AI编译器-后端优化

循环优化

ZOMI

Talk Overview

I. AI 编译器后端优化

- 后端优化概念
- 算子计算与调度
- 算子调度优化
- Auto-Tuning
- Polyhedral

算子调度优化方法

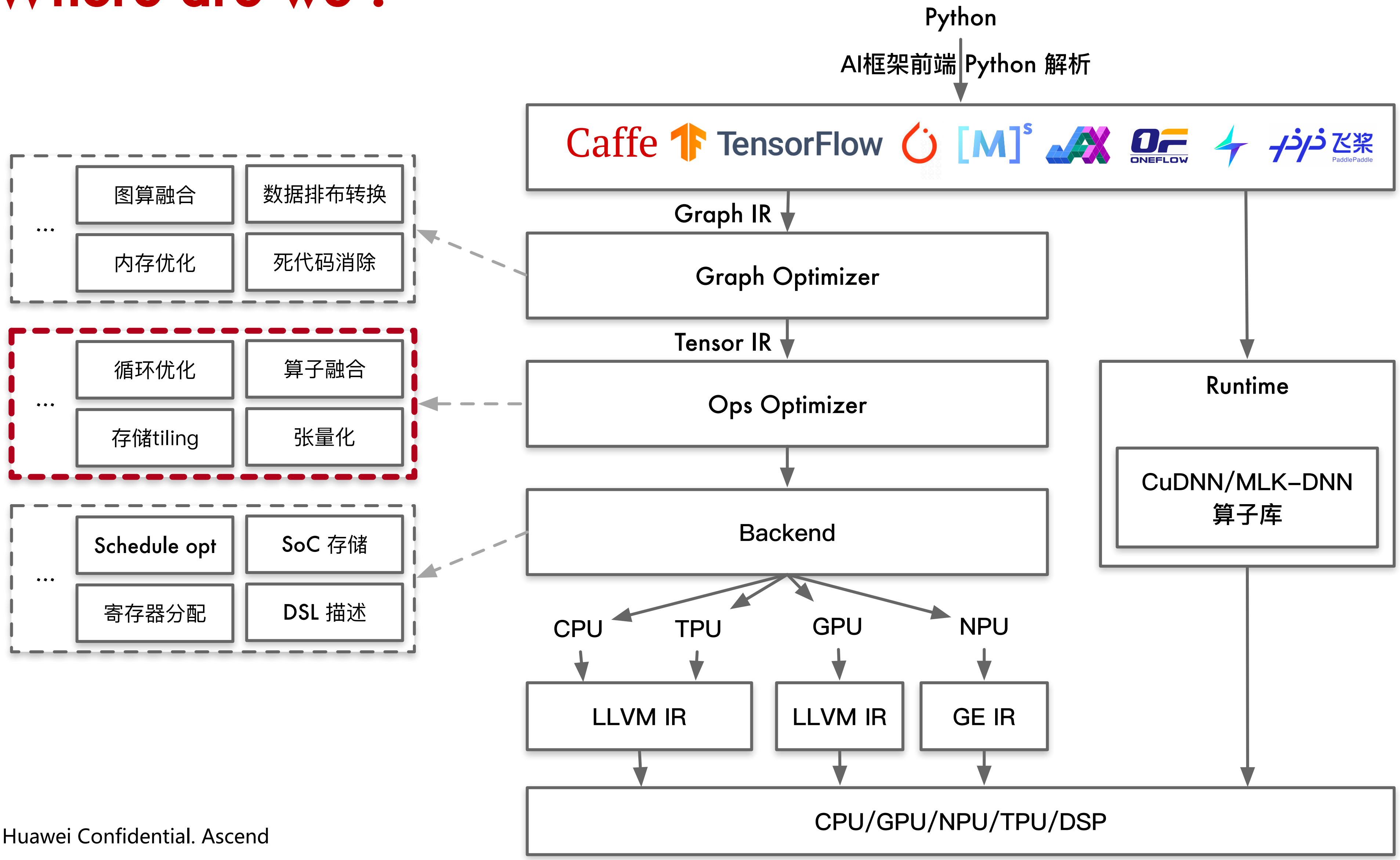
- 循环展开（Loop Unrolling）
- 循环分块（loop tiling）
- 循环重排（loop Reorder）
- 循环融合（loop Fusion）
- 循环拆分（loop Split）
- 向量化 （Vector）
- 张量化（Tensor）
- 访存延迟（ Latency Hiding ）
- 存储分配（Memory Allocation）

循环优化（Loop Optimization）

指令优化（Instructions Optimization）

存储优化（Memory Optimization）

Where are we ?


