1. Introduction to Economics and Computation

What do economics and computation have to do with each other, and what can be gained through thinking about economics and computation at the same time?

In this chapter, we introduce the main themes of the book, namely the interplay between economic and computational considerations and especially the way in which (i) many socio-economic systems are now computational (with social and economic transactions mediated through computation) and (ii) many computational systems are also economic systems in that they are operated by multiple, often self-interested, parties.

Economics can be described as *the study of decision making by multiple actors, each with individual preferences, capabilities, and information, and motivated to act in regard to these preferences.* Fundamental to economics is to understand what outcomes can be achieved in systems with multiple actors (e.g., what resource allocations, production decisions, actions) under incentive constraints.

Computer science can be described as *the study of the representation and processing of information for the purpose of specific calculation tasks.* Fundamental to computer science is to understand what types of computation can be carried out efficiently, under time, resource and communication constraints.

Taken together, by studying topics at the intersection of economics and computer science ("EconCS" in short) we have in mind,

the analysis of existing systems, and the design of new systems, whose performance depends on addressing both incentive constraints and computational constraints.

We emphasize in particular the need for both economic and computational thinking for the purpose of design. Given a focus on systems involving multiple users or firms, it is essential to consider the effect of design on the behavior of participants. Design influences behavior, which in turn determines the properties of a system.

In the next sections we consider three concrete examples where economic and computational thinking are important. These will lead into a broad discussion of various touch-points at the intersection between economics and computer science.

1.1. The Braess' Paradox

In our first example we analyze a network flow problem. This could be a computer network to route Internet traffic, or a network of streets to route cars from one location to another.

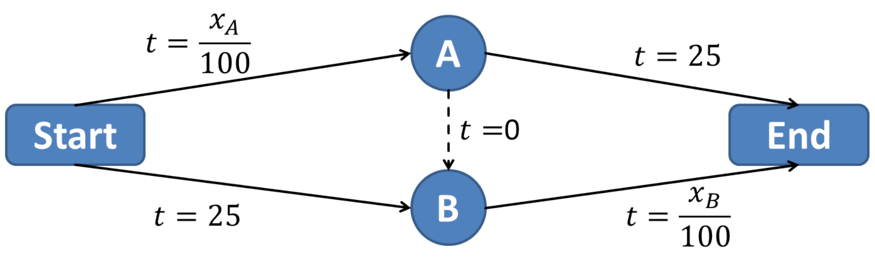


Figure 1.1.: The Braess' Paradox: Traffic flow in equilibrium.

Consider the network shown in Figure 1.1. A flow of $x = 2000$ units needs to move from *Start* to *End*. We first consider the network where the dashed line connecting points A and B is not present. Each unit of flow is associated with a single user and can take a different route, in this case via A or B.

The edges in the network have different delay functions. The time it takes to get from *Start* to A depends on the total flow x_A on that edge, in particular, $t = \frac{x_A}{100}$. The time it takes to get from A to *End* is fixed, with $t = 25$. The second route is symmetric such that the time it takes to get from *Start* to B is fixed, with $t = 25$, and the time it takes to get from B to *End* is $t = \frac{x_B}{100}$.

The first question we ask is: what is the system optimal flow of traffic, and what will be the travel time for each individual user? By symmetry of costs, the optimal traffic flow splits 50:50, such that $x_A = x_B = 1000$. This results in a total traveling time for each user of 35 minutes.

But what if we consider that users are self interested? For this, we model an *incentive constraint*: each user will take the route that minimizes his or her own travel time given the routes adopted by other users. But we immediately see that with 1000 users on each route, no user can improve their own travel time. If a user who previously took route A switches to route B, the number of users on that route increases to 1001, and the resulting travel time increases to $T = 25 + 1001/100 = 35.01$. Thus, each user is best off staying with their current route. The system is *in equilibrium*.

Now, we add a new edge, namely the dashed line connecting A with B. The time to travel this edge is $t = 0$. Thus, users now have an additional route from *Start* to B, namely by going *Start*-A-B. The original travel flow is no longer an equilibrium. No user would ever want to go from *Start* to B, because this takes 25 minutes and even if all 2000 users take the route from *Start*-A-B, this takes only 20 minutes. The same reasoning holds true for A to *End*: taking the route A-B-*End* will always be better.

Thus, each user choose to go from *Start*-A-B-*End*, whatever the actions of other users. This is a dominant strategy, and defines the *equilibrium* of the new system. The paradox is that the travel time for each user has *increased* from previously 35 minutes to now 40 minutes. By adding a new route with zero delay, the travel time has become worse for everyone! This is the famous *Braess' Paradox* named after the mathematician Dietrich Braess.

The Braess' Paradox makes one thing crystal clear: *incentives matter*! The rational self-interested behavior of individuals can lead to socially sub-optimal outcomes in equilibrium. This is a kind of "tragedy of the commons": individual actors use too much of a commonly-shared resource (like a path on a routing network) so that the end result is bad for everyone. Thus, when designing computational systems where the actions of multiple self-interested participants determines the overall outcome, it is important to take into account incentive constraints on the design as well as computational constraints.

Since the advent of the Internet, such systems are ubiquitous. Consider for example e-commerce platforms (eBay, Amazon), online advertising markets (Google, Yahoo!, Microsoft), social and micro-blogging networks (Facebook, Twitter), and Web 2.0 platforms (Wikipedia, Yelp, Tripadvisor, Digg). However, the design of computational systems with multiple participants was a challenge even before the Internet. We see this in the next section.

