On the Complexity of Formal Reasoning in State Space Models (Extended Abstract)

Eric Alsmann[0000−0002−2603−7827] and Martin Lange[0000−0002−1621−0972]

University of Kassel, Kassel, Germany
{eric.alsmann,martin.lange}@uni-kassel.de

Abstract. We study the computational complexity of the satisfiability problem for state space models (ssmSAT), a recurrent architecture that has recently emerged as a promising alternative to the widely used transformer architecture in language modelling. We show that ssmSAT is undecidable in the general case. However, under certain practically motivated restrictions, the problem becomes decidable, and we establish complexity bounds for these cases. When the context length is bounded, ssmSAT is NP-complete. When the model is quantised, i.e. restricted to fixed-width arithmetic, satisfiability remains decidable. Nevertheless, the problem remains computationally hard: depending on the encoding, we identify instances where ssmSAT, when restricted to fixed-width arithmetic, is PSPACE-complete or lies between PSPACE and EXPSPACE, depending on the encoding of the bit-width. Given these results, we discuss possible implications for the verification of state-space models used in language modelling.

Keywords: state space models · formal reasoning · complexity.

1 Introduction

State Space Models (SSM) have recently emerged as an promising alternative to transformer-based architectures for sequence modeling tasks. By relying on linear recurrences and pointwise transformations, SSM are able to capture long-range dependencies while maintaining computational simplicity. This structural difference compared to transformers raises new questions about the formal properties of SSM. While the empirical performance of SSM has been studied extensively, their formal verification remains largely unexplored. In particular, understanding the complexity of basic reasoning tasks, such as satisfiability and reachability, is essential for assessing the reliability and robustness of models deployed in safety-critical applications.

In this paper, we initiate a systematic investigation of the satisfiability problem for SSM. The satisfiability problem asks whether there exists an input sequence that causes the model to reach an accepting configuration. We first show that, in full generality, the problem is undecidable by a reduction from the halting problem for Minsky machines. Motivated by practical settings, we then consider two natural restrictions that make the problem decidable. First, we study SSM

under bounded context length, a common assumption in deployed language models. We show that in this case, the satisfiability problem is NP-complete. Second, we investigate SSM operating over fixed-width arithmetic, reflecting the use of quantised computation in practical implementations. We prove that the satisfiability problem is PSPACE-complete when the bit-width is constant or encoded in unary and in EXPSPACE when the bit-width is encoded in binary.

These results establish a first complexity landscape for formal reasoning in SSM. They provide a foundation for future work on verification techniques and a better theoretical understanding of the strengths and limitations of SSM.

Recent work has explored the expressive power of SSM. Sarrof et al. [11] showed that general SSM can express all star-free regular languages and that a restricted class of SSM exactly characterises the star-free languages. Furthermore, Merrill et al. [7] proved that quantised SSM can recognise only languages contained in the class TC^0. Related complexity questions have also been investigated for transformer architectures: Sälzer et al. [10] studied the satisfiability problem for transformer encoders, establishing computational hardness results similar to ours.

2 State Space Models

We consider State Space Models (SSM) in the sense of [4], adopting the formalisation introduced in [11] for a structured analysis. An SSM layer takes an input sequence $\boldsymbol{x_1} \cdots \boldsymbol{x_n}$ and maintains a hidden state $\boldsymbol{h}_t$ that evolves via an input-dependent linear recurrence and produces outputs through a pointwise transformation ϕ.

$$\begin{aligned} \boldsymbol{h}_t &= A(\boldsymbol{x}_t) \cdot \boldsymbol{h}_{t-1} + B(\boldsymbol{x}_t) \\ \boldsymbol{y}_t &= \phi(\boldsymbol{x}_t, \boldsymbol{h}_t) \end{aligned}$$

A full SSM working over some finite alphabet Σ comprises a sequence of such layers, followed by an output projection and preceded by an embedding function which maps the input word to the initial sequence of vectors. Given an input sequence, the model applies each layer successively, using the output of one layer as the input to the next. The final output is obtained by applying the output function element-wise to the top layer's result.

