Wordlength Optimization for Custom Floating-point Systems

DASIP 2024

Authors :
Quentin Milot, Mickaël Dardaillon,
Justine Bonnot and Daniel Menard

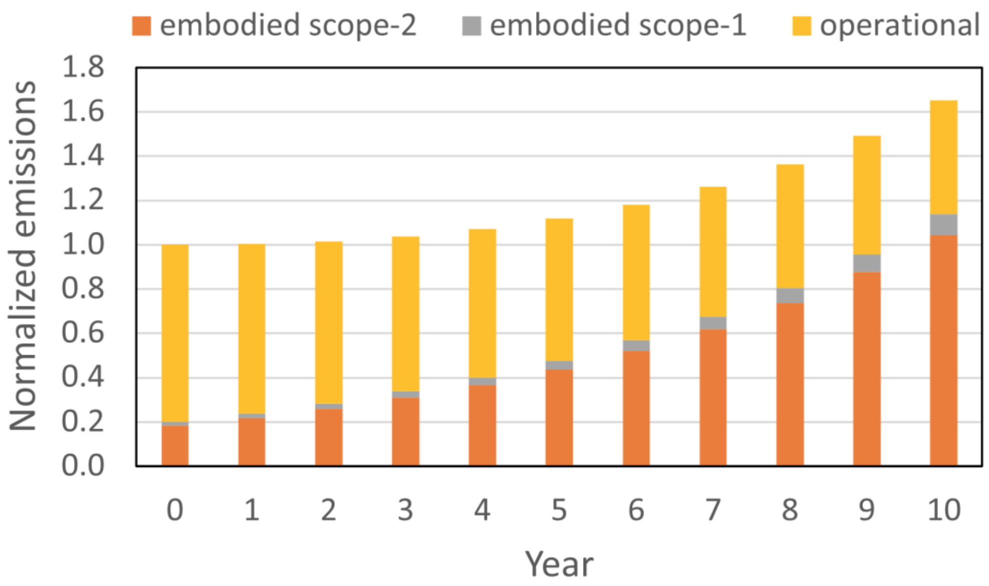
cnrs
CentraleSupélec
INSA RENNES
Nantes Université
Université de Rennes

26/01/2024

www.ietr.fr

Introduction

Increase and impact of computer systems



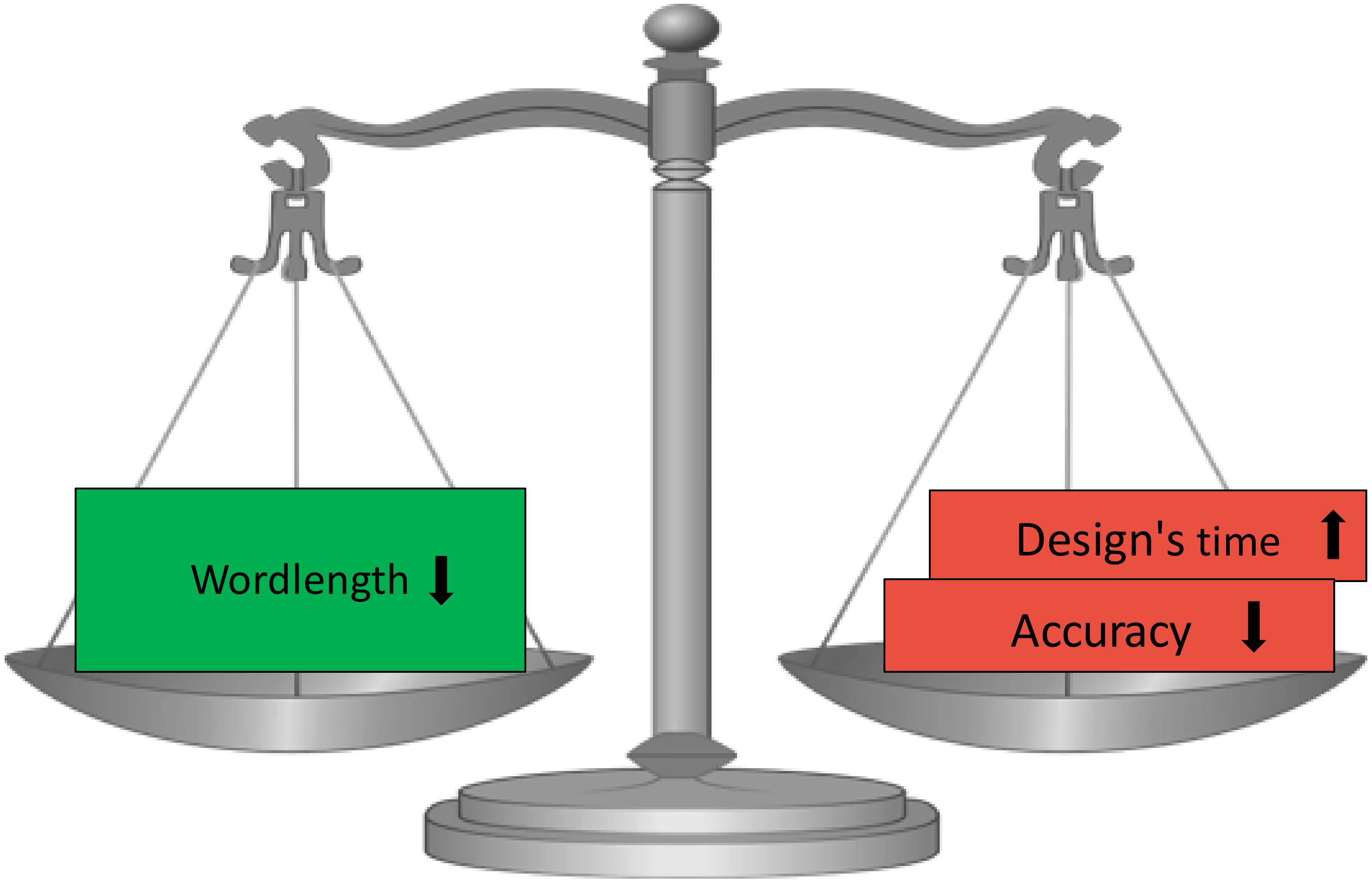
Extracted from [1].
A scenario of normalized emissions projections for computer systems over the year

- Environmental impact of computer systems increase and will increase over the years
- Impact come from production, deployment and usage
- Proposed solution :
  - Reduce die size
  - Reduce operational energy

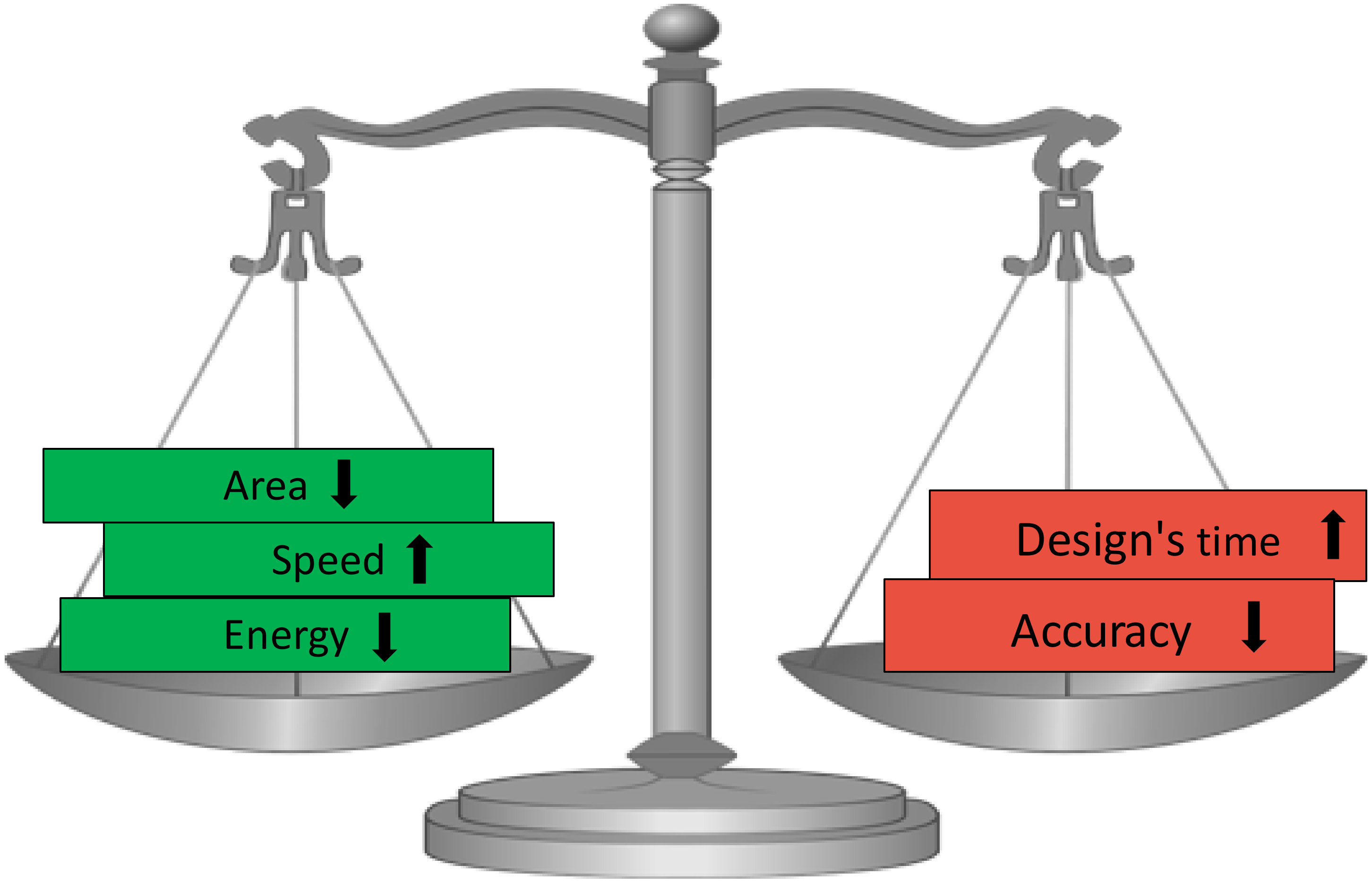
=> A lead : Use of approximate computing

[1]"Kaya for Computer Architects: Toward Sustainable Computer Systems", Lieven Eeckhout, IEEE Micro, Volume 43, 2023