1.2. Using an Auction for Sharing Compute Resources

Early computers were very costly, shared amongst many users, and in great demand. In the 1960's, researchers at Harvard University used an auction system to determine who would gain access to the PDP-1, the world's first interactive, commercial computer.

Users could mark bids in integer amounts of currency for a block of time on a long roll of transparent paper, starting and ending on quarter hours, and with different colors to indicate the bid amount. The paper was moved up each day so that the next two weeks were always displayed. A user could out bid a current bid (including one of their own bids) by bidding for a block of time at a higher hourly rate, by writing a new bid above the current bid on the schedule. See Figure 1.2.[1]

Different users were allocated different amounts of currency, this budget allowing for a simple priority scheme. A user's outstanding bids for future time could not exceed a user's allocated budget. Bids could be canceled and the currency re-used elsewhere, but a user could not lower a bid, or bid again at some later instance on a block of time where a bid had been canceled. Once an allocation was made to a user and payments collected, the currency was returned to the user and could be used again.

Other rules were designed to preclude fragmentation of time on the shared machine, so that users received large enough contiguous blocks of time. For example:

(i) out-bidding part of an existing bid could only be done at the beginning or end of a block of time; e.g., a bid of \$2 for [1:00,3:00] could be out-bid with a bid of \$3 for [12:30,1:30], but not with a bid of \$3 for [1:30,2:30],

(ii) out-bidding part of an existing bid that was holding an hour or more could only be done while leaving the existing bid with at least an hour on the machine; e.g., a bid of \$3 for

[1] For example, the bid by KSL on Tuesday at 3 Yen (the name given to the virtual currency) for 13:00-15:00 is at rate 3/2 per hour. This out bids the bid by CJ at 1 Yen for 13:00-14:30 (rate 1/1.5) and the first 30 mins of the bid by TM for 14:30-18:00 at 1 Yen (rate 1/3.5). The effect is that the bid of CJ is no longer active and the bid of TM is retained at 1 Yen for 13:00-18:00 (rate 1/3).

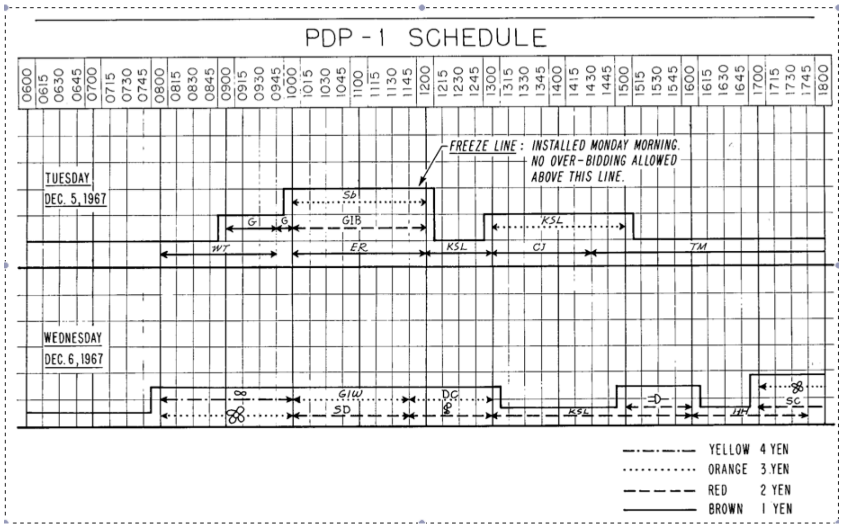


Figure 1.2.: Bidding in the PDP-1 auction (from Sutherland 1969)

[12:30,2:00] but not \$3 for [12:30,2:30] would be allowed given an existing bid of \$2 for [1:00,3:00].

In summarizing the useful properties of the PDP-1 auction, Ivan Sutherland, the designer of the system, remarked:

> *"We have found that under the auction system... our computer utilization is very high... The computer is never idle, as often happens under other allocation schemes, merely because everyone has used up his current monthly allotment of time. If the computer ever is idle, its price automatically becomes attractively low... Although users complain when their bids are preempted, they are generally glad to have a choice between short periods of expensive prime time and long periods of time at less desirable periods..."*

By "preemption," Sutherland is referring to the possibility that bid can be out-bid, perhaps leaving some part of it standing. Sutherland is making the basic case for the use of economic approaches for resource allocation within multi-party computational systems. The auction design allowed users to express the intensity of preference and heterogeneity in regard to time of day and length of time. Prices provided coordination, such that anyone could use the machine when it was under-utilized. At other times those with the highest value could gain access. The effect was to promote *allocative efficiency*, those with the most value could gain access to the resource, and avoid a tragedy of the commons.

Strategic behavior was observed by users of the PDP-1 auction. One example was through a particular pattern of submitting multiple, contiguous bids. The rules of the auction interpreted multiple bids by the same user on consecutive blocks of time as if they were a single bid from the perspective that they could only be out-bid at the beginning or end according to rule (i).

This was intended to allow a user to express a preference for an early portion of time through a bid \$3 for [1:00,1:30] and \$1 for [1:30,3:00]. But this also allowed for strategic behavior.

