6.890: Algorithmic Lower Bounds: Fun With Hardness Proofs Fall 2014

Lecture 23 Scribe Notes

Prof. Erik Demaine

1 Last time

In the last lecture we:

- Introduced the SPERNER, BROUWER, and NASH search problems
- Discussed the notion of Totality
- Defined the PPAD complexity class
- Saw that SPERNER and BROUWER are PPAD-complete

We also defined the problem of Arithmetic Circuit SAT, or `ArithmCircuitSAT` for (not so) short. We'll recall the definition since this problem will be a key player in what follows.

ArithmCircuitSAT

Input:

1. A circuit comprising of:
   - variables nodes $v_1, \ldots, v_n$
   - gate nodes $g_1, \ldots, g_m$ of types: assignment, addition, subtraction, constant, multiplication by a constant, comparison
   - Directed edges connecting variables to gates and vice versa
   - variable nodes have in-degree 1, gates have 0,1, or 2 inputs depending on type; nodes have arbitrary fan-out

Output: An assignment of values $v_1, \ldots, v_n \in [0, 1]$ satisfying gate operations (or possibly to within some ϵ).

In this lecture we will show that NASH is also PPAD-Complete. The full reduction requires a number of steps; starting from a generic PPAD problem, reducing to a PPAD-type problem in $[0, 1]^3$, then to a 3D-Sperner problem, and then to `ArithmCircuitSAT` before finally getting to NASH. Here, we will focus mainly on the reduction from `ArithmCircuitSAT` to NASH. Following all this, we will give a few other examples of complexity classes that arise from existance problems.

2 Reducing ArithmCircuitSAT to Polymatrix Nash

Rather than reducing directly to NASH, it will be useful to reduce to a more general kind of game.

2.1 Graphical Games [4]

In this setting, players are nodes in a directed graph. The payoff of a player depends only on his own strategy and those of his in-neighbors. In particular, we can specialize these to get a very useful type of game.

Polymatrix Games [5] These are graphical games with edge-wise separable utility functions. For player a v,

$$u_v(x_1, \ldots, x_n) = \sum_{(w,v)\in E} u_{w,v}(x_w, x_v) = \sum_{(w,v)\in E} x_v^T A^{(v,w)} x_w.$$

Here, $A^{(v,w)}$ are matrices, x_v is the mixed strategy of v, and x_w is the mixed strategy of w. Now, our strategy for reducing from ArithmCircuitSAT to NASH will be via PolymatrxNash, so our next goal is to reduce ArithmCircuitSAT to PolymatrxNash

2.2 ArithmCircuitSAT→PolymatrixNash

The key to this reduction is the invention of *game gadgets*, small polymatrix games that model arithmetic at their Nash equilibrium. As an example, consider the following gadget for addition.

Addition Gadget Suppose each player has two strategies, call these $\{0, 1\}$. Then, a mixed strategy is a number in $[0, 1]$, i.e. the probability of playing 1. We construct a gadget with players w, x, y, z and edges $(x, w), (y, w), (z, w), (w, z)$. Player w is paid expected:

$$\begin{array}{l} \Pr[x : 1] + \Pr[y : 1] \text{ for playing } 0 \\ \Pr[z : 1] \text{ for playing } 1 \end{array}$$

It is easy to construct this payoff matrix: $u_w(0) = x + y$ and $u_w(1) = z$.

Player z is paid for "playing the opposite" of w: $u_z(0) = .5$ and $u_z(1) = 1 - w$.

Claim: In any Nash equilibrium of a game containing the above gadget, $\Pr[z : 1] = min\{\Pr[x : 1] + \Pr[y : 1], 1\}$.

Proof. Suppose $\Pr[z : 1] < min(\Pr[x : 1] + \Pr[y : 1], 1)$. Then w will play 0 with probability 1, but then z should have played 1 with probability 1, a contradiction.

Conversely, suppose $\Pr[z : 1] > \Pr[x : 1] + \Pr[y : 1]$. Then w will play 1 with probability 1, but then z should have played 0 with probability 1, again a contradiction.

Thus, $\Pr[z:1] = \Pr[x:1] + \Pr[y:1]$.

More gadgets We need such a gadget for all the possible gates in `ArithmCircuitSAT`. If z is the output of the gadget and x, y are the inputs, we need (conflating x with $\Pr[x:1]$)

- copy: $z = x$
- addition: $z = min(1, x + y)$
- subtraction: $z = max(0, x - y)$
- set equal to constant: $z = a$
- multiply by a constant: $z = ax$
- comparison: $z = 1$ if $x > y$, $z = 0$ if $x < y$, unconstrained otherwise

Another example: Comparison gadget Players x, y, z. $u_z(0) = y$, $u_z(1) = x$. Then $x > y \Rightarrow \Pr[z:1] = 1$ and similarly for $x < y$.

