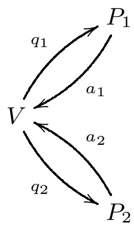
Probabilistic Games

1 Two-prover Games

Assume we have the following game. A verifier V interacts with two provers P_1 and P_2. The verifier is probabilistic and acts honestly according to its (known) strategy. The goal of the provers is to convince the verifier of a certain fact. The provers may collaborate to set up a strategy to maximize their chances of success. However, they are not allowed to exchange information once they received their respective question from the verifier.



More formally, the verifier V is a Turing machine with a private random tape, and the provers P_1 and P_2 are deterministic Turing machines. V writes a question each on the input tapes of P_1 and P_2. Then the provers are invoked, and after invocation they write their answers to V's input tape, and V is started again. V reads the answers and decides to either accept or reject.

In the above description, only V is allowed to use random data. One might ask if P_1 and P_2 could gain from being probabilistic. Assume there is a strategy which allows probabilistic P_1 and P_2 to convince V with probability p. Then P_1 can, for each of its possible answers, compute the probability V will accept over the internal randomness of V and P_2 and choose the answer that maximizes this probability. P_2 can use the same reasoning to construct a deterministic strategy. Clearly the deterministic strategy will succeed with probability at least p.

Now consider the following game.

1. V chooses two random and independent bits q_1 and q_2. V sends q_1 to P_1 and q_2 to P_2.
2. P_1 and P_2 respond by sending one bit a_i each to V.
3. V accepts if $(q_i = 1) \rightarrow (a_i = 1)$ for $i = 1, 2$ and $a_0 \neq a_1$.

If $q_1 = q_2 = 1$, the provers can never succeed. It is clear that the strategy to always answer 1 always fails, and that the strategy to set $a_i = q_i$ succeeds with probability $p = 1/2$. Since the provers must set $a_i = 1$ when $q_i = 1$, we only need to consider different answers to $q_i = 0$. This gives a total of four possible joint strategies, and it can be easily verified that there is none succeed with probability higher than $1/2$.

1.1 Parallel composition

Now consider two such games played in parallel. If P_1 and P_2 use the strategy above that succeeds with $p = 1/2$, their chance of success in both games is $p^2 = 1/4$. An interesting question is whether there exists a strategy that beats $1/4$, and the somewhat surprising answer is that there is such a strategy. Assume both players acts as follows. We denote the bits that P_i receives q_i^1 and q_i^2, and the answer bits a_i^1 and a_i^2.

- If $(q_i^1, q_i^2) = (0, 0)$ answer with $(a_i^1, a_i^2) = (0, 0)$.
- In all other cases, answer with $(a_i^1, a_i^2) = (1, 1)$.

Using this strategy, P_1 and P_2 succeed $(q_1^1, q_2^1) = (0, 0)$ and $(q_2^1, q_2^2) \neq (0, 0)$. The probability for this to happen is $1/4\,3/4 = 3/16$. Since the strategy is symmetric, they succeed also if $(q_1^1, q_2^1) \neq (0, 0)$ and $(q_2^1, q_2^2) = (0, 0)$, giving a total success probability of $2\,3/16 = 3/8 > 1/4$. Hence this strategy performs better than p^2.

The above example shows that parallel repetition does not in general lower the acceptance probability for the verifier as much as one might suspect. However, Raz [2] has proven the following theorem, which says that the acceptance probability does decrease exponentially, but possibly with a smaller factor.

Theorem 1. *Let (V, P_1, P_2) be a two-prover game with acceptance probability $c < 1$ and answer size d. Then there exists a constant $c_d < 1$ such that the acceptance probability of k parallel executions of the game is at most c_d^k.*

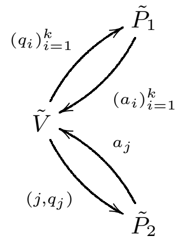
1.2 Completeness and Soundness

In the remainder of the text, the provers will try to convince the verifier of a certain fact, namely that a boolean formula is satisfiable. Two important properties of such a game is *completeness* – the probability of the provers to convince the verifier of a true statement – and *soundness* – the probability that the verifier will accept an attempted proof of a false statement. A proof system that has completeness 1 is said to have *perfect completeness* and a proof system with soundness 0 is said to have *perfect soundness*.

1.3 Extending to k-prover games

Now consider a k-prover game $(V, P_1, P_2, \ldots, P_k)$, where the verifier produces k questions $(q_i)_{i=1}^k$ and receives k answers $(a_i)_{i=1}^k$. We show how to convert this game into a two-prover game $(\tilde{V}, \tilde{P}_1, \tilde{P}_2)$. The verifier $\tilde{V}$ sends all k questions to $\tilde{P}_1$ which (by simulating $P_1, \ldots, P_k$) produces k answers. V's question to $\tilde{P}_2$ is

(q_j, j) for a random $j \in \{1, 2, \ldots, k\}$. This question, which can be thought of as a control question, is answered by simulating P_j and forwarding the answer. V accepts if, in addition to the requirements of the original games, $\tilde{P}_1$'s answer on question j is equal to the answer from $\tilde{P}_2$.



The idea behind this construction is that $\tilde{P}_1$ cannot do better than simulating $P_1, P_2, \ldots, P_k$ honestly, because otherwise the answer from $\tilde{P}_2$ is inconsistent with $\tilde{P}_1$'s answers. It can be shown that this construction gives a protocol with soundness bounded away from 1 if the original protocol has soundness bounded away from 1.

