Comparison of Arithmetic Number Formats for Inference in Sum-Product Networks on FPGAs

Lukas Sommer, Lukas Weber, Martin Kumm, Andreas Koch

TECHNISCHE UNIVERSITÄT DARMSTADT

Authors

Lukas Weber
TU Darmstadt

Lukas Sommer
TU Darmstadt

Martin Kumm
HS Fulda

Andreas Koch
TU Darmstadt

SUM-PRODUCT NETWORKS

Sum-Product Networks (SPNs)

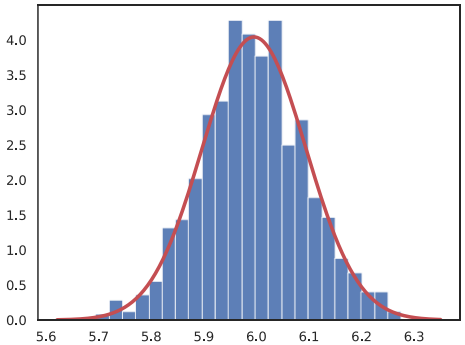
TECHNISCHE UNIVERSITÄT DARMSTADT

- ML technique from the class of **probabilistic models**
- Capture **joint probability** over a set of random variables
- Advantage over Neural Networks (NNs): **Exact inference**, express uncertainty over output
- Advantage over other Probabilistic Graphical Models (PGMs): **Tractable inference** in linear time wrt. network size
- Three kinds of nodes in DAG:
  - Sum nodes
  - Product nodes
  - Leaf nodes

SPNs – Node Types

Leaf Nodes:

- Univariate distributions
- Queried with evidence to retrieve probability value
- Represented by histograms



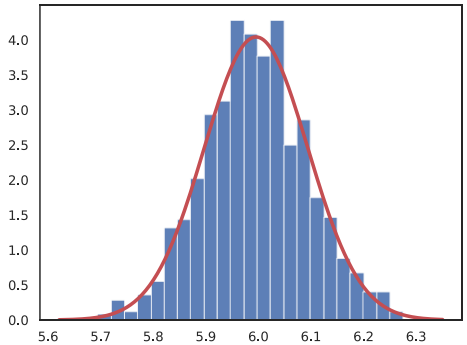
SPNs – Node Types

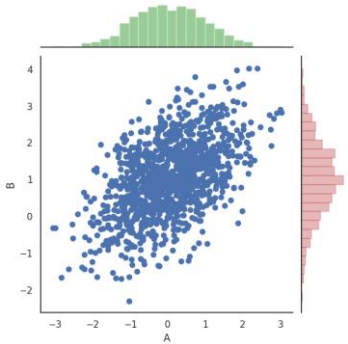
Leaf Nodes:

- Univariate distributions
- Queried with evidence to retrieve probability value
- Represented by histograms

Product Nodes:

- Factorization over independent random variables
- Multiplication of child nodes



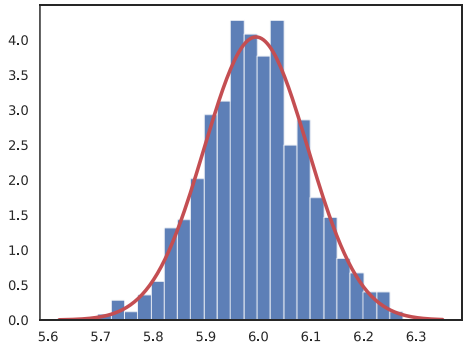


SPNs – Node Types

TECHNISCHE UNIVERSITÄT DARMSTADT

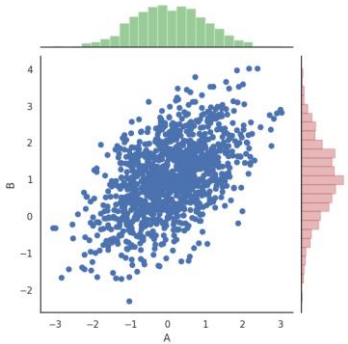
Leaf Nodes:

- Univariate distributions
- Queried with evidence to retrieve probability value
- Represented by histograms



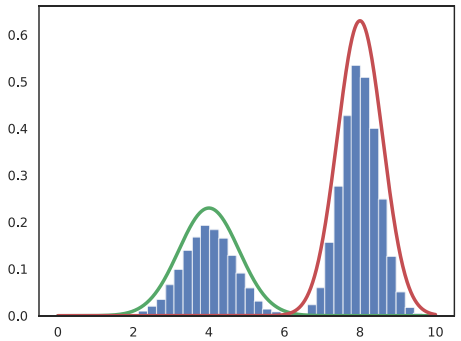
Product Nodes:

- Factorization over independent random vairables
- Multiplication of child nodes

