On the Hardness of Algorithmic Bayesian Persuasion with Succinctly Described Inputs

Chirag Pabbaraju[1 [0000−0002−3424−691X]]

Stanford University, Stanford CA 94305, USA
cpabbara@stanford.edu

Abstract. We consider the algorithmic problem of implementing the optimal signaling scheme in Bayesian persuasion, where a sender wants to sample efficiently from a conditional distribution over signals given a state of nature, with a view to persuade a receiver to take an action that maximizes the sender's payoff. Dughmi and Xu (2016) give an efficient algorithm for this task in the special case where the payoffs are i.i.d. Notably, the input model for this result assumes that all the relevant payoffs and distribution parameters are specified explicitly, which might either be prohibitive or redundant with large action spaces.
In this work, we study the tractability of implementing the optimal scheme in Bayesian persuasion when all the payoffs and distribution parameters are not given explicitly, but are instead given implicitly via circuits computing them; correspondingly, we seek algorithms that have a running time polynomial in the description of the instance. In this *succinct input model*, we find, perhaps interestingly, that implementing the optimal signaling scheme becomes computationally very hard — namely, if an efficient algorithm were to exist even for the special class of instances having i.i.d. payoffs, then the polynomial hierarchy collapses. Our result adopts proof ideas from the seminal work of Aaronson and Arkhipov (2011) that showed classical hardness of the quantum task of Boson sampling.

Keywords: Bayesian Persuasion · Computational Hardness

1 Introduction

Imagine that a financial advisor is being consulted by a client about whether they should invest in stock A or stock B, and the advisor has access to some confidential information about the stocks that the client does not have. For example, the advisor and client might both have a common prior on the "state" of the market, but off the record, the advisor may additionally also know the *realization* of the state, leading to a certain *information asymmetry*. Now suppose that the client really only benefits if they buy a stock that will make a profit in the long term. However, the advisor is up for a performance assessment soon, and stands to benefit if they can get the client to buy a stock that yields high profit in the short term even if it does not gain in the long term, since that would reflect well on their record. In such scenarios with strategic incentives and asymmetric information, it turns out that if the advisor plays it right, they can succeed in persuading the client to buy a stock *even* when the (realized) market state suggests no long term gains — this is regardless of the computational resources of the client, and assuming also that the

client is a rational agent looking to optimize their utility given all the information that they have. An extremely clean model to study the scope of such strategic persuasion is that of *Bayesian persuasion*, proposed by Kamenica and Gentzkow (2011) [22]. We formally describe the model ahead, but refer the reader to the insightful survey by [12] for motivating examples and a discussion of persuasion viewed through the computational lens, as well as the surveys by [5,21] for a broader perspective of the phenomenon.

Bayesian Persuasion. The setup comprises of a *sender*, who is trying to persuade a *receiver* to take an *action* that is most beneficial to the sender. Namely, there is a set of n actions that the receiver can choose from, which we will denote by the set $[n] := \{1, 2, \ldots, n\}$. Both the sender and the receiver have knowledge of a common prior distribution D over the state of nature $\theta \in \Theta$. Associated with each possible state θ and action $i \in [n]$ is a receiver payoff $r(\theta, i)$ as well as a sender payoff $s(\theta, i)$. These payoff functions are known to both the sender and the receiver. Now, a state $\theta \sim D$ is realized from the prior D, but is revealed *only to the sender* — crucially, the realized value of θ is unknown to the receiver, and this is where information asymmetry arises. Based on the realization of θ, the sender sends a "signal" σ to the receiver, by drawing σ from a probability distribution $\psi(\theta, \cdot)$ over an abstract set of signals Σ. Here, $\psi : \Theta \times \Sigma \to [0, 1]$ is known as the "signaling scheme", and satisfies $\sum_\sigma \psi(\theta, \sigma) = 1$ for every fixed θ. The setup assumes that the receiver has complete knowledge about the signaling scheme ψ as well, in that the sender *commits* to it beforehand (i.e., before the state θ is realized). Thus, the only information that the receiver does not have is the realization of θ — the sender's signal σ is all the indication they get about it.

Nevertheless, upon receiving the signal σ, knowing ψ and the prior D from which θ was drawn, the receiver computes, for every action $i \in [n]$, the expected posterior payoff $T(i, \sigma)$ they would get if they take action i — this is precisely $T(i, \sigma) = \mathbb{E}_{\theta \sim D(\cdot|\sigma)}[r(\theta, i)]$, where $D(\theta \mid \sigma) \propto D(\theta)\psi(\theta, \sigma)$. The receiver chooses the action which maximizes this posterior payoff[1], i.e., $i^\star(\sigma) \in \arg\max_{i \in [n]} T(i, \sigma)$, and derives payoff $r(\theta, i^\star(\sigma))$. The sender then correspondingly gets payoff $s(\theta, i^\star(\sigma))$.

Given this rational behavior of the receiver, the sender's task is to choose a strategic signaling scheme ψ with a view to maximize their own expected payoff. Concretely, the sender wishes to compute a ψ that maximizes $\mathbb{E}_{\theta \sim D}\, \mathbb{E}_{\sigma \sim \psi(\theta, \cdot)}[s(\theta, i^\star(\sigma))]$. Alternatively, one can consider the seemingly easier task of merely *implementing the optimal scheme*; that is, given a state realization θ, the sender wishes to sample $\sigma \sim \psi(\theta, \cdot)$ from the scheme ψ that maximizes $\mathbb{E}_{\theta \sim D}\, \mathbb{E}_{\sigma \sim \psi(\theta, \cdot)}[s(\theta, i^\star(\sigma))]$.

The Algorithmic Task. For the setup above, it turns out, via an argument similar to the *revelation principle* in mechanism design [22], that there always exists an optimal signaling scheme which uses a set of signals *equal* to the set of actions (i.e., $\Sigma = [n]$). Moreover, this scheme is *persuasive*: it has the property that whenever the sender signals any $i \in [n]$, the receiver, upon calculating their posterior payoffs, finds that the posterior payoff of taking action i is no less than the posterior payoff of taking any other action $j \neq i$. Thus, it is in the receiver's best interest to simply follow the sender's signal. Given this characterization of an optimal scheme, the task faced by the sender for computing an optimal scheme can be expressed in the form of the following linear program:

[1] As per convention, ties between receiver-optimal actions are broken in favor of the sender.

Bayesian Persuasion Linear Program

$$\begin{aligned}
\max_{\psi(\theta,i)} \quad & \sum_{\theta} D(\theta) \sum_{i\in[n]} \psi(\theta,i)s(\theta,i) \\
\text{subject to} \quad & \sum_{\theta} D(\theta)\psi(\theta,i)r(\theta,i) \geq \sum_{\theta} D(\theta)\psi(\theta,i)r(\theta,j) \quad \text{for all } i,j \in [n], \\
& \psi(\theta,i) \geq 0 \quad \text{for all } \theta \in \Theta, i \in [n], \\
& \sum_{i\in[n]} \psi(\theta,i) = 1 \quad \text{for all } \theta \in \Theta. \qquad (1)
\end{aligned}$$

The number of variables in this linear program is $|\Theta| \cdot n$ — one for each value of $\psi(\theta,i)$ — and the number of constraints is $\text{poly}(|\Theta|,n)$. The sender can thus compute the optimal scheme by solving this linear program in $\text{poly}(|\Theta|,n)$ time, provided they know all the relevant quantities (namely the payoffs, and the prior distribution D).

Structured Distributions. The work of Dughmi and Xu (2016) [14], among other things, studies the computational aspects of the Bayesian persuasion problem for special classes of structured distributions D. In their viewpoint, one identifies the state of nature rather with a realized *action type* for each action. Namely, imagine that the payoffs that the sender and receiver get when the receiver takes a certain action i are not fixed, but instead depend on the action's *type*, which is drawn from a distribution over action types, independently of the types of the other actions. In addition, the payoffs for action i depend *only* on its realized type, and not the realized types of the other actions.

