Search versus Decision for S_2^P

Lance Fortnow
Illinois Institute of Technology

Abstract

We compare the complexity of the search and decision problems for the complexity class S_2^P. While Cai (2007) showed that the decision problem is contained in ZPP^NP, we show that the search problem is equivalent to $\mathsf{TFNP}^\mathsf{NP}$, the class of total search problems verifiable in polynomial time with an NP oracle. This highlights a significant contrast: if search reduces to decision for S_2^P, then $\Sigma_2^\mathsf{P} \cap \Pi_2^\mathsf{P} \subseteq \mathsf{ZPP}^\mathsf{NP}$.

1 Introduction

The complexity class S_2^P, introduced by Canetti [Can96] and Russell and Sundaram [RS98], is defined by a game between two competing provers. A language L is in S_2^P if there exists a polynomial-time predicate P such that for $x \in L$, there is a strategy for the first prover that wins against any move of the second prover, and for $x \notin L$, there is a strategy for the second prover that wins against any move of the first. Alternatively we can think of an exponential-sized matrix generated by P where x is in L if there is a row of all ones and x is not in L if there is a column of all zeros.

Cai [Cai07] showed that the decision problem for S_2^P is in ZPP^NP, suggesting it is relatively easy, at least compared to $\Sigma_2^\mathsf{P} \cap \Pi_2^\mathsf{P}$. However Cai's algorithm does not necessarily find a witness, the row of all ones or the column of all zeros. Cai's algorithm might eliminate all possible columns guaranteeing that x is in L but leave many possible rows. In this paper, we explore the complexity of this search problem and contrast it with the complexity of the decision problem.

The search problem appears to be much harder. We relate this search problem to TFNP, the class of total function problems introduced by Megiddo and Papadimitriou [MP91]. Specifically, we consider $\mathsf{TFNP}^\mathsf{NP}$, the class of total search problems where the validity of a solution can be checked in polynomial time given access to an NP oracle.

Our main result is that finding an S_2^P witness is complete for $\mathsf{TFNP}^\mathsf{NP}$. While S_2^P is unlikely to contain $\Sigma_2^\mathsf{P} \cap \Pi_2^\mathsf{P}$, the latter class reduces to the search problem for S_2^P.

2 Preliminaries

2.1 Definitions

Definition 1 (S_2^P)**.** *A language L is in S_2^P if there exists a polynomial-time predicate P and a polynomial q such that:*

- *If $x \in L$, then $\exists y \in \{0,1\}^{q(|x|)}\, \forall z \in \{0,1\}^{q(|x|)}\, P(x,y,z) = 1$.*
- *If $x \notin L$, then $\exists z \in \{0,1\}^{q(|x|)}\, \forall y \in \{0,1\}^{q(|x|)}\, P(x,y,z) = 0$.*

Definition 2 (S_2^P-Search)**.** *Given an input x and the predicate P defining an S_2^P language, output y such that $\forall z\, P(x,y,z) = 1$ or output z such that $\forall y\, P(x,y,z) = 0$.*

Definition 3 ($\mathsf{TFNP}^\mathsf{NP}$)**.** *A binary relation $R(x,y)$ is in $\mathsf{TFNP}^\mathsf{NP}$ if:*

- *R is decidable in polynomial time with an NP oracle (i.e., $R \in \mathsf{P}^\mathsf{NP}$).*
- *For every x, there exists a y such that $R(x,y)$ holds.*
- *The length of y is bounded by a polynomial in the length of x.*

The search problem is: Given x, find y such that $R(x,y)$.

3 The Equivalence

Theorem 4. *S_2^P-Search is equivalent to $\mathsf{TFNP}^\mathsf{NP}$.*

3.1 S_2^P-Search in $\mathsf{TFNP}^\mathsf{NP}$

To show that S_2^P-Search is in $\mathsf{TFNP}^\mathsf{NP}$, we need to show that the problem is total and that verifying a solution can be done in P^NP. By the definition of S_2^P, for every x, either there exists a y such that $\forall z P(x,y,z)$ holds or there exists a z such that $\forall y \neg P(x,y,z)$ holds. Thus, a witness always exists. To verify a claimed witness y (asserting $x \in L$), we need to check if $\forall z P(x,y,z)$. This is a coNP question, which can be answered by an NP oracle. Similarly, verifying a witness z (asserting $x \notin L$) requires checking $\forall y \neg P(x,y,z)$, which is also a coNP question.

Therefore, S_2^P-Search is in $\mathsf{TFNP}^\mathsf{NP}$.

3.2 $\mathsf{TFNP}^\mathsf{NP}$ reduces to S_2^P-Search

We now show that any problem in $\mathsf{TFNP}^\mathsf{NP}$ can be reduced to finding an S_2^P witness. Let R be a total relation in P^NP. We construct an S_2^P predicate Q such that finding a strategy for Q yields a y satisfying $R(x,y)$.

Since $R \in \mathsf{P}^\mathsf{NP}$, there is a polynomial-time machine M with oracle access to SAT that decides $R(x,y)$. Let x be an input. We define the predicate $Q(x,Y,Z)$ as follows.

