memory time vs valgrind

JX-Ma

2023/11/24

1 实验环境

- 系统: Ubantu 22.01
- gcc version : 9.5.0
- 优化选项: -O2
- cpu:AMD Ryzen 7 6800H 3.20GHz

2 实验步骤

2.1 测试数据

- 存储类：Tensor1D
- inputTensor: 10 256 14 14
- filterTensor: 256 256 3 3
- outputTensor: 10 256 12 12
- stride : 1

2.2 实验代码

```
#define BATCH 10
#define GIGA 1e9
#define RANDOMMAX 10

struct tensorDimensions {
    int64_t batch;
    int64_t channel;
    int64_t height;
    int64_t width;
```

```
};

int main()
{
    Timer T;
    tensorDimensions input = {BATCH, 256, 14, 14};
    tensorDimensions fiter = {256, 256, 3, 3};
    int64_t stride = 1;
    Tensor1D<double> A(input.batch, input.channel, input.height, input.width);
    Tensor1D<double> B(fiter.batch, fiter.channel, fiter.height, fiter.width);
    Tensor1D<double> C(input.batch, fiter.batch, (input.height - fiter.height);
    stride + 1, (input.width - fiter.width) / stride + 1);
    int64_t operations = BATCH * 256 * 12 * 12 * 256 * 3 * 3;
    A.randomAssign(RANDOMMAX);
    B.randomAssign(RANDOMMAX);
    T.start();
    directConvolution_tensor(A,B,C,stride);
    T.stop();
    cout<<"time:"<< T.elapsed()<<endl;
    cout<<(double)operations/T.elapsed()/GIGA<<endl;

}
```

2.3 实验说明

分别使用了 time 和 valgrind 测试直接卷积和 3*3 卷积在是否开启 openmp 的内存使用量对比

3 实验结果

3.1 直接卷积

所用 Timer 所得运行时以及算出来的 gflops:
time:3.74388
gflops:0.453725
使用 time 测试的卷积结果:
运行/usr/bin/time -v 程序名

```
Command being timed: "./main"
User time (seconds): 4.22
System time (seconds): 0.06
```

```
Percent of CPU this job got: 100%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:04.28
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 168728
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 20210
Voluntary context switches: 1
Involuntary context switches: 15
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

使用 valgrind 的可视化工具结果:

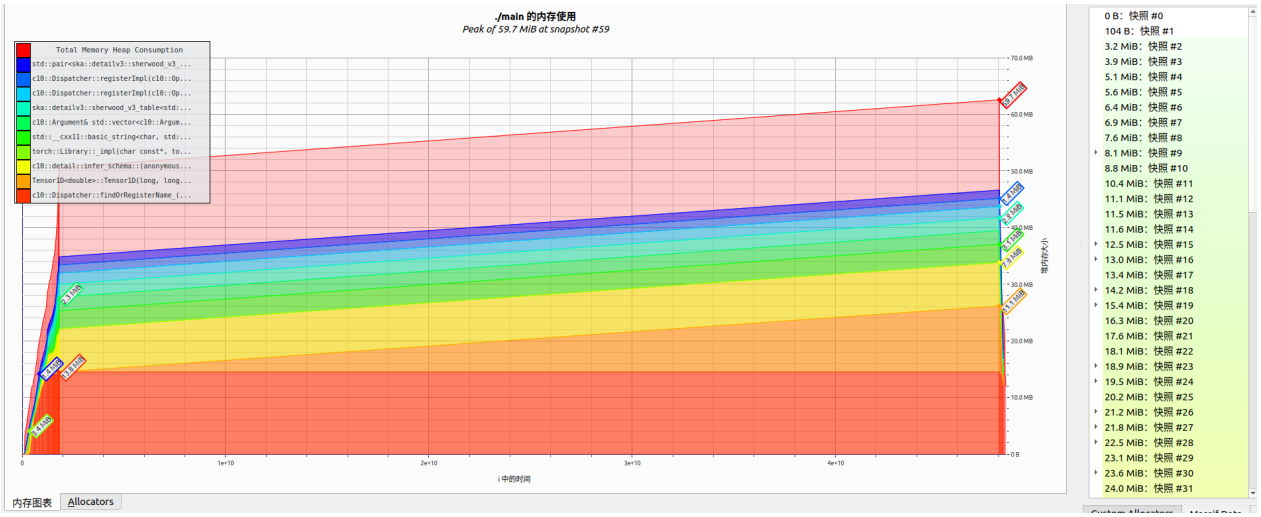

图 1: tensor1D-directConvolution

3.2 直接卷积-openmp

所用 Timer 所得运行时以及算出来的 gflops:

time:0.59608
gflops:2.84977
使用 time 测试的卷积结果:
运行/usr/bin/time -v 程序名