A lead : Approximate computing



A lead : Approximate computing

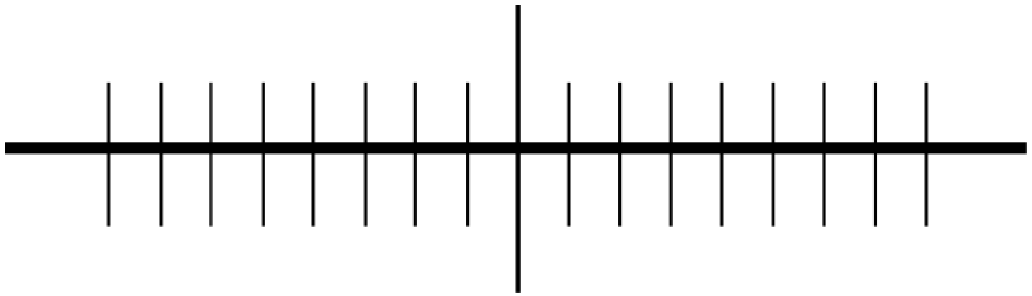


Fixed point

$$x * 2^{-E} = (-1)^{S} * Int$$

0	0	1	0	0	1	0	1	1	0	1	0	0

- Exponent implicit in the code

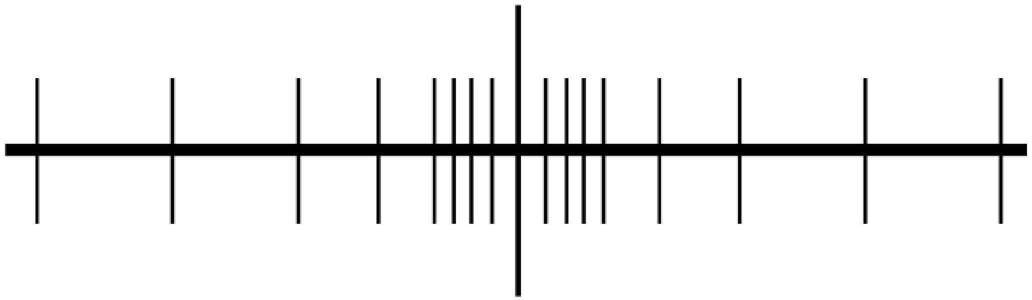


Floating point

$$x = (-1)^{S} * (1 + M) * 2^{E-\Delta}$$

0	0	1	0	0	1	0	1	1	0	1	0	0

- Exponent explicit in the coded value



Floating-point : a new contender ?

Operation		Picojoules per operation	
		45nm	7nm
+	Int 32	0.1	0.03
	IEEE FP 32	0.9	0.38
X	Int 32	3.1	1.48
	IEEE FP 32	3.7	1.31
SRAM	8KB	10	7.5
	32KB	20	8.5
	1MB	100	14

Data extracted from [2]
Energy per operation for a 45nm and 7 nm systems

- Fixed point was first choice thanks to low operational energy
- Floating point operational energy reduces over the year (~ - 2X)
- Memory usage is the higher energy requirement now
- Floating point can lead to lower memory usage

=> Custom floating-point systems became an option for embedded systems

[2]"Ten Lessons From Three Generations Shaped Google's TPUv4i", Norman P. Jouppi et al, ISCA 2021

Challenge :

Determine a generic **automatic** wordlength optimization method for a given accuracy

Outline :

I. Introduction

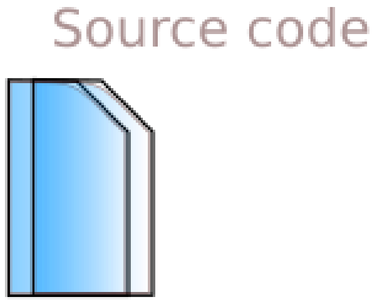
II. Optimization flow

III. Proposed strategies

IV. Experiments

Outline :

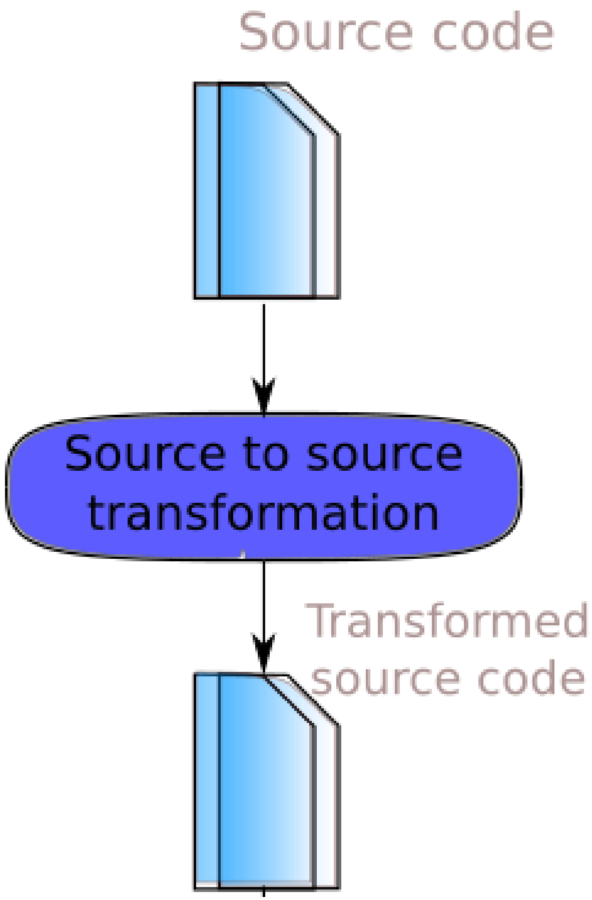
Simple strategy



User source code :

- Write in C/C++
- Any algorithm

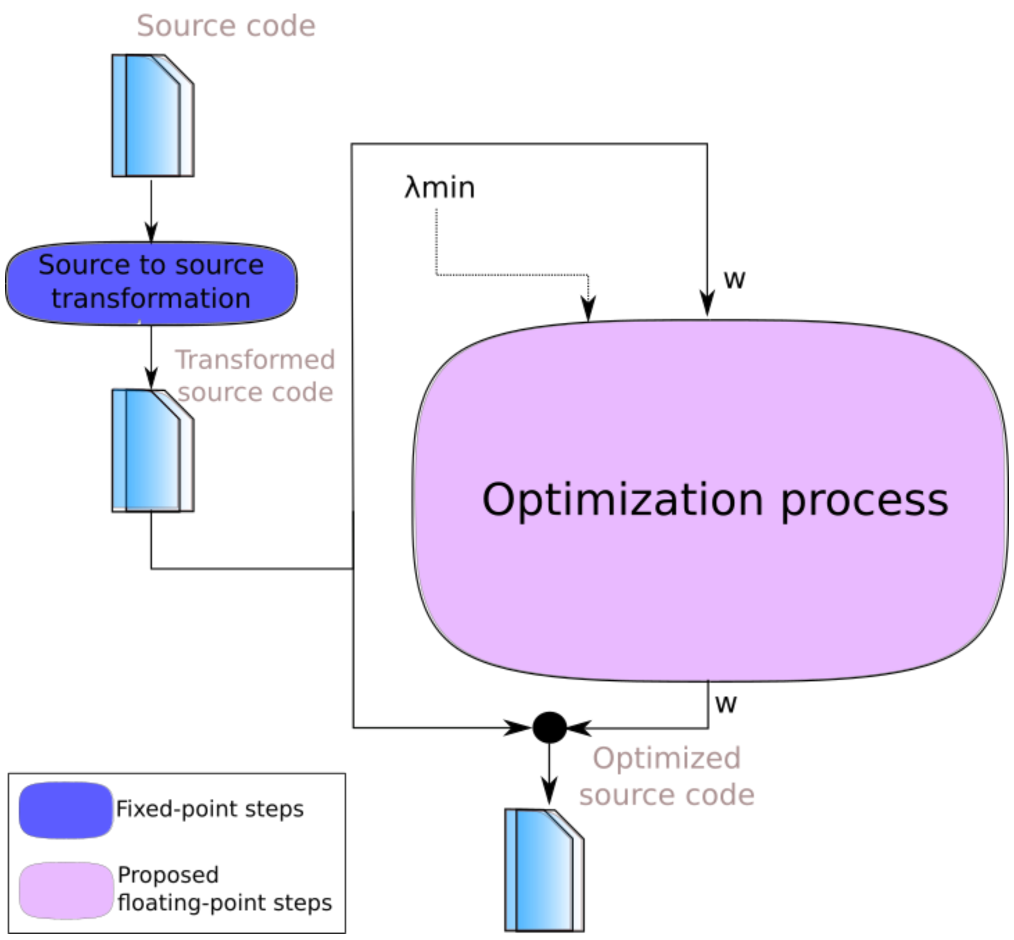
Simple strategy



Source to source transformation :

- Write in C/C++
- Any algorithm

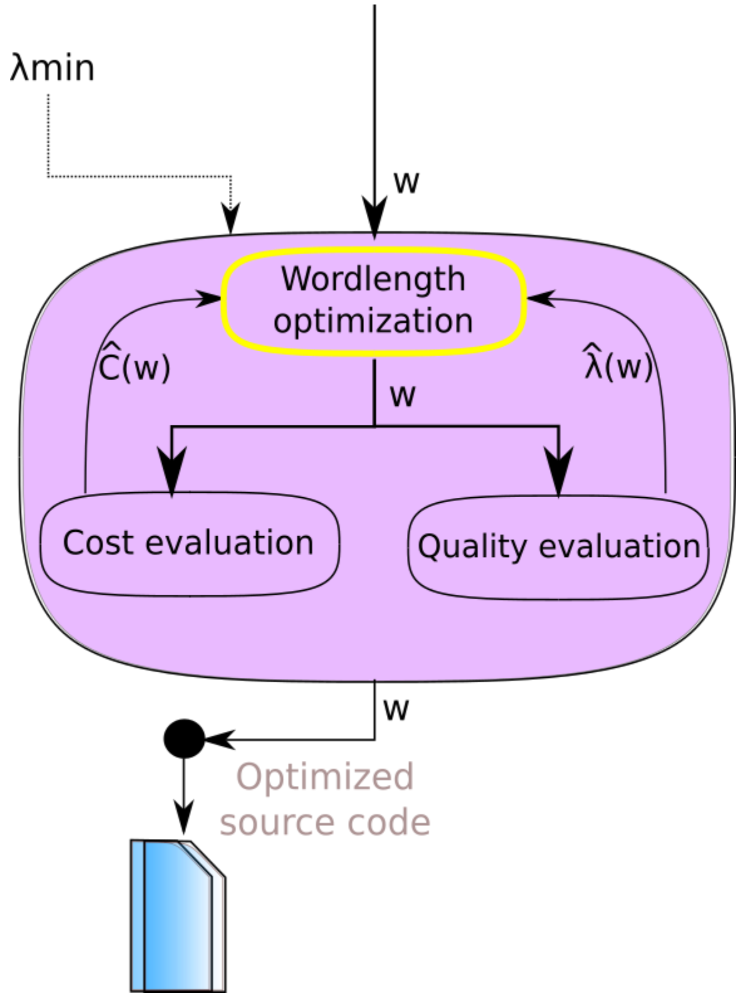
Simple strategy



Optimization : $\overbrace{(W_{E_{max}}, W_{E_{max}}, ..., W_{E_{max}}}^{\text{Variable's exponent}}, \overbrace{W_{M_{max}}, W_{M_{max}}, ..., W_{M_{max}})}^{\text{Variable's mantissa}}$