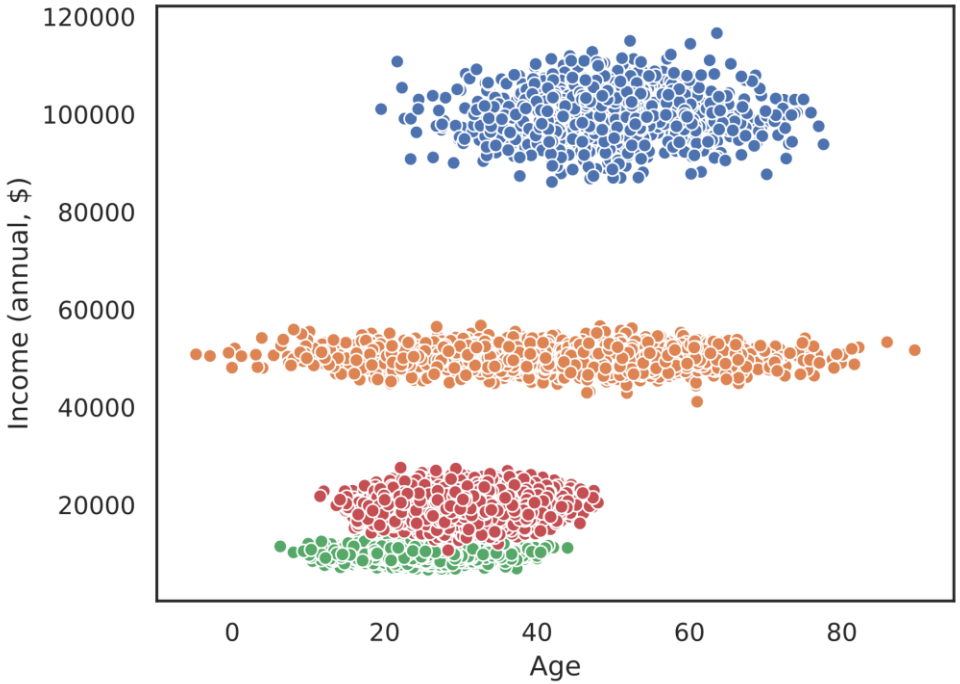


Sum Nodes:

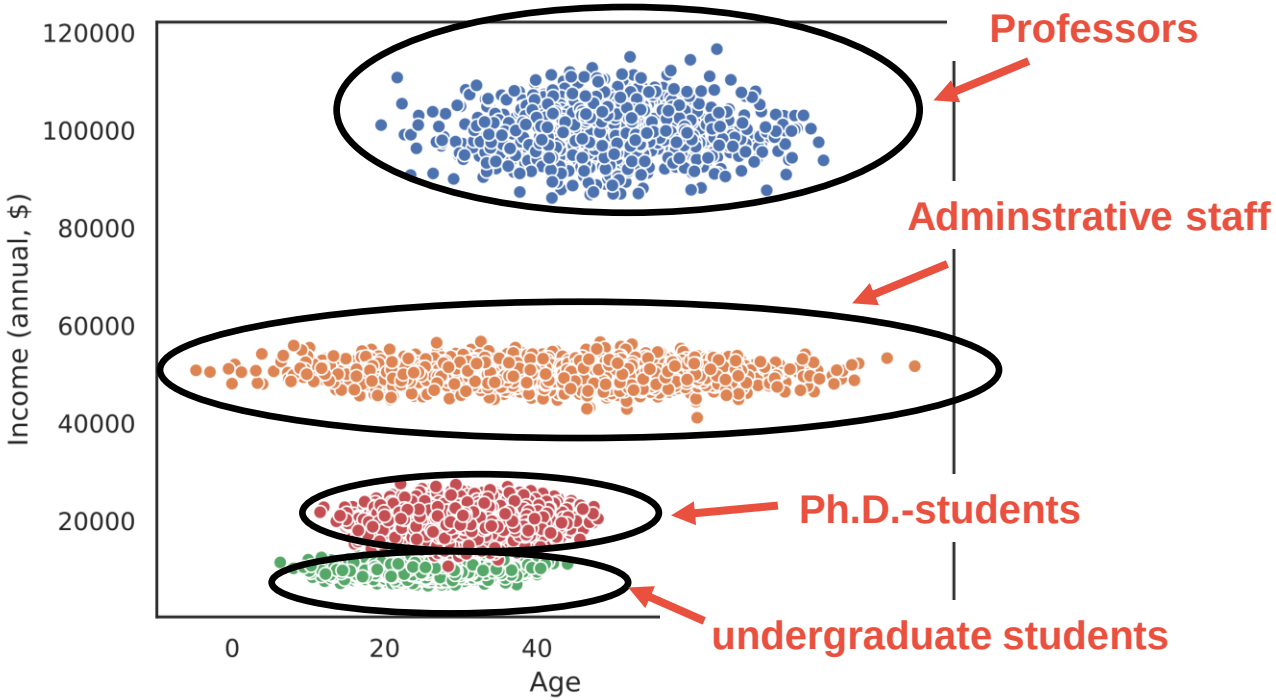
- Mixture of two distributions over the same set of random variables
- Weighted addition of child nodes