Let's suppose a user wants to obtain time block [9:00,10:00] for at most \$2. Rather than bid \$2 for [9:00,10:00], the user could bid \$1 for [9:00,9:15] and \$1 for [9:15,10:00]. With the first bid, the user could be out-bid by a bid of \$3 for [9:00,10:00]. But this bid is not possible under rule (ii), since it would leave only 15 minutes (the bid \$1 for [9:00,9:15]) would stand. Instead, another user would need to bid \$2 for [9:00,9:15] and \$2 for [9:15,10:00], for a total of \$4. Thus, by submitting multiple bids the user gains more protection against being out-bid.[2]

There are a multiple reasons why a useful design stance is to preclude the possibility of beneficial strategic behavior. One is to make a system easier to use, so that each user can interact as if he or she is the only user, and straightforwardly express preferences for resources. Second, there are fairness considerations, since not all users may be aware of the possibility of strategic behavior. Third, strategic behavior may lead to inefficient resource allocation where resources are not allocated to those with the highest value. Fourth, the need to adapt bids in response to what other users are doing can place overhead on system infrastructure, with bids continually being modified to gain advantage. We will revisit this concern in the next section, in the context of ad auctions.

1.3. Using an Auction to Allocate Advertisements

Alongside the "organic" search results, which are links to content that an Internet search engine considers to be most relevant for a user query, are sponsored search results. For example, Figure 1.3 shows 3 ads above and 1 ad to the right in response to a query for "flowers." Which ads are displayed to a user is determined via an auction.

Auctions are suitable for selling items without a standardized value. This is the case when selling access to users entering queries into a search engine. Not only is there an essentially unlimited number of different queries, but the value of advertisers can depend on context such as time of day, location, proximity of holidays, and so forth. Ad auctions, for ads adjacent to search results but also for user attention more broadly (e.g., on social network platforms or news sites) have become the main driver of revenue for large Internet firms.

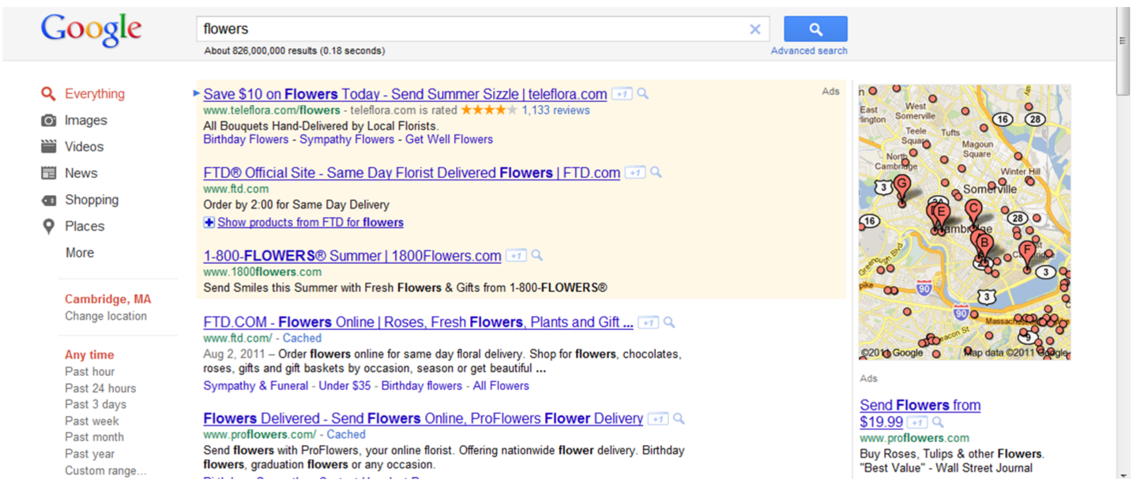
Multiple ad slots are offered for sale in sponsored search, both above and to the right of organic results. Higher slots tend to be more desirable.

Three typical design components of ad auctions are,

(i) bids are placed per-click, with payment only collected from an advertiser in the event of a click,

(ii) bids are evaluated in terms of expected value, with estimates made of the quality of an ad (the probability that the ad will receive a click from the user if allocated in slot 1), and bids with higher expected value receiving higher slots, and

(iii) the payment in the event of a click is not the bid price but a second price, namely the smallest bid amount at which the advertiser would have retained the same slot.



flowers

About 826,000,000 results (0.18 seconds)

Advanced search

Everything
Images
Videos
News
Shopping
Places
More

Cambridge, MA
Change location

Any time
Past hour
Past 24 hours
Past 3 days
Past week
Past month
Past year
Custom range...

Ads

Save $10 on **Flowers** Today - Send Summer Sizzle | teleflora.com
www.teleflora.com/**flowers** - teleflora.com is rated ★★★★★ 1,133 reviews
All Bouquets Hand-Delivered by Local Florists.
Birthday Flowers - Sympathy Flowers - Get Well Flowers

FTD® Official Site - Same Day Florist Delivered **Flowers** | FTD.com
www.ftd.com
Order by 2:00 for Same Day Delivery
Show products from FTD for **flowers**

1-800-**FLOWERS**® Summer | 1800Flowers.com
www.1800**flowers**.com
Send Smiles this Summer with Fresh **Flowers** & Gifts from 1-800-**FLOWERS**®

FTD.COM - **Flowers** Online | Roses, Fresh **Flowers**, Plants and Gift ...
www.ftd.com/ - Cached
Aug 2, 2011 – Order **flowers** online for same day floral delivery. Shop for **flowers**, chocolates, roses, gifts and gift baskets by occasion, season or get beautiful ...
Sympathy & Funeral - Under $35 - Birthday flowers - All Flowers

Flowers Delivered - Send **Flowers** Online, ProFlowers **Flower** Delivery
www.pro**flowers**.com/ - Cached
Send **flowers** with ProFlowers, your online florist. Offering nationwide **flower** delivery. Birthday **flowers**, graduation **flowers** or any occasion.