- Input : $(W_{E_{max}}, W_{E_{max}}, ..., W_{E_{max}}, W_{M_{max}}, W_{M_{max}}, ..., W_{M_{max}})$
- λ_{min} : The user minimal quality
- Problem : $\min_{\hat{\lambda}(W) \geq \lambda_{min}} \hat{C}(W)$

Wordlength Optimization



MIN+1 algorithm:

- Greedy heuristic algorithm
- Two steps algorithm :

  1) Find the minimal value for each dimension that satisfy the criterion

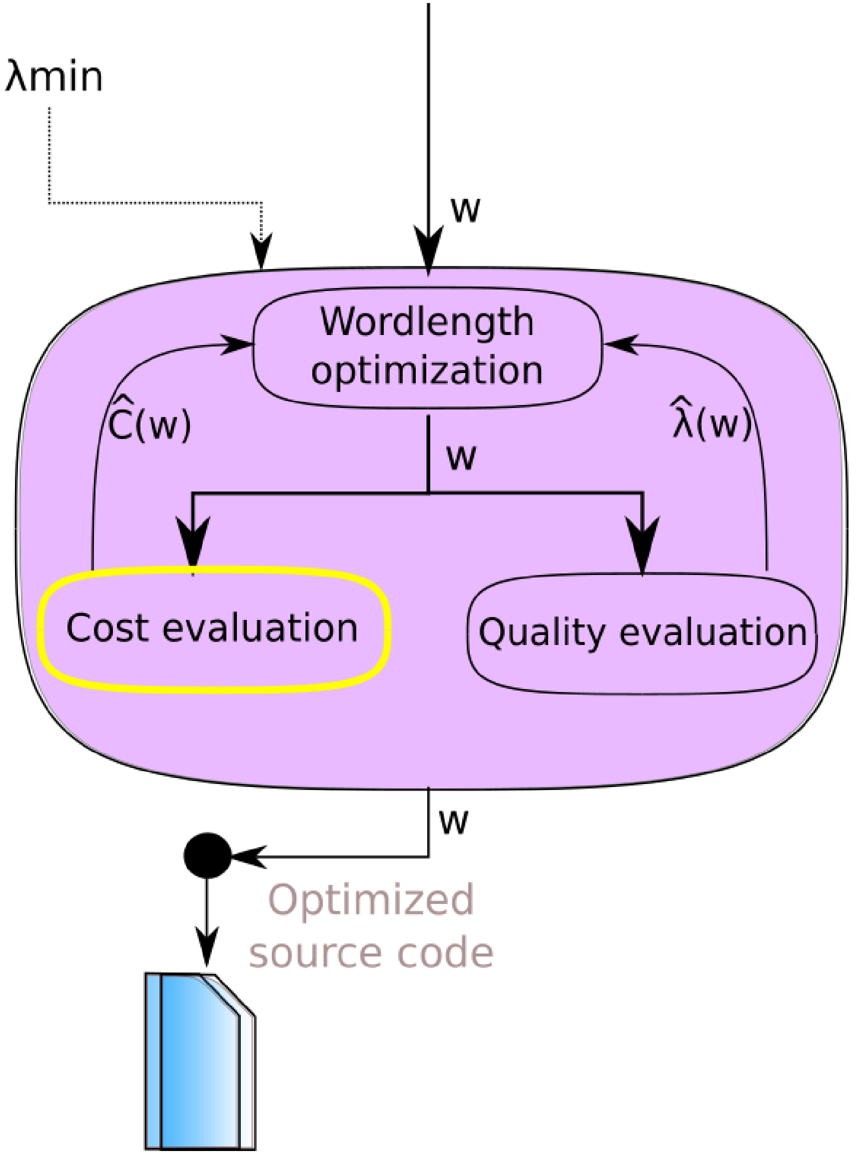
  - Set all value to the minimal found

  2) Increment the dimension with the best gradient until the criterion is met

Cost evaluation

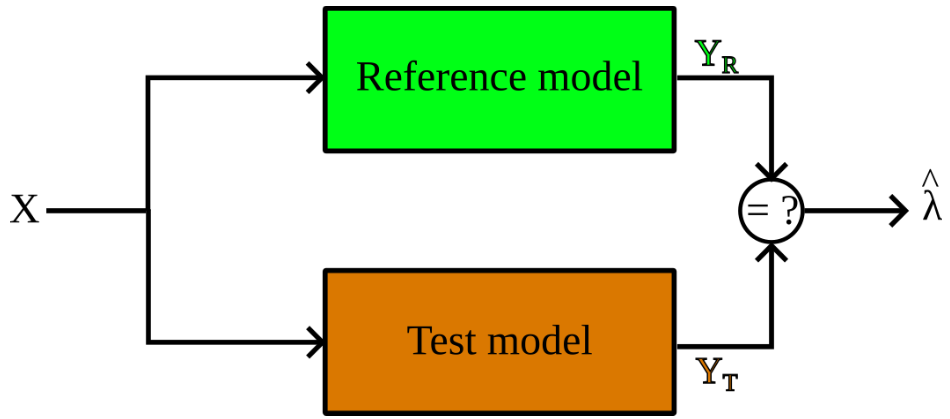
Cost evaluation

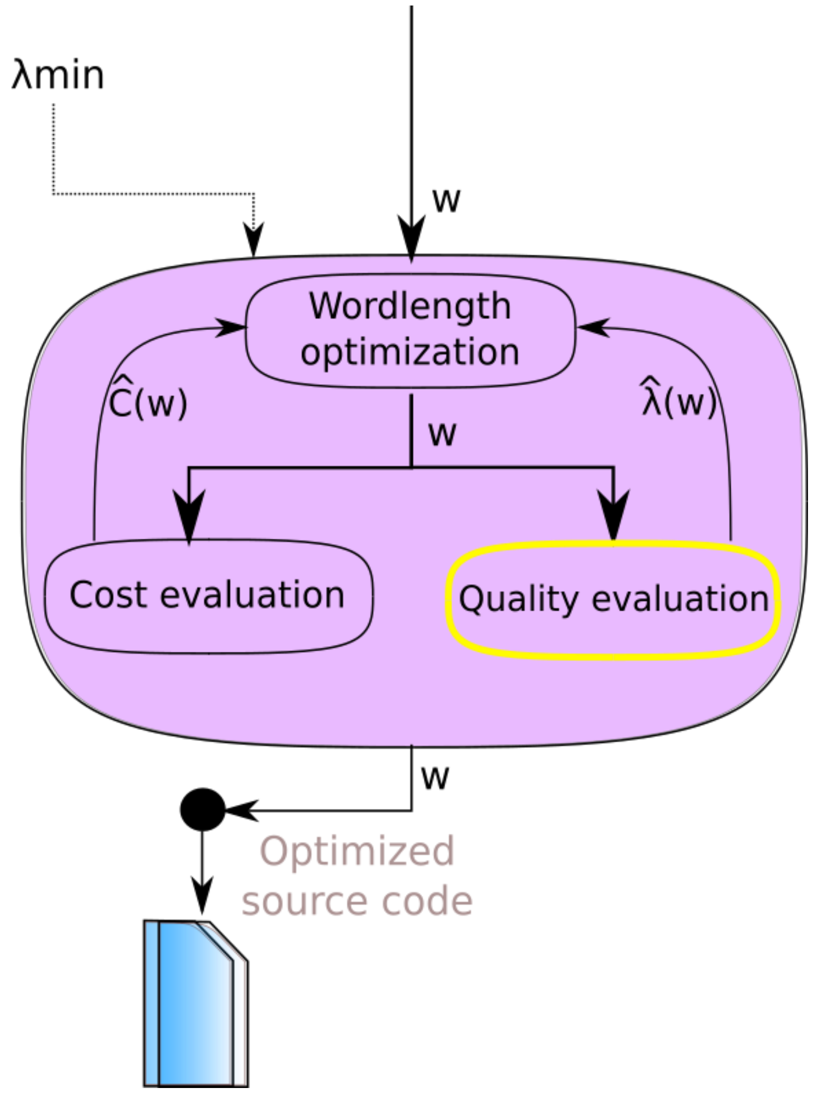
Requirement :
- Types of memory
- Energy for a given operation and wordlength
- Architectures



