Hadwiger Separability, or: Turing meets von Neumann and Morgenstern

Modibo K. Camara[*]

June 22, 2021

Abstract

This paper incorporates time constraints into decision theory, via computational complexity theory. I use the resulting framework to better understand common behavioral heuristics, known as choice bracketing, where agents separate some choices from others. My main result shows that a time-constrained agent who satisfies the expected utility axioms must have a *Hadwiger separable* utility function. This separability condition is a relaxation of additive separability that allows for some complementarities and substitutions but limits their frequency. One implication of this result is that a time-constrained agent may be better off violating the expected utility axioms. This can occur when the agent wants to maximize the expected value of a utility function that is not Hadwiger separable.

[*]Department of Economics, Northwestern University. Email: mcamara@u.northwestern.edu.

Contents

1 Introduction

It can be time-intensive to make optimal choices, especially when making many related choices at once. For example, consider a customer purchasing a subset of hundreds of different products, a firm managing dozens of branches and hundreds of employees, or an investor trading in dozens of assets. It is widely believed that combinatorial optimization problems like these are hard, for fundamental reasons that go beyond our own cognitive limitations (e.g. Karp 1972).

To ensure that decisions are made in a reasonable amount of time, people often follow heuristics like choice bracketing that isolate some of their choices from others (e.g. Tversky and Kahneman 1981, Read et al. 1999, Rabin and Weizsäcker 2009). This tendency to separate choices can reduce one high-dimensional optimization problem into several low-dimensional subproblems that are presumably easier to solve. And this tendency has important implications for economic behavior. How these decision-makers separate their choices affects the appropriate boundaries for economic models, how we model their behavior in a given model, what we can learn from behavioral data, and how much data we need to learn it.

This paper incorporates time constraints into decision theory, and uses the resulting framework to better understand how and why people separate choices. I combine classic approaches in decision theory and computational complexity theory to obtain a model of a time-constrained agent. I show that a time-constrained agent who satisfies the expected utility axioms must have a Hadwiger separable utility function. This condition relaxes additive separability by allowing for some complementarities and substitutions but limiting their frequency. However, I also show that a time-constrained agent may be better off violating the expected utility axioms, even if she intrinsically wants to maximize expected utility. This can occur when the agent's true objective function is not Hadwiger separable.

Choice, Time, and Dimensionality. There are three ingredients to my model: choice under risk, time constraints, and high-dimensionality. I begin with a standard model of choice under risk. The agent chooses between random variables called lotteries. The set of lotteries available to the agent is called a menu. The agent's choice from a given menu is described by a choice correspondence. A choice correspondence is called *rationalizable* if it satisfies the expected utility axioms (von Neumann and Morgenstern 1944).

For any given menu, there is a bound on how much time the agent has to express her choice, and a choice correspondence that can be executed by an algorithm that respects this time constraint is called *computationally tractable*. I allow the agent more time when facing a menu that is more complicated. However, the amount of time taken may grow at most polynomially in the amount of space it takes to describe the menu. This notion of polynomial-time tractability is used in computational complexity theory to identify intractable computational problems that are presumed to be too difficult for computers to solve within any reasonable amount of time.

In order to study separability, I specialize the model to focus on high-dimensional choice problems. This involves four assumptions. First, the consequences – i.e. the outcomes that the agent cares about – are arbitrarily high-dimensional vectors. Second, the consequences must include all finite-dimensional vectors in the unit cube. Third, the agent must be able to make a choice from any menu with three or fewer options. Fourth, she must be able to make a choice from any finite-dimensional *product menu*, where her choice in some dimension i does not affect what is feasible in another dimension j. These assumptions make the agent's choice problem high-dimensional, but do not require her to make choices from menus with complicated constraints.

Representation Theorem (warmup). I begin by motivating additive separability in a special case where the choice correspondence is symmetric. Symmetry captures the idea that different dimensions of the agent's choice are interchangeable. For example, if each dimension of the consequence represents a source of income, this would suggest that the agent does not care whether she earns income from source i or from source j.

My first theorem shows that rational, tractable, and symmetric choice implies expected utility maximization with an additively separable utility function. The theorem's proof follows a reduction argument, and relies on a computational hardness conjecture known as $P \neq NP$. Suppose for contradiction that an agent has a utility function that is not additively separable but can nonetheless make an optimal choice from any given menu within polynomial time. I show that this agent could also solve MAX2SAT in polynomial time, which violates $P \neq NP$. The key challenge in proving the result is that it must apply to *every* utility function in a large class, as opposed to *some* utility function in that class. In that sense, it is similar to Schaefer's (1978) dichotomy theorem.

Hadwiger Separability. To generalize the representation theorem beyond symmetric choice correspondences, I need to relax additive separability to Hadwiger separability.

As stated, Hadwiger separability is a relaxation of additive separability that allows for some complementarities and substitutions but limits their frequency. The definition is graph-theoretical. Given a function with multiple arguments, I define an *inseparability graph* that indicates whether the function is additively inseparable across any pair of arguments. Functions are considered more separable if their inseparability graph is more sparse, acccording to a sparsity measure called the Hadwiger number. If the Hadwiger number of a function's inseparability graph is small relative to its number of arguments, then the function is Hadwiger separable. This criterion reduces to additive separability if the function is summetric.

Representation Theorem (continued). My second theorem shows that rational and tractable choice implies expected utility maximization with a Hadwiger separable utility function. This relies on a strengthening of $P \neq NP$, called the non-uniform exponential time hypothesis (NUETH). The proof generalizes the reduction technique used in the proof of theorem 1.

My third theorem argues that this result is tight by proving a partial converse: maximizing expected utility in product menus is tractable when the utility function is Hadwiger separable. The caveat is that I refer to a slightly weaker notion of non-uniform tractability. The proof is constructive and the proposed algorithm – *dynamic choice bracketing* – may be interesting in its own right. This heuristic algorithm leverages principles of dynamic programming to generalize choice brack-

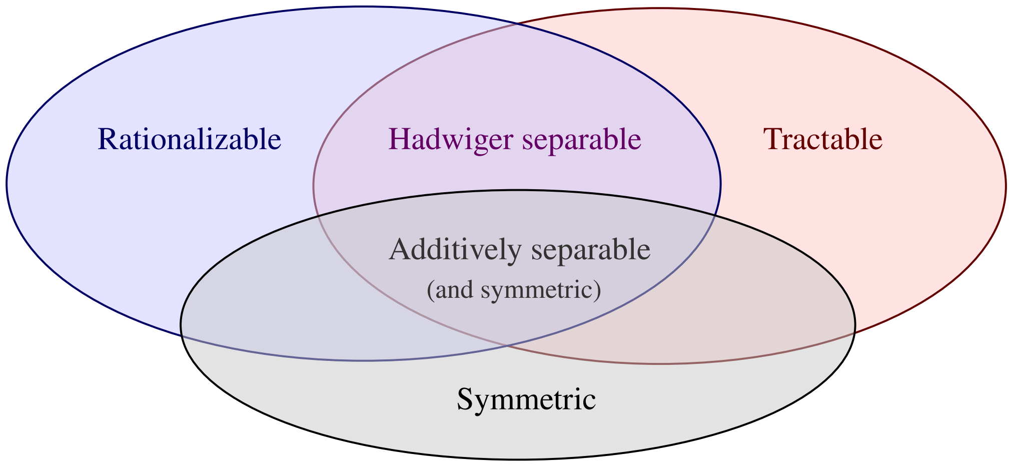


Figure 1: The blue oval depicts the set of rationalizable behaviors. The red oval depicts the set of tractable behaviors. I characterize their intersection, the purple region, as corresponding to behavior that is rationalized by an Hadwiger separable utility function. The grey oval depicts the set of symmetric behaviors. I characterize the intersection of all three ovals as corresponding to behavior that is rationalized by a symmetric, additively separable utility function.

eting. When the agent's utility function is Hadwiger separable, I can construct a dynamic choice bracketing algorithm that maximizes expected utility in polynomial time.

Existence of a Rationality Gap. I use these results to obtain a *rationality gap* theorem: a time-constrained agent that intrinsically wants to maximize expected utility may be better off violating the expected utility axioms. The rationality gap quantifies "better off".

To understand the rationality gap and why it may exist, consider a time-constrained agent whose true objective $\bar{u}$ is not Hadwiger separable. Her first-best choice correspondence maximizes her expected objective, but theorem 3 shows that it is not tractable. Intuitively, a second-best choice correspondence should approximately maximize her expected objective, but still be tractable. This need not be rationalizable. Imposing rationalizability as an additional constraint gives us a third-best choice correspondence, which should approximately maximize her expected objective, be tractable, and be rationalizable. The rationality gap measures the gap between the second- and third-best, in line with how computer scientists evaluate approximation algorithms.

Interestingly, it turns out that the rationality gap can be quite large. To establish this, I identify a natural class of objective functions where no rationalizable approximation algorithm guarantees any positive fraction of the agent's optimal expected utility. On the other hand, a greedy algorithm will guarantee at least half of the agent's optimal expected utility. Given a rationalizable approximation algorithm, it is easy to modify the greedy algorithm so that it weakly dominates the original algorithm, and still improves the approximation ratio from zero to $1/2$. The rationality gap is the ratio of these approximation ratios; in this case, it is unbounded.

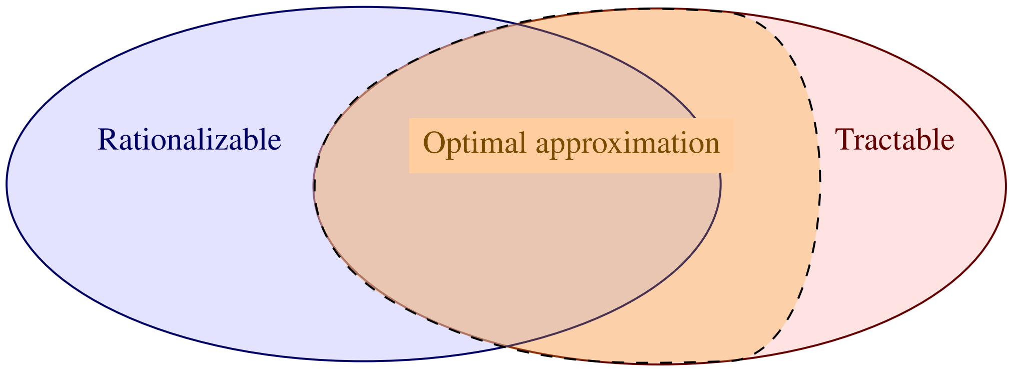


Figure 2: The blue oval depicts the set of first-best behaviors, for some objective function. The orange region depicts the set of second-best behaviors, or optimal approximations for some objective function. The intersection of the blue and red ovals depicts the set of third-best behaviors. The rationality gap establishes that there are behaviors, depicted by red horizontal lines, that are optimal approximations but not rationalizable. Moreover, for the relevant objective functions, rationalizable behaviors give terrible approximations.

Organization. Subsection 1.1 surveys the related literature. Section 2 introduces the model of choice under risk, time constraints, and high-dimensional choice. In section 3, I show that rationality, tractability, and permutation-invariance imply additive separability. Section 4 defines Hadwiger separability. In section 5, I show that rationality and tractability imply Hadwiger separability. Section 6 defines the rationality gap, and shows that it can be large. Omitted proofs can be found in appendix A.

1.1 Related Literature

This work builds on a long-standing research effort on bounded rationality, and highlights dimensionality as a source of complexity. Methodologically, it is closely related to work in economics and computation that applies computational complexity theory to gain insight into economic phenomena. Finally, it relates to a large literature in behavioral and empirical economics on choice bracketing and related phenomena. In this section, I briefly discuss some related literature.

In decision theory, computational complexity theory has previously been employed in Echenique et al. (2011) and Apesteguia and Ballester (2010). Of these papers, Echenique et al. (2011) is more similar to mine. The authors consider a classical model of consumer choice, and ask whether a dataset consisting of observed choice behavior could be generated by an algorithm whose runtime is polynomial in the number of products and the size of the dataset. They find that computational tractability has no bite, since any rationalizable choice correspondence is tractable. In contrast, I look at a different setting – high-dimensional choice under risk – and show that an axiom of computational tractability can significantly restrict the agent's behavior. These results are entirely compatible, as we take a similar conceptual approach to study different kinds of choice problems.[1]

[1]One possibly minor distinction is that, in my paper, I interpret the choice correspondence as reflecting potential

In addition, Apesteguia and Ballester (2010) consider the problem of an analyst trying to rationalize a choice correspondence, in a model of choice under certainty that allows violations of WARP. They find that the analyst's problem is computationally hard in general.

In contrast, the literature on bounded rationality often uses less permissive models of computation (e.g. Rubinstein 1986, Mandler 2015, Mandler et al. 2012, Jakobsen 2020, Al-Najjar et al. 2003). Of course, polynomial-time tractability is a rather weak definition of tractability. Most humans would struggle to solve any number of computational problems that are technically polynomial-time solvable. In that sense, the literature on bounded rationality may seek more restrictive notions in order to get more interesting results. In my case, even this weak notion of polynomial-time tractability gives quite restrictive results.

Outside of decision theory, polynomial-time tractability has been used to critique various concepts and results in economics (e.g. Fortnow and Vohra 2009, Hązła et al. forthcoming). The premise of this and other work is that problems that are intractable for the computer are also intractable for other physical, biological, and social processes. For example, a notable line of work studies the time complexity of finding Nash equilibria (see e.g. Daskalakis et al. 2009). Hardness results suggest that Nash equilibrium is unlikely to arise in complicated games, regardless of which introspective or learning process agents may employ. This approach is appealing precisely because intractability is a property of the problem itself, not a property of the solution. On the other hand, Barman et al. (2014) take an alternative approach. They characterize a set of observed behaviors that are consistent with Nash equilibrium play in games where finding a Nash equilibrium is tractable. My work bears some resemblance to this, insofar as I characterize the set of behaviors that are consistent with the computational tractability of expected utility maximization.

