Direct Convolution: Performance Effects of Loops Ordering and Parallelization

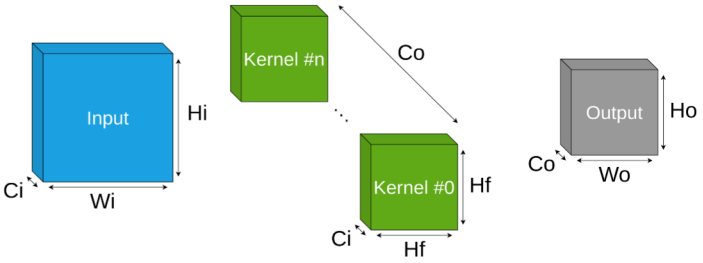
Mirco Mannino[a], Andrea Mondelli[b], Sandro Bartolini[a]

[a] Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche, Via Roma 56, 53100 Siena, Italy
[b] Huawei Technologies Research Development (UK) Ltd, Cambridge, United Kingdom

e-mail: mannino@diism.unisi.it

BACKGROUND

- Direct convolution is a technique used to address convolution that can be implemented through (at least) six perfect nested loops surrounding an accumulation operation over output elements.
- Advantages over indirect convolution methods (i.e., methods that apply transformations to tensors) [1]:
  - No memory overhead.
  - Better scaling as the number of threads increases.
- HW/SW codesign is needed to design convolutional accelerators that guarantee high performance and low power consumption.



LOOP ORDERINGS

```
for l=1 to Ho do
  for n=1 to Hf do
    for m=1 to Wf do
      for i=1 to Ci do
        for k=1 to Wo do
          for j=1 to Co do
```

Order N.1

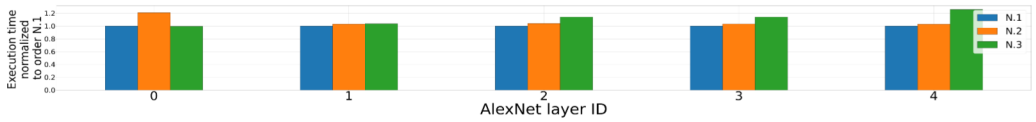
```
for n=1 to Hf do
  for m=1 to Wf do
    for i=1 to Ci do
      for i=1 to Ho do
        for k=1 to Wo do
          for j=1 to Co do
```

Order N.2

```
for l=1 to Ho do
  for n=1 to Hf do
    for m=1 to Wf do
      for k=1 to Wo do
        for i=1 to Ci do
          for j=1 to Co do
```

Order N.3

- Memory layout: HWC.
- **Order N.1** iterates over fastest dimensions of kernel and output tensors.
- **Order N.2** iterates over output tensor in the innermost loops.
- **Order N.3** iterates over all depth dimensions in the innermost loops.

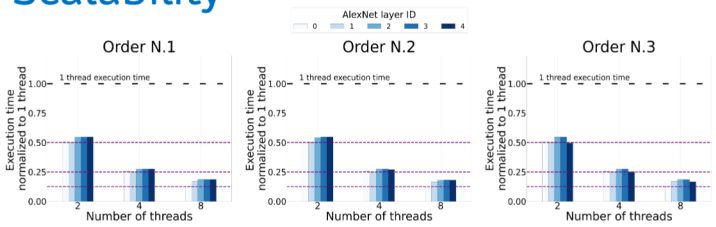


- Order N.1 shows best results on AlexNet [2] convolution layers.
- In first layers of AlexNet (low Ci values) order N.3 has similar execution time to N.1.
- In last layers of AlexNet (high Ci and Co values) order N.2 has similar execution time to N.1.

PARALLELIZATION

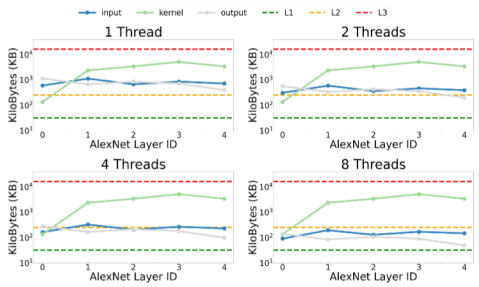
The dimension used to parallelize is the output height (Ho). Each thread has been assigned a portion of the output rows to be processed. It is possible to assign a portion of the output to each thread because every output element is completely independent from others.

Scalability



- Even if order N.1 shows best performance, the scalability with respect to the number of threads is quite the same for all the orders.
- Using 2 and 4 threads, performance doubles and quadruples respectively, using all orders.
- Using 8 threads, there is no increase of performance as in the cases of 2 and 4 threads.
- One of the problems causing non-scalability is cache usage.

Cache usage



- In AlexNet, kernel tensor has the highest memory footprint.
- Using up to 4 threads, input and output memory footprint decreases to L2 threshold. This is reflected in the doubling of performance.
- Using 8 threads, memory footprint of output and input further decreases, but performance are limited to kernel tensor (stored in L3 shared cache).

PLATFORM

- Results obtained with gem5 simulator [3] using RISC-V ISA in Syscall Emulation (SE) mode.
- Cache configuration:
  - L1: 32 KB
  - L2: 256 KB
  - L3 (shared): 16 MB

FUTURE WORK

- Use multiple degrees of parallelization.
- Investigation of other memory layouts for scalability improvements.
- Definition of ad-hoc computational and memorization units in RISC-V accelerator, based on results of this work.
- We are investigating general instructions for handling RISC-V accelerators. This work will be of crucial importance to test them.

[1] Jiyuan Zhang, Franz Franchetti, and Tze Meng Low. High performance zero-memory overhead direct convolutions. arXiv preprint arXiv:1809.10170, 2018
[2] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. In Pereira, F., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), Advances in Neural Information Processing Systems 25, pp. 1097–1105. Curran Associates, Inc., 2012
[3] N. Binkert, B. Beckmann, G. Black, S. K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D. R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M. D. Hill, and D. A. Wood, "The GEM5 Simulator," SIGARCH Computer Architecture News, vol. 39, no. 2, May 2011.