PPAD Completeness of `PolymatrixNash` [1] We omit the construction of the remaining gadgets, but they are not much more complicated than the addition gadget. From here, given an arbitrary instance of `ArithmCircuitSAT`, we can create a polymatrix game by composing game gadgets corresponding to each of the gates. At any Nash equilibrium of the resulting polymatrix game, the gate conditions are satisfied, which completes the reduction.

2.3 `Polymatrix` to 2-player games [2]

Since we can construct the game gadgets we used to reduce to `PolymatrixNash` using only bipartite game gadgets (input and output players are on one side, and auxiliary nodes are on the other side), without any loss of generality we can also assume the polymatrix game is bipartite. This implies that the graph is 2-colorable, say by two colors 'red' and 'blue'. Now, we can think of this as a two-player game between the 'red lawyer' and the 'blue lawyer', where each lawyer represents all the nodes of their color.

Each lawyer's set of pure strategies is the union of the pure strategy sets of her clients; importantly, this is not the same as having a strategy set equal to the product of the strategy sets of the clients, as this would cause an exponential blowup in the problem size (and thus invalidate the reduction). One way of picturing this is that the red lawyer selects one of her clients to represent, and then selects a strategy for that client.

The payoff of $(u:i), (v:j)$ (the red lawyer plays strategy i of client u, and the blue lawyer plays strategy j of client v) is $A_{i,j}^{(u,v)}$ for u and $A_{j,i}^{(v,u)}$ for v. Informally, the payoff of the lawyers is just the payoffs of the respective clients had they played the chosen strategies. Also, note that the payoff is 0 for either lawyer if their client choice doesn't have an edge incoming from the other lawyer's client choice (and 0 for both lawyers if they choose non-neighboring clients).

Wishful thinking If (x, y) is a Nash equilibrium of the lawyer game, then the marginal distributions that x assigns to the strategies of the red nodes and the marginals that y assigns to the blue nodes comprise a Nash equilibrium.

This doesn't work because lawyers might gravitate only to the 'lucrative' clients - we need to enforce that lawyers will (at least approximately) represent each client equally. This is true not only because a lawyer might choose to *never* represent a client (and hence the marginals are undefined), but because if the red lawyer's clients are not equally represented, the optimal marginals for the blue lawyer are distorted correspondingly.

Enforcing Equal Representation Lawyers play a 'high stakes' game on the side. Without loss of generality, assume that each lawyer represents n clients (can create 'dummy' clients to equalize number of clients). Label each lawyer's clients $1, \ldots, n$ arbitrarily. Payoffs of the high stakes game: Suppose the red lawyer plays any strategy of client j and blue lawyer plays any strategy of client k, then if $j \neq k$ then both players get 0. If $j = k$ then red laywer gets $+M$ while blue lawyer gets $-M$.

Claim: In any Nash equilibrium of the high stakes game, each lawyer assigns probability (close to) $1/n$ to the set of pure strategies of each of his clients.

Now, the game we need for the reduction is simply the sum of the original lawyer game and the high stakes game. Choose $M >> 2n^2 u_{max}$, where u_{max} is the maximum absolute value of payoffs in the original game. We can show that the total probability mass is distributed almost evenly among the different nodes.

Lemma 1. *If (x, y) is an equilibrium of the lawyer game, for all u, v:*

$$x_u = \frac{1}{n}\left(1 \pm \frac{2u_{max}n^2}{M}\right)$$
$$y_v = \frac{1}{n}\left(1 \pm \frac{2u_{max}n^2}{M}\right)$$

However, within a particular node u, only the original low stakes game matters, since the different strategies of u are all identical from the perspective of the high stakes game.

Lemma 2. *The payoff difference for the red lawyer from strategies $(u : i)$ and $(u : j)$ is*

$$\sum_v \sum_\ell (A^{(u,v)}_{i,\ell} - A^{(u,v)}_{j,\ell}) y_{v:\ell}$$

Corollary If $x_{u:i} > 0$, then for all j:

$$\sum_v \sum_\ell (A^{(u,v)}_{i,\ell} - A^{(u,v)}_{j,\ell}) y_{v:\ell} \geq 0$$

Define $\hat{x}_u(i) := \frac{x_{u:i}}{x_u}$ and $\hat{y}_v(j) := \frac{y_{v:j}}{y_v}$ (these are the marginals given by lawyers to different nodes).

Observation: If we had $x_u = 1/n$ for all u and $y_v = 1/n$ for all v, then $\{\{\hat{x}_u\}_u, \{\hat{y}_v\}_v\}$ would be a Nash equilibrium. Since we have $\pm\frac{2u_{max}n^2}{M}$ deviation, we get approximate Nash equilibrium instead. Fortunately, `ArithmCircuitSAT` is still PPAD-hard with ϵ approximation, so we still get PPAD-hardness for NASH from this reduction.

