An Efficient Hybrid Engine to Perform Range Analysis and Allocate Integer Bit-widths for Arithmetic Circuits

Yu Pang
Chongqing University of Posts and Telecommunications
pangyu@cqupt.edu.cn

Katarzyna Radecka, Zeljko Zilic
McGill University
katarzyna.radecka@mcgill.ca
zeljko.zilic@mcgill.ca

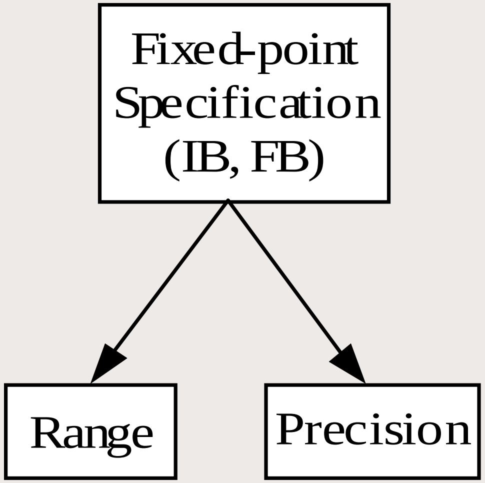
Outline

- Introduction
- Background
- Proposed Solution
- Experimental Results
- Conclusion

Fixed-point Representation

- Fixed-point revival: FPGAs, reconfigurable computing

- Q ⟶ [IB].[FB]
- IB = # of integer bits
- FB = # of fractional bits
- Wordlength = IB + FB
- IB ⟶ Range analysis
- FB ⟶ Precision analysis



Range analysis

An important task in DSP circuit synthesis

- Allocate integer bit-widths (IBs)
- Issue: Over- or under-allocate bit-widths
- Too few bits cause overflow, too many are costly
- Exact ranges lead to the smallest bit-widths and a reduction in the circuit area and delay

Background

- Dynamic Methods: simulation-based
  - Low efficiency
- Static Methods

➢ Interval Arithmetic (IA): coarse results

➢ Affine Arithmetic (AA): tighter ranges than IA

◆ An interval $[x_{min}, x_{max}]$ $x_A = x_0 + x_1\varepsilon$

◆ $$x_0 = \frac{x_{max} + x_{min}}{2}, \quad x_1 = \frac{x_{max} - x_{min}}{2}$$

◆ The intermediate signal or the output is represented as a first degree polynomial: $y_A = y_0 + y_1\varepsilon_1 + y_2\varepsilon_2 \ldots + y_n\varepsilon_n$

◆ $\varepsilon_1, \varepsilon_2, \ldots \varepsilon_n$ bolic *uncertain variables* , lie in the range [-1, +1]