Ads

Send **Flowers** from $19.99
www.pro**flowers**.com
Buy Roses, Tulips & other **Flowers**.
"Best Value" - Wall Street Journal

Figure 1.3.: An example of sponsored search listings. The ads are shown above and to the right of the organic search results.

Earlier auction designs were first price, or pay-your-bid. This means that in the event of being allocated a slot and receiving a click from a user, the advertiser would pay the amount of its bid.

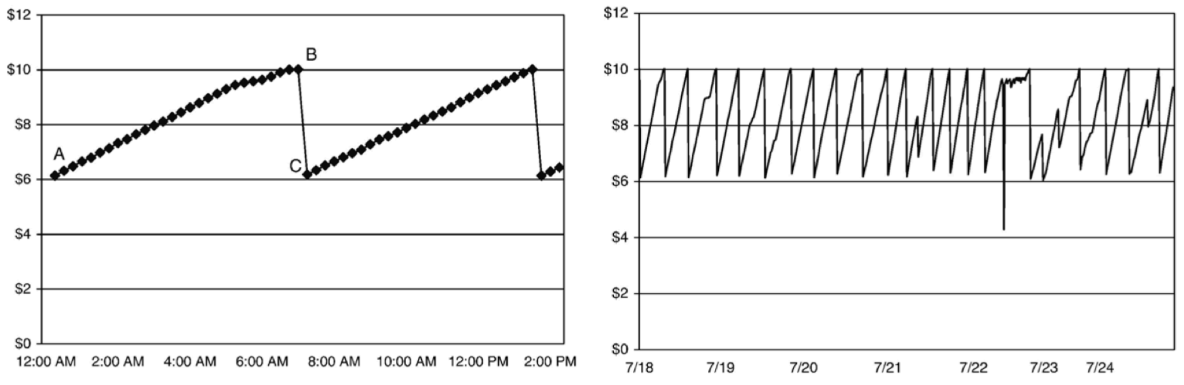
Strategic behavior is beneficial in such auctions. For example, in order to maintain position in a particular slot while pay as little as possible an advertiser should try to bid just enough to maintain position over the next highest bid. Advertisers quickly realized this and adopted "bidding robots" in order to adjust bids in response to other bids. Early auctions didn't estimate the quality of ads, and simply ranked bids by bid amount, with the highest bid receiving the highest slot. Moreover, early auctions published bid amounts. Based on this, it was a simple matter to write a program to monitor an auction and update bids to maximize profit.

Figure 1.4 (a) plots the highest bid submitted for a specific keyword in the Overture search engine from 12:15am to 2:15pm on July 18, 2002.[3] The top two bidders are engaged in a bidding war. Between A and B the two bidders continually out bid each other. At B one of the bidders stops competing for the top slot and bids just enough to retain the second slot instead. By C the other bidder has responded by just bidding enough to gain the top slot, and the cycle begins again. Figure 1.4 (b) shows this sawtooth dynamic continuing for the week following July 18.

Let's consider a simplified explanation for the observed pattern:

Example 1.1. *Let's consider an auction with two slots and where the clickthrough rate (CTR) on slot 1 is 0.1 and slot 2 is 0.02 (irrespective of the ad.) Let CTR_j denote the CTR in slot j. Advertisers have per-click values of \$15, \$11 and \$5.99, for advertisers 1, 2 and 3 respectively.*

[2]User G can be seen to adopt this strategy in Figure 1.2.
[3]The first sponsored search auctions were introduced by Goto.com in 1998, later re-branded as Overture.



a: 14 hours

b: 1 week

Figure 1.4.: Sawtooth Bidding Pattern in First-Price Sponsored Search Auctions (from Edelman and Ostrovsky 2007).

If advertiser i with value v_i for a click wins slot j with CTR_j and pays bid b_i, its expected profit is $CTR_j(v_i - b_i)$, i.e., the difference between value and bid, multiplied by the probability of receiving a click. Typical best-response dynamics between advertisers 1 and 2 (with advertiser 3 always bidding \$5.99) are:

b_1:	6.01	6.01	6.03	6.03	6.05	...	9.98	9.98	10.00	10.00	6.01	...
b_2:	6.00	6.02	6.02	6.04	6.04	...	9.97	9.99	9.99	6.00	6.00	...
b_3:	5.99	5.99	5.99	5.99	5.99	...	5.99	5.99	5.99	5.99	5.99	...

In period 1, advertiser 1 wins slot 1 for \$6.01 and advertiser 2 wins slot 2 for \$6.00. Assuming that advertisers 1 and 3 leave their bids unchanged, then advertiser 2 can now do the following analysis:

- *Bid \$6.00 again: win slot 2, for expected profit of* $0.02(\$11.00 - \$6.00) = \$0.1$.
- *Bid \$6.02: win slot 1, for an expected profit of* $0.1(\$11.00 - \$6.02) = \$0.498$.

