FPL'04

Second Order Function Approximation Using a Single Multiplication on FPGAs

Jérémie Detrey Florent de Dinechin

Projet Arénaire – LIP
UMR CNRS – ENS Lyon – UCB Lyon – INRIA 5668
http://www.ens-lyon.fr/LIP/Arenaire/

CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE
ECOLE NORMALE SUPERIEURE DE LYON
INRIA

Overview

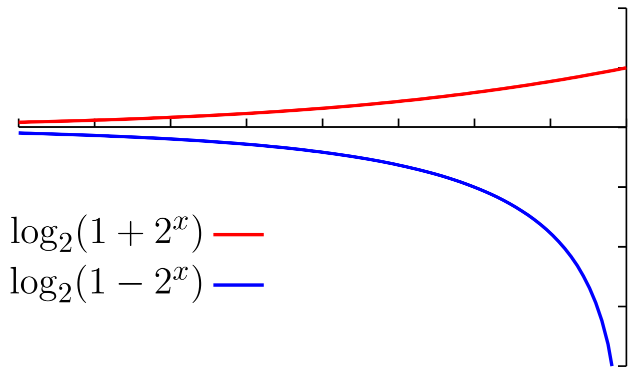
- Context
- The SMSO method
- Optimization
- Results
- Conclusion

Context

- Context
  - Function evaluation
  - State of the art
  - Objectives
- The SMSO method
- Optimization
- Results
- Conclusion

Context: function evaluation

- elementary functions $\sin(x)$, $\cos(x)$, $\log(x)$, e^x, ...
  - signal or image processing
  - neural networks
  - ...
- special functions:
  - logarithmic number system: $\log_2(1+2^x)$ and $\log_2(1-2^x)$

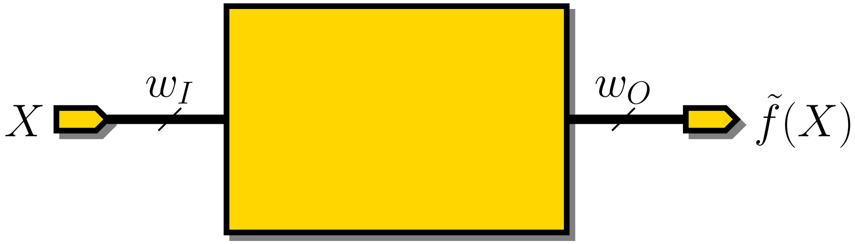


Context: function evaluation

- input:
  - function $f : [0; 1[\rightarrow [0; 1[$
  - input precision w_I
  - output precision w_O (usually $w_O = w_I$)

Context: function evaluation

- input:
  - function $f : [0;1[\to [0;1[$
  - input precision w_I
  - output precision w_O (usually $w_O = w_I$)
- output:



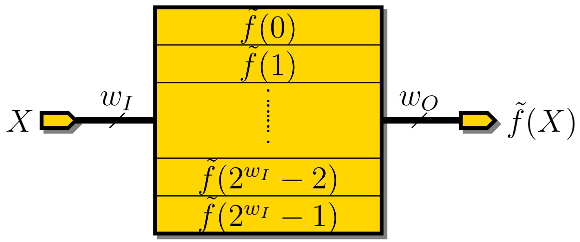
  - where $X = .x_1 x_2 \cdots x_{w_I}$
  - and $\tilde{f}(X) = Y = .y_1 y_2 \cdots y_{w_O} \approx f(X)$ at the required precision

Context: function evaluation

- input:
  - function $f : [0;1[\to [0;1[$
  - input precision w_I
  - output precision w_O (usually $w_O = w_I$)
- output:

  - where $X = .x_1 x_2 \cdots x_{w_I}$
  - and $\tilde{f}(X) = Y = .y_1 y_2 \cdots y_{w_O} \approx f(X)$ at the required precision

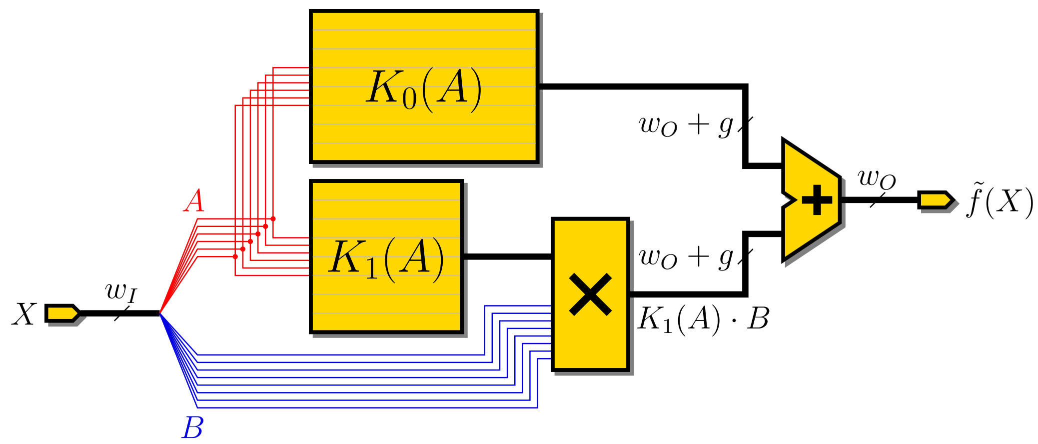
Order 0: direct look-up table



- very short critical path: only 1 table look-up
- huge look-up table: $w_O \times 2^{w_I}$ bits