Range: Metrics and Goals

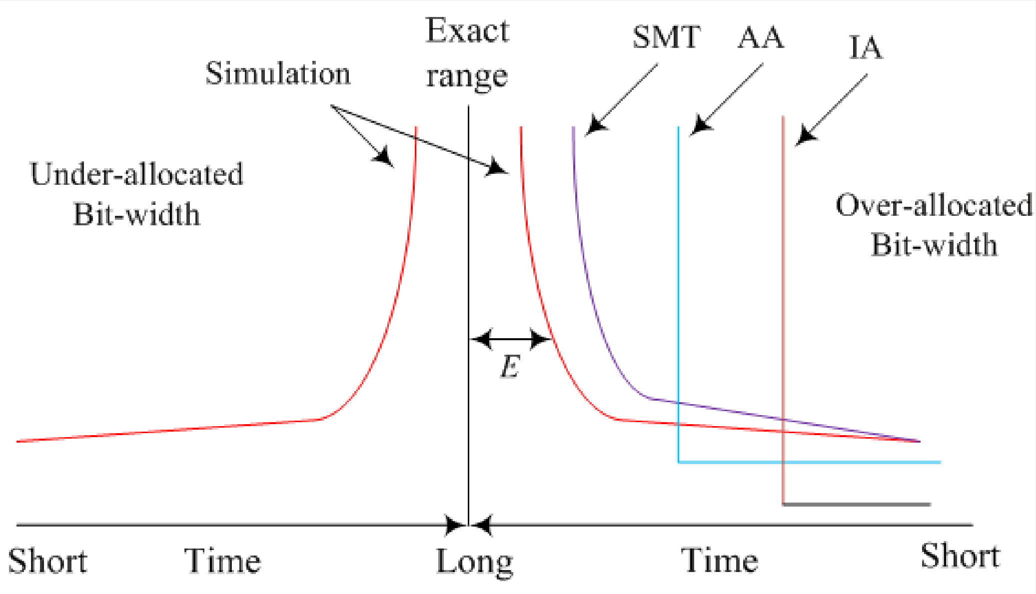
- Error Metrics

➢ Error bound E: the largest difference between the exact and the obtained ranges

➢ Error ratio: $e_r = \left(\frac{\text{obtained range}}{\text{exact range}} - 1\right) * 100\%$

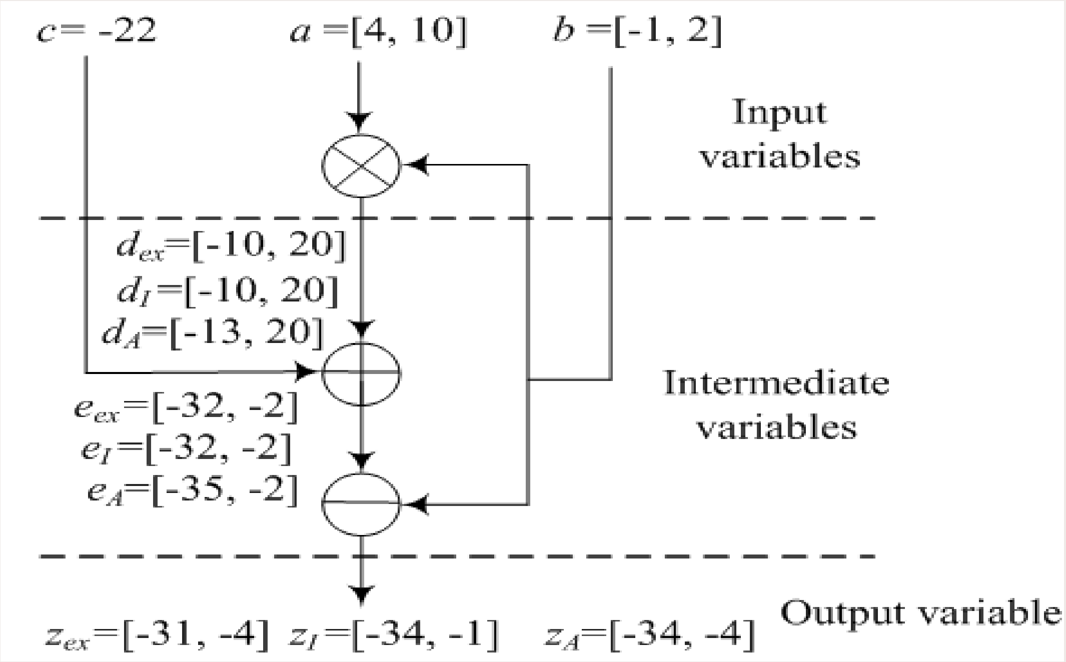
- Goal: -find the smallest value of E or e_r

  -cannot underestimate the bit-width



Example: IA and AA

Datapath *z=ab+c-b*



$$E = -31 - (-34) = 3$$

$$e_{r\_I} = \left(\frac{-1+34}{-4+31} - 1\right) * 100\%$$
$$= 22.2\%$$

$$e_{r\_A} = 11.1\%$$

- Range by AA is tighter than that by IA
- *e* and *z* require 7 signed integer bits by IA and AA. However, 6 bits suffice for their exact ranges, so IA and AA cause additional hardware area

Improving Range Analysis

- Correlation is a major cause of overestimation
- Two correlated variable might not reach their maximum or values at the same time
  - handling the correlation becomes a key task in range analysis
- Correlation in two monomials means that if the value of one monomial changes, the other will follow the change
- Example:

  $z=ab + c - b$ exhibits correlation

  <- ab and $-b$ both include variable "b"

Applying Arithmetic Transforms

- Arithmetic Transform (AT): useful to explore precision
- Definition:

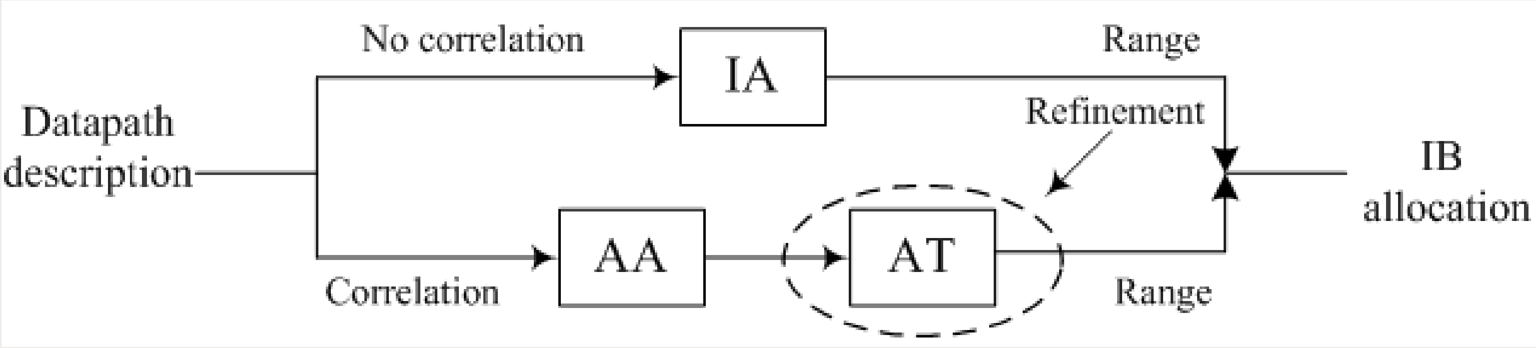
$$AT(f) = \sum_{i_1=0}^{1} \sum_{i_2=0}^{1} \cdots \sum_{i_n=0}^{1} (c_{i_1 i_2 \dots i_n} * x_1^{i_1} x_2^{i_2} \dots x_n^{i_n})$$

- If *X* and *Y* are unsigned input factional numbers represented by 2 and 3 bits, the polynomial is:

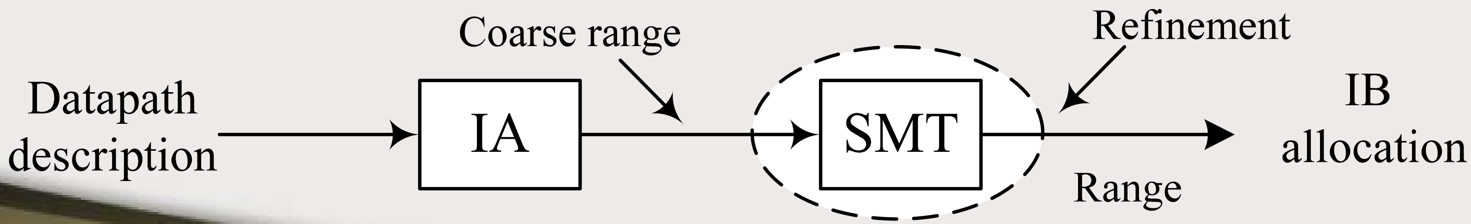
$$\begin{aligned} AT[f(X,Y)] &= AT(2X^2 + Y^2) = 2AT(X^2) + AT(Y^2) \\ &= 2 * (\textstyle\sum_{i=0}^{1} 2^{-(i+1)} x_i)^3 + (\textstyle\sum_{k=0}^{2} 2^{-(k+1)} y_i)^2 \\ &= 0.25x_0 + 0.5625x_1x_0 + 0.03125x_1 + 0.25y_0 + 0.25y_1y_0 \\ &= 0.125y_2y_0 + 0.0625y_1 + 0.0625y_2y_1 + 0.015625y_2 \end{aligned}$$