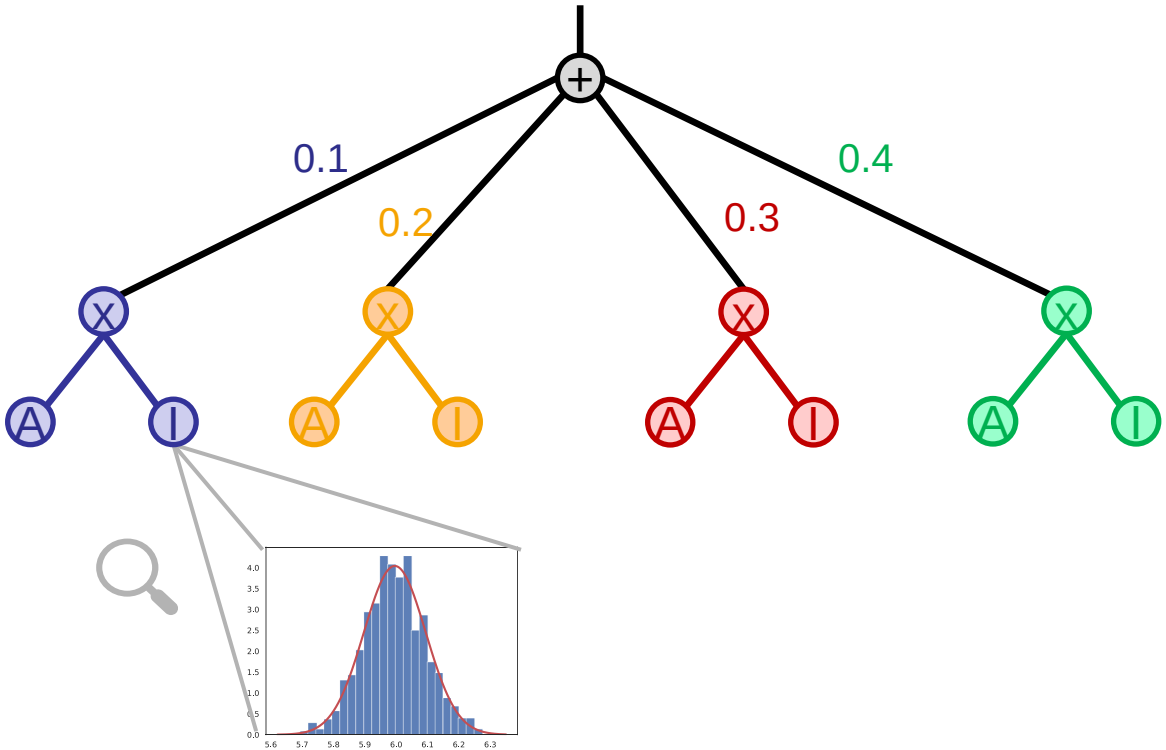
Sum-Product Networks – Example



Sum-Product Networks – Example



Sum-Product Networks – Example Network

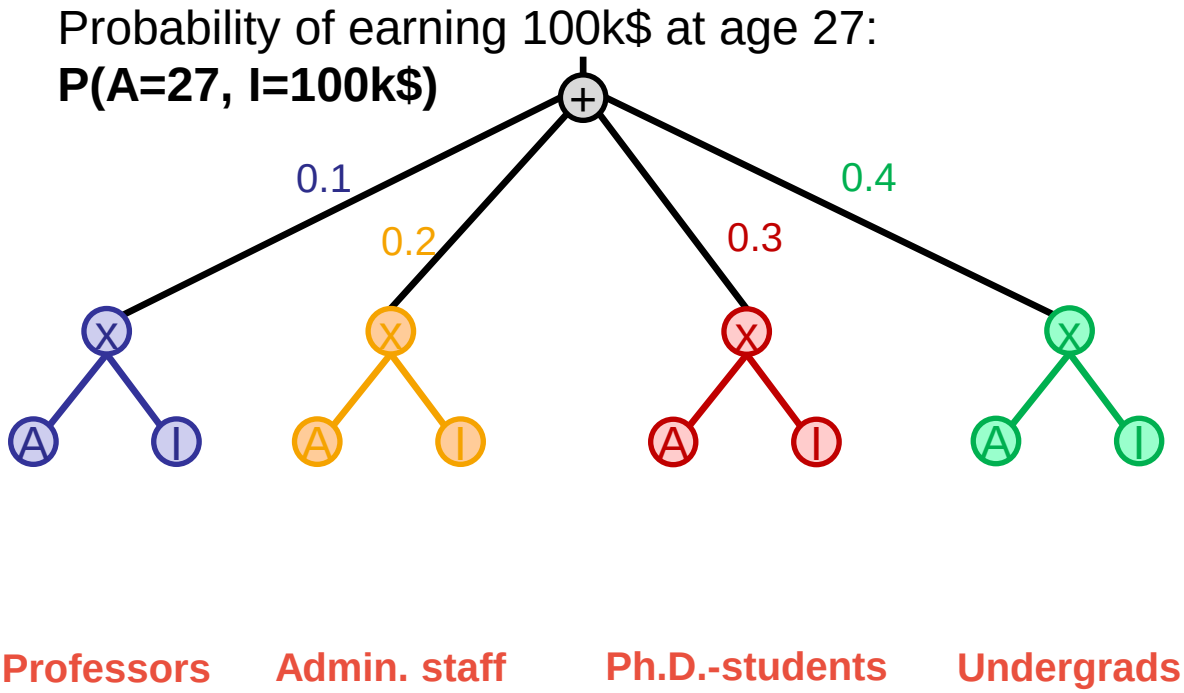


Sum-Product Networks - Inference

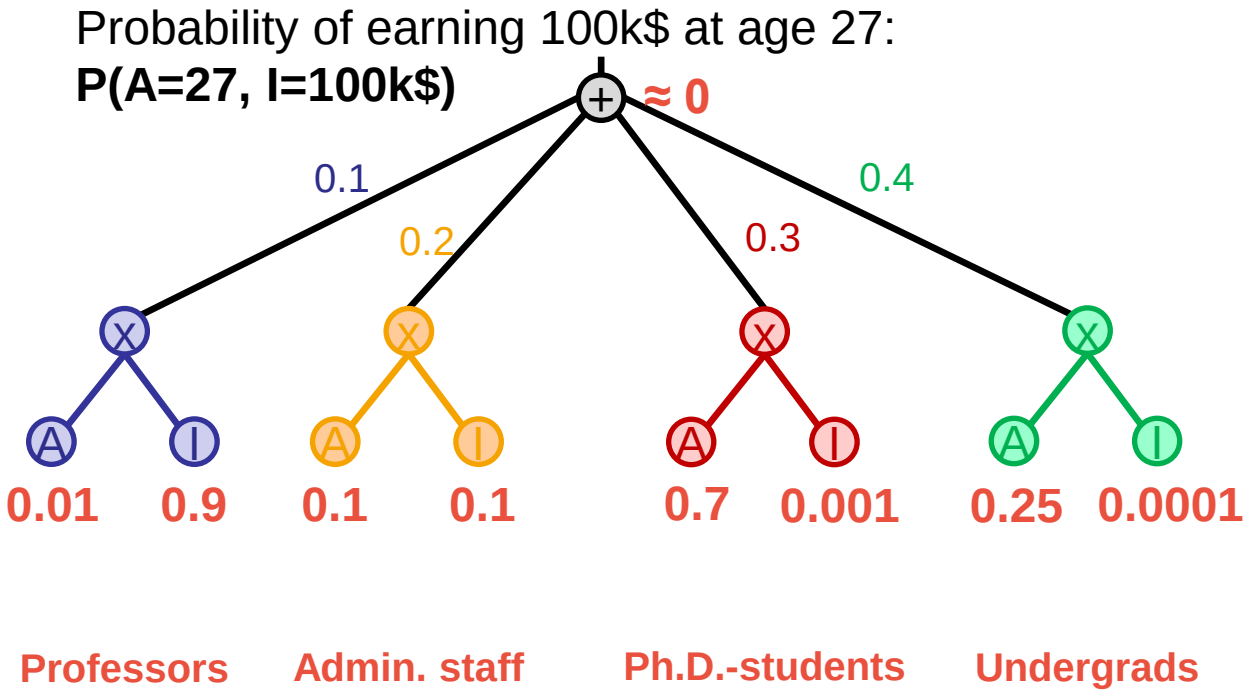
- Answer probabilistic queries & solve ML tasks
  - Probability of earning 100k$ at age 27: **P(A=27, I=100k$)**
  - Probability of earning 150k$: **P(I=150k$)** – marginalization
  - Add label {student, Ph.D.-student, admin, professor} as input variable, do classification based on information about age and income

- Inference is bottom-up evaluation of the SPN graph with (partial) evidence

Sum-Product Networks – Example Network



Sum-Product Networks – Example Network



TECHNISCHE UNIVERSITÄT DARMSTADT

ARITHMETIC NUMBER FORMATS

Arithmetic Number Formats

TECHNISCHE UNIVERSITÄT DARMSTADT

- Many Examples: Fixed Point, Floating Point, Posit, Logarithmic Number System (LNS)
- Each has unique characteristics
  - Best/worst case ranges
  - Resource-requirement
  - Special cases

SPNs & Arithmetic Number Formats

TECHNISCHE UNIVERSITÄT DARMSTADT

- SPNs compute probabilities (within the range [0, 1])
- SPNs only require addition and multiplication

