Game Theory

5. Complexity

Albert-Ludwigs-Universität Freiburg

Bernhard Nebel and Robert Mattmüller

May 9th, 2018

1 Motivation

Motivation

Motivation: We already know some algorithms for finding Nash equilibria in restricted settings from the previous chapter, and upper bounds on their complexity.

- For finite zero-sum games: polynomial-time computation.
- For general finite two player games: computation in **NP**.

Question: What about lower bounds for those cases and in general?

Approach to an answer: In this chapter, we study the computational complexity of finding Nash equilibria.

Finding Nash Equilibria as a Search Problem

Definition (The problem of computing a Nash equilibrium)

NASH

Given: A finite two-player strategic game G.

Find: A mixed-strategy Nash equilibrium (α, β) of G.

Remarks:

- No need to add restriction "…if one exists, else 'fail'", because existence is guaranteed by Nash's theorem.
- The corresponding decision problem can be trivially solved in constant time (always return "true").
  Hence, we really need to consider the search problem version instead.

In this form, NASH looks similar to other search problems, e. g.:

SAT

Given: A propositional formula φ in CNF.

Find: A truth assignment that makes φ true, if one exists, else 'fail'.

Note: This is the search version of the usual decision problem.

2 Search Problems

A search problem is given by a binary relation $R(x,y)$.

Definition (Search problem)

A search problem is a problem that can be stated in the following form, for a given binary relation $R(x,y)$ over strings:

SEARCH-R

Given: x.

Find: Some y such that $R(x,y)$ holds, if such a y exists, else ‘fail’.

Complexity Classes for Search Problems

Some complexity classes for search problems:

- **FP**: class of search problems that can be solved by a deterministic Turing machine in polynomial time.
- **FNP**: class of search problems that can be solved by a nondeterministic Turing machine in polynomial time.
- **TFNP**: class of search problems in **FNP** where the relation R is total, i. e., $\forall x \exists y . R(x, y)$.
- **PPAD**: class of search problems that can be polynomially reduced to END-OF-LINE.

  (PPAD: Polynomial Parity Argument in Directed Graphs)

To understand **PPAD**, we need to understand what the END-OF-LINE problem is.

The End-of-Line Problem

Definition (End-of-Line instance)

Consider a directed graph $\mathscr{G}$ with node set $\{0,1\}^n$ such that each node has in-degree and out-degree at most one and there are no isolated vertices. The graph $\mathscr{G}$ is specified by two polynomial-time computable functions π and σ:

- $\pi(v)$: returns the predecessor of v, or $\perp$ if v has no predecessor.
- $\sigma(v)$: returns the successor of v, or $\perp$ if v has no successor.

In $\mathscr{G}$, there is an arc from v to v' if and only if $\sigma(v) = v'$ and $\pi(v') = v$.

Definition (END-OF-LINE instance (ctd.))

We call a triple (π, σ, v) consisting of such functions π and σ and a node v in $\mathscr{G}$ with in-degree zero (a “source”) an END-OF-LINE instance.

With this, we can define the END-OF-LINE problem:

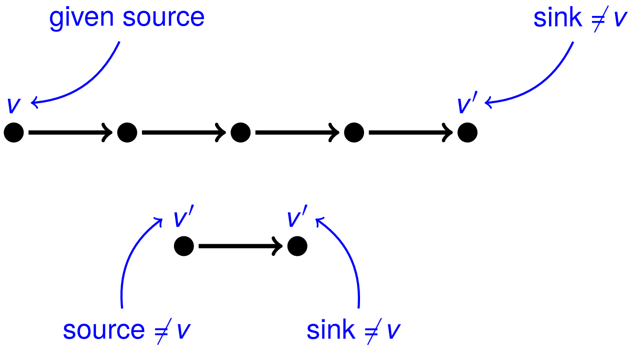
Definition (END-OF-LINE problem)

END-OF-LINE

Given: An END-OF-LINE instance (π, σ, v).

Find: Some node $v' \neq v$ such that v' has out-degree zero (a “sink”) or in-degree zero (another “source”).

Example (END-OF-LINE)



Comparison of Search Complexity Classes

Relationship of different search complexity classes:

$$FP \subseteq PPAD \subseteq TFNP \subseteq FNP$$

Compare to upper runtime bound that we already know:
Lemke-Howson algorithm has exponential time complexity in the worst case.

3 Complexity Results

PPAD-Completeness of NASH

Theorem (Daskalakis et al., 2006)

NASH *is* ***PPAD****-complete.*

The same holds for k-player instead of just two-player NASH. □

Thus, NASH is presumably "simpler" than the SAT search problem, but presumably "harder" than any polynomial search problem.

FNP-Completeness of 2ND-NASH

Another search problem related to Nash equilibria is the problem of finding a second Nash equilibrium (given a first one has already been found). As it turns out, this is at least as hard as finding a first Nash equilibrium.

Definition (2ND-NASH problem)

2ND-NASH

Given: A finite two-player game G and a mixed-strategy Nash equilibrium of G.

Find: A second different mixed-strategy Nash equilibrium of G, if one exists, else 'fail'.

Theorem (Conitzer and Sandholm, 2003)

2ND-NASH *is* ***FNP****-complete.* □

Some Further Hardness Results

Theorem (Conitzer and Sandholm, 2003)

*For each of the following properties P^ℓ, $\ell = 1,2,3,4$, given a finite two-player game G, it is **NP**-hard to decide whether there exists a mixed-strategy Nash equilibrium (α,β) in G that has property P^ℓ.*

P^1 : *player 1 (or 2) receives a payoff $\geq k$ for some given k.*
("Guaranteed payoff problem")

P^2 : $U_1(\alpha,\beta) + U_2(\alpha,\beta) \geq k$ *for some given k.*
("Guaranteed social welfare problem")

P^3 : *player 1 (or 2) plays some given action a with prob. > 0.*

P^4 : *(α,β) is Pareto-optimal, i. e., there is no strategy profile (α',β') such that*

- $U_i(\alpha',\beta') \geq U_i(\alpha,\beta)$ *for both $i \in \{1,2\}$, and*
- $U_i(\alpha',\beta') > U_i(\alpha,\beta)$ *for at least one $i \in \{1,2\}$.* □

4 Summary

Summary

- **PPAD** is the complexity class for which the END-OF-LINE problem is complete.
- Finding a mixed-strategy Nash equilibrium in a finite two-player strategic game is **PPAD**-complete.
- **FNP** is the search-problem equivalent of the class **NP**.
- Finding a second mixed-strategy Nash equilibrium in a finite two-player strategic game is **FNP**-complete.
- Several decision problems related to Nash equilibria are **NP**-complete:
  - guaranteed payoff
  - guaranteed social welfare
  - inclusion in support
  - Pareto-optimality of Nash equilibria