深度学习技术及应用
国家工程实验室 | 飞桨

PFCC-算子性能优化快速入门

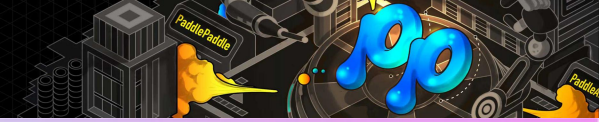
马贺达
/WintersMontagne10335
小小小菜鸟一枚~~

学习内容

- 往届PFCC会议PPT
- 第三、四期黑客松代码
- 补充的学习资料

学习成果

- 最基本的性能优化的知识
- 能够给paddle提PR[给心心][给心心]



部分一星题

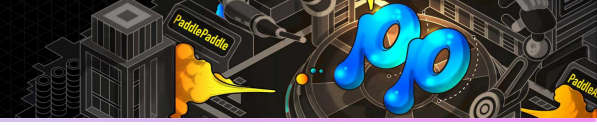
kp优化工具库

部分二星题

自己写kernel

剩余难题

GEMM优化



B站C++、Python的入门视频

- C++、Python基础知识

谭生的博客

- cuda基础知识
- 墙裂推荐！

OP Benchmark的使用

- 官方教程

核心

拓展

B站深度学习的入门视频

- 深度学习基础知识

kps优化工具库

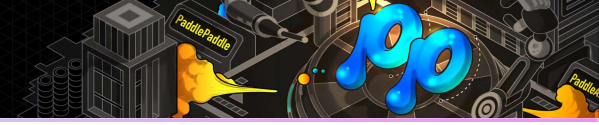
- OP Benchmark测试
  - 找到高耗时kernel，分析原因
  - 比较与竞品的性能差距
- 阅读paddle与竞品的代码
  - 如果竞品更快，找到原因
  - 如果paddle更快，也找到原因，积累优化经验
- 利用kp优化工具库实现代码优化
  - 官网的基本介绍（看懂ElementwiseAdd如何实现的）
- 模仿大佬的代码
  - 3th-33、4th-35

20 paddle/phi/kernels/gpu/erfinv_kernel.cu

```
@@ -13,9 +13,25 @@
// limitations under the License.

#include "paddle/phi/kernels/erfinv_kernel.h"
-
#include "paddle/phi/backends/gpu/gpu_context.h"
#include "paddle/phi/core/kernel_registry.h"
- #include "paddle/phi/kernels/impl/erfinv_kernel_impl.h"
+ #include "paddle/phi/kernels/funcs/elementwise_base.h"
+
+ namespace phi {
+
+ template <typename T>
+ struct ErfinvFunctor {
+   HOSTDEVICE inline T operator()(const T x) const { return erfinv(x); }
+ };
+
+ template <typename T, typename Context>
+ void ErfinvKernel(const Context& ctx, const DenseTensor& x, DenseTensor* out) {
+   ctx.template Alloc<T>(out);
+   std::vector<const DenseTensor*> ins = {&x};
+   std::vector<DenseTensor*> outs = {out};
+   phi::funcs::ElementwiseKernel<T>(ctx, ins, &outs, ErfinvFunctor<T>());
+ }
+
+ }  // namespace phi

PD_REGISTER_KERNEL(erfinv, GPU, ALL_LAYOUT, phi::ErfinvKernel, float, double) {}
```



《Performance Analysis and Tuning for General Purpose Graphics Processing Units》

- GPGPU架构
- 设计原则
- 分析、优化思路

《CUDA C编程权威指南》

- 更细致、全面的基础知识
- 流和并发、GPU加速库等

paddle内置的线程配置优化

- 简单易用
- 较优的线程配置

核心

拓展

Nsight Compute

- OP Benchmark的上位替代

其他paddle内置的资源

- warp计算优化
- 快速整型除法

第三方库

- cuBlas
- thrust

自己写kernel

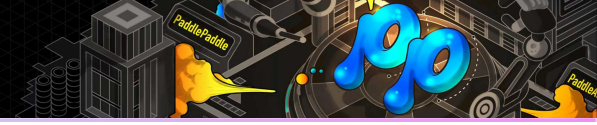
- OP Benchmark（Nsight Compute）测试
  - 找到高耗时kernel，分析原因
  - 比较与竞品的性能差距
- 阅读paddle与竞品的代码
  - 如果竞品更快，找到原因
  - 如果paddle更快，也找到原因，积累优化经验
- 书中提到的优化思路
  - 优化计算，增加并行性
  - 优化访存，依据局部性
- 模仿大佬的代码
  - 4th-33、3th-34

```
template <typename T>
__global__ void GetPoisson(
    const T* in, T* out, const int N, unsigned int seed, unsigned int offset) {
  CUDA_KERNEL_LOOP_TYPE(idx, N, int64_t) {
#ifdef __NVCC__
    curandStatePhilox4_32_10_t state;
    curand_init(seed, idx, offset, &state);
    out[idx] = static_cast<T>(curand_poisson(&state, in[idx]));
#elif __HIPCC__
    hiprandStatePhilox4_32_10_t state;
    hiprand_init(seed, idx, offset, &state);
    out[idx] = static_cast<T>(hiprand_poisson(&state, in[idx]));
#endif
  }
}

template <typename T, typename Context>
void PoissonKernel(const Context& ctx, const DenseTensor& x, DenseTensor* out) {
  const T* x_data = x.data<T>();
  T* out_data = ctx.template Alloc<T>(out);
  const int size = x.numel();
  const int kMaxBlockDim = 256;

  int block_size = std::min(kMaxBlockDim, ctx.GetMaxThreadsPerBlock());
  dim3 dim_block(block_size);
  dim3 dim_grid((size + block_size - 1) / block_size);
  phi::backends::gpu::LimitGridDim(ctx, &dim_grid);

  auto gen_cuda = ctx.GetGenerator();
  auto seed_offset = gen_cuda->IncrementOffset(20);
  uint64_t seed = seed_offset.first;
  uint64_t offset = seed_offset.second;
  GetPoisson<T><<<dim_grid, dim_block>>>(x_data, out_data, size, seed, offset);
}
```



