Computational Complexity in Economics and Bounded Rationality

Daniel Brous, Tenghao Li, Ruqing Xu[*]

December 15, 2023

1 Introduction

In classical economics, agents are modeled as rational, Bayesian agents who take actions to maximize their expected utility. This assumption has faced substantial criticism over the years, leading to the emergence of alternative models of "bounded rationality," where the decision maker deviates from full rationality in some way. One prominent approach is to apply computational complexity constraints from theoretical computer science to human decision makers – if no computer on earth can solve the problem until the end of time, maybe it is reasonable to assume that no human can do so as well. Of course, it can be a very loose limitation on humans' computational ability, since most humans still struggle with polynomial time problems like adding two four-digits numbers. For this very reason, it becomes even more fascinating when computational complexity accounts for certain human behaviors observed in real life.

This survey aims to provide a detailed overview of various strands of literature that relate computational complexity to economics. Specifically, we will demonstrate the hardness results of computing Nash equilibria and reaching market efficiency. We will review prior research that employs complexity constraints to explain deviations from classical models of Bayesian updating and utility maximization. Finally, we will delve into the paper by Camara (2022), as it is a noteworthy example of how complexity theory can be effectively applied to economics, yielding sharp results.

2 Overview of the Literature

2.1 Hardness of Computing a Solution Concept in Economics

The first strand of literature that relates computational complexity to economics shows that many analytic solution concepts used in economics are in fact computationally hard, so we may not expect that computationally constrained agents can reach them efficiently in practice.

[*]All authors contributed equally and are listed in alphabetical order.

2.1.1 Hardness of Nash Equilibrium

For example, a series of works (Daskalakis & Papadimitriou 2005, Goldberg & Papadimitriou 2006, Chen et al. 2006, Chen & Deng 2006, Daskalakis et al. 2009) culminate in a famous result: computing mixed Nash Equilibria for general non-zero-sum games is PPAD-complete. The concept of NP-hardness does not directly apply to Nash Equilibria, because NP-complete problems, like SAT, derive their complexity from the possibility that a solution may not exist. In contrast, John Nash demonstrates that all games have at least one mixed Nash Equilibrium (i.e. a Nash equilibrium where the choices are probabilitiy distributions over a finite set of "atomic" choices), guaranteeing the existence of a solution (Nash 1951). Daskalakis et al. (2009) prove that, instead, Nash is PPAD-complete. The class PPAD contains several challenging problems like End-Of-The-Line and Brouwer, where a solution is guaranteed to exist but is difficult to find. This result suggests, from a computational perspective, that mixed Nash equilibrium may be unnatural as a prediction of the behavior by a group of agents in general games.

2.1.2 Hardness of Efficient Markets

Another paper along a similar vein is Maymin (2011). Maymin shows a close tie between the Efficient Market Hypothesis (ETH) in finance and the $\mathsf{P} \neq \mathsf{NP}$ conjecture in computer science. He argues that markets are efficient, implying that current prices reflect all available information in past prices, if and only if $\mathsf{P} = \mathsf{NP}$.

The general idea is that we want to show that given a history of past prices, finding a strategy that is consistently profitable (more profitable than random guessing) is hard. This implies that profitable strategies exploiting market anomalies (patterns in past prices) cannot be efficiently found by everyone and thus eliminated through competitive behavior. Therefore, markets are not efficient since there exist positive returns to those who happen to find a pattern, at least until those patterns become widely known.

We will go into the details of the proof as it is both innovative and instructive. Define a *strategy* S as a function that takes in a length-t sequence of price movements UPs and DOWNs (encoded as 1's and 0's) and outputs a position, either $+1$ for long, -1 for short, or 0 for neutral.

$$S : \{0,1\}^t \to \{-1, 0, +1\}$$

In the proof, it is without loss to consider only the strategies that never output -1. Note that for a particular strategy, the lookback window t is fixed, meaning the strategy can only depend on the most recent t price histories. This ensures the generalizability of the strategies.

Now, the question is: for a given lookback window t and a critical value of profit K (that distinguishes the profit from random chance) does there exist a strategy S that generates a profit more than K, such that the sum of the prices of the purchased assets does not exceed a budget constraint B?

This formulation of the problem happens to be an instance of the Knapsack problem as formulated in Garey & Johnson (1979): Given a finite set U and for each $u \in U$, given two positive integers $s(u)$ and $v(u)$ which are referred to as the "size" and "value" of u, and given two positive integers B and K, is there a subset $U' \subseteq U$ such that $\sum_{u \in U'} s(u) \leq B$ and such that $\sum_{u \in U'} v(u) \geq K$?

If we think of u as length-t price histories, $s(u)$ as the price for a given asset following a particular price history, $v(u)$ as the future return for a given asset following a particular price history, then picking u to put in U' is equivalent to picking the set of length-t price histories which would result in a "long" position (so that the complement of $U - U'$ are price histories that result in a "neutral" position). In that sense, picking u is the same as picking a strategy. We know that KNAPSACK is NP-complete. Therefore, the question of whether or not there exists a budget-conscious strategy that generates statistically significant profit is also NP-complete. In other words, investors would be able to efficiently compute such strategies if and only if $\mathsf{P} = \mathsf{NP}$.

To strengthen the only if direction, the author also shows that we can "program" the market to solve NP-complete problems, in particular, 3SAT. The general idea is to write a 3SAT formula as orders that can be placed in the market. If a formula is satisfiable, then an efficient market should be able to complete the order in polynomial time.