Finally, my work relates to the literature in behavioral economics on choice bracketing. Read et al. (1999) coined the term choice bracketing as a way to explain behavior observed in experiments. I refer to their definition:

> [Choice bracketing] designates the grouping of individual choices together into sets. A set of choices are bracketed together when they are made by taking into account the effect of each choice on all other choices in the set, but not on choices outside of the set. When the sets are small, containing one or very few choices, we say that bracketing is narrow, while when the sets are large, we say that it is broad. Broad bracketing allows people to consider all the hedonic consequences of their actions, and hence promotes utility maximization. Narrow bracketing, on the other hand, is like fighting a war one battle at a time with no overall guiding strategy, and it can have similar consequences.

Of course, choice bracketing is closely related to additive separability.

There is considerable experimental evidence that choice bracketing is a feature of human decision-making (Andersen et al. 2018; Andreoni et al. 2018; Chaudhry et al. 2020; Martin 2017; Rabin and Weizsäcker 2009; Read et al. 1999; Tversky and Kahneman 1981). Some evidence and interpretations explicitly point towards bounded rationality, where bracketing is a response to the complexity

outcomes: how the agent would behave if presented with a particular menu. Echenique et al. (2011) appear to deal with a dataset of observed choices.

of combinatorial optimization (Brown et al. 2017; Koch and Nafziger 2019; Pennings et al. 2008; Simonsohn and Gino 2013; Stracke et al. 2017). Others tend towards framing effects, where the type of bracketing depends on the framing of the problem (Brown et al. 2017; Haisley et al. 2008; Moher and Koehler 2010). Of course, a tendency to frame problems narrowly may itself be a response to complexity. On the theoretical side, Zhang (2021) provides an axiomatization of narrow choice bracketing and variations of that phenomenon. This is a more traditional decision-theoretic analysis that does not explicitly consider the role of computational complexity.

2 Preliminaries

In this section, I (a) introduce a standard model of choice under risk, (b) formalize time constraints and tractability, and then (c) specialize the model to focus on high-dimensional choice problems.

2.1 Choice under Risk

An agent chooses from a *menu* of available *lotteries*. A *choice correspondence* describes her choices from any given menu. Roughly-speaking, a choice correspondence is *rational* if it always chooses lotteries that maximize expected utility for some utility function.

To be more precise, let Z be a compact set of *consequences*. Let $(\Omega, \mathcal{F}, P)$ be a probability space where the sample space $\Omega = [0, 1]$ is the unit interval, $\mathcal{F}$ is the Borel σ-algebra, and P is the Lebesgue measure. A lottery is a random variable $l : \Omega \to Z$ with finite support. A menu L is a set of lotteries available to the agent. Let $\mathcal{L}$ be a collection of menus.

The agent's behavior is described by a choice correspondence c. If the agent were presented with menu $L \in \mathcal{L}$, then $c(L)$ would describe her choices $l \in L$ from that menu.

Definition 1. *A* choice correspondence $c : \mathcal{L} \twoheadrightarrow Z^\Omega$ *maps menus L to a set of lotteries $\phi(L)$. It must be nonempty, i.e. $c(L) \neq \emptyset$, and respect feasibility, i.e. $\phi(L) \subseteq L$.*[2]

I restrict attention to choice correspondences that satisfy a monotonicity property, reflecting an assumption that higher consequences are better for the agent. This assumption does not appear to be necessary and I hope to remove in the next iteration of the paper.

Assumption 1. *The choice correspondence is* strictly monotone. *Consider a menu consisting of degenerate lotteries δ_z and $\delta_{z'}$, which put full probability on consequences z and z', respectively. If $z > z'$ then the agent will choose lottery δ_z, i.e. $c(\{\delta_z, \delta_{z'}\}) = \{\delta_z\}$.*

Researchers often restrict attention to choice correspondences that are consistent with rational preferences. A *preference* $\succeq$ is a binary relation on the set of all lotteries. An agent with preferences $\succeq$ makes the following choices:

$$c_{\succeq}(L) = \{l \in L \mid l \succeq l', \forall l' \in L\}$$

[2]When an agent makes multiple choices, i.e. $|\phi(L)| > 1$, this means she might choose any single one of the lotteries $l \in \phi(L)$. Typically, this reflects indifference between lotteries $l, l' \in \phi(L)$.

A preference is rational if it is consistent with the expected utility axioms proposed by von Neumann and Morgenstern (1944). I restate them now, for the reader's convenience.

Definition 2. *A preference $\succeq$, or a choice correspondence $c_{\succeq}$, is* rational *if it satisfies the following axioms. Throughout, let l, l', l'' be generic lotteries.*

1. Completeness: *either $l \succeq l'$ or $l' \succeq l$.*
2. Transitivity: *if $l'' \succeq l'$ and $l' \succeq l$ then $l'' \succeq l$.*
3. Continuity: *if $l'' \succeq l' \succeq l$, then there exists a constant $\lambda \in (0, 1)$ such that*
$$\lambda l'' + (1 - \lambda) l \sim l'$$
4. Independence: *for any constant $\lambda \in [0, 1]$, we have*
$$l' \succeq l \iff \lambda l' + (1 - \lambda) l'' \succeq \lambda l + (1 - \lambda) l''$$

For a finite sample space Ω, von Neumann and Morgenstern (1944) showed that rational choice correspondences are observationally equivalent to expected utility maximization for some continuous utility function $u : Z \to \mathbb{R}$. Theorems 1 and 2 will include a similar statement. It follows that a choice correspondence that fails to exactly maximize expected utility, on even a single menu, may violate the expected utility axioms.[3]

2.2 Time Constraints

From a computational perspective, a choice correspondence c describes a *computational problem*.[4] A menu is a particular *instance* of that problem. Informally, a choice correspondence c is *computationally tractable* if there exists an algorithm that computes the agent's choice $c(L)$ from any given menu L, while satisfying a (polynomial) *time constraint*.

The motivation for this approach is simple. First, a mortal agent has a limited amount of time to make her choices. Second, given a description of some menu L, there must be some mechanical process that will return the lottery $l \in c(L)$ that the agent would have chosen. Presumably, this process is part of the agent's nervous system, which induces her to make a particular choice when faced with a particular menu. Going forward, I assume that this process can either be modeled as a *Turing machine*, or as a *boolean circuit*. These models are closely related, and will correspond to strong and weak notions of tractability, respectively. In this paper, I mostly focus on weak tractability, but it is pedagogically useful to introduce strong tractability first.

[3] Many readers will object that the expected utility axioms are too strong. However, it is important to understand the implications of a definition in order to properly critique it. For example, the rationality gap theorem (section 6) is a novel critique of the expected utility axioms, and it would not be possible without theorem 2, which articulates their implications. In any case, I will discuss alternative definitions of rationality in the conclusion (section **??**).

[4] In general, a choice correspondence may or may not be an optimization problem. Of course, any rational choice correspondence is an optimization problem, since it corresponds to expected utility maximization.

2.2.1 Turing Machines and Strong Tractability

A Turing machine is an abstract model of computation that takes in a string of characters and outputs another string. It substantially generalizes finite automata, which have found application in the literature on repeated games (Rubinstein 1986). For a formal definition of the Turing machine, the reader may refer to any textbook on computational complexity (Arora and Barak 2009, ch.1). Informally, the reader should think of a Turing machine as an *algorithm*.[5]

I model the agent as a Turing machine M whose choice correspondence c_M is the output of M. The time it takes for the agent to make a choice $c(L)$ from menu L is the number of steps taken by M before it arrives at its output. That number of steps is called the *runtime* of M. Formally, I let $r_M(L)$ denote the runtime of M on menu L.

It is natural that an agent should take more time to make a decision on menus that have more lotteries or are otherwise more complicated. For this reason, time constraints restrict how quickly the runtime increases as the menu becomes more complicated. Formally, a menu L is described by a string of length $k(L)$. A time constraint f is a function $f : \mathbb{N} \to \mathbb{R}_+$ that maps the description length $k(L)$ to the maximum allowable runtime. The Turing machine M satisfies the *(strong, or uniform)* time constraint $f : \mathbb{N} \to \mathbb{R}_+$ if

$$r_M(L) \leq f(k(L)) \quad \forall L \in \mathcal{L} \tag{1}$$

The next definition calibrates the time constraint based on the Cobham-Edmonds thesis, which asserts a computation is tractable only if its runtime is at most polynomial in the length k of the input string. This captures the sense that the agent should make her decisions within a reasonable amount of time, for reasonably complicated menus.[6]

Definition 3. *The choice correspondence c_M is* strongly tractable ($c \in \mathrm{P}$)[7] *if M satisfies the time constraint* (1) *for some function $f(k)$ that grows at most polynomially in k.*

Tractability (weak or strong) should be thought of as an axiom, like the expected utility axioms. This is because it is a restriction on the choice correspondence: it constrains how an agent's choices vary as the menu changes. It is not a constraint in the sense of a budget or other constraints on the menu itself. In particular, there is no clear sense in which an agent can maximize expected utility in a given menu "subject to" the time constraint.[8]

[5] I do not explicitly define the Turing machine because the definition is unwieldy, and also unnecessary. The reader should feel free to think of the Turing machine as code written in their favorite all-purpose programming language, like C or Java. These languages are Turing-complete, which means that they can simulate any Turing machine.

[6] In other words, any algorithm whose runtime is superpolynomial (e.g. exponential) in k will take an unreasonable amount of time for moderately large k – regardless of how powerful the computer running it is. Clearly, the converse is not true. A computation that requires k^{100} steps has time complexity $O(\mathrm{poly}(k))$ but is not feasible in practice. Furthermore, even $O(k)$ problems, like adding two numbers, can be challenging for human beings if k is large. In that sense, both definitions of tractability are quite restrained: they only rule out the very hardest problems.

[7] This is a slight abuse of notation. The class P usually refers to *decision problems*, i.e. computational problems with a binary output. However, this should not cause any confusion and does not affect any of my results.

[8] Let me make the point more concretely. Consider a menu L and lottery $l \in L$ that maximizes expected utility

2.2.2 Boolean Circuits and Weak Tractability

To motivate weak tractability, I introduce boolean circuits. Boolean circuits are alternative models of computation that are closely related to but more powerful than the Turing machines. A boolean circuit is a directed graph, where each vertex corresponds to a simple logical operation (AND, OR, and NOT). It receives an input string written in binary, which is transformed into an output string as it passes through the graph. Generally, we restrict attention to circuits where the number of vertices is at most polynomial in the length k of the input string.

The key difference between a boolean circuits and Turing machines is that boolean circuits are adapted to menus L of a certain complexity, as measured by their description length $k(L)$. In that sense, an agent that is modeled as a boolean circuit has an advantage over an agent that is modeled as a Turing machine. To be more precise, the class of problems that can be solved by a polynomial-size boolean cirtcuit is equivalent to the class of problems that can be solved by a polynomial-time Turing machine using polynomial-size *advice* (Arora and Barak 2009, ch.6)

This motivates the following definition of *weak tractability*. Consider a Turing machine M that takes an advice string A as additional input. That is, given menu L and advice A, the agent's choice is given by $c_M(L) = M(L, A)$, with runtime $r_M(L, A)$. Then M satisfies the *(weak, or nonuniform)* time constraint $f : \mathbb{N} \to \mathbb{R}_+$ if there exists a sequence of advice strings A_k whose length grows at most polynomially in k, where

$$r_M\left(L, A_{k(L)}\right) \leq f(k(L)) \quad \forall L \in \mathcal{L} \tag{2}$$

Definition 4. *The choice correspondence c_M is* weakly tractable ($c \in \mathrm{P/poly}$) *if M satisfies the time constraint* (9) *for some function $f(k)$ that grows at most polynomially in k.*

As stated, my representation theorem refers to weak, rather than strong, tractability. On the one hand, this strengthens the hardness result, which motivates Hadwiger separability. On the other hand, it weakens the tractability result, which establishes that my characterization is tight. It is an open question whether the latter result still holds if we replace weak with strong tractability.

2.3 High-Dimensional Choice

I specialize the model of choice under risk in order to focus on high-dimensional choice problems. The high-dimensional model is defined by four assumptions.

Assumption 2. *The consequences $Z \subseteq \mathbb{R}^\infty$ are infinite sequences $z = (z_i)_{i=1}^\infty$ of real numbers.*

according to some utility function u. There always exists a strongly tractable choice correspondence c_L that chooses l from menu L. It simple hardcodes the choice: given a menu L', if $L = L'$ then output $\{l\}$, otherwise output the entire menu L'. Of course, c_L does not maximize expected utility on every menu, only on L. One could redefine c_L to maximize expected utility on any finite set of menus, by hardcoding the agent's choices. However, when the collection of menus is infinite, this is not possible. One has to either find a better algorithm than hardcoding, or accept a basic tradeoff: optimal choices in some menus may imply suboptimal choices in others.

Treating consequences as infinite sequences lets us define *n-dimensional* consequences for arbitrarily large n. Specifically, the consequence z is n-dimensional if it takes on value $z_i = 0$ for all $i > n$. For example, suppose the agent is a consumer and z_i describes how of good i she consumes. There is an infinite number of possible goods, but in practice, she will only be presented with menus where $n < \infty$ goods are available. The unavailable goods are given a default value $z_i = 0$.

The next assumption ensures that the consequence space is rich: it includes the n-dimensional unit cube for all $n < \infty$. Since this is the only assumption on the consequence space, my results do not rely on the agent making choices involving truly infinite-dimensional consequences.

Assumption 3. *For every dimension $n \geq 1$,*

$$Z \supseteq [0,1]^n \times \prod_{i=n+1}^{\infty} \{0\}$$

The next two assumptions say that the agent is capable of expressing choices over a sufficiently rich collection of menus. The first is needed for the expected utility theorem of von Neumann and Morgenstern (1944), which is a prerequisite for my representation theorem.