Order 1: lookup-multiply method

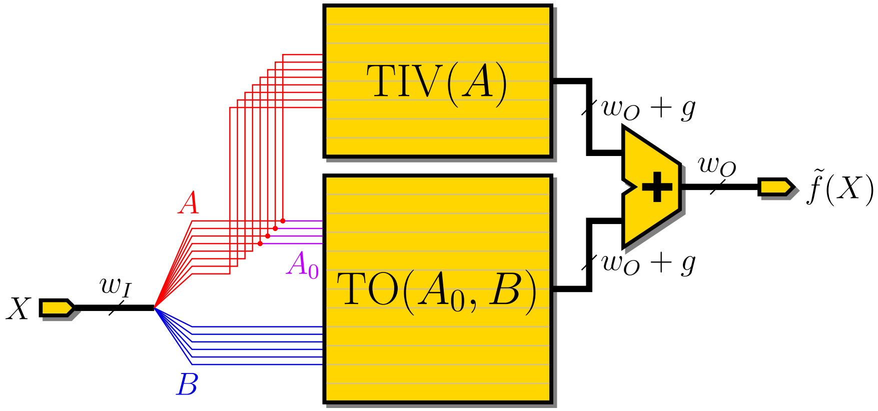
- Mencer, Boullis, Luk and Styles



  - smaller tables
  - longer critical path: 1 table look-up, 1 mult and 1 add

Order 1: bipartite table method

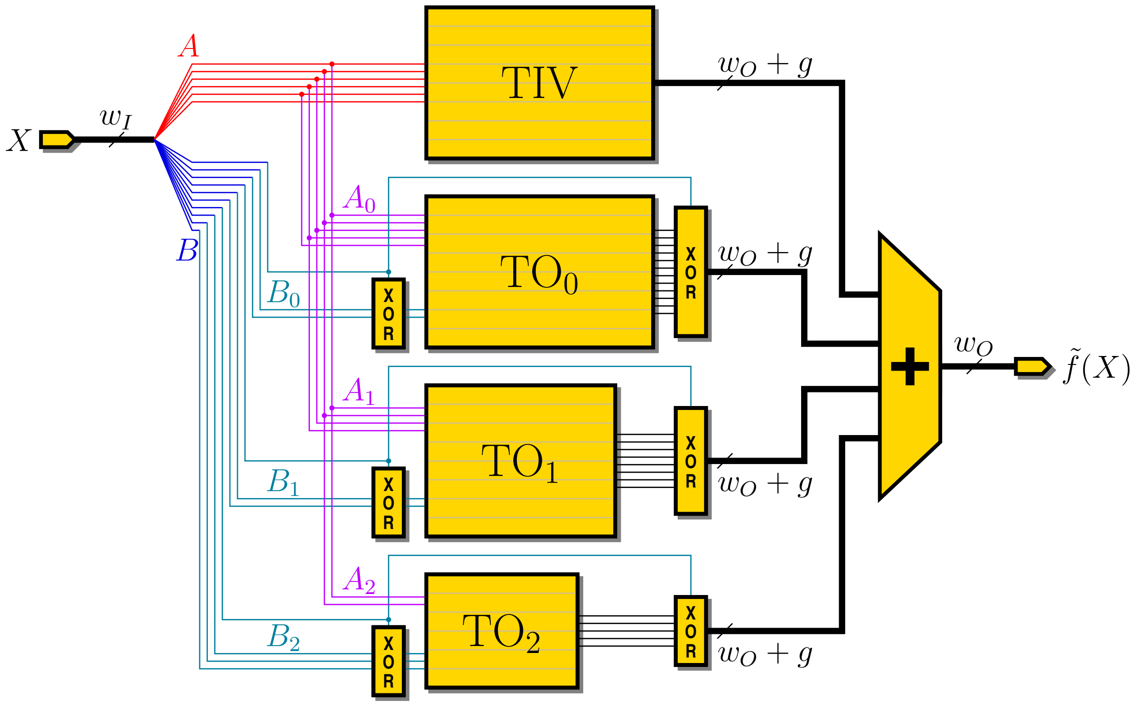
- Das Sarma and Matula, generalized by Schulte and Stine



  - shorter critical path: 1 table look-up and 1 add
  - slightly larger tables

Order 1: multipartite table method

- Schulte and Stine, Muller, de Dinechin and Tisserand: generalization and extension of the idea of the bipartite table method



- critical path: 2 XOR stages, 1 table look-up and $log_2(n)$ adds
- much smaller tables, but adder tree

Higher order methods

- Hörner evaluation
- interleaved memory interpolators: Lewis
- partial product arrays: Hassler and Takagi
- specialized squaring unit: Piñero, Bruguera and Muller
- simplified order 5 Taylor approximation: Defour, de Dinechin and Muller
- ...

Objectives

- simplify, generalize and extend Defour's method
- use ideas from order 1 methods (bipartite and multipartite table) for an order 2 approximation

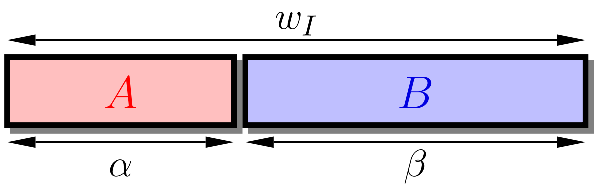
Objectives

- simplify, generalize and extend Defour's method
- use ideas from order 1 methods (bipartite and multipartite table) for an order 2 approximation
- maintain short critical path while keeping tables as small as possible
- use only small multipliers (Virtex-II)
- accurate error analysis for a fine tuning of the operators

The SMSO method

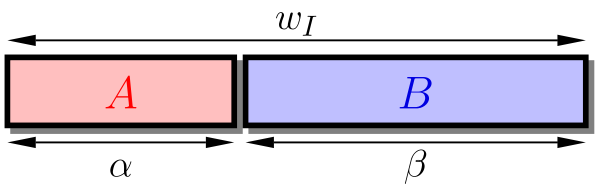
General idea: second order approximation

- input word decomposition: $X = A + 2^{-\alpha} B = .a_1 a_2 \cdots a_\alpha b_1 b_2 \cdots b_\beta$



General idea: second order approximation

- input word decomposition: $X = A + 2^{-\alpha}B = .a_1a_2\cdots a_\alpha b_1b_2\cdots b_\beta$

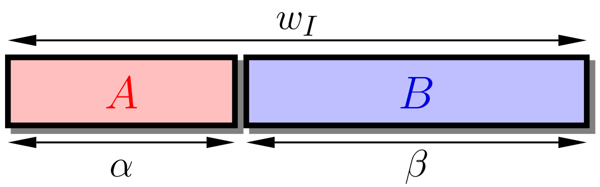


- order 2 approximation on each interval addressed by A:

$$\tilde{f}(X) \quad = \quad K_0(A) \quad + \quad K_1(A)\cdot 2^{-\alpha}B \quad + \quad K_2(A)\cdot 2^{-2\alpha}B^2$$

General idea: second order approximation

- input word decomposition: $X = A + 2^{-\alpha}B = .a_1a_2\cdots a_\alpha b_1b_2\cdots b_\beta$