Open Problem : Approximate Nash equilibrium for constant ϵ where $R, C \in [0, 1]$ can be done in time $n^{O(\log n/\epsilon^2)}$. Can this be improved?

Remark: Consider a 2-player game with payoff matrices R & C. Zero-sum games correspond to having $R+C \equiv 0$, but we can also consider games where the rank of $R+C$ is r, for some constant r. It is known that rank 1 games have a polytime algorithm for finding a NASH equilibrium (just like zero-sum games), but a result of Mehta in [6] shows that, in rank 3 games, it is already PPAD-hard to find a Nash Equilibrium.

3 Other arguments of existence and resulting complexity classes

Just as a degree parity existence problem on directed graphs defines the class PPAD, we can consider other types of existence problems and define corresponding complexity classes.

The class PPA [3] "If a graph has a node of odd degree, then it must have another."

Suppose that an exponentially large graph with vertex set $\{0, 1\}^n$ is defined by one circuit:

$C(u) = (v_1, \ldots, v_k)$, with an edge (u, v) if $v \in C(u) \wedge u \in C(v)$.

`OddDegreeNode`: Given C: if 0^n has odd degree, find another node with odd degree. Otherwise return 0^n.

PPA = {Problems reducible to `OddDegreeNode`}

`Smith`: Given Hamiltonian cycle in 3-regular graph, find another one. **Claim:** `Smith` $\in$ PPA. Note that it is not even trivial that another such cycle exists. This follows from a theorem of Smith and Thomason (see [7]).

The class PLS [8] "Every DAG has a sink."

Suppose that DAG with vertex set $\{0, 1\}^n$ is defined by two circuits: $C(u) = (v_1, \ldots, v_k)$, $F(u) = r \in \mathbb{R}$.

We add the edge (u, v) if $[v \in C(u)] \wedge [F(v) > F(u)]$; the second condition enforces that there are no cycles since moving on an edge always (strictly) increases the function F, so the graph is a DAG.

`FindSink`: Given C, F: Find x s.t. $F(x) \geq F(y)$, for all $y \in C(x)$. PLS = {Search problems reducible to `FindSink`}.

`LocalMaxCut`: Given weighted graph $G = (V, E, w)$, find a partition $V = V_1 \cup V_2$ that is locally optimal (i.e. can't move any single vertex to the other side to increase the cut size).

`LocalMaxCut` is PLS-complete for aribtrary weights.

The class PPP [3] "If a function maps n elements to $n-1$ elements, then there is a collision." (i.e. the Pigeonhole Principle)

One circuit $C(u) = v$ defines directed edges.

`Collision` Given C, either find v such that $C(v) = 0^n$ or find u, v such that $C(u) = C(v)$.

The proof that such an answer exists is that if no v exists such that $C(v) = 0^n$, then C can be regarded as a function $C : \{0,1\}^n \to \{0,1\}^n \backslash 0^n$, i.e. as a function from a set of 2^n elements to a set of $2^n - 1$ elements. Thus, by the Pigeonhole Principle, there must be a collision, i.e. u, v such that $C(u) = C(v)$.

PPP = {Problems reducible to `Collision`}.

References

[1] C. Daskalakis, P. Goldberg, C. Papadimitriou, *The Complexity of Computing a Nash Equilibrium*. SIAM Journal on Computing, 39(1), 195-359, 2006.

[2] X. Chen, X. Deng, *Settling the Complexity of Two-Player Nash Equilibrium*. Proc. of 47th FOCS, 261-272, 2006.

[3] C. Papadimitriou, *Computational Complexity*. Addison-Wesley, Reading MA, 1994.

[4] M. Kearns, M. Littman, S. Singh, *Graphical models for game theory*. In Proceedings of the Conference on Uncertainty in Artificial Intelligence, 253-260, 2001.

[5] E. B. Janovskaya, *Equilibrium points in polymatrix games*. Latvian Mathematical Collection, 1968.

[6] Ruta Mehta, *Constant Rank Bimatrix Games are PPAD-hard*. In proceedings of STOC, 2014.

[7] A. G. Thomason, *Hamiltonian cycles and uniquely edge colourable graphs*. Annals of Discrete Mathematics 3, 259268, 1978.

[8] D. S. Johnson, C. H. Papadimitriou, M. Yannakakis *How easy is local search?*. Journal of computer and system sciences, 37(1), 79-100, 1988.

[9] C. H. Papadimitriou, A. A. Schaeffer, M. Yannakakis *On the complexity of local search*. Proceedings of the twenty-second annual ACM symposium on Theory of computing, 438-445, 1990.

6.890 Algorithmic Lower Bounds: Fun with Hardness Proofs

Fall 2014