Assumption 4. *The collection $\mathcal{L}$ includes all menus with three or fewer lotteries.*

Finally, I require the collection of menus to include *product menus*. These reflect the simplest kind of high-dimensional choice, where an agent faces n decisions and her choice in dimension i does not affect the feasibility of choices in dimension j. Formally, L is a product menu if it can be decomposed as the Cartesian product of n *submenus* L_i, i.e.

$$L = L_1 \times \ldots \times L_n \times \prod_{i=n+1}^{\infty} \{0\}$$

For convenience, let $\mathcal{L}^n$ be the collection of n-dimensional product menus.

Assumption 5. *The collection $\mathcal{L}$ includes all product menus.*

These are my only richness assumptions on the collection of menus, and they do not require the agent to solve constrained optimization problems. Instead, my results are driven by the combinatorics of high-dimensional unconstrained optimization. If we required the agent to make choices involving constraints (e.g. budgets), that could only strengthen my hardness results.

Finally, let us consider the reason why dimensionality drives computational hardness. Let L be an n-dimensional product menu. The number of characters $k(L)$ needed to describe L is increasing linearly in n. However, the number of lotteries $l \in L$ is growing exponentially in n. This means that the *brute-force algorithm*, which evaluates the payoff of every $l \in L$ and chooses the best one, is impractical. Nonetheless, we will see that the brute-force algorithm – although naive and impractical – is essentially the best we can do, unless u is Hadwiger separable.

3 Representation Theorem (warmup)

In this section, I motivate additive separability.

Definition 5. *A utility function $u : Z \to \mathbb{R}$ is* additively separable *if there exist univariate functions $u_i : \mathbb{R} \to \mathbb{R}$ such that*

$$u(z) = \sum_{i=1}^{\infty} u_i(z_i), \quad \forall z \in Z$$

Theorem 1 shows that rational, strongly tractable, and symmetric choice implies expected utility maximization with an additively separable utility function.

Definition 6. *A utility function u is* symmetric *if for any permutation k of $(z_1, \ldots, z_n)$,*

$$u(z) = u(z_{k_1}, \ldots, z_{k_n}, z_{n+1}, \ldots)$$

This result relies on a computational hardness conjecture called P $\neq$ NP.[9] Proposition 1 gives a partial converse: if we restrict attention to product menus, expected utility maximization is strongly tractable when the utility function is additively separable.

3.1 Satisfiability and Computational Hardness Conjectures

To state the P $\neq$ NP conjecture, and to outline the theorem 1's proof, I need to review an important class of computational problems that arise in mathematical logic.

The satisfiability problem (SAT) asks whether a logical expression is possibly true, or necessarily false. To define it, I need to introduce terminology from mathematical logic. A *boolean variable* x can be assigned values, $x = \mathtt{true}$ or $x = \mathtt{false}$. A *literal* is an assertion that x is true (x) or false ($\neg x$). A *clause* c is a sequence of literals combined by "or" statements. For example,

$$c = (x_1 \vee \neg x_2 \vee x_3)$$

A *boolean formula* b in *conjunctive normal form* (CNF) is a sequence of clauses combined by "and" statements. For example,

$$b = c_1 \wedge c_2 = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_3)$$

A formula b is *satisfied* if there exists some assignment of values to variables $x_1, \ldots, x_n$ that makes b true. SAT asks whether a given formula is satisfiable.

There are many variants of SAT. The most important is 3SAT. It restricts attention to formulas where each clause c_j has at most three literals. Similarly, 2SAT restricts attention to formulas where each clause c_j has at most two literals. MAX2SAT also restricts attention to formulas where each

[9] Most results in computational complexity theory rely on a computational hardness conjecture, like P $\neq$ NP. For example, the well-known result that finding Nash equilibria is computationally-hard relies on the conjecture that PPAD $\neq$ FP (Daskalakis et al. 2009). This is somewhat stronger than P $\neq$ NP.

clause c_j has at most two literals, but it does not ask whether the formula is satisfiable. Instead, it asks for the largest number of clauses that can be satisfied simultaneously.

The famous $\mathrm{P} \neq \mathrm{NP}$ conjecture states that there is no Turing machine that solves SAT in polynomial time (Cook 1971). An equivalent formulation states that there is no Turing machine that solves 3SAT in polynomial time (Cook 1971), or that solves other notoriously-hard problems in polynomial time (Karp 1972).

Assumption 6 ($\mathrm{P} \neq \mathrm{NP}$). *There is no Turing machine that solves 3SAT in polynomial time.*

3.2 Motivating Additive Separability

Theorem 1 will conclude that the agent's choices can be rationalized by an additively separable utility function u. It will also conclude that u is *efficiently computable*.

Roughly, a utility function u is efficiently computable if there exists a quick algorithm that computes $u(z)$ for any given consequence z, with at most ϵ imprecision.

Definition 7. *A utility function u is* efficiently computable *if there exists a Turing machine M that takes in a constant $\epsilon > 0$ and an n-tuple $(z_1, \dots, z_n) \in \left[\underline{z}, \bar{z}\right]$, and then outputs $x \in \mathbb{R}$ where*

$$x - \epsilon \leq \frac{u^n(z_1, \dots, z_n) - u^n\left(\underline{z}, \dots, \underline{z}\right)}{u^n\left(\bar{z}, \dots, \bar{z}\right) - u^n\left(\underline{z}, \dots, \underline{z}\right)} \leq x + \epsilon \tag{3}$$

with runtime $O(\mathrm{poly}(n, 1/\epsilon))$.

This is a mild condition that likely holds for most or all utility functions of interest to economists. In particular, efficient computability does not imply that expected utility maximization is tractable, and efficient computability does not imply any kind of separability.

Theorem 1. *Let c be a rational, symmetric, and strongly tractable choice correspondence. There exists a continuous, efficiently computable, symmetric, and additively separable utility function $u : Z \to \mathbb{R}$ such that*

$$c(L) = \arg\max_{l \in L} \mathrm{E}[u(l)]$$

for any given menu $L \in \mathcal{L}$.

The fact that there is continuous utility function u follows from von Neumann and Morgenstern (1944). The substance of the proof is showing that u must be additively separable. Essentially, we must prove that if u is not additively separable, then expected utility maximization is computationally hard. Note that we have to prove this for *any* utility function u that is not additively separable, rather than just prove it for *some* utility function that is not additively separable. [10]

[10] Recall that utility functions u define choice correspondences which are computational problems themselves, not instances of a problem. In that sense, this result (combined with proposition 1) is similar to the dichotomy theorem of Schaefer (1978), which partitions a class of computational problems into two categories: tractable or intractable (i.e. NP-hard).

The hardness proof follows a reduction argument. Suppose for contradiction that an agent has a utility function that is not additively separable but can nonetheless make an optimal choice $c(L)$ from any given menu L within polynomial time. Let f be a boolean formula with k variables. I construct an n-dimensional menu L_u^f such that $c(L_u^f)$ gives a solution to MAX2SAT for instance f. Since c is strongly tractable, MAX2SAT can be solved in polynomial time. Since MAX2SAT is NP-hard (Johnson 1974), this contradicts P $\neq$ NP The construction of menu L_u^f takes some time to describe, so I refer the interested reader to the full proof in the appendix.

Next, I argue that my characterization of rational and tractable choice is tight, by presenting a partial converse to theorem 1.

Proposition 1. *Let the utility function u be efficiently computable and additively separable. Let c be the choice correspondence induced by expected utility maximization. Then c is strongly tractable as long as the collection $\mathcal{L}$ is restricted to the menus specified in assumptions 4 (ternary menus) and 5 (product menus).*[11]

This proof is constructive. The behavioral heuristic known as narrow choice bracketing is both tractable and can maximize expected utility when the utility function is additively separable. However, it is only well-defined in product menus. To complete the proof, note that it is straightforward to maximize expected utility in ternary menus in polynomial time using brute-force search.

4 Hadwiger Separability

To generalize the representation theorem beyond symmetric choice correspondences, I need to relax additive separability to Hadwiger separability.

Hadwiger separability is a relaxation of additive separability that allows for some complementarities and substitutions but limits their frequency. Its main advantage relative to additive separability is that it preserves computational tractability (see section 5.2) while being capable of modeling a much richer class of phenomena. Nonetheless, it is restrictive enough to prove non-trivial results (see section 6). It also rules out many utility functions that seem natural but describe computationally-implausible behavior.

In this section, I define Hadwiger separability and clarify its economic meaning through a series of examples. To do this, I require some notation. Let $Z \subseteq \mathbb{R}^\infty$ be a set of real-valued sequences. Let $u : Z \to \mathbb{R}$ map sequences to real numbers. This function u is *additively separable* if there exist univariate functions $u_i : \mathbb{R} \to \mathbb{R}$ such that

$$u(z) = \sum_{i=1}^{\infty} u_i(z_i), \quad \forall z \in Z$$

[11] The restriction on $\mathcal{L}$ is necessary. If $\mathcal{L}$ included all possible menus, then no choice correspondences would be both rational and tractable. For example, it is straightforward to construct menus L where it is computationally hard to find any feasible lottery $l \in L$. Of course, these menus may not be of economic interest.

It is (i, j)-*separable* if there exist functions u_i, u_j such that,

$$u(z) = u_i(z_i, z_{-ij}) + u_j(z_j, z_{-ij}), \quad \forall z \in Z$$

Informally, the *inseparability graph* $G(u, n)$ describes patterns of inseparability across the first n dimensions of u. The function u is *Hadwiger separable* if the inseparability graph $G(u, n)$ becomes sparse as n grows large. To formalize these concepts, I require some terminology from graph theory.

Remark 1. An *undirected graph* $G = (V, E)$ has a set of vertices V and a set of edges $E \subseteq V^2$. The *order* of G is the number $|V|$ of vertices, while its *size* is the number $|E|$ of edges. The vertices $i, j \in V$ are *adjacent* if there exists an edge $(i, j) \in E$ connecting them. The graph is *complete* if every pair of vertices is adjacent and it is *empty* if no pair is adjacent.

I require a measure of graph sparsity. It turns out that the right measure, for my representation, was formulated by Hadwiger (1943) to state his famous conjecture about the chromatic number of graphs. Defining it requires some additional terminology. The graph $G' = (V', E')$ is a *subgraph* of $G = (V, E)$ if $V' \subseteq V$ and $E' \subseteq E$. The subgraph G' is a *minor* of G if it can be formed by repeatedly applying the following operations to G:

1. Delete a vertex i and all of its incident edges (i, j).
2. Contract an edge (i, j). This deletes vertices i, j and replaces them with a new vertex k. It also replaces any edges (i, l) and (j, l) with a new edge (k, l).

The *Hadwiger number* $\text{Had}(G)$ of G is the order of its largest complete minor. Figure 3 clarifies this definition, through an example. □

Definition 8. *The* inseparability graph $G(u, n)$ *of a function u is an undirected graph with n vertices, where vertices i, j are adjacent iff if u is not (i, j)-separable.*

Definition 9. *The function u is* Hadwiger separable *if*[12]

$$\text{Had}(G(u, n)) = O(\log n)$$

The easiest way to understand this concept is through examples. In the next three examples, I illustrate how Hadwiger separability becomes more interpretable when combined with other modeling assumptions.

Example 1. Consider a firm with n different branches. Branch i generates revenue r_i at cost c_i. Let $u(r, c)$ be the firm's utility function. It is natural to assume that the firm cares about profits

$$\pi(r, c) = \sum_{i=1} (r_i - c_i)$$

[12]Since computing the Hadwiger number is NP-hard (Eppstein 2009), some readers may wonder whether this condition can be verified in a reasonable amount of time. Basically, it can be. That is, given an inseparability graph $G(u, n)$ and a constant C, it is possible to verify whether $\text{Had}(G(u, n)) \leq C \log n$ within poly(n, C)-time. This follows from a fixed parameter tractability result due to Alon et al. (2007).

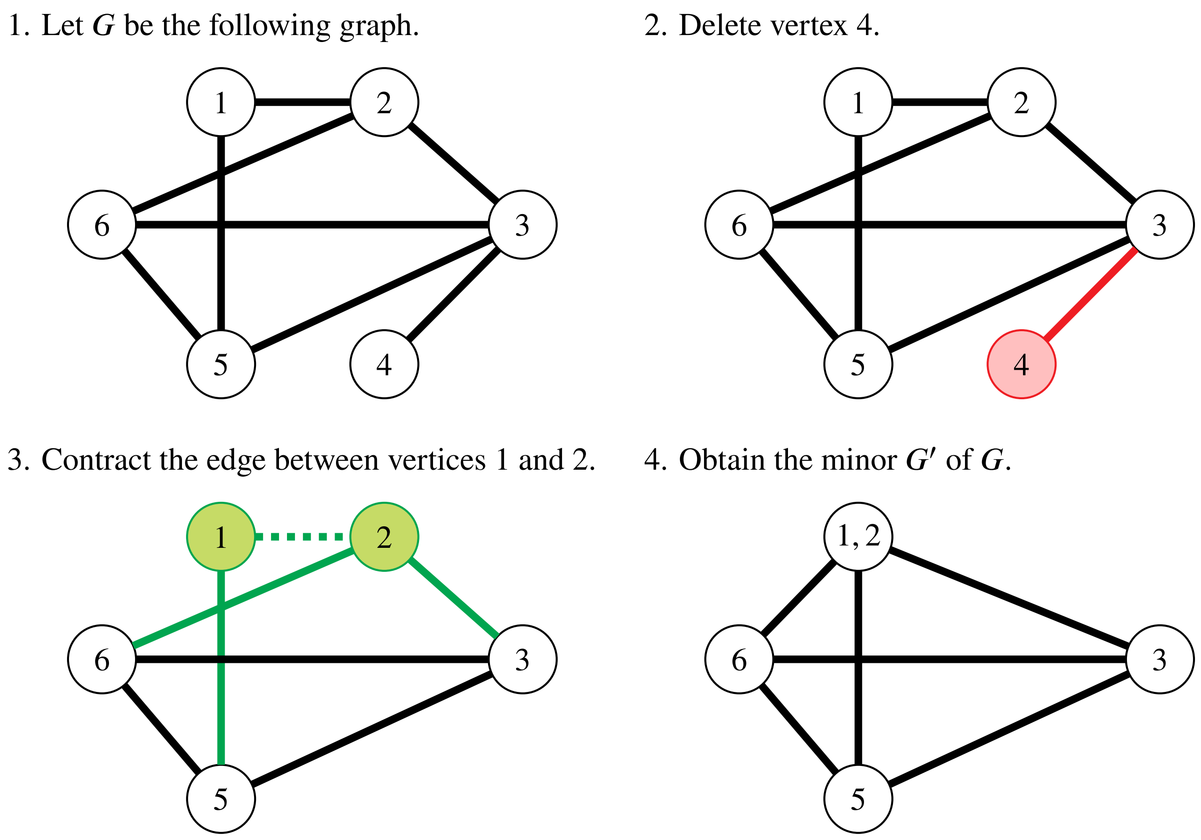

1. Let G be the following graph.
2. Delete vertex 4.
3. Contract the edge between vertices 1 and 2.
4. Obtain the minor G' of G.
1
2
3
4
5
6
1, 2

Figure 3: In this example, we find the Hadwiger number of the graph G. The minor G' is complete and has four vertices. In fact, this is the largest complete minor, so $\mathrm{Had}(G) = 4$.

