1 Complexity, Games, Polytopes and Gale Strings

1.1 Some Complexity Classes

reference - take Papadimitriou (book) for general, then article Megiddo and Papadimitriou (1991) for (T)FNP and Papadimitriou 1994 for PPAD

A *computational problem* is given by the combination of an *input* and a related *output*. A specific input gives an *instance* of the problem.

Computational problems can be classified according to the form of their output. A *decision problems* outputs either "YES" or "NO". An instance x *function problem*, on the other hand, returns a more generic output y that satisfies a given relation $R(x, y)$.

Search problems are function problems that return either an output y satisfying a given relation $R(x, y)$ or "NO", if it's not possible to find any such y. If y is guaranteed to exist, the problem is called a *total function problem*. *Counting problems*, finally, return the *number* of y's that satisfy $R(x, y)$; given a problem R we denote the associated counting problem $\#R$.

An example of decision problem is: "(input) given a graph, (question) is it possible to find an *Euler tour* for the graph?" A search problem is: "(input) given a graph, (output) return one Euler tour of the graph, or "NO" if no such tour exists." A total function problem is: "(input) given an Euler graph, (output) return one of its Euler tours." A counting problem is "(input) given a graph, (output) return the number of its Euler tours."

Computational problems are also classified according to their *computational complexity*, given by the *reducibility* from each other.

Turing machines: here

Let P_1 be a computational problem. For an instance x of P_1, let $|x|$ be the the number of bits needed to encode x. P_1 *reduces to the problem P_2 in polynomial time*, denoted $P_1 \leq_P P_2$, if there exists a *polynomial-time reduction*, that is, a function $f : \{0, 1\}^* \to \{0, 1\}^*$ and a *Turing machine* $\mathcal{M}$

such that for all $x \in \{0,1\}^*$

1. $x \in P_1 \iff f(x) \in P_2$;
2. $\mathcal{M}$ computes $f(x)$;
3. $\mathcal{M}$ stops after $p(|x|)$ steps, where p is a polynomial.

The complexity class **P** contains all the *polynomially decidable problems*, that is, all problems P such that there exists a Turing machine $\mathcal{M}$ that outputs either "YES" or "NO" for all inputs $x \in \{0,1\}^*$ of P after $p(|x|)$ steps, where p is a polynomial. Intuitively, a decision problem is in **P** if the answer to its question can be found in a number of steps that is polynomial in the input of the problem. The class **FP** of all the function problems that can be solved in polynomial time is analogously defined.

The class **NP**, *non-deterministic polynomial-time problems*, is the class of problems P such that there exists a Turing machine $\mathcal{M}$ and polynomials p_1, p_2 such that

1. for all $x \in P$ there exists a *certificate* $y \in \{0,1\}^*$ which satisfies $|y| \leq p_1(|x|)$;
2. $\mathcal{M}$ accepts the combined input xy, stopping after at most $p_2(|x| + |y|)$ steps;
3. for all $x \notin P$ there does not exist $y \in \{0,1\}^*$ such that $\mathcal{M}$ accepts the combined input xy.

Intuitively: a decision problem is in **NP** if it takes polynomial time to verify whether the "certificate solution" y is, indeed, a correct answer to the question posed by the problem.

P

The class **FNP**, *function non-deterministic polynomial*, is defined as the class of binary relations $R(x, y)$ such that there is a polynomial-time algorithm that decides whether $R(x, y)$ holds for given x, y satisfying $|y| \leq p(|x|)$, where p is a polynomial. If a y as above is guaranteed to exist, the problem belongs to the class **TFNP**, *total function non-deterministic polynomial*. That is, **FNP** and **TFNP** are analogous to **NP**, but they allow for problems of (respectively) function and total function form.

More on TFNP: no complete pbls unless NP=co-NP (def co-NP)
$\Rightarrow$
definition of PPA(D)

1.2 Normal Form Games and Nash Equilibria

until here

1.3 Best Response Polytopes

file: polytopes-subsection

1.4 Cyclic Polytopes and Gale Strings

1.5 The Problem Another Gale

file: gale-def-subsection
merge in one section "Gale strings" or "CP and GS"?

References

[1] M. M. Casetti, J. Merschen, B. von Stengel (2010). Finding Gale Strings. *Electronic Notes in Discrete Mathematics*

issue, pp. n–m.

[2] X. Chen, X. Deng (2006). Settling the complexity of two-player Nash equilibrium. *Proc. 47th FOCS*, pp. 261–272.

[3] C. Daskalakis, P. W. Goldberg, C. H. Papadimitriou (2006). The complexity of computing a Nash equilibrium. *Proc. Ann. 38th STOC*, pp. 71–78.

change ref to econometrica(?)

[4] J. Edmonds (1965). Paths, trees, and flowers. *Canad. J. Math.* 17, pp. 449–467.

[5] D. Gale (1963), Neighborly and cyclic polytopes. *Convexity, Proc. Symposia in Pure Math.*, Vol. 7, ed. V. Klee, American Math. Soc., Providence, Rhode Island, pp. 225–232.

check if right typography

[6] J. Merschen (2012).

thesis

[7] C. E. Lemke, J. T. Howson, Jr. (1964). Equilibrium points of bimatrix games. *J. Soc. Indust. Appl. Mathematics* 12, pp. 413–423.

[8] C. H. Papadimitriou (1994). On the complexity of the parity argument and other inefficient proofs of existence. *J. Comput. System Sci.* 48, pp. 498–532.

[9] R. Savani, B. von Stengel (2006). Hard-to-solve bimatrix games. *Econometrica* 74, pp. 397–429.

[10] L. Végh, B. von Stengel

ref