Formally, suppose that each of the n actions can have one of m types, and let $\mathcal{P}$ be a distribution over $[m]$. The state of nature $\theta = (\theta_1, \ldots, \theta_n)$ is realized by drawing each $\theta_i \sim \mathcal{P}$ *independently*. When the receiver takes action i, they get payoff $r(\theta_i)$, and the sender gets payoff $s(\theta_i)$. So, the payoffs depend only on the *type* of the action taken, and the payoffs for each type are *identical* across the different actions. We will denote this setting as that of "i.i.d. payoffs". One can also consider the setting where the payoffs are not identical, but merely independent. Here, the type of each action i is drawn independently from a distribution $\mathcal{P}_i$ over $[m]$, and the state θ is realized by drawing each $\theta_i \sim \mathcal{P}_i$ independently. When the receiver takes action i, they get payoff $r(\theta_i, i)$, and the sender gets payoff $s(\theta_i, i)$. So, the payoffs depend on *both* the action as well as its type: we denote this setting as "independent but non-identical payoffs".

Importantly, for both these settings, [14] assume that the input *explicitly* comprises of all the relevant quantities. For the i.i.d. setting, this constitutes n, m, the distribution $\mathcal{P} = (\mathcal{P}(1), \ldots, \mathcal{P}(m))$ over action types, the receiver payoffs $r(1), \ldots, r(m)$ and the sender payoffs $s(1), \ldots, s(m)$. For the independent but non-identical setting, this constitutes n, m, the distributions $\mathcal{P}_1, \ldots, \mathcal{P}_n$ over action types for each action (each specified as a vector of m probabilities), the receiver payoffs $\{r(j,i)\}_{j\in[m],i\in[n]}$ and the sender payoffs $\{s(j,i)\}_{j\in[m],i\in[n]}$. In either case, the description of the input has size $\text{poly}(n,m)$ bits, assuming polynomial bit complexities of the probabilities and payoffs.

The typical benchmark for an algorithm to be efficient is a polynomial running time in the size of the input description — for the above, this would be $\text{poly}(n,m)$ time. However, in both the settings above, note that the size of the state space is m^n, and

so, naively solving the linear program (1) above for obtaining the optimal persuasive scheme requires $\exp(n)$ time. Nevertheless, for the setting with i.i.d. payoffs, [14] give an algorithm that can (1) implement an optimal persuasive scheme, as well as (2) compute the optimal sender payoff (i.e., the optimal objective value of the linear program), in $\text{poly}(n, m)$ time. This algorithm exploits the underlying symmetry of the action payoffs, together with a connection to Border's characterization of reduced-form auctions [7,8], to instead efficiently optimize over a smaller-sized variable. On the other hand, for the setting with independent but non-identical payoffs, [14] show that it is #P-hard in general to exactly compute the optimal sender payoff. The question of whether it is possible to efficiently implement the optimal persuasive scheme in this setting remains open.

While most of the literature on the computational aspects of the Bayesian persuasion problem typically assumes the model of explicitly stated inputs above, many instances allow for the input to be described in a much more compact (yet complete) manner. As a simple example, consider an i.i.d. instance where the action type can be described as a t-bit string in $\{0,1\}^t$, the distribution $\mathcal{P}$ over action types is uniform, and all (or nearly all) of the payoffs are constant. In this case, explicitly specifying all the $\exp(t)$ relevant parameters of the instance in the input is prohibitive and redundant; furthermore, allowing an $\exp(t)$ running time budget seems unjustified, given that the instance permits a succinct $\text{poly}(t)$-sized representation that still captures its entire essence. More generally, one can imagine instances that have a simple distribution over action types, and where the payoffs can be computed by simple-to-describe functions having $\text{poly}(t)$-sized representations. In such cases, it is natural to wonder if there are algorithms for implementing the optimal persuasive scheme that run in $\text{poly}(t)$ time, even though the number of action types $m = \exp(t)$. This brings us to our main question:

For Bayesian persuasion instances that can be succinctly described, is there an algorithm that implements the optimal persuasive scheme for the sender and runs in time polynomial in the size of the input description?

The Succinct Input Model. We will, in fact, even restrict our attention to the special class of instances that have i.i.d. payoffs, and find interesting results. We now formally define how such a Bayesian persuasion instance $\mathcal{I}$ is described in the *succinct input model*.

Definition 1 (Succinct Input Model). *Let $\mathcal{I}$ be a Bayesian persuasion instance with i.i.d. payoffs having n actions and m action types. Let $cr : [m] \to \mathbb{R}$ and $cs : [m] \to \mathbb{R}$ be circuits that compute the receiver and sender payoff functions r and s respectively. Similarly, let $c\mathcal{P} : [m] \to [0,1]$ be a circuit that computes the probability under $\mathcal{P}$ of each action type, and let $cSample\mathcal{P} : \{0,1\}^* \to [m]$ be a circuit that samples an action type from $\mathcal{P}$; that is, $cSample\mathcal{P}$ takes as input a fixed finite number of random bits and outputs an action type. Then, the description of $\mathcal{I}$ in the succinct input model comprises of an encoding of n, m, the circuits computing the receiver and sender payoffs, and the circuits computing the probabilities and sampling from the distribution over action types. Concretely, $\mathcal{I} := \langle n, m, cr, cs, c\mathcal{P}, cSample\mathcal{P} \rangle$.*

The corresponding benchmark for an algorithm to be efficient in this succinct input model is a polynomial running time in the description $|\mathcal{I}|$ (and any other problem inputs). We emphasize that the succinct input model is not artificially compressing the

input description and unjustly demanding algorithms with a stricter guarantee. Indeed, for certain instances, it may very well be the case that the smallest circuits that compute the receiver and sender payoffs (or the circuits for the associated distributions) are as large as the explicit descriptions of these quantities. In these cases, there isn't much structure in the parameters of the instance, and we are allowing the same running time budget as that in the explicit input model. It is only for those instances that have a simple-to-capture structure in their parameters that we are non-trivially compressing the input, and correspondingly demanding faster algorithms for. Therefore, the benchmark for efficiency is commensurate with the structure present in the instance.

1.1 Main Result

Recall that in the explicit input model, for instances with i.i.d. payoffs, [14] give an algorithm that can do both, implement the optimal persuasive scheme (i.e., sample from the optimal scheme given a state realization θ), as well as compute the optimal sender payoff, in time polynomial in the input description size. For instances with independent but non-identical payoffs, [14] only show #P-hardness for exactly computing the optimal scheme, whereas little-to-no progress has been made on settling the computational complexity of the easier-looking task of implementing the optimal scheme.

With succinctly described inputs, it is not too hard to see (Section 2) that computing the optimal sender utility is NP-hard, even in the setting with i.i.d. payoffs (in fact, even approximating it is hard). More interestingly, our main result shows that with i.i.d. payoffs but succinct inputs, even implementing the optimal signaling scheme is hard.

Theorem 1 (Optimal Scheme Sampling Hardness with Succinct Inputs). *Suppose there is an algorithm, which, given a Bayesian persuasion instance $\mathcal{I}$ with i.i.d. payoffs and a state realization θ in the succinct input model, outputs a signal $\sigma \sim \psi^\star(\theta, \cdot)$ from the optimal persuasive scheme $\psi^\star$ in* $\text{poly}(|\mathcal{I}| + |\theta|)$ *time. Then,* $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{NP}}$*, and the polynomial hierarchy collapses to the third level.*

To derive Theorem 1, we use a technique similar to that used by [1] for showing the classical hardness of the quantum Boson Sampling task. Namely, we show how the task of multiplicatively approximating the squared permanent of a matrix, which is #P-hard, can be reduced to sampling from the optimal persuasive scheme in a carefully constructed Bayesian persuasion instance. This allows us to then use the seminal result of Stockmeyer [28], which shows that #P computations can be approximated at a low level in the polynomial hierarchy. We give a detailed proof overview in Section 2.

We can also frame the result above in the language of oracle hardness, where an algorithm must query an oracle to infer any desired parameter of the instance.

Corollary 1 (Optimal Scheme Sampling Hardness with an Oracle). *Consider Bayesian persuasion instances $\mathcal{I}$ with i.i.d. payoffs on just two actions, where each action's type can be specified as a t-bit string in $\{0,1\}^t$. Let $\mathcal{O}_{\mathcal{I}}$ be an oracle which can be queried for the probability of any action type, the receiver/sender payoffs, or for a sample from the distribution over action types. There does not exist an algorithm, which, given as input a state realization θ, and access to the oracle $\mathcal{O}_{\mathcal{I}}$, outputs a signal $\sigma \sim \psi^\star(\theta, \cdot)$ from the optimal persuasive scheme $\psi^\star$ using only* $\text{poly}(t)$ *queries to $\mathcal{O}_{\mathcal{I}}$ (and running in* $\text{poly}(t)$ *time otherwise), unless the polynomial hierarchy collapses.*

Contrasting with the poly(n, m) algorithm of [14] for the i.i.d. setting in the explicit input model, we note in passing that Corollary 1 also implies a super-logarithmic lower bound for implementing the optimal persuasive scheme in that model.