Proposed Solution

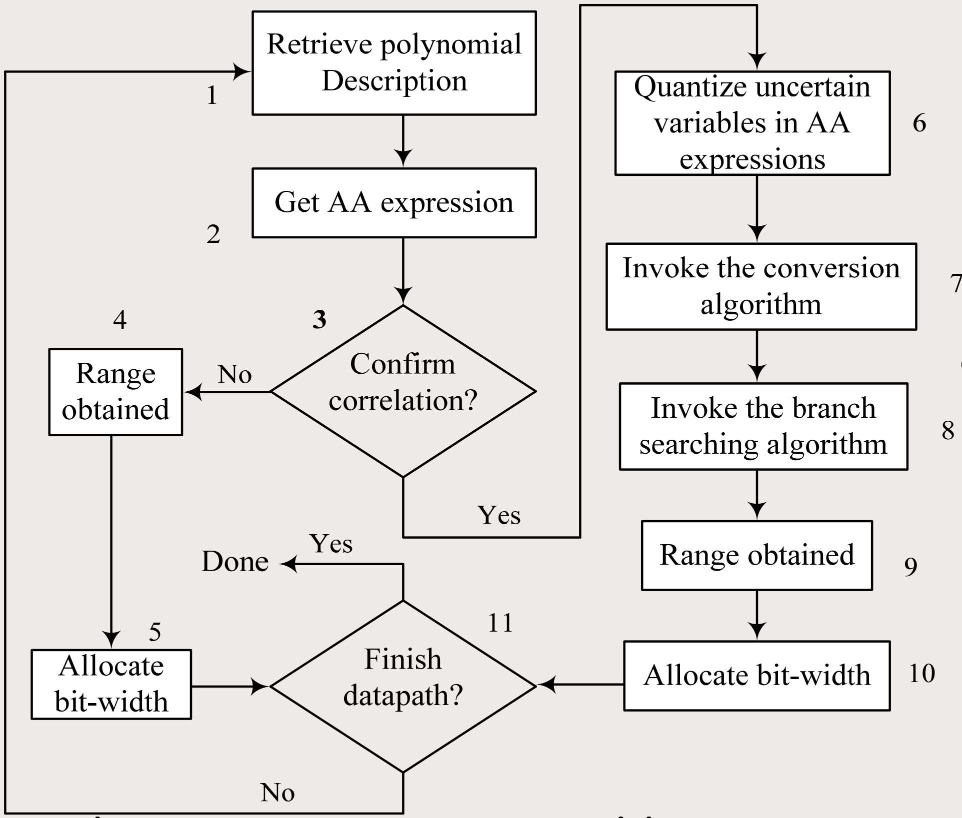
- The model of the proposed algorithm



- The algorithm invokes different methods to handle a datapath
- Distributes correlation to AA and AT for the two-step processing
- SMT-based method is time consuming, as it invokes underlying exhaustive engine pretty much all the time to refine initial IA ranges



Range Analysis - Details



- Takes the polynomial representing a datapath (Step 1)
  -> generates AA expression (Step 2)
- If no correlation, IA is used to compute the exact range
  -> bit-widths determined (Steps 4 and 5)

- Otherwise, uncertain variables are quantized in the AA expression (Step 6) and the conversion algorithm is invoked to obtain AT (Step 7)
- Then the branch-and-bound searching algorithm is applied to find the upper and the lower bounds, and estimate the bound intervals (Step 8)
- Finally, the IBs of the datapath are allocated (Steps 9 and 10)

Applying the Scheme

- Example: Consider only primary output *z=ab+c-b*

$$z_A = e_A - b_A = -19 + 1.5\varepsilon_1 + 9\varepsilon_2 + 4.5\varepsilon_2\varepsilon_1$$

- ε_1 and ε_2 belong to [-1, 1], which can be represented as a signed fractional number:

sign	0.5	0.25	0.125
x_0	x_1	x_2	x_3

- $\tilde{\varepsilon}_1$ and $\tilde{\varepsilon}_2$ are quantized uncertain variables to replace ε_1 and ε_2
- m_1 and m_2 represent the number of quantization bits

$$AT(\tilde{\varepsilon}_1) = (1 - 2x_0)\sum_{i=1}^{m_1} 2^{-i} x_i$$
$$AT(\tilde{\varepsilon}_2) = (1 - 2y_0)\sum_{k=1}^{m_2} 2^{-k} y_k$$

