# Table of Contents

Editor’s Introduction  
SHADDIN DUGHMI  

Reflections on the First Man vs. Machine No-Limit Texas Hold ‘em Competition  
SAM GANZFRIED  

Two Desirable Fairness Concepts for Allocation of Indivisible Objects under Ordinal Preferences  
HARIS AZIZ, SERGE GASPERS, SIMON MACKENZIE, and TOBY WALSH  

Algorithms as Mechanisms: The Price of Anarchy of Relax-and-Round  
PAUL DÜTTING, THOMAS KESSELHEIM, and ÉVA TARDOS  

Incentivizing High Quality Crowdwork  
CHIEN-JU HO, ALEKSANDRS SLIVKINS, SIDDHARTH SURI, and JENNIFER WORTMAN VAUGHAN  

Why Prices Need Algorithms  
TIM ROUGHGARDEN and INBAL TALGAM-COHEN  

The Complexity of Simplicity in Mechanism Design  
AVIAD RUBINSTEIN  

By submitting an article to the SIGecom Exchanges, and on condition of acceptance by the editor, authors grant to ACM the following non-exclusive, perpetual, worldwide rights:

— to publish the article in print;
— to digitize and post the article in the electronic version of the publication;
— to include the article in the ACM Digital Library and in any Digital Library related services;
— to allow users to make a personal copy of the article for noncommercial, educational or research purposes.

Contributing authors retain copyright to the article, and ACM will refer requests for republication directly to them.
Editor’s Introduction

SHADDIN DUGHMI
University of Southern California

Issue 14.2 features 5 excellent letters, as well a retrospective by Sam Ganzfried on the first man vs machine no-limit Texas Hold ’em competition. In his retrospective, Sam reflects on the design of their agent — Claudico — and the lessons learned from observing its performance against world-class poker players.

I hope you enjoy this issue. Please continue to volunteer letters, surveys, and position papers. As usual, thanks to Felix Fischer for his help in putting the issue together!

Author’s address: shaddin@usc.edu
Reflections on the First Man vs. Machine No-Limit Texas Hold ’em Competition

SAM GANZFRIED

The first ever human vs. computer no-limit Texas hold ’em competition took place from April 24–May 8, 2015 at River’s Casino in Pittsburgh, PA. In this article I present my thoughts on the competition design, agent architecture, and lessons learned.

Categories and Subject Descriptors: I.2.11 [Distributed Artificial Intelligence]: Multiagent Systems; J.4 [Social and Behavioral Sciences]: Economics

General Terms: Algorithms, Design, Documentation, Economics, Experimentation, Theory

Additional Key Words and Phrases: Artificial Intelligence, Game Theory, Imperfect Information

1. INTRODUCTION

The first ever human vs. computer no-limit Texas hold ’em competition took place from April 24–May 8, 2015 at River’s Casino in Pittsburgh, PA, organized by Carnegie Mellon University Professor Tuomas Sandholm. 20,000 hands of two-player no-limit Texas hold ’em were played between the computer program “Claudico” and four of the top human specialists in this variation of poker, Dong Kim, Jason Les, Bjorn Li, and Doug Polk (so 80,000 hands were played in total).1

To evaluate the performance, we used “duplicate” scoring, in which the same hands were played twice with the cards reversed to reduce the role of luck (and

---

1Doug Polk tweeted a list on 2/28/2015 ranking himself at number one, Kim number two, Li number three, and Les (according to speculation on his screenname) within the top ten, https://twitter.com/DougPolkPoker/status/571647246074163201. Several other players have also created lists placing Polk at number one (e.g., Nick Frame tweeted one on 9/28/2014, https://twitter.com/TCfromUB/status/516396810433486848). While these rankings are largely subjective, they are based on some objective factors; e.g., if player A beats player B over a significant sample of hands, or if player A is willing to play against player B but player B refuses to play against player A (i.e., by leaving the table when player A sits in against him), then these indicate an advantage of player A over player B. If one player contests the ranking and believes he is better than someone ranked higher, then a challenge can ensue (e.g., Kim and Frame played a challenge match in February 2015, https://www.pokerstars.com/en/blog/2015/dong-donger-kim-kyu-and-nick-tcfromub-frame-on-their-unique-heads-up-challenge-up-challenge-154091.shtml).

The competition was organized by Professor Tuomas Sandholm, and the agent was created by Noam Brown, Sam Ganzfried, and Tuomas Sandholm. This article contains the author’s personal thoughts on the event. Some of the work described in this article was performed while the author was a student at Carnegie Mellon University before the completion of his PhD. The article reflects the views of the author alone and not necessarily those of Carnegie Mellon University. The work done at Carnegie Mellon University was supported by the National Science Foundation under grants IIS-1320620, IIS-0964579, and CCF-1101668, as well as XSEDE computing resources provided by the Pittsburgh Supercomputing Center.

Author’s address: sam.ganzfried@gmail.com

thereby the variance). Each human was given a partner, who played the identical hands against Claudico with the cards reversed. Polk was paired with Les, and Kim was paired with Li. The players played in two different rooms of the casino simultaneously, with one player from each of the pairings in each room. In total, the humans ended up winning the match by 732,713 chips, which corresponds to a win rate of 9.16 big blinds per 100 hands (BB/100), a common metric used to evaluate performance in poker. This was a relatively decisive win for the humans and was statistically significant at the 90% confidence level, though it was not statistically significant at the 95% level.

The chips were just a placeholder to keep track of the score and did not represent real money; the humans were paid at the end from a prize pool of $100,000 which had been donated from River’s Casino and Microsoft Research. The human with the smallest profit over the match received $10,000, while the other humans received $10,000 plus additional payoff in proportion to the profit above the lowest profit. Formally, let $x_1, x_2, x_3, x_4$ denote the profits of the four humans from highest to smallest, and let $p_i$ denote the corresponding payoffs. Then

\begin{align*}
\text{If } x_1 &> x_4 \\
n_1 &= 10,000 + 60,000 \cdot \frac{x_1 - x_4}{x_1 + x_2 + x_3 - 3x_4} \\
n_2 &= 10,000 + 60,000 \cdot \frac{x_2 - x_4}{x_1 + x_2 + x_3 - 3x_4} \\
n_3 &= 10,000 + 60,000 \cdot \frac{x_3 - x_4}{x_1 + x_2 + x_3 - 3x_4} \\
n_4 &= 10,000 \\
\text{Else } \\
n_1 &= n_2 = n_3 = n_4 = 25,000
\end{align*}

This scheme ensured that all players received at least $10,000 and that payoffs were increasing in profit, giving each human a financial incentive to try their best individually.

While this was the first man vs. machine competition for the no-limit variant of Texas hold ’em, there had been two prior competitions for the limit variant. In the limit variant all bets are of a fixed size, while in no-limit bets can be of any number.

For example, suppose human A has pocket aces and the computer has pocket kings, and A wins $5,000. This would indicate that the human outplayed the computer. However, suppose human B has the pocket kings against the computer’s pocket aces in the identical situation and the computer wins $10,000. Then, taking both of these results into account, an improved estimator of performance would indicate that the computer outplayed the human, after the role of luck in the result was significantly reduced.

The small blind (SB) and big blind (BB) correspond to initial investments, or “antes” of the players. In the match, the SB was 50 chips and the BB was 100 chips.

To put these results into some perspective, Dong Kim won the challenge described above against Nick Frame by 13.87 BB/100 (he won by $103,992 over 15,000 hands with blinds SB=$25, BB=$50), http://www.pokergurublog.com/content/donger-kim-wins-heads-challenge-against-tcfromub, and Doug Polk defeated Ben Sulsky in another high-profile challenge match by 24.67 BB/100 (he won by $740,000 over 15,000 hands with blinds SB = $100, BB = $200), http://www.pokernews.com/news/2013/10/doug-polk-defeats-ben-sulsky-16618.htm.
of chips up to the amount remaining in a player’s stack (the stacks are reset to a fixed amount of 200 big blinds at the start of each hand). Thus, the game tree for no-limit has a much larger branching factor and is significantly larger; there are $10^{165}$ nodes in the game tree for no-limit, while there are around $10^{17}$ nodes for limit [Johanson 2013]. In 2007 a program called Polaris that was created by researchers at the University of Alberta played four duplicate 500-hand matches against human professionals. The program won one match, tied one, and lost two, thus losing the match overall. In 2008 an improved version of Polaris competed against six human professionals in a second match, this time coming out victorious (three wins, two losses, and one tie). There have also been highly-publicized man vs. machine competitions for other games; for example, chess program Deep Blue lost to human expert Garry Kasparov in 1996 and beat him in 1997, and Jeopardy agent Watson defeated human champions in 2011.

Claudico is Latin for “I limp.” Limping is the name of a specific play in poker. After the initial antes have been paid, the first player to act is the small blind and he has three available actions; fold (forfeit the pot), call (match the big blind by putting in 50 chips more), or raise by putting in additional chips beyond those needed to call (a raise can be any integral amount from 200 chips up to 20,000 chips in this situation). The second option of just calling is called “limping” and has traditionally been viewed as a very weak play only made by bad players. In one popular book on strategy, Phil Gordon writes, “Limping is for Losers. This is the most important fundamental in poker—for every game, for every tournament, every stake: If you are the first player to voluntarily commit chips to the pot, open for a raise. Limping is inevitably a losing play. If you see a person at the table limping, you can be fairly sure he is a bad player. Bottom line: If your hand is worth playing, it is worth raising” [Gordon 2011]. Claudico actually limps close to 10% of its hands, and based on discussion with the human players who did analysis it seems to have profited overall from the hands it limped. Claudico also makes several other plays that challenge conventional human poker strategy; for example it sometimes makes very small bets of 10% of the pot, and sometimes very large all-in bets for many times the pot (e.g., betting 20,000 into a pot of 500). By contrast, human players typically utilize a small number of bet sizes, usually between half pot and pot.

2. AGENT ARCHITECTURE

Claudico was an improved version of an earlier agent called Tartanian7 that came in first place in the 2014 AAAI computer poker competition, beating each opposing agent with statistical significance. The architecture of that agent has been described in detail in a recent paper [Brown et al. 2015]. At a very high level, the design of the agent follows the three-step procedure depicted in Figure 1, which is the leading paradigm used by many of the strongest agents for large games.

In the first step, the original game is approximated by a smaller abstract game that hopefully retains much of the strategic structure of the initial game. The first abstractions for two-player Texas hold ’em were manually generated [Shi and Littman 2002; Billings et al. 2003]; while current abstractions are computed algorithmically [Gilpin and Sandholm 2006; 2007a; Gilpin et al. 2008; Waugh et al.
2009; Johanson et al. 2013]. For smaller games, such as Rhode Island hold 'em, abstraction can be performed losslessly, and the abstract game is actually isomorphic to the full game [Gilpin and Sandholm 2007b]. However, for larger games, such as Texas hold 'em, we must be willing to incur some loss in the quality of the modeling approximation due to abstraction.

The second step is to compute an $\epsilon$-equilibrium in the smaller abstracted game, using a custom iterative equilibrium-finding algorithm such as counterfactual regret minimization (CFR) [Zinkevich et al. 2007] or a generalization of Nesterov’s excessive gap technique [Hoda et al. 2010].

