Electronic Colloquium on Computational Complexity, Report No. 115 (2011)

Learning Hurdles for Sleeping Experts

Varun Kanade[*]
SEAS, Harvard University
vkanade@fas.harvard.edu

Thomas Steinke[†]
SEAS, Harvard University
tsteinke@fas.harvard.edu

August 7, 2011

Abstract

We study the online decision problem where the set of available actions varies over time, also called the *sleeping experts* problem. We consider the setting where the performance comparison is made with respect to the *best ordering* of actions in hindsight. In this paper, both the payoff function and the availability of actions is adversarial. Kleinberg et al. (2008) gave a computationally efficient no-regret algorithm in the setting where payoffs are stochastic. Kanade et al. (2009) gave an efficient no-regret algorithm in the setting where action availability is stochastic.

However, the question of whether there exists a *computationally efficient* no-regret algorithm in the adversarial setting was posed as an open problem by Kleinberg et al. (2008). We show that such an algorithm would imply an algorithm for PAC learning DNF, a long standing important open problem. We also show that a related problem, the *gambling* problem, posed as an open problem by Abernethy (2010) is related to agnostically learning halfspaces, albeit under restricted distributions.

[*]This work was supported in part by grants NSF-CCF-04-27129 and NSF-CCF-09-64401
[†]This work was supported in part by the Lord Rutherford Memorial Research Fellowship.

ISSN 1433-8092

1 Introduction

In online decision problems, a decision-maker must choose one of n possible *actions*, in each of the total T rounds. The decision-maker receives a payoff in the range $[0, 1]$. In the *full information* or *expert* setting, at the end of each round, the decision-maker sees the payoff corresponding to each of the possible actions. In the *bandit* setting, she only observes the reward of the action that she chose. The goal of the decision maker is to maximize her total payoff across T rounds, or as is common in the *non-stochastic* setting, to minimize her *regret* with respect to a class of strategies. The regret of the decision-maker is defined as the difference between the payoff she would have received by following the best strategy in hindsight from the class and the payoff that she actually received.

In this paper, we focus on the so-called *sleeping experts* problem. In this problem, the set of actions available to the decision-maker at round t is a subset S^t of n possible actions. The class of strategies we compare against is the set of *rankings* over the n total actions. Each ranking induces a simple strategy for the online decision problem: pick the highest-ranked available action. As a motivating example, consider the problem of choosing an advertisement to display alongside a search query. Of all the ads that match the particular keyword, only a subset might be actually available for displaying because of budget, geographical or other constraints. In this case, we would like the decision-making algorithm to compare well against the best (in hindsight) hypothetical ranking on the ads.

Our work focuses on the fully non-stochastic setting, where both the set of available actions and their payoffs are decided by an adversary[1]. In this paper, we consider the case of an *oblivious adversary*, i.e. one that does not observe the actual (random) choices made by the decision-maker. Since, our results show computational difficulties in designing efficient no-regret algorithms, they are equally applicable to the more challenging case of an adaptive adversary. An algorithm that selects an action a^t at time step t is efficient, if it makes its choice (possibly using history) in time that is polynomial in n. An algorithm is said to be a *no-regret* algorithm if its regret is $O(\text{poly}(n)T^{1-\delta})$ for some constant $\delta > 0$. An informal statement of our main result is:

Theorem 1. *If there exists a computationally efficient no-regret algorithm for the sleeping experts problem (with respect to ranking strategies), then the class of polynomial size* DNF*s is PAC-learnable under arbitrary distributions.*

In contrast to the above result, if computational efficiency is not a concern, it is easy to see that the Hedge algorithm [FS95] achieves regret $O(\sqrt{n \log(n) T})$, by treating each of the $n!$ rankings as an expert. This observation was made by Kleinberg et al. [KNMS08]. Kleinberg et al. also show that when the class of online algorithms is restricted to those that select actions by sampling over rankings and *without* observing the set of available actions S^t, there is no efficient no-regret algorithm unless $\mathsf{RP} = \mathsf{NP}$. However, this is a severe restriction and whether there exists an efficient no-regret algorithm without such restrictions was posed by Kleinberg et al. as an open question. Our result shows that such an algorithm would imply PAC-learning DNFs under arbitrary distributions, a long standing important open problem. In fact, the best known algorithm for PAC-learning DNFs takes time $2^{\tilde{O}(n^{1/3})}$.

[1] No-regret algorithms are known for the case when either the payoffs or action availabilities are stochastic; these are discussed in the related works section.

We also study a related problem, the *gambling problem*. In this problem, each round t is a game between two out of a total of n players[2]. On each round, the decision-maker must place a wager on one of the two players, and receives a payoff of 1 or 0 depending on the actual outcome. As in the sleeping experts problem, the decision-maker competes with the class of all possible rankings over the n players. Clearly, this problem is a special case of the sleeping experts problem, where the number of available actions on each round is exactly 2. Abernethy [Abe10] observed that there exists an efficient algorithm that achieves regret $O(n\sqrt{T})$, essentially by learning the outcomes between every possible pair of players. However, the Hedge algorithm (which is computationally *in*efficient because it keeps track of all $n!$ rankings) gives a regret bound of $O(\sqrt{n\log(n)T})$. Abernethy [Abe10] posed as an open question whether there exists an efficient algorithm that achieves a regret better than $O(n\sqrt{T})$. We show that this problem is related to agnostically learning halfspaces, although in a very restricted setting. However, an algorithm that achieves regret $O(n^{1-\delta}\sqrt{T})$ for the gambling problem, would result in an agnostic halfspace learning algorithm that has sample complexity $O(n^{2-2\delta})$, in this setting. We show that proper agnostic learning even in this restricted setting is hard unless $\mathsf{RP} = \mathsf{NP}$. Also, most existing techniques for improper agnostic learning of halfspaces output thresholds of polynomial functions and hence are unlikely to achieve sample complexity $O(n^{2-2\delta})$.

Our contributions: The proof of our main result follows from the fact that online agnostic learning of disjunctions reduces to the sleeping experts problem. As far as we are aware, computational hardness assumptions have not been used to show lower bounds on regret in experts/bandits problems. Lower bounds in the literature are usually based on information theoretic arguments (such as predicting coin tosses). In the sleeping experts setting, the information-theoretic lower bound of $\Omega(\sqrt{n\log(n)T})$ can indeed be achieved if computational efficiency is not a concern.