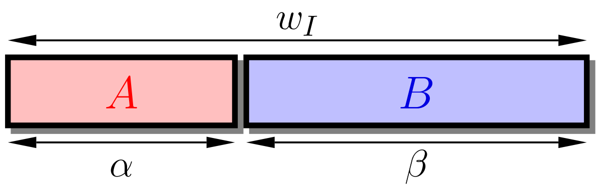


- order 2 approximation on each interval addressed by A:

$$\tilde{f}(X) \quad = \quad \underbrace{K_0(A)}_{\substack{\text{look-up table} \\ \mathrm{TIV}(A)}} \quad + \quad K_1(A)\cdot 2^{-\alpha}B \quad + \quad K_2(A)\cdot 2^{-2\alpha}B^2$$

General idea: second order approximation

- input word decomposition: $X = A + 2^{-\alpha}B = .a_1a_2\cdots a_\alpha b_1b_2\cdots b_\beta$

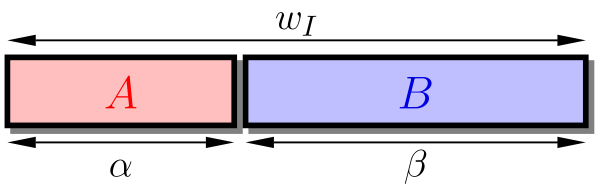


- order 2 approximation on each interval addressed by A:

$$\tilde{f}(X) = \underbrace{K_0(A)}_{\text{look-up table } \mathrm{TIV}(A)} + K_1(A)\cdot 2^{-\alpha}B + \underbrace{K_2(A)\cdot 2^{-2\alpha}B^2}_{\text{look-up table } \mathrm{TO}_2(A,B)}$$

General idea: second order approximation

- input word decomposition: $X = A + 2^{-\alpha} B = .a_1 a_2 \cdots a_\alpha b_1 b_2 \cdots b_\beta$

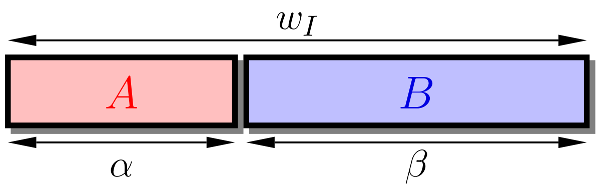


- order 2 approximation on each interval addressed by A:

$$\tilde{f}(X) = \underbrace{K_0(A)}_{\substack{\text{look-up table} \\ \mathrm{TIV}(A)}} + \underbrace{K_1(A) \cdot 2^{-\alpha} B}_{\substack{\text{multiplier?} \\ \text{look-up table?}}} + \underbrace{K_2(A) \cdot 2^{-2\alpha} B^2}_{\substack{\text{look-up table} \\ \mathrm{TO}_2(A, B)}}$$

General idea: second order approximation

- input word decomposition: $X = A + 2^{-\alpha}B = .a_1 a_2 \cdots a_\alpha b_1 b_2 \cdots b_\beta$

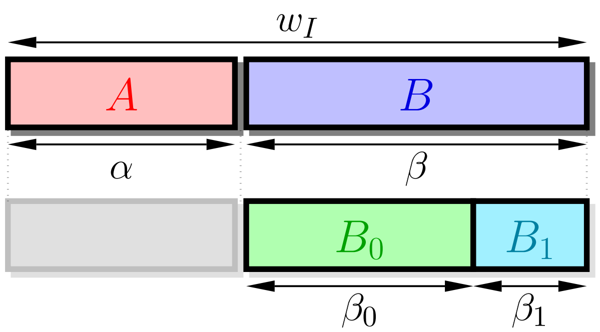


- order 2 approximation on each interval addressed by A:

$$\tilde{f}(X) = \underbrace{K_0(A)}_{\substack{\text{look-up table}\\ \mathrm{TIV}(A)}} + \underbrace{K_1(A) \cdot 2^{-\alpha}B}_{\substack{\text{multiplier?}\\ \text{look-up table?}\\ \text{both}}} + \underbrace{K_2(A) \cdot 2^{-2\alpha}B^2}_{\substack{\text{look-up table}\\ \mathrm{TO}_2(A, B)}}$$

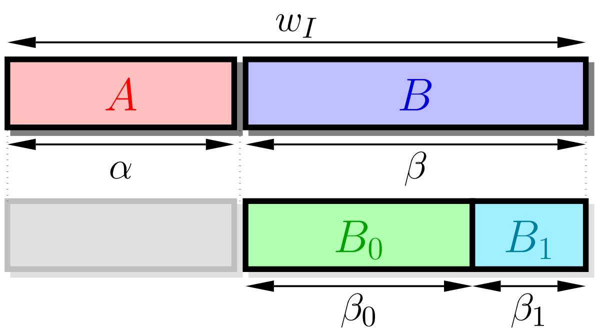
General idea: multiplication vs. look-up table tradeoff

- second decomposition: $B = B_0 + 2^{-\beta_0} B_1 = .b_1 b_2 \cdots b_{\beta_0} b_{\beta_0+1} \cdots b_{\beta}$:



General idea: multiplication vs. look-up table tradeoff

- second decomposition: $B = B_0 + 2^{-\beta_0} B_1 = .b_1 b_2 \cdots b_{\beta_0} b_{\beta_0+1} \cdots b_\beta$:

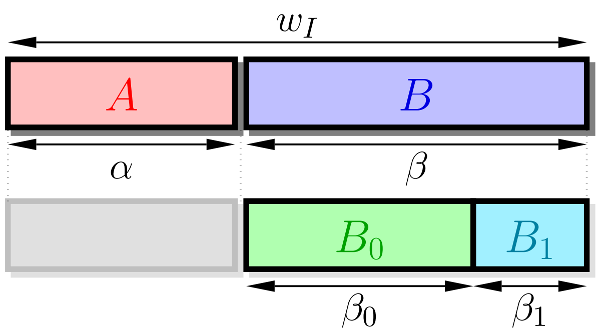


- we obtain:

$$K_1(A) \cdot 2^{-\alpha} B \quad = \quad K_1(A) \cdot 2^{-\alpha} B_0 \quad + \quad K_1(A) \cdot 2^{-\alpha-\beta_0} B_1$$

General idea: multiplication vs. look-up table tradeoff

- second decomposition: $B = B_0 + 2^{-\beta_0} B_1 = .b_1 b_2 \cdots b_{\beta_0} b_{\beta_0+1} \cdots b_\beta$:



- we obtain:

$$K_1(A) \cdot 2^{-\alpha} B \quad = \quad \underbrace{K_1(A) \cdot 2^{-\alpha} B_0}_{\substack{\text{multiplier} \\ \mathrm{TS}(A) \times B_0}} \quad + \quad \underbrace{K_1(A) \cdot 2^{-\alpha-\beta_0} B_1}_{\substack{\text{look-up table} \\ \mathrm{TO}_1(A, B_1)}}$$