有了琦琦的棍子大佬的专栏

- GEMM优化
- reduce优化
- ……

核心

拓展

《General-purpose graphics processor architectures》

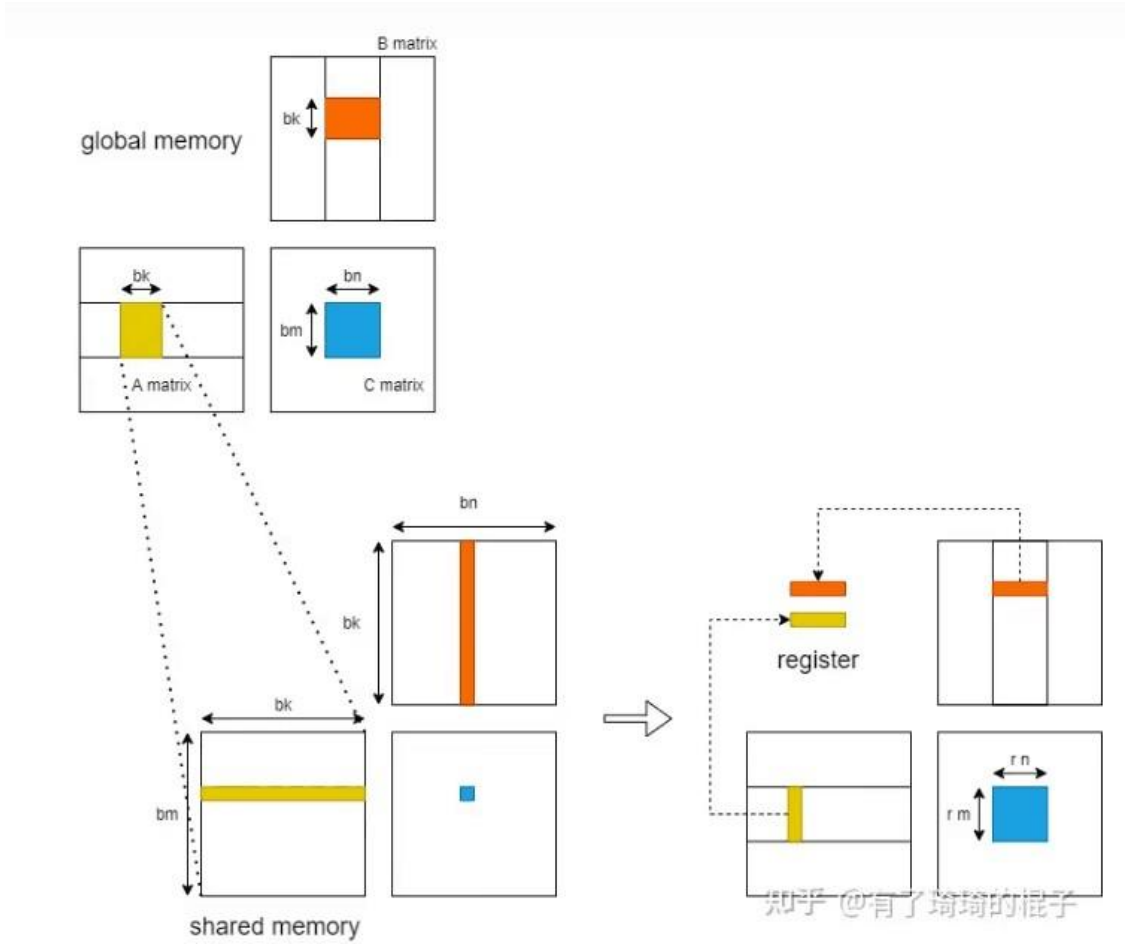
- 深入的GPGPU架构知识

B站汇编的入门视频

- 汇编基础知识

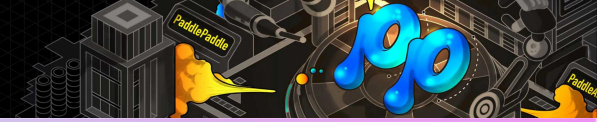
GEMM优化

- **优化方法**
  - 共享内存和寄存器
  - 数据分块
  - 数据预取与双缓冲
  - 汇编级别的优化方法
  - ……



总结

- **分析工具**
  - OP Benchmark
  - Nsight Compute
- **优化思路**
  - 阅读竞品的源码
  - 书中提到的优化思路
- **实现方法**
  - paddle内置的优化方法
  - 自己写kernel



佬们不要白嫖呜呜呜，以后请带带我！