The set of available experts may be thought of as context information at each time step, and hence allows for encoding learning problems (in our case agnostic learning of disjunctions). We believe that such techniques may be applicable to other contextual experts/bandits problems. When not in the contextual experts/bandits setting, it is often possible to compete against a class of exponentially many experts (cf. [CBL06] Chap. 5).

Related Work: The most relevant related work to ours is that of Kleinberg et al. [KNMS08] and Kanade et al. [KMB09]. Kleinberg et al. showed that in the setting where payoffs are stochastic (i.e. are drawn from a fixed distribution on each round and independently for each action) and action availability is adversarial, there exists an efficient no-regret algorithm that is essentially information-theoretically optimal. Kanade et al. gave an efficient no-regret algorithm in the setting when the payoffs are set by an oblivious adversary, but the action availability is decided stochastically, i.e. a subset $S \subseteq [n]$ of available actions is drawn according to a fixed distribution at each time step. In contrast, our results in this paper show that an adversarial coupling between action availability and payoffs makes the problem much harder.

In earlier literature, different versions of the sleeping experts problems have been considered by Freund et al. [FSSW97] and Blum and Mansour [BM07]. Our results are not applicable to their settings, and in fact computationally efficient no-regret algorithms are known in those settings.

Organization. In section 2, we formally define the sleeping experts problem and the gambling problem. Section 3 provides the relevant definitions of batch and online agnostic learning. Section 4 contains the main reduction showing that the sleeping experts problem is at least as hard as

[2] This is not necessarily a tournament, i.e. it is not necessary for every player to play against every other player.

PAC learning DNF. Section 5 discusses the gambling problem in more detail and its relation to the problem of agnostically learning halfspaces.

2 Setting and Notation

Let $A = \{a_1, \ldots, a_n\}$ be the set of actions. Let T be the total number of time steps for the online decision problem. In the sleeping experts setting, at time step t, a subset $S^t \subseteq A$ of actions is available, from which the decision-maker picks an action $a^t \in S^t$. Let $p^t : S^t \to [0, 1]$ be the payoff function, and for any action a, let $p^t(a)$ denote the payoff associated with action a at time step t. At the end of round t, the entire payoff function p^t is revealed to the decision-maker. The total payoff of the decision-maker across T rounds is simply:

$$P_{\text{DM}} = \sum_{t=1}^{T} p^t(a^t)$$

When choosing an action $a^t \in S^t$, at time step T, the decision-maker may use the history to guide her choice. If the adversary cannot see any of the choices of the decision-maker we say that the adversary is *oblivious*. An *adaptive* adversary can see the past choices of the decision-maker and may then decide the payoff function and action availability. In this paper, we only consider the oblivious adversarial setting, since the hardness of designing no-regret algorithms against oblivious adversaries also applies in the case of (stronger) adaptive adversaries. Also, we only consider the full information setting, since the bandit setting is strictly harder.

The set of *strategies* that the decision-maker has to compete against is defined by the set of *rankings* over the actions. Let Σ_A denote the set of all possible $n!$ rankings over the n total actions. Given a particular ranking $\sigma \in \Sigma_A$, the strategy is to play the highest ranked available action according to σ. For subset $S \subseteq A$ of available actions, let $\sigma(S) \in A$ denote the action in S which is ranked highest according to σ. Thus, the payoff obtained by playing according to strategy σ is:

$$P_\sigma = \sum_{t=1}^{T} p^t(\sigma(S^t))$$

The quantity of interest is the *regret* of the decision-maker with respect to the class of strategies defined by rankings. The regret is defined as the difference between the payoff that would have been attained by playing according to the *best ranking strategy in hindsight* and the actual payoff received by the decision maker. Thus,

$$\text{Regret}_{\text{DM}} = \max_{\sigma \in \Sigma_A} P_\sigma - P_{\text{DM}}$$

We say that the algorithm is no-regret, if by playing according to this algorithm the decision-maker can achieve regret $O(p(n)T^{1-\delta})$, where $p(n)$ is a polynomial in n and $\delta \in (0, 1/2]$. Furthermore, we say that such an algorithm is *computationally efficient*, if at each time step t, given the set S^t of available actions (and possibly using history), it selects an action $a^t \in S^t$ in time polynomial in n.

2.1 Gambling Problem

In this section we describe a related problem, the *gambling* problem, which was posed as an open question by Abernethy [Abe10]. We use the notation described above in the context of the sleeping experts problem. Here, at each round the subset S^t of available actions is of size 2. Thus, alternatively one may think of this as predicting an outcome of a game between two players (actions). The payoff function is defined as 1 for choosing the winner and 0 for choosing the loser. We assume that the actual outcomes of the games are decided by an oblivious adversary. As in the sleeping experts problem, the goal of the decision maker is to minimize regret with respect to the class of ranking strategies.

As observed by Abernethy [Abe10], it is easy to show that there is an algorithm that achieves $O(n\sqrt{T})$ regret that is also computationally efficient. This algorithm essentially learns to predict the winner for games between players a_i and a_j for each pair of players; suppose a_i and a_j play each other for T_{ij} rounds, then it is easy to achieve $O(\sqrt{T_{ij}})$ regret on those rounds by using Hedge. Thus, the total regret is

$$\sum_{i,j} O(\sqrt{T_{ij}}) = O(n\sqrt{T}).$$

However, by keeping track of every ranking over the players (actions) as a different expert and using Hedge, it is possible to achieve $O(\sqrt{n \log(n) T})$ regret; this algorithm is computationally *in*efficient. Abernethy posed the following open question: Does there exist a computationally efficient algorithm that has regret $o(n\sqrt{T})$? In section 5, we show that any algorithm that achieves $O(n^{1-\delta}\sqrt{T})$ regret is likely to lead to interesting new techniques for agnostically learning halfspaces, albeit under very restricted distributions.

3 Agnostic Learning

In this section, we define online and batch agnostic learning. Let X be an instance space and n be a parameter than captures the representation size of X (e.g. $X = \{0,1\}^n$ or $X = \mathbb{R}^n$).

Online Agnostic Learning. The definition of online agnostic learning used here is slightly different to those previously used in the literature (cf. Ben-David et al. [BDPSS09]), but is essentially equivalent. Our definition simplifies the presentation of our results.