One way to interpret the result above is the following. Recall our example from the introduction, where a financial advisor is trying to strategically recommend a client to purchase a stock. Imagine that the state of nature manifests as t binary market indicators for each stock that are revealed to the advisor when market opens for the day, assumed to be drawn i.i.d. from a known prior distribution. Then, even if the advisor has a strategy to persuade the client to the advisor's benefit, this comes at a computational cost. Namely, the result above says that even if the advisor can quickly compute how each stock fares in the long run based on its revealed indicators, they still need to make $\omega(\text{poly}(t))$ computations on potentially unrealized indicator states in order to play the right move.

The upshot of our result is that the tractability of algorithmic Bayesian persuasion is heavily contingent on specifying all the inputs explicitly. Indeed, with succinctly described inputs, even specialized instances like those having i.i.d. payoffs become computationally hard. In this regard, the algorithmic persuasion task is perhaps unlike other hard problems in computer science (like determinantal point processes [20,24]) that remain efficiently samplable even when the input is specified in a succinct form.

1.2 Related Work

Several works have studied various computational aspects of Bayesian persuasion and information design more generally. [14] studied the algorithmic problem facing the sender for computing an optimal scheme in Bayesian persuasion with a single sender and receiver. The setup with multiple receivers, where the receivers play a Bayesian zero-sum game upon being signaled by a sender, has also been studied, and the computational landscape there is more pessimistic [11,6,26]. [2,3] study the setup where a sender can send *private signals* to receivers. [15,31] further study the tractability of the sender's problem in the multi-receiver setting with binary actions, when there are no *externalities* between the receivers, whereas [33] study the setting with negative externalities between the receivers. A different line of work [16,23,32] also considers the computational aspects of persuasion under different notions of *robustness* to inaccurate/unknown information. Finally, [9,10,4,19] study Bayesian persuasion from an online learning perspective.

Our work is most closely related to the work of [14] who show a poly(n, m) algorithm for implementing the optimal scheme in n-action, m-action type Bayesian persuasion instances with i.i.d. payoffs and explicitly specified inputs. The authors also show that it is #P-hard to exactly compute the optimal sender payoff with independent and non-identical payoffs and explicitly specified inputs. For this latter result, [14] reduce the problem of deciding membership in a certain hard "Khintchine polytope" [18] to the problem of computing the optimal sender payoff; however, their reduction depends on an exponentially small separation in the optimal sender payoffs for deciding membership in the polytope, and it is unclear if this tiny separation could be leveraged to further show hardness for the comparatively easier task of sampling from the optimal scheme.

[14] also consider an input model where the action types of the n actions follow an arbitrary joint distribution, and the algorithm is only given access to a sampling oracle. The oracle, upon invocation, returns a sample in $[-1, 1]^{2n}$, which represents a realization

of the receiver and sender payoffs for each of the n actions. In this setting, the authors show a $\text{poly}(n, 1/\epsilon)$-time algorithm for implementing a signaling scheme that is both ϵ-optimal and ϵ-persuasive (namely, the persuasiveness constraints in (1) are only satisfied upto an additive ϵ slack). Since this algorithm is only a bicreteria approximation, it does not contradict our lower bounds. The authors also show that this bicriteria approximation is necessary by showing an information-theoretic lower bound. Essentially, with sample-only access, the authors show that any bounded number of samples do not suffice to simultaneously achieve both exact optimality and persuasiveness. In contrast, in this work, the input (even if succinctly described) has all the information necessary to exactly solve for the optimal signaling scheme (albeit with a large amount of computation); hence, the information-theoretic lower bounds of [14] do not apply to our setting.

Finally, while the tractability of the independent and non-identical payoffs setting remains open, we note a result by [13] showing that a natural class of efficient signaling schemes based on *Myersonian dice* provably does not implement the optimal scheme.

2 Technical Overview

Warm-up: Hardness of Estimating the Optimal Sender Payoff. We begin by providing intuition for how succinctly specified inputs can encode a computationally hard quantity. Let F be a 3-SAT formula on t variables, and consider the following Bayesian persuasion task $\mathcal{I}_F$ with i.i.d. payoffs: there are n actions (where $n \leq \exp(t)$), each with $m = 2^t$ types. Each action type may be identified with an $x \in \{0,1\}^t$. Let the distribution $\mathcal{P}$ over action types be the uniform distribution over $\{0,1\}^t$. Our objective will be to specify payoffs for an action type x in terms of $F(x)$: we set $s(x) = F(x)$, and $r(x) = 1 - F(x)$. Note that the instance $\mathcal{I}_F$ can be described succinctly with just $\text{poly}(t)$ bits: n, m together require $\text{poly}(t)$ bits, the circuits for the payoffs are at most the size of F, which is $\text{poly}(t)$, the circuit for evaluating the probability of an action type just needs to output the constant $1/2^t$, and the circuit for sampling from $\mathcal{P}$ needs to simply output t random bits.

Now consider any persuasive signaling scheme ψ. Persuasiveness requires that whenever the sender signals action i, the posterior payoff of the receiver from taking action i is no worse than their payoff from action j. Denoting x_i to be the type of action i, we get

$$\mathbb{E}_{x_i}[r(x_i)|\sigma = i] \geq \mathbb{E}_{x_j}[r(x_j)|\sigma = i] \implies \mathbb{E}_{x_i}[F(x_i) \mid \sigma = i] \leq \mathbb{E}_{x_j}[F(x_j) \mid \sigma = i].$$

Since this inequality holds for every $j \in [n]$, we get that

$$\mathbb{E}_{x_i}[F(x_i) \mid \sigma = i] \leq \frac{1}{n}\sum_{j=1}^{n} \mathbb{E}_{x_j}[F(x_j) \mid \sigma = i].$$

By persuasiveness, the LHS is exactly the sender's expected payoff conditioned on them signaling $\sigma = i$. Taking expectation over σ, and denoting $\#F = \sum_{x\in\{0,1\}^t} F(x)$, we get

$$\text{Expected Sender Payoff} \leq \frac{1}{n}\sum_{j=1}^{n} \mathbb{E}_{x_j}[F(x_j)] = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2^t} \sum_{x_j\in\{0,1\}^t} F(x_j) = \frac{\#F}{2^t},$$

where we used that each x_j is uniformly drawn from $\{0,1\}^t$. Thus, the expected sender payoff in any persuasive scheme is at most $\#F/2^t$. But this upper bound can be attained by the trivial uninformative scheme that always recommends action 1, regardless of the realized state. Since this signal contains no information, it does not change the prior of the receiver, and they might as well take action 1. So, this scheme is trivially persuasive, but also yields the sender an expected payoff $\mathbb{E}_{x_1}[F(x_1)] = \#F/2^t$. We therefore get that the optimal sender payoff in any scheme is equal to $\#F/2^t$. So, if an algorithm exactly computes the optimal expected sender payoff in $\text{poly}(|\mathcal{I}_F|) = \text{poly}(t)$ time, we can infer $\#F$ by scaling its output by the known constant 2^t. Since #3-SAT is #P-complete, we conclude that exactly computing the optimal expected sender payoff is #P-hard. In fact, even if the algorithm outputs a y satisfying $\frac{y}{g} \leq$ Expected Sender Payoff $\leq gy$ for any $g \geq 1$, we can decide whether F is satisfiable; this shows that even the task of multiplicatively approximating the optimal sender utility to any factor $g \geq 1$ is NP-hard.

Crucially, note that while computing the optimal sender payoff for the instance above is hard, sampling from the optimal scheme is easy — the uninformative scheme is optimal. In fact, for the hardness argument above to go through, it would have sufficed to consider a vacuous persuasion instance with just a single action, where the sender payoffs encode the evaluations of the 3-SAT formula. Nevertheless, the analysis above illustrates that simply encoding a computationally hard quantity into the payoffs of an action and naively replicating the action (possibly even an exponential number of times) does not preclude efficiently computable signaling schemes that are optimal.

Hardness of Sampling from the Optimal Signaling Scheme. Therefore, for establishing hardness of sampling, we would have to, at the very least, ensure that simple schemes like the one above are not optimal. But beyond that, there is a more central issue. For showing hardness of computing the optimal sender payoff, the strategy was clear: encode a computationally hard quantity to be proportional to the optimal sender payoff; for showing hardness of *sampling*, what should a hard quantity be encoded into?