Finding Exact Range

- As smallest integer unit is “1”, the error between the exact value and the calculated value contained:

$$|z_{ex} - z| < 0.5 \;\Rightarrow z - 0.5 < z_{ex} < z + 0.5 \qquad (1)$$

- The exact output z_{ex} is in [z-0.5, z+0.5] based on the calculated value z, so the approximation error will be limited to 1
- Based on Eqn. (1),

$$|(1.5\varepsilon_1 - 1.5\tilde{\varepsilon}_1) + (9\varepsilon_2 - 9\tilde{\varepsilon}_2) + (4.5\varepsilon_2\varepsilon_1 - 4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2)| < 0.5 \qquad (2)$$

- In order to satisfy Eqn. (2):

$$|(1.5\varepsilon_1 - 1.5\tilde{\varepsilon}_1)| + |(9\varepsilon_2 - 9\tilde{\varepsilon}_2)| + |(4.5\varepsilon_2\varepsilon_1 - 4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2)| < 0.5$$

$$\Rightarrow |1.5\tilde{\varepsilon}_1|_{err} + |9\tilde{\varepsilon}_2|_{err} + |4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2|_{err} < 0.5 \qquad (3)$$

where $|1.5\tilde{\varepsilon}_1|_{err}, |9\tilde{\varepsilon}_2|_{err}$ and $|4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2|_{err}$ are error bounds of monomials

Finding Max Error

- Address each monomial individually (assume m_1=m_2):

$$|4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2|_{err} < 0.5 \Rightarrow 4.5[1-(1-2^{-m+1})^2] < 0.5 \quad (4)$$

●The reason to choose the monomial $4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2$ as the first one is that it has uncertainty degree “2” while for the remaining monomials $1.5\tilde{\varepsilon}_1$ and $9\tilde{\varepsilon}_2$ this degree is one

●The largest approximation error is 2^{-m+1}

● The maximum error is represented as $4.5[1-(1-2^{-m+1})^2]$

●The value of m to be 6 which means $\tilde{\varepsilon}_1$ and $\tilde{\varepsilon}_2$ both require at least six bits to satisfy Eqn. (4)

Exploring Fractional Parts

- Maximum fractional value represented by six bits is 0.96875, and by substituting $\tilde{\varepsilon}_1$ = $\tilde{\varepsilon}_2$ = 0.96875, real value of $4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2$ is 4.5 * 0.96875^2 = 4.223
- The maximum error is 4.5 – 4.223 = 0.277 for the monomial $4.5\varepsilon_2\varepsilon_1$
- The remaining error space is 0.5 – 0.277 = 0.223, so

$$|1.5\tilde{\varepsilon}_1|_{err} + |9\tilde{\varepsilon}_2|_{err} < 0.223 \qquad (5)$$

- Next, we explore the monomial "$1.5\tilde{\varepsilon}_1$".

$$1.5 * 2^{-m_1+1} < 0.223$$

- $\tilde{\varepsilon}_1$ must be expressed using at least four bits, so to satisfy Eqn. (4) and (5), $\tilde{\varepsilon}_1$ is determined 6 signed bits.

$$max(1.5\tilde{\varepsilon}_1) = 1.5 * 0.96875 = 1.4531$$

Bound on Error Sum

- The obtained maximum errors for the monomial $4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2$ and $1.5\tilde{\varepsilon}_1$ are then 4.5 – 4.223 = 0.277 and 1.5-1.4531 = 0.0469
- The remaining error space: 0.5 - 0.277 -0.0469 =0.1761
- Final monomial $9\tilde{\varepsilon}_2$ must satisfy error bound:

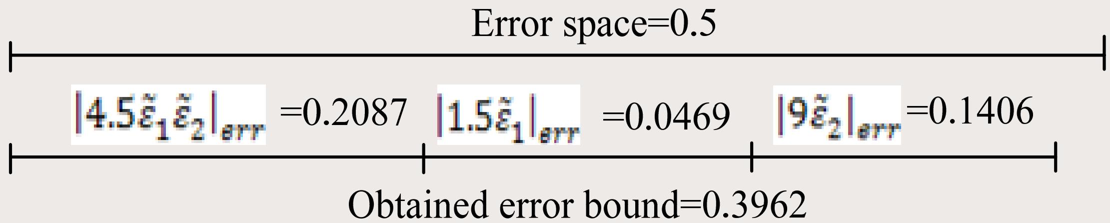
$$9 \ * 2^{-m_2+1} < \ 0.1761 \qquad (6)$$