- General purpose operators are **inefficient** for SPNs
  - Support addition & subtraction
  - Support negative values

Approach

TECHNISCHE UNIVERSITÄT DARMSTADT

- Implement addition,multiplication & conversion operators in a custom floating point format (CFP), LNS, Posit
- Exploit existing toolflow to generate FPGA-based SPN-accelerators
- Evaluate which number system is best wrt:
  - Required FPGA resources (LUTs, DSPs, BRAM)
  - Energy consumption
  - Performance, i.e. end-to-end throughput

Problems

TECHNISCHE UNIVERSITÄT DARMSTADT

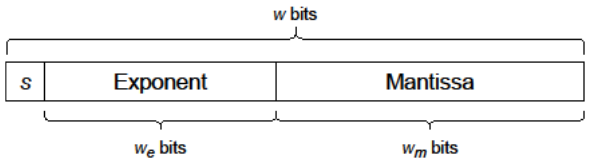
- LNS implementation more flexible and optimized than others
- **Solution:** Adapt state-of-the-art operators and optimize them
  - Posit: PACoGen
  - Floating Point: FloPoCo

- Goal: SPN-optimized & parameterized operators
  - Numeric range: [0,1]
  - No special values like NaN, infinities

Number Formats - Optimizations

Custom Floating Point:

- Based on FloPoCo
  - Only positive Inputs (Adder)
  - No subtractions (Adder)
  - Truncated multiplication and faithful rounding (Multiplier)



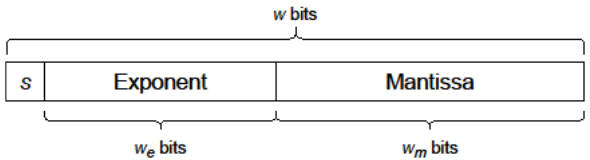
Number Formats - Optimizations

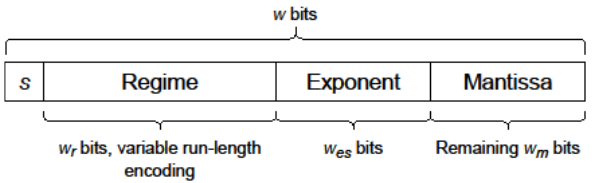
Custom Floating Point:

- Based on FloPoCo
  - Only positive Inputs (Adder)
  - No subtractions (Adder)
  - Truncated multiplication and faithful rounding (Multiplier)

Posit:

- Based on PaCoGen
  - No subtractions (Adder)
  - Deeper pipelines (Both)
  - Optimal DSP-based multiplication-scheme (Multiplier)





Number Formats - Optimizations

TECHNISCHE UNIVERSITÄT DARMSTADT

Custom Floating Point:

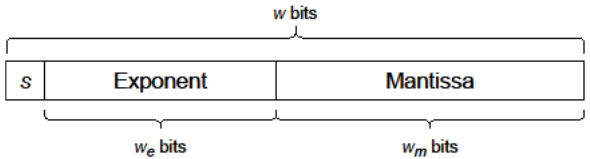
- Based on FloPoCo
  - Only positive Inputs (Adder)
  - No subtractions (Adder)
  - Truncated multiplication and faithful rounding (Multiplier)

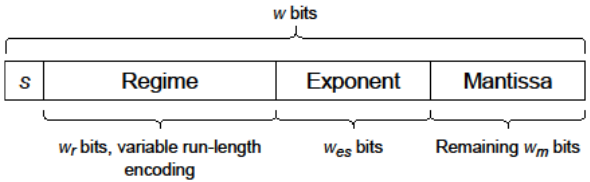
Posit:

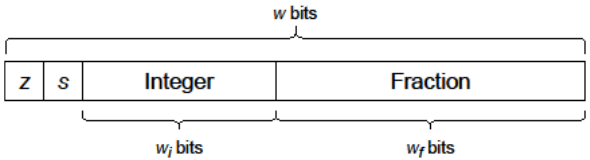
- Based on PaCoGen
  - No subtractions (Adder)
  - Deeper pipelines (Both)
  - Optimal DSP-based multiplication-scheme (Multiplier)

Logarithmic Number System:

- Taken from prior work
  - Only positive numbers (Both)
  - Optimal DSP-based multiplication-scheme (Adder)
  - 2nd degree polynomial interpolation (Adder)







What about Fixed Point?