An online agnostic learning algorithm observes examples one at a time; at time step t it sees example $\mathbf{x}^t$, makes a prediction $\hat{y}^t \in \{0,1\}$ (possibly using history) and then observes y^t. Let $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ be a sequence of length T, where $\mathbf{x}^t \in X$ and $y^t \in \{0,1\}$. We consider the oblivious adversarial setting, where the sequence s may be fixed by an adversary but is fixed ahead of time, i.e. without observing the past predictions made by the online learning algorithm. We define error of an online agnostic learning algorithm $\mathcal{A}$ with respect to a sequence $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ as:

$$\mathrm{err}_s(\mathcal{A}) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(\hat{y}^t \neq y^t)$$

where $\mathbb{I}$ is the indicator function. For any boolean function $f : X \to \{0,1\}$ we can define error of f with respect to the sequence $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ as,

$$\mathrm{err}_s(f) = \frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(f(\mathbf{x}^t) \neq y^t).$$

For a concept class C of boolean functions over X, online agnostic learnability of C is defined as[3]:

Definition 2 (Online Agnostic Learning)**.** *We say that a concept class C over X is online agnostically learnable if there exists an online agnostic learning algorithm $\mathcal{A}$, that for all T, for all example sequences $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$, makes predictions $\hat{y}^1, \ldots, \hat{y}^T$ such that,*

$$\mathrm{err}_s(\mathcal{A}) \leq \min_{f \in C} \mathrm{err}_s(f) + O(p(n)/T^{\zeta})$$

for some polynomial $p(n)$ and $\zeta \in (0, 1/2]$. Furthermore, the running time of $\mathcal{A}$ at each time step must be polynomial in n. We say that $\mathcal{A}$ has regret bound *$O(p(n)/T^{\zeta})$.*

Batch Agnostic Learning. We also give a definition of (batch) agnostic learning (cf. Haussler [Hau92], Kearns, Schapire and Sellie [KSS94]). For a distribution D over $X \times \{0, 1\}$ and any boolean function $f : X \to \{0, 1\}$ define,

$$\mathrm{err}_D(f) = \Pr_{(x,y)\sim D}[f(x) \neq y]$$

Definition 3 ((Batch) Agnostic Learning [KSS94])**.** *We say that a concept class C is (batch) agnostically learnable, if there exists an efficient algorithm that for every $\epsilon, \delta > 0$ and for every distribution D over $X \times \{0, 1\}$, with access to a random example oracle from D, with probability at least $1 - \delta$ outputs a hypothesis h such that,*

$$\mathrm{err}_D(h) \leq \min_{f \in C} \mathrm{err}_D(f) + \epsilon$$

The running time of the algorithm is polynomial in $n, 1/\epsilon, 1/\delta$ and h is polynomially evaluatable. The sample complexity *of the algorithm is the number of times it queries the example oracle.*

It is well-known that batch learning is no harder than online learning. Theorem 4 follows more or less directly from [Lit89, CBCG04], but we provide a proof in Appendix A for completeness. Roughly speaking after an online to batch conversion, the sample complexity of the resulting batch algorithm is the number of time steps required to make the regret $O(\epsilon)$.

Theorem 4. *If a concept class C is online agnostically learnable with regret bound $O(p(n)/T^{\zeta})$ then it is (batch) agnostically learnable. Furthermore the sample complexity for (batch) agnostic learning is $O((p(n)/\epsilon)^{1/\zeta}) + O(1/\varepsilon^4 + \log^2(1/\delta) + (1/\zeta\varepsilon^2)\log(n/\varepsilon\delta))$.*

4 Sleeping Experts Problem

In this section, we show that the sleeping experts problem is at least as hard as online agnostic learning of disjunctions. Theorem 4 implies that the class of disjunctions is also (batch) agnostically learnable. It is known that agnostic learning of disjunctions implies PAC learning of DNF (cf. [KSS94, KKM09])[4], hence proving Theorem 1.

[3] The definition assumes that the online algorithm is deterministic; one may instead also allow a randomized algorithm that achieves low regret with high probability over its random choices. But, the guarantee must hold with respect to all sequences.

[4] Actually, agnostically learning conjunctions implies PAC learning DNF, but because of the duality between conjunctions and disjunctions an agnostic learning algorithm for learning disjunctions also implies an algorithm for learning conjunctions.

```
Algorithm. Online Agnostic Learning DISJ
For $t = 1, \ldots, T$,
  1. $\mathcal{L}$ receives $x^t$. Define $S^t = \{\perp\} \cup \{O_i \mid x_i^t = 1\} \cup \{Z_i \mid x_i^t = 0\}$.
  2. Give $S^t$ as the set of available actions to Alg. Let Alg choose $a^t$.
  3. If $a^t = \perp$, then set $\hat{y}^t = 0$, else set $\hat{y}^t = 1$.
  4. Observe $y^t$. Define $p^t(\perp) = 1 - y^t$ and $p^t(a) = y^t$ for all other actions $a \in S^t \setminus \{\perp\}$.
     Return $p^t$ as the payoff function to Alg.
```

Figure 1: Algorithm for online agnostically learning DISJ.

Recall that in the sleeping experts setting we consider, the action availability and payoff functions are set by an oblivious adversary. First, we define the notation used in this section. Let $X = \{0,1\}^n$ and let DISJ denote the class of disjunctions over X. Let $\mathbf{x} = x_1 \cdots x_n \in X$; for each bit x_i we define two actions O_i (corresponding to $x_i = 1$) and Z_i (corresponding to $x_i = 0$). We define an additional action $\perp$. Thus, the set of actions is $A = \{\perp, O_1, Z_1, \ldots, O_i, Z_i, \ldots, O_n, Z_n\}$.

Suppose there exists an algorithm Alg for the sleeping experts problem, that achieves regret $O(p(n)T^{1-\delta})$ for some polynomial $p(n)$ and $\delta \in (0, 1/2]$. We use Alg to construct an online learning algorithm $\mathcal{L}$ (see Fig. 1) for online agnostic learning DISJ that has average regret $O(p(n)/T^{\delta})$. The instance $\mathbf{x}^t$ is used to define the set of available actions at round t and the label y^t to define the payoffs.

Proposition 5. *Suppose there exists an efficient algorithm for the sleeping experts problem with regret $O(p(n)T^{1-\delta})$, then there exists an efficient online agnostic algorithm for learning disjunctions with average regret $O(p(n)/T^{\delta})$.*

The proof of Proposition 5 follows immediately from Lemma 6.