Towards this, we take inspiration from the proof technique used by [1] for showing classical hardness of Boson Sampling. Their strategy is as follows: Suppose X is a *very* hard quantity to compute, in that it is #P-hard to even multiplicatively approximate X to a polynomial factor. [1] show that the squared permanent of a matrix is such a quantity. Now, suppose that there exists an efficient algorithm which samples from the desired distribution. Construct an event E over the execution of the algorithm, such that $\Pr[E]$ is proportional to the hard quantity X. Since the algorithm is efficient, we get that the computation of $\Pr[E]$ (upto proportionality) is in #P. We can then appeal to a classic result of Stockmeyer [28]: every #P computation can be approximated to a polynomial factor at a low level of the polynomial hierarchy PH. Invoking this, we will have shown that $\Pr[E]$, and hence the hard quantity X, can be approximated at this level of PH; since X was #P-hard-to-approximate to begin with, we would then have shown that the entire class $\mathsf{P}^{\#\mathsf{P}}$ is contained within this level of PH. Toda's theorem [29], which places the entirety of PH within $\mathsf{P}^{\#\mathsf{P}}$, completes the collapse of the polynomial hierarchy.

We adopt this blueprint to Bayesian persuasion, with the hard quantity X being $\text{Per}(M)^2$: the squared permanent of a $t \times t$ matrix M. The key technical challenge is to start with a matrix M, and construct an associated Bayesian persuasion instance $\mathcal{I}_M$ with i.i.d. payoffs, together with a load-bearing event E, which ensures that $\Pr[E] \propto \text{Per}(M)^2$.

The mechanism to encode $\mathrm{Per}(M)^2 = \left(\sum_\pi \prod_i M_{i,\pi_i}\right)^2$ into a persuasion instance is as follows: assume for simplicity that the entries in M are integers with polynomial bit complexity, and bounded in absolute value by R. Consider sampling a uniformly random permutation π over $[t]$. Now independently sample $z \in \{0, 1, \ldots, 2R^t - 1\}$ uniformly at random. Define $u(\pi, z) = +1$ if $z < R^t + \prod_i M_{i,\pi_i}$, and -1 otherwise. Conditioned on π, we have that $\mathbb{E}_z[u(\pi, z)|\pi] \propto \prod_i M_{i,\pi_i}$, which gives $\mathbb{E}_{z,\pi}[u(\pi, z)] \propto \sum_\pi \prod_i M_{i,\pi_i} \propto \mathrm{Per}(M)$. Therefore, if we sample two *independent* such pairs (π, z) and (τ, w), and consider $u(\pi, z) \cdot u(\tau, w)$, we will have that $\mathbb{E}[u(\pi, z) \cdot u(\tau, w)] \propto \mathrm{Per}(M)^2$, as desired. While one needs to be careful about being able to sample all the relevant quantities with a finite (and polynomial) number of random bits, this is essentially the main idea.

We therefore consider an action type in the persuasion instance to consist of the quantities π, τ, w, z sampled as above. For technical reasons, we also include a fair coin flip h within the action type. We then specify the sender and receiver payoffs as a function of $u(\pi, z) \cdot u(\tau, w)$ as well as h in the realized type. Recall that our ultimate goal is to be able to specify an event E whose probability is proportional to $\mathrm{Per}(M)^2$. We do this by first defining three auxiliary events A, B, C over the draw of the action type. The payoffs are such that the sender prefers action types in A, whereas the receiver prefers action types in B; furthermore, these events satisfy $\Pr[B] = \frac{1}{2}$, and $\Pr[A] - \Pr[C] \propto \mathrm{Per}(M)^2$.

Next, we characterize optimal persuasive schemes ψ for this instance, by writing down the objective and constraints in the linear program (1) in terms of $\Pr[A], \Pr[B]$ and $\Pr[C]$. A careful analysis of the constraints reveals that at optimality, any persuasive scheme ψ necessarily satisfies a certain equality condition. This condition essentially requires the scheme to signal an action with type A with larger probability than an action with type B, thus matching the sender's preference.[2] It remains to then construct an event, whose probability, given this characterization of the optimal scheme, is directly proportional to $\Pr[A] - \Pr[C] \propto \mathrm{Per}(M)^2$. Using $\Pr[B] = \frac{1}{2}$, we are able to successfully construct such an event E. The rest of the pipeline is as sketched above: assuming that the optimal scheme is efficiently computable, we monitor for the event E, and invoke Stockmeyer's theorem to conclude the collapse of the polynomial hierarchy.

We make one final remark: in order for the Bayesian persuasion instance above to be succinctly describable in $\mathrm{poly}(t)$ bits, it is necessary that the entries of the matrix M that we started with have $\mathrm{poly}(t)$ bit complexity. However, [1] write their #P-hardness proof of multiplicatively approximating the squared permanent for matrices with real entries, and do not explicitly bound the bit complexity of the entries in the matrix (since this was not necessary for their application). For the hardness reduction sketched above, we *do* require #P-hardness of multiplicatively approximating the squared permanent of matrices even when the entries have only $\mathrm{poly}(t)$ bit complexity. Therefore, along the way, we also give an explicit proof of this strengthened hardness result.

3 Hardness of Exactly Sampling from the Optimal Sender Scheme

We now give a complete derivation of Theorem 1. The two primary tools that we will use are: (1) #P-hardness of multiplicatively approximating the squared permanent of a matrix

[2] Note that a trivial, uninformative scheme does not satisfy such a condition.

[1], and (2) approximation of the probability that a randomized polynomial-time Turing machine accepts in $\mathsf{FBPP}^{\mathsf{NP}}$ [28]. Here, FBPP is the class of functions computable (with constant success probability) by a randomized polynomial-time Turing machine.[3]

The following #P-hardness result for the squared permanent was shown by [1], albeit for matrices with real entries. Since we require hardness even when the matrix is assumed to have rational entries with polynomial bit complexity, we give a self-contained proof of this strengthened version of their result in Section A for completeness.

Theorem 2 (Essentially Theorem 28 in [1]). *Fix any $g(t) \in \mathrm{poly}(t)$, where $g(t) \geq 1$. The following problem is #P-hard: Given any $t \times t$ rational matrix M, where entries in M are of the form $\frac{p}{q}$, $q > 0$, $|p|, q \leq 2^{t \log t}$, output z such that $\frac{\mathrm{Per}(M)^2}{g(t)} \leq z \leq g(t)\,\mathrm{Per}(M)^2$.*

We will use the following form of Stockmeyer's theorem [28] (Theorem 26 in [1]).

Theorem 3 (Stockmeyer's Theorem [28]). *Given a Boolean function $f : \{0,1\}^N \to \{0,1\}$, let $p = \Pr_{r \sim \{0,1\}^N}[f(r) = 1] = \frac{1}{2^N}\sum_{r \in \{0,1\}^N} f(r)$. Then, for all $g(N) \geq 1 + \frac{1}{\mathrm{poly}(N)}$, we can multiplicatively approximate p to a factor $g(N)$ in $\mathsf{FBPP}^{\mathsf{NP}^f}$.*

We now restate our main Theorem 1, and give its complete proof.

Theorem 1 (Optimal Scheme Sampling Hardness with Succinct Inputs). *Suppose there is an algorithm, which, given a Bayesian persuasion instance $\mathcal{I}$ with i.i.d. payoffs and a state realization θ in the succinct input model, outputs a signal $\sigma \sim \psi^\star(\theta, \cdot)$ from the optimal persuasive scheme $\psi^\star$ in $\mathrm{poly}(|\mathcal{I}| + |\theta|)$ time. Then, $\mathsf{PH} \subseteq \mathsf{BPP}^{\mathsf{NP}}$, and the polynomial hierarchy collapses to the third level.*

Proof. With Theorem 2 in mind, we will reduce from the #P-hard problem of multiplicatively approximating the squared permanent of a matrix. Given a $t \times t$ rational matrix M as input, where the entries $\{m_{i,j}\}$ in M are rationals having bit complexity $O(t \log t)$, we will construct an associated Bayesian persuasion instance $\mathcal{I}$ where sampling the optimal signaling scheme for the sender encodes $\mathrm{Per}(M)^2$.