2 Probabilistically Checkable Proofs (PCPs)

A probabilistically checkable proof, PCP, is a proof for a certain statement, such as satisfiability of a formula, that can be verified only by reading a few bits rather than the complete proof. The goal is to construct a proof such that only a small number of bits need to read in order to achieve good soundness.

Suppose we have a two-prover game for a certain claim where the number of possible question is low and the size of the answers is small. Then it is possible to write down how each prover would answer to each possible question. For certain proof systems, such a proof can be verified by reading the answers to only a few questions, rather than reading the complete proof. Hence this is a good starting point for creating an efficient PCP. The PCP theorem formalizes these ideas.

Let φ be a formula, and we want to determine whether or not φ is satisfiable. We will use a proof systems (P, V). On input φ the prover P outputs a proof for the satisfiability of φ. However, rather than reading the complete proof, the verifier V only reads a constant number of bits, and decides whether to accept or reject only on that information. Completeness and soundness are defined in a similar way to the interactive case.

Definition 1. *Let (P, V) be a proof system. Let L be the language of satisfiable formulas. The proof system has completeness c if for any formula φ it holds that $\varphi \in L \Rightarrow \exists \pi \;\; \Pr[V(\pi) = 1] \geq c$.*

Definition 2. *Let (P, V) be a proof system. Let L be the language of satisfiable formulas. The proof system has soundness s if for any formula φ it holds that $\varphi \notin L \Rightarrow \forall\pi \ \ \Pr[V(\pi) = 1] \leq s$.*

From now on we will consider only proof systems where the execution time of V and the size of the proof π are polynomially bounded in $|\varphi|$. We also restrict ourselves to non-adaptive verifiers, i.e., to verifiers which decides which bits to read based only on the random tape. However, we don't impose any restrictions on P. The following theorems state the existence of such proof systems.

Theorem 2 (The PCP Theorem, variant 1). *There exists a proof system (P, V) with completeness 1 and soundness $1/2$ such that V and the output of P are polynomially bounded. Furthermore V reads a constant number of bits from the proof and uses a logarithmic number of bits from its random tape.*

Theorem 3 (The PCP Theorem, variant 2). *There exists a proof system (P, V) with completeness 1 and soundness $s < 1$ such that V and the output of P are polynomially bounded. Furthermore V reads a 3 bits from the proof and uses a logarithmic number of bits from its random tape.*

As a corollary we get the following

Corollary 1. Max-3SAT *is NP-hard to approximate within a factor d for some $d > 1$.*

Proof. Consider the problem of deciding whether a formula φ is satisfiable or not. According to the PCP theorem there exists a proof system (P, V) for this task such that the probability that $V(\pi) = 1$ is at most $1 - s$ for an unsatisfiable φ. We will now proceed by constructing the proof for φ.

Let r be the random bits used by V. From the theorem we know that r is logarithmic in $|\varphi|$. Since V is non-adaptive, which 3 bits V reads from the proof is determined by r and φ. Denote these three bits $b_{i_1}, b_{i_2}, b_{i_3}$. Now our approach is to determine whether φ is satisfiable or not by constructing a proof that V will accept.

V will reject for certain values of $b_{i_1}, b_{i_2}, b_{i_3}$. For each tuple (c_1, c_2, c_3), $c_i \in \{0, 1\}$, such that V rejects if $(b_{i_1}, b_{i_2}, b_{i_3}) = (c_1, c_2, c_3)$, add the clause $(l_{i_1} \vee l_{i_2} \vee l_{i_3})$ to ψ, where

$$l_i = \begin{cases} b_i & \text{if } c_i = 0 \\ \bar{b}_i & \text{if } c_i = 1 \end{cases} .$$

Now one of the clauses will be 0 for a proof which makes V reject on random bits r, but all clauses are 1 for a proof that V accepts. For each value of r this gives at the most 7 clauses. Should V reject for all 8 values of $b_{i_1}, b_{i_2}, b_{i_3}$, perfect completeness gives that ϕ is not satisfiable.

We construct the set of at most 7 clauses for every possible value of r, which gives a formula that consists of $\leq 7 \cdot 2^{|r|}$ clauses. If φ is satisfiable, then all clauses of ψ can be satisfied. If φ is not satisfiable, then V rejects with probability at least s, e.g., there are at least $s2^{|r|}$ random strings that make V reject. Since at least one clause will remain unsatisfied if V rejects, no more than a fraction $1 - \frac{s2^{|r|}}{7 \cdot 2^{|r|}} = 1 - s/7$ of all clauses of ψ can be satisfied in this case.

Since ψ can be constructed in polynomial time, an algorithm that can distinguish between the case when all clauses are satisfiable and just a fraction $1 - s/7$ can solve SAT. Since an algorithm that approximates MAX-3SAT within $(1 - s/7)^{-1}$ is able to do that, this approximation problem is **NP**-hard.

Further results are given in [1].

References

1. J. Håstad. Some optimal inapproximability results. *Journal of the ACM*, 48(4):798–859, 2001. Available at http://www.nada.kth.se/ johanh/optimalinap.ps.
2. R. Raz. A parallel repetition theorem. *SIAM Journal of Computing*, 27(3):763 – 803, 1998.