The first player's move Y consists of:

- A candidate solution y.
- A transcript of oracle queries and answers $\vec{a} = (a_1, \ldots, a_k)$ for the computation of $M(x,y)$.
- For every query i where $a_i = 1$ (asserting the query formula ϕ_i is satisfiable), a witness w_i satisfying ϕ_i.

The second player's move Z consists of:

- An index $j \in \{1, \ldots, k\}$.
- A witness w'_j.

The predicate $Q(x,Y,Z)$ evaluates to 1 if all of the following hold:

1. The simulation of $M(x, y)$ using oracle answers $\vec{a}$ results in acceptance.

2. For all i where $a_i = 1$, w_i is a valid satisfying assignment for ϕ_i.

3. If $a_j = 0$ (asserting ϕ_j is unsatisfiable), then w'_j does *not* satisfy ϕ_j.

If a valid y exists (which is true by totality), Player 1 can choose y, the correct oracle answers, and valid witnesses for all satisfiable queries. In this case, conditions 1 and 2 are met. For condition 3, since the oracle answers are correct, if $a_j = 0$, then ϕ_j is truly unsatisfiable, so no w'_j can satisfy it. Thus, Player 1 has a winning strategy.

Conversely, if Player 1 has a winning strategy Y, condition 1 implies M accepts y given $\vec{a}$. Condition 2 ensures all YES answers are correct. Condition 3 ensures that Player 2 cannot produce a witness for any query answered NO, implying all NO answers are correct (since Player 2 could play any witness). Thus, y must be a valid solution to $R(x, y)$.

4 Implications and Discussion

This result bridges the gap between the symmetric hierarchy and total search problems.

Corollary 5. *If search reduces to decision for* $\mathsf{S}_2^{\mathsf{P}}$*, then* $\Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}} \subseteq \mathsf{ZPP}^{\mathsf{NP}}$*.*

Proof. Cai [Cai07] showed that $\mathsf{S}_2^{\mathsf{P}} \subseteq \mathsf{ZPP}^{\mathsf{NP}}$. If search reduces to decision for $\mathsf{S}_2^{\mathsf{P}}$, then finding an $\mathsf{S}_2^{\mathsf{P}}$ witness can be done in $\mathsf{ZPP}^{\mathsf{NP}}$.

Consider any language $L \in \Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}}$. Since $L \in \Sigma_2^{\mathsf{P}}$, there is a polynomial-time predicate A such that $x \in L \iff \exists y \forall z A(x, y, z)$. Since $L \in \Pi_2^{\mathsf{P}}$, there is a polynomial-time predicate B such that $x \notin L \iff \exists y' \forall z' B(x, y', z')$. The search problem for L is to find either a y such that $\forall z A(x, y, z)$ or a y' such that $\forall z' B(x, y', z')$. Since one of these must exist, the problem is total. Furthermore, verifying a solution requires checking a universal quantifier, which can be done with an NP oracle. Thus, the search problem for L is in $\mathsf{TFNP}^{\mathsf{NP}}$.

Since $\mathsf{TFNP}^{\mathsf{NP}}$ is equivalent to $\mathsf{S}_2^{\mathsf{P}}$-Search, this implies that finding witnesses for languages in $\Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}}$ can be done in $\mathsf{ZPP}^{\mathsf{NP}}$. Consequently, $\Sigma_2^{\mathsf{P}} \cap \Pi_2^{\mathsf{P}} \subseteq \mathsf{ZPP}^{\mathsf{NP}}$. □

This highlights a significant difference between search and decision for $\mathsf{S}_2^{\mathsf{P}}$. While the decision version is relatively "easy" (in $\mathsf{ZPP}^{\mathsf{NP}}$), the search version captures the full complexity of $\mathsf{TFNP}^{\mathsf{NP}}$.

5 Conclusion

We have shown that $\mathsf{S}_2^{\mathsf{P}}$ is the "correct" complexity class for characterizing $\mathsf{TFNP}^{\mathsf{NP}}$. This equivalence provides new insights into both the structure of the symmetric hierarchy and the nature of total search problems with oracle verification.

Acknowledgment

While the results are fully due to the author, this paper was mostly generated using the large language model Gemini 3 Pro with prompting from the author. The author takes full responsibility for its contents.

References

[Can96] R. Canetti. More on BPP and the polynomial-time hierarchy. *Information Processing Letters*, 57(5):237–241, 1996.

[RS98] A. Russell and R. Sundaram. Symmetric alternation captures BPP. *Computational Complexity*, 7(2):152–162, 1998.

[Cai07] J.-Y. Cai. $\mathsf{S}_2^{\mathsf{P}} \subseteq \mathsf{ZPP}^{\mathsf{NP}}$. *Journal of Computer and System Sciences*, 73(1):25–35, 2007.

[MP91] N. Megiddo and C. H. Papadimitriou. On total functions, existence theorems and computational complexity. *Theoretical Computer Science*, 81(2):317–324, 1991.