The translation is as follows. Each variable in 3SAT represents an asset. Bare variables imply buy orders and negated variables imply sell orders. Notably, these orders need to be order-cancels-orders (OCO), meaning that as soon as the market fulfills one part of the order, the other parts are cancelled automatically.

For a given 3SAT formula, for example,

$$(a \vee b \vee \neg c) \wedge (a \vee \neg b \vee d),$$

we can translate the formula as two simultaneously placed orders:

1. Order-cancels-order buy A, buy B, or sell C
2. Order-cancels-order buy A, sell B, or buy D

If the 3SAT formula is satisfiable, i.e., there exists some way to execute all of those separate OCO orders such that an overall profit is guaranteed, then the market, by its assumed efficiency, ought to find a way to do so. If that is the case, then the market allows us to compute in polynomial time the solution to an arbitrary 3SAT problem.

This paper connects perhaps the most famous problem in finance, whether the market is efficient, to perhaps the most famous problem in computer science, whether $\mathsf{P} = \mathsf{NP}$. Comparing these two problems reveals an interesting paradox, as the paper writes:

> The majority of financial academics believe in market efficiency and the majority of computer scientists believe that $\mathsf{P} \neq \mathsf{NP}$. The result of this paper is that they cannot both be right: either $\mathsf{P} = \mathsf{NP}$ and the markets are efficient, or $\mathsf{P} \neq \mathsf{NP}$ and the markets are not efficient.

2.2 Complexity as an Explanation for Deviations from Rationality

An alternative strand of the literature goes beyond theoretically identifying computationally hard problems, but shows that computational constraints could explain some observed behaviors in real life that deviate from the predictions of classical economic models.

2.2.1 Finding Regularities from Existing Knowledge

Aragones et al. (2005) observe the fact that individuals, with guidance, often realize new regularities from their existing knowledge without acquiring additional facts. This contradicts the model of Bayesian learning, in which agents learn only when new information is acquired. As the paper wittily notes:

> This phenomenon is so pervasive that it has been canonized in literature: Sherlock Holmes regularly explains how the combination of a variety of clues leads inexorably to a particular conclusion, following which Watson exclaims, “Of course!”

The authors argue that this phenomenon can be explained by the complexity of searching for regularities. Formally, they relate the problem of finding regularities in one’s knowledge base as an econometrician’s problem of determining whether there exists a small set of regressors that can obtain a certain value of R^2, a measure of how well the chosen regressors explains the outcome variable in linear regressions.

They prove that, given explanatory variables $(X_1, \ldots, X_m)$ and an outcome variable Y, for any $r \in (0, 1]$ and any integer $k \geq 1$, determining whether there exists a subset of explanatory variables that has size $\leq k$ and achieves an $R^2 \geq r$ is NP-complete.

This means that for practical purposes, it is infeasible to determine the optimal set of regressors for moderate to large datasets. Translating that to practice, it may explain why people may often find that they have overlooked a simple linear regularity that, once pointed out, seems evident.

2.2.2 Cooperation in Finitely Repeated Prisoner’s Dilemma

Consider the game of finitely repeated prisoners’ dilemma, where a classical prisoner’s dilemma is played finitely many times (say N) between the same two players. In such a game, a strategy of the player is a function which maps a history of play to a subsequent action.

The classical game theory proves that the only rationalizable strategy is to defect in every turn, which can be shown by backward induction. Consider the last stage game ($t = N$). At this stage, the players’ action has no implication for future payoffs since the game ends after this turn. Therefore, both players would play the dominant strategy in the stage game, i.e., defect. Knowing this, however, the players should also play defect in the second-to-last stage game, since they would play defect in the last stage no matter what. Repeating this argument shows that rational players should play defect in every turn.

However, experimental observations often show that players do cooperate, often over long periods of time, as oppose to always choosing the single-period dominant action. Neyman (1985, 1998) aims to reconcile this discrepancy from the lens of complexity constraints.

Neyman uses finite automata as the computation model. An automaton is defined as having a finite number of states, an initial state, an action function that prescribes the action taken in each state, and a transition function that determines the next state based on the current state and the actions of other players. The size of the automaton is the number of states. In particular, Neyman measures the complexity of a strategy by the size of the smallest automaton capable of implementing it. The paper argues that if players are restricted to using finite automata whose size is polynomial in the total number of stages N, there exists an equilibrium that yields a payoff close to the cooperative payoff.

This paper is among the first to provide a mathematically rigorous definition of complexity in the context of games. This formulation proved fruitful, in the sense that restricting the complexity of strategies that players can use justifies the observed cooperative behavior in such games. It is probably a stretch to claim that finite automata are accurate descriptions of the cognitive process of humans. Nonetheless, it highlights the importance of complexity considerations in predicting human behaviors.

2.2.3 The Complexity of Decision-making

Modelling decision-making is a cornerstone of economics. One of the most well-known models of decision-making is the one of utility maximization. It posits that the choices of an agent are rational if and only if they maximize some real-valued function (Debreu et al. 1954). If the choices are over probabilistic outcomes, the expected utility axioms require that the choices maximize the expected utility of some function (Von Neumann & Morgenstern 1944).

When considering the utility-maximization problem, economists generally disregard the complexity of making such decisions. There has been a series of efforts to bring this consideration into economic modeling. For example, finite automata have been used to study games (Aumann 1981, Rubinstein 1986, Neyman 1985, 1998), procedural choice (Salant 2011), and Bayesian reasoning (Wilson 2014). Other researchers have used Turing machines to study the *computability* of choices (Richter & Wong 1999). However, computability is much weaker than computational tractability, since it only requires that a behavior can be generated by a Turing machine, without regard to time constraints.