An empty graph G, with $\text{Had}(G) = 0$. A complete graph G, with $\text{Had}(G) = 8$.

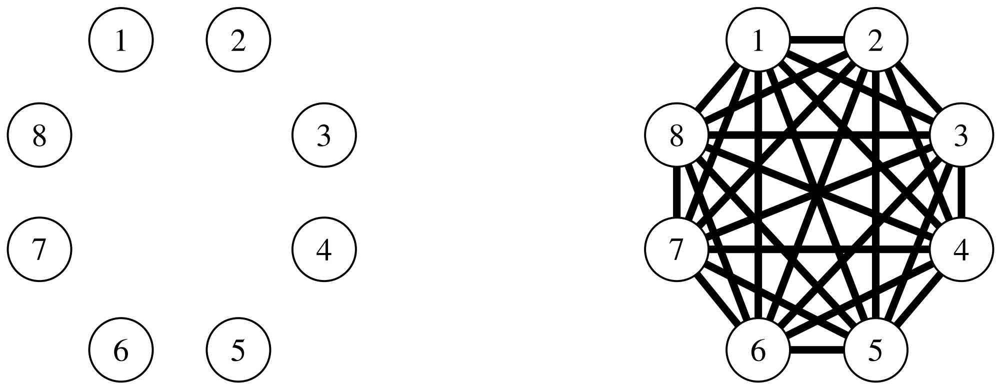


Figure 4: When the function u is symmetric, the inseparability graph $G(u, n)$ is either empty or complete. If $G(u, n)$ is empty (left), then u is additively separable. If $G(u, n)$ is complete (right), then $\text{Had}(G(u, n)) = n$ and therefore u is not Hadwiger separable. It follows that Hadwiger separability is equivalent to additive separability when u is symmetric.

Let g describe the firm's utility from wealth, and define $u(r, c) = g(\pi(r, c))$. Then u is Hadwiger separable iff g is linear. That is, Hadwiger separability is equivalent to risk-neutrality.

More generally, suppose the firm cares about per-branch profit $\pi_i = r_i - c_i$, but views branches as interchangeable. That is, if branch j generated profit π_i and branch i generated profit π_j, the firm's utility would not change. Then $u(\pi_1, \dots, \pi_n)$ is Hadwiger separable iff it is additively separable. □

That last point can be made more generally.

Proposition 2. *Suppose that u is symmetric. Then u is Hadwiger separable iff it is additively separable.*

Proof. See figure 4. □

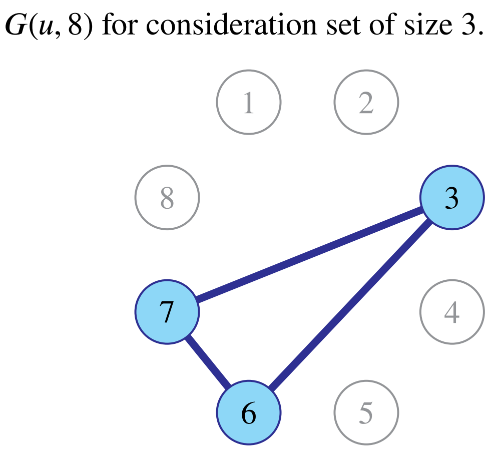
Example 2. Consider a consumer in the market for ready-to-eat cereal. There are n brands available at her grocery store (Corn Flakes, Cheerios, Raisin Bran, Froot Loops, etc). Let x_i indicate how many units of brand i she purchases, and let p denote her total payments. Her utility function

$$u(x, p) = v(x) - p$$

is quasilinear in payments, where v describes her value from the cereals purchased. The consumer sees different brands as substitutes, and I formalize this by letting v be strictly submodular.

The consumer may be overwhelmed by the number of brands available. Rather than maximize her utility directly, she forms a consideration set $C \subseteq \{1, \dots, n\}$. This is an exogenous set of brands that she may be interested in. Without loss of generality, suppose that brands $1, \dots, m$ are in the consideration set, and the remaining brands $m + 1, \dots, n$ are not. Her new utility function

$$\hat{u}(x) = v(x_1, \dots, x_m, 0, \dots, 0) - p$$



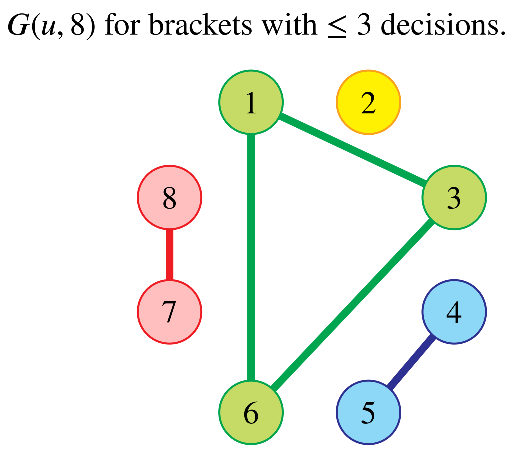


Figure 5: Depiction of examples 2 and 3. On the left, the agent forms consideration set that consists of brands $\{3, 6, 7\}$. On the right, the agent forms brackets $\{1, 3, 6\}$, $\{2\}$, $\{4, 5\}$, and $\{7, 8\}$. In both cases, the inseparability graphs $G(u, 8)$ appear to be consistent with Hadwiger separability, because their Hadwiger numbers are $3 = \log_2 8$.

ignores purchases x_i of brands $i \notin C$. It is Hadwiger separable iff $m = O(\log n)$. That is, Hadwiger separability limits the size of the consideration set when the number of brands is large. $\square$

Example 3. Consider a consumer in a grocery store, choosing from n generic products (e.g. milk, bread, cheese, salami, cereal, etc). Let x_i indicate how many units of product i she purchases, and let p denote her total payments. Her utility function

$$u(x, p) = v(x) - p$$

is quasilinear in payments, where v describes her value from the products purchased. The value v may feature complementarities across different products, like cereal and milk, or substitutions, like cheese and salami.

The consumer may be overwhelmed by the number of product combinations available. Rather than maximize her utility directly, she partitions the set products into m brackets $C_j \subseteq \{1, \ldots, n\}$. For example, one bracket could represent breakfast foods, while another might represent vegetables. Then she follows a choice bracketing heuristic. Specifically, she attempts to optimize her choices within each bracket C_j without considering her purchases $i \notin C_j$. Her revealed utility function $\hat{u}(x)$ is additively separable across brackets C_j, but not necessarily within brackets. It is Hadwiger separable iff $m = O(\log n)$. That is, Hadwiger separability forces the agent to bracket relatively narrowly by limiting the size of the largest bracket. $\square$

Figure 5 illustrates the preceding two examples.

5 Representation Theorem (continued)

In this section, I motivate Hadwiger separability. Theorem 2 shows that rational and weakly tractable choice implies expected utility maximization with a Hadwiger separable utility function. This result relies on a computational hardness conjecture called the non-uniform exponential time hypothesis.[13] Theorem 3 gives a partial converse: if we restrict attention to product menus, expected utility maximization is weakly tractable when the utility function is Hadwiger separable.

I assume that the following conjecture holds, throughout the paper.

Assumption 7 (Nonuniform Exponential Time Hypothesis)**.** *There is no Turing machine that solves* 3SAT *in subexponential time with at most polynomial-size advice.*

5.1 Tractability implies Hadwiger Separability

Theorem 2 will conclude that the agent's choices can be rationalized by a Hadwiger separable utility function u. It will also conclude that u is efficiently computable.

Theorem 2. *Let c be a rational and weakly tractable choice correspondence. There exists a continuous, efficiently computable, and Hadwiger separable utility function $u : Z \to \mathbb{R}$ such that*

$$c(L) = \arg\max_{l \in L} \mathrm{E}[u(l)]$$

for any given menu $L \in \mathcal{L}$.

The fact that there is continuous utility function follows from von Neumann and Morgenstern (1944). The substance of the proof is showing that u must be Hadwiger separable. Essentially, we must prove that if u is not Hadwiger separable, then expected utility maximization is computationally hard. A key challenge is that we must simultaneously prove it for any utility function u that is not Hadwiger separable. Recall that utility functions u define choice correspondences which are computational problems themselves, not instances of a problem. In that sense, this result (combined with theorem 3) is similar to the dichotomy theorem of Schaefer (1978), which partitions a class of computational problems into two categories: tractable or intractable (i.e. NP-hard).

The hardness proof follows a reduction argument. Suppose for contradiction that an agent has a utility function that is not Hadwiger separable but can nonetheless make an optimal choice $c(L)$ from any given menu L within polynomial time. Take the largest complete minor $H(u, n)$ of $G(u, n)$ as advice. This object can be described with poly(n) space. Let k be the order of $H(u, n)$. By definition, this is the Hadwiger number of $G(u, n)$. Since u was presumed to not be Hadwiger separable, this implies $k = \omega(\log n)$. Let f be a boolean formula with k variables. I construct an n-dimensional menu L_u^f such that $c(L_u^f)$ gives a solution to MAX2SAT for instance f. Essentially, the more inseparable u is, the more powerful the algorithmic reduction. Since c is weakly tractable

[13]Most results in computational complexity theory rely on a computational hardness conjecture, like P $\neq$ NP. For example, the well-known result that finding Nash equilibria is computationally-hard relies on the conjecture that PPAD $\neq$ FP (Daskalakis et al. 2009). This is somewhat stronger than P $\neq$ NP.

and $k = \omega(\log n)$, MAX2SAT can be solved in subexponential time with advice $H(u, n)$. Then 3SAT can be solved in subexponential time with polynomial advice, which contradicts non-uniform exponential time hypothesis. The construction of menu L_u^f takes some time to describe, so I refer the interested reader to the full proof in the appendix.

5.2 When Hadwiger Separability implies Tractability

Next, I show that my characterization of rational and tractable choice is tight, by presenting a partial converse to theorem 2.

Theorem 3. *Let the utility function u be efficiently computable and Hadwiger separable. Let c be the choice correspondence induced by expected utility maximization. Then c is weakly tractable as long as the collection $\mathcal{L}$ is restricted to the menus specified in assumptions 4 (ternary menus) and 5 (product menus).*[14]

This proof is constructive and the proposed algorithm – *dynamic choice bracketing* – may be interesting in its own right. This heuristic algorithm – like narrow choice bracketing – is only well-defined on product menus.[15] It leverages a form of dynamic programming on graphs to generalize choice bracketing. For example, dynamic choice bracketing may condition on decisions in one (presumably small) subset of submenus L_i, and then narrowly bracket the remaining submenus. The algorithm itself is parameterized by an objective function u and a graph G, which determines the order in which the dynamic program conditions on submenus, and which dimensions i it regards as separable. If $G = G(u, n)$ is the inseparability graph of u, dynamic choice bracketing will exactly maximize expected utility, just like narrow choice bracketing for additively separable u.

To prove theorem 3, it suffices to show that dynamic choice bracketing runs in polynomial time when parameterized by a graph G where $\text{Had}(G) = O(\log n)$. This is because the inseparability graph $G(u, n)$ satisfies this property, by definition, when u is Hadwiger separable. The inseparability graph $G(u, n)$ functions as advice that can be described in polynomial space (of course, for many utility functions, it is possible to describe $G(u, n)$ in polynomial time).[16] Therefore, the algorithm would maximize expected utility in polynomial time with polynomial-size advice. By definition, this means that expected utility maximization is weakly tractable.

The main reason why dynamic choice bracketing may not run in polynomial time is if it reaches a bottleneck, where it is forced to solve a d-dimensional optimization problem. Generally, if the parameter $G(u, n)$ does not include an edge (i, j), the algorithm ignores any dependence in u between dimensions i and j. The fact that there are few edges reduces the likelihood that the algorithm will reach a bottleneck, but does not rule it out. However, I show that, if this algorithm finds itself

[14] The restriction on $\mathcal{L}$ is necessary. If $\mathcal{L}$ included all possible menus, then no choice correspondences would be both rational and tractable. For example, it is straightforward to construct menus L where it is computationally hard to find any feasible lottery $l \in L$. Of course, these menus may not be of economic interest.

[15] Note that it is straightforward to maximize expected utility in ternary menus in polynomial time using brute-force search.

[16] Indeed, it is not clear to me whether there exist any efficiently computable utility functions u where the sequence $G(u, n)$ is not efficiently computable. If none exist, then one could strengthen theorem 3 from weak to strong tractability.

searching over a d-dimensional set, then there exists a complete minor of $G(u, n)$ with order d. Therefore, $d \leq \text{Had}(G(u, n))$. By Hadwiger separability, this implies that $d = O(\log n)$. Brute-force search over $O(\log n)$-dimensional sets takes $O(n)$ time, and so the algorithm halts in $O(\text{poly}(n))$ time.