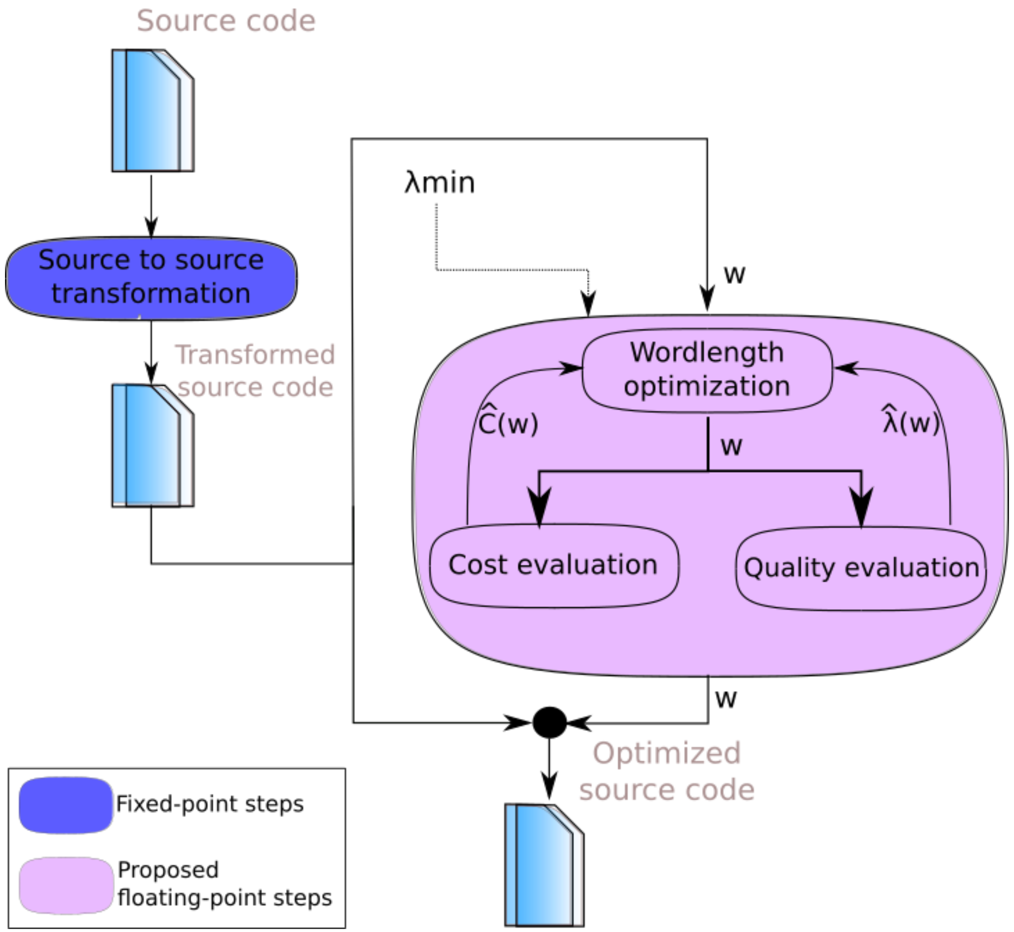
Quality evaluation

Quality evaluation





Simple strategy



Optimization process :

- Input : $(W_{E_{max}}, W_{E_{max}}, ..., W_{E_{max}}, W_{M_{max}}, W_{M_{max}}, ..., W_{M_{max}})$
- λ_{min} : The user minimal quality

Optimization time

$$T_{Opt} = N_c * T_{Sim}$$

Optimization time

Time for a quality estimation

Number of tested configuration

Depends on :

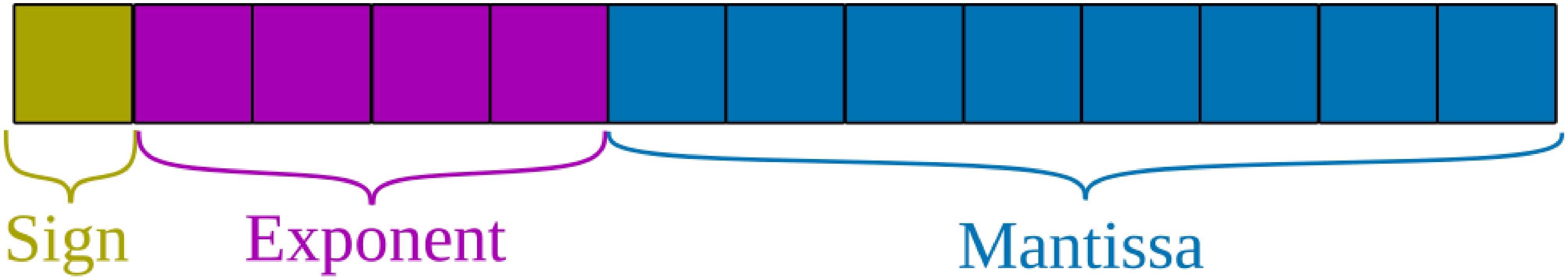
- Possibilities in space's exploration : n_p
- Number of variables : N_v

Challenge :

Determine a generic **automatic** wordlength optimization method for a given accuracy in a **reasonable amount of time**

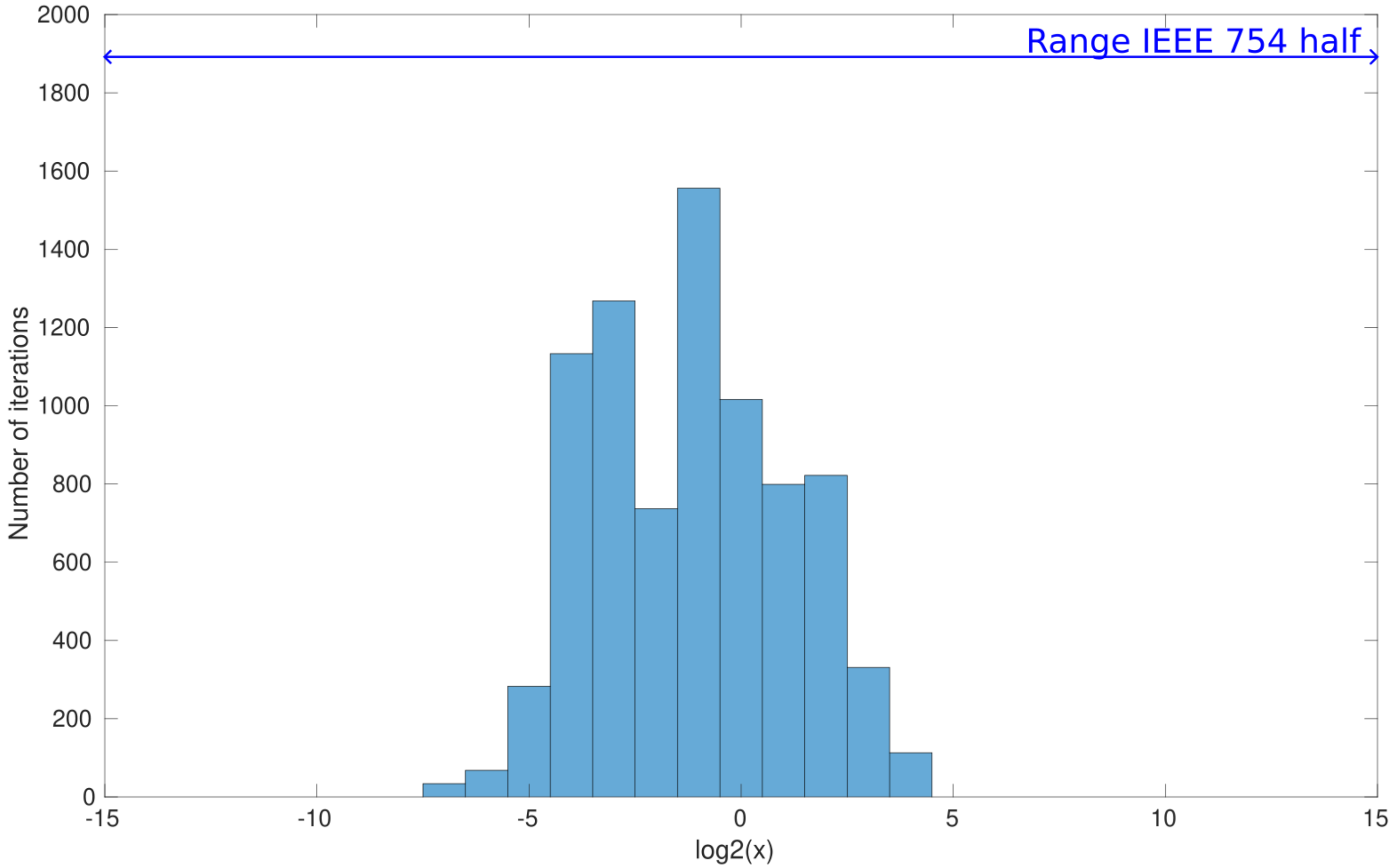
Outline :

