Precision Unveiled: Formal Methods in Dot Product Accumulate ULP Analysis

Suraj Kamble, Mohit Choradia, Vichal Verma, Kyung-Nam Han
Microsoft Technology Pvt Ltd

***Abstract*-Dot product accumulate operations are fundamental in machine learning, yet their numerical precision can vary across platforms due to differences in floating-point formats and execution models. This poster presents a formal verification methodology for rigorously bounding and analyzing precision errors in dot product computations, using Unit in the Last Place (ULP) as the key metric. The approach leverages Over Constraint Analysis (OCA) for rapid case isolation and binary search strategies to efficiently identify the smallest error bounds that satisfy architectural specifications. Through formal modeling and exhaustive proof techniques, the methodology enables early detection of precision discrepancies and provides mathematical guarantees that inference results remain within safe and acceptable limits across heterogeneous systems. The results highlight the effectiveness of formal methods in ensuring robust and reliable AI model deployment.**

I. INTRODUCTION

Dot product accumulate operations are essential in machine learning, forming the basis of neural networks and matrix computations by multiplying and summing vector elements. The accuracy of these operations is highly dependent on numerical precision, which is measured using the Unit in the Last Place (ULP)—the smallest possible difference between two representable floating-point numbers near a given value. Even small ULP errors can result in noticeable deviations in dot product results, potentially affecting the reliability and performance of AI models. Precision discrepancies often arise when machine learning inference algorithms are executed on different hardware platforms, such as CPUs and accelerators, due to variations in floating-point formats and execution models.

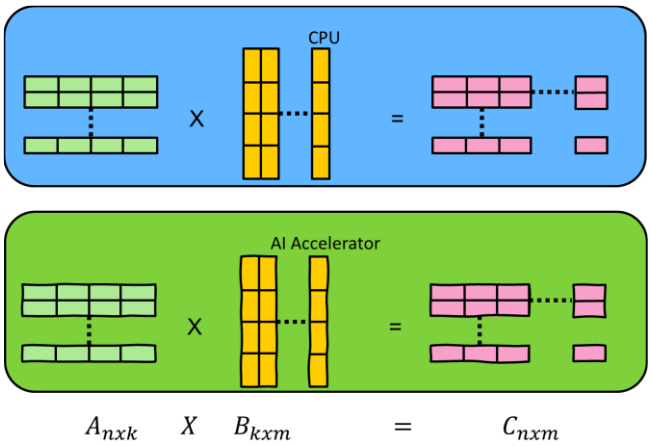

Figure 1. CPU vs AI Accelerator Dot Product Accumulate.

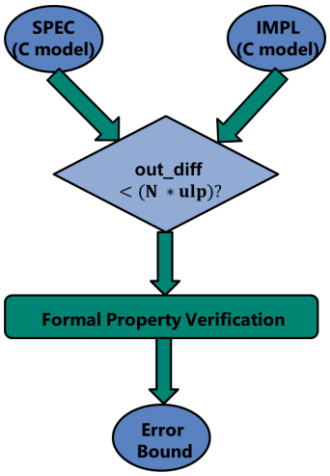

Figure 2.C Formal Property Verification.

Addressing these discrepancies requires a clear definition and enforcement of acceptable error bounds to maintain consistent and reliable inference across heterogeneous systems. One of the main challenges is the absence of standardized methods for bounding precision errors, which makes it difficult to ensure that results consistently fall within safe and acceptable limits. Formal verification methods provide a rigorous approach to modeling and bounding these errors mathematically. However, the complexity introduced by multiple multiply-accumulate operations and symbolic error bounds can lead to large formal models that are difficult to analyze. Efficient strategies, such as Over Constraint Analysis (OCA) for rapid case isolation and binary search techniques, are necessary to manage these challenges and enable early detection and analysis of precision discrepancies in dot product computations.

Challenge & Impact	Solution & Benefit
Multiple MulAdd ops and symbolic error bound forms huge formal model results blocking of proof execution.	hardcode (assume) error bound to enable proofs
Exhaustive trials for error bound are resource-heavy	Use OCA for case isolation strategy for efficient convergence

Figure 3. Challenges in C Formal Property Verification.

II. RELATED WORK

The increasing importance of precision in floating-point arithmetic has been extensively studied in various domains, ranging from scientific computing to safety-critical systems like flight control and collision avoidance. Previous work has explicitly highlighted the implications of Unit in the Last Place (ULP) errors in real-world applications. Studies such as those on the Patriot missile failure emphasize the catastrophic consequences of minute floating-point inaccuracies, which this paper builds upon to ensure robustness and reliability in dot product accumulate (DPA) computations [2]. The concept of ULP as a measure of precision in floating-point arithmetic has been foundational in understanding rounding errors. Earlier efforts, such as those discussed by Goldberg (1991) in "What Every Computer Scientist Should Know About Floating-Point Arithmetic," laid the groundwork for quantifying precision and guiding error analysis [1]. These efforts have influenced modern applications including AI systems, as they provide a means to assess computational accuracy systematically.