Given this, the best response is to bid \$6.02 to win slot 1, since the expected profit is larger in that case. In the next round, advertiser 1 faces a similar decision, coming to the conclusion that a bid of \$6.03 is optimal. This leads to a cyclic bidding pattern, which continues until the bids reach \$10.00. At that point, the situation changes for advertiser 2. Bidding \$10.01 to win slot 1 would lead to an expected profit of $0.1(\$11.00 - \$10.01) = \$0.099$, *while bidding \$6.00 to* *win slot 2 would lead to an expected profit of \$0.1. Thus, now a bid of \$6.00 is best, and the cycle repeats.*

This bidding war reduces allocative efficiency, with the top slot allocated to the advertiser with the second highest value half of the time. Although the effect on revenue could in principle be positive or negative, in practice it was estimated to be at least 7% relative to a more stable design. In addition, these bidding dynamics placed a lot of "churn" on the auction servers. Advertisers were making use of the robot bidders, and thus millions of advertisers were continuously sending messages to the auction servers to update their bids.

The design feature typical to current ad auctions, where the payment in the event of a click is set to exactly the minimal value the advertiser could bid and retain the same slot, removes the need for this bid dynamic.[4] Once the search engines switched to these "generalized second price" (GSP) auctions, there was significantly less churn on the auction servers. Changes in bids in order to minimize the payment for a given slot were no longer necessary.

The basic idea in the GSP auction, of charging a price that depends on that of the next highest bid, goes back to the birth of auction theory.

For expositional purposes, let's suppose we're selling a single space for an ad on the front page of a newspaper. In a first price (or pay-your-bid) auction, bids are collected and the space is sold to the highest bidder for the amount of his bid. In a second price auction, bids are collected and the space is sold to the highest bidder for the amount of the second-highest bid. Let's consider a simple example of a second-price auction. If bidders have values \$1,700, \$1,500 and \$1,000, then the space is sold to to bidder 1, but in a second-price auction for an amount of \$1,500. The optimal strategy for an advertiser is to bid his true value (or willingness to pay), irrespective of the other bids. Suppose that the bid amounts in the example reflect the bidders true values. Bidder 1 can do no better than bid \$1,700, and in particular, a smaller bid does not change the amount he pays (and could lead to losing the auction.) Bidder 2 can do no better than bid \$1,500, and in particular a bid that is greater than \$1,700 leads to winning the space for \$1,700, which is undesirable.

In fact, it is a simple exercise to formally establish that in a second-price auction for a single item, each bidder's optimal strategy is to bid his true value no matter what the bids from others. Bidding truthfully is a *dominant strategy*. This is in stark contrast to the strategic properties of a first-price auction, where each bidder should try to estimate the bids of others, and bid to maximize profit given these beliefs. This property of a second-price auction is known as *strategyproofness*.

But doesn't a second price auction reduce the auctioneer's revenue? For standard models of bidder valuations, the answer is "no." The simple reason is that bidders in a second price auction bid truthfully, whereas bidders in a first price auction shade down their bids. It turns out that the amount by which the winner shades down his bid in the equilibrium of a first-price auction is equal to the expected difference between the highest bid and the second highest bid in a second-price auction. The two auctions have the same expected revenue.

[4]The feature was first adopted by Google in 2002, and shortly thereafter by Yahoo!/Overture, and as of 2012 is used by all major search engines.

1.4. Economics and Computation: A Rich Tapestry

Ad auctions are one of the most prominent current examples of where economic and computational thinking come together. However, there is a long established, rich tapestry of interaction between economics and computer science. Beginning as early as the 1920's, pioneers such as John von Neumann contributed some of the most basic theory of economics including equilibrium concepts for simple situations of strategic interdependence, while at the same time providing the architectural basis for modern computers.

Out of these cross-currents have emerged many new ideas, both practical and theoretical. Disruptive e-commerce and social-computing technologies continue apace, including systems of *peer production* such as Wikipedia and Quora, *crowdsourcing* platforms such as oDesk and Amazon Mechanical Turk, *social networks* such as Facebook, *micro-blogging* systems such as Twitter, and *reputation and recommendation systems* such as those of eBay, Amazon and TripAdvisor. Computational finance is another area where computational and economic thinking collides, with markets that were originally designed for human traders populated with high frequency, algorithmic trading programs, leading to concerns in regard to emergent system failures such as flash crashes.

From a more theoretical perspective, computational thinking can inform economic thinking. For an example, consider the essential idea from economics of an equilibrium, wherein every participant is best-responding to the actions of everyone else. For a very simple example, consider the traffic routing problem, where an equilibrium constituted a particular flow of traffic such that no user has an incentive to change his route. But what about the computational difficulty in identifying an equilibrium? If it is too hard to identify an equilibrium, then perhaps we should be suspicious that equilibria can be descriptive of behavior. For this reason, it can be useful to apply a computational lens to concepts that are fundamental to economic thinking, such as equilibria. Indeed, some have observed that *"if your laptop can't compute it then neither can the market"*, expressing the concern that economic theories should not require market systems to solve intractable computational problems. Computational advances are also being made in how to represent (or describe) very large situations of strategic interdependence in the first place. This is a necessary precursor to being able to compute equilibrium outcomes.