Lemma 6. *Let $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^{T}$ be any sequence of examples from $X \times \{0,1\}$. Let $A = \{\perp, O_1, Z_1, \ldots, O_n, Z_n\}$, Σ_A be the set of rankings over A and let S^t and p^t be as defined in Fig. 1.Then*

$$\min_{f \in \mathsf{DISJ}} \mathrm{err}_s(f) + \frac{1}{T} \max_{\sigma \in \Sigma_A} P_\sigma = 1$$

where P_σ is the payoff achieved by playing the sleeping experts problem according to ranking strategy σ.

Proof. Let σ be a ranking over the set of actions $A = \{\perp, O_1, Z_1, \ldots, O_n, Z_n\}$. For any two actions $a_1, a_2 \in A$, define $a_1 \prec_\sigma a_2$ to mean that a_1 is ranked higher by σ than a_2. For a ranking σ define a disjunction f_σ as:

$$f_\sigma = \bigvee_{i : O_i \prec_\sigma \perp} x_i \vee \bigvee_{i : Z_i \prec_\sigma \perp} \bar{x}_i$$

If for some i, both $O_i \prec_\sigma \perp$ and $Z_i \prec_\sigma \perp$, then $f_\sigma \equiv 1$. Note that several permutations may map to the same disjunction, since only which O_i and Z_i are ranked above $\perp$ is important, not their

```
Algorithm. Online Agnostic Learning $\mathsf{HHS}_2$
For $t = 1, \ldots, T$,
    1. $\mathcal{L}$ receives example $\mathbf{x}^t$. Suppose $i, j$ are such that $x_i = 1$, $x_j = -1$ and $x_k = 0$ for $k \neq i, j$. Define $S^t = \{i, j\}$.
    2. Give $S^t$ as the set of available actions (players) to $\mathsf{Alg}$.
    3. If $\mathsf{Alg}$ chooses $i$, set $\hat{y}^t = 1$; else if $\mathsf{Alg}$ chooses $j$, set $\hat{y}^t = 0$.
    4. Observe $y^t$. Define $p^t(i) = y^t$ and $p^t(j) = 1 - y^t$. Return $p^t$ as the payoff function to $\mathsf{Alg}$.
```

Figure 2: Algorithm Halfspace-Learner (under X_2)

ranking relative to each other. We show that,

$$\mathrm{err}_s(f_\sigma) + \frac{1}{T} P_\sigma = 1 \tag{1}$$

Consider some vector $\mathbf{x}^t = \{0,1\}^n$ and let $S^t \subseteq A$ be the corresponding subset of available actions (see Fig. 1). Then, note that $f_\sigma(\mathbf{x}^t) = 0$ if and only if $\sigma(S^t) = \perp$. If the true label is $y^t = 1$, f_σ suffers error $1 - f_\sigma(\mathbf{x}^t)$ and $\sigma(S^t)$ receives payoff $f_\sigma(\mathbf{x}^t)$. If the true label is $y^t = 0$, then f_σ suffers error $f_\sigma(\mathbf{x}^t)$ and $\sigma(S^t)$ receives payoff $1 - f_\sigma(\mathbf{x}^t)$. Summing over $(\mathbf{x}^t, y^t)$ in the sequence s, we get (1). But, this also completes the proof of the lemma, since for every disjunction g there exists a ranking π such that $g = f_\pi$. □

5 Gambling Problem

In this section, we consider the gambling problem (Section 2.1) posed by Abernethy [Abe10]. We show that this problem is related to that of agnostically learning halfspaces under a restricted distribution.

Let $X_2 \subseteq \{-1, 0, 1\}^n$ such that for every $\mathbf{x} \in X_2$, there is exactly one i such that $x_i = 1$, exactly one j such that $x_j = -1$ and for the remaining positions $k \neq i, j$, $x_k = 0$. Let HHS_2 be the class of homogeneous halfspaces with a positive margin over X_2. Thus,

$$\mathsf{HHS}_2 = \{\mathbf{w} \mid \mathbf{w} \in \mathbb{R}^n, \forall \mathbf{x} \in X_2, \mathbf{w} \cdot \mathbf{x} \neq 0\}$$

For $\mathbf{w} \in \mathsf{HHS}_2$, the halfspace is defined by the boolean function $\mathbb{I}(\mathbf{w} \cdot \mathbf{x} \geq 0)$. We show that a no-regret algorithm for the gambling problem immediately implies online agnostic learnability of HHS_2. Figure 2 gives an algorithm, $\mathcal{L}$, for online agnostically learning HHS_2, using an algorithm Alg for the gambling problem. Define the set of actions (players) to be $A = \{1, \ldots, n\}$, where n is the length of vectors in X_2.

Proposition 7. *Suppose that there exists an algorithm for the gambling problem with regret $O(n^{1-\delta}\sqrt{T})$, then there exists an algorithm for online agnostic learning HHS_2 with regret $O(n^{1-\delta}/\sqrt{T})$.*

The proof of Proposition 7 follows immediately from Lemma 8.

Lemma 8. *Let $s = \langle (\mathbf{x}^t, y^t) \rangle_{t=1}^T$ be the sequence of examples with $(\mathbf{x}^t, y^t) \in X_2 \times \{0,1\}$, let $A = \{1, \ldots, n\}$ and S^t and p^t be the set of available actions and payoffs as defined in Figure 2. Then,*

$$\min_{f \in \mathsf{HHS}_2} \mathrm{err}_s(f) + \frac{1}{T} \max_{\sigma \in \Sigma_A} P_\sigma = 1$$

where Σ_A is the set of rankings over the set of A and P_σ is the payoff received by playing according to σ on each round.