Floating-point description

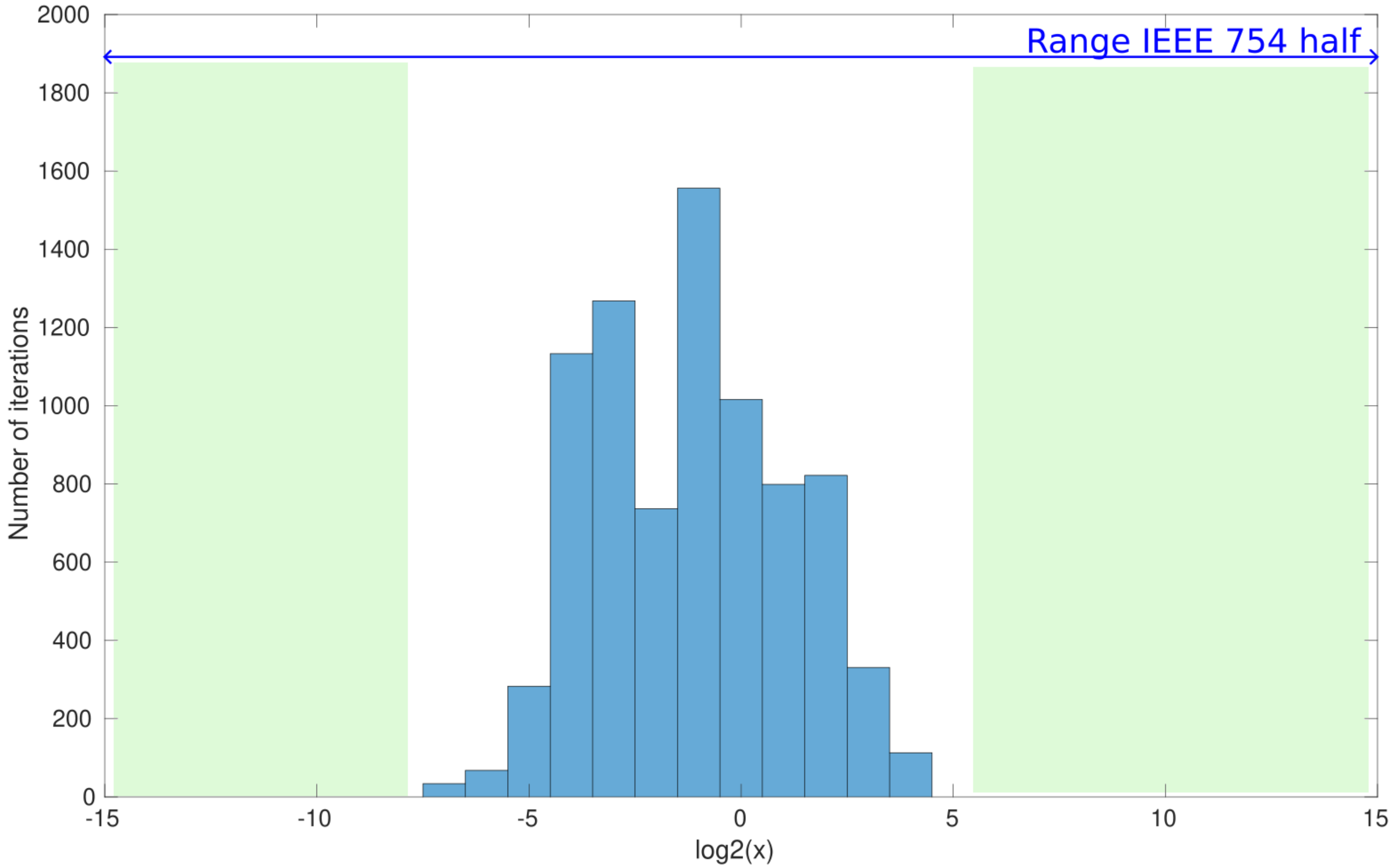


$$x = (-1)^{S} * (1 + M) * 2^{E-\Delta}$$

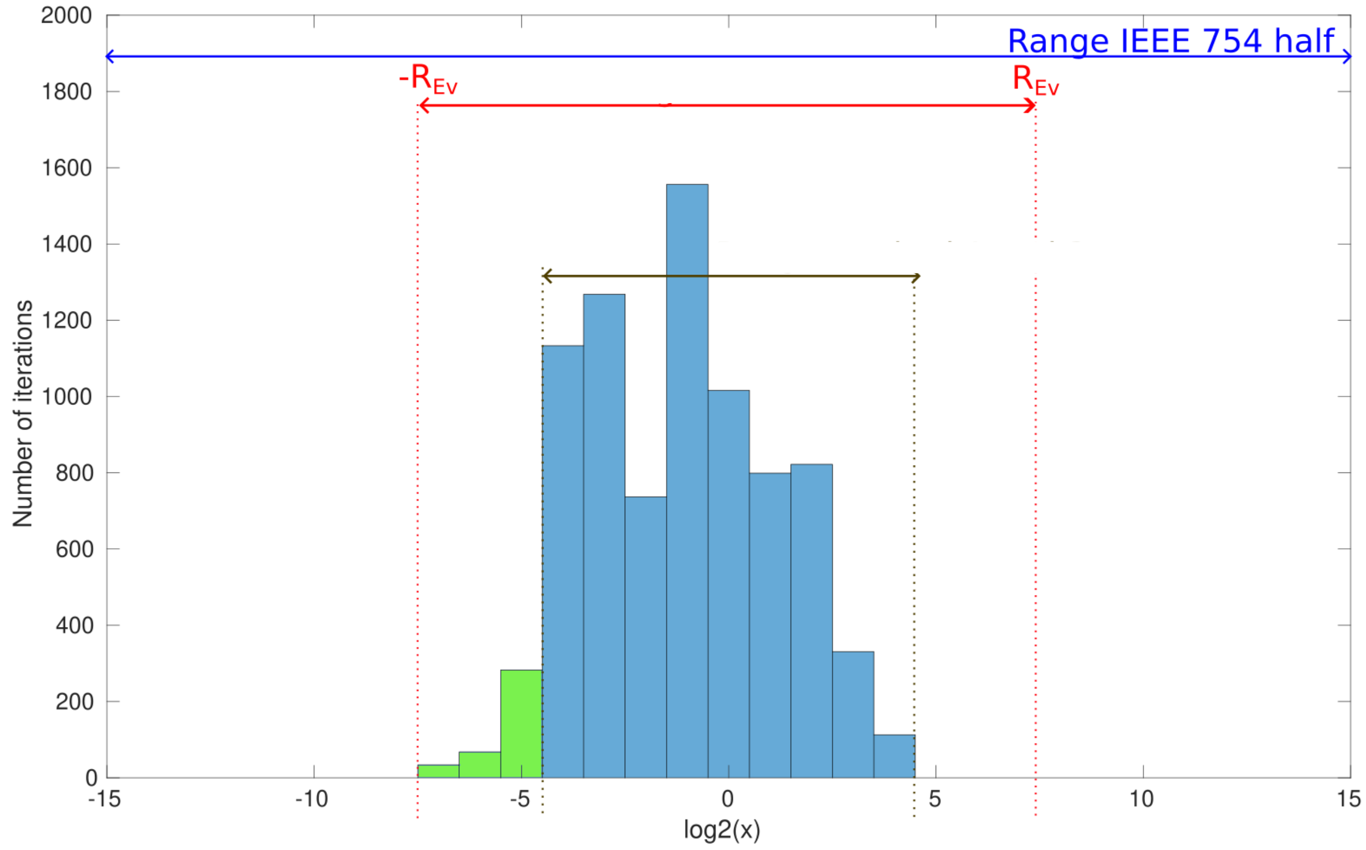
Exponent bits required



Exponent bits required



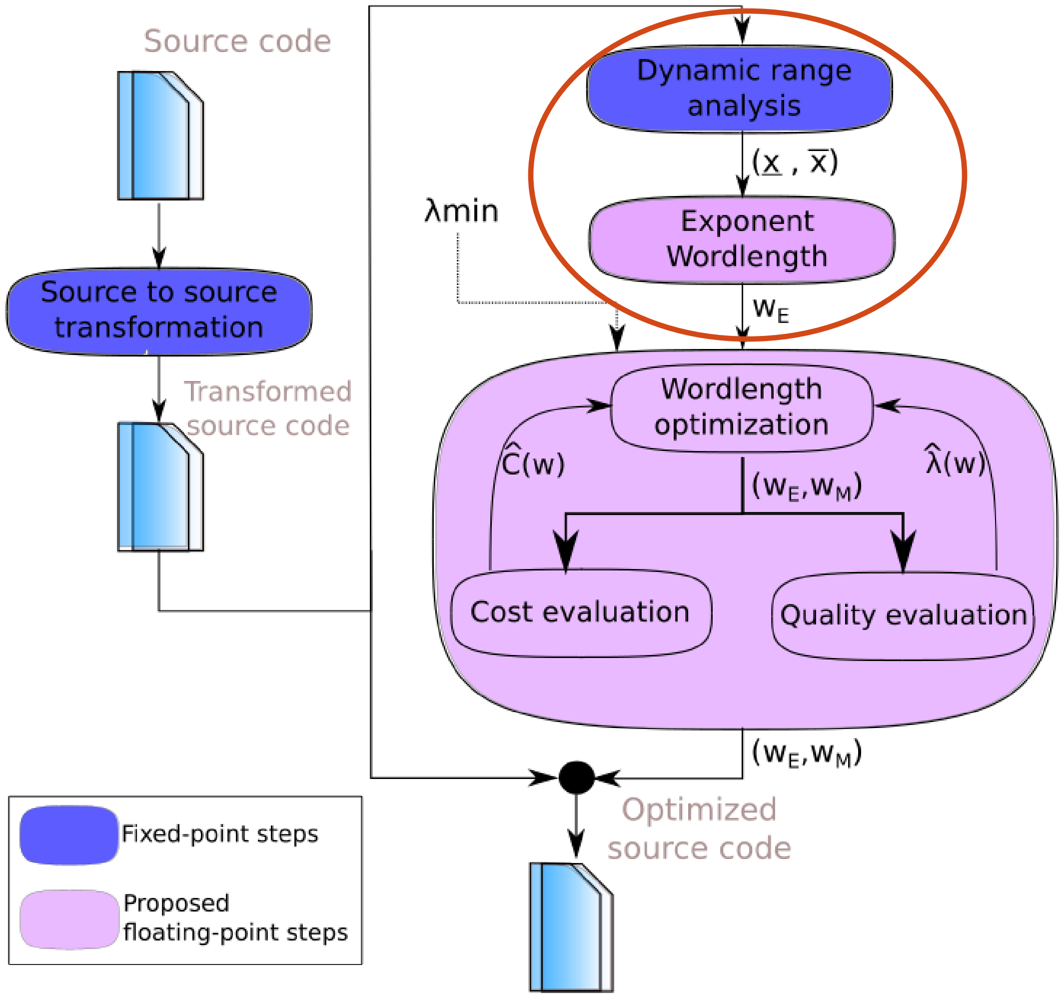
Exponent bits required



<u>Number of exponents bits required :</u>

$$W_{E_{Ri}} = \lceil \log_2 (R_{E_v}) \rceil + 1$$

Strategy 1 : Exponent and mantissa wordlength simultaneous optimization



Reduced exponent wordlength

With :

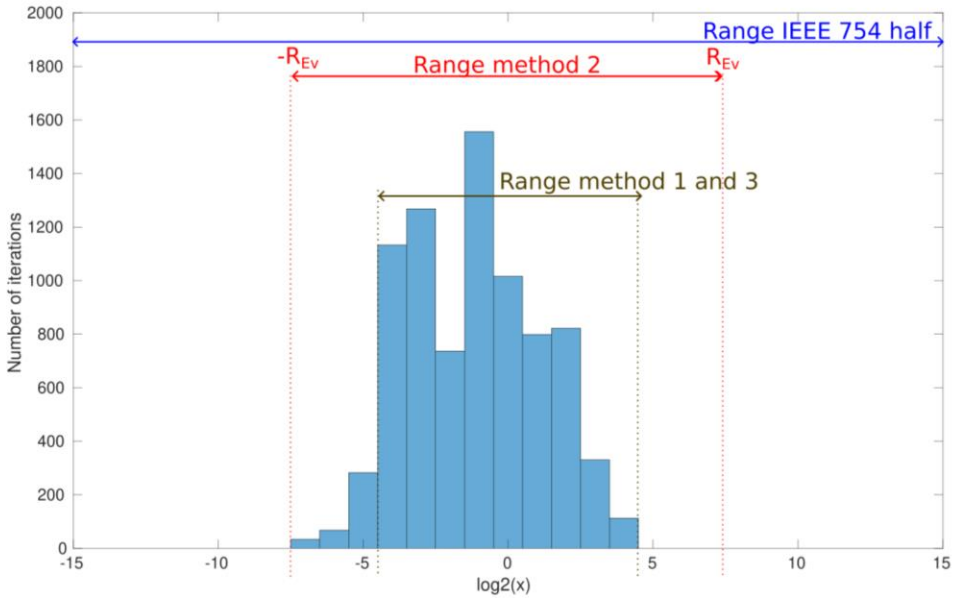
- **Input** : $(W_{E_{R1}}, W_{E_{R2}}, ..., W_{E_{RN}}, W_{M_{max}}, W_{M_{max}}, ..., W_{M_{max}})$
- λ_{min} : The user minimal quality