Python
AI框架前端 Python 解析
Caffe
TensorFlow
飞桨
PaddlePaddle
ONEFLOW
Graph IR
Graph Optimizer
Tensor IR
Ops Optimizer
Backend
CPU
TPU
GPU
NPU
LLVM IR
LLVM IR
GE IR
Runtime
CuDNN/MLK-DNN
算子库
CPU/GPU/NPU/TPU/DSP
…
图算融合
数据排布转换
内存优化
死代码消除
…
循环优化
算子融合
存储tiling
张量化
…
Schedule opt
SoC 存储
寄存器分配
DSL 描述

循环优化
Loop Optimization

循环展开 Loop Unrolling

- 对循环进行展开，以便每次迭代多次使用加载的值，使得一个时钟周期的流水线上尽可能满负荷计算。在流水线中，会因为指令顺序安排不合理而导致NPU等待空转，影响流水线效率。循环展开为编译器进行指令调度带来了更大的空间。

循环展开 Loop Unrolling

```

for j = 1,2 * n
    for i = 1,m
        A(j) = A(j) + B(i)
    endfor
endfor

# After Unrolling

for j = 1,2 * n by 2
    for i = 1,m
        A(j) = A(j) + B(i)
        A(j+1) = A(j+1) + B(i)
    endfor
endfor

```

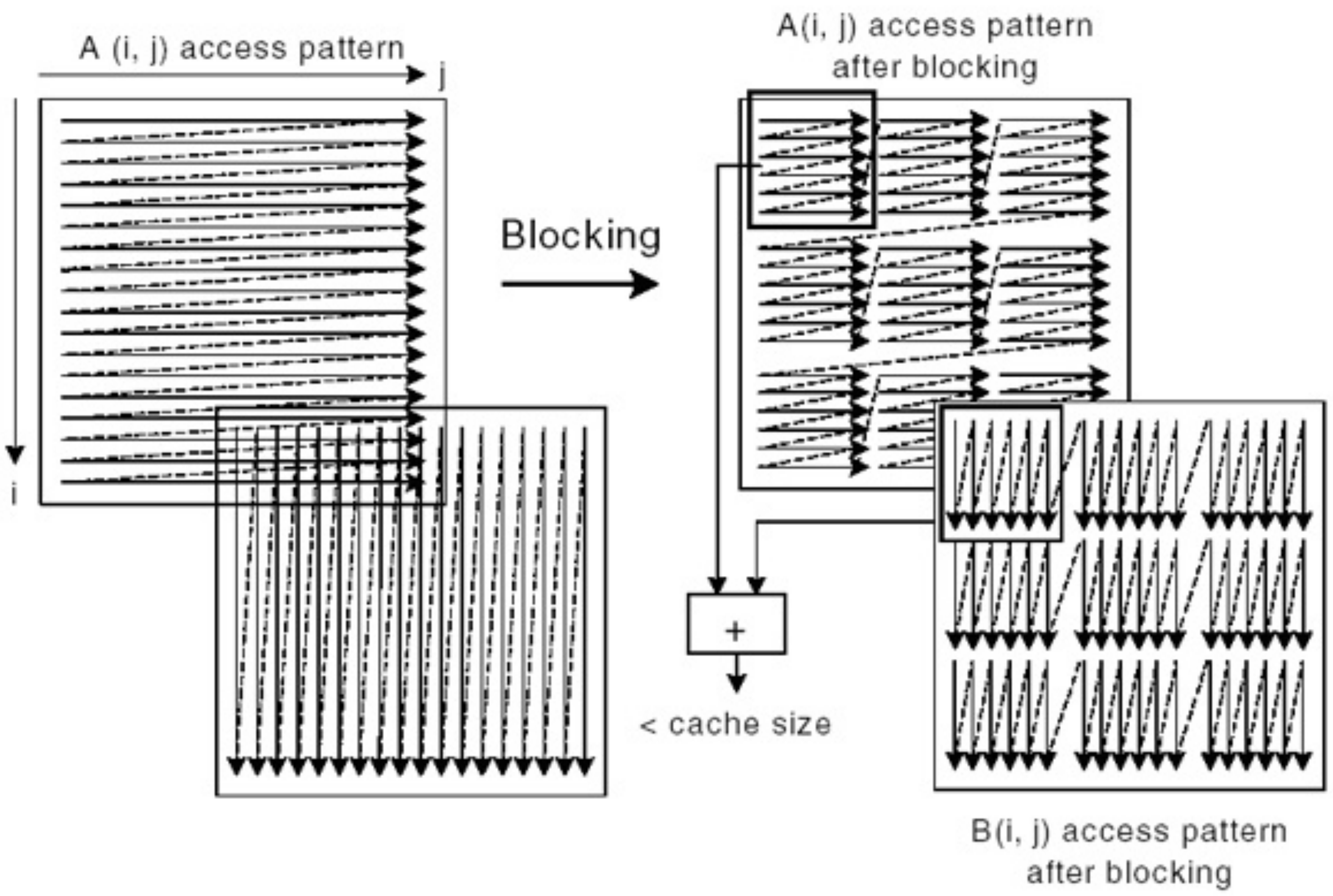
循环分块 Loop tiling

- 由于内存空间有限，代码访问的数据量过大时，无法一次性将所需要的数据加载到设备内存，循环分块能有效提高NPU cache 上的访存效率，改善数据局部性。
- 如果分块应用于外部循环，会增加计算的空间和时间局部性；分块应与缓存块一起作用，可以提高流水线的效率。