SSM architectures used in practice mainly differ on the allowed functions for A, B and ϕ. While some architectures are time-invariant [6, 12, 9], meaning that A is not input-dependent, others allow A to be an arbitrary smooth function [3, 1, 13]. However, almost all architectures assume $A(\boldsymbol{x}_t)$ to be a diagonal matrix. The pointwise transformation ϕ is usually a non-linear function using some kind of linear projection together with non-linear activations like GeLU or Sigmoid. Established upper bounds will hold for any reasonable choice of functions, subsuming all variants of SSM used in practice. The only assumption we make is that all functions can be computed in polynomial-time. This ensures that for a given input word, the output of the SSM can be also computed in polynomial

time, depending on the input length. For the lower bounds we use a fairly weak choice of functions by assuming A, B to be input-dependent linear maps and all pointwise applied functions to be simple Feedforward Neural Networks (FNN). In this work we only consider SSM for sequence classification. In this case the output vector of the last input symbol is used to determine whether the input sequence is accepted or not. The leads to the satisfiability problem for SSM SSMSAT which is defined as follows:

SSMSAT
Input: SSM $\mathcal{S}$ over alphabet Σ.
Question: Is there a word $w \in \Sigma^*$, such that $\mathcal{S}(w) = 1$?

In order to obtain upper complexity bounds for this problem, we need to measure the representation size of an SSM. We measure this representation size in the usual way, meaning given an SSM over an alphabet Σ, with vector dimension d and number of layers L the size is $|\mathcal{S}| = |\Sigma| + L + d$. We now assume the syntactic representation of $\mathcal{S}$ to be polynomial in $|\mathcal{S}|$. For a given word, the output of an SSM is computed layer-wise and each layer only requires a linear amount of calculations. This leads to the polynomial-evaluation property for SSM.

Theorem 1. *Given an SSM $\mathcal{S}$ over an alphabet Σ and a word $w \in \Sigma^*$, the output $\mathcal{S}(w)$ can be computed in time polynomial in $|\mathcal{S}| + |w|$.*

3 Satisfiability for General SSM is Undecidable

We show the undecidability of SSMSAT by reducing from the halting problem of Minsky Machines, a classical undecidable problem [8]. The key idea is to simulate the computation of a Minsky Machine within the framework of an SSM, thereby encoding the question of whether a Minsky Machine halts as a satisfiability question for an SSM. A Minsky Machine operates on two counters and transitions between states based on increment, decrement, and conditional zero-check instructions. To simulate such computations using an SSM, the approach separates control-flow information (machine states and transition rules) from the counter values. The control flow is encoded directly into the input sequence, while the counter values are computed internally by the SSM through its linear recurrence mechanism.

We construct a SSM with three layers. The first layer reconstructs the previous machine state at each step, allowing access to state transitions. The second layer uses the linear recurrence and the pointwise applied neural network to simulate counter updates and perform consistency checks: it verifies that the encoded transition is valid, and in the case of conditional transitions, correctly handles zero-checks. A final layer ensures that the entire sequence corresponds to a valid run by aggregating error flags. The construction guarantees that the SSM accepts a sequence if and only if it encodes a valid halting run of the given Minsky Machine.

Theorem 2. *SSMSAT is undecidable.*

4 Bounded Context Length

After establishing the undecidability of the general model, we now consider two natural restrictions that render the problem decidable and for which we establish complexity bounds. The first restriction, commonly encountered in practical language models, is a bound on the context length. When the context length is fixed and not part of the input, the satisfiability problem becomes trivial, as only a constant number of input sequences need to be checked. Therefore, we focus on the following problem, denoted SSMSAT$^{\leq}$:

SSMSAT$^{\leq}$
Input: SSM $\mathcal{S}$ over alphabet Σ and number n (in unary).
Question: Is there a word $w \in \Sigma^*$ with $|w| \leq n$, such that $\mathcal{S}(w) = 1$?