TECHNISCHE UNIVERSITÄT DARMSTADT

- Not suited for this use case:
  - Relative error margins of 1e-6
  - Very small numbers may occur (1e-82 as smallest value)
  - Required Fixed Point format would be ~300bits

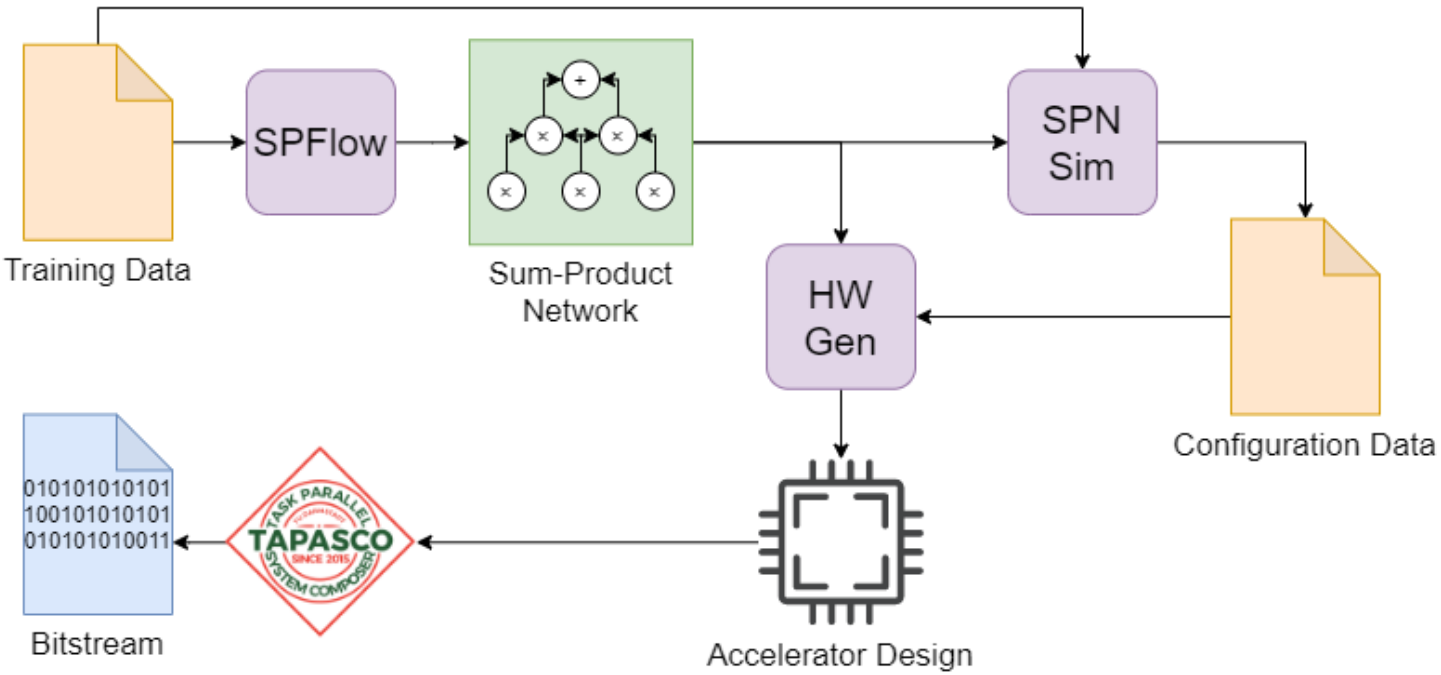
Operator Configuration

TECHNISCHE UNIVERSITÄT DARMSTADT

- All evaluated operators are parameterized
- SPN error approximation using software-implementations of the operators
  - Simulation of possible configurations in a Design Space Exploration (DSE)
  - DSE runtime is negligible compared to P&R time
  - Highly parallelized toolflow for simulations

- **Smallest** possible configuration, where maximum relative error is < 1e-6

Toolflow


Training Data
SPFlow
Sum-Product Network
SPN Sim
HW Gen
Configuration Data
010101010101
100101010101
010101010011
TAPASCO
Bitstream
Accelerator Design

