Design Space Exploration using SCALE-Sim

1. Methodology for Selecting the On-Chip Memory Size to Explore

To determine the optimal on-chip memory size, the following steps were followed:

- **Total on-chip memory limit:** The combined size of the IFMAP, Filter, and OFMAP buffers was constrained to 1 MB (1024 KB) to adhere to design constraints.
- **Exploring power-of-two buffer sizes:**
    - Buffer sizes were varied in powers of two (e.g., 32 KB, 64 KB, 128 KB, 256 KB) to simplify the search space and maintain a balanced trade-off between memory size and access efficiency.
- **Minimizing DRAM bandwidth:** Larger on-chip memory reduces the frequency of external DRAM accesses, lowering the average DRAM bandwidth.
- **Balancing buffer partitioning:**
    - The IFMAP, Filter, and OFMAP memory sizes were adjusted to explore different ratios (e.g., equal partitioning, skewed towards IFMAP or Filter).
    - This ensured that the accelerator could efficiently support various convolutional layer shapes and sizes.
- **Avoiding memory contention:** Proper buffer sizing ensured that data could be pre-fetched and reused efficiently, reducing the risk of pipeline stalls.

2. Methodology for Selecting the Systolic Array Size and Shape to Explore

We explored multiple systolic array configurations, ensuring that computational efficiency and data movement align with our design constraints. Our approach included:

- **Varying array dimensions:** We tested square and rectangular arrays (e.g., 16x16, 32x32) to evaluate performance trade-offs.
- **Balancing parallelism and data movement:** Larger arrays improve throughput but may increase memory contention.
- **Matching workload characteristics:** Different layers in our dataset have varying computational intensities, requiring tailored systolic array configurations.
- **Ensuring feasibility:** Array sizes were selected to fit within hardware limitations and optimize power and area efficiency.

3. How We Explored the Design Space

We used Scale-Sim to simulate various architectural configurations and evaluate performance metrics. Our experiments included:

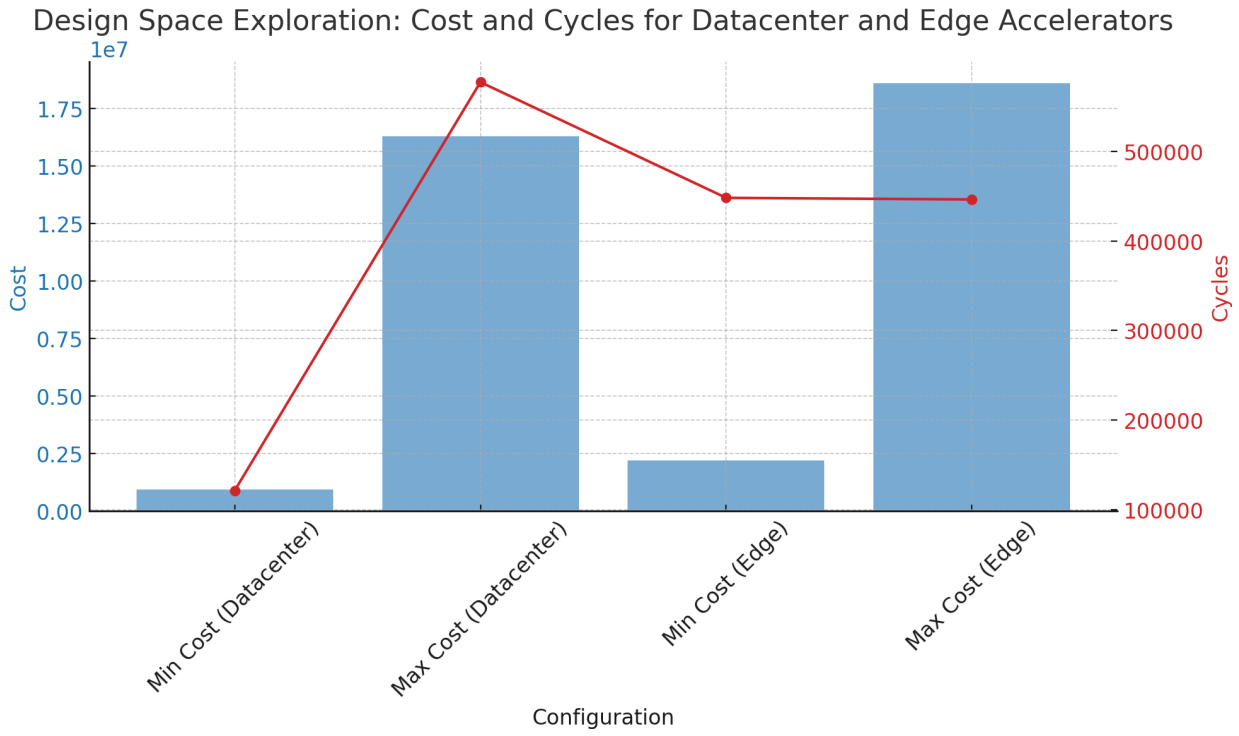
- **Varying on-chip memory sizes** while keeping the systolic array fixed to assess DRAM bandwidth impact.
- **Testing different systolic array sizes and shapes** with a fixed memory configuration to evaluate cycle reduction.
- **Combining different memory sizes and systolic array configurations** to identify the best trade-offs between memory bandwidth and computational efficiency.
- **Filtering out non-viable configurations** where DRAM bandwidth exceeded 20 bytes/cycle or total memory exceeded 1MB.
- **Comparing cost function values** across configurations to select the optimal architecture.