Strategy 1 : Exponent and mantissa wordlength simultaneous optimization

Expected pros

- Explore a lot of useful configuration

- Explore all dimensions



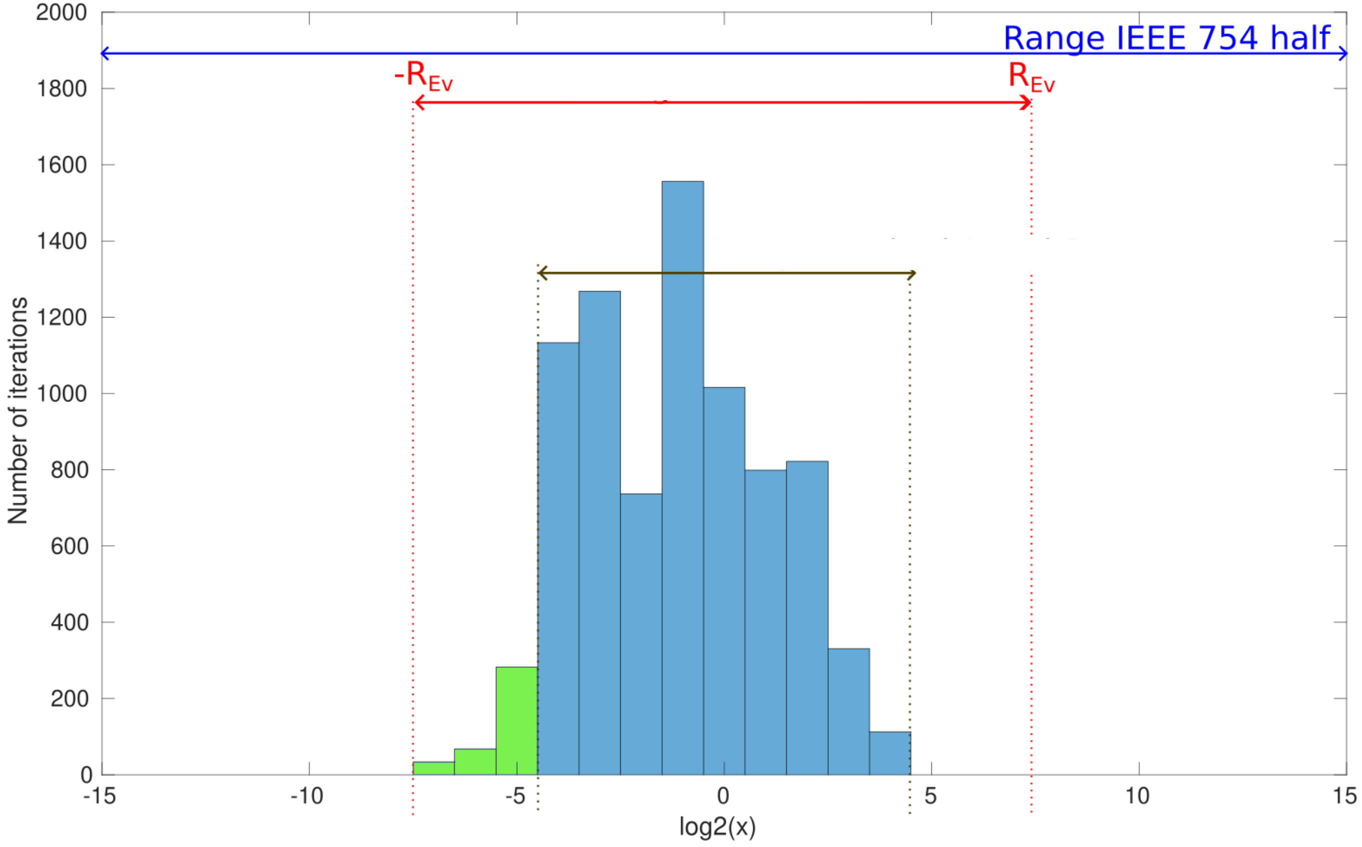
Expected cons

- Still a huge configuration number required :

$$n_p = M^{N_v} * \prod_{i=0}^{N_v} w_{E_i}^{\max}$$

- Huge Optimization time

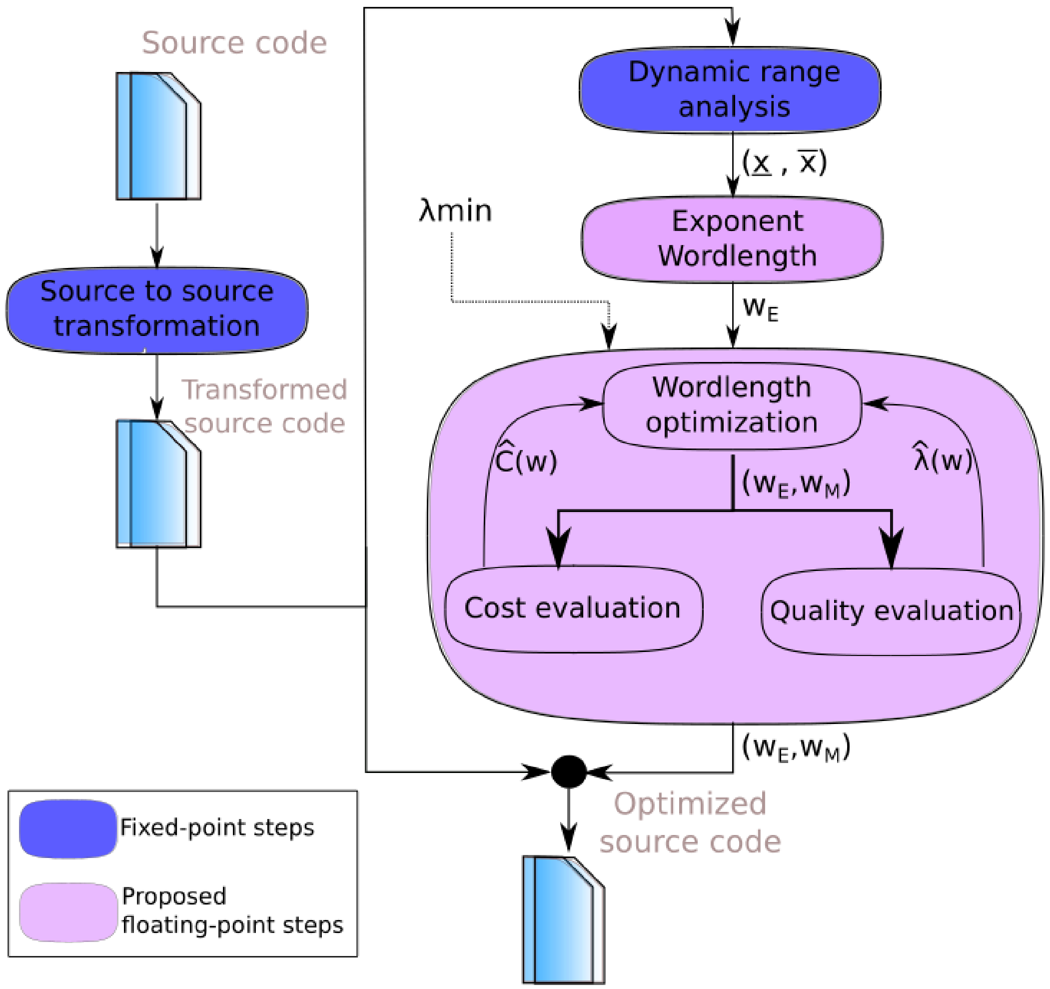
Strategy 2 : Mantissa wordlength only optimization



Number of exponents bits required :

$$W_{E_{Ri}} = \lceil \underbrace{\log_2}_{} (R_{E_v}) \rceil + 1$$

Strategy 2 : Mantissa wordlength only optimization



Only mantissa wordlength

With :

- **Input :** $\underbrace{(W_{M_{max}}, W_{M_{max}}, \ldots, W_{M_{max}})}$
- λ_{min} : The user minimal quality
- **All exponent size set to** $W_{E_{Ri}}$

Quentin Milot

Strategy 2 : Mantissa wordlength only optimization

Expected pros

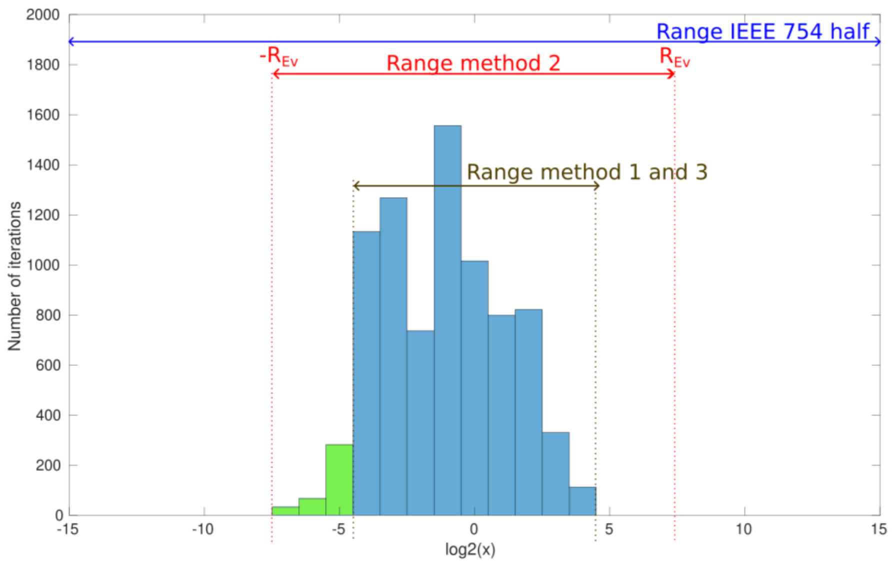
- Small configuration number required :