The final step is to construct a strategy profile in the original game from the approximate equilibrium of the abstracted game by means of a reverse mapping procedure. When the action spaces of the original and abstracted games are identical, this step is often straightforward, since the equilibrium of the abstracted game can be played directly in the full game. However, even in this simplified setting often significant performance improvements can be obtained by applying a nontrivial reverse mapping. Several procedures have been shown to significantly improve performance that modify the action probabilities of the abstract equilibrium strategies by placing more weight on certain actions [Ganzfried et al. 2012; Brown et al. 2015]. These post-processing procedures are able to achieve robustness against limitations of the abstraction and equilibrium-finding phases of the paradigm.

When the action spaces of the original and abstracted games differ, an additional procedure is needed to interpret actions taken by the opponent that are not allowed in the abstract game model. Such a procedure is called an action translation mapping. The typical approach for performing action translation is to

---

In general the problem of computing a Nash equilibrium (or an $\epsilon$ approximation) is challenging computationally [Chen and Deng 2006], and it is widely conjectured that no efficient algorithms exist. For the case of two-player zero-sum (i.e., competitive) games (such as two-player poker), efficient exact algorithms exist [Koller et al. 1994]; however they only scale to games with around $10^8$ states. For larger games such as abstractions of no-limit Texas hold 'em we must apply approximation algorithms that converge to equilibrium in the limit.
map the opponent’s action to a nearby action that is in the abstraction (perhaps probabilistically), and then respond as if the opponent had taken this action.

An additional crucial component of Claudico, that was not present in Tartanian7 due to a last-minute technical difficulty (though a version of it was present in prior agent Tartanian6), is an approach for real-time computation of solutions in the part of the game tree that we have reached to a greater degree of accuracy than in the offline computation, called endgame solving, which is depicted in Figure 2 [Ganzfried and Sandholm 2015]. At a high level, endgame solving works by assuming both agents follow the precomputed approximate equilibrium strategies for the trunk portion of the game prior to the endgame; then the endgame induced by these trunk strategies is solved, using Bayes’ rule to compute the input distributions of players’ private information leading into the endgame. In general, such a procedure could produce a non-equilibrium strategy profile (even if the full game has a unique equilibrium and a single endgame); for example, in a sequential version of rock-paper-scissors where player 1 acts and then player 2 acts without observing the action taken by player 1, if we fix player 1 to follow his equilibrium strategy of randomizing equally among all three actions, then any strategy for player 2 is an equilibrium in the resulting endgame, because each one yields her expected payoff 0. In particular, the equilibrium solver could output the pure strategy Rock for her, which is clearly not an equilibrium of the full game. On the other hand, endgame solving is successful in other games; for example in a game where player 1 first selects an action $a_i$ and then an imperfect-information game $G_i$ is played, we could simply solve the $G_i$ corresponding to the action $a_i$ that is actually taken, provided that the $G_i$ are independent and no information sets extend between several $G_i$. Furthermore, endgame solving has been previously demonstrated to improve performance empirically against strong computer programs in no-limit Texas hold ‘em [Ganzfried and Sandholm 2015].

We used the endgame solver to compute our strategies in real time for the final betting round of each hand, called the river.\footnote{There are (up to) four betting rounds in a hand of Texas hold ‘em poker. First both players are dealt two private cards and there is an initial round called preflop. Then three public cards are dealt and there is the flop. Then there is one more additional public card on the turn, followed by one final public card in the river betting round.} Despite the theoretical limitation of the approach, Doug Polk related to me in personal communication after the

---

competition ended that he thought the river strategy of Claudico using the endgame
solver was the strongest part of the agent.

2.1 Offline abstraction and equilibrium computation
Claudico’s action abstraction was manually generated and consisted of sizes ranging
from 0.1 pot in certain situations to all-in (wagering all of one’s remaining chips). The
information abstraction was computed using a hierarchical algorithm that first
clustered the three-card public flop boards (i.e., the three cards dealt face up in the
middle of the table for the flop round that can be observed by both players) into
public “buckets” (i.e., groupings), then clustered the private information states
for each postflop round (i.e., flop, turn, river) separately for each public bucket
(no information abstraction was performed for the preflop round) [Brown et al.
2015]. This hierarchical abstraction algorithm allowed us to apply a new scalable
distributed version of CFR [Brown et al. 2015]. We ran the equilibrium-finding
algorithm for several months on Pittsburgh’s Blacklight supercomputer using 961
cores (60 blades of 16 cores each, plus one core for the head blade, with each blade
having 128 GB RAM).

2.2 Action translation
For the action translation mapping, we used the pseudo-harmonic mapping, which
maps a bet $x$ of the opponent to one of the nearest sizes $A, B$ in the abstraction
according to the following formula, where $f(x)$ the probability that $x$ is mapped to
$A$ [Ganzfried and Sandholm 2013]:

$$f(x) = \frac{(B - x)(1 + A)}{(B - A)(1 + x)}.$$ 

This mapping was derived from analytical solutions of simplified poker games and
has been demonstrated to outperform prior approaches in terms of exploitability in
simplified games, as well as the best prior approach in terms of empirical perfor-
mance against no-limit Texas hold ’em agents. The mapping also satisfies several
axioms and theoretical properties that the best prior mappings do not satisfy, for
example it is Lipschitz continuous in $A$ and $B$, and therefore robust to small changes
in the actions used in the action abstraction.

As an example to demonstrate the operation of the algorithm, suppose the op-
ponent bets 100 into a pot of 500, and that the closest sizes in our abstraction are
to “check” (i.e., bet 0) or to bet 0.25 pot: so $A = 0$ and $B = 0.25$. Plugging these
in gives $f(x) = \frac{1}{5} = 0.167$. This is the probability we map his bet down to 0 and
interpret it as a check. So we pick a random number in $[0,1]$, and if it is above $\frac{1}{5}$
we interpret the bet as 0.25 pot, and otherwise as a check.

2.3 Post-processing
We used additional post-processing techniques to round the action probabilities
that had been computed by the offline equilibrium-finding algorithm [Ganzfried
et al. 2012]. We used a generalization of the prior approach that applied a differ-
ext rounding threshold for each betting round (i.e., action probabilities below the
threshold were rounded to zero and then all probabilities were renormalized), with
a more aggressive (i.e., larger) threshold used for the later betting rounds, since
the equilibrium-finding algorithm obtains worse convergence for those rounds due to having fewer samples from each state in that part of the game tree.\footnote{The counterfactual regret minimization algorithm works by repeatedly sampling private and public information and updating “regrets” for each action at each information set during self play (i.e., while running the same algorithm for the other player).} We did not apply any post-processing for ourselves on the river when using the endgame solver, and assumed neither agent used any post-processing in the generation of the trunk strategies used as inputs to the endgame solver.\footnote{It may seem somewhat strange that we applied post-processing for our own play, but assumed that no post-processing was applied for the trunk strategies entering the endgame, and that this may be problematic due to the mismatch between our own strategy and the model of it entering the endgame. We chose to do this because the endgame solving approach can be less robust if the input strategies have weight on only a small number of hands (as an extreme example, if all the weight was on one hand, then the endgame solver would assume that the other agent knew our exact hand, and the solution would require us to play extremely conservatively). The approach is much more robust if we include a small probability on many different hands before the post-processing was applied. We believed that the gain in robustness outweighed the limitation of the mismatch (in addition to the reasons given above, we already expect there to be a mismatch between the input trunk strategy for the opponent, which is based off our offline equilibrium computation, and his own actual strategy, and thus we would not be removing this mismatch completely even if we eliminated it for our own strategy).}

2.4 Endgame solving

The endgame solving algorithm consists of several steps [Ganzfried and Sandholm 2015]. First, the joint hand-strength input distributions are computed by applying Bayes’ rule to the precomputed trunk strategies, utilizing a recently developed technique that requires only a linear number of lookups in the large strategy table (while the naïve approach requires a quadratic number of lookups and is impractical). Then the equity is computed for each hand, given these distributions.\footnote{The equity of a hand against a distribution for the opponent is the probability of winning plus one half times the probability of tying.} Then hands are bucketed separately for each player based on the computed equities for the given situation by applying an information abstraction algorithm. Finally an exact Nash equilibrium is computed in the game corresponding to this information abstraction and an action abstraction that had been precomputed for the specific pot and stack size of the current hand. All of this computation was done in real time during gameplay. To compute equilibria within the endgames, we used Gurobi’s parallel linear program solver [Inc. 2014] to solve the sequence-form optimization formulation [Koller et al. 1994].

3. PROBLEMATIC HANDS

Several notable hands stood out during the course of the competition that highlighted weaknesses of the agent, which have been singled out in a thread that was devoted entirely to the competition on the most popular poker forum, the Two Plus Two Poker Forum.\footnote{The thread discussing the event has 232,252 views and 1,609 posts as of September 23, 2015, \url{http://forumserver.twoplustwo.com/29/news-views-gossip-sponsored-online-poker-report/wcgrider-dong-jason-les-bjorn-li-play-against-new-hu-bot-1526750/}. Here are links to some of the posts in the thread that relate to the hands described: hand 1
}
In one hand, we had A4s (ace and four of the same suit) and folded preflop after we had put in over half of our stack (the human opponent had 99). This is regarded as a bad play, since we would only need to win around 25% of the time against the opponent’s distribution for a call to be profitable at this point (we win about 33% of the time against the hand he had). The problem was that our translation mapping mapped the opponent’s raise down to a smaller size, which caused us to look up a strategy for ourselves that had been computed thinking that the pot size was much smaller than we thought it was (we thought we had invested around 7,000 when we had actually invested close to 10,000—recall that the starting stacks are 20,000). These translation issues can get magnified further as the hand develops if we think we have bet a percentage (e.g., $\frac{1}{3}$) of the (correct) size of the pot, while the strategies we have precomputed assumed a different size of the pot.

In another hand we had KT and folded to an all-in bet on the turn after putting in about $\frac{3}{4}$ of our stack despite having top pair and a flush draw (there were three diamonds on the board and we had the king of diamonds; the opponent actually had A2 with the ace of diamonds, for a better flush draw but worse hand due to us having a pair already). The issue for this hand was that the human made a raise on the flop which was slightly below the smallest size we had in our abstraction in that situation, and we ended up mapping it down to just a call (it was just mapped down with around 3% probability in that situation, and so we ended up getting pretty “unlucky” that we mapped it in the “wrong” direction). This ended up causing us to think we had committed far fewer chips to the pot at that point than we actually had.

The problem in these hands was not due simply to a flaw in the action translation mapping, or even to a flaw in the action abstraction (though of course improvements to those would be very beneficial as well); even if we had used a different translation mapping and/or used different action sizes in the abstraction, we would still have potentially sizable gaps between certain sizes of the abstraction due to the fact that we can only select so many to keep the abstraction sufficiently small so that it can be solved within time and memory limits. That means that, given the current paradigm, we will necessarily have to map bets to sizes somewhat far away with some probability, which will cause our perception of the pot size to be incorrect, as these hands indicate. This is called the “off-tree problem,” which has received very little study thus far. Some agents, such as versions of the agent from the University of Alberta, attempt to mitigate this problem by specifically taking actions aimed to get us back on the tree (e.g., making a bet that we would not ordinarily make to correct for the pot size disparity). However, this is problematic too, as it requires us to take an undesirable action. The endgame solving approach provides a solution to this problem by inputting the correct pot size to the endgame solving algorithm, even if this differs from our perception of it at that point due to the opponent having taken an action outside of the action abstraction. In general,
real-time endgame solving could correct for many misperceptions in game state information that have been accumulated along the course of game play; however, this would not apply to the preflop, flop, and turn rounds, where we are not using endgame solving. Thus it is necessary to explore additional approaches to this problem; improved algorithms for real-time computation for the earlier rounds is a potentially promising direction, and perhaps new approaches can also be developed for addressing the off-tree problem independently of endgame solving.