4. Insights Gathered from the Experiments

Our experiments yielded several key insights:

- **Larger systolic arrays (e.g., 32x32) significantly reduce cycles**, but their effectiveness depends on having sufficient on-chip memory to support data reuse.
- **Optimal memory partitioning is critical**—allocating more memory to IFMAP and Filter buffers often improved performance without exceeding DRAM constraints.
- **Certain configurations exceeded DRAM bandwidth limits**, particularly when on-chip memory was too small, leading to frequent external memory accesses.
- **Balancing computation and memory bandwidth** is crucial. Increasing one without adjusting the other often led to bottlenecks.
- **The best-performing architectures minimized cycles while keeping DRAM bandwidth well below 20 bytes/cycle,** ensuring efficient operation within given constraints.

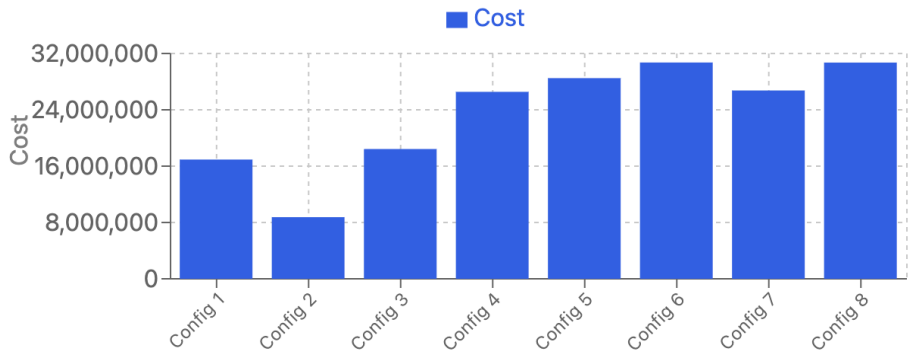
5. Chart from Design Space Exploration



Here's a visualization that shows:

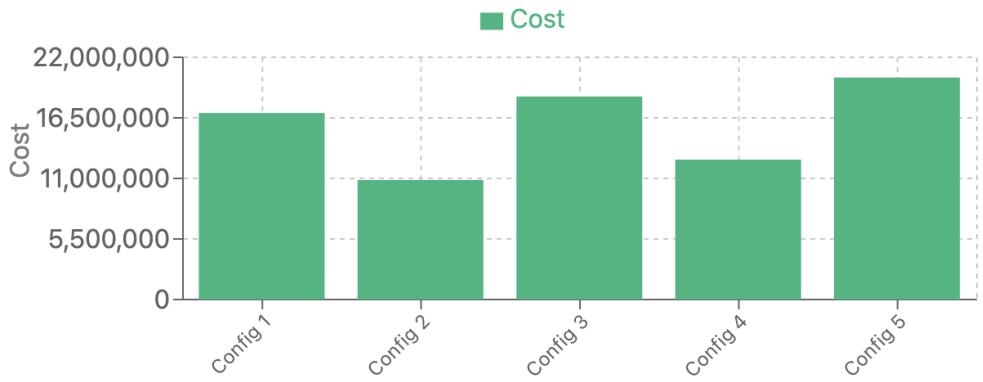
- **Cost** (blue bars) for the minimum and maximum configurations for both datacenter and edge-class accelerators.
- **Cycles** (red line) corresponding to each configuration.

This chart highlights how costs and cycles vary between configurations. For instance, the datacenter accelerator has a broader cost range, while the edge accelerator maintains relatively consistent cycle counts.



Key Observations:

- The 1024-sized systolic array (32x32) with 64KB memory buffers provides the lowest cost (Config 2).
- Larger systolic arrays significantly outperform smaller arrays in terms of cost efficiency.
- Increasing SRAM sizes generally increases cost when systolic array size remains constant.