Proof. For any ranking σ, define $\mathbf{w}^\sigma$ in HHS_2 as follows: $\mathbf{w}_i^\sigma = n - \mathrm{pos}(\sigma, i)$, where $\mathrm{pos}(\sigma, i)$ is the position of i according ranking σ, i.e. the highest ranked element has position 1 and so on. For any $\mathbf{x} \in X_2$, let $x_i = 1$, $x_j = -1$, then $\mathbb{I}(\mathbf{w}^\sigma \cdot \mathbf{x} \geq 0) = 1$ if and only if σ ranks i higher than j, i.e. $\mathrm{pos}(\sigma, i) < \mathrm{pos}(\sigma, j)$. Thus it holds that,

$$\mathrm{err}_s(\mathbf{w}^\sigma) + \frac{1}{T} P_\sigma = 1 \tag{2}$$

Conversely, it is easy to see that any function $\mathbf{h} \in \mathsf{HHS}_2$ induces a ranking on the n actions as follows: Since $\mathbf{h}$ has positive margin over X_2 for $i \neq j$, $\mathbf{h}_i \neq \mathbf{h}_j$. Thus, let π be the ranking over $A = \{1, \ldots, n\}$, that is in descending order according to $\mathbf{h}_i$. Then for every $\mathbf{x} \in X_2$, $\mathbb{I}(\mathbf{h} \cdot \mathbf{x} \geq 0) = \mathbb{I}(\mathbf{w}^\pi \cdot \mathbf{x} \geq 0)$. This observation together with eq. (2) completes the proof of the lemma. □

5.1 Batch Agnostic Learning Halfspaces

In this section, we briefly discuss the implications of an algorithm for the gambling problem with regret $O(n^{1-\delta}\sqrt{T})$ for any constant δ. First, using Proposition 7 and Theorem 4 we immediately get the following result.

Proposition 9. *If there exists an efficient algorithm for the gambling problem with regret $O(n^{1-\delta}\sqrt{T})$ for some constant δ, then there exists an algorithm that agnostically learns halfspaces under any distribution over $X_2 \times \{0,1\}$ using $O(n^{2-2\delta})$ examples for any* constant $\epsilon > 0$.

Observe that $|X_2| = O(n^2)$ and, as such, agnostic learning any concept class over instance space X_2 is trivial. A simple algorithm is the following: Draw a sample S of size $O(n^2)$ from D, and define a hypothesis h as follows: If x is appears in the sample k times, $h(x)$ is the median of the observed labels, else $h(x)$ is randomly 1 or 0. Indeed, the trivial algorithm for the gambling problem that achieves $O(n\sqrt{T})$ mentioned in Section 2.1 and observed by Abernethy [Abe10] would reduce to this algorithm.

However, the above algorithm is essentially *memorizing* the data and not *learning*. To achieve meaningful learning, the size of the output hypothesis and also the number of examples observed should be $o(n^2)$. The VC-dimension of the class of halfspaces is only n, thus purely *statistically* it would suffice to take a sample of size $O(n)$ and perform empirical risk minimization. However, we show that proper agnostic learning of HHS_2 (i.e. the output hypothesis is also from HHS_2) is computationally intractable unless $\mathsf{RP} = \mathsf{NP}$. In order to show this we need the following theorem; this hardness of approximation result follows from standard techniques, however, for completeness, a proof is provided in Appendix B.

Theorem 10. *Let $\mathcal{G}$ be the set of directed graphs $G = (V, E)$, such that if F is the smallest feedback arc set* [5]*, then $|F| = \Theta(|E|)$. For instances of $G \in \mathcal{G}$ it is* NP*-hard to find a feedback arc set F' such*

[5] For a directed graph $G = (V, E)$, a *feedback arc set* is a set $F \subset E$ such that $(V, E \backslash F)$ has no directed cycles.

that $|F'| \leq (1 + C_f)|F|$ where F is the optimal (smallest) feedback arc set and C_f is a universal constant.

Using the above theorem we are able to prove the following result.

Proposition 11. *Let X_2 be the instance space and HHS_2 be the class of homogeneous halfspaces with non-zero margin over X_2. HHS_2 is not* proper-*agnostically learnable under arbitrary distributions over $X_2 \times \{0, 1\}$ unless* $\mathsf{RP} = \mathsf{NP}$.

Proof. Let $G = (V, E)$ be a directed graph such that $|V| = n$. For every $e = (i, j) \in E$, let $\mathbf{x}^e$ be such that $x^e_i = 1$, $x^e_j = -1$ and $x^e_k = 0$ for $k \neq i, j$; let $y^e = 1$ be the label of $\mathbf{x}^e$. Also, define $\bar{\mathbf{x}}^e = -\mathbf{x}^e$ and let $\bar{y}^e = 0$ be the label. Define D_S to be the distribution over $X_2 \times \{0, 1\}$ that is uniform over the set

$$S = \{(\mathbf{x}^e, 1) \mid e \in E\} \cup \{(\bar{\mathbf{x}}^e, 0) \mid e \in E\}$$

Let F be the smallest feedback arc-set of G. Note that the graph $G' = (V, E \setminus F)$ admits a topological ordering; let σ be this ordering. Let $\mathbf{w}^\sigma$ be a halfspace defined by σ, as in the proof of Lemma 8. Then, $\mathbf{w}^\sigma \cdot \mathbf{x}^e \geq 0$ (also $\mathbf{w}^\sigma \cdot \bar{\mathbf{x}}^e < 0$) for all $e \in E \setminus F$, and $\mathbf{w}^\sigma \cdot \mathbf{x}^e < 0$ (also $\mathbf{w}^\sigma \cdot \bar{\mathbf{x}}^e \geq 0$) for all edges in F. Thus $\mathrm{err}_{D_S}(\mathbf{w}^\sigma) = |F|/|E| = \mathrm{opt}$, which is a constant for $G \in \mathcal{G}$.

Next, we show that using any $\mathbf{h} \in \mathsf{HHS}_2$, we can construct a feedback arc set $F' \subseteq E$ such that $|F'| = \mathrm{err}_{D_S}(\mathbf{h})|E|$. Let π be the order (this is unique because of homogeneity and margin assumptions) on the vertices of V sorted according to $\mathbf{h}_i$ in descending order. Define $F' = \{(i, j) \in E \mid j$ appears before i in $\pi\}$. Then it is easy to see that $\mathrm{err}_{D_S}(\mathbf{h}) = |F'|/|E|$.

Let ϵ be a constant such that $\mathrm{opt} + \epsilon \leq (1 + C_f)\mathrm{opt}$. Thus, no polynomial time algorithm can output an $\mathbf{h} \in \mathsf{HHS}_2$ with error $\mathrm{opt} + \epsilon$ unless $\mathsf{RP} = \mathsf{NP}$.

