A Formal Characterization of Non-Monotonicity in Tensor Cores

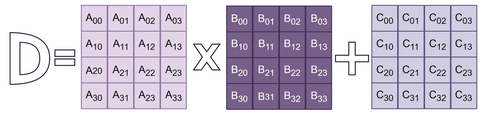
Paul Jiang[1], Vivian Zheng[2], Ganesh Gopalakrishnan[3] (Advisor)

[1]Department of Computer Science, Purdue University
[2]Department of Computer Science, Stony Brook University
[3]Kahlert School of Computing, University of Utah

Research under REU Site NSF 2244492
Trust and Reproducibility Education for Undergraduates

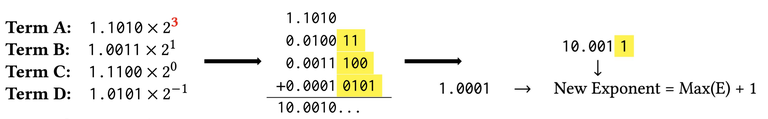
INTRODUCTION

Modern computing relies on hardware accelerators like NVIDIA **Tensor Cores** for high performance



- The internal arithmetic of these accelerators is non-standard and evolves with each generation [1] [3]
  - This can violate the fundamental mathematical property **monotonicity** when summing 4+ terms [4]
- Non-monotonicity can head to numerical errors
  - **Violate triangular inequality, interval arithmetic [4]**
- Our work builds upon that of Valpey et al. [5], who previously provided executable formal specifications of these hardwares' behaviors

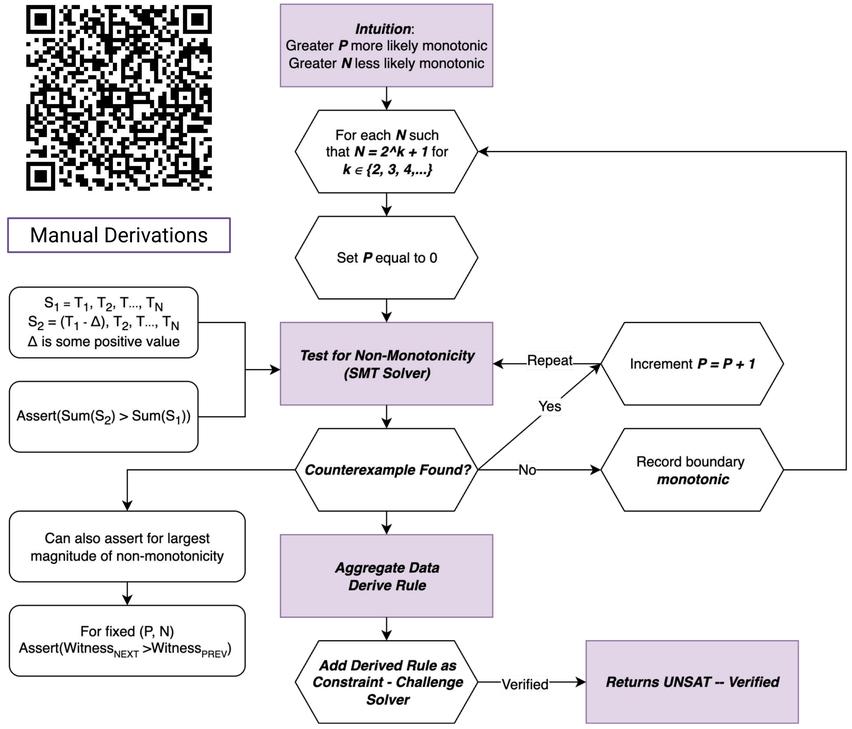
```
 1.000...000 | 00
+0.000...000 | 10
+0.000...000 | 10
+0.000...000 | 10
+0.000...000 | 10
 1.000...00  | 00

 0.111...111 | 0 0
+0.000...000 | 1 0
+0.000...000 | 1 0
+0.000...000 | 1 0
+0.000...000 | 1 0
 1.000...01  | 00
```



Research Questions

- Can we use formal methods to build an automated and exhaustive framework for verifying numerical properties in hardware accelerators?
- Can this framework identify the specific hardware parameters required to guarantee correct behavior, such as monotonicity?

METHODOLOGY



- We primarily investigate monotonicity in addition
- **Example P = 0, N = 5:**
  - $2^0 + 2^{-24} + 2^{-24} + 2^{-24} + 2^{-24} = 2^0$
  - $(2^0 - 2^{-24}) + 2^{-24} + 2^{-24} + 2^{-24} + 2^{-24} = 2^0 + 2^{-23}$
- To study this, we translate the property of monotonicity into a logical query for SMT
  - Two inputs sets, **S1 = {a,...,x}** and **S2 = {a,...,y}** where **y > x**
    → Find values such that the hardware sum of S1 > the hardware sum of S2
- Our approach involves systematically varying the model's parameters
  - We adjust the number of terms (***N***), and the number of internal padding bits (***P***)
  - This tells us whether non-monotonicity can arise under these specifications