Camara (2022) is the first to study the *tractability* of *choices under uncertainty* using a *Turing machine*. It stands out from previous literature in a few aspects. It requires computational tractability rather than computability of choices, which connects closer to most of complexity theory developed in computer science. It considers choices under uncertainty, which is more involved than the classic model of budget-constrained consumer choice. Finally, it uses Turing machines, the most general computational model, to describe choice behavior. Besides methodological innovations, it is able to obtain strong characterization results that make sharp predictions about economic behavior. It explains behavioral heuristics, such as choice bracketing, as responses to computational constraints. It also brings into question the applicability of classical rationality axioms in the presence of such constraints. For these reasons, we believe it is valuable to dive deep into this paper in particular, as it serves as an epitome of marrying complexity theory and economic modeling, and so the rest of this report is an overview of the main results from this paper.

3 Computationally Tractable Choice: Introduction and Preliminaries

3.1 Introduction

The first three theorems of Camara (2022) show that forms of choice under certain definitions of "tractability" are equivalent to forms of a behavioral heuristic called "choice bracketing," under the assumption that certain widely believed computation complexity conjectures are true. At a loose level, an agent exhibits "choice bracketing" if they make choices over a set of items in a manner in which they do not consider all possible choices of each item. As an informal example, consider a worker choosing $X_i \in \{\text{Low}, \text{High}\}$ effort on days $i \in [n]$, and suppose wages can vary on a daily basis. Putting in high effort is more costly to the worker. One way to make a choice here is to always guarantee the optimal sequence of efforts by considering all 2^n choices and trade off the total wages with the life-time disutility of working. Another way the worker might make a choice is by considering each day independently and trading off wage and effort on those days, and then using the effort levels they found for each day to determine their final sequence of efforts. This only requires considering n choices, and so is clearly much easier in practice, although this cannot guarantee an optimal effort choice since the worker disregards intertemporal substitution of wages. The last theorem of Camara (2022) demonstrates that this intuition is correct, and in practice we cannot have a way of making tractable choices that guarantee optimal payoffs in all cases.

3.2 Preliminaries

To be general, we will define the sequences of items that agents choose to be lotteries, i.e., probability distributions, over a set of outcomes called $\mathcal{X}$. We view $x \in \mathcal{X}$ as an "effectively n-dimensional" vector, i.e.,

$$\exists n \quad \text{such that} \quad x = (x_1, \dots, x_n, 0, \dots) \in \mathbb{Q}^\infty.$$

A **lottery** $X : [0,1] \to \mathcal{X}$ is viewed as a probability distribution over outcomes where X gives us an outcome based on a uniform random variable $\omega \in [0,1]$. If we take the highest index of a non-zero component to be n, then we can view X as an n-tuple of **partial lotteries** X_i, i.e.,

$$X = (X_1, \dots, X_n).$$

These partial lotteries are simply probability distributions over $\mathbb{Q}$. A **menu** M is a set of lotteries that an agent can choose from, and a partial menu is a set of partial lotteries. We define partial lotteries so that we can have the notion of **product menus**, which are just the product set of partial menus. We will call the set of all menus that the agent might be presented with as $\mathcal{M}$. For this paper, we assume that $\mathcal{M}$ includes all product menus over binary partial menus and that it also includes all **binary menus**, i.e., menus consisting of two lotteries.

We will model the concept of choice with a correspondence c which maps a menu $M \in \mathcal{M}$ to a lottery $X \in M$. We will say that a choice correspondence c is **rational** if it can be generated by preferences that satisfy the expected utility axioms (Von Neumann & Morgenstern 1944), which are a set of axioms that capture the idea of "rational" preferences and are used throughout economics to define rationality. Von Neumann & Morgenstern (1944) prove that a choice correspondence c satisfies those axioms if and only if there exists a utility function $u : \mathcal{X} \to \mathbb{R}$ such that

$$c(M) = \arg\max_{X \in M} E[u(X)], \quad \forall M \in \mathcal{M},$$

which is the definition we will use throughout this paper. Now, we will define tractable choices in the following way:

Definition 1. *Choice correspondence c is* ***tractable*** *if there exists a Turing machine that*

- *takes in the description of menu M and outputs choice $c(M)$*
- *and halts in $\mathsf{poly}(n)$ steps for all $M \in \mathcal{M}$.*

We will also need a notion of symmetry for utility functions and choice correspondences.

Definition 2. *A utility function u is* ***symmetric*** *if for any permutation $\sigma : [n] \to [n]$,*

$$u(x_1, \ldots, x_n) = u(x_{\sigma(1)}, \ldots, x_{\sigma(n)})$$

for all $(x_1, \ldots, x_n) \in \mathcal{X}$.