Towards this, let L be the product of denominators of all entries in M, and let $H = LM \in \mathbb{Z}^{t \times t}$. We have that $\mathrm{Per}(H) = L^t \mathrm{Per}(M)$. Now, let Q and R be the smallest powers of two satisfying $Q \geq t$ and $R \geq 2\max\left\{1, \max_{i,j} |H_{i,j}|\right\}$. Note that $L > 0$, and all of L, H, Q, R have polynomial bit complexity.

Now consider a persuasion instance $\mathcal{I}$ with $n = 2$ actions. The state is $\theta = (\theta_1, \theta_2)$, where each θ_i is drawn i.i.d. from a distribution $\mathcal{P}$ (so that $D = \mathcal{P} \times \mathcal{P}$). The distribution $\mathcal{P}$ is over action types specified by the tuple $\lambda = (\pi, \tau, z, w, h)$, and sampled as follows:

1. π and τ are drawn independently and uniformly at random from $[Q]^t$.
2. z and w are drawn independently and uniformly at random from $\{0, 1, \ldots, 2R^t - 1\}$.
3. h is independently drawn uniformly at random from $\{0, 1\}$.

We now relate the distribution $\mathcal{P}$ to $\mathrm{Per}(M)^2$. For any $\lambda = (\pi, \tau, z, w, h)$, consider

$$v(\pi) = \begin{cases} \prod_{i=1}^{t} H_{i,\pi_i}, & \pi \text{ permutation of } [t], \\ 0, & \text{otherwise}, \end{cases} \qquad v(\tau) = \begin{cases} \prod_{i=1}^{t} H_{i,\tau_i}, & \tau \text{ permutation of } [t], \\ 0, & \text{otherwise}. \end{cases}$$

[3] For definitions of the other standard complexity classes used in this paper, please see [17].

Note that both $v(\pi)$, $v(\tau)$ are integers bounded in absolute value by $\left(\frac{R}{2}\right)^t$. Next, define

$$u(\pi, z) = \begin{cases} +1, & \text{if } z < R^t + v(\pi), \\ -1 & \text{otherwise,} \end{cases} \qquad u(\tau, w) = \begin{cases} +1, & \text{if } w < R^t + v(\tau), \\ -1 & \text{otherwise,} \end{cases}$$

$$b(\lambda) = u(\pi, z) \cdot u(\tau, w).$$

Then, a routine probability calculation gives us that

$$\Pr_{\lambda \sim \mathcal{P}}[b(\lambda) = +1] = \frac{1}{2} + \frac{\mathrm{Per}(H)^2}{2(QR)^{2t}}.$$

Define $\delta := \frac{\mathrm{Per}(H)^2}{2(QR)^{2t}}$; note that $0 \le \delta \le \frac{1}{8}$. For $\lambda \sim \mathcal{P}$, define the events

$$A := \{\lambda : h = 0,\ b(\lambda) = +1\}, \quad B := \{\lambda : h = 1\}, \quad C := \{\lambda : h = 0,\ b(\lambda) = -1\}.$$

We then have that

$$\Pr[A] = 1/4 + \delta/2, \quad \Pr[B] = 1/2, \quad \Pr[C] = 1/4 - \delta/2. \tag{2}$$

Let $a = \Pr[A], b = \Pr[B], c = \Pr[C]$, and note that all of a, b, c are non-zero.

We are now ready to specify the receiver and sender payoffs.

Receiver Payoffs

$$r(\lambda) = \begin{cases} 0 & \text{if } \lambda \in A, \\ 1 & \text{if } \lambda \in B, \\ 0 & \text{if } \lambda \in C. \end{cases}$$

Sender Payoffs

$$s(\lambda) = \begin{cases} 1 & \text{if } \lambda \in A, \\ 0 & \text{if } \lambda \in B, \\ 0 & \text{if } \lambda \in C. \end{cases} \tag{3}$$

Namely, the receiver only gets a payoff if $\lambda \in B$, the sender only gets a payoff if $\lambda \in A$, and both are indifferent to $\lambda \in C$. This completes the specification of the instance $\mathcal{I}$.

We first argue that this instance has description length $|\mathcal{I}| + |\theta| = \mathrm{poly}(t)$ in the succinct input model. To see this, note that $n = 2$. Since the tuple (π, τ, z, w, h) can take at most $Q^{2t} \cdot 4R^{2t} \cdot 2$ values, the number of action types m can be described in at most $\log(Q^{2t} \cdot 4R^{2t} \cdot 2) = \mathrm{poly}(t)$ bits, and so can any particular state θ. The circuit for evaluating $\mathcal{P}(\lambda)$ has to simply output $\frac{1}{8(QR)^{2t}}$, which can be done with a circuit of size $\mathrm{poly}(t)$. We also have that the circuit for sampling $\lambda \sim \mathcal{P}$ has size $\mathrm{poly}(t)$. To see this, observe that sampling π and τ requires precisely $t \log(Q)$ random bits each, sampling z and w each requires exactly $1 + t \log(R)$ random bits, and sampling h requires just a single random bit. Finally, the circuits for implementing the receiver and sender payoffs $r(\lambda)$ and $s(\lambda)$ also have size $\mathrm{poly}(t)$, since this requires simply evaluating $b(\lambda)$.

We now calculate the sender's utility in a persuasive scheme ψ. For $U, V \in \{A, B, C\}$, let UV be shorthand for the event $\theta_1 \in U, \theta_2 \in V$, and let $q_{UV} \in [0, 1]$ denote the probability that the sender signals action 1, conditioned on $\theta_1 \in U, \theta_2 \in V$. That is,

$$q_{UV} = \Pr_{\theta, \psi}[\sigma = 1 \mid \theta_1 \in U, \theta_2 \in V] = \frac{\sum_{\theta_1 \in U, \theta_2 \in V} \mathcal{P}(\theta_1)\mathcal{P}(\theta_2)\psi((\theta_1, \theta_2), 1)}{\sum_{\theta_1 \in U, \theta_2 \in V} \mathcal{P}(\theta_1)\mathcal{P}(\theta_2)}.$$

Then, recall that the sender utility in any persuasive ψ is $\mathbb{E}_\theta\left[\mathbb{E}_{\sigma\sim\psi(\theta,\cdot)}[s(\theta_\sigma)]\right]$. Observe from (3) that if neither of θ_1, θ_2 belong to A, the sender gets no payoff. Namely, the inner expectation is non-zero only on the events in $\{AA, AB, BA, AC, CA\}$. Thus,

$$\mathbb{E}_\theta\left[\mathbb{E}_{\sigma\sim\psi(\theta,\cdot)}[s(\theta_\sigma)]\right] = a^2 + ab + ac + ab(q_{AB} - q_{BA}) + ac(q_{AC} - q_{CA}).$$

Now consider the persuasiveness constraints that any persuasive ψ must satisfy. Suppose that the sender signals $\sigma = 1$; the receiver's posterior utility in taking action 1 is $\mathbb{E}[r(\theta_1)|\sigma = 1]$. Then, observe from (3) that the receiver gets payoff only if $\theta_1 \in B$; that is, $r(\theta_1)$ is non-zero (and equal to 1) only on the events in $\{BB, BA, BC\}$. Thus,

$$\mathbb{E}[r(\theta_1)|\sigma = 1] = \frac{b^2 q_{BB} + abq_{BA} + bcq_{BC}}{\Pr[\sigma = 1]}.$$

The receiver's posterior utility in taking action 2 is $\mathbb{E}[r(\theta_2) \mid \sigma = 1]$. Again, from (3), $r(\theta_2)$ is non-zero (and equal to 1) only on the events in $\{BB, AB, CB\}$. Thus,

$$\mathbb{E}[r(\theta_2)|\sigma = 1] = \frac{b^2 q_{BB} + abq_{AB} + bcq_{CB}}{\Pr[\sigma = 1]}.$$

The persuasiveness constraint then requires:

$$ab(q_{BA} - q_{AB}) + bc(q_{BC} - q_{CB}) \geq 0 \implies ab(q_{AB} - q_{BA}) \leq bc(q_{BC} - q_{CB}),$$

Repeating the above calculation for when the sender signals $\sigma = 2$ yields the same persuasiveness constraint. Thus, the sender's optimization problem is:

$$\max_{\psi} \quad a^2 + ab + ac + ab(q_{AB} - q_{BA}) + ac(q_{AC} - q_{CA}) \tag{4}$$

$$\text{subject to} \quad ab(q_{AB} - q_{BA}) \leq bc(q_{BC} - q_{CB}). \tag{5}$$

Recall that $a, b, c > 0$. Since q_{AC}, q_{CA} don't appear in the constraint, any optimal $\psi^\star$ must satisfy $q_{AC} = 1, q_{CA} = 0$. The term $ab(q_{AB} - q_{BA})$ appears in the objective, but is constrained to be at most $bc(q_{BC} - q_{CB})$; however, we are free to set $q_{BC} - q_{CB}$, since neither q_{BC} nor q_{CB} appear in the objective. Thus, it is optimal to set $q_{BC} = 1, q_{CB} = 0$, and have $ab(q_{AB} - q_{BA}) = bc$, meaning that $q_{AB} - q_{BA} = \frac{c}{a}$. Note that this is feasible, since $c \leq a$. In conclusion, we have argued that any optimal $\psi^\star$ satisfies $q_{AB} - q_{BA} = \frac{c}{a}$.