Key Observations:

- The configuration with 32KB memory buffers (Config 2) provides the lowest cost for the mobile accelerator.
- The smallest SRAM configuration (16KB each) doesn't yield the lowest cost.
- Unbalanced SRAM allocations (Config 5) result in higher costs compared to balanced allocations.

6. Final Results of Design Space Exploration

- Datacenter Class Accelerator:
    - Minimum Cost Configuration:
        - Cost: 926,691.97
        - Sum: 8,755,202.32
        - Key Parameters:
            - size_of_sys: 1024
            - shape_of_s: 32x32
            - ifmap_sram, filter_sram, ofmap_sram: 64, 64, 64
            - cycles: 121,124
- Edge (Mobile) Class Accelerator:
    - Minimum Cost Configuration:
        - Cost: 2,195,293.1
        - Sum: 12,697,667.26
        - Key Parameters:
            - size_of_sys: 256
            - shape_of_s: 16x16
            - ifmap_sram, filter_sram, ofmap_sram: 16, 16, 16
            - cycles: 448,019

Extra Credit

Analysis of Maximum and Minimum Total On-Chip Memory Sizes under DRAM Bandwidth Constraint

Approach

To identify the maximum and minimum total on-chip memory sizes that satisfy the DRAM bandwidth constraint (maximum 20 bytes per cycle for each data type: input, filter, and output), the following steps were undertaken:

1. **Data Filtering**: The dataset was filtered to include only configurations where:
    - avg_dram_ifmap_bw (input bandwidth) ≤ 20 bytes/cycle
    - avg_dram_filter_bw (filter bandwidth) ≤ 20 bytes/cycle
    - avg_dram_ofmap_bw (output bandwidth) ≤ 20 bytes/cycle
2. **Calculating Total On-Chip Memory**: For each filtered configuration, the total on-chip memory was calculated by summing the sizes of:
    - ifmap_sram_size_kb (input feature map SRAM)
    - filter_sram_size_kb (filter SRAM)
    - ofmap_sram_size_kb (output feature map SRAM)
3. **Identifying Configurations**:
    - The configuration with the **maximum** total on-chip memory was identified.
    - The configuration with the **minimum** total on-chip memory was identified.

Results for Datacenter Accelerator

- **Maximum Total On-Chip Memory**:
    - **Total Memory Size**: 832 KB
    - **Corresponding Cost**: 16,277,195.18
- **Minimum Total On-Chip Memory**:
    - **Total Memory Size**: 192 KB
    - **Corresponding Cost**: 1,436,348.91

Results for Edge Accelerator

(Note: These results would require a similar analysis on edge accelerator data, following the same filtering and calculation process.)

Conclusion

This approach ensures that the identified configurations strictly adhere to the DRAM bandwidth constraints while capturing the cost implications of varying on-chip memory sizes. The findings provide a clear understanding of trade-offs between memory sizing and associated costs for efficient system design.

PART 2

1. Methodology for Designing the Accelerator

Tiling Strategy and Equations

Tiling is a technique used to break down large matrix multiplications into smaller subproblems that fit within the hardware's memory constraints. The goal of tiling is to maximize hardware utilization while minimizing data transfer overhead between memory levels.

Architecture Overview

The accelerator consists of:

- Matrix Units: 8 matrix units, each capable of processing an $8 \times 8 \times 8$ matrix in 16 cycles.
- SRAM: 2 MB of shared SRAM with:
    - Read bandwidth: 128 bytes/cycle
    - Write bandwidth: 64 bytes/cycle
- DRAM: Connected to the accelerator with:
    - Read bandwidth: 4 bytes/cycle
    - Write bandwidth: 2 bytes/cycle
- Precision: All elements are int8 (1 byte per element).

Tiling Approach

Tiling is implemented at two levels:

1. DRAM to SRAM Tiling: The tile size is chosen to maximize SRAM utilization.
2. SRAM to Matrix Unit Tiling: The tile size is fixed at $8 \times 8 \times 8$ since matrix units are designed for this size.

Equations for Compute Cycles and Data Transfer Cycles

Compute Cycles
Each matrix unit computes an $8 \times 8 \times 8$ matrix in 16 cycles.