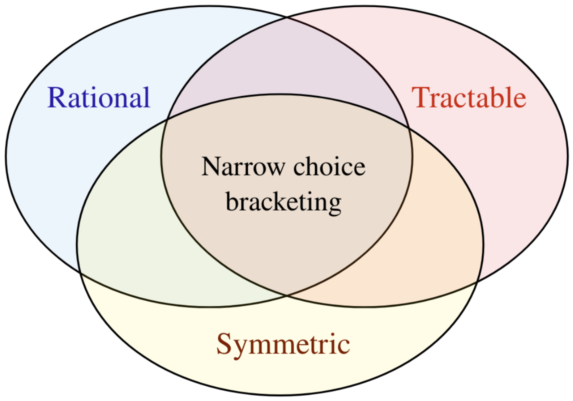
Definition 3. *A choice correspondence c is* ***symmetric*** *if $c(M) = c(M')$ for any menus M, M' where M' is a permutation menu of M. We define a* ***permutation menu*** *M' of M to be a menu with all the same lotteries except each lottery has its first n partial lotteries permuted by the same permutation $\sigma : [n] \to [n]$.*

Lastly, we define a notion that is essentially equivalent to a certain type of choice bracketing.

Definition 4. *A utility function u is* ***additively separable*** *if there exist functions $u_1, \ldots, u_n$ such that*

$$u(x_1, \ldots, x_n) = u_1(x_1) + \cdots + u_n(x_n).$$

for all $(x_1, \ldots, x_n) \in \mathcal{X}$.



The form of choice bracketing this is equivalent to is called narrow choice bracketing, which means that for all $i \in [n]$, the i-th component of an agent's choice only depends on the i-th partial menu. The intuition for this is that if making partial choices is done one partial menu at a time, then this can be modeled with an additively separable utility function in which a maximizer can choose an outcome by choosing the maximal outcome for each component independently.

Lastly, we require the concept of **efficient computability**, but it is not important so we will omit a formal definition. Informally, a utility function is efficiently computable if there exists a polynomial-time machine that computes the function with at most ϵ imprecision. This is essentially a regularity condition – it is much weaker than tractability of the choice correspondence, and unrelated to separability. Intuitively, being able to calculate utility for a given outcome is very different from being able to choose the best lottery amongst a large set of lotteries.

4 Theorem 1

Theorem 1. *Let choice correspondence c be rational, tractable, and symmetric. If $\mathrm{P} \neq \mathrm{NP}$ then c reveals an additively separable, symmetric, and efficiently computable utility function.*

We will outline the proof of this theorem. Camara reduces the proof to the following four lemmas:

Lemma 1. *Let u be a symmetric utility function. Then u is additively separable iff there do not exist constants $a, b \in \mathbb{Q}$ and an outcome $x \in \mathcal{X}$ such that*

$$u(a, a, x_3, x_4, \ldots) + u(b, b, x_3, x_4, \ldots) \neq u(a, b, x_3, x_4, \ldots) + u(b, a, x_3, x_4, \ldots)$$

Lemma 2. *Suppose a tractable choice correspondance c maximizes expected utility, where there exist constants $a, b \in \mathbb{Q}$ and an outcome $x \in \mathcal{X}$ such that*

$$u(a, a, x_3, x_4, \ldots) + u(b, b, x_3, x_4, \ldots) > u(a, b, x_3, x_4, \ldots) + u(b, a, x_3, x_4, \ldots)$$

Then there exists a polynomial-time algorithm for MAX $2-SAT$.

Lemma 3. *Suppose a tractable choice correspondence c maximizes expected utility, where there exist constants $a, b \in \mathbb{Q}$ and an outcome $x \in \mathcal{X}$ such that*

$$u(a, a, x_3, x_4, \ldots) + u(b, b, x_3, x_4, \ldots) < (a, b, x_3, x_4, \ldots) + u(b, a, x_3, x_4, \ldots)$$

Then there exists a polynomial-time algorithm for MIN $2{-}SAT$.

Lemma 4. *A choice correspondence that is rational and strongly tractable reveals an efficiently computable utility function.*

Theorem 1 follows directly from these Lemmas. Suppose we have some choice correspondence c which is rational, tractable, symmetric. Since c is rational, by definition, c maximizes expected utility for some utility function u. Since c is symmetric, u must also be symmetric. Now, assuming $\mathsf{P} \neq \mathsf{NP}$, since SAT has polynomial time reductions to $MAX\ 2{-}SAT$ and $MIN\ 2{-}SAT$, there can't be polynomial time algorithms for $MAX\ 2{-}SAT$ or $MIN\ 2{-}SAT$. Using all these facts, Lemmas 2 and 3 imply that there do not exist constants $a, b \in \mathbb{Q}$ and an outcome $x \in \mathcal{X}$ such that

$$u(a, a, x_3, x_4, \ldots) + u(b, b, x_3, x_4, \ldots) \neq u(a, b, x_3, x_4, \ldots) + u(b, a, x_3, x_4, \ldots)$$

Thus, Lemma 1 implies that u is additively separable. Finally, since u is strongly tractable, Lemma 4 impies that u is efficiently computable.

4.1 Proof Overview of Lemma 2

Now we will prove Lemma 2. The idea is to construct an algorithm for $MAX\ 2{-}SAT$ by translating a 2-*CNF* formula to a product menu over binary partial menus, choosing a lottery that maximizes expected utility, and translating this into an assignment that satisfies the maximum number of clauses in the formula. Since we assume c to be tractable, this algorithm runs in polynomial time.

$$\begin{tikzcd}
\text{Formula} \arrow[r, "\overset{\text{(poly-time)}}{f}"] & \text{Menu} \arrow[r, "c"] & \text{Choice} \arrow[r, "\overset{\text{(poly-time)}}{g}"] & \text{Assignment} \\
BF & M & X & v_1, \ldots, v_n \\
\text{Two literals per clause} & M_i = (X_i^T, X_i^F) & \text{Maximizes } E[u(X)] & \text{Solves MAX (MIN) 2-SAT}
\end{tikzcd}$$

The menu that the algorithm constructs is just the product over all binary partial menus $M_i = (X_i^T, X_i^F)$, where X_i^T and X_i^F correspond to assignments of true and false to variable v_i, respectively, and so the menu has a one to one mapping to the set of assignments to the variables of the formula. Based on this idea, define g so that if $X_i = X_i^T$, then it assigns variable $v_i =$ True, and if $X_i = X_i^F$, then it assigns variable $v_i = False$. To demonstrate the proof, we will show a simpler case and then explain how it generalizes to the general case.