We went over the log files for these two specific hands with Doug Polk in person after the competition had ended, and he agreed that our plays in both hands were reasonable had the pot size been what our computed strategies perceived it to be at that point. Of course, we both agree that the hands were both major mistakes if you include the misperception of the pot size. Even though these were only low probability mistakes due to the randomization outcome selected by the translation mapping, these types of mistakes can become a significant liability in aggregate, particularly when playing against humans who are aware of them and actively trying to exploit them. Doug alluded to this point as well in an interview after the competition. Based on Doug’s interview and subsequent conversations it seems that he views this as Claudico’s biggest weakness, and it will be interesting to see what improvements can be found, and whether those can be exploited in turn by good countermeasures.

(3) In one other problematic hand, we made a large all-in bet (of around 19,000) into a relatively small pot of around 1700. There were three of a suit (spades) on the board, and we had a very weak hand without a fourth spade (so our bet was a “bluff,” hoping the opponent would fold a stronger hand). The problem is not that we made a large bet per se, or even that we did it with a very weak hand; extremely large bets are correct and part of equilibrium strategy in certain situations, and in such situations they must be made with some weak hands as bluffs to balance with the very strong “value” hands or else our strategy would be too predictable (if we never bluffed, then the opponent would just fold everything except his hands that beat half of our value hands, and then the bets with the bottom half of our value hands would be unprofitable). Thus, making large bets as bluffs is needed in certain situations. The problem is that certain


12 As one example, Ankenman and Chen describe a game called the “Clairvoyance Game” where player 1 is dealt a winning/losing hand with probability \( \frac{1}{2} \) each, and is allowed to bet any amount up to initial stack \( n \) into a pot of 1; then player 2 can call or fold [Ankenman and Chen 2006]. (Player 1 knows whether he has a winning or losing hand, while player 2 does not know player 1’s hand.) They analytically solve for the unique Nash equilibrium of the game, and it has player 1 betting all-in for \( n \) with his winning hand, and betting all-in with some probability with his losing hand, and checking with some probability (the probability is selected to make player 2 indifferent between “bluffing” and checking with his losing hand). This solution holds regardless of the stack size \( n \); so even if \( n = 1,000,000 \), it would be optimal for player 1 to bet all-in for 1,000,000 to win a pot of 1 (a sketch of Ankenman and Chen’s argument with the computed equilibrium strategies also appears in [Ganzfried and Sandholm 2013]). Thus, it is clear that at least in certain situations extremely large bets, both with strong and weak hands, are part of optimal strategies.
hands are much better suited for them than others. For example, suppose the board was JsTs4sKcQh, and suppose we could have 3c2c (three and two of clubs) vs. 3s2c (three of spades and two of clubs). Both hands are extremely weak (they produce the worst possible five-card hand); however, if we have the 3s, it actually has a subtle and very significant benefit: it significantly reduces the probability that the opponent holds an extremely strong hand (e.g., an ace-high or king-high flush) because several of the hands that would constitute that strength would contain that card, e.g., As3s and Ks3s. Thus, this would make a much better choice for our hand to make a large bet with, since he is less likely to have a hand strong enough to call, making the bluff bet more effective. Our endgame-solving algorithm described in Section 2.4 takes this “card removal” factor into account to an extent, since the equities are computed for each hand against the distribution the opponent could hold given that hand; however, this does not fully take into account the card removal effect. For example, the 3c2c and 3s2c hands would both have the lowest possible equity (it would be slightly above zero only because of possible ties), and would be necessarily grouped into the same bucket by our endgame information abstraction algorithm (the worst bucket) despite the fact that they have very different card removal properties.

Doug Polk said that he thought the river strategy using the endgame solver overall was the strongest part of Claudico; however, he thought that utilizing the large betting sizes without properly accounting for card removal was actually a significant weakness, since we would be bluffing with non-optimal hands. We came to this conclusion ourselves as well during the competition, and for this reason decided to take out the large bets for ourselves from the endgame solver partway through the competition, since this issue is most problematic for those bet sizes (for smaller bet sizes, card removal is still important, but significantly less important since we are not just trying to “block” the opponent from having a small number of extremely strong hands, since he will be calling with many more hands). Interestingly, Dong Kim told me after the competition that they had conducted analysis and we were actually profiting on the large bet sizes during the time we used them, despite the theoretical issue described above. I think everyone agrees that massive “overbets” are part of full optimal strategies, and likely underutilized by even the best human players. But card removal is also particularly important for these sizes, and I think for an agent to use them successfully an improved algorithm for dealing with blockers/card removal would need to be developed, though I am still quite curious how well we would have performed if we continued with those sizes included in the agent.

4. CONCLUSION

It is one thing to evaluate a poker agent against other computer agents, who largely also play static approximations of equilibrium strategies; it is another to compete against the strongest human specialists, who will adapt and attempt to capitalize on even the smallest perceived weaknesses. This was the first time a no-limit Texas hold ’em agent has competed against human players of this caliber, and we really had no idea what to expect entering the competition, as previously all of our experiments had been against computer agents from the AAAI Annual Com-
puter Poker Competition. We learned many valuable lessons that will be pivotal in developing improved agents going forward. We have highlighted the two most important avenues for future research. The first is to develop an improved approach for the “off-tree” problem where we make a mistake due to a misperception of the actual size of the pot after translating an action for the opponent that is not in our action abstraction. We have outlined promising agendas for attacking this problem, including improved action abstraction and translation algorithms, novel approaches for real-time computation that address the portion of the game prior to the final round, and entirely new approaches specifically geared at solving the off-tree problem independently of the other problems. And the second is to develop an improved approach for information abstraction that better accounts for card removal/“blockers” (i.e., that accounts for the fact that us having certain cards in our hand modifies the probability of the opponent having certain hands). This issue is most problematic within the information abstraction algorithm for the endgame, where the card removal effect is most significant due to the distributions for us and the opponent being the most well defined (i.e., there is no more potential remaining in the hand due to uncertainty of public cards, and this relative certainty will likely cause the distributions to put positive weight on fewer hands), and it limits our ability to utilize large bet sizes, which have been demonstrated to be optimal in certain settings. Of course, it would be beneficial to develop an improved information abstraction algorithm that accomplishes this in the part of the game prior to the endgame as well.

At first glance it may appear that these issues are purely pragmatic and specific to poker. While one of the main goals is certainly to produce a poker agent that can beat the strongest humans in two-player no-limit Texas hold ’em, there are deeper theoretical questions related to each component of the agent that has been described. Endgame solving has been proven to have theoretical guarantees in certain games while it can lead to strategies with high exploitability in others (even if the full game has a single Nash equilibrium and just a single endgame is considered) [Ganzfried and Sandholm 2015]. It would be interesting to prove theoretical bounds on its performance on interesting game classes, perhaps classes that include variants of poker. Empirically the approach appears to be very successful on poker despite its lack of theoretical guarantees. Recently an approach has been developed for game decomposition that has theoretical guarantees [Burch et al. 2014], however from personal communication with the authors I have learned that the approach performs worse empirically than our approach that does not have a worst-case guarantee.

The main abstraction algorithms that have been successful in practice are heuristic and have no theoretical guarantees. It is extremely difficult to prove meaningful theoretical guarantees when performing such a large degree of abstraction, e.g., approximating a game with $10^{165}$ states by one with $10^{14}$ states. There has been some recent work done on abstraction algorithms with theoretical guarantees, though that work does not scale to games nearly as large as no-limit Texas hold ’em. One line of work performs lossless abstraction, that guarantees that the abstract game is exactly isomorphic to the original game [Gilpin and Sandholm 2007b]. This work has been applied to compute equilibrium strategies in Rhode Island hold ’em, a
medium-sized (3.1 billion nodes) variant of poker. Recent work has also presented
the first lossy abstraction algorithms with bounds on the solution quality [Kroer
and Sandholm 2014]. However, the algorithms are based on integer programming
formulations, and only scale to a tiny poker game with a 5-card deck. It would be
very interesting to bridge this gap between heuristics that work well in practice for
large games with no theoretical guarantees, and the approaches with theoretical
guarantees that have more modest scalability.

Scalable algorithms for computing Nash equilibria have diverse applications, in-
cluding cybersecurity (e.g., determining optimal thresholds to protect against phish-
ing attacks), business (e.g., auctions and negotiations), national security (e.g., com-
puting strategies for officers to protect airports), and medicine. For medicine, algo-
rithms that were created in the course of research on poker [Johanson et al. 2012]
have been applied to compute robust policies for diabetes management [Chen and
Bowling 2012]; recently it has been proposed that equilibrium-finding algorithms
are applicable to the problem of treating diseases such as the HIV virus that can
mutate adversarially [Sandholm 2015].

For the pseudo-harmonic action translation mapping, in addition to showing that
it outperforms the best prior approach in terms of exploitability in several games,
we have also presented several axioms and theoretical properties that it satisfies;
for example, it is Lipschitz continuous in $A$ and $B$, and therefore robust to small
changes in the actions used in the action abstraction [Ganzfried and Sandholm
2013]. Another mapping that has very high exploitability in several games also
satisfies these axioms, and further investigation can lead to deeper theoretical un-
derstanding of this problem and potentially new improved approaches.

Even the post-processing approaches, which appear to be purely heuristic, have
interesting theoretical open questions. For example, it has been shown that purifi-
cation (i.e., selecting the highest-probability action with probability 1) leads to an
improved performance in uniform random $4 \times 4$ matrix games using random $3 \times 3$
abstractions when playing against the Nash equilibrium of the full $4 \times 4$ game for
the opponent [Ganzfried et al. 2012]. These results were based off simulations that
were statistically significant at the 95% confidence level, and it would be interest-
ing to provide a formal proof. Furthermore, that paper provided a conjecture for
the specific supports of the games for which the approach would improve or not
change performance, which was also based on statistically-significant simulations.
It would be interesting to prove this formally as well, and to generalize the results
to games of arbitrary size. On a broader level, there is relatively little theoretical
understanding for why the post-processing approaches—which one would expect to
make the strategies more predictable—have been shown to be consistently success-
ful. Surprisingly, the improvements in empirical performance do not necessarily
come at the expense of worst-case exploitability, and a degree of thresholding has
been demonstrated to actually reduce exploitability for a limit Texas hold ‘em
agent [Ganzfried et al. 2012].

REFERENCES


Billings, D., Burch, N., Davidson, A., Holte, R., Schaeffer, J., Schauenberg, T., and
Reflections on the First Man vs. Machine No-Limit Texas Hold ’em Competition


the AAAI Conference on Artificial Intelligence (AAAI). Senior Member Track, Blue Skies Subtrack.