6 Existence of a Rationality Gap

In this section, I use the representation theorem to obtain a *rationality gap* theorem. Specifically, consider an agent that intrinsically wants to maximize the expected value of some objective function $\bar{u} : Z \to \mathbb{R}$. In the presence of time constraints, she may be better off making choices that violate the expected utility axioms. The rationality gap quantifies "better off".

To understand the rationality gap and why it may exist, consider a time-constrained agent whose true objective $\bar{u}$ is not Hadwiger separable. Her first-best choice correspondence maximizes her expected objective, but theorem 3 shows that it is not tractable. Intuitively, a second-best choice correspondence should approximately maximize her expected objective, but still be tractable. This need not be rationalizable. Imposing rationalizability as an additional constraint gives us a third-best choice correspondence, which should approximately maximize her expected objective, be tractable, and be rationalizable. The rationality gap measures the gap between the second- and third-best, in line with how computer scientists evaluate approximation algorithms.

More formally, let $\mathcal{C}$ be the set of all weakly tractable choice correspondences. Let $\mathcal{C}^R$ be the set of all weakly tractable and rationalizable choice correspondences. This is a strict subset of $\mathcal{C}$ and may exclude weakly tractable choice correspondences that obtain good approximations. The rationality gap quantifies the agent's loss of approximate optimality from restricting attention to choice correspondences $c \in \mathcal{C}^R$, relative to the larger set of choice correspondences $c \in \mathcal{C}$.

Theorem 4 shows that the rationality gap is unbounded, for a large class of objective functions. This kind of argument has precedent, in the *revelation gap* theorem of Feng and Hartline (2018) that evaluates the revelation principle in mechanism design. In prior-independent settings, it shows that designers may obtain strictly better approximations to optimal welfare if they are willing to use non-revelation mechanisms. In that sense, restricting attention to revelation mechanisms is without loss of optimality only if one ignores informational requirements. Similarly, restricting attention to rational choice correspondences is without loss of optimality only if one ignores time constraints.

6.1 Defining the Rationality Gap

To define the rationality gap, I need to quantify how much better one choice correspondence c is than another, say c'. I combine two approaches: weak dominance to rank choice correspondences, and the approximation ratio to compare choice correspondences that are already ranked.

Definition 10. *Let c, c' be choice correspondences. Then c'* weakly dominates *c, denoted by $c' \succeq_{\bar{u}} c$, if*

$$\mathrm{E}\left[\bar{u}(c'(L))\right] \geq \mathrm{E}[\bar{u}(c(L))]$$

for all product menus L, where the inequality is strict for at least one menu.

Unlike weak dominance, the approximation ratio can quantify how much better or worse c is compared to c'. Specifically, it measures how well c can approximate the optimal choice $l \in L$, given an adversarially chosen menu $L \in \mathcal{L}^n$. The approximation ratio is widely used in computer science to evaluate the performance of approximation algorithms.

Definition 11. *Let c be a choice correspondence. Its* approximation ratio *is*

$$\mathrm{APX}(n; c, \bar{u}) = \inf_{L \in \mathcal{L}^n} \left(\frac{\mathrm{E}[\bar{u}(c(L))] - \min_z \bar{u}(z)}{\max_{l \in L} \mathrm{E}[\bar{u}(l)] - \min_z \bar{u}(z)} \right)$$

A rationality gap exists if every rational choice correspondence is weakly dominated by another choice correspondence with a better approximation ratio. Formally, the rationality gap is the value of a zero-sum game. First, the agent chooses $c \in \mathcal{C}^R$. Then the adversary chooses $c' \in \mathcal{C}$, subject to the weak dominance constraint ($c' \succeq_{\bar{u}} c$).[17]

Definition 12. *The* rationality gap *is defined as*

$$\mathrm{RG}(n; \bar{u}) = \inf_{c \in \mathcal{C}^R} \sup_{c' \in \mathcal{C}, c' \succeq_{\bar{u}} c} \frac{\mathrm{APX}(n, c', \bar{u})}{\mathrm{APX}(n, c, \bar{u})}$$

6.2 Class of Objective Functions

My next task is to identify objective functions $\bar{u}$ for which a rationality gap exists, i.e.

$$\lim_{n \to \infty} \mathrm{RG}(n; \bar{u}) > 1$$

Clearly this cannot be true in all cases. For example, suppose that $\bar{u}$ is Hadwiger separable. By theorem 3, exact optimization is weakly tractable. Therefore, restricting attention to rational choice correspondences is without loss, and the agent can guarantee $\mathrm{RG}(n; \bar{u}) = 1$.

It turns out that there is a large class of objectives for which the rationality gap not only exists, but is unbounded. This class is defined by three assumptions. First, $\bar{u}$ must be strictly increasing. Second, $\bar{u}(z)$ should grow at most sublinearly in the dimension n of consequence z.

Definition 13. *The objective function $\bar{u}$ is* sublinear *if there exists a constant $\epsilon > 0$ such that*

$$\bar{u}(\underbrace{1, \ldots, 1}_{n \text{ times}}, 0, \ldots) = O(n^{1-\epsilon})$$

The third assumption reflects diminishing returns: an agent that prefers consequences z to z' should not prefer z to z' even more if both options are made better by the same amount z''.

[17] Readers with a computer science background might have expected a different definition, without weak dominance, where the adversary takes a supremum over all choice correspondences $c \in \mathcal{C}$. I incorporate weak dominance in order to reassure readers who are skeptical of the approximation ratio as a way to rank choice correspondences. In any case, the distinction turns out to be unimportant, since theorem 4 holds according to either definition.

Definition 14. *The objective function $\bar{u}$ features* diminishing returns *if*

$$u(z) - u(z') \geq u(z + z'') - u(z' + z'')$$

for any n-consequences $z, z', z'' \in Z$ where $z'' \in \mathbb{R}^n_+$ is non-negative.

The easiest way to understand these conditions is through an example.

Example 4. Consider an objective function that takes the form

$$\bar{u}(z) = g\left(\sum_{i=1}^{n} z_i\right)$$

This satisfies all three assumptions as long as g is strictly increasing, concave, and $g(x) = O(x^{1-\epsilon})$. For example, $g(x) = \sqrt{x}$ and $g(x) = \log x$ satisfy these conditions. The same is true if g satisfies constant absolute risk aversion or constant relative risk aversion, as long as it is not risk-neutral. □

6.3 Theorem and Proof Sketch

Finally, I am ready to state the rationality gap theorem.

Theorem 4. *Let $\bar{u}$ be an efficiently computable, strictly increasing, and sublinear objective function with diminishing returns. Then the rationality gap is unbounded, i.e.*

$$\lim_{n\to\infty} \mathrm{RG}^n(\bar{u}) = \infty$$

The proof relies on two lemmas. The first lemma establishes that no rational choice correspondence $c \in C^R$ can guarantee a constant, nonzero approximation ratio.

Lemma 1. *Let $\bar{u}$ be a strictly increasing, sublinear objective function. For any weakly tractable and rational choice correspondence $c \in C^R$, the approximation ratio vanishes, i.e.*

$$\lim_{n\to\infty} \mathrm{APX}(n; c, \bar{u}) = 0$$

The proof involves constructing a menu L^c where the choice correspondence performs poorly according to the objective $\bar{u}$. First, since c is weakly tractable and rational, theorem 2 ensures that it corresponds to expected utility maximization for a Hadwiger separable utility function. By proposition **??**, this implies that the utility function is degenerate. Next, consider the inseparability graph $G(u, n)$. By definition of degeneracy, $\mathrm{dgn}(G(u, n)) = O(\log n)$, which implies that the chromatic number of $G(u, n)$ is at most logarithmic in n (Szekeres and Wilf 1968). I color the graph $G(u, n)$ using the minimal number of colors. Finally, I construct a menu L^c where the agent appears to be narrowly bracketing dimensions associated with vertices of the most frequent color, and ignores the remaining dimensions. There are $\Omega(n/\log n)$ such vertices, a number which grows faster in n

than $\bar{u}$ does, by sublinearity. This allows me to finish specifying Z^c in a way that (a) makes narrow choice bracketing suboptimal, and (b) ensures that the suboptimality grows in n.

The second lemma establishes that it is possible to guarantee a constant, nonzero approximation ratio using a choice correspondence $c \in C$ where $c \notin C^R$.

Lemma 2. *Let $\bar{u}$ be a strictly increasing objective function with diminishing returns. There exists a weakly tractable choice correspondence $c^G \in C$ where*

$$\mathrm{APX}(n; c^G, \bar{u}) \geq \frac{1}{2}$$

The proof is constructive. I begin with Johnson's (1974) greedy algorithm, which guarantees an approximation ratio of at least $1/2$ for the MAX2SAT problem. This immediately gives an approximation algorithm for the objective function $\bar{u}(z) = \max_i z_i$. I generalize the algorithm and the approximation result to the larger class of strictly increasing objective functions with diminishing returns, which capture the essential properties needed for Johnson's (1974) result to go through. Then I define c^G to be the output of this greedy algorithm.

Of course, these lemmas are not quite sufficient to prove theorem 4. They only show that c^G obtains a better approximation ratio than any $c \in C^R$, not that c^G weakly dominates c. However, it is easy to modify c^G to a meta-algorithm c^M that weakly dominates c. Specifically, for a given menu L, let c^M choose the best lottery $l \in c^G(L) \cup c(L)$ chosen by either c^G or c. We know this is weakly tractable, since both c^G and c are weakly tractable. Moreover, c^M weakly dominates c and preserves the superior approximation ratio of c^G. This completes the proof.

References

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

Andersen, S., Cox, J. C., Harrison, G. W., Lau, M. I., Rutström, E. E., & Sadiraj, V. (2018, December). Asset integration and attitudes to risk: theory and evidence. *The Review of Economics and Statistics*, *100*(5), 816–830.

Andreoni, J., Gravert, C., Kuhn, M. A., Saccardo, S., & Yang, Y. (2018, November). *Arbitrage or narrow bracketing? on using money to measure intertemporal preferences* (Working Paper No. 25232). National Bureau of Economic Research.

Apesteguia, J. & Ballester, M. A. (2010). The computational complexity of rationalizing behavior. *Journal of Mathematical Economics*, *46*(3), 356–363.

Arora, S. & Barak, B. (2009). *Computational complexity: a modern approach*. Cambridge University Press.

Barman, S., Bhaskar, U., Echenique, F., & Wierman, A. (2014). On the existence of low-rank explanations for mixed strategy behavior. In T.-Y. Liu, Q. Qi, & Y. Ye (Eds.), *Web and internet economics* (pp. 447–452). Cham: Springer International Publishing.

Bergman, C. H. (2012). *Universal Algebra: Fundamentals and Selected Topics*. Pure and Applied Mathematics: A Series of Monographs and Textbooks. Boca Raton, FL: CRC Press.

Bernstein, E. & Vazirani, U. (1997). Quantum complexity theory. *SIAM Journal on Computing*, *26*(5), 1411–1473.

Brown, J. R., Kapteyn, A., Luttmer, E. F., Mitchell, O. S., & Samek, A. (2017, December). *Behavioral impediments to valuing annuities: evidence on the effects of complexity and choice bracketing* (Working Paper No. 24101). National Bureau of Economic Research.

Chaudhry, S. J., Hand, M., & Kunreuther, H. (2020, June). *Broad Bracketing for Low Probability Events* (NBER Working Papers No. 27319). National Bureau of Economic Research, Inc.

Cook, S. A. (1971). The complexity of theorem-proving procedures. In *Proceedings of the third annual acm symposium on theory of computing* (pp. 151–158). STOC '71. Shaker Heights, Ohio, USA: ACM.

Daskalakis, C., Goldberg, P. W., & Papadimitriou, C. H. (2009). The complexity of computing a nash equilibrium. *SIAM Journal on Computing*, *39*(1), 195–259.

Echenique, F., Golovin, D., & Wierman, A. (2011). A revealed preference approach to computational complexity in economics. In *Proceedings of the 12th acm conference on electronic commerce* (pp. 101–110). EC '11. San Jose, California, USA: Association for Computing Machinery.

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

Feng, Y. & Hartline, J. D. (2018, October). *An end-to-end argument in mechanism design (prior-independent auctions for budgeted agents)*. Los Alamitos, CA, USA: IEEE Computer Society.

Fortnow, L. & Vohra, R. V. (2009). The complexity of forecast testing. *Econometrica*, *77*(1), 93–105.

Hadwiger, H. (1943). Über eine klassifikation der streckenkomplexe. *Vierteljahrsschr Naturforsch, Ges. Zurich*, *88*, 133–142.

Haisley, E., Mostafa, R., & Loewenstein, G. (2008, August). Myopic risk-seeking: the impact of narrow decision bracketing on lottery play. *Journal of Risk and Uncertainty*, *37*(1), 57–75.

Hązła, J., Jadbabaie, A., Mossel, E., & Rahimian, M. A. (forthcoming). Bayesian decision making in groups is hard. *Operations Research*.

Jakobsen, A. M. (2020, May). A model of complex contracts. *American Economic Review*, *110*(5), 1243–73.

Johnson, D. S. (1974). Approximation algorithms for combinatorial problems. *Journal of Computer and System Sciences*, *9*(3), 256–278.

Karp, R. M. (1972). Reducibility among combinatorial problems. In R. E. Miller, J. W. Thatcher, & J. D. Bohlinger (Eds.), *Complexity of computer computations: proceedings of a symposium on the complexity of computer computations* (pp. 85–103). Boston, MA: Springer US.

Karp, R. M. & Lipton, R. J. (1980). Some connections between nonuniform and uniform complexity classes. In *Proceedings of the twelfth annual acm symposium on theory of computing* (pp. 302–309). STOC '80. Los Angeles, California, USA: Association for Computing Machinery.

Koch, A. K. & Nafziger, J. (2019). Correlates of narrow bracketing. *The Scandinavian Journal of Economics*, *121*(4), 1441–1472.