General idea

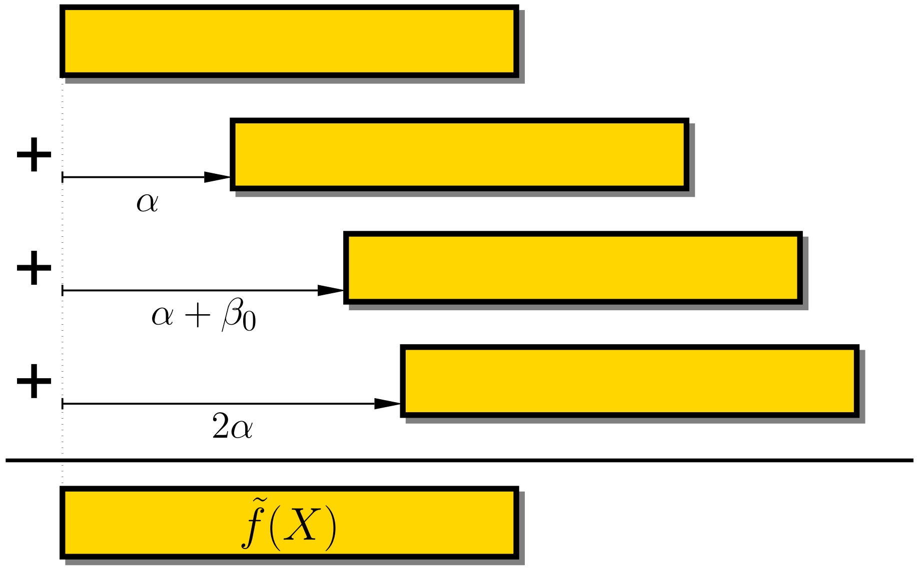
- we have 4 tables:
  - Table of Initial Values: $\mathrm{TIV}(A) = K_0(A)$
  - Table of Slopes: $\mathrm{TS}(A) = 2^{-\alpha} \cdot K_1(A)$
  - Table of Offsets (order 1): $\mathrm{TO}_1(A, B_1) = 2^{-\alpha-\beta_0} \cdot K_1(A) \cdot B_1$
  - Table of Offsets (order 2): $\mathrm{TO}_2(A, B) = 2^{-2\alpha} \cdot K_2(A) \cdot B^2$

General idea

- we have 4 tables:
  - Table of Initial Values: $\mathrm{TIV}(A) = K_0(A)$
  - Table of Slopes: $\mathrm{TS}(A) = 2^{-\alpha} \cdot K_1(A)$
  - Table of Offsets (order 1): $\mathrm{TO}_1(A, B_1) = 2^{-\alpha-\beta_0} \cdot K_1(A) \cdot B_1$
  - Table of Offsets (order 2): $\mathrm{TO}_2(A, B) = 2^{-2\alpha} \cdot K_2(A) \cdot B^2$
- we obtain:

$$\tilde{f}(X) = \mathrm{TIV}(A) + \mathrm{TS}(A) \times B_0 + \mathrm{TO}_1(A, B_1) + \mathrm{TO}_2(A, B)$$

General idea: degrading accuracy



$$\mathrm{TIV}(A) = K_0(A)$$

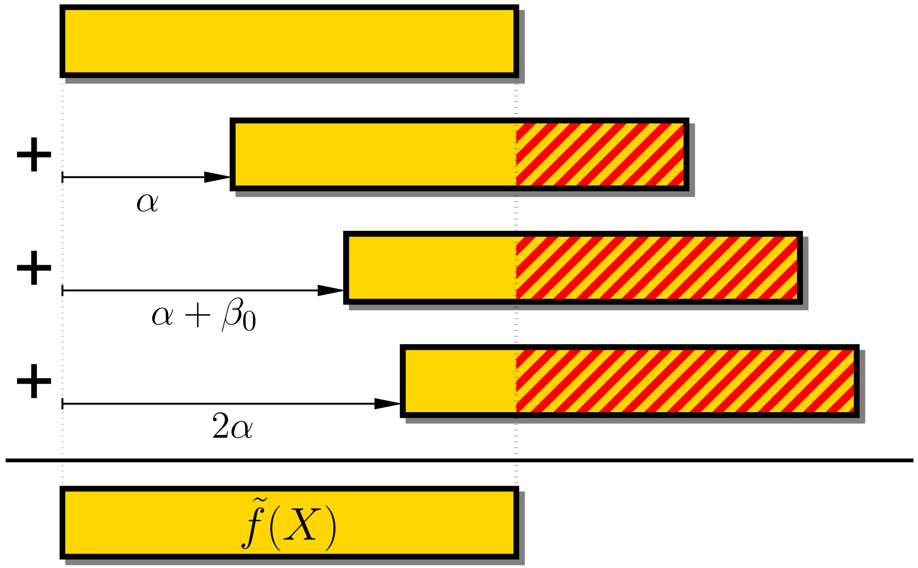
$$\mathrm{TS}(A) \times B_0 = 2^{-\alpha} \cdot K_1(A) \times B_0$$