Two Desirable Fairness Concepts for Allocation of Indivisible Objects under Ordinal Preferences

HARIS AZIZ
and
SERGE GASPERS
and
SIMON MACKENZIE
and
TOBY WALSH
Data61 and UNSW

Fair allocation of indivisible objects under ordinal preferences is an important problem. Unfortunately, a fairness notion like envy-freeness is both incompatible with Pareto optimality and is also NP-complete to achieve. To tackle this predicament, we consider a different notion of fairness, namely proportionality. We frame allocation of indivisible objects as randomized assignment but with integrality requirements. We then use the stochastic dominance relation to define two natural notions of proportionality. Since an assignment may not exist even for the weaker notion of proportionality, we propose relaxations of the concepts — optimal weak proportionality and optimal proportionality. For both concepts, we propose algorithms to compute fair assignments under ordinal preferences. Both new fairness concepts appear to be desirable in view of the following: they are compatible with Pareto optimality, admit efficient algorithms to compute them, are based on proportionality, and are guaranteed to exist.

Categories and Subject Descriptors: F.2.2 [Analysis of Algorithms and Problem Complexity]: Nonnumerical Algorithms and Problems; I.2.11 [Distributed Artificial Intelligence]: Multiagent Systems; J.4 [Computer Applications]: Social and Behavioral Sciences—Economics

General Terms: Theory, Algorithms, Economics

Additional Key Words and Phrases: Fair allocation, Multiagent resource allocation

1. INTRODUCTION

The principled allocation of resources is one of most pressing problems faced by society [see, e.g., Bezúková and Dani, 2005, Bouveret and Lemaître, 2014]. Within the rich field of resource allocation, a typical allocation setting has a set of agents \( N = \{1, \ldots, n\} \), a set of objects \( O = \{o_1, \ldots, o_m\} \) with \( m \geq n \) and each agent \( i \in N \) expressing complete and transitive ordinal preferences \( \succsim_i \) over \( O \). The goal is to allocate all the objects in \( O \) to the agents in a fair manner. Since eliciting preferences over bundles of objects requires exponential time, we only assume that agents express preferences over individual objects. These preferences over objects can involve indifference (\( \succ_i \) denotes strict preference whereas \( \sim_i \) denotes indifference). The setting is referred to as the assignment problem or the house allocation problem [see, e.g., Baumeister et al., 2014, Bouveret et al., 2010, Gärdenfors, 1973,
The model is applicable to many resource allocation or fair division settings in which multiple objects may be allocated to agents.

**Example 1 Assignment problem.**

1: $a_1 \succ_1 a_2 \succ_1 a_3 \succ_1 a_4 \succ_1 a_5$

2: $a_2 \sim_2 a_3 \succ_2 a_1 \sim_2 a_4 \sim_2 a_5$

Since the goal is to identify fair allocations, there is a need to formalize what fairness entails. Two of the most fundamental concepts of fairness are envy-freeness and proportionality. Envy-freeness requires that no agent prefers another agent’s allocation. Proportionality requires that each agent should get an allocation that gives him at least $1/n$ of the utility that he would get if he got all the objects. When agents’ ordinal preferences are known but utility functions are not given, then ordinal notions of envy-freeness and proportionality need to be formulated. Since envy-freeness is defined by comparing agents’ allocations, in order to reason about envy-freeness we need to make some assumptions on how preferences over objects are extended to preferences over objects. One basic assumption we can make is that of responsiveness: if an agent gets an extra object or one of his objects is replaced by a strictly more preferred object, he is happier. Based on the assumption of responsiveness, one can define weak envy-freeness (another agent’s allocation is not strictly more preferred) and strong envy-freeness (one’s allocation is weakly preferred over others’ allocations). Not only are both notions incompatible with Pareto optimality but it is NP-complete to check whether an envy-free assignment exists [Aziz et al., 2014, Bouveret et al., 2010]. In view of this, we seek a fairness concept that satisfies the following requirements: (1) captures a natural fairness goal along similar lines as envy-freeness or proportionality, (2) guaranteed to exist, (3) efficiently computable\(^1\), and (4) compatible with Pareto optimality.

2. **Proportionality Based on Stochastic Dominance**

We take proportionality as a starting point and can define an ordinal version of proportionality. On face value, proportionality appears to be based on cardinal utilities. Indeed one can superimpose cardinal utilities consistent with the ordinal preferences and check whether a proportional assignment exists. However checking whether a proportional assignment exists for cardinal utilities is also NP-complete [Demko and Hill, 1988]. We adopt instead a different perspective in which we define ordinal notions of proportionality by first considering discrete assignments as special kind of random assignments in which each agent gets a fraction (probability) of getting each object as follows. We will still compute fair discrete assignments but will do so via comparisons with fractional assignments.

A fractional assignment $p$ is a $(n \times m)$ matrix $[p(i)(o_j)]$ such that $p(i)(o_j) \in [0, 1]$ for all $i \in N$, and $o_j \in O$, and $\sum_{i \in N} p(i)(o_j) = 1$ for all $j \in \{1, \ldots, m\}$. The value $p(i)(o_j)$ represents the probability of object $o_j$ being allocated to agent $i$. Each row $p(i) = (p(i)(o_1), \ldots, p(i)(o_m))$ represents the allocation of agent $i$. The columns correspond to the objects $o_1, \ldots, o_m$. A fractional assignment is discrete if $p(i)(o) \in \{0, 1\}$ for all $i \in N$ and $o \in O$.

\(^1\)The input is agents expressing preferences over objects. Hence we will say that an algorithm is polynomial-time if it runs in time polynomial in the number of agents and objects.
We then use the stochastic dominance relation [see e.g., Aziz et al., 2013] to compare fractional allocations. Informally, an agent ‘SD-prefers’ one allocation over another if for each object $o$, the former allocation gives the agent as many units of objects that are at least preferred as $o$ as the latter allocation. More formally, given two fractional assignments $p$ and $q$, $p(i) \succ_{SD}^S q(i)$, i.e., agent $i$ SD prefers allocation $p(i)$ to allocation $q(i)$ if

$$\sum_{o_j \in \{o_k : o_k \succ_i o\}} p(i)(o_j) \geq \sum_{o_j \in \{o_k : o_k \succ_i o\}} q(i)(o_j)$$

for all $o \in O$.

Agent $i$ strictly SD prefers $p(i)$ to $q(i)$ if $p(i) \succ_{SD}^S q(i)$ and $\neg[q(i) \succ_{SD}^S p(i)]$. SD can also be viewed from a utility perspective which underlines its fundamental nature: an agent prefers one allocation over another with respect to the SD relation if he gets at least as much utility from the former allocation as the latter for all cardinal utilities consistent with the ordinal preferences.

Based on SD, one can define two fairness notions [Aziz et al., 2014]. In particular, we define weak SD proportionality as requiring that no agent strictly prefers the allocation in which $1/n$ of each object is obtained to his own allocation. We define SD proportionality as requiring that each agent weakly SD-prefers his allocation over the allocation in which $1/n$ of each object is obtained. An assignment $p$ satisfies weak SD proportionality if no agent strictly SD prefers the uniform assignment to his allocation: $\neg[(1/n, \ldots, 1/n) \succ_{SD}^S p(i)]$ for all $i \in N$. An assignment $p$ satisfies SD proportionality if each agent SD prefers his allocation to the allocation under the uniform assignment: $p(i) \succ_{SD}^S (1/n, \ldots, 1/n)$ for all $i \in N$.

SD proportionality and weak SD proportionality are not only desirable fairness concepts but they are also computationally more tractable than ordinal notions of envy-freeness.

**Theorem 1 [Aziz et al., 2014].** We can check in polynomial time whether a discrete SD proportional assignment exists even if agents are allowed to express indifference between objects. For a constant number of agents, we can check in polynomial time whether a weak SD proportional discrete assignment exists.

A possible criticism of weak SD proportionality and SD proportionality concepts is that even the weaker of the two is not achievable in general. Consider the following example. The weak SD proportionality constraint is violated for the agent who gets at most one object.

**Example 2.** Assume that the preferences of the agents are as follows.

1: $a_1 \sim_1 a_2 \sim_1 a_3$

2: $a_1 \sim_2 a_2 \sim_2 a_3$

3. **OPTIMAL PROPORTIONALITY AND OPTIMAL WEAK PROPORTIONALITY**

When a weak SD proportional assignment does not exist, we would still like to allocate the objects in a principled manner. We relax weak SD proportionality and SD proportionality to propose optimal proportionality and optimal weak proportionality.

**Definition 1** Optimal proportionality [Aziz et al., 2015]. We say that an assignment satisfies $1/\alpha$ proportionality if $p(i) \succ_{SD}^{SD} (1/\alpha, \ldots, 1/\alpha)$ for all $i \in N$. We note
that $1/n$-proportionality is equivalent to SD proportionality. An assignment satisfies optimal proportionality if $p(i) \succeq_{SD}^{1/\alpha} (1/\alpha, \ldots, 1/\alpha)$ for all $i \in N$ for the smallest possible $\alpha$. We will refer to the smallest such $\alpha$ as $\alpha^*$ and call $1/\alpha^*$ as the optimal proportionality value.

Definition 2 Optimal weak proportionality [Aziz et al., 2015]. Just like the concept of SD proportionality can be used to define optimal proportionality, weak SD proportionality can be used to define optimal weak proportionality. We say that an assignment satisfies $1/\beta$ weak proportionality if $(1/\beta, \ldots, 1/\beta) \not\succeq^{SD}_i p(i)$ for all $i \in N$. We note that $1/n$ weak proportionality is equivalent to weak SD proportionality. An assignment satisfies optimal weak proportionality if $(1/\beta, \ldots, 1/\beta) \not\succeq^{SD}_i p(i)$ for all $i \in N$ for the infimum of the set $\{ \beta \mid \exists a 1/\beta$ weak proportional assignment $\}$. We will refer to the infimum as $\beta^*$ and call $1/\beta^*$ as the optimal weak proportionality value.

Theorem 1 can be generalized from $1/n$ proportionality to $1/\alpha$ proportionality for any value of $\alpha$. The algorithm can be used to check the existence of a $1/\alpha$ proportional assignment for different values of $\alpha$. However, among other cases, if $m < n$, we then know that a $1/\alpha$ proportional assignment does not exist for any finite value of $\alpha$. We show that $\alpha^*$ is finite if and only if there exists an assignment in which each agent gets one of his most preferred objects. Since $\alpha$ is a positive real in the interval $(0, \infty]$, it may appear that even binary search cannot be used to find the optimal proportional assignment in polynomial time. Interestingly, we only need to check a polynomial number of values of $\alpha$ to find the optimal proportional assignment.

Theorem 2 [Aziz et al., 2015]. An optimal proportional assignment can be computed in polynomial time.

We point out that an SD proportional assignment (if it exists) is an optimal proportional assignment. Moreover, even if an SD proportional assignment does not exist, an optimal proportional assignment suggests a desirable allocation of objects. For example, for the preference profile in Example 2, we observed that there exists no weak SD proportional assignment. On the other hand, the assignment that gives two objects to one agent and one object to the other is an optimal proportional assignment where the optimal proportionality value is $1/3$.

In a similar approach as for optimal proportionality, for a constant number of agents, it can be checked in polynomial time whether a $1/\beta$ weak proportional discrete assignment exists. We also show that for any assignment setting, $\beta^* \geq 1$ and is finite if and only if $m \geq n$.

Theorem 3 [Aziz et al., 2015]. If the number of agents is constant, an optimal weak proportional assignment can be computed in polynomial time.