Mandler, M. (2015). Rational agents are the quickest. *Journal of Economic Theory*, *155*, 206–233.

Mandler, M., Manzini, P., & Mariotti, M. (2012). A million answers to twenty questions: choosing by checklist. *Journal of Economic Theory*, *147*(1), 71–92.

Martin, V. (2017). When to quit: narrow bracketing and reference dependence in taxi drivers. *Journal of Economic Behavior & Organization*, *144*, 166–187.

Moher, E. & Koehler, D. J. (2010, August). Bracketing effects on risk tolerance: Generalizability and underlying mechanisms. *Judgment and Decision Making*, *5*(5), 339–346.

Al-Najjar, N. I., Casadesus-Masanell, R., & Ozdenoren, E. (2003). Probabilistic representation of complexity. *Journal of Economic Theory*, *111*(1), 49–87.

Pennings, J. M., Isengildina-Massa, O., Irwin, S. H., Garcia, P., & Good, D. L. (2008). Producers' complex risk management choices. *Agribusiness*, *24*(1), 31–54.

Rabin, M. & Weizsäcker, G. (2009, September). Narrow bracketing and dominated choices. *American Economic Review*, *99*(4), 1508–43.

Read, D., Loewenstein, G., & Rabin, M. (1999, December). Choice bracketing. *Journal of Risk and Uncertainty*, *19*(1), 171–197.

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

Schaefer, T. J. (1978). The complexity of satisfiability problems. In *Proceedings of the tenth annual acm symposium on theory of computing* (pp. 216–226). STOC '78. San Diego, California, USA: Association for Computing Machinery.

Simonsohn, U. & Gino, F. (2013). Daily horizons: evidence of narrow bracketing in judgment from 10 years of m.b.a. admissions interviews. *Psychological Science*, *24*(2), 219–224.

Stracke, R., Kerschbamer, R., & Sunde, U. (2017). Coping with complexity: experimental evidence for narrow bracketing in multi-stage contests. *European Economic Review*, *98*, 264–281.

Szekeres, G. & Wilf, H. S. (1968). An inequality for the chromatic number of a graph. *Journal of Combinatorial Theory*, *4*(1), 1–3.

Tversky, A. & Kahneman, D. (1981). The framing of decisions and the psychology of choice. *Science*, *211*(4481), 453–458.

von Neumann, J. & Morgenstern, O. (1944). *Theory of games and economic behavior*. Princeton University Press.

Zhang, M. (2021, February). *A theory of choice bracketing under risk*.

A Omitted Proofs

A.1 Proof of Theorem 1

First, I prove a simpler theorem: that tractability, rationality, and permutation-invariance imply additive separability of the utility function. This highlights the basic idea behind the reduction. Then I adapt the argument to prove theorem 2, which drops permutation-invariance and motivates Hadwiger separability of the utility function.

The following observation will be useful. If a rational, monotone choice correspondence c is tractable, then the utility function u is efficiently computable (up to affine transformations).

Lemma 3. *Assume rationality and weak tractability. Then u is efficiently computable.*

Proof. Let $z = (z_1, \dots, z_n, e_{n+1}, e_{n+2}, \dots)$. Define a constant n-dimensional lottery h^z where $h^z(\theta) = z$ for all $\theta \in [0, 1]$. Define an regular n-dimensional lottery h^x where $h^x(\theta) = \left(\underline{z}, \dots, \underline{z}, e_{n+1}, e_{n+2}, \dots\right)$ for $\theta \leq x$ and $h^\alpha(\theta) = \left(\bar{z}, \dots, \bar{z}, e_{n+1}, e_{n+2}, \dots\right)$ for $\theta > x$. Let M_c be the TM referred to in the tractability axiom. Define

$$I_x = \mathbf{1}\left(h^\alpha = M_c\left(\{h^z, h^x\}\right)\right)$$

Redefine $\epsilon := \min\{\epsilon, 1\}$, let $k = \lfloor 1/\epsilon \rfloor$, and construct a grid

$$X = \{\epsilon, 2\epsilon, \dots, (k-1)\epsilon, k\epsilon\}$$

Iterate over all $x \in X$. Output the smallest x such that $I_x = 1$. □

I want to show that if u is label-invariant and not additively separable, then $c \in$ NP-hard. First, I require some notation.

Definition 15 (Pairwise Violation)**.** *A finite-dimensional consequence $z \in Z$ and quadruple $(a, b, c, d) \in [\underline{z}, \bar{z}]^4$ constitutes a (i, j)-pairwise violation of additive separability if*

$$\begin{aligned} &u\left(\dots, z_{i-1}, a, z_{i+1}, \dots, z_{j-1}, b, z_{j+1}, \dots\right) + u\left(\dots, z_{i-1}, c, z_{i+1} \dots, z_{j-1}, d, z_{j+1}, \dots\right) \\ &\neq u\left(\dots, z_{i-1}, a, z_{i+1}, \dots, z_{j-1}, d, z_{j+1}, \dots\right) + u\left(\dots, z_{i-1}, c, z_{i+1}, \dots, z_{j-1}, b, z_{j+1}, \dots\right) \end{aligned}$$

The utility function is (i, j)-pairwise additively separable if no (i, j)-violation exists.

Remark 2. If u is (i, j)-pairwise additively separable then

$$u(z) = \alpha(z_{-j}) + \beta(z_{-i}) \tag{4}$$

for some functions α, β.[18] If u is label-invariant, it is sufficient to verify that there exist no violations with $a = b$ and $c = d$.[19]

[18] To see this, set $a = z_i$, $b = z_j$, $c = d = \underline{z}$ in the inequality (now equality) used to define violations.

[19] To see this, set $z_i = a = b$ and $z_j = c = d$ in the same inequality (now equality). For a counterexample in the absence of label-invariance, consider $f(a, b) = a + b - \frac{1}{3}(a - b)^3$.

Since u is not additively separable, it must possess an (i, j)-pairwise violation of additive separability. By label-invariance and remark 2, there is an N-dimensional consequence z and a pair (a, b) such that

$$u^2(a, a) + u^2(b, b) \neq u^2(a, b) + u^2(b, a) \tag{5}$$

where we define $u^2 : \left[\underline{z}, \overline{z}\right] \to \mathbb{R}$ as follows:

$$u^2(x, y) := u\left(\ldots, z_{i-1}, x, z_{i+1}, \ldots, z_{j-1}, y, z_{j+1}, \ldots\right)$$

By label-invariance, u is not affected by rearranging the entries (x, y) and z. Therefore, this one (i, j)-pairwise violation easily leads to (k, l)-pairwise violation for dimensions k, l. This includes dimensions $k, l > N$ that exceed the dimensionality of the consequence z. Likewise, the function u^2 is not affected by the choice of (k, l). We can describe z, a, b in $O(N)$ time, evaluate $u^2(x, y)$ in $O(N)$ time by proposition 2, and rearrange dimensions in $O(n)$ time. Since N is a property of u and not affected by the dimension n of the instance H, these operations are all tractable.

The next two lemmas describe reductions from MAX2SAT and MIN2SAT to c, respectively, depending on the direction of inequality (5).

Lemma 4. *The choice correspondence c is NP-hard if*

$$u^2(a, a) + u^2(b, b) > u^2(a, b) + u^2(b, a) \tag{6}$$

Proof. I show that if the choice correspondence c^n were tractable, then we could solve MAX2SAT in polynomial time. First, recall that the objective for MAX2SAT is

$$\max_{x_1,\ldots,x_n} \sum_{j=1}^{m} \mathbf{1}(c_j = \mathrm{true}) \tag{7}$$

where

$$c_j = (x_{j1} \vee x_{j2}) = (x_{j1} \wedge x_{j2}) \vee (\neg x_{j1} \wedge x_{j2}) \vee (x_{j1} \wedge \neg x_{j2}) \tag{8}$$

Our goal is to construct a menu H such that, given the solution $c(H)$, we could solve program (7) in poly(n, m). For every clause c_j, create three states j_1, j_2, j_3 (i.e. create three isometric intervals used to define a regular lottery). For every variable x_i, create a dimension i. Let each submenu H_i consist of a two lotterys: h_i^T, which indicates $x_i = \mathrm{true}$, and h_i^F, which indicates $x_i = \mathrm{false}$. We defer the definition of these objects until later. For now, define a menu $H = H_1 \times \ldots \times H_n$ where an lottery h corresponds to an assignment x as described.

As written, the objective for BDT is

$$\max_{h \in H} \sum_{j=1}^{m} \sum_{k=1}^{3} u(h_1(j_k), \ldots, h_n(j_k))$$

Clearly, the following condition is sufficient for our purposes:

$$\sum_{k=1}^{3} u\left(h_1^{x_1}(j_k), \dots, h_n^{x_n}(j_k)\right) \propto \underbrace{\mathbf{1}\left((x_{j1} \wedge x_{j2}) \vee (\neg x_{j1} \wedge x_{j2}) \vee (x_{j1} \wedge \neg x_{j2})\right)}_{\mathbf{1}(c_j = \text{true})} \tag{9}$$

assuming that the proportionality constants are the same for all clauses j, which will turn out to be true for my construction.

Let us simplify this condition. On the left-hand side, we refer to all n variables whereas on the right-hand side we refer only to two variables. Let z be the consequence used to define u^2. As discussed earlier, we can rearrange the consequence as needed so that the arguments (x, y) to $u^2(x, y)$ correspond to dimensions $j1, j2$ of u. If $x_i \notin c_j$ and $\neg x_i \notin c_j$, set $h_i(j_1) = h_i(j_2) = h_i(j_3) = z_i$. Then, we can rewrite the left-hand side as

$$\sum_{k=1}^{3} u^2\left(h_{j1}^{x_{j1}}(j_k), h_{j2}^{x_{j2}}(j_k)\right)$$

At this point, we can focus on the problem where $n = 2$, $m = 1$. Without loss of generality, let $c_j = (x_1 \vee x_2)$. This will simplify our notation. The left-hand side of condition (9) becomes

$$u^2\left(h_1^{x_1}(1), h_2^{x_2}(1)\right) + u^2\left(h_1^{x_1}(2), h_2^{x_2}(2)\right) + u^2\left(h_1^{x_1}(3), h_2^{x_2}(3)\right) \tag{10}$$

while the right-hand side becomes

$$\mathbf{1}\left((x_1 \wedge x_2) \vee (\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2)\right) \tag{11}$$

The reader may have gathered that I intend to somehow associate $u\left(h_1^{x_1}(1), h_2^{x_2}(1)\right)$ with $(x_1 \wedge x_2)$, $u\left(h_1^{x_1}(2), h_2^{x_2}(2)\right)$ with $(\neg x_1 \wedge x_2)$, and $u^2\left(h_1^{x_1}(3), h_2^{x_2}(3)\right)$ with $(x_1 \wedge \neg x_2)$. Indeed, this is true. This would be immediate if, for example, $u(z_1, z_2) = \min\{z_1, z_2\}$ since we could set

$$h_1^T(1) = 1; \quad h_1^T(2) = 0; \quad h_1^T(3) = 1; \quad h_2^T(1) = 1; \quad h_2^T(2) = 1; \quad h_2^T(3) = 0$$

and $h_i^F(k) = 1 - h_i^T(k)$ for all i, k. In that case, $\min\left(h_1^{x_1}(1), h_2^{x_2}(1)\right) = \mathbf{1}(x_1 \wedge x_2)$ and so forth. Since only one of the three expressions in the disjunctive normal form (DNF) (8) can be true at one time, the sum (10) would be one if and only if (8) were true, and zero otherwise. In other words, (10) would equal (11) and we would be done.

The case for more general utility functions u is not quite so straightforward. Roughly, we can think of u as a function of two literals (e.g. (x_1, x_2), $(\neg x_1, x_2)$, $(x_1, \neg x_2)$). Since u is increasing, it will assign high utility when both variables are true and low utility when both variables are false. However, it will also assign medium utility when one variable is true and the other false. Essentially, this implies a three-valued logic where "medium" corresponds to a statement that is not quite true but also not quite false. This makes it difficult to map our problem onto MAX2SAT, which deals with a traditional two-valued logic.

To clear this hurdle, we rely on our supermodularity condition (6) and asymmetry across the states $k = 1, \dots, 3$. While it will not be true that $u\left(h_1^{x_1}(1), h_2^{x_2}(1)\right) \propto (x_1 \wedge x_2)$, our sufficient condition (9) will hold because it sums across these three states. Formally, define

$$h_1^T(1) = c; \quad h_1^T(2) = a; \quad h_1^T(3) = b; \quad h_2^T(1) = c; \quad h_2^T(2) = b; \quad h_2^T(3) = a$$

$$h_1^F(1) = a; \quad h_1^F(2) = b; \quad h_1^F(3) = a; \quad h_2^F(1) = a; \quad h_2^F(2) = a; \quad h_2^F(3) = b$$

By setting $b > c > a$, we devalue the satisfaction of the first expression in the DNF (8) relative to the latter two. Now, condition (9) is true if and only if

$$u\left(h_1^T(1), h_2^T(1)\right) + u\left(h_1^T(2), h_2^T(2)\right) + u\left(h_1^T(3), h_2^T(3)\right) = u(c, c) + u(a, b) + u(b, a) = B \tag{12}$$
$$u\left(h_1^T(1), h_2^F(1)\right) + u\left(h_1^T(2), h_2^F(2)\right) + u\left(h_1^T(3), h_2^F(3)\right) = u(c, a) + u(a, a) + u(b, b) = B \tag{13}$$
$$u\left(h_1^F(1), h_2^T(1)\right) + u\left(h_1^F(2), h_2^T(2)\right) + u\left(h_1^F(3), h_2^T(3)\right) = u(a, c) + u(b, b) + u(a, a) = B \tag{14}$$
$$u\left(h_1^F(1), h_2^F(1)\right) + u\left(h_1^F(2), h_2^F(2)\right) + u\left(h_1^F(3), h_2^F(3)\right) = u(a, a) + u(b, a) + u(a, b) = A \tag{15}$$

for some $B > A$. This is because the first three choices make the DNF (8) true, which demands a high value B, and the last one makes it false, which demands a low value A.