```
Command being timed: "./main"
User time (seconds): 5.72
System time (seconds): 0.05
Percent of CPU this job got: 621%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.93
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 170200
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 20627
Voluntary context switches: 29
Involuntary context switches: 152
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

使用 valgrind 的可视化工具结果:

3.3 3x3 卷积

所用 Timer 所得运行时以及算出来的 gflops:
time:0.559578
gflops:3.03567
使用 time 测试的卷积结果:
运行/usr/bin/time -v 程序名

```
Command being timed: "./main"
    User time (seconds): 0.79
    System time (seconds): 0.05
```

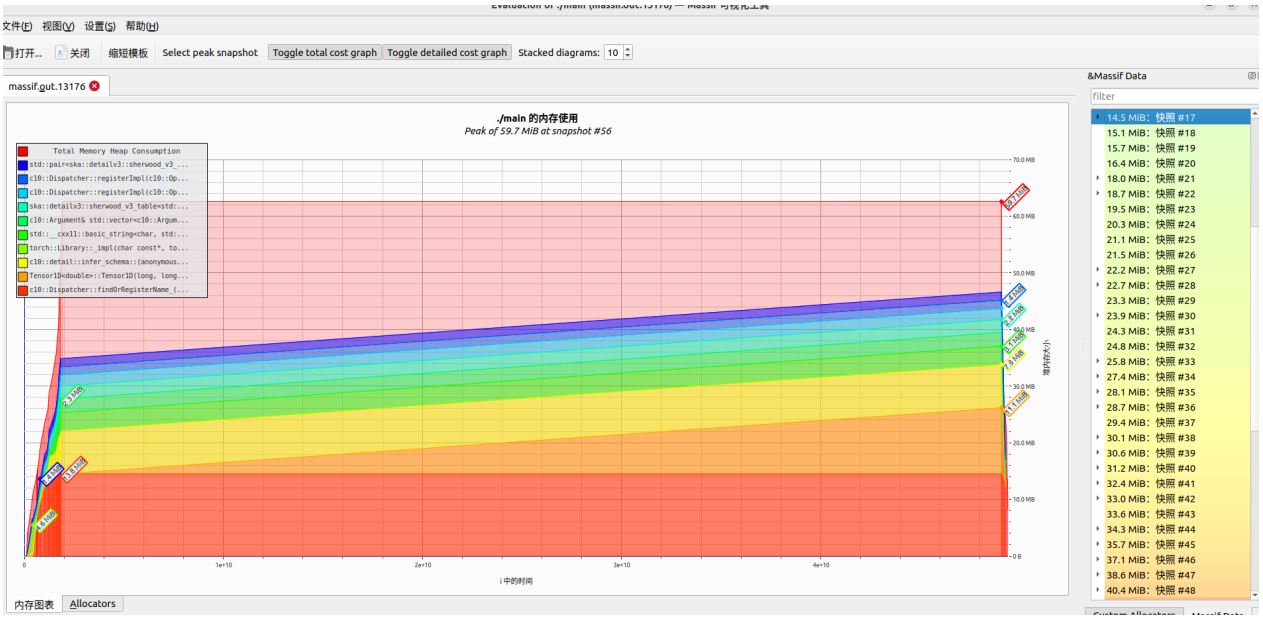

图 2: tensor1D-directConvolution-openmp

```
Percent of CPU this job got: 99%
Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.85
Average shared text size (kbytes): 0
Average unshared data size (kbytes): 0
Average stack size (kbytes): 0
Average total size (kbytes): 0
Maximum resident set size (kbytes): 168600
Average resident set size (kbytes): 0
Major (requiring I/O) page faults: 0
Minor (reclaiming a frame) page faults: 20208
