AI???????

????

1. 信息维度映射，信息过滤，信息选择 : Dot / GEMM => Join+Reduce
2. 激活 : ElementWise Broadcast
3. 统计、动态选择、排序 : Reduce 、G/S
4. 随机数生成
5. 是不是可以把所有的计算都抽象成查找表的方式
    1. 操作的方式由操作码 +密码数来指定，实际上是定义一个查找表
    2. 操作码可以是指令的 imm，密码数可以是约定的，也可能是提前载入的
    3. 乘法等价?
    4. ```
       /* emulate GPU's LOP3.LUT (three-input logic op with 8-bit truth table) */
       uint32_t lop3_fast (uint32_t a, uint32_t b, uint32_t c, uint8_t ttbl)
       {
           uint32_t r = 0;
           if (ttbl & 0x01) r |= ~a & ~b & ~c;
           if (ttbl & 0x02) r |= ~a & ~b &  c;
           if (ttbl & 0x04) r |= ~a &  b & ~c;
           if (ttbl & 0x08) r |= ~a &  b &  c;
           if (ttbl & 0x10) r |=  a & ~b & ~c;
           if (ttbl & 0x20) r |=  a & ~b &  c;
           if (ttbl & 0x40) r |=  a &  b & ~c;
           if (ttbl & 0x80) r |=  a &  b &  c;
           return r;
       }
       ```

???????

1. 一堆向量和另外一堆向量 交叉 计算相关度
    1. 存内计算只是保持一支数据更靠近计算单元
2. 一堆向量的变换

???????

??

1. 电路模块不能太大，所以要大量的相同功能进行组合
2. 组合带来了复杂的判断逻辑，需要流水化提高 throughput
3. 流水化带来的 latency需要用复杂的同步机制
    1. 微观上说，各种 scoreboard， forwarding，多级调度，都是在控制 pipeline的精密配合
    2. 宏观上说， mailbox， barrier是在更高的维度进行同步
       ```

4. 复杂同步对软件和应用有一定的要求
5. 以上几个因素根据不同的设计都有非常大的弹性，整体的方案设计难度大
6. 各种硬件 (比如 roq， sram读写口等)的数量是有限制的，不像软件视角不感知这个冲突。
7. 各种操作，运算都是有延迟的，都不是实时单拍完成的

????

1. 存储和传输
    1. 大量，大力度的并行搬运数据
    2. 各个级别的速度不一样
    3. 各个级别的暂存容量大小不一样
2. 并行计算
    1. 大量的充分的并行计算
    2. 大量的数据复用
    3. 卷积，矩阵乘
3. 同步

????????????????

??/???

1. 难以做复杂的算法（调度、分析、统计）
2. 难以做逻辑深度比较深的计算
    1. latency的长和不确定，模块间的时间不确定性
    2. 需要大量实现 “异步 ”逻辑 ,消耗大量的面积
    3. 需要很大的 buffer存储用于状态 footprint,消耗大量的面积
    4. 增加外部控制的复杂度，
    5. PC需要对所有的操作异步化，降低实时性
    6. 为了高频、增加很多级流水线
        1. 浪费大量用于同步的逻辑
        2. 容易撞到功耗墙
        3. latency的延长和调度不确定性

??/??

1. 并行任务非常适合，适合简单的、重复的、大量的计算，比如 MMU
2. 带宽足够，可以做大量的并发处理
3. 实时性好，可以做到 cycle级别的调度
4. 可以利用数字电路表达的特点加速
    1. 对 2的幂次的乘除运算
    2. 固定算法的 hash， mmu等处理
    3. 查找表计算: cuda的 LOP3.LUT指令
    4. 通过软件的静态预先计算，最大化压缩硬件需要处理的表达空间

Revision #7
Created 2025-03-02 06:28:15 UTC by Colin
Updated 2026-04-29 07:33:20 UTC by Colin