4.1.1 Proof of the Special Case

For the special case, suppose that the corresponding utility function is

$$u(x_1, \ldots, x_n) = \max\{x_1, \ldots, x_n\}.$$

In this case, we *won't* have to use the fact that

$$u(a, a, x_3, x_4, \ldots) + u(b, b, x_3, x_4, \ldots) \neq u(a, b, x_3, x_4, \ldots) + u(b, a, x_3, x_4, \ldots)$$

for some $a, b \in \mathbb{Q}$ and $x \in \mathcal{X}$. Suppose we have some boolean CNF formula over n variables, $v_1, \ldots, v_n$, where there are m clauses of two literals each, and we will denote the jth clause by $v_{j_1} \vee v_{j_2}$. Define X_i^T and X_i^F so that if $\omega \in [(j-1)/m, j/m)$, then

$$X_i^T(\omega) := \begin{cases} 1 & v_{j_1} = v_i \\ 1 & v_{j_2} = v_i \\ 0 & \text{otherwise} \end{cases} \qquad X_i^F(\omega) := \begin{cases} 1 & v_{j_1} = \neg v_i \\ 1 & v_{j_2} = \neg v_i \\ 0 & \text{otherwise} \end{cases},$$

In words, based on a randomly picked subinterval of $[0,1]$, if the clause corresponding to this subinterval is true when $v_i =$ True, then $X_i^T = 1$, and if it is true when $v_i =$ False, then $X_i^F = 1$, and otherwise both lotteries are 0. Now, given any choice of lottery $X = (X_1, \ldots, X_n) \in \prod_{i=1}^n \{X_i^T, X_i^F\}$, one can see that

$$\begin{aligned} E[u(X_1, \ldots, X_n)] &= E[\max\{X_1, \ldots, X_n\}] \\ &= \frac{1}{m}\sum_{i=1}^m E[\max\{X_1, \ldots, X_n\} \mid \omega \in [(j-1)/m, j/m)] \\ &= \frac{1}{m}\sum_{i=1}^m E[\mathbf{1}(v_{j_1} \vee v_{j_2} \mid g(X_1 \ldots, X_n)) \mid \omega \in [(j-1)/m, j/m)] \\ &= \frac{1}{m}\sum_{i=1}^m \mathbf{1}(v_{j_1} \vee v_{j_2} \mid g(X_1 \ldots, X_n)) \end{aligned}$$

Therefore, maximizing expected utility is equivalent to maximizing the number of clauses satisfied. The equality from the second to the third line is because of the following: Given some choice of lotteries $(X_1, \ldots, X_n) \in \prod_{i=1}^n \{X_i^T, X_i^F\}$ and that $\omega \in [(j-1)/m, j/m)$,

$$\max\{X_1, \ldots, X_n\} = 1$$
$$\Longleftrightarrow$$
$$X_i(\omega) = 1 \text{ for some } i \in [n]$$
$$\Longleftrightarrow$$
$$\text{for some } i \in [n],\ ((v_{j_1} = v_i \text{ or } v_{j_2} = v_i) \text{ and } X_i = X_i^T) \bigvee ((v_{j_1} = \neg v_i \text{ or } v_{j_2} = \neg v_i) \text{ and } X_i = X_i^F)$$

$$\Longleftrightarrow$$

$$v_{j_1} = \text{True or } v_{j_2} = \text{True given the assignment } g(X_1, \ldots, X_n)$$

$$\Longleftrightarrow$$

$$\mathbf{1}(v_{j_1} \vee v_{j_2} \mid g(X_1 \ldots, X_n)) = 1$$

4.1.2 Extension to the General Case

The reason that this argument cannot apply to the general case is that the step where we claim that

$$E[u(X_1, \ldots, X_n) \mid \omega \in [(j-1)/m, j/m)]] = E[\mathbf{1}(v_{j_1} \vee v_{j_2} \mid g(X_1 \ldots, X_n)) \mid \omega \in [(j-1)/m, j/m)]]$$

is true when utility is the max function but not true in general. We only need these two values to be proportional, not necessarily equal, so we just need that in general, there exists some c only dependent on the utility function u such that

$$E[u(X_1, \ldots, X_n) \mid \omega \in [(j-1)/m, j/m)]] = cE[\mathbf{1}(v_{j_1} \vee v_{j_2} \mid g(X_1 \ldots, X_n)) \mid \omega \in [(j-1)/m, j/m)]]. \tag{1}$$

But even in a simple case where $u(X_1, \ldots, X_n) = (X_1 + \cdots + X_n)^2$, this fails. If, for some j, the j-th clause is $v_1 \vee v_2$, then if $\omega \in [(j-1)/m, j/m)$ and we choose $X_1 = X_1^T$ and and $X_2 = X_2^T$, then

$$E[u(X_1, \ldots, X_n) \mid \omega \in [(j-1)/m, j/m)]] = (1+1)^2 = 4,$$

whereas if we chose $X_1 = X_1^T$ and $X_2 = X_2^F$, then

$$E[u(X_1, \ldots, X_n) \mid \omega \in [(j-1)/m, j/m)]] = (1+0)^2 = 1.$$

This is bad because

$$E[\mathbf{1}(v_{j_1} \vee v_{j_2} \mid g(X_1 \ldots, X_n)) \mid \omega \in [(j-1)/m, j/m)]] = 1$$

in both of these cases.