- Obtain m_2 = 7. In combination with the bit-width of 6 in the monomial $4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2$, $\tilde{\varepsilon}_2$ is determined 7 bits
- The error bound for the monomial $9\tilde{\varepsilon}_2 = 9 * 2^{-7+1} = 0.1406$
- The maximum error of $4.5\tilde{\varepsilon}_1\tilde{\varepsilon}_2$ is re-calculated as:

$$4.5[1-(1-2^{-6+1}) * (1-2^{-7+1})] = 0.2087$$

- Finally, the error bound for all monomials is 0.3962 as:

$$|1.5\tilde{\varepsilon}_1|_{err} = 0.0469$$
$$|9\tilde{\varepsilon}_2|_{err} = 0.1406$$
$$|4.5\tilde{\varepsilon}_2\tilde{\varepsilon}_1|_{err} = 0.2087$$



Applying AT

- AT representation of *z* is determined by expanding $\tilde{\varepsilon}_1$ and $\tilde{\varepsilon}_2$ into their bit-levels:

$$AT(z) = -19 + 1.5(1-2x_0)\sum_{i=1}^{5} 2^{-i}x_i + 9(1-2y_0)\sum_{k=1}^{6} 2^{-k}y_k + 4.5\left[(1-2x_0)\sum_{i=1}^{5} 2^{-i}x_i\right]\left[(1-2y_0)\sum_{k=1}^{6} 2^{-k}y_k\right]$$

- By invoking the conversion algorithm and the branch-and-bound searching algorithm, the upper and the lower bounds for the scope of *z*, can be computed as -4.4 and -30.7
- The exact upper and lower bounds belong to the following intervals

$$\begin{cases} z_{ex\_upp} = -4.4 \pm 0.3962 = [-4.7962, -4.0038] \\ z_{ex\_low} = -30.7 \pm 0.3962 = [-31.0962, -30.3038] \end{cases}$$

- The final obtained range is [-32, -4]. The error bound *E*=1 and the error ratio is *e*= 2.86%. Both two error metrics are optimized compared to that of AA (1, 2.86%)

Experiments: Benchmarks

- Filter polynomial: $F = 4X^4 + 16X^3 + 20X^2$. The implementation has four intermediate variables ($X \in [-20, 10]$) :

  $q_1 = X^2 \quad q_2 = q_1X \quad q_3 = q_2X \quad q_4 = 4q_2 + 16q_3 \quad z = q_4 + 20q_1$

- Hermite polynomial:

$$H_6(x) = z = x^6 - 15x^4 + 45x^2 - 15$$
$$= x^2(x^2(x^2 - 15) + 45) - 15 \qquad x \in [-6, 10]$$

  The implementation contains following intermediate signals:

  $q_1 = x^2 \quad q_2 = q_1(q_1 - 15) \quad q_3 = q_1(q_2 + 45) \quad z = q_3 - 15$

- Dickson polynomial: $D_4(x, a) = x^4 - 4x^2a + 2a^2 = x^2(x^2 - 4a) + 2a^2$ (assume $x \in [-50, 50]$, $a \in [-20, 40]$). This benchmark includes two word-level input variables: $q_1 = x^2 \quad q_2 = q_1 - 4a \quad q_3 = q_1q_2 \quad q_4 = 2a^2 \quad z = q_4 + q_3$

- Multivariate Datapath: $F = 30A^2 - 60AB - 40BC$

  ($A \in [-20, 30]$, $B \in [-10, 40]$ and $C \in [-10, 30]$)

  $q_1 = 30A^2 \quad q_2 = 60AB \quad q_3 = 40BC \quad q_4 = q_1 - q_2 \quad z = q_4 - q_3$