We outline in the following sections some examples of computational systems with economic characteristics, economic systems with computational characteristics, and then a number of cross-cutting issues related to networks, models of user motivations, and broader considerations such as those of privacy and security.

1.4.1. Computational Systems with Economic Characteristics

The Internet itself. The Internet is owned and operated by commercial entities, and thus exhibits the characteristics of both an economic and a computational system. Distributed algorithms that are used for determining the way traffic passes through the fiber-optic networks of the Internet can be configured to prefer some routes over others according to business preferences. In particular, the so-called *autonomous systems* (AS'es) are the business entities that exchange traffic, with traffic flowing across interconnections between their networks before ultimately reaching a destination machine. In some cases, the AS'es are reasonably balanced in size and they form what are called *peering agreements* and do not charge each other for traffic flows. In other cases, AS'es reach agreements to make payments based on traffic flows (e.g., "you pay me when I transit traffic on your behalf.") Through the *Border Gateway Protocol* (BGP), information is exchanged between AS'es to advertise the different routes that can be taken over the Internet in order to reach a particular destination. Rather than provide a specific algorithm for determining a path, the actual routes determined through BGP depend on how individual routers are configured. For example, an AS might configure its routers to prefer a longer (higher latency) path if this avoids making a payment. Another AS might configure its routers to selectively advertise some paths and not others. This raises questions in regard to the effect that incentives and self-interest have on the robustness of the Internet.

Internet search. Algorithm design for the ranking of Internet search results is a problem that is both computational and economic in nature. The position of a website in the ranking of search results is extremely important in driving traffic to a site. For this reason, an entire "search engine optimization" industry is dedicated to influencing search results. This is paradigmatic of one of the new considerations that economic thinking brings to computer science. The design of an algorithm can influence the inputs to the algorithm. Traditionally, we think about the inputs to algorithms as coming from some known distribution, or being selected to provide hard instances. But when algorithms are used in economic domains, rational self-interested participants can become motivated to try to change the inputs in order to change the outputs in their own favor. For this reason, consideration to the impact that participants can have on outputs, by changing inputs, becomes of first-order importance.

Peer-to-peer systems. In a peer-to-peer (P2P) system, each computer both contributes and consumes resources. This enables scalable systems at scale without the deployment of massive amounts of centralized infrastructure. A well known example is the BitTorrent protocol for file sharing, which splits a large file into a bunch of smaller pieces, and allows machines to exchange pieces until the entire file can be put together. Voice-over-IP platforms such as Skype employ P2P technology: a user's machine routes calls for other users, while benefiting from the network for his or her own calls.

A fundamental problem in P2P systems is *free-riding*: a self-interested user would prefer to consume but not contribute resources. Whereas earlier file-sharing protocols such as Gnutella and Kazaa failed to successfully align incentives, the BitTorrent protocol incorporates basic reciprocation. Each peer divides its upload bandwidth resource into "slots" and allocates slots preferentially to peers that are providing high bandwidth in return. In this way, users are generally able to improve their own download speed by sharing some bandwidth for upload—and the system provides better incentive alignment.

Social computing systems. Social computing systems and the social web, such as social news aggregators (e.g., Digg, Reddit), collaborative content systems (e.g., Wikipedia), online communities and Q&A systems (e.g., Flickr, Quora, Yahoo Q&A, etc.), prediction markets (e.g., Hollywood Stock Exchange, intrade), and social networks (e.g., Facebook, Google+), weave together socio-economic incentives and algorithms for ranking, organizing and search, in order to promote useful user actions and contributions. For example, the social news aggregator Digg considers user votes along with the recency of an article in deciding how to rank articles. Yahoo! Q&A employs a point-based economy, where users earn points by answering questions and spend points by posting questions, with leader-boards used to incentivize contributions. Wikipedia allows user contributions to be acknowledged through virtual awards such as "barn-stars." Social networks are designed to align incentives for sharing information and content, for example with the tagging of photos with a person's name providing a mechanism by which photos are shared with friends.

1.4.2. Economic systems with Computational Characteristics

Ad Exchanges Ad exchanges are increasingly used for selling display ads adjacent to content on web pages. Publishers such as online news sites notify an exchange when a user loads a page. In realtime, the exchange makes a "call out" to multiple ad networks, each representing multiple advertisers. This call out can provide information in regard to the web site in addition to user behavioral and demographic information. Each network submits a bid and the highest bid wins and gains the right to place an ad on the page. On social networks, advertisers can also target information about a user's profile (e.g., the user is a friend of someone, or likes a particular entity, such as a brand or artist.) All aspects of these markets are typically automated and thus computational, including the processes for (i) determining whether to send an ad opportunity to an exchange, (ii) determining who is the winner and how much to charge, and (iii) determining how much to bid.

Wireless Spectrum Auctions. Wireless spectrum is increasingly congested, and sold by governments in high stakes *combinatorial auctions* (CAs). A combinatorial auction is used to sell multiple items where items can be both substitutes and complements to bidders. For example, a bidder might want to bid on a package of spectrum licenses in a contiguous region in order to put together a large and coherent network. A combinatorial auction would allow a bidder to submit a package bid, indicating that the bidder is only interested in winning *every* item. In this way, bidders avoid the risk of financial exposure, where they are left with only a part of a desired package. In addition to economic challenges, CAs are interesting computationally because the problem of determining winning bids is difficult to solve. The way to think about why this is a hard problem is that the auctioneer must find a good way to fit together bids on different packages of items. Another challenge with CAs is in designing bidding languages that are expressive but compact. It is not possible to submit a distinct bid for every package because the number of such packages grows exponentially in the number of items.