Conditions (13) and (14) are equivalent since u is symmetric across dimensions. Conditions (12) and (13) hold if and only if $\psi(c) = 0$ where

$$\psi(z) = u(z, z) - u(z, a) - u(a, a) - u(b, b) + 2u(a, b)$$

Note that $\psi(a) < 0$ and $\psi(b) \geq 0$ by assumption (6). Since u is continuous, it follows from the intermediate value theorem that there exists $c \in (a, b]$ such that $\psi(c) = 0$. Finally, the fact that $A < B$ follows from

$$u(a, a) + u(b, a) + u(a, b) < u(c, c) + u(a, b) + u(b, a) \iff u(a, a) < u(c, c)$$

which is true since $c > a$ and u is strictly increasing along the diagonal. This completes the reduction, since the consequences $h_i(j)$ can be defined similarly for other variables and states. Moreover, defining the menu H only requires us to define $O(nm)$ such consequences.

Notice that c depends on u^2 but not on any other aspect of the problem, including n, m, or the clauses $c_1, \dots, c_m$. This proof is non-constructive in the sense that I only prove the existence of a polynomial-time reduction (parameterized by c) from MAX2SAT to BDT, but do not provide an algorithm to find c itself. But whether c is easy or hard to compute is irrelevant for our purposes, so long as it does not need to be re-computed for different inputs to MAX2SAT. □

Lemma 5. *The choice correspondence c is NP-hard if*

$$u^2(a, a) + u^2(b, b) < u^2(a, b) + u^2(b, a) \tag{16}$$

Proof. I show that if the choice correspondence c^n were tractable, then we could solve MIN2SAT in polynomial time. First, recall that the objective for MIN2SAT is

$$\max_{x_1,\ldots,x_n} \sum_{j=1}^{m} \mathbf{1}(c_j = \text{false}) \tag{17}$$

Our goal is to construct a menu H such that, given the solution $c(H)$, we could solve program (17) in poly(n,m) time. My approach will be almost identical to the proof of lemma 4, so I skip ahead to the point of divergence.

Here, I define lotterys similarly to the previous lemma. However, because our objective is to minimize rather than maximize the number of satisfied clauses, we give false literals a high utility and true literals a low utility. Formally, define

$$h_1^T(1) = a; \quad h_1^T(2) = b; \quad h_1^T(3) = a; \quad h_2^T(1) = a; \quad h_2^T(2) = a; \quad h_2^T(3) = b$$

$$h_1^F(1) = c; \quad h_1^F(2) = a; \quad h_1^F(3) = b; \quad h_2^F(1) = c; \quad h_2^F(2) = b; \quad h_2^F(3) = a$$

Now, our (negatively proportional) analog to condition (9) is true if and only if

$$u\left(h_1^T(1), h_2^T(1)\right) + u\left(h_1^T(2), h_2^T(2)\right) + u\left(h_1^T(3), h_2^T(3)\right) = u(a,a) + u(b,a) + u(a,b) = A \tag{18}$$
$$u\left(h_1^T(1), h_2^F(1)\right) + u\left(h_1^T(2), h_2^F(2)\right) + u\left(h_1^T(3), h_2^F(3)\right) = u(a,c) + u(b,b) + u(a,a) = A \tag{19}$$
$$u\left(h_1^F(1), h_2^T(1)\right) + u\left(h_1^F(2), h_2^T(2)\right) + u\left(h_1^F(3), h_2^T(3)\right) = u(c,a) + u(a,a) + u(b,b) = A \tag{20}$$
$$u\left(h_1^F(1), h_2^F(1)\right) + u\left(h_1^F(2), h_2^F(2)\right) + u\left(h_1^F(3), h_2^F(3)\right) = u(c,c) + u(a,b) + u(b,a) = B \tag{21}$$

for some $B > A$. This is because the first three choices make the DNF (8) true, which demands a low value A, and the last one makes it false, which demands a high value B.

Conditions (19) and (20) are equivalent since u is symmetric across dimensions. Conditions (18) and (19) hold if and only if $\psi(c) = 0$ where

$$\psi(z) = 2u(b,a) - u(b,b) - u(a,a) + u(a,a) - u(a,z)$$

Note that $\psi(a) = 2u(b,a) - u(b,b) - u(a,a) > 0$ and $\psi(b) = u(b,a) - u(b,b) \leq 0$ by assumption (16). Since u is continuous, it follows from the intermediate value theorem that there exists $c \in (a,b]$ such that $\psi(c) = 0$. Finally, the fact that $B > A$ follows from

$$u(c,c) + u(a,b) + u(b,a) > u(a,a) + u(b,a) + u(a,b) \iff u(c,c) > u(a,a)$$

which is true since $c > a$ and u is strictly increasing along the diagonal. This completes the reduction, for the same reasons as in the previous lemma. $\square$

A.2 Proof of Theorem 2

Given the violation minor S_n of order k as advice, I want to show that an efficient algorithm for c can be used to solve MAX2SAT with k variables in $O(\text{poly}(n))$ time.

1. If $k = \Omega(\text{poly}(n))$, this is a polynomial-time algorithm for MAX2SAT.

2. If $k = \omega(\log(\text{poly}(n)))$, this is a subexponential-time algorithm for MAX2SAT. There is a well-known polynomial-time reduction from 3SAT with n' variables and m' clauses to MAX2SAT with $n' + m'$ variables and $10m'$ clauses. The description length of the 3SAT instance is $\Theta(n') + \Theta(m')$, which is the same order as the description length of the MAX2SAT instance. So, a subexponential-time algorithm for MAX2SAT implies the same for 3SAT.

Consider a boolean formula with k variables $x_1, \dots, x_k$, i.e. an instance of MAX2SAT. I will refer to this as the original formula. This terminology is meant to contrast with an auxilliary formula with variables $y_1, \dots, y_n$ that will describe an instance of weighted MAX2SAT. As we will see, the auxilliary problem is constructed in a way that its solution corresponds to the solution of the original problem. Then, we will reduce the auxilliary problem to SEU maximization.

Suppose dimensions $d_1, \dots, d_l$, corresponding to nodes of $G(u, n)$, be combined via edge contractions in the contracted node d of S_n. In the auxilliary problem, we wish to impose the constraint $y_{d_i} = y_{d_j}$ for all $i, j = 1, \dots, l$. This allows us to treat the contracted node d as a single variable in the original formula, where any dimension d_i can represent the variable d. This is useful because, for a given dimension j, some dimensions in d may share a pairwise violation (i.e. an edge in $G(u, n)$) with j, but not others.

Fortunately, it is possible to impose the constraint $x = y$ by adding clauses $x \vee \neg y$ and $\neg x \vee y$ to the auxilliary instance, with weight exceeding twice the total weight of all other clauses where x, y are represented (excepting other clauses representing equality constraints). The reason is that setting $x = y$, regardless of whether the value is true or false, will make both of these clauses true. In contrast, setting $x \neq y$ will make one of these clauses false. Given the weights, no benefit from setting $x \neq y$ that comes from other clauses can outweigh this advantage.

Since S_n is a complete graph, every contracted node d is adjacent to every other contracted node d'. To represent a clause $x \vee y$ in the original formula, it suffices to choose some node in d representing x and some node in d' representing y. These nodes, corresponding to dimensions of u, must have a pairwise violation in order for the reduction to work. Equivalently, they must share an edge in $G(u, n)$. Of course, it is always possible to find such a pair. If no such pair existed, there would be no sequence of edge contractions in $G(u, n)$ that would make d adjacent to d'.

Having described the auxilliary problem, it remains to construct a menu such that SEU maximization corresponds to solving weighted MAXSAT. Essentially, we want to recreate the argument that we used in lemmas 4 and 5. There, we observed that the following condition (9)

$$\sum_{l=1}^{3} u\left(h_1^{y_1}(j_l), \dots, h_n^{y_n}(j_l)\right) \propto \underbrace{\mathbf{1}\left((y_{j1} \wedge y_{j2}) \vee (\neg y_{j1} \wedge y_{j2}) \vee (y_{j1} \wedge \neg y_{j2})\right)}_{\mathbf{1}(c_j = \text{true})}$$

is nearly sufficient for SEU with submenus $H_i = \{h_i^T, h_i^F\}$ to correspond to an optimal assignment in MAX2SAT. The qualifier “nearly” reflects the fact that we also need the proportionality constants to be the same across all clauses $y_{j1} \vee y_{j2}$. In the case of weighted MAXSAT, these constants should reflect the weight of that clause.

Consider a clause $y_i \vee y_j$. As in the proof of theorem 2, let (z, a_0, b_0, a_1, b_1) constitute an (i, j)-pairwise violation of additive separability, and define

$$u^2(a, b) := u\left(\dots, z_{i-1}, a, z_{i+1}, \dots, z_{j-1}, b, z_{j+1}, \dots\right)$$

However, without label invariance, u^2 is not the same across all dimensions i, j. For this reason, we need each (i, j)-pairwise violation to be included in the description of S_n, which we take as advice. This description takes $O(n^3)$ space. In any case, we are focusing on a single clause, so I will suppress the dependence on i, j. As before (see lemma 4), we construct lotteries so that condition (9) can be rewritten in terms of only two variables, i.e.

$$\sum_{l=1}^{3} u^2\left(h_1^{y_l}(l), h_2^{y_l}(l)\right) \propto \mathbf{1}\left((y_1 \wedge y_2) \vee (\neg y_1 \wedge y_2) \vee (y_1 \wedge \neg y_2)\right) \tag{22}$$

As before, there are two cases to consider, based on the direction of the (i, j)-pairwise violation. We consider each in turn. The next two observations will imply that we can satisfy the above condition, and then we complete the definition of subacts h_i^T, h_i^F in the same way as theorem 2.

Assume without loss of generality that $b_1 > a_1$ and $b_0 > a_0$. Suppose that

$$u^2(b_0, a_1) + u^2(a_0, b_1) < u^2(b_0, b_1) + u^2(a_0, a_1) \tag{23}$$

I claim that there exist constants $B > A$ and weights $\alpha \in \Delta\{1, 2, 3\}$ such that

$$\alpha_1 u^2(b_0, b_1) + \alpha_2 u^2(b_0, a_1) + \alpha_3 u^2(a_0, b_1) = B \tag{24}$$
$$\alpha_1 u^2(b_0, a_1) + \alpha_2 u^2(b_0, b_1) + \alpha_3 u^2(a_0, a_1) = B \tag{25}$$
$$\alpha_1 u^2(a_0, b_1) + \alpha_2 u^2(a_0, a_1) + \alpha_3 u^2(b_0, b_1) = B \tag{26}$$
$$\alpha_1 u^2(a_0, a_1) + \alpha_2 u^2(a_0, b_1) + \alpha_3 u^2(b_0, a_1) = A \tag{27}$$

To see this, note that (24) exceeds (27) by the fact that u^2 is strictly increasing (inherited from monotonicity of u). Therefore, we only need to satisfy the first three equations. The fact that a strictly positive solution exists follows from (23).

These weights α represent the likelihood of the underlying states. Recall that the effective state corresponds to a subinterval of $\Theta = [0, 1]$ on which the lotteries $h \in H$ are constant. Consider the subintervals corresponding to the three states associated with clause $y_1 \vee y_2$. These three subinterval should have length proportional to

$$\frac{\alpha_1}{\alpha_1 + \alpha_2 + \alpha_3}$$

Next, consider the case

$$u^2(b_0, a_1) + u^2(a_0, b_1) > u^2(b_0, b_1) + u^2(a_0, a_1) \tag{28}$$

The previous observation no longer holds, since a solution α is not guaranteed to exist. On the other hand, there do exist constants $B > A$ and weights $\beta \in \Delta\{1, 2, 3\}$ such that

$$\beta_1 u^2(b_0, b_1) + \beta_2 u^2(b_0, a_1) + \beta_3 u^2(a_0, b_1) = A \tag{29}$$
$$\beta_1 u^2(b_0, a_1) + \beta_2 u^2(b_0, b_1) + \beta_3 u^2(a_0, a_1) = A \tag{30}$$
$$\beta_1 u^2(a_0, b_1) + \beta_2 u^2(a_0, a_1) + \beta_3 u^2(b_0, b_1) = A \tag{31}$$
$$\beta_1 u^2(a_0, a_1) + \beta_2 u^2(a_0, b_1) + \beta_3 u^2(b_0, a_1) = B \tag{32}$$

We used a similar observation in lemma 5 to obtain a reduction from MIN2SAT. Unfortunately, without label-invariance, u^2 depends on the dimension pair i, j; condition (28) may apply to some pairs while condition (23) applies to other pairs. So, we need to stick with MAX2SAT.

A straightforward way to handle this case is to modify the auxilliary formula. Replace the original clause $y_1 \vee y_2$ with three new clauses: $\neg y_1 \vee \neg y_2$, $\neg y_1 \vee y_2$, and $y_1 \vee \neg y_2$. The original clause is satisfied iff only two of the new clauses are satisfied. It is falsified iff all three of the new clauses are satisfied. So, instead of trying to satisfy the original clause, we can aim to falsify as many of the new clauses as possible. The way to interpret equations (29) through (32) is that, with effective states that are proportional to β, SEU conditioned on those states is proportional to the falsification of a clause. This is exactly what we need.

At this point, our construction satisfies (22). The only remaining issue is that, unlike in the proof of theorem 2, the proportionality constants will differ from clause to clause. Furthermore, some of the clauses are weighted. This can be handled in a straightforward way by manipulating the intervals used to define the effective state space. Suppose a clause receives weight w and the difference between its satisfied value and its unsatisfied value is Δ. Then the three effective states associated with said clause should have interval length proportional to w/Δ. Integrating over states in the sense of SEU gives precisely the objective function of weighted MAX2SAT, so c gives a solution to the auxilliary problem.