Membership in NP follows from a standard guess-and-verify approach. We non-deterministically select a word w with $|w| \leq n$ and verify whether $\mathcal{S}$ accepts it. Due to the assumed polynomial-evaluation property of SSM, this verification can be performed in polynomial time. To establish NP-hardness, we provide a reduction from the well-known NP-complete problem Zero-One Linear Programming [5]. In this problem, given a matrix $A \in \mathbb{R}^{d \times d}$ and a vector $\mathbf{b} \in \mathbb{R}^d$, the task is to determine whether there exists a vector $\mathbf{x} \in \{0,1\}^d$ such that $A\mathbf{x} = \mathbf{b}$. Our reduction constructs a single-layer SSM that accepts an input of maximum length d encoding $\mathbf{x}$ as a sequence of unit vectors. The linear recurrence is used to compute $A\mathbf{x}$. The output of the SSM is determined by a neural network that verifies equality with $\mathbf{b}$.

Theorem 3. *SSMSAT$^{\leq}$ is NP-complete.*

5 Fixed-Width Arithmetic

This section establishes the precise complexity of the satisfiability problem for SSM operating over fixed-point arithmetic. Our notion of fixed-width arithmetics are representations of numbers using a fixed amount of bits, like floating- or fixed-point arithmetic. Our results are independent of the specific choice of an implementation. We assume that all values represented in a fixed-width arithmetic use b bits for representing numbers. We say that an SSM works over fixed-with arithmetics, if all computations and values occuring in the computation of the SSM are carried out using only b bits. Therefore, for given $b \in \mathbb{N}$ we define the problem:

b-SSMSAT$^{\text{fix}}$
Input: SSM $\mathcal{S}$ over alphabet Σ
Question: Is there a word $w \in \Sigma^*$ such that $\mathcal{S}(w) = 1$ when $\mathcal{S}$ works over fixed-width arithmetic with b bits?

The upper bound is obtained by exploiting the finiteness of the reachable configuration space under fixed-point arithmetic. Specifically, any SSM with L

layers, d-dimensional hidden states, and bit-width b can produce at most $2^{L \cdot d \cdot b}$ distinct internal states. A simple pigeonhole argument shows that if the SSM accepts any input at all, it must also accept an input of length exponential in the model size. Additionally, the next hidden state of an SSM only depends on the previous hidden state and the next vector of the input sequence. This enables a nondeterministic algorithm that guesses such an input symbol-by-symbol while tracking the internal state with polynomial space, leading to a PSPACE membership result.

The lower bound is shown via a reduction from the satisfiability problem for Linear Temporal Logic on finite traces (LTL_f) [2], which is known to be PSPACE-hard. The reduction constructs, for any given LTL_f formula φ, a corresponding SSM $\mathcal{S}_\varphi$ that accepts exactly the models of φ. The construction encodes subformulas of φ layer-wise in the SSM, carefully respecting their structural dependencies. Atomic propositions, boolean operations and temporal operators are translated into fixed-point computations using the SSM's linear recurrence and pointwise output function ϕ. The encoding only requires fixed-point precision of 4 bits, ensuring that the hardness result holds already for modest bit-widths.

Together, these results yield the following tight characterisation:

Theorem 4. *b-SSMSAT^{fix} is PSPACE-complete when $b \geq 4$.*

When the bit-width is part of the input and encoded in unary, the PSPACE-membership still holds, because the space requirements of $L \cdot d \cdot b$ is still polynomial in the size of the input. However, when the bit-width is encoded in binary the first natural upper bound is EXPSPACE, because now an internal state has size $L \cdot d \cdot 2^b$. The question whether the problem also becomes EXPSPACE-complete is an open problem for future research.