Experimental Results

Case	Output	Range			Bit			Error Ratio (%)		Time (s)			
		AA	SMT	Ours	AA	SMT	Ours	AA	Ours	Sim	SMT	AT	Ours
Image Filter	q_1	[-350, 400]	[-1, 401]	[0, 400]	10	10	9	87.5	0	5.12	15.6	26.1	7.8
	q_2	[-8000, 7750]	[-8001, 1001]	[-8000, 1000]	14	14	14	75	0				
	q_3	[-158750,160000]	[-1, 160001]	[0, 160000]	19	19	18	99	0				
	q_4	[-511000,534000]	[-112, 512001]	[-109,512001]	21	20	20	99	<0.01				
	z	[-511000,542000]	[-2, 520001]	[-1, 520001]	21	20	20	99	<0.01				
Hermite	q_1	[-92, 100]	[-1, 101]	[0, 100]	8	8	7	92	0	9.32	52.7	83.5	33.7
	q_2	[-9036, 8948]	[-60, 8502]	[-58, 8501]	15	15	15	100	<0.01				
	q_3	[-865820, 865828]	[-42, 854501]	[-40, 854501]	21	21	21	103	<0.01				
	z	[-865835, 865813]	[-57, 854490]	[-55, 854486]	21	21	21	103	<0.01				
Dickson	q_1	[-2500, 2500]	[-1, 2501]	[0, 2500]	13	13	12	100	0	163	124	213	51.5
	q_2	[-2660, 2580]	[-162, 2582]	[-160, 2580]	13	13	13	91	0				
	q_3	[-6450000,6450000]	[-6401,6450003]	[-6401,6450001]	24	24	24	99.8	<0.01				
	q_4	[-2800, 3200]	[-1, 3201]	[0, 3200]	13	13	12	87.5	0				
	z	[-6452800,6453200]	[-6403,6453205]	[-6401,6453201]	24	24	24	99.8	<0.01				
Multi-variate polyno-mial	q_1	[-25650, 27000]	[-1, 27001]	[0, 27000]	16	16	15	95	0	>500	19	>500	9.2
	q_2	[-57000,72000]	[-48001, 72001]	[-48000, 72000]	18	18	18	7.5	0				
	q_3	[-28000, 48000]	[-16001, 48001]	[-16000, 48000]	17	17	17	18.8	0				
	q_4	[-82500, 60000]	[-45002, 60002]	[-45001, 60001]	18	17	17	35.7	<0.01				
	Z	[-130500, 97000]	[-93004, 76003]	[-93001, 76001]	18	18	18	34.6	<0.01				

Analysis of Experimental Results

- Error ratios of our method are far smaller than that of AA
- AA may require one additional bit for representing some signals, which adds to the implementation costs
- Simulations take much longer for datapaths beyond one
- SMT often needs a long time for computation
- Execution time of our method is acceptable both for high order and multivariate polynomials

Area and Delay Results

Circuit	Area (Slices)			Delay (ns)		
	Ours	AA	Saving	Ours	AA	Saving
Filter	686	740	7.3%	23.5	25.4	7.5%
Filter	725	768	5.6%	24.6	26.2	6.1%
Filter	756	787	3.9%	25.4	26.8	5.2%
Hermite	809	870	7%	31.3	33.5	6.6%
Hermite	845	897	5.8%	32	33.9	5.6%
Hermite	876	919	4.7%	32.4	34.1	5%
Dickson	532	578	8%	27.4	29.9	8.3%
Dickson	557	596	6.5%	27.9	30.2	7.6%
Dickson	588	623	5.6%	28.7	30.7	6.5%

- With increase in input ranges, the saving ratio decreases because the auxiliary area caused by additional bits is diminishing
- Our method can achieve the optimized implementations with the area smaller by around 4% to 8%, and decrease delay by around 5% to 9%.
- Calculation time of AA is around 1 second, while our method requires at most 10 – 50 seconds. Increase in computation time is justified as the obtained ranges are far tighter

Conclusion and Future Work

- Range analysis can directly impact the overall design cost and performance
- Previous methods have disadvantages of low efficiency and coarse bounds.
- Coarse ranges may generate unnecessary bits, costly circuits
- This paper propose a new static method to calculate ranges
- Method handles correlation for efficient, exact range analysis
- It combines AA and AT to find ranges efficiently

Thank You!