$$n_p = M^{N_v}$$

- Smaller Optimization time

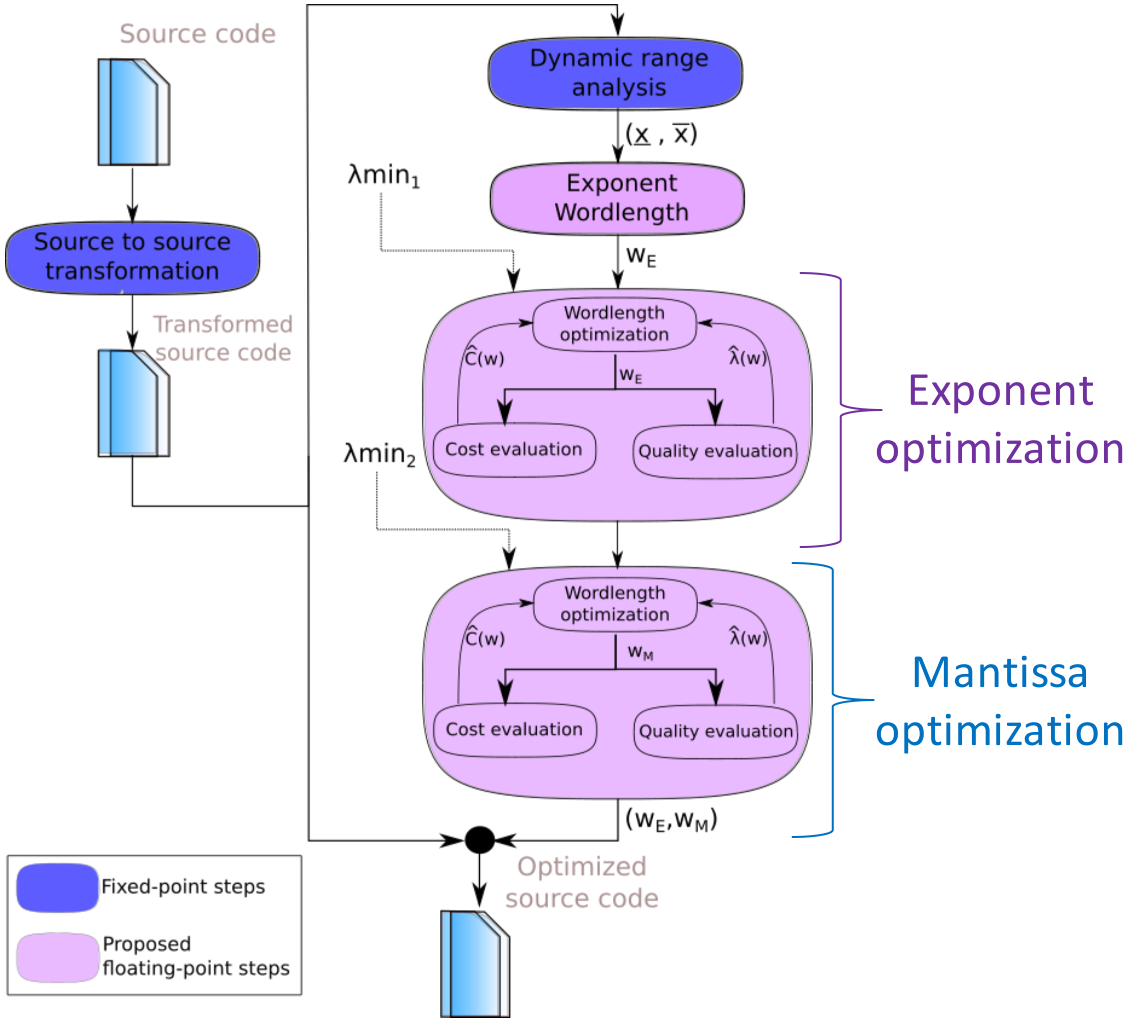
Expected cons

- Doesn't try to reduce the exponent wordlength



- Lower quality of optimization

Strategy 3 : Exponent and mantissa wordlength sequential optimization



With :

- Input 1 : $(W_{E_{R1}}, W_{E_{R2}}, \ldots, W_{E_{RN}})$
- λ_{min_1} : First user minimal quality
- All mantissa size set to $W_{M_{max}}$

- Input 2 : $(W_{M_{max}}, W_{M_{max}}, \ldots, W_{M_{max}})$
- λ_{min_2} : Second user minimal quality
- All exponent size set to W_{E_i} obtained with first optimization

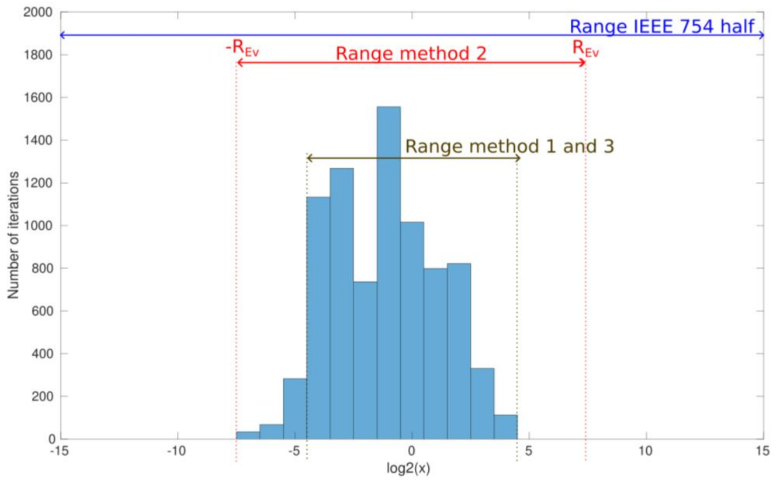
Strategy 3 : Exponent and mantissa wordlength sequential optimization

Expected pros

- Middle ground for configuration number :

$$n_p = M^{N_v} + \prod_{i=0}^{N_v} w_{E_i}^{\max}$$

- Middle ground for the result quality



Expected cons

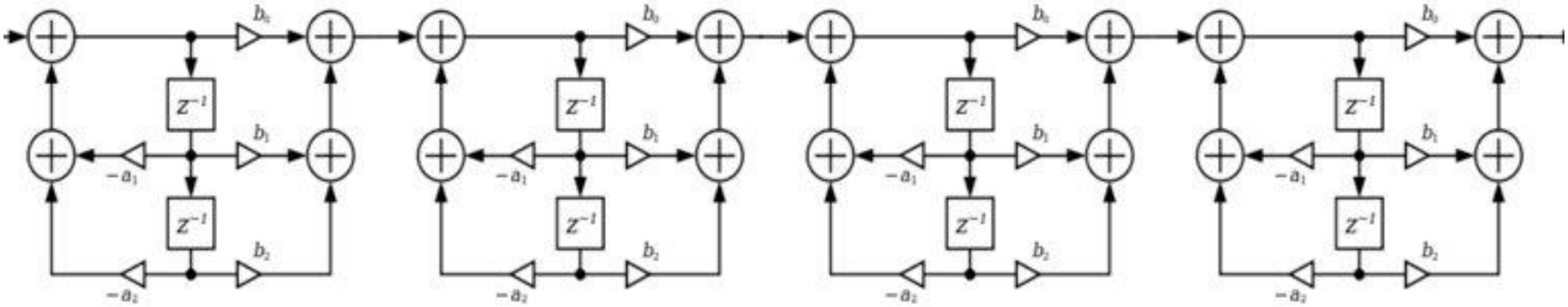
- Required and dependent of an hyperparameters : λ_{min_1}

Quentin Milot

Outline :

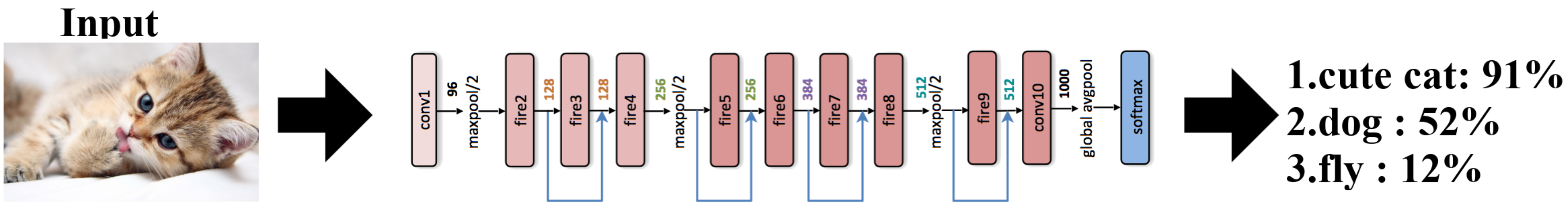
<u>Experiment 1:</u> Infinite Impulse Response filter

- Quality metric : Signal to Quantization Noise Ratio (SQNR)
- Number of variable to optimize : 7