Voluntary context switches: 1
Involuntary context switches: 3
Swaps: 0
File system inputs: 0
File system outputs: 0
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

使用 valgrind 的可视化工具结果:

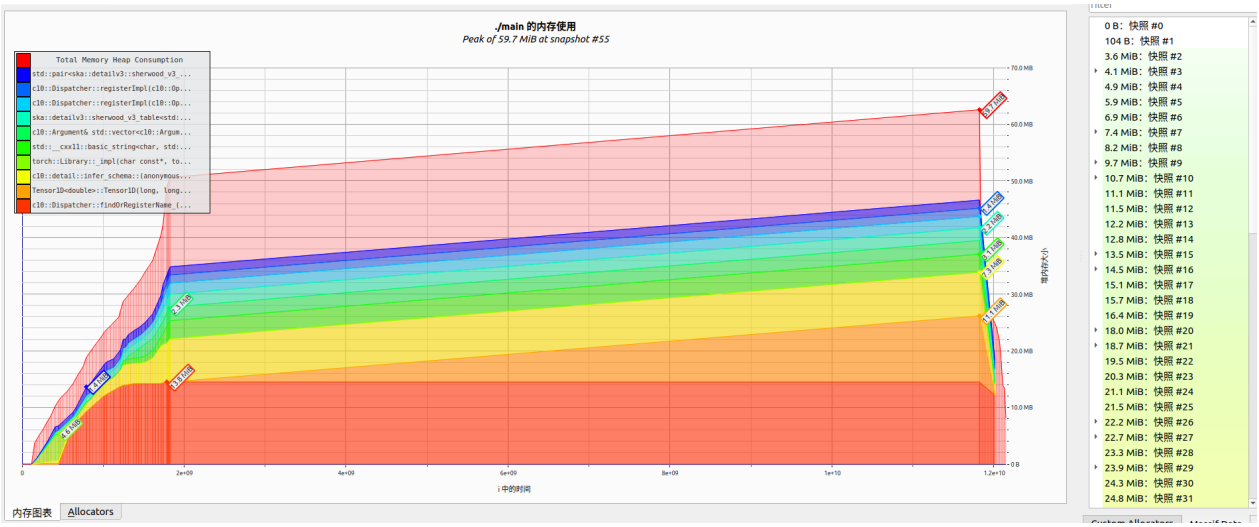

图 3: tensor1D-3x3

3.4 3x3 卷积-openmp

所用 Timer 所得运行时以及算出来的 gflops:
time:0.124319
gflops:13.664
使用 time 测试的卷积结果:
运行/usr/bin/time -v 程序名

```
    Command being timed: "./main"
        User time (seconds): 0.79
        System time (seconds): 0.05
        Percent of CPU this job got: 99%
        Elapsed (wall clock) time (h:mm:ss or m:ss): 0:00.85
        Average shared text size (kbytes): 0
        Average unshared data size (kbytes): 0
        Average stack size (kbytes): 0
        Average total size (kbytes): 0
        Maximum resident set size (kbytes): 168600
        Average resident set size (kbytes): 0
        Major (requiring I/O) page faults: 0
        Minor (reclaiming a frame) page faults: 20208
        Voluntary context switches: 1
        Involuntary context switches: 3
        Swaps: 0
        File system inputs: 0
        File system outputs: 0
```