Remark. With some more effort the non-zero margin assumption on the output hypothesis $\mathbf{h}$ may be removed. If for some subset $V' \subseteq V$, $\mathbf{h}_i$ is the same for every $i \in V'$, then let $G' = (V', E')$ be the induced subgraph. Any hypothesis on the set $\{(\mathbf{x}_e, y_e) \mid e \in E'\} \cup \{(\bar{\mathbf{x}}_e, \bar{y}_e) \mid e \in E'\}$ has error exactly $1/2$. However, since it is relatively straightforward to find 2-optimal feedback arc set for any graph, this is not a problem. □

Of course, the above proof does not rule out an improper agnostic learning algorithm that uses $O(n^{2-\delta})$ examples. However, the current known techniques for agnostically learning halfspaces use thresholds of polynomials (cf. Kalai et al. [KKMS05], Shalev-Shwartz et al. [SSSS10]) or other kernel functions. These techniques typically have sample complexity $n^{f(1/\epsilon)}$. Thus, progress on an efficient algorithm for the gambling problem would most likely lead to interesting techniques for the more general problem of agnostically learning halfspaces.

Acknowledgments

We would like to thank Adam Kalai, Tal Moran, Justin Thaler, Jonathan Ullman, Salil Vadhan, and Leslie Valiant for helpful discussions.

References

[Abe10] J. Abernethy. Can we learn to gamble efficiently? (open problem). In *COLT*, 2010.

[ALM$^+$98] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45:501–555, May 1998.

[BDPSS09] S. Ben-David, D. Pál, and S. Shalev-Shwartz. Agnostic online learning. In *COLT*, 2009.

[BM07] A. Blum and Y. Mansour. From external to internal regret. *Journal of Machine Learning Research*, 8:1307–1324, 2007.

[CBCG04] N. Cesa-Bianchi, A. Conconi, and C. Gentile. On the generalization ability of on-line learning algorithms. *IEEE Transactions on Information Theory*, 50(9):2050–2057, 2004.

[CBL06] N. Cesa-Bianchi and G. Lugosi. *Prediction, Learning, and Games*. Cambridge University Press, 2006.

[FS95] Y. Freund and R. E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *Proceedings of the second European conference on computational learning theory*, 1995.

[FSSW97] Y. Freund, R. E. Schapire, Y. Singer, and M. K. Warmuth. Using and combining predictors that specialize. In *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*, 1997.

[GJ79] M. R. Garey and D. S. Johnson. *Computers and Intractability.* W.H. Freeman and Co., New York, NY, USA, 1979.

[Hås01] J. Håstad. Some optimal inapproximability results. *J. ACM*, 48:798–859, July 2001.

[Hau92] D. Haussler. Decision theoretic generalizations of the pac model for neural net and other learning applications. *Information and Computation*, 100:78–150, 1992.

[KKM09] A. T. Kalai, V. Kanade, and Y. Mansour. Reliable agnostic learning. In *COLT*, 2009.

[KKMS05] A. T. Kalai, A. R. Klivans, Y. Mansour, and R. A. Servedio. Agnostically learning halfspaces. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science*, 2005.

[KMB09] V. Kanade, B. McMahan, and B. Bryan. Sleeping experts and bandits with stochastic action availability and adversarial rewards. In *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, pages 272–279, 2009.

[KNMS08] R. Kleinberg, A. Niculescu-Mizil, and Y. Sharma. Regret bounds for sleeping experts and bandits. *Machine learning*, pages 1–28, 2008.

[KSS94] M. J. Kearns, R. E. Schapire, and L. M. Sellie. Toward efficient agnostic learning. *Machine Learning*, 17(2-3):115–141, 1994.

[Lit89] N. Littlestone. From on-line to batch learning. In *Proceedings of the second annual workshop on computational learning theory*, 1989.

[SSSS10] S. Shalev-Shwartz, O. Shamir, and K. Sridharan. Learning kernel-based halfspaces with the zero-one loss. In *Proceedings of the 23rd Annual Conference on Learning Theory*, 2010.

A On-line to Batch Learning

We prove Theorem 4 using the following lemma.

Lemma 12. *Let $\mathcal{A}$ be an online agnostic learning algorithm for a concept class $\mathcal{C}$ over X with regret bound $O(p(n)/T^\zeta)$. We run $\mathcal{A}$ for T steps on $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$. At each step $\mathcal{A}$ can be interpreted as a hypothesis H_s^t which computes $\hat{y}^t = H_s^t(\mathbf{x}^t)$.*

Then we can choose $T = O\left(p(n)/\varepsilon\right)^{1/\zeta} + O\left(1/\varepsilon^4 + \log^2(1/\delta)\right)$ such that the following holds.

Let D be a distribution over $X \times \{0,1\}$. Take a sequence $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ of T examples from D. Let $\langle H^t\rangle_{t=1}^T$ be the hypotheses produced by $\mathcal{A}$ running on s. Then, with probability $1-\delta$ over the choice of s, there exists t^* such that $\mathrm{err}_D(H_s^{t^*}) \le \min_{f\in C} \mathrm{err}_D(f) + \varepsilon$.

Lemma 12 allows us to convert an online agnostic learning algorithm into a hypothesis, which we can use for (batch) agnostic learning.

Proof. Let $Q_s^t = \sum_{t' \le t} \mathrm{err}(H_s^{t'})$. Then, clearly, $\langle Q_s^t\rangle_{t=1}^T$ is a submartingale. Moreover, induction on T gives

$$\mathbb{E}_s[\mathrm{err}_s(\mathcal{A})] = \mathbb{E}_s\left[\frac{1}{T}\sum_{t=1}^{T} \mathbb{I}(H_s^t(\mathbf{x}^t) \ne y^t)\right] = \mathbb{E}_s\left[\frac{1}{T}\sum_{t=1}^{T} \mathrm{err}_D(H_s^t)\right] = \mathbb{E}_s\left[\frac{Q_s^T}{T}\right]. \tag{3}$$

We will now use standard bounds to show that (i) the expectation (3) is close to (or better than) the optimal error and that (ii) Q_s^T is close to its expectation with high probability. It follows that at least one hypothesis $H_s^{t^*}$ must have error close to (or better than) an optimal concept.