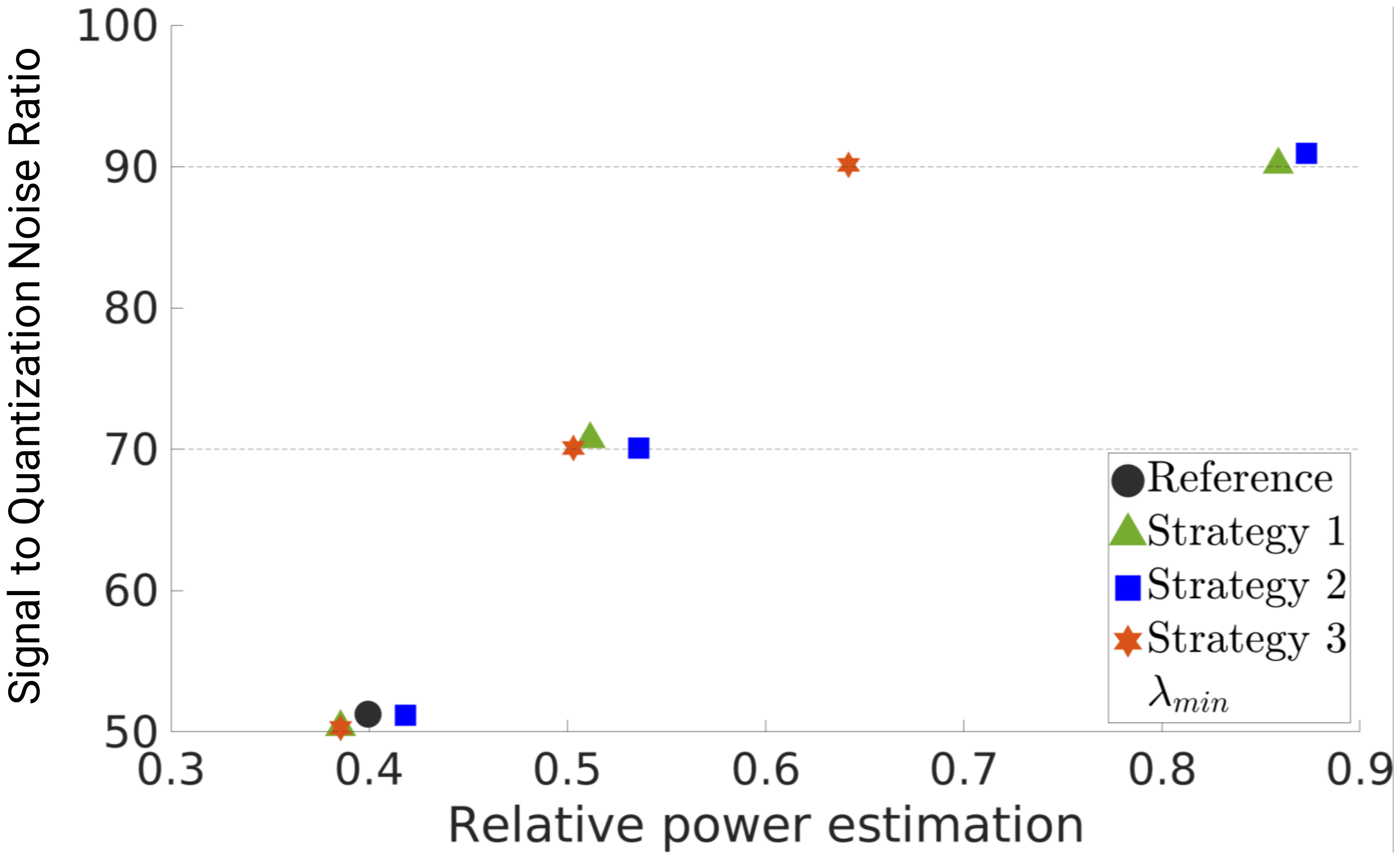


Experiment 2: Squeezenet

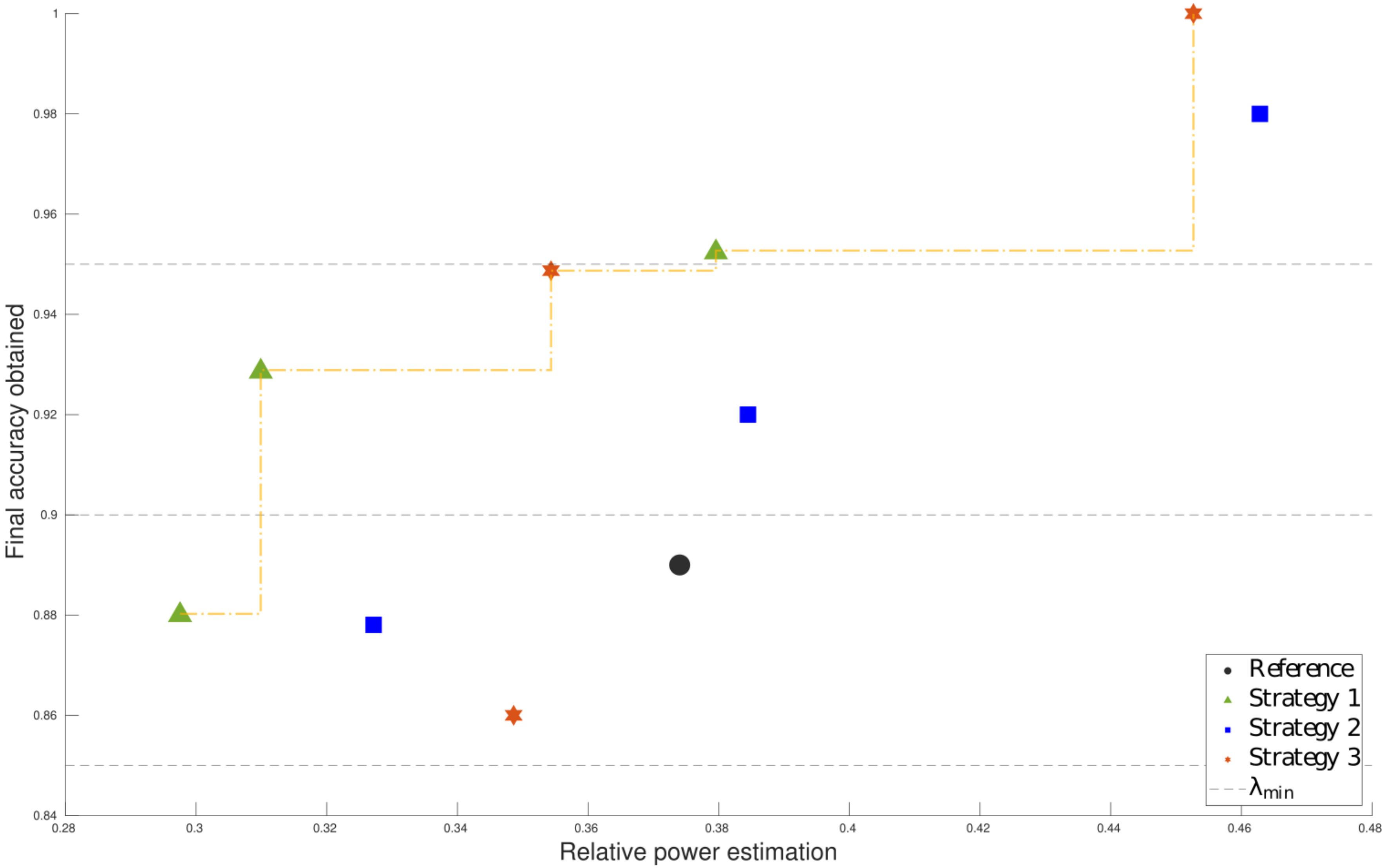
- Quality metric : proportion of similar top 1 classes detected
- Number of variable to optimize : 15



Power improvement : Infinite Impulse Response filter



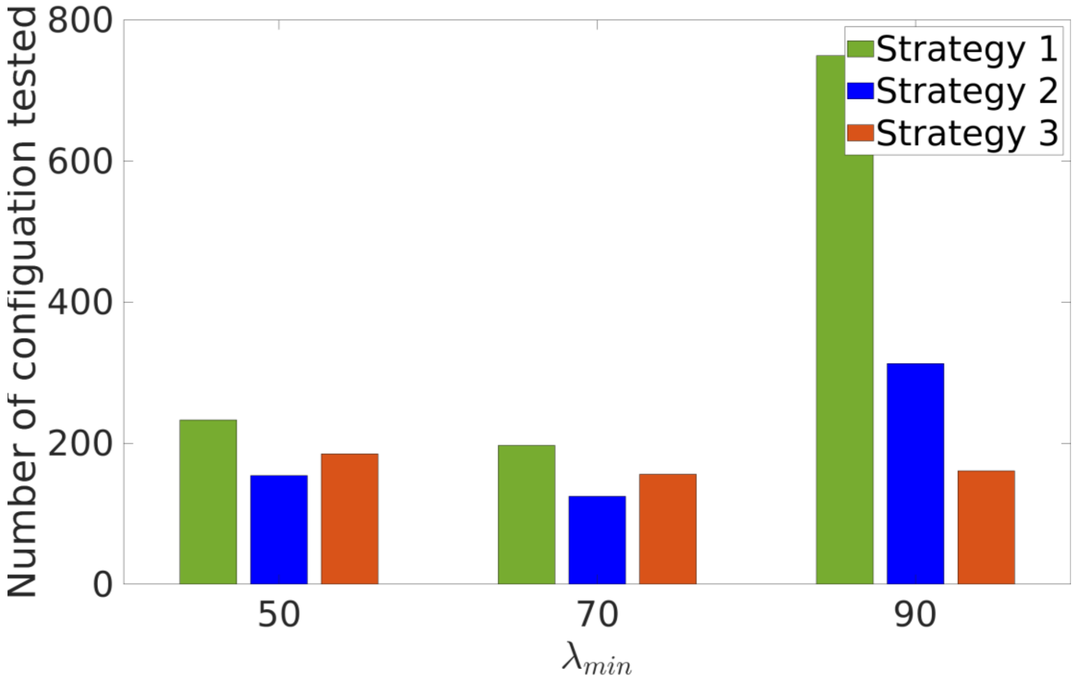
Power improvement : Squeezenet



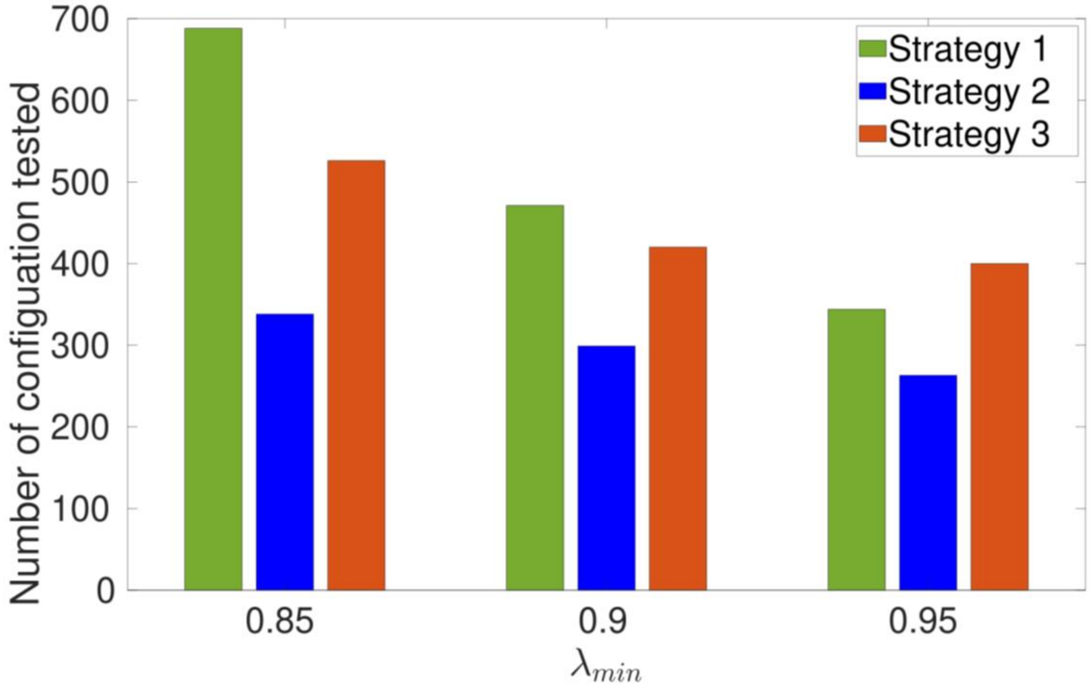
Quentin Milot

Number of configuration tested

Infinite Impulse Response filter



Squeezenet



Conclusion

- Proposed :
  - New custom floating-point refinement
  - Three strategies to optimize exponent and mantissa wordlength
  - Allow to follow a user-defined quality constraint
- Find improvement in terms of power and memory consumption compared to half floating-point
- Perspective :
  - Increase the quality of optimization
  - Implement memory cost reading/storing energy in power estimation