Prediction markets. Prediction markets are designed for the purpose of aggregating the information of multiple parties in order to form an accurate prediction about an uncertain future event. For example, the price of an asset that pays $1 if a category-5 hurricane will hit Florida in 2012 can provide a probabilistic forecast of this catastrophic event. If the price is

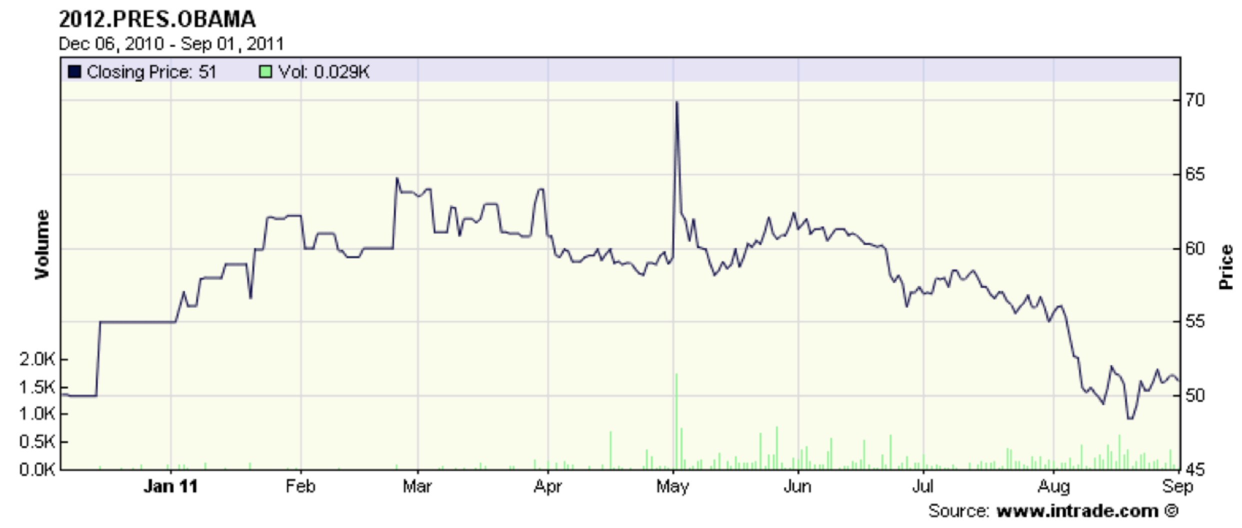


Figure 1.5.: Prediction Market for whether Obama will be re-elected in the 2012 US Election

$0.3, it implies a consensus belief that this is an event that will occur with probability 0.3. The Iowa Electronic Market forecasts the outcomes of political elections, and intrade.com predicts events ranging from the outbreak of avian flu to whether Barack Obama will be re-elected president. See Figure 1.5. Prediction markets are also used within organizations, for example to predict whether a target date for a software release will be met. There is a broad range of associated computational challenges: both in the design of languages to allow traders to express complex information ("if event A does not occur then I think event B will occur with probability 0.7"), and in algorithms to generate prices for trades of conditional contracts such as this.

Matching problems. Matching problems include those of matching students to schools, doctors to hospitals, students to dorm rooms, and patients to organ-transplant donors. Some of these problems have reasonably straightforward algorithmic properties. For example, *deferred-acceptance* algorithms match students to schools based on preference lists submitted by students and priorities provided by schools, and can be run quickly to determine a match. In particular, the number of algorithmic steps increases only quadratically in the number of students and schools. On the other hand, some of these matching problems are computationally intractable. In *kidney paired-donation*, the participants are patient-donor pairs, where the patient needs a kidney and the donor a friend or relative, willing to donate a kidney but incompatible with the patient. A feasible match is a set of disjoint cycles on pairs, such that each pair is involved in at most one cycle. For example, a two-cycle could be $A \to B \to A$. A three-cycle could be $A \to B \to C \to A$; the way to read this is that the donor of pair A donates to the patient of pair B, the donor of pair B to the patient of pair C, and the donor of pair C to the patient of pair A. Whereas the problem of finding disjoint two cycles that include a maximum number of pairs is tractable, the problem of finding disjoint two or three cycles that include a maximum number of pairs is intractable.

Recommender and Reputation systems. Successful e-commerce models such as Amazon and Netflix rely heavily on computational approaches to make recommendations to users. To enable consumers to find the products they want, it is essential to provide them with access to the so-called "long tail" of the product distribution since users often have very idiosyncratic interests, in addition to being interested in popular items. One kind of recommender systems uses *collaborative filtering*, where the feedback of other users is used to predict which items a user might like. Intuitively, if in the past, user A liked items that user B liked and disliked items that user B disliked, then user B's ratings are likely a good predictor for how much user A will like a new item. In addition to providing high value recommendations to users, recommender systems should be robust to manipulation through such behaviors as submitting false feedback to unfairly boost a hurt a particular product.