Now, let E denote the event:

$$E := (AB \text{ and the sender signals } \sigma = 2) \text{ or } (BA \text{ and the sender signals } \sigma = 1).$$

Namely, E is the event that either $\theta_1 \in A, \theta_2 \in B$ or $\theta_1 \in B, \theta_2 \in A$, and the sender signals the action realized to have type in B. Then, for any optimal persuasive $\psi^\star$,

$$\Pr_{\theta,\psi^\star}[E] = \Pr[AB](1 - q_{AB}) + \Pr[BA]q_{BA} = ab - ab(q_{AB} - q_{BA}) = ab\,(1 - c/a)$$

(from the characterization of $\psi^\star$ above)

$$= \frac{\delta}{2} = \frac{\mathrm{Per}(H)^2}{4(QR)^{2t}} = \frac{L^{2t}\mathrm{Per}(M)^2}{4(QR)^{2t}}. \qquad \text{(using (2) and substituting } \delta)$$

We have thus shown that over the draw of $\theta \sim \mathcal{P} \times \mathcal{P}$, followed by sampling $\sigma \sim \psi^\star(\theta, \cdot)$ from any optimal persuasive $\psi^\star$, the probability of event E is exactly equal to $\frac{L^{2t}\text{Per}(M)^2}{4(QR)^{2t}}$.

Now suppose that there is a (possibly randomized) algorithm $\mathcal{A}$ which takes as input the instance $\mathcal{I}$ above, along with a state realization $\theta \sim D$ in the succinct input model, and outputs a signal $\sigma \sim \psi^\star(\theta, \cdot)$ from the optimal persuasive signaling scheme in $\text{poly}(|\mathcal{I}| + \theta|)$ time. Consider executing the following procedure: (1) Sample $\theta \sim D$. (2) Run $\mathcal{A}(\mathcal{I}, \theta)$ to obtain $\sigma \sim \psi^\star(\theta, \cdot)$. (3) Output 1 iff the event E occurs, 0 otherwise.

Running the procedure above requires $\text{poly}(t)$ random bits and time. To see this, note that sampling $\theta \sim D$ requires $\text{poly}(t)$ random bits, as argued above. Thereafter, running $\mathcal{A}$ also requires $\text{poly}(|\mathcal{I}| + |\theta|) = \text{poly}(t)$ time by assumption, and hence $\text{poly}(t)$ random bits. Thus, the entire procedure uses $\text{poly}(t)$ random bits and time. This procedure can be thought of as computing a Boolean function $f : \{0,1\}^{\text{poly}(t)} \to \{0,1\}$. From Theorem 3, we then know that $p = \Pr_{r \sim \{0,1\}^{|\text{poly}(t)|}}[f(r) = 1]$ can be multiplicatively approximated to any $1 + \frac{1}{\text{poly}(t)}$ factor by an $\mathsf{FBPP}^{\mathsf{NP}^f}$ algorithm $\mathcal{A}'$. Since the computation of f is further only a polynomial computation, $\mathcal{A}'$ is simply an $\mathsf{FBPP}^{\mathsf{NP}}$ algorithm.

But crucially, by our analysis above, observe that the probability p is exactly equal to $\frac{L^{2t}\text{Per}(M)^2}{4(QR)^{2t}}$. We can thus multiply the output of $\mathcal{A}'$ by the known scaling constant $\frac{4(QR)^{2t}}{L^{2t}}$ to obtain a $1 + \frac{1}{\text{poly}(t)}$ factor multiplicative approximation to $\text{Per}(M)^2$ in $\mathsf{FBPP}^{\mathsf{NP}}$.

Theorem 2 states that multiplicatively approximating $\text{Per}(M)^2$ to any polynomial factor larger than 1 is #P-hard. Combining with the above, we get that $\#\mathsf{P} \subseteq \mathsf{FBPP}^{\mathsf{NP}}$. Toda's theorem [29] then gives that $\mathsf{PH} \subseteq \mathsf{P}^{\#\mathsf{P}} \subseteq \mathsf{BPP}^{\mathsf{NP}}$. By the Sipser-Gács-Lautemann theorem [27,25], $\mathsf{BPP} \subseteq \Sigma_2 \cap \Pi_2$, and the polynomial hierarchy collapse is complete.

4 Discussion and Concluding Thoughts

Our result shows that the task of efficiently implementing an optimal scheme in Bayesian persuasion generically requires the number of actions n and types m to be small; with an exponential number of action types whose behavior can nevertheless be succinctly described, implementing the optimal scheme in the worst case can be hard, even when the payoffs are i.i.d, in contrast to the results of [14] with explicitly described inputs.

The structural content of our hard instance is visible after compiling its action types into three payoff-relevant categories A, B, C. Types in A benefit only the sender, types in B benefit only the receiver, and types in C benefit neither. Thus, the hard instance can be interpreted as a standard two-option recommendation problem — for example, a financial advisor choosing between two stocks, where each option has many fine-grained features, but only coarse (and sometimes conflicting) consequences on payoff.

With these conflicting coarse categories, the optimal scheme in our hard instance illustrates a somewhat curious (but natural) "credibility budgeting" property. Namely, in a B-versus-C scenario, recommending the B option benefits the receiver, but is costless to the sender. In doing so, the sender nevertheless accumulates favorable evidence to sometimes persuasively recommend the A option to the receiver in an A-versus-B scenario, where the payoffs are conflicting. Indeed, inspecting equation (5) shows how the obedience constraint limits the amount of conflict state persuasion by the available mass of B-versus-C states, and every optimal scheme exhausts this constraint exactly.

This is the familiar information design logic that recommendations in aligned states can be pooled with recommendations in conflict states to make the latter credible.

Another property of our instance is that once the aggregate probabilities $\Pr[A]$, $\Pr[B]$ and $\Pr[C]$ are known, the optimal policy becomes computationally tractable. All of the computational difficulty lies in deriving these three aggregate masses from the succinctly represented input. In other words, the obstacle is not complicated signaling geometry or many actions; it is the need to compute the total probability of a region containing exponentially many payoff-equivalent microstates. The permanent construction is simply the device used to make one such aggregate probability hard to recover.

This phenomenon can arise naturally in feature-based recommendation, financial advice, or disclosure problems: an option may be described by a large vector of characteristics, while utilities depend only on whether that vector falls into a few coarse categories. Even when one can efficiently sample a feature vector and evaluate its realized payoffs, implementing the optimal policy may require knowing the exact prevalence of those categories. Our instance therefore identifies an aggregation bottleneck behind optimal persuasion with succinct priors, and clarifies that tractability in the explicit input model crucially relies on those aggregate probabilities already being available.

Going back to the question originally asked by [14]: it remains open to efficiently implement the optimal scheme in the setting with independent but *non-identical* payoffs with explicitly described inputs. This is because a polynomial-time algorithm in the succinct input model implies one in the explicit input model (since an explicit input comprising all the relevant parameters translates to a succinct input of the same size with trivial circuits computing these numbers). Conversely, hardness in the explicit input model implies hardness in the succinct input model, but not vice versa.

To resolve the open question in [14], one would have to reduce from a hard problem of size t to a persuasion instance with independent, non-identical payoffs, but with only $\mathrm{poly}(t)$ actions and types: the limitation in our instance is that it has $\exp(t)$ action types. One may try to reduce the number of types to $\mathrm{poly}(t)$, but increase the actions to $\mathrm{poly}(t)$. However, even with $\mathrm{poly}(t)$ actions and types, the number of parameters involved are only $\mathrm{poly}(t)$, since the payoffs *only* depend on the type of the action taken (and not the entire profile of action types), and the action types are realized independently. It is thus not clear how an exponential computation could be hidden within the persuasiveness constraints of such an instance. Regardless, we believe that the pipeline used in the proof of Theorem 1 is a potential stepping stone towards such a result.