We note that whereas an SD proportional assignment is an optimal proportional assignment, a weak SD proportional assignment may not be an optimal weak proportional assignment.
Example 3. Assume that the preferences of the agents are as follows.

1: \( o_1 \succ_1 o_2 \succ_1 o_3 \succ_1 o_4 \succ_1 o_5 \)

2: \( o_2 \sim_2 o_3 \succ_2 o_1 \sim_2 o_4 \sim_2 o_5 \)

Note that the assignment \( p \) that gives \( \{o_2, o_3\} \) to agent 1 and the other objects to agent 2 is weak SD proportional. In fact it is not only 1/2 weak proportional but \((3/5 - \epsilon)\) weak proportional where \( \epsilon > 0 \) is arbitrarily small. It is not 1/3 weak proportional for \( 1/\beta < 3/5 \). We now consider an assignment \( q \), that gives \( \{o_1\} \) to agent 1 and the other objects to agent 2. But \( q \) is \((1 - \epsilon)\) weak proportional where \( \epsilon > 0 \) is arbitrarily small. This shows that a weak SD proportional discrete assignment may not be an optimal weak proportional assignment.

4. CONCLUSIONS AND OPEN PROBLEMS

In this note, we highlighted two desirable fairness concepts that have recently been proposed [Aziz et al., 2015]. The most interesting remaining problem is checking whether there exists a polynomial-time algorithm for computing an optimal weak proportional assignment when the number of agents is not constant.

Acknowledgments

We thank the reviewers of the AAMAS 2014 conference and the Artificial Intelligence journal for their helpful reviews. The work is funded by the Australian Government through the Department of Communications and the Australian Research Council through the ICT Centre of Excellence Program. Serge Gaspers is the recipient of an Australian Research Council Discovery Early Career Researcher Award (project number DE120101761) and a Future Fellowship (project number FT140100048).

REFERENCES


Algorithms as Mechanisms:  
The Price of Anarchy of Relax-and-Round

PAUL DÜTTING, ETH Zürich  
and  
THOMAS KESSELHEIM, Max-Planck-Institut für Informatik  
and  
ÉVA TARDOS, Cornell University

We show that algorithms that follow the relax-and-round paradigm translate approximation guarantees into Price of Anarchy guarantees, provided that the rounding is oblivious and the relaxation is smooth. We use this meta result to obtain simple, near-optimal mechanisms for a broad range of optimization problems such as combinatorial auctions, the maximum traveling salesman problem, and packing integer programs. In each case the resulting mechanism matches or beats the performance guarantees of known mechanisms.

Categories and Subject Descriptors: F.2 [Theory of Computation]: Analysis of Algorithms and Problem Complexity; J.4 [Computer Applications]: Social and Behavioral Sciences—Economics

General Terms: Algorithms, Economics, Theory

Additional Key Words and Phrases: Algorithmic Game Theory, Price of Anarchy, Smoothness, Black-Box Reductions

1. INTRODUCTION

The “classic” approach to mechanism design is to devise mechanisms that incentivize truthful reporting. For settings where the private information of each agent consists of a single number this boils down to finding allocation rules that are monotone in each agent’s private value. Many natural approximation algorithms either satisfy this monotonicity constraint or can be tweaked to satisfy it.

For more general settings, where the private information is multi-dimensional, designing truthful mechanisms is an intriguing task, and no general transformation for turning approximation algorithms into truthful mechanisms with comparable performance is known. The most general positive algorithmic results in this context are so-called black-box reductions. These turn algorithms of a certain class into truthful mechanisms by making polynomially many calls to the approximation algorithm. Examples range from the early work of Lavi and Swamy [2005] to the more recent work of Dughmi and Roughgarden [2014].

We consider here an alternative approach that we refer to as “algorithms as mechanisms”. This approach is different from the black-box reduction approach in that it takes an approximation algorithm as is and couples it with a simple payment rule, such as pay-your-bid. Of course, this will typically not lead to a mechanism where participants want to report their true values. Instead we seek to understand
which approximation algorithms guarantee that all equilibria or even all learning outcomes of the resulting mechanism are close to optimal.

In fact, in a landmark result Lucier and Borodin [2010] showed that greedy algorithms have this property: Any equilibrium of a greedy algorithm that is an $\alpha$-approximation algorithm is within $O(\alpha)$ of the optimal solution.

Our main result is to show that the common design principle of relaxation and rounding also approximately preserves the approximation guarantee as a Price of Anarchy guarantee provided that the relaxation is smooth and the rounding process is oblivious (more on this below). The canonical example of this approach are integer linear programs that are relaxed to a fractional domain, then the relaxation is solved to optimality and converted into an integer solution via randomized rounding. Other examples that follow this pattern come from semi-definite programming or involve relaxing one combinatorial problem to another.

Our meta result has—as we show—far-reaching consequences in mechanism design: It leads to novel, simple, yet near-optimal mechanisms for sparse packing integer programs such as multi-unit auctions and generalized matching, for the maximum traveling salesman problem, for combinatorial auctions and for single source routing problems. In all cases we obtain Price of Anarchy bounds that match or beat known Price of Anarchy guarantees, or they are the first non-trivial guarantees for the respective problem.

2. MAIN RESULT

Our main result concerns the algorithmic blueprint of relaxation and rounding (see, e.g., [Vazirani 2001]). In this approach a problem $\Pi$ is relaxed to a problem $\Pi'$, with the purpose of rendering exact optimization computationally tractable. Having found the optimal relaxed solution $x'$, another algorithm derives a solution $x$ to the original problem. This process is called rounding.

A (possibly randomized) rounding scheme $r$ for translating a solution $x'$ to the relaxed problem $\Pi'$ into a solution $x = r(x')$ to the original problem $\Pi$ is $\alpha$-approximate oblivious, where $\alpha \geq 1$, if the rounding depends on the solution $x'$ only and for all possible valuation profiles each agent is guaranteed to get, in expectation, a $1/\alpha$-fraction of the value that it would have had for the solution to the relaxed problem.

Clearly an $\alpha$-approximate oblivious rounding scheme, when combined with optimally solving the relaxed problem, leads to an approximation ratio of $\alpha$. We show that it also approximately preserves the Price of Anarchy of the relaxation assuming the relaxation is smooth. We focus on pay-your-bid mechanisms for concreteness. Our result actually applies to a broad range of mechanisms and can also be extended to include settings where the relaxation is not solved optimally.

All our bounds go through smoothness [Roughgarden 2009; 2012; Syrgkanis and Tardos 2013] and therefore apply to a broad range of equilibrium concepts.

**Theorem 2.1.** If the pay-your-bid mechanism $M$ for problem $\Pi$ is obtained by solving the relaxation $\Pi'$ optimally and applying an $\alpha$-approximate oblivious rounding scheme and the pay-your-bid mechanism $M'$ for problem $\Pi'$ that solves the relaxation optimally has a Price of Anarchy of $\beta$ via smoothness, then the mechanism $M$ has a Price of Anarchy of $2\alpha\beta$ via smoothness.

In most applications, including the ones below, the factor 2 loss from the general framework can be avoided.

3. APPLICATIONS

Our result significantly broadens the algorithmic toolbox for designing mechanisms with near-optimal equilibria. We use the richer set of tools to obtain novel mechanisms for a broad range of optimization problems. We note that in all of our applications, it is important to use the relaxation to show smoothness of the problem. For example, optimally solving the original (integer) problem would give a very high Price of Anarchy.

(1) Combinatorial Auctions. We use the ingenious rounding scheme by Feige [2009] to obtain a Price of Anarchy of $4e/(e-1)$ for fractionally subadditive valuations. For the more general class of MPH-$k$ valuations defined in [Feige et al. 2014] we obtain a Price of Anarchy of $O(k^2)$.

(2) Sparse Packing Integer Programs. Here we use the rounding scheme by Bansal et al. [2010] to show a Price of Anarchy of $O(d^2)$ when the column sparsity is $d$. This result applies to multi-unit auctions with general valuations or the generalized assignment problem (where $d = 1$) or combinatorial auctions with bounded bundle size (where $d$ is the maximum bundle size).

(3) Maximum Traveling Salesman Problem. We show that the classic algorithm by Fisher et al. [1979] that relaxes the problem to finding a cycle cover yields a Price of Anarchy of 12. We also show how this bound can be improved to 9 by using the algorithm of Paluch et al. [2012].

(4) Single-Source Unsplittable Flow. We adapt the “original” randomized rounding algorithm of Raghavan and Thompson [1987; 1988] to obtain a Price of Anarchy of $2(1 + \epsilon)$ for high enough capacities.

4. FUTURE WORK

It is still largely open whether reductions similar to the one presented here also apply to other classes of algorithms or whether greedy and relax-and-round algorithms are the only classes that admit such a result. More generally, one could try to characterize which algorithms lead to mechanisms with small Price of Anarchy. A first step towards such a characterization was recently made by Dütting and Kesselheim [2015].

REFERENCES


unifying hierarchy of valuations with complements and substitutes. Electronic Colloquium on
Computational Complexity 21, 103.
LAVI, R. AND SWAMY, C. 2005. Truthful and near-optimal mechanism design via linear program-
Symposium on Discrete Algorithms. 537–553.
PALUCH, K. E., ELBASSIONI, K. M., AND VAN ZUYLEN, A. 2012. Simpler approximation of the max-
imum asymmetric traveling salesman problem. In Proc. of the 29th International Symposium
on Theoretical Aspects of Computer Science. 501–506.
RAGHAVAN, P. 1988. Probabilistic construction of deterministic algorithms: Approximating pack-
ing integer programs. Journal on Computer Systems Sciences 37, 2, 130–143.
RAGHAVAN, P. AND THOMPSON, C. 1987. Randomized rounding: a technique for provably good
ROUGHGARDEN, T. 2009. Intrinsic robustness of the price of anarchy. In Proc. of the 41st Sym-
the 13th Conference on Electronic Commerce. 862–879.
SYRGKANIS, V. AND TARDOS, É. 2013. Composable and efficient mechanisms. In Proc. of the 45th
Symposium on Theory of Computing. 211–220.
NY, USA.
We study the causal effects of financial incentives on the quality of crowdwork. We focus on performance-based payments (PBPs), bonus payments awarded to workers for producing high quality work. We design and run randomized behavioral experiments on the popular crowdsourcing platform Amazon Mechanical Turk with the goal of understanding when, where, and why PBPs help, identifying properties of the payment, payment structure, and the task itself that make them most effective. We provide examples of tasks for which PBPs do improve quality. For such tasks, the effectiveness of PBPs is not too sensitive to the threshold for quality required to receive the bonus, while the magnitude of the bonus must be large enough to make the reward salient. We also present examples of tasks for which PBPs do not improve quality. Our results suggest that for PBPs to improve quality, the task must be effort-responsive: the task must allow workers to produce higher quality work by exerting more effort. We also give a simple method to determine if a task is effort-responsive a priori. Furthermore, our experiments suggest that all payments on Mechanical Turk are, to some degree, implicitly performance-based in that workers believe their work may be rejected if their performance is sufficiently poor. In the full version of this paper, we propose a new model of worker behavior that extends the standard principal-agent model from economics to include a worker’s subjective beliefs about his likelihood of being paid, and show that the predictions of this model are in line with our experimental findings. This model may be useful as a foundation for theoretical studies of incentives in crowdsourcing markets.

