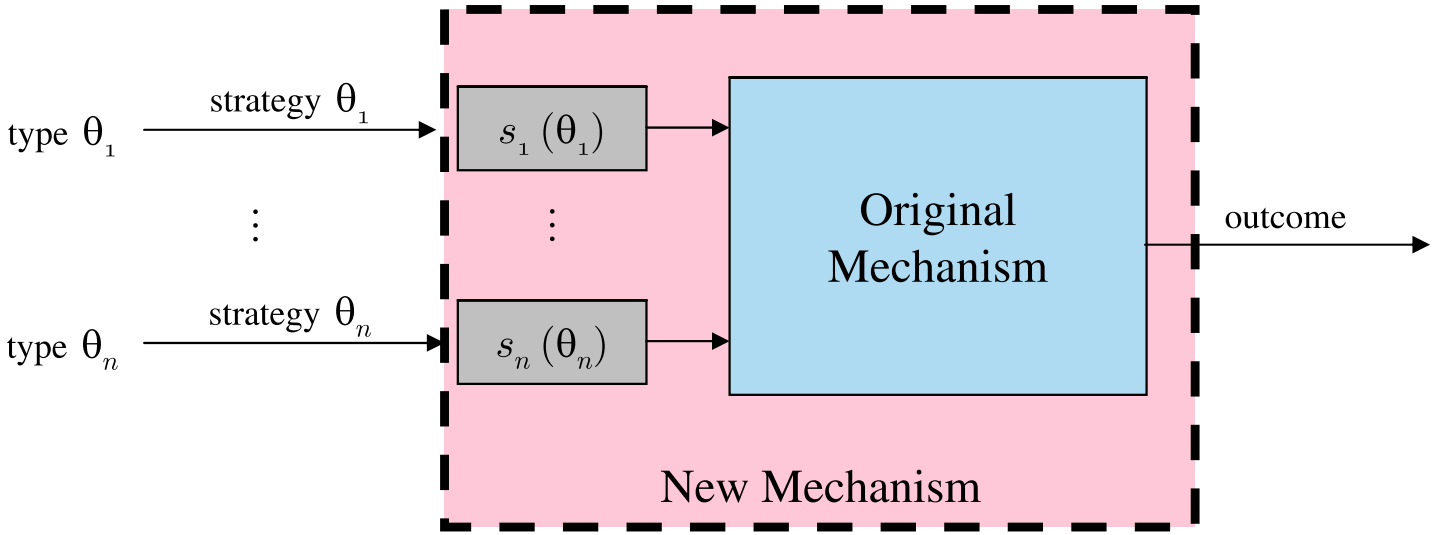
کنش غالب هر عامل i در $\mathcal{M}'$ گزارش دادن **نوع واقعی** خود θ_i است.

اصل فاش‌سازی به ما این امکان را می‌دهد که نتایج متعددی را در طراحی مکانیسم بنا کنیم.

اصل فاش‌سازی

THE REVELATION PRINCIPLE





چگونه می‌توانیم یک مکانیسم جدید با یک **تعادل راست‌گویی** بسازیم،
اگر مکانیسم اصلی دارای تعادل $(s_1, \dots, s_n)$ داده شده باشد؟

مکانیسم افشای مستقیم

DIRECT-REVELATION MECHANISM

مکانیسم افشای مستقیم
Direct-Revelation Mechanism

یک مکانیسم به فرم

$$\mathcal{M} = (\Theta_i, f, p)$$

که در آن از هر عامل به‌صورت ساده خواسته می‌شود که نوعش را گزارش دهد.

مکانیسم دافع استراتژی

STRATEGY-PROOF MECHANISM

مکانیسم دافع استراتژی
Strategy-Proof Mechanism

یک مکانیسم **افشای مستقیم** به فرم

$$\mathcal{M} = (\Theta_i, f, p)$$

که در آن راست‌گویی *(truth-telling)* کنش غالب برای همه‌ی عامل‌ها باشد.

نتیجه‌ی اصل فاش‌سازی:

اگر نتوانیم یک تابع گزینش اجتماعی را توسط یک مکانیسم دافع استراتژی پیاده‌سازی کنیم، آن‌گاه هیچ راهی برای پیاده‌سازی این تابع توسط **هر** مکانیسم عمومی دیگری وجود ندارد.