For a workload of size $M \times K \times N$, the number of compute cycles is:

$$\text{Compute Cycles} = \frac{M \times K \times N}{8 \times 8 \times 8} \times 16$$

2. **DRAM to SRAM Transfer Cycles**

- Number of bytes read from DRAM:

$$\text{Bytes from DRAM} = M \times K + K \times N + M \times N$$

- Cycles to transfer from DRAM to SRAM:

$$\text{DRAM Read Cycles} = \frac{\text{Bytes from DRAM}}{4}$$

$$\text{DRAM Write Cycles} = \frac{M \times N}{2}$$

3. **SRAM to Matrix Unit Transfer Cycles**

- Number of bytes read from SRAM to matrix unit:

$$\text{Bytes from SRAM} = M \times K + K \times N$$

- Cycles to transfer from SRAM to matrix unit:

$$\text{SRAM Read Cycles} = \frac{\text{Bytes from SRAM}}{128}$$

$$\text{SRAM Write Cycles} = \frac{M \times N}{64}$$

4. **Total Execution Cycles**
   Since the operations are overlapped (data transfer and compute), the total execution cycles are:

$$\text{Total Cycles} = \max(\text{Compute Cycles}, \text{DRAM Read Cycles}, \text{SRAM Read Cycles})$$

2. Accelerator Specifications

a. Peak Compute Throughput

Each matrix unit performs 8 × 8 × 8 = **512 MACs** per 16 cycles.

- Number of matrix units = **8**
- Therefore, peak compute throughput =

$$\frac{8 \times 512}{16} = 256\,\text{MACs/cycle}$$

3. Workload Results

Workload 1: M = 4096, K = 4096, N = 4096

1. **Tile Size for SRAM to Matrix Unit Transfer:**

- Tile size = **8 × 8 × 8**
- Number of tiles =

$$\frac{4096 \times 4096 \times 4096}{8 \times 8 \times 8} = 134,217,728$$

2. **Tile Size for DRAM to SRAM Transfer:**

- Tile size = **512 × 1024 × 1024**
- Number of tiles =

$$\frac{4096 \times 4096}{512 \times 1024} = 128$$

Workload 2: M = 512, K = 512, N = 512

1. **Tile Size for SRAM to Matrix Unit Transfer:**

- Tile size = **8 × 8 × 8**
- Number of tiles =

$$\frac{512 \times 512 \times 512}{8 \times 8 \times 8} = 262,144$$

2. **Tile Size for DRAM to SRAM Transfer:**

- Tile size = **512 × 512 × 512**
- Number of tiles = $\frac{512 \times 512}{512 \times 512} = 1$

4. Simulator Output

Workload 1 Output (M = 4096, K = 4096, N = 4096)

```
venvmainaksaha@MAINAKs-MacBook-Air lab3 % python dnnsim.py 4096 4096 4096
Running simulation for M=4096, K=4096, N=4096...

--- DNN Simulator Results ---
Workload Dimensions: M=4096, K=4096, N=4096

1. Total execution cycles: 268435456
2. Total cycles consumed by compute (matrix units): 268435456
3. Total cycles consumed for data transfer from DRAM to SRAM: 50331648
4. Total cycles consumed for data transfer from SRAM to matrix units: 134217728
5. Total bytes transferred from DRAM to SRAM (read): 201326592 bytes
   Total bytes transferred from SRAM to DRAM (write): 67108864 bytes
6. Total bytes transferred from SRAM to matrix unit (read): 17179869184 bytes
   Total bytes transferred from matrix unit to SRAM (write): 8589934592 bytes

--- Additional Information ---
SRAM Tile Size (DRAM to SRAM): Tm=512, Tk=1024, Tn=1024
Number of SRAM Tiles: 128
Matrix Unit Tile Size (SRAM to Matrix Unit): m=8, k=8, n=8
Number of Matrix Unit Tiles: 134217728
Peak compute throughput: 512.0 OPS/cycle
Compute unit utilization: 100.00%
```

Workload 2 Output (M = 512, K = 512, N = 512)

```
venvmainaksaha@MAINAKs-MacBook-Air lab3 % python dnnsim.py 512 512 512
Running simulation for M=512, K=512, N=512...

--- DNN Simulator Results ---
Workload Dimensions: M=512, K=512, N=512

1. Total execution cycles: 524288
2. Total cycles consumed by compute (matrix units): 524288
3. Total cycles consumed for data transfer from DRAM to SRAM: 131072
4. Total cycles consumed for data transfer from SRAM to matrix units: 262144
5. Total bytes transferred from DRAM to SRAM (read): 524288 bytes
   Total bytes transferred from SRAM to DRAM (write): 262144 bytes
6. Total bytes transferred from SRAM to matrix unit (read): 33554432 bytes
   Total bytes transferred from matrix unit to SRAM (write): 16777216 bytes

--- Additional Information ---
SRAM Tile Size (DRAM to SRAM): Tm=512, Tk=512, Tn=512
Number of SRAM Tiles: 1
Matrix Unit Tile Size (SRAM to Matrix Unit): m=8, k=8, n=8
Number of Matrix Unit Tiles: 262144
Peak compute throughput: 512.0 OPS/cycle
Compute unit utilization: 100.00%
```