Acknowledgments. I would like to thank Aviad Rubinstein for encouraging me to be ambitious for my PhD quals, and Shaddin Dughmi for helpful comments. This work is supported by a Google PhD Fellowship, and Moses Charikar's and Gregory Valiant's Simons Investigator Awards.

Disclosure of Interests. The author has no competing interests to declare.

A #P-Hardness of Approximating $\mathrm{Per}(X)^2$

Theorem 2 (Essentially Theorem 28 in [1]). *Fix any $g(t) \in \mathrm{poly}(t)$, where $g(t) \geq 1$. The following problem is #P-hard: Given any $t \times t$ rational matrix M, where entries in M are of the form $\frac{p}{q}$, $q > 0$, $|p|, q \leq 2^{t \log t}$, output z such that $\frac{\mathrm{Per}(M)^2}{g(t)} \leq z \leq g(t)\,\mathrm{Per}(M)^2$.*

We first distill out a discrete version of the main argument in [1]:

Lemma 1 (Interval Shrinking Lemma). *Let $g \geq 1$ be given. For an unknown $a \in \mathbb{Z}$, suppose it is known that $a \in [L, U]$, where $L, U \in \mathbb{Z}$ are given, and the interval $[L, U]$ has width $W := U - L + 1 > 20g$. Fix $d = \lceil \frac{W}{20g} \rceil$, $m = \lfloor \frac{U-L}{d} \rfloor$ and define*

$$K := \{L, L + d, L + 2d, \ldots, L + md\} \cup \{U\}.$$

Now, suppose that for each $k \in K$, we are given a number z_k satisfying

$$\frac{(a-k)^2}{g} \leq z_k \leq g(a-k)^2.$$

Let $k_{\min} \in \arg\min_k z_k$. Then, for $R := \lceil gd/2 \rceil$, we have that $a \in [k_{\min} - R, k_{\min} + R]$, and furthermore, this interval has width $2R + 1 \leq W/4$.

Proof. First, observe that every point in the interval $[L, U]$ lies at a distance at most $d/2$ from some point in K. Therefore, since $a \in [L, U]$, there exists some $k^\star \in K$ such that $|a - k^\star| \leq d/2$. Now, consider $k_{\min} \in \arg\min_k z_k$. We have that

$$z_{k_{\min}} \leq z_{k^\star} \leq g(a - k^\star)^2 \leq g(d/2)^2.$$

But also, it holds that $\frac{(a-k_{\min})^2}{g} \leq z_{k_{\min}}$. Combining, we get $|a - k_{\min}| \leq gd/2$. Defining $R := \lceil gd/2 \rceil$ then immediately gives that $a \in [k_{\min} - R, k_{\min} + R]$. It remains to argue that this interval has width at most $W/4$. For this, observe that

$$2R + 1 = 2\lceil gd/2 \rceil + 1 \leq gd + 3.$$

But, since $d = \lceil \frac{W}{20g} \rceil \leq \frac{W}{20g} + 1$, and $1 < \frac{W}{20g}$ by assumption, we get that $d \leq \frac{W}{10g}$. Thus,

$$gd + 3 \leq \frac{W}{10} + 3 < \frac{W}{10} + \frac{3W}{20g} \leq \frac{W}{10} + \frac{3W}{20} = \frac{W}{4},$$

where we used that $g \geq 1$ in the second-to-last inequality.

We are now ready to prove Theorem 2.

Proof (Proof of Theorem 2). Let O be an oracle, which, given a $t \times t$ matrix M whose entries satisfy the restrictions of Theorem 2, outputs $O(M)$ satisfying

$$\frac{\mathrm{Per}(M)^2}{g(t)} \leq O(M) \leq g(t)\,\mathrm{Per}(M)^2.$$

We will reduce from the problem of computing the permanent of a matrix $X \in \{0, 1\}^{t \times t}$, which was shown to be #P-hard in the seminal work of Valiant [30]. Namely, given $X \in \{0, 1\}^{t \times t}$, and access to O, we will show how we can exactly compute $\mathrm{Per}(X)$.

We proceed by induction on the matrix dimension t. For $t = 1$, $\mathrm{Per}(X)$ is simply the unique entry in X. So, let us assume as our induction hypothesis that we can exactly compute the permanent of any $(t-1) \times (t-1)$ matrix with entries in $\{0, 1\}$. Given

$X \in \{0,1\}^{t \times t}$, we will then show how, with polynomially many queries to O as well as some additional polynomial amount of computation, we can exactly compute $\mathrm{Per}(X)$.

First, note that we can check if $\mathrm{Per}(X) = 0$ in $\mathrm{poly}(t)$ time. This is because $\mathrm{Per}(X)$ for $X \in \{0,1\}^{t \times t}$ is equal to the count of distinct perfect matchings in the bipartite graph associated with X (rows correspond to the left vertices, columns correspond to the right vertices, and $x_{i,j} = 1$ indicates that the ith left vertex is connected to the jth right vertex). Thus, $\mathrm{Per}(X) = 0$ if and only if there exists no perfect matching in this associated bipartite graph, which can be determined in $\mathrm{poly}(t)$ time.[4] So, we assume that $\mathrm{Per}(X) \geq 1$. This means that there exists a permutation π such that $\prod_{i=1}^{t} x_{i,\pi(i)} = 1$. By permuting the columns of X, we can assume without loss of generality that $x_{1,1} = x_{2,2} = \cdots = x_{t,t} = 1$, while keeping $\mathrm{Per}(X)$ unchanged.

Then, let Y be the $(t-1) \times (t-1)$ bottom-right submatrix of X. Note that all the diagonal entries of Y are 1, and hence $\mathrm{Per}(Y) \geq 1$. Furthermore, by our induction hypothesis, we can compute $\mathrm{Per}(Y)$ exactly, so let $b = \mathrm{Per}(Y) \geq 1$. Now, let $a = \mathrm{Per}(X)$: our goal is to compute a exactly. Note that a is an integer, and $1 \leq a \leq t!$.

Adopting notation from [1], for an integer k, define $X^{[k]}$ to be the matrix that is identical to X, except that its top-left entry is equal to $1 - \frac{k}{b}$ instead of 1. Then, the following crucial linear relation may be readily verified:

$$\mathrm{Per}(X^{[k]}) = \mathrm{Per}(X) - \frac{k}{b}\mathrm{Per}(Y) = a - k.$$

Thus, the output $O(X^{[k]})$ upon querying the oracle with $X^{[k]}$, provided $X^{[k]}$ is in the promised input class, satisfies that

$$\frac{(a-k)^2}{g(t)} \leq O(X^{[k]}) \leq g(t)\,(a-k)^2. \tag{6}$$

In particular, for any k satisfying $1 \leq k \leq t!$, note that all the entries in $X^{[k]}$ except the top-left entry are either 0 or 1. The top-left entry is $1 - \frac{k}{b} = \frac{b-k}{b}$. Note that $b > 0$, and also that $b = \mathrm{Per}(Y) \leq (t-1)! \leq 2^{t \log t}$. Furthermore, $|b - k| \leq t! \leq 2^{t \log t}$. Thus, for any k satisfying $1 \leq k \leq t!$, the entries in the matrix $X^{[k]}$ satisfy the requirements of the oracle O.

Our strategy then is to repeatedly invoke Lemma 1, with $g = g(t)$. We initialize $I_0 = [L_0, U_0]$, where $L_0 = 1$ and $U_0 = t!$, and we know that $a \in [L_0, U_0]$. Then, for $i = 0, 1, 2, \ldots$, if the width $W_i = U_i - L_i + 1 > 20g$, we proceed as follows: we set $d_i = \lceil \frac{W_i}{20g} \rceil$, $m_i = \lfloor \frac{U_i - L_i}{d_i} \rfloor$ and $K_i = \{L_i, L_i + d_i, L_i + 2d_i, \ldots, L_i + m_i d_i\} \cup \{U_i\}$. For each $k \in K_i$, we let $z_k = O(X^{[k]})$, and $k_{\min} \in \arg\min_k z_k$. We then set $R_i = \lceil g d_i / 2 \rceil$, and update $I_{i+1} = [k_{\min} - R_i, k_{\min} + R_i] \cap [1, t!]$. Otherwise, if the width $W_i = U_i - L_i + 1 \leq 20g$, we break out of the loop.