The way Camara (2022) generalizes this argument is to do two things:

1. Subdivide the interval $[0, 1]$ into more subintervals.
2. Add more cases to each X_i^T and X_i^F so that in all cases, we get that equation 1 holds for some c.

He defines the lotteries in terms of a parameter α, and he uses the fact that

$$u(a, a, x_3, x_4, \ldots) + u(b, b, x_3, x_4, \ldots) \neq u(a, b, x_3, x_4, \ldots) + u(b, a, x_3, x_4, \ldots)$$

for some $a, b \in \mathbb{Q}$ and $x \in \mathcal{X}$ to prove that there exists a certain choice of α that makes equation 1 hold for some c. Since the casework is long and arduous, we will refer the reader to the original paper for details.

Theorem 1 shows that, while assuming symmetry, rational and tractable choices reveal an additively separable utility function, which means that the decision maker must exhibit narrow choice bracketing. This is a little unsatisfactory in the sense that symmetric choice correspondences may be palatable in some contexts (e.g. when different dimensions represent income earned from different assets) but unnatural in others (e.g. when different dimensions represent different goods). Theorem 2 and 3 will generalize the results in Theorem 1 by dropping the assumption of symmetry. The resulting utility functions will be Hadwiger separable, a concept we will introduce in the next section.

5 Hadwiger Separability

Hadwiger separability is a relaxation of additive separability. It captures a sense in which most pairs (x_i, x_j) are evaluated separately and independently from each other, but not necessarily all.

Definition (Pairwise separability)**.** *A utility function u is (i, j, n)-separable if there exist functions u_i, u_j such that for all n-dimensional outcomes x,*

$$u(x) = u_i(x_i, x_{-ij}) + u_j(x_j, x_{-ij})$$

where x_{-ij} denotes the all the other coordinates except for x_i, x_j.

Example: A decisionmaker is choice bracketing. She partitions dimensions n into mutually exclusive brackets B_j. For each bracket B_j, she maximizes expected utility according to a utility function u_j which is defined over the coordinates $i \in B_j$. Then, the utility function u is (i, j, n)-separable iff i, j are not in the same bracket. More concretely, if a possible consumer's revealed utility function for $n = 5$ is

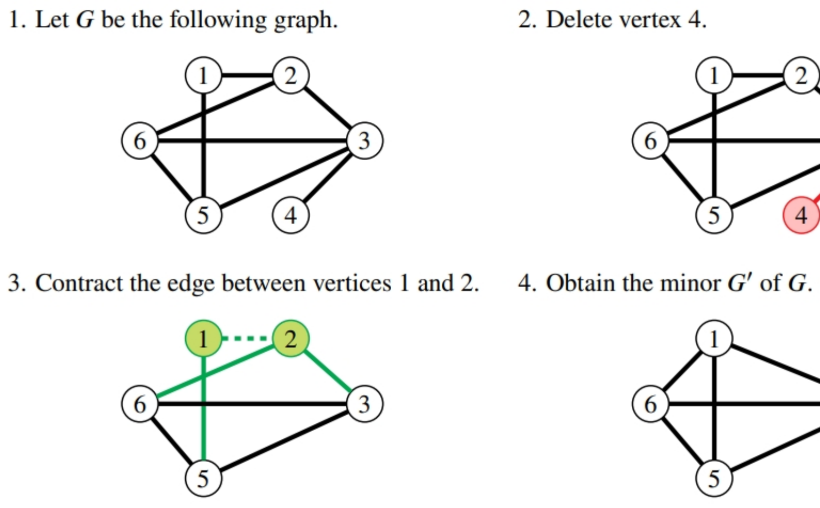
$$u(x) = u_1(x_1, x_2) + u_2(x_3, x_4, x_5)$$

then u is $(1, 3, n = 5)$-separable, but not $(1, 2, n = 5)$-separable. Intuitively, each bracket represents natural complements (e.g. cereal and milk) or substitutes (e.g. apples and oranges), and the consumer ignores any complementarity or substitutability across the brackets.

Definition (Inseparability Graph)**.** *For all n-dimensional outcomes x, the inseparability graph $G_n(u)$ of a utility function u is an undirected graph with n vertices.*

We connect the (in)separability of a utility function to the sparsity of the inseparability graph. Notice that there is an edge between nodes i and j if and only if the pair (x_i, x_j) is not separable. If the inseparability graph $G_n(u)$ is sparse, then almost all pairs (x_i, x_j) are separable and the utility function u is "almost" separable. The measure of graph sparsity was formulated by Hadwiger (1943) to state his longstanding conjecture about the chromatic number of graphs.

Definition (Minor Graph). *G' is a minor if it can be formed from G by some sequence of the following two operations:*
1. Delete a vertex v and all of its incident edges.
2. Contract an edge (i, j) This deletes nodes i, j and replaces them with a new vertex k.