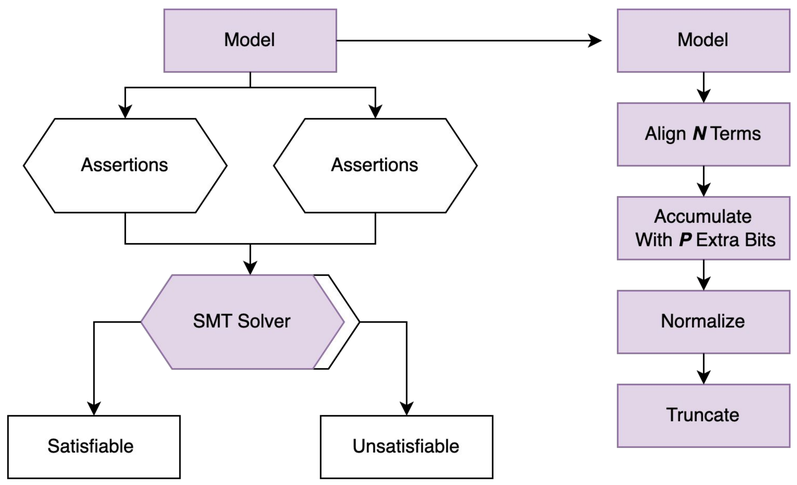
From this data, we manually derive conditions that define the behavior of this phenomena

- These conditions are fed back to the SMT solver and challenged
  - If the Solver returns UNSAT, the derived rules are exhaustively and provably correct

PRELIMINARIES

Satisfiability Modulo Theories (SMT) [2] Solvers are constraint reasoning tools designed **automatically analyze the satisfiability of logical formulas**

- Recently become powerful enough to be practical for floating point
- Standard SMT floating-point libraries are IEEE-754 compliant
- To accurately leverage this tool for non-standard hardware, we create custom encodings built from the ground up with **bit-blasting**



RESULTS

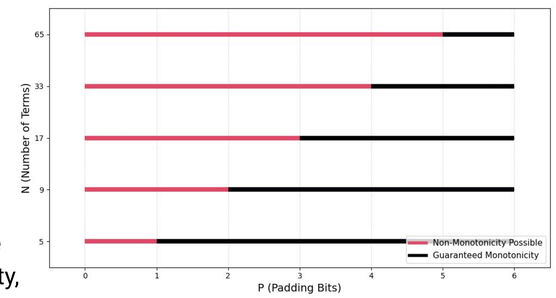
- We reveal that a block-adder is provably monotonic when the number of padding bits (p) is greater than the floor of the base-2 logarithm of the number of addends minus two

$$p \leq \lfloor (\log_2(n-1)-2) \rfloor$$

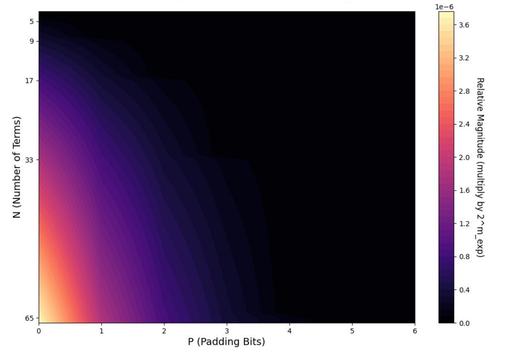
- We also derive a formula to calculate the maximum magnitude of non-monotonicity, M, for any given (n, p) configuration and maximum exponent (m_exp) for FP32 accumulation

$$M(n,p,m\_exp) = 2^{(m\_exp - (24 + p - \log(n-1)))} - 2^{(m\_exp - 23)}$$

- With this, hardware architects can eliminate non-monotonicity at the design stage
- Software engineers and numerical analysts can bound potential error when leveraging existing hardware
  - → **enables the development of more robust algorithms**



(1) Required Padding Bits ('p') for Guaranteed Monotonicity



(2) Relative Magnitude of Non-Monotonicity

CONCLUSION

- Our SMT-based approach transforms the process of hardware verification from a laborious, manual task into a systematic, semi-automated, and formal one
- We demonstrate that our framework can be used to identify the exact hardware configurations to guarantee fundamental mathematical properties like monotonicity

REFERENCES

[1] Massimiliano Fasi, Nicholas J Higham, Mantas Mikaitis, and Srikara Pranesh. Numerical Behavior of NVIDIA Tensor Cores. PeerJ Computer Science, 7:e330, 2021.
[2] Leonardo De Moura and Nikolaj Bjørner. 2008. Z3: an efficient SMT solver. In Proceedings of the Theory and practice of software, 14th international conference on Tools and algorithms for the construction and analysis of systems (TACAS'08/ETAPS'08). Springer-Verlag, Berlin, Heidelberg, 337–340.
[3] Xinyi Li, Ang Li, Bo Fang, Katarzyna Swirydowicz, Ignacio Laguna, and Ganesh Gopalakrishnan. FTTN: Feature-Targeted Testing for Numerical Properties of nvidia & AMD Matrix Accelerators. In 2024 IEEE/ACM 24th International Symposium on Cluster, Cloud and Internet Computing (CCGrid). IEEE, 2024.
[4] Mantas Mikaitis. Monotonicity of multi-term floating-point adders. IEEE Transactions on Computers, 73(6):1531–1543, 2024.
[5] Benjamin Valpey, Xinyi Li, Sreepathi Pai, and Ganesh Gopalakrishnan. An SMT Formalization of Mixed-Precision Matrix Multiplication: Modeling Three Generations of Tensor Cores. In NASA Formal Methods Symposium, pages 360–379, Cham, Jun 2025. Springer Nature Switzerland.