6 Outlook

The satisfiability problem studied in this paper provides a natural foundation for the study on formal verification of State Space Models. Although abstract, it captures the core challenge underlying many verification tasks, such as proving or refuting safety properties. Its general formulation, detached from specific property types, ensures that undecidability and complexity results immediately extend to broader reasoning problems. Thus, satisfiability serves as a baseline for understanding the fundamental limits of verifying SSM and motivates future research into efficient and sound verification techniques.

References

1. De, S., Smith, S.L., Fernando, A., Botev, A., Cristian-Muraru, G., Gu, A., Haroun, R., Berrada, L., Chen, Y., Srinivasan, S., Desjardins, G., Doucet, A., Budden, D., Teh, Y.W., Pascanu, R., Freitas, N.D., Gulcehre, C.: Griffin: Mixing Gated Linear Recurrences with Local Attention for Efficient Language Models (Feb 2024). https://doi.org/10.48550/arXiv.2402.19427

2. De Giacomo, G., Vardi, M.Y.: Linear temporal logic and linear dynamic logic on finite traces. In: Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence. pp. 854–860. IJCAI '13, AAAI Press, Beijing, China (Aug 2013)
3. Gu, A., Dao, T.: Mamba: Linear-Time Sequence Modeling with Selective State Spaces. In: First Conference on Language Modeling (Aug 2024)
4. Gu, A., Goel, K., Re, C.: Efficiently Modeling Long Sequences with Structured State Spaces. In: International Conference on Learning Representations (2022)
5. Karp, R.M.: Reducibility among Combinatorial Problems. In: Miller, R.E., Thatcher, J.W., Bohlinger, J.D. (eds.) Complexity of Computer Computations: Proceedings of a Symposium on the Complexity of Computer Computations, Held March 20–22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York, and Sponsored by the Office of Naval Research, Mathematics Program, IBM World Trade Corporation, and the IBM Research Mathematical Sciences Department, pp. 85–103. Springer US, Boston, MA (1972). https://doi.org/10.1007/978-1-4684-2001-2_9
6. Mehta, H., Gupta, A., Cutkosky, A., Neyshabur, B.: Long range language modeling via gated state spaces. In: The Eleventh International Conference on Learning Representations (2023)
7. Merrill, W., Petty, J., Sabharwal, A.: The illusion of state in state-space models. In: Proceedings of the 41st International Conference on Machine Learning. ICML'24, vol. 235, pp. 35492–35506. JMLR.org, Vienna, Austria (Jul 2024)
8. Minsky, M.L.: Computation: Finite and Infinite Machines. Prentice-Hall, Inc., USA (1967)
9. Orvieto, A., Smith, S.L., Gu, A., Fernando, A., Gulcehre, C., Pascanu, R., De, S.: Resurrecting recurrent neural networks for long sequences. In: Proceedings of the 40th International Conference on Machine Learning. ICML'23, JMLR.org, Honolulu, Hawaii, USA (2023)
10. Sälzer, M., Alsmann, E., Lange, M.: Transformer Encoder Satisfiability: Complexity and Impact on Formal Reasoning. In: The Thirteenth International Conference on Learning Representations (Oct 2024)
11. Sarrof, Y., Veitsman, Y., Hahn, M.: The Expressive Capacity of State Space Models: A Formal Language Perspective. In: The Thirty-eighth Annual Conference on Neural Information Processing Systems (Nov 2024)
12. Sun, Y., Dong, L., Huang, S., Ma, S., Xia, Y., Xue, J., Wang, J., Wei, F.: Retentive Network: A Successor to Transformer for Large Language Models (Aug 2023). https://doi.org/10.48550/arXiv.2307.08621
13. Yang, S., Wang, B., Shen, Y., Panda, R., Kim, Y.: Gated linear attention transformers with hardware-efficient training. In: Proceedings of the 41st International Conference on Machine Learning. ICML'24, JMLR.org, Vienna, Austria (2024)