In the realm of floating-point dot product accumulate, prior research has explored how computational errors can propagate and amplify due to their rhythmic data flow. Applications in signal processing, machine learning, and scientific simulations underscore the need for precise arithmetic to ensure reliability and performance. This paper discusses how Formal Verification techniques can be used for “worst case error” quantification/analysis The novelty lies in applying binary search technique integrated with formal property verification to find the error-factor “N”/ factor “N” in the error-term, represented as (n * ULP).

The industrial relevance of this study extends to several fields that rely on DPA computations for accuracy and performance:

- Signal Processing: Ensuring precision in DPA is vital for avoiding error propagation, which could compromise the accuracy of signal processing tasks.
- Machine Learning: In machine learning applications, even small computational errors can lead to incorrect predictions. This research provides a foundation for reliable model training and inference.
- Scientific Simulations: High-performance scientific computations often require DPA, where precision errors can disrupt synchronization and amplify inaccuracies. The proposed techniques effectively address these challenges.

III. PROPOSED SOLUTION

The proposed solution begins with an architectural analysis, where an initial error bound (n × ULP) is suggested based on design intent and expected behavior for dot product accumulate operations. To accelerate the verification process, Over Constraint Analysis (OCA) is employed for case isolation. In this step, the formal environment is intentionally over constrained by limiting the range of computation operations and input scenarios. This restriction enables the formal verification engine to run faster proofs, quickly determining whether the error bound holds for the architect-suggested value of n.

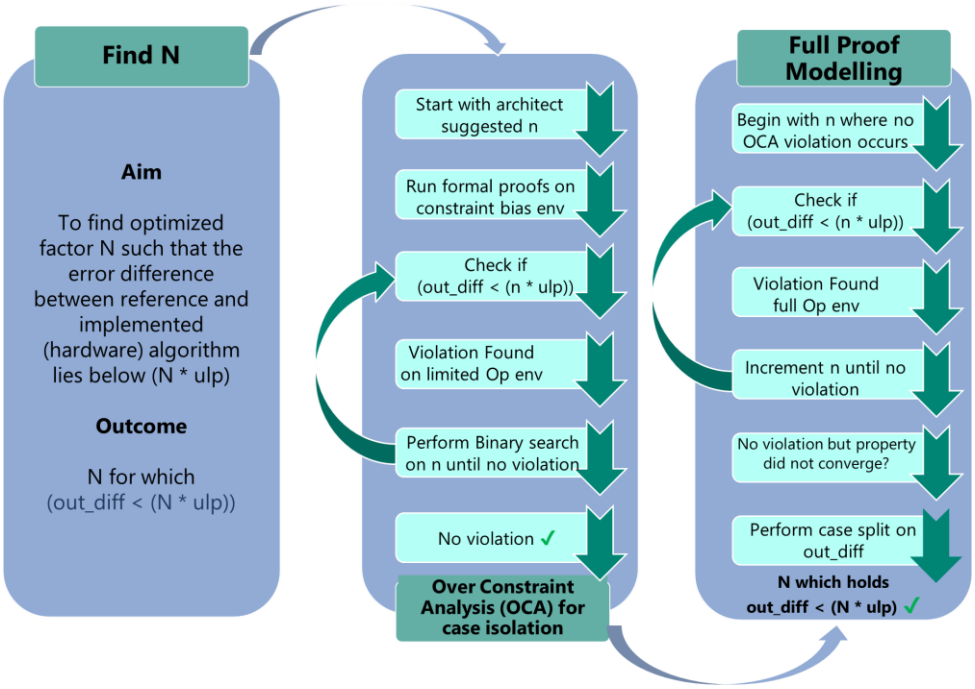

Figure 4.OCA for case isolation based Formal Verification Flow.

Efficiency is further enhanced by integrating a binary search technique within the OCA framework. Instead of exhaustively testing every possible error bound, the binary search rapidly narrows down the smallest value of n for which no violations occur in the over constrained environment. This targeted approach minimizes resource consumption and proof runtime, allowing the verification process to focus only on promising candidate bounds. Once a suitable value of n is identified through OCA and binary search, we aim for full proof.

After OCA analysis determines a viable error bound, the methodology transitions to formal verification with the actual constrained environment. Here, the full range of computation operations and correct constraints are restored, and formal proofs are executed to achieve exhaustive correctness. During this stage, case splitting and incremental search are applied to handle scenarios where the proof does not converge or where specific output differences require deeper analysis. These techniques help isolate challenging cases and ensure that the error bound remains valid across all operational scenarios, providing a mathematically rigorous guarantee that inference results stay within safe and acceptable limits across heterogeneous systems.