TECHNISCHE UNIVERSITÄT DARMSTADT

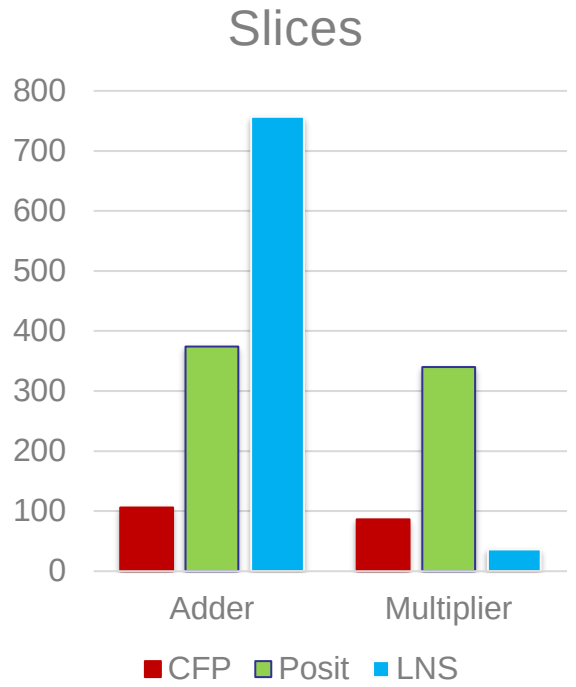
EVALUATION

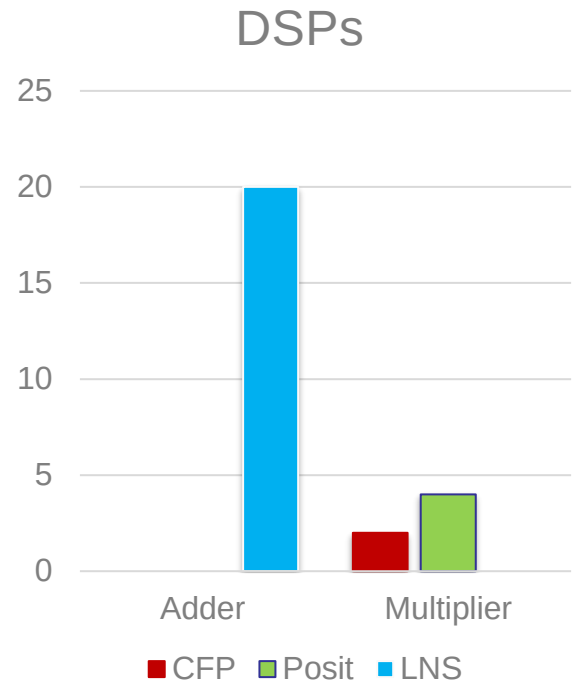
Evaluation Setup

TECHNISCHE UNIVERSITÄT DARMSTADT

- 16 benchmarks
- **Implemented & evaluated** on the **Xilinx VC709** evaluation board
- Comparison of all three number formats
- Additional comparison to prior work (using FloPoCo-based double precision)

Operator Comparison

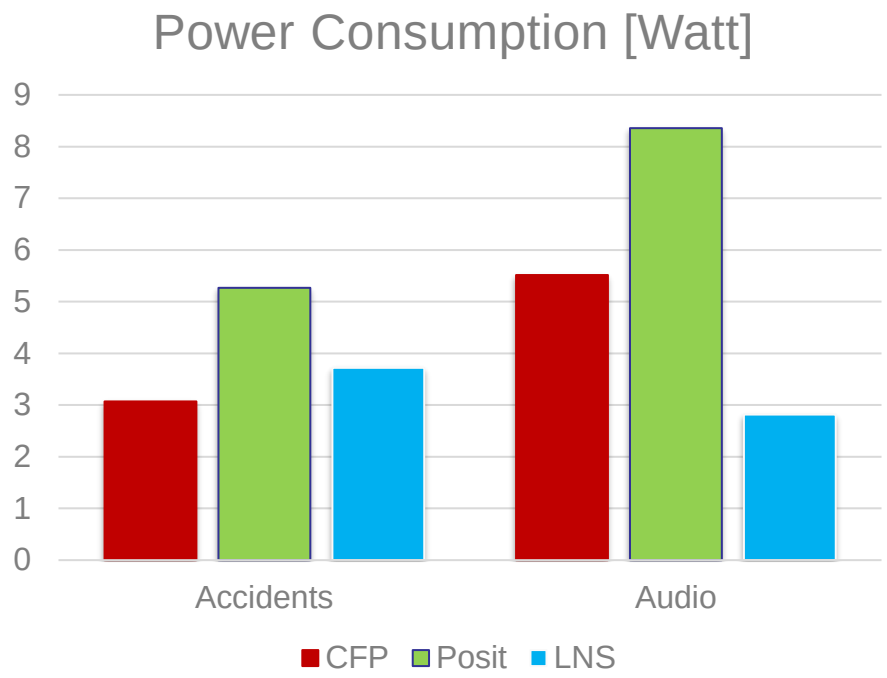




- Posit behaves similar to CFP
- LNS behaves inversely

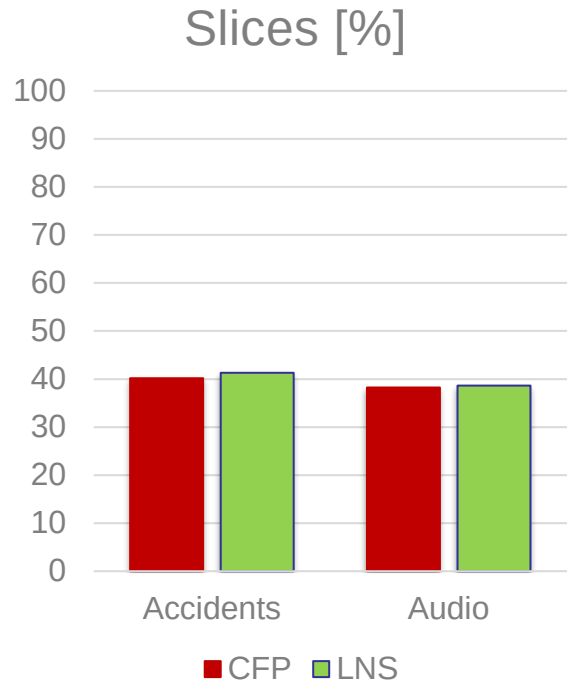
- **Multiplier:Adder-Ratio might be interesting**
  - Audio: 23:1
  - Accidents: 8:1