$$\mathrm{TO}_1(A, B_1) = 2^{-\alpha-\beta_0} \cdot K_1(A) \cdot B_1$$

$$\mathrm{TO}_2(A, B) = 2^{-2\alpha} \cdot K_2(A) \cdot B^2$$

- some of the terms are more accurate than others

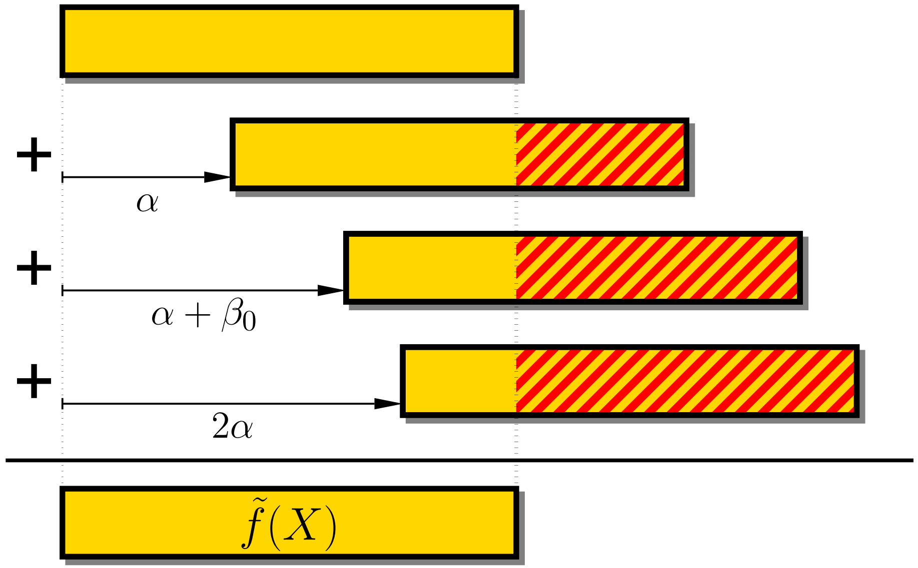
General idea: degrading accuracy



$$\begin{aligned}
\mathrm{TIV}(A) &= K_0(A) \\
\mathrm{TS}(A) \times B_0 &= 2^{-\alpha} \cdot K_1(A) \times B_0 \\
\mathrm{TO}_1(A, B_1) &= 2^{-\alpha-\beta_0} \cdot K_1(A) \cdot B_1 \\
\mathrm{TO}_2(A, B) &= 2^{-2\alpha} \cdot K_2(A) \cdot B^2
\end{aligned}$$

- some of the terms are more accurate than others

General idea: degrading accuracy

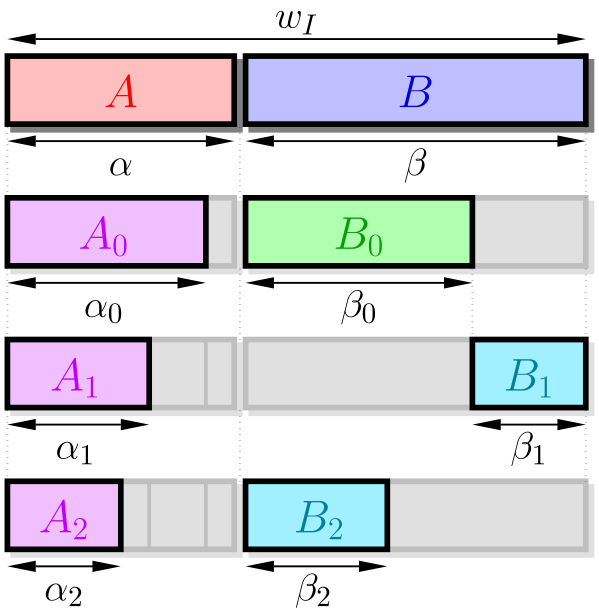


$$\begin{aligned}
\mathrm{TIV}(A) &= K_0(A) \\
\mathrm{TS}(A) \times B_0 &= 2^{-\alpha} \cdot K_1(A) \times B_0 \\
\mathrm{TO}_1(A, B_1) &= 2^{-\alpha-\beta_0} \cdot K_1(A) \cdot B_1 \\
\mathrm{TO}_2(A, B) &= 2^{-2\alpha} \cdot K_2(A) \cdot B^2
\end{aligned}$$

- some of the terms are more accurate than others
- we can save area by addressing the more accurate tables with less bits

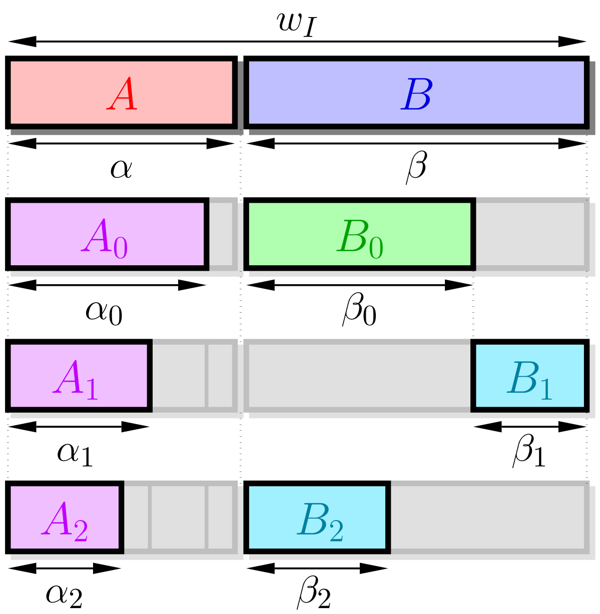
General idea: degrading accuracy

- input word decomposition:



General idea: degrading accuracy

- input word decomposition:

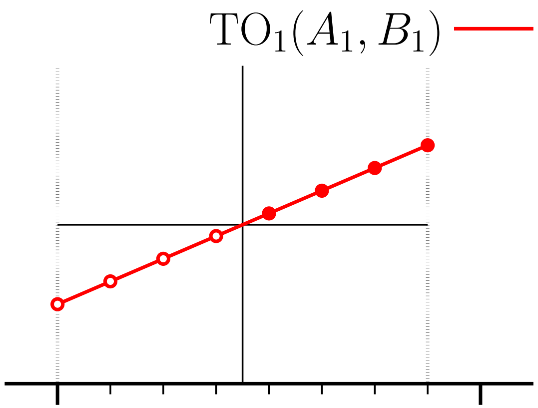


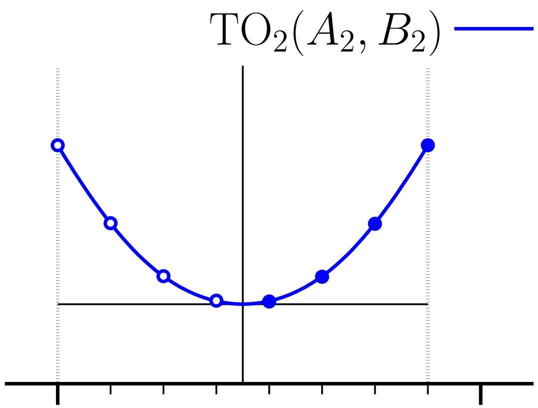
- we obtain the final SMSO formula:

$$\tilde{f}(X) = \mathrm{TIV}(A) + \mathrm{TS}(A_0) \times B_0 + \mathrm{TO}_1(A_1, B_1) + \mathrm{TO}_2(A_2, B_2)$$