Categories and Subject Descriptors: H.3.5 [Online Information Services]: Web-Based Services; J.4 [Social and Behavioral Sciences]: Economics
General Terms: Economics, Experimentation
Additional Key Words and Phrases: Crowdsourcing; Performance-Based Payments; Incentives

1. INTRODUCTION
Crowdsourcing markets are platforms on which workers around the world perform tasks for pay. In a crowdsourcing market like Amazon Mechanical Turk, requesters post tasks along with the payment amount. Workers can then browse the available tasks and choose tasks to work on.
Crowdsourcing markets are used to conduct user studies [Kittur et al. 2008], run behavioral experiments [Horton et al. 2011; Mason and Suri 2012], collect data [Horton and Chilton 2010; Wah et al. 2011], test or even build business applications [Schall 2012; Alonso 2013], and more. While these markets are effective at recruiting diverse labor pools, the quality of work produced varies widely across tasks and workers. The prevalence of low quality crowdwork has inspired a growing literature on techniques to boost accuracy, for example, by using redundant assignments for labeling tasks [Sheng et al. 2008; Ipeirotis et al. 2010; Karger et al. 2011; Liu et al. 2012; Ho et al. 2013], smartly assigning tasks to workers [Ho and Vaughan 2012; Ho et al. 2013], introducing social incentives [Rogstadius et al. 2011; Shaw et al. 2011], or altering financial incentives [Mason and Watts 2009; Rogstadius et al. 2011; Buhrmester et al. 2011; Shaw et al. 2011; Harris 2011; Yin et al. 2013; 2014; Gilchrist et al. 2014]. These solutions have had mixed success, and how to improve the quality of work in general is still not well understood.

In this paper, we study the use of financial incentives to encourage high quality crowdwork on Amazon Mechanical Turk. In particular, we focus on the use of performance-based payments (PBPs), bonus payments awarded to workers for producing high quality work. Previous empirical studies of performance-based payments in crowdsourcing markets have produced mixed and somewhat contradictory recommendations. Harris [2011] and Yin et al. [2014] suggested that PBPs can improve work quality, while Shaw et al. [2011] found no improvement and Yin et al. [2013] found no difference in quality when varying bonus size.

Our results explain these disparities in prior work. Furthermore, we show how to generalize previous findings beyond the particular tasks that were studied. We design and run experiments with the goal of understanding not just whether PBPs improve work quality for a specific task or bonus size, but when, why, and where they improve work quality. We identify properties of the payment, payment structure, and the task itself that make PBPs effective.

2. DOES PBP WORK?

Workers were asked to proofread an article and correct spelling errors. For each article, we randomly inserted 20 typos from a list of common spelling errors. Workers were asked to input the line number of each typo, the misspelled word, and the correct spelling of the word.

This task has two key properties. First, we would expect that workers could produce better work by exerting more effort—the more carefully a worker reads or the more passes a worker takes over the text, the more typos he will find—and that this would open up the possibility of PBPs improving quality. (We study this conjecture in more detail in Section 4.) Second, since we injected the typos into the text, the quality of each worker’s output could be measured objectively, though this was not known to the workers.

After workers accepted the task (HIT), they were randomly assigned to different treatments and then shown treatment-specific instructions, when applicable. Our experiment had a 2 × 3 design, with 2 treatments governing the base payment and 3 treatments governing the bonus payment (if any). We discuss the bonus treatments first:
• **No Bonus**: This is the control group. It had no bonus and no mention of a bonus.
• **Bonus for All**: All workers earned a $1 bonus after submitting the HIT.
• **PBP**: Workers earned a $1 bonus if they found 75% of the typos found by the other workers.

We are also interested in whether workers have subjective assumptions on how much effort they must exert to get their work accepted. Workers may be afraid that if they do not find a sufficient number of typos their work will be rejected, resulting in no pay and a negatively affected MTurk reputation. To estimate this, we designed a treatment in which workers were explicitly guaranteed acceptance provided that they completed a very small amount of work. We had two treatments for the base payment:
• **Non-Guaranteed**: There were no extra instructions. This is the control and emulates most MTurk tasks.
• **Guaranteed**: Workers were told they would get paid if they found at least one typo.

The first typo appeared before line 3 in each article. Thus a worker would only have to do a trivial amount of work to ensure they got paid in the guaranteed base treatment.

### 2.1 Results
The HIT was completed by 1,000 unique workers, who were each assigned uniformly to one of the six treatments. The primary dependent variable was the number of true typos found. In the analysis we made six comparisons that we spell out below. We performed this analysis using an analysis of variance (ANOVA) with one-sided, planned comparisons [Seltman 2014] and report p-values that have been corrected for these multiple (six) comparisons. The results of this experiment are shown in Figure 1 and described below.

**PBPs improve quality.** To determine whether PBPs increase quality for this task, we focus on the non-guaranteed base treatments since almost all HITs on MTurk do not explicitly guarantee any kind of acceptance criteria. Workers in the PBP bonus treatment found on average 1.3 more typos than workers in the No Bonus treatment ($p = 0.042$), showing that PBPs did improve quality for this task.

**All payment schemes may be implicitly performance-based.** In the No Bonus treatment, the guaranteed base resulted in 1.5 fewer typos found on average compared with the non-guaranteed base ($p = 0.015$). Similarly, in the Bonus for All treatment, the guaranteed base resulted in 1.3 fewer typos found on average ($p = 0.024$). While there may be other explanations, this suggests that workers do have subjective beliefs on the amount of work that needs to be done for their work to be accepted, lending support to our conjecture that payments are already implicitly performance-based.

In the PBP bonus treatment, we did not see a significantly different effect between the guaranteed base and non-guaranteed base treatments. We offer two related explanations of this finding. First, the only way to grant a bonus using the MTurk
API is to first accept the work. This means that in the PBP bonus treatment, workers would likely believe that finding 75% of typos would almost certainly result in their work being accepted, already altering their subjective beliefs. Second, the treatment might have made this 75% threshold more salient to the workers. This gave a clear goal for the workers to strive for.

Simply paying more improves quality. Focusing again on the non-guaranteed base treatment, workers in the Bonus for All treatment found on average 1.3 more typos than workers in the No Bonus treatment ($p = 0.036$). Thus offering an unconditional bonus—which is essentially just paying more—increased quality. This finding is perhaps surprising since it appears to contradict the results of prior work [Mason and Watts 2009; Rogstadius et al. 2011; Buhrmester et al. 2011]. We give two potential explanations. First, since the announcement of the bonus came after workers accepted the HIT, the workers may be exhibiting reciprocity by doing higher quality work [Gilchrist et al. 2014], rewarding the requester for this pleasant surprise. We further test and refute this hypothesis in Section 3. Second, this could be explained by the implicit PBP effect described above. That is, workers might have subjective beliefs about the number of typos they must find to get paid. If we increase the bonus payment, workers might be willing to put in more effort to increase their probability of earning this higher amount.

This observation is not inconsistent with previous work. In most prior work, either easy tasks were chosen which might cause workers to perform well even for low pay [Rogstadius et al. 2011; Buhrmester et al. 2011] or additional instructions or tutorials were provided which may have primed workers’ subjective beliefs [Mason and Watts 2009].

PBPs can save money compared with high unconditional payments. In the non-guaranteed base treatment, the difference in the number of typos found in
the PBP and Bonus for All treatments is not significant. Both resulted in higher quality work than the control. However, we spent much less money on the PBP treatment. We paid each worker $1.50 in the Bonus for All treatment, while we paid each worker only $0.97 on average in the PBP treatment with non-guaranteed base and $0.96 on average in the PBP treatment with guaranteed base. Therefore, it may still be advantageous for requesters to offer PBPs even if they could achieve the same quality work with unconditional payments.

Having established that PBPs can improve quality for the proofreading task, we investigated the effect of varying two parameters of the payment scheme: the bonus threshold and the bonus amount, to better understand when PBPs help. Due to space restrictions we omit the description of this round of follow-up experiments. Our results indicate that PBPs improve quality for a wide range of possible thresholds, provided that the requester offers a bonus that is high enough to make the extra reward salient. More specifically, if the bonus offered is too small, PBPs do not improve quality (and can even reduce quality), which could explain why Shaw et al. [2011] reported little or no quality improvement using PBPs compared with fixed payments. Additionally, we found diminishing returns from increases in the payment beyond a certain point, which could explain why Yin et al. [2013] found that bonus size had little effect on quality.

3. WHY DOES PBP WORK?

There are two primary motivations for our next experiment. First, we wanted to verify that PBPs are useful in other tasks beyond finding typos. Second, we wanted to explore potential reasons why PBPs work. In particular, as pointed out in Section 2.1, simply increasing the amount of the bonus payment led to almost as much of an improvement as using PBPs in the proofreading experiment. While it could be that workers are responding rationally to the provided incentives, it could also be the case that workers are increasing their effort due to a reciprocity effect; workers are pleasantly surprised to discover the opportunity to receive a (performance-based or unconditional) bonus after accepting the HIT, and reward the requester for this kind action by working harder. Indeed, Gilchrist et al. [2014] found, in a different crowdsourcing context, that workers who accept a task and then receive an unexpected bonus do higher quality work than workers who are paid the same amount total but are told up front. This experiment is designed to test whether this “unexpected bonus effect,” is the (partial) cause of the observed increases in performance using PBPs.

3.1 Experiment Design

In this task, workers were shown twenty pairs of images. Ten of the pairs were identical images, while the other ten pairs contained minor differences. Workers were asked to specify whether each pair was identical or not, and were not told how many pairs of images were identical in advance. Again, this task has two key properties we desire. First, we speculated that workers would be more likely to spot the differences between images if they spent more time and effort looking. Second, we can objectively measure the quality of workers’ output by the number of correctly answered pairs. A similar task was used in experiments by Yin et al. [2013]. We next describe the treatments:
Fig. 2. The effect of different payment schemes on work quality in the spot the differences task. Error bars indicate the mean ± one standard error.

- **Low Base**: The base payment was $0.50. No opportunity for a bonus was given. This was our control.
- **High Base**: The base payment was $1.50. No opportunity for a bonus was given.
- **Unexpected Bonus**: The base payment was $0.50. After accepting the HIT, workers were told they would receive an additional bonus of $1.
- **PBP**: The base payment was $0.50. In addition to the base payment, workers could earn a bonus of $1 if they correctly labeled 80% of the image pairs as identical or not. Workers were informed of the bonus and rules for receiving the bonus before accepting the HIT.

Note that the payment amounts in the High Base and Unexpected Bonus treatments are the same. The difference is only how and when the payments were described.

### 3.2 Results

To avoid selection bias in workers accepting HITs with varying pay rates we randomly chose 800 workers from a pool that completed a qualification HIT and randomly assigned them to the four treatments, 200 workers each. After assigning qualifications corresponding to each treatment, we posted the HITs for each simultaneously and sent each worker a notification with a link to their treatment’s HIT. We conducted a chi-squared test to check for significant differences in the number of participants finishing the four treatments and found none ($p = 0.90$). In the analysis (see Figure 2), we make six comparisons, described below. We did this analysis using an ANOVA with one-sided, planned comparisons [Seltman 2014] and report p-values that have been corrected for these multiple comparisons.