```
Socket messages sent: 0
Socket messages received: 0
Signals delivered: 0
Page size (bytes): 4096
Exit status: 0
```

使用 valgrind 的可视化工具结果:

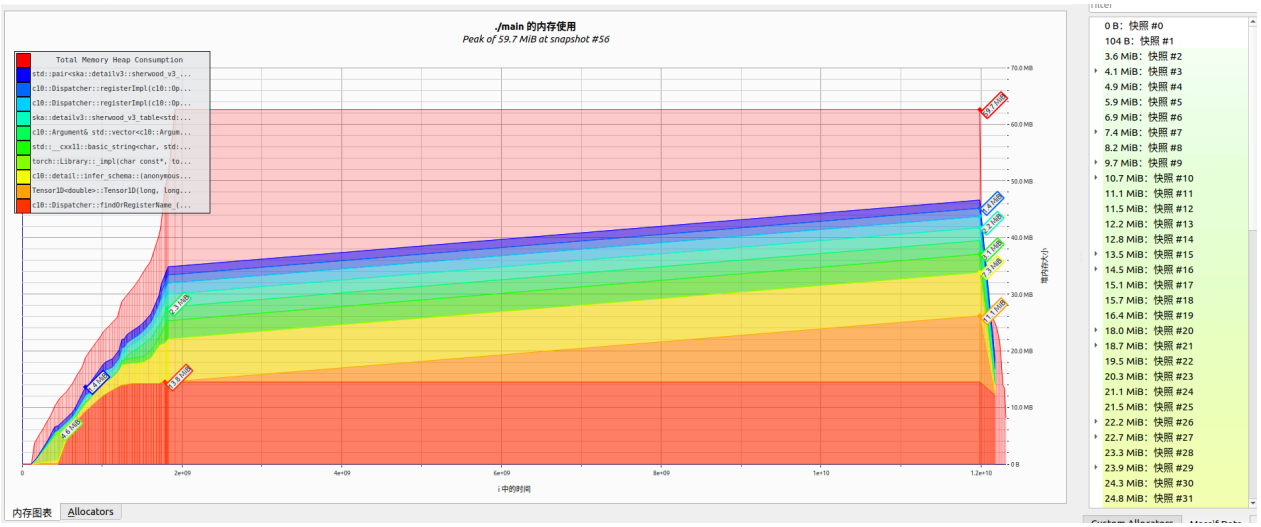

图 4: tensor1D-3x3

4 结果分析

4.1 内存汇总

表 1: Gfloat 表

类型	直接卷积	openmp	3x3	3x3-openmp
time	168.728MB	170.200MB	168.6MB	168.6MB
massif	59.7MB	59.7MB	59.7MB	59.7MB

4.2 time 参数解析

- Command being timed: 正在计时的命令，即被测量性能的程序的命令行。
- User time (seconds): 用户态时间，程序在用户态执行的时间，单位为秒。
- System time (seconds): 内核态时间，程序在内核态执行的时间，单位为秒。
- Percent of CPU this job got: 该作业占用 CPU 的百分比。

- Elapsed (wall clock) time (h:mm:ss or m:ss): 程序从开始到结束的实际经过的时间，即挂钟时间，以小时、分钟、秒的格式表示。
- Average shared text size (kbytes): 平均共享文本段大小，以千字节为单位。
- Average unshared data size (kbytes): 平均非共享数据段大小，以千字节为单位。
- Average stack size (kbytes): 平均栈大小，以千字节为单位。
- Average total size (kbytes): 平均总大小，包括共享和非共享的大小，以千字节为单位。
- Maximum resident set size (kbytes): 最大驻留集大小，表示程序运行期间所使用的最大物理内存大小，以千字节为单位。
- Average resident set size (kbytes): 平均驻留集大小，表示程序运行期间所使用的平均物理内存大小，以千字节为单位。
- Major (requiring I/O) page faults: 产生的主要页面错误数，表示由于需要 I/O 操作而导致的页面错误次数。
- Minor (reclaiming a frame) page faults: 产生的次要页面错误数，表示由于内存回收而导致的页面错误次数。
- Voluntary context switches: 自愿上下文切换次数，表示程序主动切换到另一个进程的次数。
- Involuntary context switches: 非自愿上下文切换次数，表示由于资源争用或中断等原因而被迫切换到另一个进程的次数。
- Swaps: 交换次数，表示发生的页面交换次数。
- File system inputs: 文件系统输入次数，表示从文件系统读取的次数。
- File system outputs: 文件系统输出次数，表示向文件系统写入的次数。
- Socket messages sent: 发送的套接字消息数，表示发送的套接字消息数量。
- Socket messages received: 接收的套接字消息数，表示接收的套接字消息数量。
- Signals delivered: 传递的信号数，表示传递的信号数量。
- Page size (bytes): 页面大小，表示操作系统中的页面大小，以字节为单位。
- Exit status: 退出状态，表示程序的退出状态码。0 表示正常退出，非零值表示异常退出。

4.3 valgrind

- Total memory Heap comsumption: 堆内存总量
- std::pair<T1,T2>: 用于表示两个不同类型的值的组合。它提供了一种简单的方式来存储和操作这种键-值对
- c10::dispatch::registerImp:PyTorch C++ 代码中的一个函数，用于在分派表中注册实现函数。
- Tensor1D<double>::Tensor1D<long,long,long,long>: Tensor1D 所占的内存空间

5 工具获取

5.1 time

ubantu 系统自带，使用方法，/usr/bin/time -v ./main(程序名称)
环境变量中的 time 和/usr/bin/time 不是同一个命令

5.2 valgrind

工具安装：sudo apt install valgrind
验证: valgrind –version
massif 可视化工具安装: sudo apt-get install massif-visualizer
使用方法：1. 先使用 valgrind –tool=massif ./main(程序名称) 生成 massif 文件，一般文件名为 massif.out. 进程号
2. sudo massif-visualizer massif.out. 进程号 (生成的 massif 文件名)