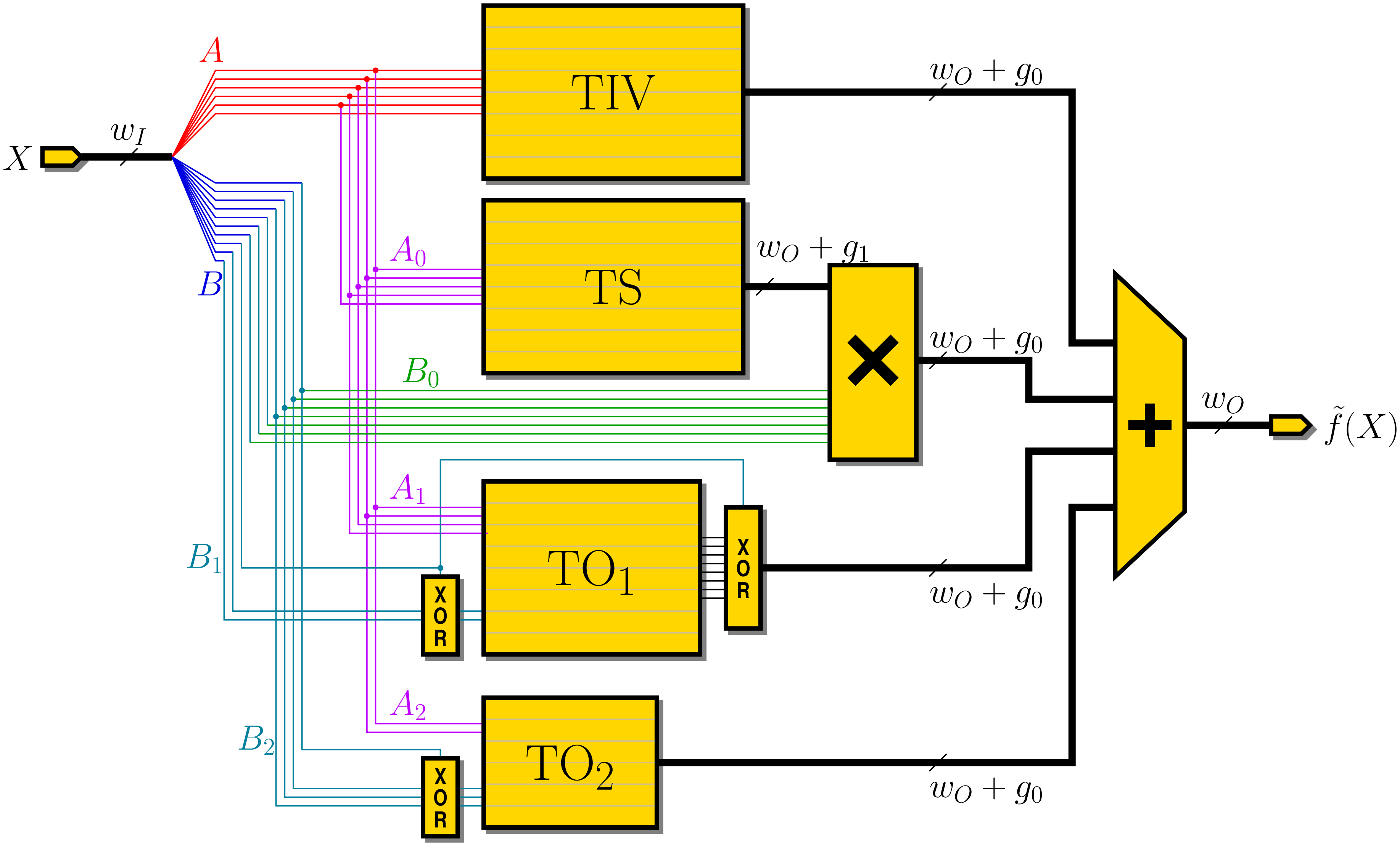
General idea: exploiting symmetry

- both TO_1 and TO_2 have symmetry property:
  - $\mathrm{TO}_1(A_1, -B_1) = -\mathrm{TO}_1(A_1, B_1)$
  - $\mathrm{TO}_2(A_2, -B_2) = \mathrm{TO}_2(A_2, B_2)$





Architecture



Optimization

- Context
- The SMSO method
- Optimization
  - Rounding considerations
  - Parameter space exploration
- Results
- Conclusion

Rounding considerations

- we want to achieve faithful rounding: $\left|\tilde{f}(X) - f(X)\right| < 2^{-w_O}$

Rounding considerations

- we want to achieve faithful rounding: $\left|\tilde{f}(X) - f(X)\right| < 2^{-w_O}$
- the SMSO operator entails several errors:
  - polynomial approximation: ϵ_{poly}
  - degrading table accuracy: ϵ_{tab}
  - rounding table values: $\epsilon_{\text{rt}} < 4 \times 2^{-w_O - g_0 - 1} + 2^{-w_O - g_1 - 1}$
  - final rounding: $\epsilon_{\text{rf}} < 2^{-w_O - 1} \cdot (1 - 2^{-g_0})$

Rounding considerations

- error constraint:

$$\epsilon_{\text{poly}} + \epsilon_{\text{tab}} + \epsilon_{\text{rt}} + \epsilon_{\text{rf}} < 2^{-w_O}$$

Rounding considerations

- error constraint:

$$\epsilon_{\text{poly}} \quad + \quad \epsilon_{\text{tab}} \quad + \quad \epsilon_{\text{rt}} \quad + \quad \epsilon_{\text{rf}} \quad < \quad 2^{-w_O}$$

- constraint on g_0 and g_1:

$$\epsilon_{\text{poly}} + \epsilon_{\text{tab}} + 4 \cdot 2^{-w_O - g_0 - 1} + 2^{-w_O - g_1 - 1} + 2^{-w_O - 1} \cdot (1 - 2^{-g_0}) < 2^{-w_O}$$

Rounding considerations

- error constraint:

$$\epsilon_{\text{poly}} + \epsilon_{\text{tab}} + \epsilon_{\text{rt}} + \epsilon_{\text{rf}} < 2^{-w_O}$$

- constraint on g_0 and g_1:

$$\epsilon_{\text{poly}} + \epsilon_{\text{tab}} + 4 \cdot 2^{-w_O - g_0 - 1} + 2^{-w_O - g_1 - 1} + 2^{-w_O - 1} \cdot (1 - 2^{-g_0}) < 2^{-w_O}$$

- assuming $g_0 = g_1$, we compute the bound:

$$g_0 > \log_2 \left(\frac{4}{2^{-w_O - 1} - \epsilon_{\text{poly}} - \epsilon_{\text{tab}}} \right) - w_O - 1$$

Parameter space exploration

- lots of parameters: α, β, α_0, β_0, α_1, β_1, α_2, β_2, g_0, g_1
- to find the best decomposition: parameter space exploration