1. Let *G* be the following graph.
2. Delete vertex 4.
3. Contract the edge between vertices 1 and 2.
4. Obtain the minor *G′* of *G*.
1
2
3
4
5
6

Definition (Hadwiger Number). *The Hadwiger number $Had(G)$ of an undirected graph G is the number of nodes in its largest complete minor.*

In the above example, the Hadwiger number $Had(G) = 4$, because G' is the largest complete minor of G and G' has 4 vertices.

Definition (Hadwiger Separability). *The function u is Hadwiger separable if*

$$Had(G_n(u)) = O(\log n)$$

Compared to additive separability, it is capable of modeling a richer set of preferences (e.g. limited number of complementarities and substitutions). Eppstein (2009) showed that computing the Hadwiger number of any arbitrary graph is NP-hard. However, based on the result of Alon (2007), for any inseparability graph $G_n(u)$ and constant C, it is possible to determine whether $Had(G_n(u)) \leq C \log n$ in $O(\text{poly}(n, C))$ time.

Proposition 1. *A symmetric utility function is Hadwiger separable iff it is additively separable.*

Proof. When the function u is symmetric, the inseparability graph $G_n(u)$ is either empty or complete. If $G_n(u)$ is empty, then u is additively separable. If $G_n(u)$ is complete, then $Had(G_n(u)) = n$ and therefore u is not Hadwiger separable. It follows that Hadwiger separability is equivalent to additive separability when u is symmetric. $\square$

6 Representation Theorem

Assumption 1 (Non-uniform Exponential Time Hypothesis (NU-ETH))**.** *There is no family of algorithms (one for each length of the input, in the spirit of advice) that can solve 3-SAT in time* $2^{O(n)}$.

Theorem 2. *Assume NU-ETH holds. If choice correspondence c is rational and weakly tractable. then c reveals a Hadwiger separable utility function, which is efficiently computable with advice.*

This is stronger than Theorem 1, and in fact it implies Theorem 1 as a corollary. Suppose that the choice correspondence c is symmetric, as well as rational and weakly tractable. By Theorem 2, c is Hadwiger separable. Since c is also symmetric, Proposition 1 implies that c is additively separable. Theorem 3 is a partial converse of Theorem 2.

Theorem 3. *Assume NU-ETH holds. Let the utility function u be Hadwiger separable and efficiently computable with advice. Then expected utility maximization is weakly tractable on a smaller set of menus (i.e., product menus).*

6.1 Proof Outline of Theorem 2

Lemma 6.1. *Suppose a weakly tractable choice correspondence c maximizes expected utility, where*

$$d_n := Had(G_n(u))$$

Then there exists an $O(poly(n))$-time algorithm to solve MAX 2-SAT for any boolean formula with at most d_n variables. The algorithm uses at most $O(poly(n))$-size advice.

The advice in Lemma 6.1 provides two kinds of information. First, it describes the largest complete minor of the inseparability graph $G_n(u)$. Second, it identifies the coordinates pair (x_i, x_j) where the utility function u is not (i, j, n)-separable.

The proof of Lemma 6.1 has 2 main steps: In step 1, we construct an auxilliary formula BF' with n variables using polynomial-size advice. This will be an instance of a weighted MAX 2-SAT problem. A solution to this auxilliary problem is corresponding to a solution to the original problem. In step 2, we will reduce the weighted MAX 2-SAT problem to expected utility maximization using polynomial-size advice. This will be similar to the proof of lemmas for Theorem 1. It follows that the solving MAX 2-SAT for the original formula is weakly tractable if the expected utility maximization is weakly tractable. Here is a direct corollary of Lemma 6.1.

Corollary 1. *Suppose a weakly tractable choice correspondence c maximizes expected utility, where*

$$Had(G_n(u)) = \omega(\log n)$$

Then the there exists a $O(2^{O(n)})$-time algorithm for 3-SAT with n variables with at most $O(poly(n))$-size advice.

Since we assume NU-ETH holds, $Had(G_n(u)) = O(\log n)$ if c is a weakly choice correspondence. Then by definition, u is Hadwiger separable. This completes the proof of Theorem 2

6.2 Proof Outline of Theorem 3

We directly construct an algorithm that maximizes the expected utility in polynomial time with advice (see algorithm 1 on the next page). The algorithm takes an input of a Hadwiger separable utility function u. The advice for the algorithm includes the inseparability graph $G = G_n(u)$ and any advice needed to efficiently compute the utility function u. It is easy to verify that the advice has poly(n) size.

The algorithm employs dynamic programming techniques on a graph. As more nodes/coordinates are visited, the algorithm redefines the optimal choice $X_i^*(\cdot)$ so that it remains a function of unvisited coordinates. Eventually, the algorithms visit all coordinates, and the functions $X_i^*(\cdot)$ have no remaining arguments.

Assume there are k alternative partial lotteries $X_i \in M_i$. In step 9-10, the set $M_{S_i \cup I_i}$ has up to $k^{2|I_i|}$ elements. Since u is Hadwiger separable, the number of indirect influencers of any node i is $|I_i| = O(\log n)$. Therefore, step $9 - 10$ take $O(k^{2|I_i|+1} \cdot \text{poly}(n)) = O(\text{poly}(n))$ time. Since all the other steps in the algorithm also takes $O(\text{poly}(n))$ time, algorithm 1 is a polynomial time algorithm with advice of polynomial size.

7 Conclusion and Discussion

This survey demonstrates how computational complexity theory can inform our understanding of economic decision-making. It explores the limitations of classical economic models in addressing the complexity of tasks like computing Nash equilibria and achieving market efficiency. By integrating computational constraints, this survey offers insights into why certain economic behaviors, traditionally viewed as irrational, are actually rational when considering human cognitive limitations.