A.3 Proof of Theorem 3

Before defining the dynamic choice bracketing algorithm, it is useful to review the (much simpler) narrow choice bracketing algorithm.

Definition 16. *The* narrow choice bracketing (NCB) algorithm *is parameterized by some function $u : Z \to \mathbb{R}$. Given a product menu L, the NCB algorithm proceeds as follows:*

1. *Initialize the lottery $\hat{l}$ by putting probability one on the zero vector $\vec{0} \in Z$.*

2. *Iterate $i = 1, \dots, n$.*

(a) Redefine

$$\hat{l}_i \in \arg\max_{l_i \in L_i} \mathrm{E}\left[u\left(0, \dots, 0, l_i, 0, \dots, 0\right)\right]$$

3. Return act $\hat{l}$, which belongs to the menu L by construction.

Definition 17 (DCB Algorithm)**.** *Specify a utility function u and a graph $G = (V, E)$ of order n and contraction degeneracy k. Given an n-dimensional product menu H, the* dynamic programming *(DP) algorithm proceeds as follows:*

1. *Construct a directed acyclic graph $\tilde{G} = (V, \tilde{E})$ where each node has out-degree at most k.*
2. *Perform a topological sort of $\tilde{G}$. In particular, nodes are numbered, i.e. $V = \{1, \dots, n\}$, and if $i < j$ then there is no path in $\tilde{G}$ from node j to node i.*
3. *Initialize the lottery $\tilde{h}$ by setting $\tilde{h}_i(\theta) = \underline{z}$ for all dimensions i and states θ.*
4. *Let i be the smallest node in $\tilde{G}$. Initialize the frontier $F \subseteq V$ as $\emptyset$.*
   - *(a) Let S_i consist of i's successor nodes j, i.e. where $(i, j) \in \tilde{E}$. Let F_i be the set of nodes $j \in F$ for which, in the undirected graph G, there exists a path from i to j that does not pass through F. I will refer to the pair (i, j) as $(G \setminus F)$-connected.*
     - *i. If $|F_i| > k$, return to step (4a) with $i := i + 1$.*
     - *ii. Else let $L_i = \prod_{j \in F_i \cup S_i} H_i$ be a set of partially-specified lotteries $h_{F_i \cup S_i}$.*
   - *(b) Initialize the lottery $\tilde{h}$ by setting $\tilde{h}_i(\theta) = \underline{z}$ for all dimensions i and states θ.*
     item In this step, we construct a function $f_i : L_i \to H_i$ where

$$f_i\left(h_{F_i \cup S_i}\right) \in \arg\max_{h_i \in H_i} \int_0^1 u\left(h_i(x), \hat{h}_{-i}(x)\right) dx$$

     To define it, iterate over all $h_{F_i \cup S_i} \in L_i$.
     - *i. Initialize the lottery $\hat{h} = \tilde{h}$ but set $\hat{h}_{F_i \cup S_i}(x) = h_{F_i \cup S_i}(x)$.*
     - *ii. Iterate through $j \notin F_i \cup S_i \cup \{i\}$.*
       - *A. If $j > i$ then $\hat{h}_j(\theta) = \underline{z}$ for all states θ.*
       - *B. Else if i, j are not $(G \setminus F)$-connected, set $\hat{h}_j(\theta) = \underline{z}$ for all θ.*
       - *C. Else proceed as follows. Because $i \leq j$ and $j \notin F_i$, we have already constructed f_j. Assume that f_j is up-to-date, in that it only depends on subacts h_l for dimensions $l \in F_i \cup \{i\}$. Define $\hat{h}_j = f_j\left(h_{F_i}\right)$.*
   - *(c) If $F_i \cup S_i = \emptyset$, set $\tilde{h}_i = f_i(\emptyset)$.*
   - *(d) Iterate through $j \notin F_i \cup S_i \cup \{i\}$.*
     - *i. If $j > i$, do nothing.*
     - *ii. Else if i, j are not $(G \setminus F)$-connected, do nothing.*

iii. *Else update f_j as follows. Currently, f_j is a function of h'_i and h'_{F_i}. Replace the argument h'_i with $f_i\left(h'_{S_i}, h'_{F_i}\right)$. Now, reconstruct f_j as a function of $h'_{S_i \cup F_i}$. Moreover, if $F_i \cup S_i = \emptyset$, set $\tilde{h}_j = f_j(\emptyset)$.*

(e) *Redefine F with node i deleted and nodes S_i added. Redefine $\tilde{G}$ with node i deleted. If any nodes remain, repeat step (3).*

5. *Return lottery $\tilde{h}$, which belongs to the menu H by construction.*

The following is a fixed-parameter tractability result that combines usefully with the log-polynomial rates established in theorem 3.

Proposition 3. *Given an n-dimensional product menu with submenus of size l and an effective state space of size m, the DP algorithm parameterized by graph G will halt in $O(\text{poly}(n, m, l^k))$ time, where k is the contraction degeneracy of G.*

Proof. Most of these steps are obviously polynomial in n, k, l or involve iterating over sets of size l^k. Here, I will justify the less obvious steps and assumptions. The key observation is the third one.

1. Step 1 takes $O(\text{poly}(n))$ time since G has degeneracy k. The algorithm is simple: find a node i with degree $\leq k$, orient all the edges outwards. In the subgraph without i, repeat.

2. Step 4 will halt in $O(n^2)$ steps. Whenever step 4.e is reached, the number of nodes in $\tilde{G}$ decreases by one, so this can be reached at most n times. The only other possibility is that step 4.a.i is reached, but this can happen at most n consecutive times before it fails.

3. Step 4.a.i will never fail. This is due to the following observation. Consider the minor S of graph G with all edges contracted except for those connecting two nodes $i, j \in F$. Then there is an edge between i and j if and only if they are adjacent in G or if there is a path from i to j that does not go through F. Note that S has a node of degree less than or equal to k, by the definition of contraction degeneracy. This node contains some $j \in F$. Step 4.a.i will eventually reach this j and move on to step 4.a.ii.

4. Step 4.b.ii.C assumes that the function f_j is up-to-date. This is guaranteed by step 4.d, which updates any function f_j that may have been affected by changes in iteration i of step 4.

5. Step 5 assumes that $\tilde{h}$ belongs to the menu H. This follows from the fact that $F = \emptyset$ by the time step 4 terminates. Since step 4.d ensures that functions are up-to-date, every function f_j must have reached a point where its argument is vacuous. At that point, $\tilde{h}_j$ would have been defined in steps 4.c or 4.d.iii.

□

A.4 Proof of Lemma 1

Let $c \in C_R$ have revealed utility function u and inseparability graph $G(u, n)$. Let $k = 1 + \mathrm{dgn}(G(u, n))$. I will argue that if $\bar{u}^n(\bar{z}, \dots, \bar{z}) = o(n/k)$, then c's approximation ratio is $o(1)$, i.e. it is bounded above by a term that is approaching zero. In contrast, the greedy algorithm obtains a ½-approximation by proposition 1. Therefore, the ratio of the approximation ratios is $\omega(1)$, i.e. it is bounded below by a term that is approaching infinity. The lemma follows immediately, since $k = O(\log n)$ by theorem 2 (which establishes that u must be Hadwiger separable) and proposition **??** (which shows that Hadwiger separability implies that u is degenerate).

To prove this, I construct a menu H where c obtains a vanishingly small fraction of the optimal SEU. In doing so, I use one additional concept from graph theory.

Definition 18 (Chromatic Number). *Let C be a set of colors. Let $G = (V, E)$ be a graph. A* C-coloring *of G is an assignment $f : V \to C$ of nodes $i \in V$ to colors $c \in C$ where adjacent nodes have different colors, i.e. $(i, j) \in E$ implies $f(i) \neq f(j)$. The* chromatic number *of G is the size of the smallest set C such that a C-coloring exists.*

The chromatic number of $G(u, n)$ is at most k. This follows from the fact that a graph $G = (V, E)$ of degeneracy d can be colored with $d + 1$ colors (Szekeres and Wilf 1968). Let $C = \{1, \dots, k\}$ and suppose we color $G(u, n)$ with C. Choose the color $c \in C$ with the most nodes, a number that must be at least n/k. Let V_c denote the set of c-colored nodes.

Next, I construct a product menu H. For all dimensions $i \notin V_c$, set $H_i = \{h_i\}$ where $h_i(\theta) = e_i$ for all states θ. For all dimensions $i \in V_c$, set $H_i = \{h_i^T, h_i^F\}$. Consider an evenly-spaced grid $X = \{0, x_1, \dots, x_{k+1}, 1\}$ on $[0, 1]$. Let $h^T(\theta) = \bar{z}$ if $\theta \geq x_k$ and $h^T(\theta) = \underline{z}$ otherwise. Let $h^T(\theta) = \bar{z}$ if $\theta \in [x_{i-1}, x_i]$ and $h^T(\theta) = \underline{z}$ otherwise.

Note that the consequences z_i for $i \notin V_c$ are fixed. Moreover, as a function of z_i for $i \in V_c$, u is additively separable. This follows from the fact that two nodes i, j with the same color c cannot be adjacent in the inseparability graph. As a consequence, the agent will always choose h_i^T over h_i^F. After all, the former obtains the high value $\bar{z}$ over two effective states (i.e. an interval of length $2/(k + 1)$) and $\underline{z}$ otherwise. Whereas the latter obtains the high value $\bar{z}$ only in one effective state (i.e. an interval of length $1/(k + 1)$). Essentially, we have constructed a menu in which the agent will narrowly choice bracket, myopically seeking the highest value in each dimension.

Unfortunately, from the perspective of $\bar{u}$, always choosing h_i^T can be a poor strategy, roughly due to satiation effects. This strategy will obtain $\bar{z}$ in every dimension but only for two effective states, for a SEU of

$$\frac{|V_c| - 2}{|V_c|} \cdot \bar{u}\left(\underline{z}, \dots, \underline{z}\right) + \frac{2}{|V_c|} \bar{u}\left(\bar{z}, \dots, \bar{z}\right)$$

In contrast, always choosing h_i^F guarantees $\bar{z}$ in one dimension for every state, for a SEU of

$$|V_c| \cdot \bar{u}\left(\bar{z}, \underline{z}, \dots, \underline{z}\right)$$

Recall that $|V_c| \geq n/k$. The latter SEU is increasing linearly in $|V_c|$, while the former SEU is increasing sublinearly, assuming $\bar{u}\left(\bar{z}, \dots, \bar{z}\right) = o(n/k)$. This is what I sought to show.

A.5 Proof of Lemma 2

First, I describe a greedy algorithm. This can be thought of as a modified version of the NCB algorithm. Instead of optimizing in each dimension myopically with respect to the her beliefs, the agent allows her beliefs to change across dimensions. The agent puts less probability on states where the agent has already secured good outcomes by her choices in previous dimensions. Thus, the agent's behavior is not consistent with SEU maximization for a fixed belief. This generalizes Johnson's (1974) approximation algorithm for MAXSAT.

Definition 19 (Greedy Algorithm)**.** *Specify a utility function u. Given a product menu H, the algorithm proceeds as follows:*

1. *Initialize the lottery $\hat{h}$ by setting $\hat{h}_i(\theta) = \underline{z}$ for all dimensions i and states θ.*
2. *Iterate $i = 1, \dots, n$.*
   (a) *Redefine*
   $$\hat{h}_i \in \arg\max_{h_i \in H_i} \int_0^1 u\left(h_i(x), \hat{h}_{-i}(x)\right) dx$$
3. *Return lottery $\hat{h}$, which belongs to the menu H by construction.*

Let c_G be the choice correspondence associated with the greedy algorithm. Given a product menu $H \in \mathcal{H}$, let $\hat{h} = c_G(H)$. For each dimension i and state x, define

$$\begin{aligned}
\Delta_i(x) &= \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_i(x), \underline{z}, \dots, \underline{z}\right) - \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_{i-1}(x), \underline{z}, \dots, \underline{z}\right) \\
&\geq \bar{u}^n\left(\hat{h}_1(x), \dots, h_i(x), \underline{z}, \dots, \underline{z}\right) - \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_{i-1}(x), \underline{z}, \dots, \underline{z}\right) \\
&\geq \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_{i-1}(x), h_i(x), \dots, h_n(x)\right) - \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_{i-1}(x), \underline{z}, h_{i+1}(x), \dots, h_n(x)\right) \\
&\geq \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_{i-1}(x), h_i(x), \dots, h_n(x)\right) - \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_i(x), h_{i+1}(x), \dots, h_n(x)\right) \qquad (33)
\end{aligned}$$

where the first inequality follows from the the greedy hypothesis, the second follows from the SCC with

$$z'' = (0, \dots, 0, \hat{h}_{i+1}(x) - \underline{z}, \dots, \hat{h}_n(x) - \underline{z})$$

and the third follows from monotonicity. Define

$$\Delta_i = \int_0^1 \Delta_i(x) dx$$

This can be interpreted as the added value from choosing $\hat{h}_i$, as evaluated by the greedy algorithm at step i. We want to compare this to the added value from choosing $\hat{h}_i$, having commited to the greedy choices in dimensions $1, \dots, i-1$, but otherwise choosing optimally. Define

$$\text{OPT}_i = \max_{h_i, \dots, h_n} \int_0^1 \bar{u}^n\left(\hat{h}_1(x), \dots, \hat{h}_{i-1}(x), h_i(x), \dots, h_n(x)\right) dx$$

It follows from inequality (33) that

$$\mathrm{OPT}_i \leq \mathrm{OPT}_{i+1} + \Delta_i \quad \text{and} \quad \mathrm{OPT}_1 \leq \mathrm{OPT}_{n+1} + \sum_{i=1}^{n} \Delta_i$$

Observe that OPT_1 is the true optimum, obtained by $\bar{c}$. Whereas OPT_{n+1} is the payoff of the greedy algorithm. It is easy to see that the greedy algorithm obtains SEU $\sum_{i=1}^{n} \Delta_i$. Therefore, c_G is a $1/2$-approximation to $\bar{c}$.