软硬件协同设计的一点思考和展望

https://zhuanlan.zhihu.com/p/397697428

None Sun Aug, 08 21:53

最近工作告一段落，有些感慨。总结和反思这段工作的得与失，我还是想简单分享一下我的一些思考。今天也没什么干货，就是聊聊心得吧！有些表述不怎么准确，纯属抛砖引玉，欢迎讨论~

第一个感想：**软硬件的技术隔阂比我想象更深**

我最初的出发点是用自动调优在软件性能优化和硬件设计间建立直接联系，实现更高效的软硬协同。与通常自动调优不同的是，我要基于一些硬件设计参数去建模、计算程序性能，而不是用真机或机器学习的黑盒模型。因为这样才能获得硬件设计参数的直接反馈，比如哪些参数组合是均衡的，哪些存在瓶颈且瓶颈在哪。我当时的几个判断：从改硬件设计、kernel优化、瓶颈分析再回到硬件设计，这个反馈链太长，互动效率往往不高；手动优化汇编是落后生产力，工作量大扩展性差；算力比人力更有竞争力，机器自动搜索比人力搜索适用性更强；问题复杂度增加时，算力比人力更容易扩展；等等。所以我觉得完全可以在现有编译器的基础上，只加入新功能的简单支持，然后用上层语言最多是IR实现自动调优。但是，从现在看来，实际实施上还是存在很大偏差。

碰到的第一个问题：一些硬件功能，设计时就没仔细考虑软件用法，毕竟硬件工程师了解具体使用场景和流程的人是极少数。这其实是架构师的活，但架构师的理解也会有许多偏差，也会有很多想当然的时候。这导致一些功能封装起来极为费劲，甚至压根就很难用起来。应该说每个功能的软硬件对接都需要慢慢调试，没有几代产品的相互磨合，是很难做好的。这个问题我觉得还是需要有人既懂软件应用，懂一点编译器，也懂一点硬件架构，承担起技术串联的角色。这样才能打破技术壁垒，更流畅的实现从底层硬件到上层软件的功能输送。

第二个问题，编译器的后端优化自动化程度还远远不够，非常费时费力。以前我主要用x86和CUDA的编译器，所以想当然的以为只要调整调整源代码，再加上一些编译选项，大部分代码都可以有不错的性能。这也是当前很多auto-tuning的工作方式。但我还是盲目乐观了。x86和CUDA的编译器经过多年打磨，已经非常成熟。那些看起来很理所当然的优化，其实都是大量人力投入换来的，是无数前人的心血结晶。如果要支持新的ISA或硬件架构，大概率需要自己去重新实现优化。可如果每个新的逻辑都需要去改后端，我想要的软硬件自动化连接就失去了意义。我中途试图越过编译器直接将auto-tuning做在汇编上，做一些基于declarative规则的优化，然后发现自己就变成了后端…… LLVM支持的后端也很多，但除了tablegen，其中有多少优化能自动化支持？我也调研过类似Code generator generator或是Compiler Compiler这种逻辑，功能很有限，所以后端还是靠各家自己慢慢磨。我感觉后端很多优化算法和实现上是相通的，但当前没看到太多自动化的机制，工作量还是很大的。MLIR里有Table-driven Declarative Rewrite Rule，不

过似乎自动化程度也有限，不知道将来能发展成什么样。据我了解，TVM里自动调优多数在LLVM以上，硬件后端优化估计也不是重点。所以，后端优化这个问题，短时间真是绕不过去的一道坎。

第三个问题，就是硬件建模还是有很多看不清的工作量。其实对于一些简单的算子（比如GEMM，CONV这种），算法直线，实现流程也简单。所以哪些硬件参数有哪些影响，还是大致有谱的。但像cache hit rate的估计，就有些玄妙。逻辑上是有统计规律可循，但严重偏离规律的情况也时有发生。这种情况，说我算的不准我也认，但我要说硬件设计上有坑，也绝不冤枉他们。不过我倒是觉得，软硬协同的价值也就体现在这些地方，提前对接有助于暴露问题。硬件设计有它的规律，多数时候都是软件去迁就硬件。软件一要补硬件的坑，二要尽量发挥硬件能力。相对说来，软件更新迭代成本小一些。所以用软件去补足硬件短板，也算是本职工作之一了。

第二个感想：**软硬件的分工和协作是协同设计的难点**

我以前很爱看斗地主节目。有解说嘉宾说，斗地主最难且最有艺术感的，不是“压死你”，而是“要不起”。本来能“压死你”，我却选择“要不起”。这“让牌”的艺术，可不是打假赛，反而是全局视野的一种精妙体现。让过对方的一些牌，保存自身实力，同时让对方防守牌打掉而漏出破绽，从而为自己的进攻牌过关创造条件。

软硬件设计不是竞技，但有时候能做的事情选择不做，也有很多值得细品的妙味。上次专栏文章聊指令集设计的时候已经提过一些硬件设计上能做而不做的例子。这里再分享一个软件实现的例子。CUDA一些函数实现的corner case里会用到local memory，即使GPR充裕也不转GPR。因为用local memory，只要不进到这个corner分支，那就几乎没有开销（最多费一点指令cache）。而如果用GPR，那即使不进这个分支，也会影响总GPR数，从而影响Occupancy。也就是说只要用了额外的GPR，不管这个代码有没有运行，它都有开销，即使有时候是无害的。