Whereas recommender systems are designed to aggregate largely subjective information (what one agent likes need not be liked by another agent), reputation systems are designed to aggregate objective information (e.g., whether or not a seller shipped a good). Reputation systems collect reports from users in regard to their experience, and automate traditional "word-of-mouth" information propagation. On one hand, reputation systems are computational because algorithms are used to aggregate information. On the other hand, reputation systems are vital to the well-functioning of Internet markets and the design of useful feedback systems is quite a subtle problem. For example, an early reputation system on eBay suffered from a problem where a buyer might be reluctant to give negative feedback because of a threat of retaliation from the seller, with the seller threatening to retaliate by leaving negative feedback about the buyer.

1.4.3. Cross-cutting: Networks, Motivations and Systemic Issues

The intersection of economics and computation also includes cross-cutting issues, for example in regard to networks and network structure and dynamics, in regard to models of user motivations, and in regard to concerns of privacy, security and the design and robustness of virtual currency systems.

Networks. Networks are central to many computational and economic systems. Consider information networks such as the world wide web, communication networks such as the interconnected AS'es of the Internet, social networks and friend graphs, and trading networks representing trading relationships in markets. Some areas where EconCS thinking is especially relevant include:

(i) *Network dynamics*: How do networks grow, and can economic models of behavior explain the emergent structure of networks and also provide guidance as to how to design incentives in order to promote particular kinds of network structures?

(ii) *Transactions*: How does network structure affect the bargaining power of different participants, measured as the amount of value created in a transaction that goes to each participant?

(iii) *Information dynamics and influence*: Which participants are the most effective in influencing the beliefs of other users; e.g., in influencing other users to make a purchase decision for the purpose of viral marketing? Given a network that represents positive feedback between pairs of users, then how robust are different methods to compute aggregate scores to strategic behavior through false reports or other methods?

(iv) *Platform-mediated networks*: Some platforms directly mediate the formation and use of networks, by controlling how they grow, how costs or benefits are distributed to new participants, and by making some kinds of actions easy while precluding other kinds of actions. Fundamental challenges that affect many platform-mediated networks include those of *mobilization* (i.e., how to provide value to an initial user base), *network externalities* (e.g., how to promote actions that provide value to other users), and *platform control* (e.g., how to evolve a network.)

User Motivations. Traditional economic models generally assume the *selfish-rational actor model*, with participants where only care about themselves and take actions to maximize their own happiness. This provides a precise and simple model, and is useful in many e-commerce domains. On the other hand, it does not seem to explain very well contributions made by users to Wikipedia or Yahoo! Answers. For this reason, it is useful to consider additional models of user motivations, including those of *other-regarding preferences* such as inequity aversion, reciprocity, and altruism. Understanding these kinds of motivations leads to new design patterns, such as providing mechanisms for social recognition that can come from design elements such as the barnstars in Wikipedia. Another consideration is that of bounded rationality—neither people nor firms are the idealized, rational (optimizing) economic agent of economic theory. Here too, alternative models of user behavior can drive new design considerations. For example, if users tend to over-weight present welfare relative to future welfare then what is the implication of this when seeking to promote investment in anti-virus and other security technology?

Systemic Issues. There are three additional cross-cutting issues worthy of comment:

(i) *Security of Computer and E-commerce Systems*: On one hand, such concerns as *phishing*, *spoofing*, and *spamming* have a significant impact on the effectiveness of e-commerce and social network systems. Moreover, the security of computational systems such as personal computers can be viewed through an economic lens. For example, the costs of a denial-of-service attack on a web site are felt by the owners of the site rather than the users whose machines have been compromised and are used to launch the attack.

(ii) *Decentralized currency systems:* Money is valuable because it addresses the inefficiencies that arise from the "double coincidence of wants" that affects barter systems ("I want your bread but you don't want my salt.") There is interest in developing decentralized currency systems, so that there is no center that can control, monitor, and tax transactions. The main challenges are to prevent the arbitrary "printing" of new money and the "double spending" of the same money. One approach is through *credit networks*, that allow anyone to create new currency but limit currency exchange to paths along which there is mutual trust, and systems based on cryptography that make it costly to create new currency and prevent double spending through a replicated (anonymized) ledger of transactions.

(iii) *Privacy:* On one hand, data about individuals allows for personalization and advertising, and broader sharing of data (e.g., in regard to medical treatment and health outcomes could provide substantial societal benefit.) On the other hand, the continued aggregation of data about individuals leads to concerns in regard to privacy and possible unintended, and unanticipated consequences. For this reason, there is interest in developing computational responses to protecting privacy, including methods to aggregate or add noise to data in order to reduce the possibility that information about any one individual is revealed. Can economic thinking even be used to provide a way forward, with markets developed in which users can sell personal information in return for personalized services, payments, content and so forth.

1.5. Notes

In "Computational challenges in e-commerce," *Communications of the ACM* **52**, 2009, pages 70-74, Feigenbaum, Parkes and Pennock provide a related discussion that is specific to e-commerce. Al Roth has written about the economist as an engineer, the comment on "your laptop computing" is due to Kamal Jain, and Tuomas Sandholm, Noam Nisan and Joan Feigenbaum have all provided good scholarship related to the general discussions in this chapter. "Networks, Crowds and Markets" by Easley and Kleinberg (Cambridge University Press 2010) provides a broad treatment of similar topics, but with more emphasis given to social science and network science. "Algorithmic Game Theory", edited by Nisan et al. (Cambridge University Press 2007) provides an advanced treatment of a number of specialized topics related to EconCS.