We now argue inductively that $a \in I_i$ and $I_i \subseteq [1, t!]$ with integer endpoints at the beginning of the ith iteration of the loop, for every $i \geq 0$. This clearly holds for the base case when $i = 0$. So, suppose that this is the case at the beginning of the ith iteration of the loop. Then, all the numbers in the set K_i are integers belonging to $\{1, \ldots, t!\}$. This implies, by our reasoning above, that all the queries to O comprise of valid inputs, and

[4] e.g., by framing it as a max-flow problem.

hence from (6), for every $k \in K_i$, $z_k = O(X^{[k]})$ satisfies $\frac{(a-k)^2}{g(t)} \leq z_k \leq g(t) \cdot (a-k)^2$. From our update rule and the conclusion of Lemma 1, we then have that $a \in I_{i+1}$ and $I_{i+1} \subseteq [1, t!]$ with integer endpoints; furthermore, it also holds that $W_{i+1} \leq W_i/4$. This completes the induction.

Since the width shrinks by a factor of 4 at every iteration, the loop breaks in $T = O(\log(t!)) = O(t \log t)$ iterations, at which point we have that I_T has width at most $20g$, and furthermore, by the maintained loop invariant, $a \in I_T$ and $I_T \subseteq [1, t!]$. We then simply query $O(X^{[k]})$ for every integer $k \in I_T$. Because $\mathrm{Per}(X^{[k]}) = a - k \in \mathbb{Z}$, if $k = a$, then $\mathrm{Per}(X^{[k]}) = 0$, and hence $O(X^{[k]}) = 0$ necessarily from (6). On the other hand, if $k \neq a$, then $(a-k)^2 \geq 1$, and hence $O(X^{[k]}) \geq 1/g(t) > 0$. Therefore, we can exactly infer the value of $a = \mathrm{Per}(X)$ by finding the unique k which results in $O(X^{[k]}) = 0$.

We conclude by calculating the total number of oracle calls and additional computation required. In each iteration of the loop, we perform at most $|K_i| \leq m_i + 2 \leq 20g(t) + 2$ many oracle calls. After the loop breaks, we make at most $20g(t)$ further oracle calls. Since there are at most $O(t \log t)$ iterations of the loop, the total number of oracle calls is thus $O(t \cdot g(t) \cdot \log t)$. Summing further over the t levels of induction, we get that the total number of oracle calls is asymptotically $O(t^2 \cdot g(t) \cdot \log t)$. Beyond this, we only perform $\mathrm{poly}(t)$ additional computation — checking for a perfect matching, as well as polynomially many arithmetic operations/comparisons, all over $\mathrm{poly}(t)$ bit rationals. Therefore, since $g(t) \in \mathrm{poly}(t)$, we have shown a polynomial-time (Turing) reduction from the problem of computing the permanent, to the problem of multiplicatively approximating the squared permanent. This concludes the proof of #P-hardness.

References

1. Aaronson, S., Arkhipov, A.: The computational complexity of linear optics. In: Proceedings of the forty-third annual ACM symposium on Theory of computing. pp. 333–342 (2011)
2. Arieli, I., Babichenko, Y.: Private bayesian persuasion. Journal of Economic Theory **182**, 185–217 (2019)
3. Babichenko, Y., Barman, S.: Algorithmic aspects of private bayesian persuasion. In: 8th Innovations in Theoretical Computer Science Conference (ITCS 2017). pp. 34–1. Schloss Dagstuhl–Leibniz-Zentrum für Informatik (2017)
4. Bacchiocchi, F., Bollini, M., Castiglioni, M., Marchesi, A., Gatti, N.: Online bayesian persuasion without a clue. Advances in Neural Information Processing Systems **37**, 76404–76449 (2024)
5. Bergemann, D., Morris, S.: Information design: A unified perspective. Journal of Economic Literature **57**(1), 44–95 (2019)
6. Bhaskar, U., Cheng, Y., Ko, Y.K., Swamy, C.: Hardness results for signaling in bayesian zero-sum and network routing games. In: Proceedings of the 2016 ACM Conference on Economics and Computation. pp. 479–496 (2016)
7. Border, K.C.: Implementation of reduced form auctions: A geometric approach. Econometrica **59**(4), 1175–1187 (1991)
8. Border, K.C.: Reduced form auctions revisited. Economic Theory **31**(1), 167–181 (2007)
9. Castiglioni, M., Celli, A., Marchesi, A., Gatti, N.: Online bayesian persuasion. Advances in neural information processing systems **33**, 16188–16198 (2020)

10. Castiglioni, M., Marchesi, A., Celli, A., Gatti, N.: Multi-receiver online bayesian persuasion. In: International Conference on Machine Learning. pp. 1314–1323. PMLR (2021)
11. Dughmi, S.: On the hardness of signaling. In: 2014 IEEE 55th Annual Symposium on Foundations of Computer Science. pp. 354–363. IEEE (2014)
12. Dughmi, S.: Algorithmic information structure design: a survey. ACM SIGecom Exchanges **15**(2), 2–24 (2017)
13. Dughmi, S., Kempe, D., Qiang, R.: Alea iacta est: Auctions, persuasion, interim rules, and dice. arXiv preprint arXiv:1811.11417 (2018)
14. Dughmi, S., Xu, H.: Algorithmic bayesian persuasion. In: Proceedings of the forty-eighth annual ACM symposium on Theory of Computing. pp. 412–425 (2016)
15. Dughmi, S., Xu, H.: Algorithmic persuasion with no externalities. In: Proceedings of the 2017 ACM Conference on Economics and Computation. pp. 351–368 (2017)
16. Dworczak, P., Pavan, A.: Preparing for the worst but hoping for the best: Robust (bayesian) persuasion. Econometrica **90**(5), 2017–2051 (2022)
17. Goldreich, O.: Computational complexity: a conceptual perspective. ACM Sigact News **39**(3), 35–39 (2008)
18. Gopalan, P., Nisan, N., Roughgarden, T.: Public projects, boolean functions, and the borders of border's theorem. ACM Transactions on Economics and Computation (TEAC) **6**(3-4), 1–21 (2018)
19. Harris, K., Immorlica, N., Lucier, B., Slivkins, A.: Algorithmic persuasion through simulation. arXiv preprint arXiv:2311.18138 (2023)
20. Hough, J.B., Krishnapur, M., Peres, Y., Virág, B.: Determinantal processes and independence (2006)
21. Kamenica, E.: Bayesian persuasion and information design. Annual Review of Economics **11**(1), 249–272 (2019)
22. Kamenica, E., Gentzkow, M.: Bayesian persuasion. American Economic Review **101**(6), 2590–2615 (2011)
23. Kosterina, S.: Persuasion with unknown beliefs. Theoretical Economics **17**(3), 1075–1107 (2022)
24. Kulesza, A., Taskar, B.: Determinantal point processes for machine learning. Foundations and Trends® in Machine Learning **5**(2-3), 123–286 (2012)
25. Lautemann, C.: Bpp and the polynomial hierarchy. Information Processing Letters **17**(4), 215–217 (1983)
26. Rubinstein, A.: Honest signaling in zero-sum games is hard, and lying is even harder. arXiv preprint arXiv:1510.04991 (2015)
27. Sipser, M.: A complexity theoretic approach to randomness. In: Proceedings of the fifteenth annual ACM symposium on Theory of computing. pp. 330–335 (1983)
28. Stockmeyer, L.: The complexity of approximate counting. In: Proceedings of the fifteenth annual ACM symposium on Theory of computing. pp. 118–126 (1983)
29. Toda, S.: Pp is as hard as the polynomial-time hierarchy. SIAM Journal on Computing **20**(5), 865–877 (1991)
30. Valiant, L.G.: The complexity of computing the permanent. Theoretical computer science **8**(2), 189–201 (1979)
31. Xu, H.: On the tractability of public persuasion with no externalities. In: Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms. pp. 2708–2727. SIAM (2020)
32. Yang, K., Zhang, H.: Computational aspects of bayesian persuasion under approximate best response. Advances in Neural Information Processing Systems **37**, 134430–134458 (2024)
33. Zhou, C., Nguyen, T.H., Xu, H.: Algorithmic information design in multi-player games: Possibilities and limits in singleton congestion. In: Proceedings of the 23rd ACM Conference on Economics and Computation. pp. 869–869 (2022)