NV的硬件运行逻辑上也有很多限制，比如64bit或128bit访存指令的地址必须对齐，操作64bit和128bit的GPR也必须对齐。很容易找到一些case证明这些限制会影响性能，但它就是没做。NV也有一些很常见的指令不支持（比如SQRT），也有些指令走了又回来了（比如IMAD，比如VABSDIFF*，PTX一直有这个指令，大家有兴趣可以比较各代生成代码的差异）。所以，很多功能的做或不做，肯定还是有诸多考虑的。我感觉，如果是软件上一般能cover又对性能损失不明显的操作，硬件上它就尽量不弄了。毕竟每个硬件功能都是有基础开销的。而软件实现只有真的用上了，才有开销。还有另一种，NV的一些调度逻辑（比如control codes），也从硬件转移到了软件。这个应该是因为编译器具有更全局的视野，可以更好的做依赖分析和指令调度，同时也简化硬件设计，节省面积。

还有一些功能，估计是直接舍弃了。比如我没见过NV算术异常处理的功能（比如除0，溢出之类），L1 Cache之间也没有coherence。它们很难用软件找补回来，相当于就是没有了。不做算术异常应该是价值有限，没有coherence主要还是运行模型的影响。那这些就算是从软件和硬件角度都不支持了（虽然有些也可以用非常规的软件方法bypass）。

功能上哪些选择硬件做，哪些软件做，哪些都不做，怎么去选择和评估，还是有很多很开放的问题。有些可能都不一定是技术问题，而是市场问题。每个你买的芯片都有很多你从来没用过的功能，你可能觉得花了冤枉钱。可是，如果不做这些功能，可能就没有这么多目标客户来帮你分摊硬件开发成本，说不定你的芯片还是会涨到这个价钱。

第三个感想：**软硬件的信息传递和协同优化密切相关**

一个应用，从算法开始，到软件实现，再到编译的中间表示，再到机器指令，都是信息一级一级传递的过程。协同优化，同时也是信息交互的过程。比如最常见的矩阵乘法，在算子层可以用标准的实现，也可以做快速矩阵乘法（比如Strassen），如果知道是卷积或是其他特殊矩阵，那可以有特定的加速算法。普通的矩阵乘算法，有相应的矩阵乘加速器或是tensorcore，也可以做相应的加速。如果是矩阵连乘或是可融合的操作，在上层也容易实现相应的优化。可如果矩阵乘法已经被改写为直接的多重循环的C实现甚至是汇编，要识别和加速它就困难了。信息的传递是top-down，越往下视野越小。因而优化的选择应该也是top-down，我觉得这也是MLIR想解决的一个点吧！

但是，信息的传递也存在很多的困难。不管是C还是LLVM的IR，都会有一些硬件操作难以表述，最终都变成了内置函数（intrinsic functions）。内置函数多了，上层表示就难以理解和优化，最后都会堆积到硬件后端，这其中肯定会损失一些可能的优化。LLVM主分支的内置函数现在估计也有几千个了，将来肯定还会再不断增加。IR的表述能力到底应该怎么扩展，确实还是个问题。前面也提到各个后端里有不少相通的算法逻辑，如果上提到IR端，应该是可以减少些工作量的。不过，这类问题不管是分是合，都是挺复杂的系统工程，好的抽象逻辑真的是挺难做的。

然后再说指令集的问题。什么样的指令集方便优化呢？显然，每个指令一样慢最好优化，只要没冗余就算是优化好了。但这显然与设计追求不合。我认为好的指令集应该具有表述能力强（一个功能有多种独立实现），独立性好（类似函数式逻辑，相互干扰少），资源瓶颈少（比如各种barrier，carry flag都有多个选择，不会导致相关操作序列化）等特点。从上层看，就是指令调度和算术优化的可能性比较多，自由度大。另外有一点就是对信息的具体化能力比较强，比如不同的依赖关系可以用不同的scoreboard保证次序，减少干扰。再比如以前提到的IMAD可以接收MOV、SHL、IADD这种具体的modifier，从而对一些更具体的信息有相应的运行优化。包括control codes这种逻辑，都是指令对一些具体化信息处理能力的体现。但是这些也会显著地增加指令集的复杂度，给编译优化带来更大的压力，这也是我觉得后端需要更多的自动化优化机制的原因之一。

早期的指令集很多是面向人设计，方便人来写程序，将来也许可以更多的面向编译器设计指令集。毕竟指令集的主要目的是连接软硬件，而不是让人看懂或者写起来方便。编译器用起来方便就足够了。将来时机成熟，指令集设计也许可以机器自动做，至少是辅助设计一部分吧！这样软硬件的功能连接自由度就更高了。硬件EDA其实也有编译器，它是不是也可以接进来呢？上层的深度学习编译器，传统的编译器，EDA的编译器，有没有可能实现从人脸识别到晶体管的全线自动调优？嗯……算是对未来的一点点期待吧~

最后，感慨一下…… 硬件行业是我见过的技术惯性第三大的行业（第二就是医疗行业了），改啥都贼费劲。感觉真心耗不起，我要闪了，回去做算法去吧~ 祝各位关注我的硬件boy们好运~

发布于 10 小时前