(i) We have

$$\mathbb{E}_s[\mathrm{err}_s(\mathcal{A})] \le \mathbb{E}_s[\min_{f\in\mathcal{C}} \mathrm{err}_s(f)] + O(p(n)/T^\zeta) \le \min_{f\in\mathcal{C}} \mathrm{err}_D(f) + O(p(n)/T^\zeta). \tag{4}$$

The first inequality follows from $\mathcal{A}$ being an online agnostic learning algorithm. The second inequality follows from the fact that $\mathbb{E}_s[\min_{f\in\mathcal{C}} \mathrm{err}_s(f)] \le \min_{f\in\mathcal{C}} \mathbb{E}_s[\mathrm{err}_s(f)]$.

(ii) Noting that $Q_s^t \le Q_s^{t+1} \le Q_s^t + 1$, Azuma's inequality gives

$$\Pr_s\left[Q_s^T \ge \mathbb{E}[Q_s^T] + T^{1-\alpha}\right] \le \exp\left(-T^{1-2\alpha}/2\right). \tag{5}$$

Combining (3), (4), and (5), we have

$$\Pr_s\left[\frac{1}{T}\sum_{t=1}^{T} \mathrm{err}_D(H_s^t) \ge \min_{f\in\mathcal{C}} \mathrm{err}_D(f) + T^{-\alpha} + O(p(n)/T^\zeta)\right] \le \exp\left(-T^{1-2\alpha}/2\right).$$

So we can choose $\alpha = 1/4$ and

$$T = \max\left\{(2/\varepsilon)^4, O(2p(n)/\varepsilon)^{1/\zeta}, (2\log(1/\delta))^2\right\}$$

to ensure that, with probability $1-\delta$,

$$\min_{t=1}^{T} \mathrm{err}_D(H_s^t) \le \frac{1}{T}\sum_{t=1}^{T} \mathrm{err}_D(H_s^t) \le \min_{f\in\mathcal{C}} \mathrm{err}_D(f) + \varepsilon.$$

□

Proof of Theorem 4. Let $\mathcal{A}$ be an online agnostic learning algorithm for a concept class $\mathcal{C}$ over X with regret bound $O(p(n)/T^{\zeta})$. Fix $\varepsilon, \delta > 0$ and a distribution D over $X \times \{0,1\}$. Choose $T = O\left(p(n)/\varepsilon\right)^{1/\zeta} + O\left(1/\varepsilon^4 + \log^2(1/\delta)\right)$ as in Lemma 12. We sample $s = \langle(\mathbf{x}^t, y^t)\rangle_{t=1}^T$ from D and run $\mathcal{A}$ on s. Now we have a sequence of hypotheses $\langle H^t\rangle_{t=1}^T$. With probability $1-\delta/2$ over the choice of s, at least one hypothesis $H_s^{t^*}$ satisfies $\mathrm{err}_D(H^{t^*}) \leq \min_{f\in\mathcal{C}} \mathrm{err}_D(f) + \varepsilon/2$. All that remains is to identify one such hypothesis.

Take T' samples $s' = \langle(\mathbf{x}^{t'}, y^{t'})\rangle_{t'=1}^{T'}$ from D. By the Chernoff bound, for any $f : X \to \{0,1\}$,

$$\Pr_{s'}\left[|\mathrm{err}_{s'}(f) - \mathrm{err}_D(f)| \geq \varepsilon/2\right] \leq 2e^{-T'\varepsilon^2/16}.$$

Let $T' = (16/\varepsilon^2)\log(4T/\delta)$. Then

$$\Pr_{s'}\left[\forall t \quad |\mathrm{err}_{s'}(H_s^t) - \mathrm{err}_D(H_s^t)| < \varepsilon/2\right] \geq 1 - \delta/2.$$

So we can estimate the accuracy of each hypothesis and identify a good one. Thus we take $T + T'$ samples and, with probability $1-\delta$, we can find a good hypothesis. □

B Hardness of Approximation for Feedback Arc Set

We use the following version of the PCP theorem as the basis of our proof.

Theorem 13 ([ALM$^+$98], [Hås01] Theorem 2.24)**.** *Let L be a language in NP and x be a string. There is a universal constant $c < 1$ such that, we can in time polynomial in $|x|$ construct a 3CNF formula $\phi_{x,L}$ such that if $x \in L$ then $\phi_{x,L}$ is satisfiable while if $x \notin L$, $\phi_{x,L}$ is at most c-satisfiable. Furthermore, each variable appears exactly 5 times.*

We show that the reduction of 'sparse' 3SAT to vertex cover produces sparse graphs with linear-sized minimal covers.

Lemma 14. *Let ϕ be a 3CNF formula with n variables and m clauses, in which each variable appears at most $d \geq 2$ times. Then there exists a simple undirected graph G_ϕ with $2n+3m$ vertices and $n+6m$ edges such that*

(i) if ϕ has an assignment of variables that satisfies $m-k$ clauses, then G_ϕ has a vertex cover of size $n+2m+k$, and

(ii) if G_ϕ has a vertex cover of size $n+2m+k$, then ϕ has an assignment that satisfies at least $m - k\lfloor d/2\rfloor$ clauses.

Moreover, the graph G_ϕ and the correspondence between assignments and vertex covers can be computed in uniform polynomial time in n and m.

Proof. This reduction comes from [GJ79] Theorem 3.3. Let ϕ be a 3CNF formula. For each variable x_i create two vertices v_{x_i} and $v_{\overline{x_i}}$ and an edge $(v_{x_i}, v_{\overline{x_i}})$ between them. For each clause $c_i = x_{i_1}^* \vee x_{i_2}^* \vee x_{i_3}^*$ (where $x_{i_k}^*$ is either x_{i_k} or $\overline{x_{i_k}}$ for $1 \leq k \leq 3$), create three vertices $v_{c_i,1}$, $v_{c_i,2}$, and $v_{c_i,3}$ and make them a triangle with edges $(v_{c_i,1}, v_{c_i,2})$, $(v_{c_i,2}, v_{c_i,3})$, and $(v_{c_i,3}, v_{c_i,1})$. Then, for $1 \leq k \leq 3$, connect $v_{c_i,k}$ to $v_{x_{i_k}^*}$. Figure 3 gives an example of this reduction.