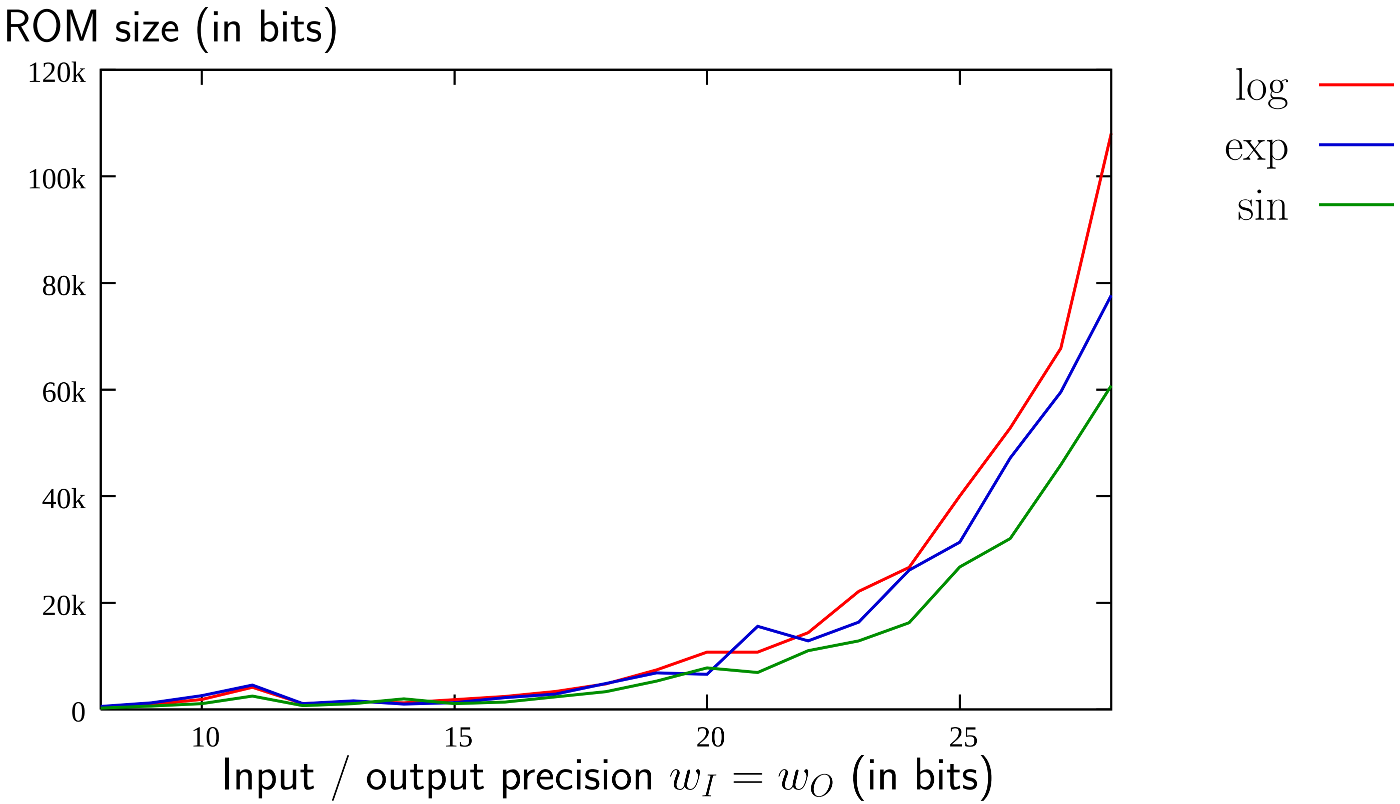
Parameter space exploration

- lots of parameters: α, β, α_0, β_0, α_1, β_1, α_2, β_2, g_0, g_1
- to find the best decomposition: parameter space exploration
- the parameter space is huge, so we need a heuristic:
  - find all the acceptable decompositions such that $\epsilon_{\text{poly}} + \epsilon_{\text{tab}} < 2^{-w_O-1}$
  - for each candidate:
    - compute the bounds for g_0 and g_1
    - fill the tables
    - compute the width of the signals
    - apply a user-defined score function (area, latency, multiplier size, ...)
  - the score determines the best decomposition
  - trial-and-error method to decrease g_0 and g_1