مکانیسم دافع استراتژی

مثال: حراج دومین قیمت، ارزش مخفی (ویکری)

STRATEGY-PROOF MECHANISM: EXAMPLE: SECOND-PRICE SEALED-BID (VICKREY) AUCTION

یک مکانیسم افشای مستقیم زیر را برای مسئله‌ی حراج در نظر می‌گیریم:

$$\mathcal{M}_3(\Theta_i, f, p)$$

۱) از هر عامل خواسته می‌شود که یک قیمت پیشنهاد بدهد.

۲) کالا به عاملی تخصیص می‌یابد که بالاترین پیشنهاد را داده است.

۳) عامل برنده باید به‌میزان دومین بالاترین پیشنهاد پرداخت انجام بدهد.

در این مکانیسم، راست‌گویی یک کنش غالب برای همه‌ی عامل‌هاست،
پس هر عامل باید ارزش‌گذاری واقعی خودش را پیشنهاد بدهد.

مکانیسم گراوز

THE GROVES MECHANISM

در یک مکانیسم **گراوز** از هر عامل می‌خواهیم که نوع خودش را گزارش دهد.

عامل i، $\hat{\theta}_i$ را گزارش می‌دهد (که می‌تواند با نوع واقعی‌اش θ_i متفاوت باشد).

حال مقدار زیر را محاسبه می‌کنیم:

$$f(\hat{\theta}) = \arg\max_{o \in \mathcal{O}} \sum_{i=1}^{n} \nu_i(o, \hat{\theta}_i)$$

و پرداخت‌های زیر را انجام می‌دهیم:

$$p_i(f(\hat{\theta})) = \sum_{j \neq i} \nu_j(f(\hat{\theta}), \hat{\theta}_j) - h_i(\hat{\theta}_{-i})$$

که در آن $h_i(\hat{\theta}_{-i})$ دلخواه است.

ثابت می‌شود که مکانیسم گراوز دافع استراتژی است.

مکانیسم کلارک

THE CLARKE MECHANISM

مکانیسم کلارک یک حالت خاص از مکانیسم گراوز است، با:

$$h_i(\hat{\theta}_{-i}) = \sum_{j \neq i} \nu_j(f'(\hat{\theta}_{-i}), \hat{\theta}_j)$$

که در آن $f'(\hat{\theta}_{-i})$ یک تابع گزینش اجتماعی است که عامل i از آن کنار گذاشته شده است:

$$f'(\theta) = \arg\max_{o \in \mathcal{O}} \sum_{j \neq i} \nu_j(o, \theta_j)$$

ثابت می‌شود که با وجود شرایطی ساده، **مکانیسم کلارک رسیونال انفرادی است.**

مکانیسم کلارک

مثال: یافتن کوتاه‌ترین مسیر

THE CLARKE MECHANISM: SHORTEST PATH

می‌خواهیم کوتاه‌ترین مسیر بین دو گره‌ی ثابت در یک گراف را پیدا کنیم.

هر یال i در گراف دارای هزینه (طول) $\theta_i \geq 0$ است.
هر یال توسط یک عامل بهره‌برداری می‌شود (مثلاً عامل یک شرکت انتقال است.)
که ترجیح می‌دهد خارج از مسیر قرار گیرد.

ما هزینه‌ی هیچ یالی را نمی‌دانیم (یعنی نوع عامل θ_i مجهول است).

چگونه می‌توانیم کوتاه‌ترین مسیر واقعی را محاسبه کنیم؟

برآمد o لیستی از یال‌ها بر روی راه‌حل کوتاه‌ترین مسیر است.

عامل i دارای نوع θ_i است (= هزینه‌ی یال آن).

ارزش‌گذاری عامل i به‌صورت زیر است:

$$\nu_i(o, \theta_i) = -\theta_i \text{ if } i \in o \text{ and } 0 \text{ otherwise.}$$

تابع گزینش اجتماعی $f(\hat{\theta})$، کوتاه‌ترین مسیر را برای هزینه‌های $\hat{\theta}$ محاسبه می‌کند (مثلاً با الگوریتم دایکسترا).

مکانیسم کلارک

مثال: یافتن کوتاه‌ترین مسیر: راه‌حل

THE CLARKE MECHANISM: SHORTEST PATH

یک **مکانیسم کلارک** می‌تواند این مسئله را حل کند:

از هر عامل می‌خواهیم نوع خود (هزینه‌ی یالش) را اعلام کند.