How well does computational complexity as defined in computer science pertain to human reasoning? Indeed, there are problems such as natural language understanding or face recognition that toddlers perform better than do computers. But these are problems for which finding an appropriate mathematical model is a major part of the solution. By contrast, for well-defined combinatorial problems such as those in the class NP, it is rarely the case that humans perform better than do computers. So long as many problems we face in economic domains are of this nature, we believe that studying the implication of computational constraints still provide a useful upper bound on human behavior.

Algorithm 1: Dynamic choice bracketing algorithm that maximizes expected utility

- 1. Convert the undirected graph G into a directed acyclic graph G' by assigning a direction to each edge.
- 2. Define a frontier set F which contains unvisited nodes that are successors to some visited node.
- 3. Set $F = \emptyset$ initially.
- 4. Let i be the smallest unvisited node in G'.
- 5. The successors S_i are unvisited nodes j where G contains an edge between i and j.
- 6. The predecessors P_i are visited nodes j where the optimal choice $X_j^*(\cdot)$ depends on X_i.
- 7. The indirect influencers I_i are frontier nodes $j \in F$ where G contains a path between i and j that does not pass through F.
- 8. Let the value function equal expected utility under the assumption that $X_j = 0$ for all coordinates $j \notin \{i\} \cup S_i \cup P_i$. More formally, define

$$V_i(X_i, X_{S_i}, X_{P_i}) = \mathbb{E}[u(X_i, X_{S_i}, X_{P_i}, 0, 0, ...)]$$

- 9. Define the optimal choice $X_i^*(\cdot)$ as a function of successors and indirect influencers.

$$X_i^*(X_{S_i}, X_{I_i}) = \arg\max_{X_i \in M_i} V_i(X_i, X_{S_i}, X_{P_i}^*(X_i, X_{I_i}))$$

- 10. Redefine the optimal choices $X_j^*(\cdot)$ for predecessors $j \in P_i$ by replacing X_i with $X_i^*(\cdot)$.

$$X_j^*(X_{S_i}, X_{I_i}) = X_j^*(X_i^*(X_{S_i}, X_{I_i}), X_{I_i})$$

- 11. Add S_i to frontier set F and remove i from F. Repeat step 4-11 until there is no unvisited nodes in G'.

The algorithm outputs $(X_1^*, ..., X_n^*)$.

References

Alon, N., L. A. . W. M. (2007), 'Approximating the maximum clique minor and some subgraph homeomorphism problems', *Theoretical Computer Science, 374(1), 149–158* .

Aragones, E., Gilboa, I., Postlewaite, A. & Schmeidler, D. (2005), 'Fact-free learning', *American Economic Review* **95**(5), 1355–1368.

Aumann, R. J. (1981), 'Survey of repeated games', *Essays in game theory and mathematical economics in honor of Oskar Morgenstern* .

Camara, M. K. (2022), Computationally tractable choice., *in* 'EC', p. 28.

Chen, X. & Deng, X. (2006), Settling the complexity of two-player nash equilibrium., *in* 'FOCS', Vol. 6, pp. 261–272.

Chen, X., Deng, X. & Teng, S.-H. (2006), Computing nash equilibria: Approximation and smoothed complexity, *in* '2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)', IEEE, pp. 603–612.

Daskalakis, C., Goldberg, P. W. & Papadimitriou, C. H. (2009), 'The complexity of computing a nash equilibrium', *Communications of the ACM* **52**(2), 89–97.

Daskalakis, C. & Papadimitriou, C. H. (2005), Three-player games are hard, *in* 'Electronic colloquium on computational complexity', Vol. 139, Citeseer, pp. 81–87.

Debreu, G. et al. (1954), 'Representation of a preference ordering by a numerical function', *Decision processes* **3**, 159–165.

Eppstein, D. (2009), 'Finding large clique minors is hard', *J. Graph Algorithms Appl. 13(2), 197–204* .

Garey, M. R. & Johnson, D. S. (1979), 'Computers and intractability', *A Guide to the* .

Goldberg, P. W. & Papadimitriou, C. H. (2006), Reducibility among equilibrium problems, *in* 'Proceedings of the thirty-eighth annual ACM symposium on Theory of computing', pp. 61–70.

Maymin, P. Z. (2011), 'Markets are efficient if and only if p= np', *Algorithmic Finance* **1**(1), 1–11.

Nash, J. (1951), 'Non-cooperative games', *Annals of mathematics* pp. 286–295.

Neyman, A. (1985), 'Bounded complexity justifies cooperation in the finitely repeated prisoners' dilemma', *Economics letters* **19**(3), 227–229.

Neyman, A. (1998), 'Finitely repeated games with finite automata', *Mathematics of Operations Research* **23**(3), 513–552.

Richter, M. K. & Wong, K.-C. (1999), 'Computable preference and utility', *Journal of Mathematical Economics* **32**(3), 339–354.

Rubinstein, A. (1986), 'Finite automata play the repeated prisoner's dilemma', *Journal of economic theory* **39**(1), 83–96.

Salant, Y. (2011), 'Procedural analysis of choice rules with applications to bounded rationality', *American Economic Review* **101**(2), 724–748.

Von Neumann, J. & Morgenstern, O. (1944), ‘Theory of games and economic behavior princeton’, *Princeton University Press* **1947**, 1953.

Wilson, A. (2014), ‘Bounded memory and biases in information processing’, *Econometrica* **82**(6), 2257–2294.