循环分块 Loop tiling

- Loop Tiling 的目的是确保一个 Cache 在被用过以后，后面再用的时候其仍然在 cache 中。
- 实现思路：当一个数组总的数据量无法放在 cache 时，把总数据分成一个个 tile 去访问，令每个 tile 都可以满足 Cache
- 具体做法：把一层内层循环分成 outer loop * inner loop。然后把 outer loop 移到更外层去，从而确保 inner loop 一定能满足 Cache

循环分块 Loop tiling



循环分块 Loop tiling

```

for j = 0, n
    for i = 1, m
        A(i) += B(j)
    endfor
endfor

# After Tiling

for j_o = 0, m, T:
    for i = 0, n:
        for j_i = j_o, j_o + T:
            A[i] += B[j_i]
        endfor
    endfor
endfor

```

Question ？

- 一般处理 CPU/GPU/NPU 都有多级缓存，Tiling如何对应到多级缓存？
- AI编译器主要是处理张量，张量的数据排布本来就复杂，人工优化到多级缓存难度高不高？

循环重排 Loop Reorder

- 内外层循环重排，改善空间局部性，并最大限度地利用引入缓存的数据。对循环进行重新排序，以最大程度地减少跨步并将访问模式与内存中的数据存储模式对齐。

循环重排 Loop Reorder

```

for i = 1, n
    for j = 1, m
        A(i,j) = B(i, j) * C(i, j)
    endfor
endfor

# After Reorder

for j= 1, m
    for i = 1, n
        A(i, j) = B(i, j) * C(i, j)
    endfor
endfor

```

循环融合 Loop Fusion

- 循环融合将相邻或紧密间隔的循环融合在一起，减少的循环开销和增加的计算密度可改善软件流水线，数据结构的缓存局部性增加。

循环融合 Loop Fusion

```
for i = 0, n
    A(i) = a(i) + b(i);
    c(i) = 2 * a(i);
endfor
for i = 1, n - 1
    D(i) = c(i) + a(i);
endfor

# After fusion

A(0) = a(0) + b(0);
c(0) = 2 * a(0);
A(n - 1) = a(n - 1) + b(n - 1);
c(n - 1) = 2 * a(n - 1);
for i = 1, n - 1
    A(i) = a(i) + b(i)
    c(i) = 2 * a(i)
    D(i) = c(i) + a(i)
endfor
```

循环拆分 Loop Split

- 拆分主要是将循环分成多个循环，可以在有条件的循环中使用，分为无条件循环和含条件循环。

循环拆分 Loop Split

```
for i = 0, n
    A(i) = a(i) + b(i)
    c(i) = 2 * a(i)
    if(temp[i] > data)
        d(i) = a(i)
endfor

# After Split

for i = 0, n
    A(i) = a(i) + b(i)
    c(i) = 2 * a(i)
endfor
for i = 0, n
    if(temp[i] > data)
        d(i) = a(i)
endfor
```

Inference

- Li, Mingzhen, et al. "The deep learning compiler: A comprehensive survey." *IEEE Transactions on Parallel and Distributed Systems* 32.3 (2020): 708-727.
- Ning, Chao, and Fengqi You. "Optimization under uncertainty in the era of big data and deep learning: When machine learning meets mathematical programming." *Computers & Chemical Engineering* 125 (2019): 434-448.
- Xu, Zhiying, et al. "ALT: Breaking the Wall between Graph and Operator Level Optimizations for Deep Learning Compilation." *arXiv preprint arXiv:2210.12415* (2022).
- Baghdadi, Riyadh, et al. "A deep learning based cost model for automatic code optimization." *Proceedings of Machine Learning and Systems* 3 (2021): 181-193.
- Chen, Tianqi, et al. "TVM: end-to-end optimization stack for deep learning." *arXiv preprint arXiv:1802.04799* 11.2018 (2018): 20.
- Loop Optimizations: how does the compiler do it? https://johnysswlab.com/loop-optimizations-how-does-the-compiler-do-it/
- Loop Optimizations: taking matters into your hands https://johnysswlab.com/loop-optimizations-taking-matters-into-your-hands/
- 编译优化之 - 通用循环优化 https://blog.csdn.net/qq_36287943/article/details/108542455
- 陈清扬编译优化，并行计算 https://www.zhihu.com/people/chenqingyang
- Loop optimization https://en.wikipedia.org/wiki/Loop_optimization

BUILDING A BETTER CONNECTED WORLD

THANK YOU