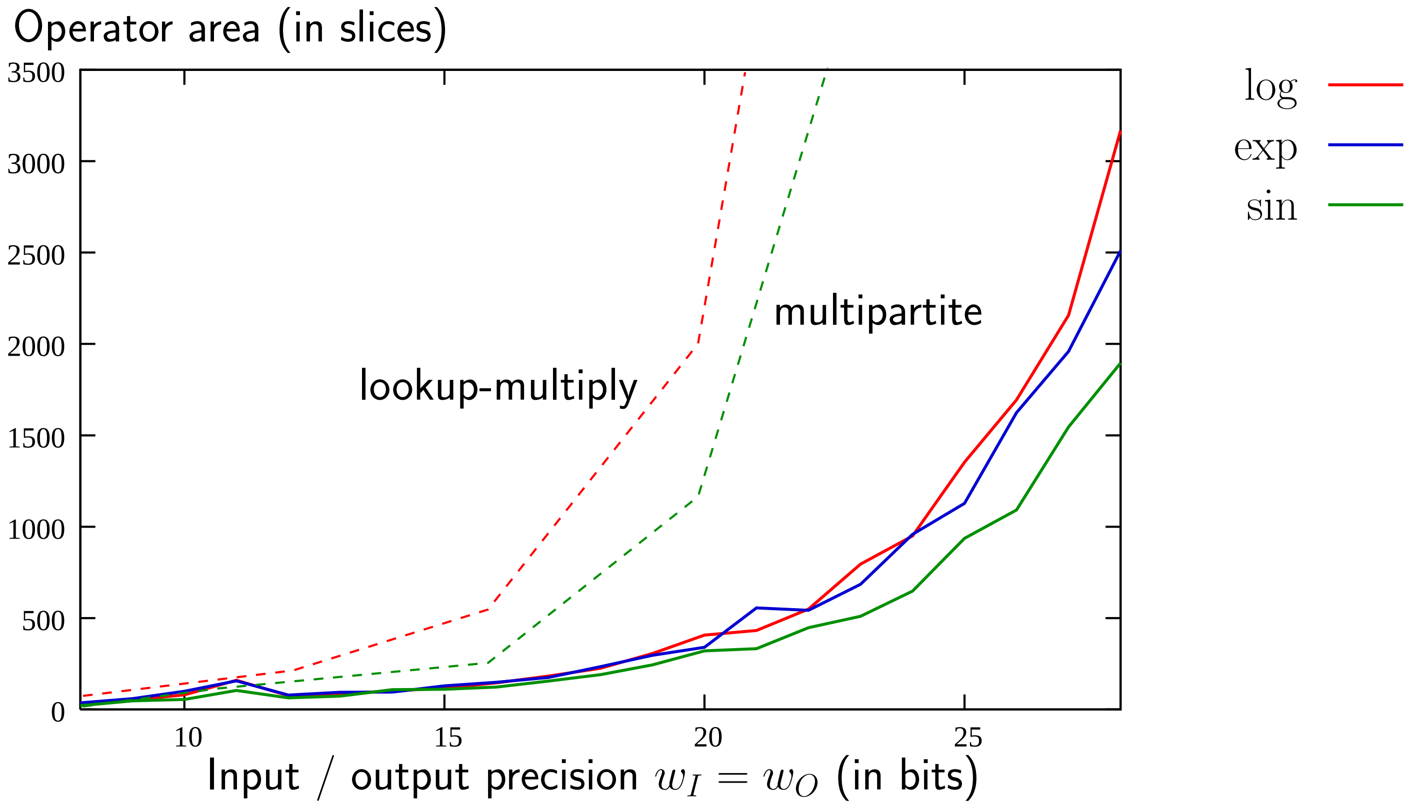
Results

- Context
- The SMSO method
- Optimization
- **Results**
- Conclusion

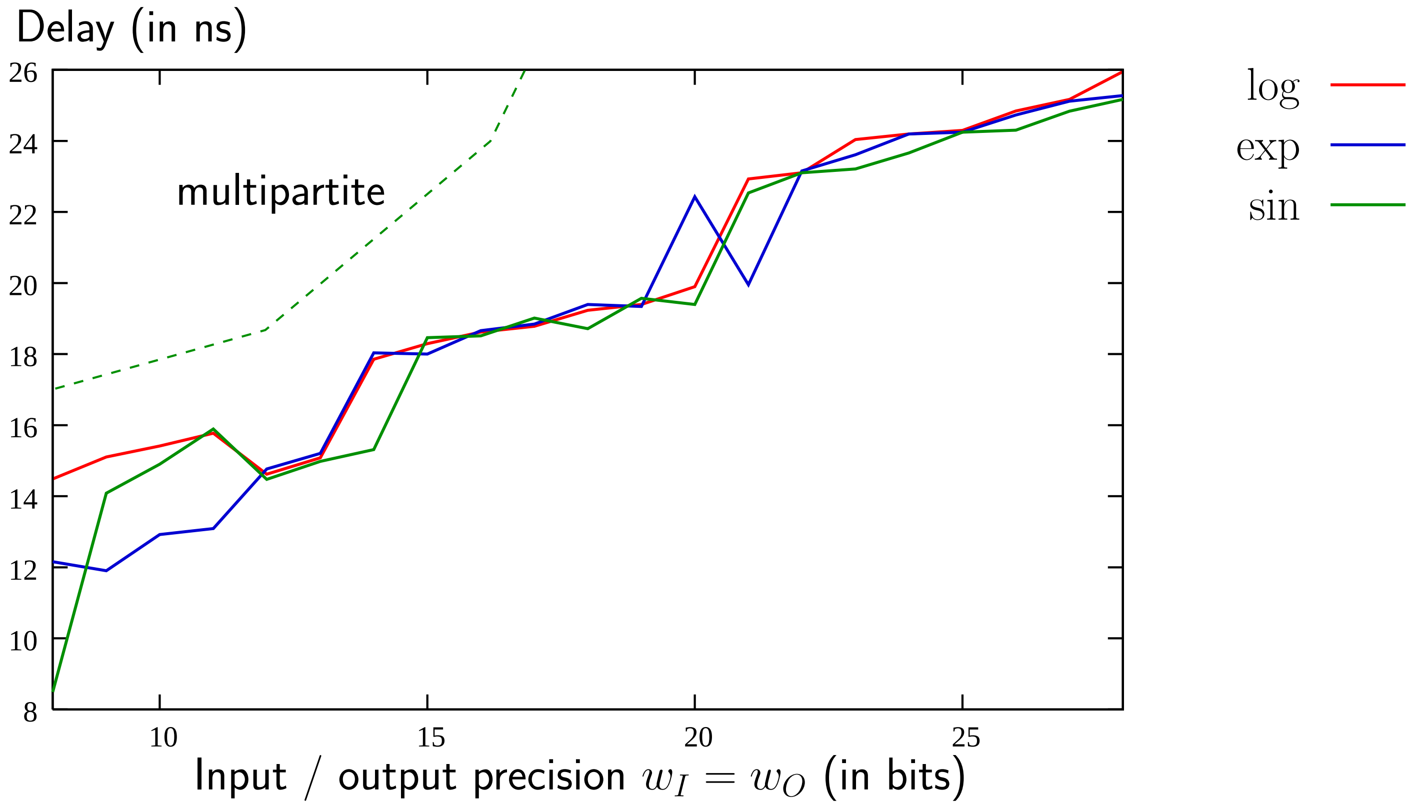
Results: ROM size



Results: operator area



Results: operator latency



Results: using Virtex-II small multipliers

Function		log		
Precision ($w_I = w_O$)		16 bits	20 bits	24 bits
Multiplier bit size		8×11	8×14	14×17
not using block multipliers	area (slices)	148	419	981
	delay (ns)	21	22	27
using block multipliers	area (slices)	102	362	855
	delay (ns)	18	21	25

Function		sin		
Precision ($w_I = w_O$) (bits)		16	20	24
Multiplier bit size		8×13	8×14	14×19
not using block multipliers	area (slices)	124	332	671
	delay (ns)	19	21	25
using block multipliers	area (slices)	71	275	540
	delay (ns)	19	21	25

Conclusion

- Context
- The SMSO method
- Optimization
- Results
- Conclusion

Contribution

- a novel function approximation method:
  - second order: smaller tables
  - only one small multiplier: shorter critical path, and can benefit from recent FPGA technologies (Virtex-II)
- accurate approximation and rounding error analysis
- automated exploration of the parameter space according to user-specified criteria

Future work

- split the TO_is on several tables, as in the multipartite table method
- work on table compression techniques
- extend this method to higher order approximations

Thank you for your attention

Thank you for your attention

Questions?