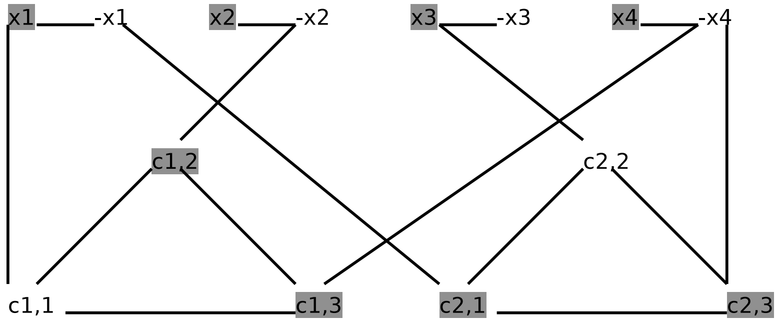

Figure 3: Vertex cover instance corresponding to 3CNF $(x_1 \vee \overline{x_2} \vee \overline{x_4}) \wedge (\overline{x_1} \vee x_3 \vee \overline{x_4})$ with an optimal vertex cover corresponding to $x_1 = x_2 = x_3 = 1$ highlighted.

(i) Let x be an assignment to the variables of ϕ that satisfies $m - k$ clauses. The corresponding vertex cover is as follows. Firstly, for each variable x_i, if x_i is true, include v_{x_i} in the vertex cover; otherwise include $v_{\overline{x_i}}$. For each unsatisfied clause c_i include $v_{c_i,1}$, $v_{c_i,2}$, and $v_{c_i,3}$. For each satisfied clause $c_i = x^*_{i_1} \vee x^*_{i_2} \vee x^*_{i_3}$, choose $1 \le k \le 3$ such that $x^*_{i_k}$ is true and include $v_{c_i,k'}$ for $k' \neq k$. Clearly this covers all edges and has size $n + 2m + k$.

(ii) Let S be a vertex cover of G_ϕ of size $n + 2m + k$. For each variable x_i either $v_{x_i} \in S$ or $v_{\overline{x_i}} \in S$, as the edge $(v_{x_i}, v_{\overline{x_i}})$ must be covered. Likewise, for each clause c_i, two of $v_{c_i,1}$, $v_{c_i,2}$, and $v_{c_i,3}$ must be in S, as the triangle $\{(v_{c_i,1}, v_{c_i,2}), (v_{c_i,2}, v_{c_i,3}), (v_{c_i,3}, v_{c_i,1})\}$ must be covered. This accounts for $n + 2m$ elements of S. So we identify k "extra" elements.

We can assume that all extra elements are of the form $v_{c_i,k}$: Suppose instead that $v_{x_i}, v_{\overline{x_i}} \in S$. Since x_i and $\overline{x_i}$ appear at most d times, we can choose one—say, x^*_i—that appears at most $\lfloor d/2 \rfloor$ times. We remove $v_{x^*_i}$ from S and insert all $v_{c_i,k}$ where $x^*_{i_k} = x_i$. Clearly all the edges that $v_{x^*_i}$ covered are now covered by the newly inserted vertices—that is, we moved the vertex cover to the other end of each edge covered by $v_{x^*_i}$. This process grows the number of extra elements by a factor of at most $\lfloor d/2 \rfloor$, so $|S| \le n + 2m + k\lfloor d/2 \rfloor$.

The assignment is computed as follows. If $v_{x_i} \in S$, then x_i is set to true. Otherwise, if $v_{\overline{x_i}} \in S$, then x_i is set to false. Each extra element of S corresponds to an unsatisfied clause, so at most $k\lfloor d/2 \rfloor$ clauses are unsatisfied. □

Now we can prove that feedback arc set is inapproximable.

Proposition 15. *Vertex cover is inapproximable, even for graphs with a number of edges linear in the number of vertices and minimal vertex covers linear in the number of vertices. Formally, there exist universal constants $c_1, c_2 > 0$, and $\varepsilon > 0$ such that the following holds. Consider graphs G with n vertices and m edges, where $m = c_1 n$. Moreover, either (i) G has a vertex cover of size $c_2 m$ or (ii) no vertex cover of size $(1 + \varepsilon)c_2 m$. If there exists a polynomial-time algorithm for deciding which of (i) or (ii) holds, then $\mathsf{NP} = \mathsf{P}$.*

Proof. This follows from Theorem 13 and Lemma 14. Firstly, Theorem 13 shows that to prove that $\mathsf{NP} = \mathsf{P}$ it suffices to distinguish in polynomial time satisfiable 3CNF formulas with $5n = 3m$, where n is the number of variables, m is the number of clauses, and each variable appears at most $d = 5$ times, from ones where at most cm clauses can be satisfied, where $c < 1$ is a universal constant. Secondly, Lemma 14 shows that such 3CNF formulas can be converted into graphs with

$n' = 2n + 3m = 7n$ vertices and $m' = n + 6m = 11n = (11/7)n'$ edges. Satisfiable 3CNF formulas become graphs with vertex covers of size $n + 2m = (13/33)m'$.

Conversely, suppose that S is a vertex cover of size $(1 + \varepsilon)(13/33)m'$. By Lemma 14 part (ii), this translates into an assignment satisfying at least $m - \varepsilon(13/33)m'\lfloor 5/2 \rfloor = (1 - (26/5)\varepsilon)m$ clauses. So, if $1 - (26/5)\varepsilon > c$, then at least cm clauses are satisfied. □

Proof of Theorem 10. We prove this by a reduction from vertex cover. Proposition 15 then gives the inapproximabililty result. Let G be a simple undirected graph with n vertices and m edges. Define a directed graph G' with $2n$ vertices and $n + 2m$ edges as follows. For each vertex v of G, create two vertices v' and v'' in G' and an edge (v', v'') from v' to v''. For each undirected edge (u, v) in G create two edges (u'', v') and (v'', u') in G'.

Note that any path in G' must alternate between edges of the form (u'', v') and edges of the form (v', v''). In particular, any cycle in G' must include the edges (u', u''), (u'', v'), and (v', v''), for some edge (u, v) in G. Consequently, if S is a vertex cover of G, then $S' = \{(v', v'') : v \in S\}$ is a feedback arc set of G'.

Let S' be a feedback arc set of G'. We may assume that S' only contains edges of the form (v', v''): If $(u'', v') \in S$, then replace it with (v', v''); note that any cycle passing through (u'', v') must also pass through (v', v''). Then $S = \{v : (v', v'') \in S'\}$ is a vertex cover of G: Otherwise suppose that (u, v) is an uncovered edge of G. Then (u'', v'), (v', v''), (v'', u'), and (u', u'') are a cycle in G' that does not intersect S'. □