Similar to the proofreading experiment described in Section 2, simply paying more resulted in higher quality work. The High Base treatment had a significantly
higher number of correct answers than the Low Base treatment \( (p = 0.030) \). Similarly, the Unexpected Bonus treatment had a significantly higher number of correct answers than the Low Base treatment \( (p = 0.047) \). Figure 2 shows no significant difference between the High Base and the Unexpected Bonus treatments. This suggests that there was no “unexpected bonus effect” in contrast to Gilchrist et al. [2014]. The absence of any reciprocity effect due to the unexpected bonus suggests that workers were doing better work to increase the probability (according to their prior assumptions) that their work got accepted and thus earn the higher pay.

We also observe that workers in the PBP treatment outperformed workers in all other treatments \( (p < 0.005) \). This suggests that workers are rational to some degree and are willing to exert more effort to increase their chances of receiving higher payments. Note that in this experiment workers knew \textit{before} they accepted the HIT that they could earn a bonus, in contrast to the experiment described in Section 2 in which workers were informed of the opportunity to earn a bonus only \textit{after} they accepted the HIT. We have therefore shown that PBPs can work whether or not the opportunity for a bonus is expected.

4. WHERE DOES PBP WORK?

![Graphs showing time vs. quality for different tasks](image)

Fig. 3. Time vs. quality for effort responsive tasks in panels 3(a) and 3(b), and non-effort responsive tasks in panels 3(c) and 3(d). The blue lines indicate the regression line and the shaded areas represent the 95% confidence interval around it. Results are similar when outliers are excluded from the analysis.
We have shown that PBPs incentivize higher quality crowdwork on two specific tasks, proofreading and spotting differences in images. It is natural to ask whether our results generalize, and in particular, what properties of a task open up the possibility of performance improvements with PBPs.

Camerer and Hogarth [1999] note that in the context of economics lab experiments, performance-based incentives tend to improve quality for effort-responsive tasks, tasks for which it is possible to generate higher quality work by exerting additional effort (presumably without requiring too much effort). One might ask if the same is true in a crowdsourcing setting. Since it is difficult to directly measure how much effort a worker has put into a task, we use the time a worker spent on a HIT as a proxy measure for effort, and examine the relationship between time spent and quality of work.

Figures 3(a) and 3(b) illustrate the correlation between work quality and time for the proofreading and spot-the-difference tasks respectively. Each shows the amount of time that a worker spent on the HIT versus the quality of his work. We see that, in general, workers who spent more time on our tasks generated better quality work. We observe similar trends in all treatments, but include only workers in the control groups in the plots since they are most comparable across tasks. This is evidence that the tasks on which we observed improvements from PBPs are effort-responsive.

To further explore this hypothesis and the generalizability of our results, we examined the effects of PBPs on two additional tasks, handwriting recognition and audio transcription, which is one of the most common tasks on MTurk. We did treatments with PBPs and control treatments (with no bonus) for both tasks. Again, due to space restrictions we omit the details of these follow up experiments. Figures 3(c) and 3(d) show that the quality of work produced was not significantly correlated with the time a worker spent for either task. In other words, neither task appears to be effort-responsive. Moreover, we did not find a significant difference between the accuracy of workers in the control groups versus the PBP treatments in either task via one-sided t-tests.

A Practical Recommendation.

While the results in this section are not causal, they are in line with the hypothesis that the extent to which a task is effort-responsive is an important reason for whether PBPs help improve quality for this task. This suggests an approach that requesters can use when deciding whether to employ PBPs in their own HIT. A requester could run a pilot of their HIT with a small number of workers and a fixed (not performance-based) payment and plot the time that workers spend on the task versus the quality of their work to determine whether and to what extent the task is effort-responsive. A requester may be able to incentivize higher quality using PBPs only if the task is (sufficiently) effort-responsive. In this case, the requester must determine whether the boost in quality is worth the extra cost of PBPs.
REFERENCES


HORTON, J. J. AND CHILTON, L. B. 2010. The labor economics of paid crowdsourcing. In ACM EC.


MASON, W. AND WATTS, D. J. 2009. Financial incentives and the “performance of crowds”. In HCOMP.


YIN, M., CHEN, Y., AND SUN, Y.-A. 2013. The effects of performance-contingent financial incentives in online labor markets. In AAAI.

YIN, M., CHEN, Y., AND SUN, Y.-A. 2014. Monetary interventions in crowdsourcing task switching. In HCOMP.
Why Prices Need Algorithms

TIM ROUGHGARDEN
and
INBAL TALGAM-COHEN
Stanford University

Understanding when equilibria are guaranteed to exist is a central theme in economic theory, seemingly unrelated to computation. In this note we survey our main result from [Roughgarden and Talgam-Cohen 2015], which shows that the existence of pricing equilibria is inextricably connected to the computational complexity of related optimization problems: demand oracles, revenue-maximization and welfare-maximization. We demonstrate how this relationship implies, under suitable complexity assumptions, a host of impossibility results. We also suggest a complexity-theoretic explanation for the lack of useful extensions of the Walrasian equilibrium concept: such extensions seem to require the invention of novel polynomial-time algorithms for welfare-maximization.

Categories and Subject Descriptors: F.2 [Theory of Computation]: Analysis of Algorithms and Problem Complexity; J.4 [Computer Applications]: Social and Behavioral Sciences—Economics

General Terms: Algorithms, Economics, Theory
Additional Key Words and Phrases: Market Equilibrium, Computational Game Theory, Equilibrium Computation, Market Design, Complexity

1. INTRODUCTION

Computational complexity has already had plenty to say about the computation of economic equilibria (for example, [Fischer et al. 2006; Chen et al. 2009; Chen et al. 2009; Daskalakis et al. 2009; Papadimitriou and Wilkens 2011]). The primary theme of [Roughgarden and Talgam-Cohen 2015], surveyed in this note, is that complexity can also be used to study equilibrium existence. In particular, it turns out that non-existence results can be derived from the computational intractability of related optimization problems, under widely-believed complexity assumptions like P \( \neq \) NP. We explore this theme in the classic setting of market equilibria for markets with indivisible items and quasi-linear consumers.

We begin by stating our main result for Walrasian equilibria (the leading notion of market equilibrium in our context), which we explain below in Section 2, and generalize in Section 3, before we conclude with open problems:

**Theorem 1.1.** A necessary condition for the guaranteed existence of a Walrasian equilibrium in markets with valuations from class \( V \) is that demand queries for \( V \) given item prices are as hard computationally as welfare maximization for \( V \).

(By “as hard as computationally” we refer to the existence of a polynomial-time Turing reduction from the latter to the former.)

Theorem 1.1 establishes a link between a purely economic question – existence
of equilibrium – and a purely algorithmic one. We restate it as its contrapositive, which is useful for proving nonexistence results as demonstrated below:

**Corollary 1.2.** If, under standard complexity assumptions, welfare maximization in markets with valuations from class $\mathcal{V}$ cannot be reduced to demand queries for $\mathcal{V}$ given item prices, then the existence of a Walrasian equilibrium is not guaranteed.

2. **WALRASIAN EQUILIBRIUM**

Consider a standard market model, with $n$ consumers and $m$ items. Each consumer $i$ has a valuation function $v_i$, which is a map from bundles of items to their value in $\mathbb{R}_{\geq 0}$. Consumers have quasi-linear utilities (their utility from a bundle is their value for it minus the bundle’s price). While a valuation in general requires $2^m$ numbers to represent, it is natural to assume that consumers either have a succinctly represented valuation or oracle access to it.

A Walrasian equilibrium, a fundamental economic concept dating back to the work of [Walras 1874], is a pair consisting of (1) an allocation of the indivisible items to the consumers, and (2) a price for each item, such that the following two conditions hold. These conditions are both necessary and sufficient for stability of the market:

1. Every consumer is allocated a bundle that is in his demand, i.e., maximizes his utility given the prices;
2. The revenue is maximized given the prices, i.e., all items with non-zero prices are allocated.

The first welfare theorem establishes that the allocation in every Walrasian equilibrium is welfare-maximizing. Moreover, it is welfare maximizing among all possible fractional allocations. In fact, the welfare maximization problem for a given market can be written as a linear program, called the configuration LP, whose optimal solution is an integral allocation if and only if a Walrasian equilibrium exists for this market (see, e.g., [Blumrosen and Nisan 2007]). This leads to the question: for which class of markets is a Walrasian equilibrium guaranteed to exist?

In their seminal work, [Kelso and Crawford 1982] introduced the valuation class of gross substitutes. There are many alternative definitions for gross substitutes (see [Paes Leme 2014]), but the intuition is that the items are treated by the consumer like coffee and tea than like coffee and dessert – they are not complementary in his view, and so if the price of one (coffee) rises, more of the other (tea) will be in demand. An example of gross substitutes valuations are unit-demand valuations, where the consumer has no value for more than one item (he will not drink both coffee and tea). Gross substitutes are important since Walrasian equilibria are guaranteed to exist in markets with such valuations. Moreover, the other direction partially holds as well – any class of valuations other than gross substitutes which includes unit-demand valuations does not guarantee existence of a Walrasian equilibrium [Gul and Stacchetti 1999; Milgrom 2000].

As the gross substitutes condition is quite stringent and unlikely to hold in practice, research did not stop at this, and proceeded to study classes of valuations that exclude unit-demand valuations in a relatively ad hoc fashion [Parkes and Ungar
2000; Sun and Yang 2006; Ben-Zwi et al. 2013; Candogan et al. 2014; Candogan and Pekec 2014; Sun and Yang 2014; Teytelboym 2014; Candogan et al. 2015]. Another research direction pursued by [Bikhchandani and Ostroy 2002] and others allows bundle prices rather than item prices, as well as personalization of prices, and studies the resulting generalizations of Walrasian equilibria called pricing equilibria (see Section 3).

2.1 Our Results for Walrasian Equilibrium

Let $\mathcal{V}$ be a class of valuations (e.g., unit-demand valuations). There are two algorithmic problems related to Walrasian equilibria in markets with valuations from $\mathcal{V}$:

(1) Welfare maximization: A social planner gets as input $n$ consumer valuations, and needs to output a welfare-maximizing allocation.

(2) Utility maximization, also known as answering demand queries: A consumer gets as input a vector of item prices, and needs to output a bundle of items that maximizes his utility given these prices.

Recall that our main result for Walrasian equilibrium, stated in its contrapositive, is that if welfare maximization cannot be reduced to demand queries under standard complexity assumptions, then the existence of a Walrasian equilibrium in markets with valuations from $\mathcal{V}$ is not guaranteed (Corollary 1.2). The proof is short and leans on the configuration LP and its solution via the ellipsoid method [Nisan and Segal 2006]. Our result can be applied to rule out guaranteed existence of Walrasian equilibria for candidate classes of markets.

For example, imagine a consumer who has different values for different kinds of dessert, and aggregate values for bundles of desserts up to a capacity (or budget) $b$ on the total value he can extract from dessert. Instantiate $\mathcal{V}$ with the class of budget-additive valuations, which captures such consumers. The goal when solving a demand query is then to maximize value within the capacity at minimum price, and this reduces to the knapsack problem. The goal of welfare-maximization is to fit maximum total value within the $n$ consumer capacities, and this is as hard as the bin packing problem. While knapsack and bin packing are both NP-hard, the former is only weakly so while the latter is strongly so [Garey and Johnson 1979]. Thus, assuming that values and capacities are all polynomially-bounded integers, if $P \neq NP$ then:

$$\text{Bin packing} \preceq \text{welfare maximization} \preceq \text{demand query} \preceq \text{knapsack}.$$ 

Applying Corollary 1.2 we conclude that there exists a market with budget-additive valuations and no Walrasian equilibrium.