سپس کوتاه‌ترین مسیر را برای هزینه‌های اعلام‌شده می‌یابیم (با الگوریتم $SSSP$ دایکسترا).

به هر عامل i پرداخت زیر صورت می‌گیرد:

$$p_i = \hat{\theta}_i - C + C'$$

که در آن:

$\hat{\theta}_i$ هزینه‌ی اعلام‌شده توسط عامل i

C طول راه‌حل کوتاه‌ترین مسیر

C' طول راه‌حل کوتاه‌ترین مسیر **بدون** یال i

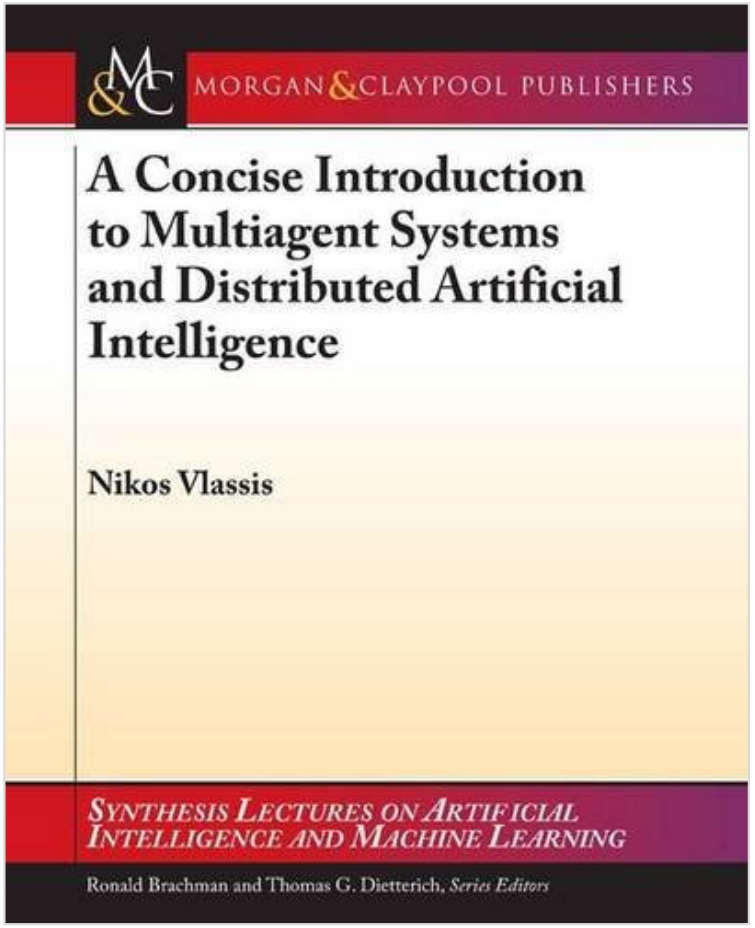
این **مکانیسم رسیونال انفرادی و دافع استراتژی است.**

سیستم‌های چندعاملی

طراحی مکانیسم

منابع

منبع اصلی



Nikos Vlassis,
A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence,
Morgan & Claypool, 2007.
Chapter 6

CHAPTER 6

Mechanism Design

In this chapter we study the problem of mechanism design, which is the development of agent interaction protocols that explicitly take into account the fact that the agents may be self-interested. We discuss the revelation principle and the Vickrey–Clarke–Groves (VCG) mechanism that allows us to build successful protocols in a variety of cases.

6.1 SELF-INTERESTED AGENTS

In the previous chapters we saw several examples of multiagent systems that consist of collaborative agents. The fact that the agents in such systems must collaborate for a common goal allows the development of algorithms, like the coordination algorithms of Chapter 4, in which the agents are assumed to be truthful to each other and behave as instructed. A soccer robot, for instance, would never violate a role assignment protocol like the one in Fig. 4.2, as this could potentially harm the performance of its team.