Over Constraint Analysis (OCA) for case isolation enables rapid validation when input space is limited, making it ideal for early error bound exploration. Full proof formal verification ensures exhaustive correctness and starting it

from OCA-cleared bounds optimizes time and memory utilization. Architect-suggested values guide this flow, aligning verification with design intent and accelerating hardware development cycles.

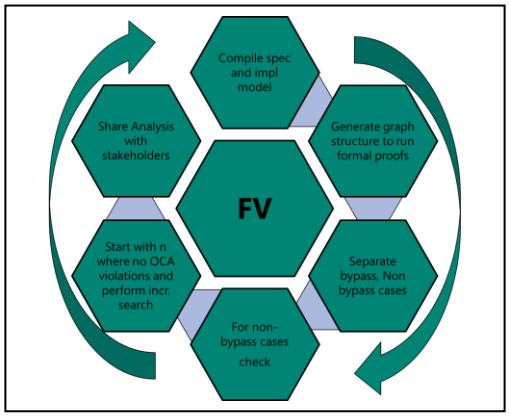

Figure 5.E2E Formal Verification.

IV. RESULTS

The proposed methodology was applied to dot product accumulate operations, focusing on bounding output error within architect-defined limits using OCA and binary search. Initial OCA runs with the architect-suggested value of n revealed counterexamples (CEX) where the error exceeded n × ULP, highlighting scenarios that required further investigation. By employing binary search within the over constrained environment, the smallest value of n with constraint bias violations was efficiently identified, significantly reducing proof runtime and memory usage compared to exhaustive approaches. Once a viable bound was found, full proof verification was performed in the real constraint environment, restoring the complete operational space.

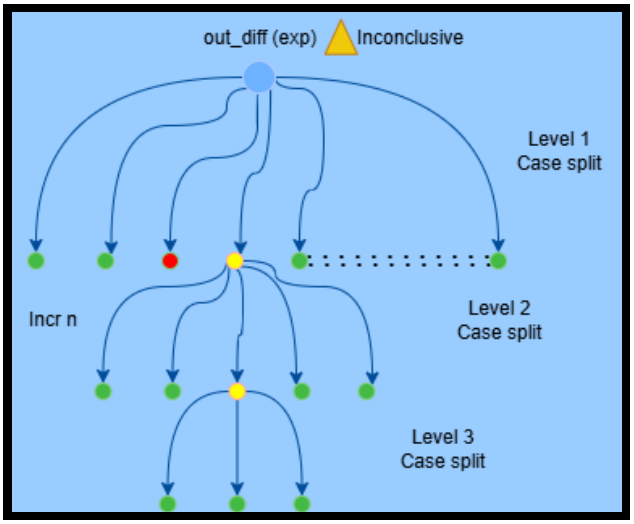

Figure 6. Case Splitting Strategy to Achieve Full Proof

Key findings include:

- OCA analysis rapidly eliminated invalid bounds and isolated cases where error exceeded n × ULP.
- Binary search enabled fast convergence to the minimum n, optimizing resource utilization.
- Case splitting was used during full proof to handle non-convergent scenarios, isolating deep corner case scenarios violating the property.
- Incremental search allowed stepwise validation of error bounds across increasingly realistic operational scenarios.

The final formal proof confirmed that, for the identified N, the output difference in dot product computations consistently remained below N × ULP, ensuring reliable inference across heterogeneous platforms.

V. CONCLUSION & FUTURE WORK

This work demonstrates that proposed solution Over Constraint Analysis (OCA) and binary search, provides an efficient and rigorous approach for bounding precision errors in dot product accumulate operations. The methodology enables early detection of error bound violations, rapid convergence to optimal bounds, and thorough validation of numerical accuracy across heterogeneous platforms. Case splitting and incremental search further enhance proof coverage, ensuring that results remain within architect-defined limits. For future work, the methodology can be extended to algorithmic error attribution, allowing targeted identification of high-error segments within complex computations and guiding focused optimization efforts for improved precision in AI accelerators.

REFERENCES

[1] Goldberg, D. (1991). "What Every Computer Scientist Should Know About Floating-Point Arithmetic." *ACM Computing Surveys*, 23(1), 5–48.
[2] Robert D. Skeel. Roundoff error and the patriot missile. SIAMNews, 25(4):11, Jul.1992.
[3] Erik Seligman, Tom Schubert, and MV Achutha Kiran Kumar. 2015. Formal verification: an essential toolkit for modern VLSI design. Morgan Kaufmann.