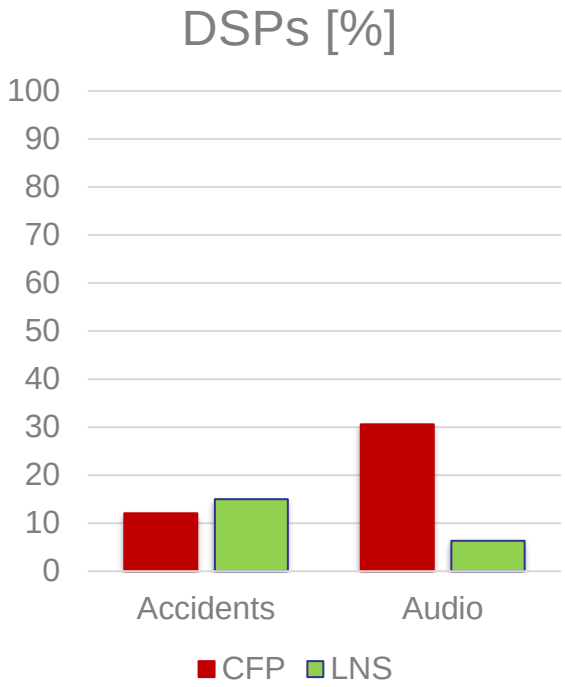
Power Consumption



- Post-implementation timing simulation for power estimation
- Prior work required 4x more power
- CFP & LNS are more power efficient than Posit
- LNS is better suited for higher M:A Ratios, CFP for lower M:A Ratios

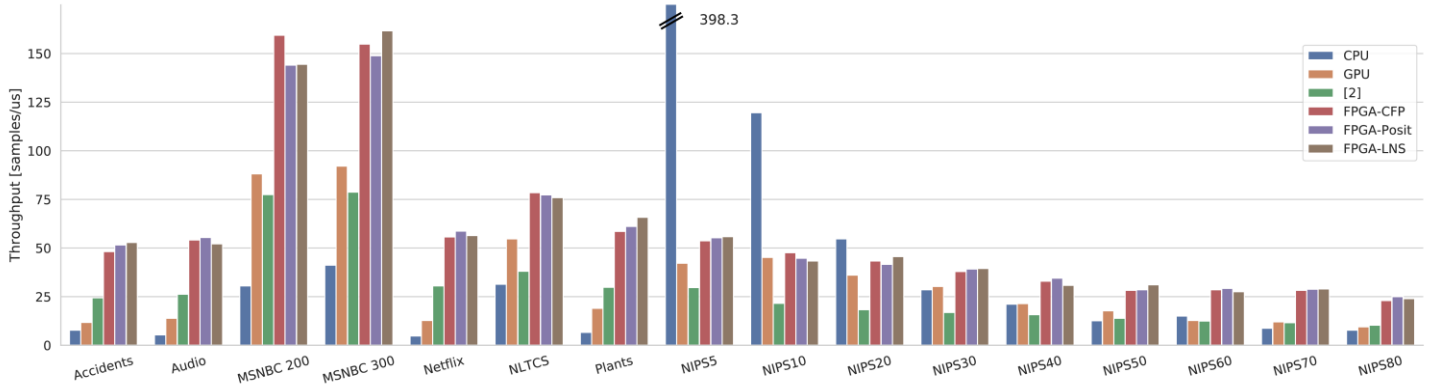
Overall Utilization





- Accidents:
  - Slices similar
  - CFP less DSPs
- Audio:
  - Slices similar
  - LNS less DSPs

Performance



- In comparison to CPU (Ryzen 1600X):
  - Up to 12x speedup, geo.-mean 2.5x
  - Smallest Benchmarks are faster on CPU, due to less overhead
- In comparison to CPU (1080Ti):
  - Up to 4.6x speedup, geo.-mean 2.1x
  - Custom CUDA-Flow, that is 90x better than our prior work

Conclusion

TECHNISCHE UNIVERSITÄT DARMSTADT

- **Performance:**
  - All formats are similar
  - Speedups of up to 12x (vs. CPU) and 4.6x (vs. GPU)
- **Utilization & Power Consumption:**
  - CFP & LNS are better suited depending on M:A-ratio
  - Posit requires more power & resources

TECHNISCHE UNIVERSITÄT DARMSTADT

THANKS FOR WATCHING!

We are looking forward to interesting discussions on the forums.