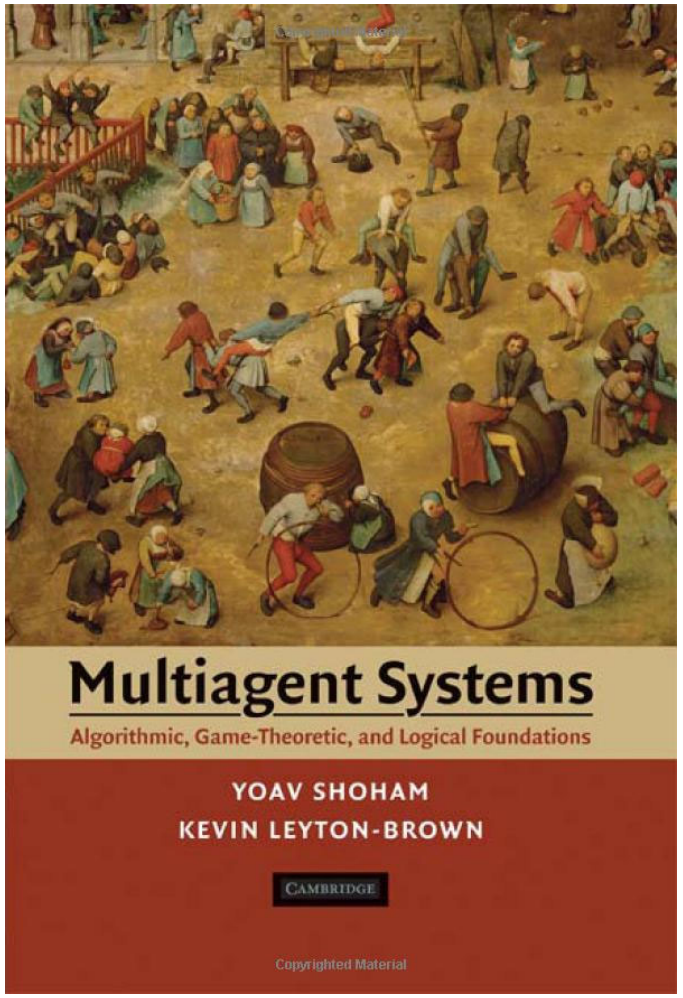
In many practical applications, however, we have to deal with **self-interested** agents, for instance agents that act on behalf of some owner who wants to maximize his or her own profit. A typical case is a software agent that participates in an electronic auction on the Internet. Developing an algorithm or protocol for such a system is a much more challenging task than in the collaborative case. First, we have to motivate an agent to participate in the protocol, which is not *a priori* the case. Second, we have to take into account the fact that an agent may try to *manipulate* the protocol for his own interest, leading to suboptimal results. The latter includes the possibility that the agent may lie, if needed.

The development of protocols that are stable (non-manipulable) and individually rational for the agents (no agent is worse off by participating) is the subject of **mechanism design** or **implementation theory**. As we will see next, a standard way to deal with the above two problems is to provide payments to the agents in exchange for their services.

6.2 THE MECHANISM DESIGN PROBLEM

In Chapter 3 we used the model of a strategic game to describe a situation in which a group of agents interact with each other. The primitives of such a game are the action sets A_i and

منبع کمکی



Yoav Shoham and Kevin Leyton-brown,
Multiagent Systems: Algorithmic, Game-Theoretic, and Logical Foundations,
Cambridge University Press, 2009.
Chapter 10

10 *Protocols for Strategic Agents: Mechanism Design*

As we discussed in the previous chapter, social choice theory is nonstrategic; it takes the preferences of the agents as given, and investigates ways in which they can be aggregated. But of course those preferences are usually not known. What you have, instead, is that the various agents *declare* their preferences, which they may do truthfully or not. Assuming the agents are self interested, in general they will not reveal their true preferences. Since as a designer you wish to find an optimal outcome with respect to the agents' true preferences (e.g., electing a leader that truly reflects the agents' preferences), optimizing with respect to the declared preferences will not in general achieve the objective.

10.1 Introduction

Mechanism design is a strategic version of social choice theory, which adds the assumption that agents will behave so as to maximize their individual payoffs. For example, in an election agents may not vote their true preference.

10.1.1 Example: strategic voting

Consider again our babysitting example. This time, in addition to Will, Liam, and Vic you must also babysit their devious new friend, Ray. Again, you invite each child to select their favorite among the three activities—going to the video arcade (a), playing basketball (b), and going for a leisurely car ride (c). As before, you announce that you will select the activity with the highest number of votes, breaking ties alphabetically. Consider the case in which the true preferences of the kids are as follows:

Will: $b \succ a \succ c$
Liam: $b \succ a \succ c$
Vic: $a \succ c \succ b$
Ray: $c \succ a \succ b$

Will, Liam, and Vic are sweet souls who always tell you their true preferences. But little Ray, he is always figuring things out and so he goes through the follow-