This example demonstrates our approach to non-existence results, which stands on the mature understanding of computational complexity and is arguably more systematic and elucidating than previous approaches, with an added dependence on complexity assumptions.

3. GENERAL PRICES AND PRICING EQUILIBRIUM

We have mentioned three levels of generality in pricing so far:

(1) Item prices (a vector of $m$ prices);
(2) Bundle prices (a vector of $2^m - 1$ prices);
(3) Personalized bundle prices ($n$ vectors of $2^m - 1$ prices).

Notice that mathematically speaking, a pricing function is identical to a valuation, since it is also a function from bundles to $\mathbb{R}_{\geq 0}$. Just as there are many natural and well-studied classes of valuations, there are many classes of pricing that we can consider beyond the above three. The generalization of the Walrasian equilibrium notion to allow for personalized pricings from such a class is called a pricing equilibrium [Nisan and Segal 2006], and as in the original definition, two conditions must hold:

(1) Every consumer’s allocation maximizes his utility given his personal pricing;
(2) The allocation maximizes revenue given the pricings.

As [Nisan and Segal 2006] show, the welfare theorems continue to hold for pricing equilibria. We remark that by Condition (2), the market cannot be stabilized by setting consumer $i$’s price for all bundles not allocated to him to be $\infty$, since then the revenue would also have to be $\infty$. Condition (2) also gives rise to a third algorithmic problem in addition to welfare maximization and demand queries: In the revenue maximization problem, a seller gets as input $n$ consumer pricings, and needs to output a revenue-maximizing allocation.

One could expect that since we have allowed more general pricing, the existence of pricing equilibria will be guaranteed for many more markets than for Walrasian equilibria. This is indeed true, but in an uninteresting way. It is straightforward to show the existence of a degenerate pricing equilibrium as follows:

**Proposition 3.1.** Consider a valuation class $\mathcal{V}$ and an identical pricing class $\mathcal{P} = \mathcal{V}$. For every market with valuations from $\mathcal{V}$ there exists a pricing equilibrium with pricings from $\mathcal{P}$, where the pricing for each consumer is simply his valuation.

What makes this equilibrium uninteresting, whereas Walrasian equilibrium is a truly remarkable notion? A Walrasian equilibrium employs prices that are “simple” in three respects: they are anonymous (different consumers face the same prices); they are succinct in comparison to the class of valuations they stabilize (they are $m$-dimensional, whereas the dimension of the gross substitutes valuation class is exponential in $m$); and they make the verification of the equilibrium tractable. For example, given an alleged Walrasian equilibrium, it is trivial for the seller to verify the revenue-maximization condition by checking whether all unsold items have zero price; but for the above degenerate equilibrium, revenue maximization is as hard as welfare maximization. The main question is then, what other simple and meaningful pricing equilibria exist? Or more accurately, why are no such equilibria known to date?

### 3.1 Our Results for Pricing Equilibria

We give a brief overview of our results for pricing equilibria; details appear in [Roughgarden and Talgam-Cohen 2015].

First, the methodology we introduced for Walrasian equilibria can be utilized to show the non-existence of interesting pricing equilibria for many markets that seem like natural candidates for positive results. For example, unit-demand valuations
guarantee an equilibrium with anonymous item prices; what if we generalize to “pair-demand” valuations, would they guarantee an equilibrium with anonymous prices on pairs of items? A corollary of one of our results (a parallel of Theorem 1.1 that applies to pricing equilibria with anonymous pricing) is that they will not, at least unless NP ⊆ coNP.

Moreover, our methodology provides an explanation to the dearth of useful extensions of the Walrasian equilibrium concept, by linking the existence of such extensions to algorithmic progress on the welfare-maximization problem. In other words, the challenge of finding a novel polynomial-time algorithm for the welfare-maximization problem, beyond solving the well-known configuration LP, poses an algorithmic barrier to such results.

4. CONCLUSION

The well-studied problem of proving or disproving the guaranteed existence of pricing equilibria seems to have nothing to do with computation. We demonstrate in [Roughgarden and Talgam-Cohen 2015] that computational complexity offers numerous insights into the problem, and provides general techniques for proving impossibility results.

Many questions remain, beginning with the design of new welfare maximization methods. Does a “converse” of Theorem 1.1 hold, i.e., will non-trivial pricing equilibria emerge from such methods? Also, is our methodology applicable beyond economies with quasi-linear utilities (see [Segal 2007])? More generally, in this work we link an economic property of gross substitutes (guaranteed equilibrium existence) to their algorithmic properties, does such a link exist for other good economic properties of this valuation class?

REFERENCES


Optimal mechanisms are often prohibitively complicated, leading to serious obstacles both in theory and in bridging theory and practice. Consider the problem of a monopolist seller facing a single additive buyer with independent valuations over \( n \) heterogeneous items. Even in this simple setting, it is known that optimal (revenue-maximizing) mechanisms may require randomization [Hart and Reny 2012], use menus of infinite size [Daskalakis et al. 2015], and may be computationally intractable [Daskalakis et al. 2014].

In a letter here last year, Babaioff et al. [Babaioff et al. 2014a] described their attempt to alleviate the problem by showing that a constant fraction of the optimal revenue can be obtained by a simple mechanism. In this letter we argue in favor of a related research direction: finding the \textit{optimal simple mechanism}. We survey our recent results in this setting [Rubinstein 2016] and draw attention to the question of what is a \textit{“simple” mechanism}?

Categories and Subject Descriptors: J.4 [Social and Behavioral Science]: Economics

General Terms: Algorithms; Economics, Theory

Additional Key Words and Phrases: Optimal Mechanisms, Simple Mechanisms, Revenue, Approximation

1. INTRODUCTION

\textit{Mechanism Design 101.} Perhaps the most central problem in mechanism design is this: find the revenue-maximizing mechanism for selling \( n \) items to a single buyer with an additive valuation function. Without any constraints, the optimum mechanism is obvious: The buyer is forced to give you all her money. Of course we don’t like this solution concept (and neither does the buyer...), so we add the constraint of individual rationality (IR): the buyer will never pay more than her utility from the items allocated. The optimum IR mechanism is also trivial: Charge the buyer her valuation. The problem here is that the true valuation is the buyer’s private information and you need incentives to make her reveal it – this is the incentive compatibility (IC) constraint.

\textit{The third constraint.} So we want to find the optimal IR and IC mechanism. That’s much harder to do, but recent works on this problem provide insights into the optimal mechanisms for some important special cases. We now know that those mechanisms require randomization [Hart and Reny 2012], must use a menu of infinite size [Daskalakis et al. 2015], and may be computationally intractable [Daskalakis et al. 2014]. Using these mechanisms is problematic for a variety of reasons: Buyers and sellers may be reluctant to participate in mechanisms that are too complicated (let alone computationally intractable or infinite). Randomization may be restricted by law, or difficult to implement in a trustworthy way, and further complicated by our poor understanding of risk aversion. Such mechanisms
are just as unrealistic – and arguably as unfair to the buyer – as robbing her money or mind-reading her valuation. We need a simplicity constraint (SC): the mechanism should be simple. So, the question becomes:

Can we find the optimal IR, IC, and simple mechanism?

Alas, it is not clear how to formalize “simple”. In fact, we argue that there is no satisfactory universal definition. “Simple” can and should mean different things in different settings; for example, compare the simplicity desiderata in the following scenarios: selling produce in a grocery store (buyers are limited in time and computational capacity); spectrum auctions (buyers may be limited by legal constraints); and ad-auctions in an online marketplace (decisions are often made by automated algorithms). Nevertheless, we believe that it is an important open question to identify useful definitions of “simple”, even for special cases of interest.

Alternatively, one may hope to avoid this inconvenient question by showing that “obviously simple” mechanisms approximate the optimal revenue well. In particular Babaioff et al.’s recent letter [Babaioff et al. 2014a], describes a line of works [Hart and Nisan 2012; Li and Yao 2013; Babaioff et al. 2014b] in our setting that culminated with a celebrated $\frac{1}{6}$-approximation of the optimal revenue by the better of the following two mechanisms: (a) sell each item separately; and (b) auction all the items together as one grand bundle. As a corollary, this mechanism gives a $\frac{1}{6}$-approximation to the optimal simple mechanism for any reasonable definition of “simple”.

Our results. We argue that for the important special case of a monopolist facing a single buyer with additive, independent valuations, partition mechanisms are a good candidate for a standard for simplicity. (In a partition mechanism the seller partitions the set of items into disjoint bundles, and posts a price for each bundle; the buyer is allowed to select any number of bundles.)

For this class of mechanisms, our technical contributions include a PTAS, i.e. for any constant $\delta > 0$, we give a polynomial time algorithm that finds a partition mechanism that obtains $(1 - \delta)$-approximation to the optimal revenue among all partition mechanisms. Rather than developing novel algorithmic techniques, our main tool is exploring the structural properties of near-optimal partitions. For example, we prove that there exists a near-optimal partition mechanism with only a constant number of non-trivial bundles. We also prove that this problem is strongly NP-hard, i.e. there is no FPTAS (assuming $P \neq NP$).

2. PARTITION MECHANISMS AS SIMPLE MECHANISMS

In this section we briefly argue for the merits of partition mechanisms as a standard for simple mechanisms in our particular setting. There are also some important disadvantages - see further discussion in [Rubinstein 2016].

Expressiveness. We argue that despite their simplicity, partition mechanisms can be used to express important auctions of interest. For example, they generalize both selling items separately and bundling all the items together; thus by [Babaioff et al. 2014b] they guarantee at least a $\frac{1}{6}$-approximation to the optimal revenue achievable with any mechanism. Furthermore, this is a strong generalization: as
we show in [Rubinstein 2016], partition mechanisms can obtain twice the revenue obtained by the better of selling items separately or bundling all the items together.

**Menu complexity and false-name-proofness.** Hart and Nisan [Hart and Nisan 2013] discuss a measure of menu-size complexity: every truthful mechanism can be represented as a menu of (potentially randomized) outcomes and prices, where the buyer is allowed to choose one of those outcomes. As noted by Hart and Nisan, the mechanism which auctions each item separately has exponential menu-size complexity under this definition. To overcome this problem, they also introduce a measure that they call *additive-menu-size*, where the buyer is allowed to buy an arbitrary number of outcomes from the menu. Under this definition, partition auctions have linear additive-menu-size complexity.

A related issue is that of false-name-proofness, i.e. can a buyer gain from participating in the mechanism several times? Partition mechanisms (and additive-menu mechanisms in general) have the advantage that they are always false-name-proof.

**Locality and buyer-side computational complexity.** Partition mechanisms also have the advantage that the buyer’s decisions are “local”, i.e. the decision to buy one bundle is independent of the decision to buy other bundles. This greatly simplifies tasks such as analyzing and reasoning about such mechanisms, learning or predicting the effects of changes to the environment or the mechanism, etc. In particular, this makes the buyer’s decisions very easy.

REFERENCES


