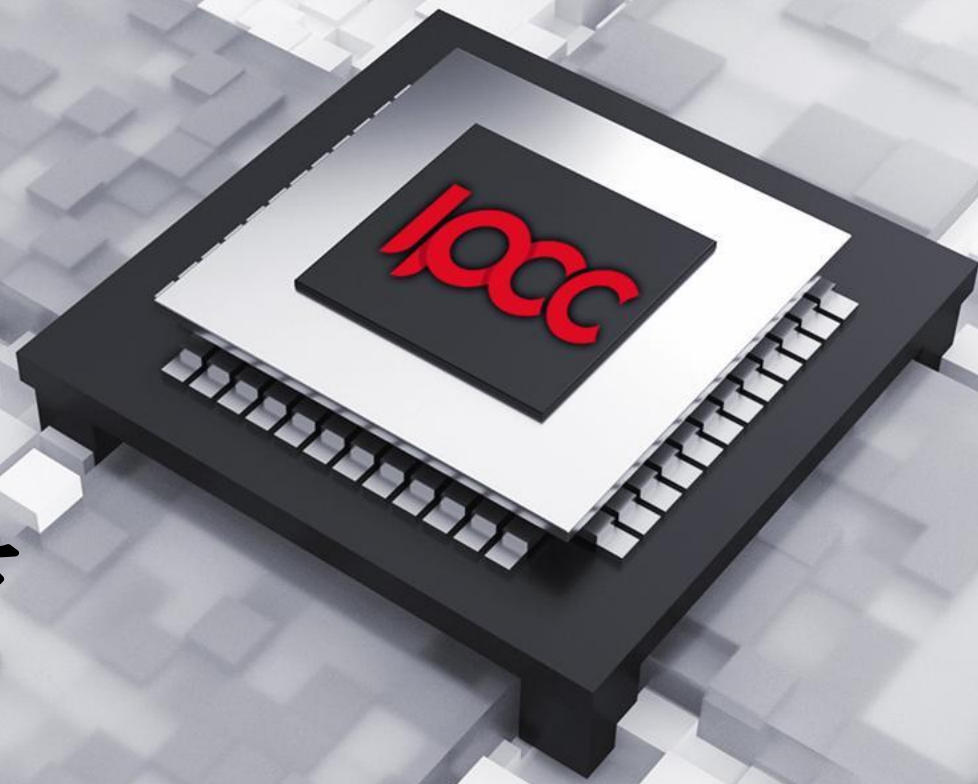


超算竞赛备赛指南

ACM 中国-国际并行挑战赛-赛前讲座2022

第 1 讲 / 共 6 讲

i@wu-kan.cn
https://wu-kan.cn

中山大学超算队 吴坎

中山大學 SUN YAT-SEN UNIVERSITY 国家超级计算广州中心 NATIONAL SUPERCOMPUTER CENTER IN GUANGZHOU

本届讲座大纲

Association for Computing Machinery

ACM中国 国际并行计算挑战赛

主讲人：吴坎

- **第一讲：超算竞赛入门指南**
  - 直播时间：05月26日 周四
- 第二讲：超算基本原理与方法论
  - 直播时间：06月01日 周三
- 第三讲：现代处理器优化概论
  - 直播时间：06月08日 周三

主讲人：叶子凌锋

- 第四讲：并行开发技术与优化方法 I
  - 直播时间：06月17日 周五
- 第五讲：并行开发技术与优化方法 II
  - 直播时间：06月24日 周五
- 第六讲：并行优化常用方法与工具
  - 直播时间：06月30日 周五

往期分享回顾

更多学习资料与往届讲座分享，尽在比赛交流群 1046805935！

学习平台：哔哩哔哩[超级云讲堂]

- 第一讲：并行计算基础概论 | 主讲人：邵奇
- 第二讲：性能优化方法论 | 主讲人：赵雄君
- 第三讲：并行开发技术概论 | 主讲人：张力越
- 第四讲：现代处理器优化技术 | 主讲人：邵奇
- 第五讲：并行优化常用工具 | 主讲人：邵奇
- 第六讲：并行优化实战技巧 | 主讲人：张力越

- 参赛之前
  - 超算竞赛（PAC、IPCC、CPC、ASC、PRA 等）介绍与 IPCC 参赛流程
  - 超算竞赛（IPCC 等）与传统的程序设计竞赛（ICPC 等）比赛的区别？
  - 应当如何选择适合的队友？
- 比赛期间
  - 当我不知道该做什么的时候，应该去做什么？（如何获得优化思路？）
  - 当我知道该做什么的时候，应该怎么做？
- 赛后总结
  - 如何组建一支可以延续的超算竞赛团队？

个人简介

- 中大超算队（前）队长
  - IPCC'21 rk3（初赛 & 决赛）
  - PAC'20 rk2
  - CPC'21 rk3
  - ASC'20-21 rk3 + Highest Linpack
  - PRA'20 rk3（最短路径赛道）+ rk5（LU 分解赛道）
  - 可能是第一个（一年内）在国内所有超算比赛都获得前三的选手？
  - CCF-CSP Top 0.05% in 11395
- 研究方向： sw-hw co-design; GPU arch

IPCC 介绍

Association for Computing Machinery

ACM中国 国际并行计算挑战赛

竞赛概述

- **目标国家/地区**

--中国（含港澳台）、海外

- **主要专业**

--计算机科学与技术

--数学/物理/航空/…

--研究生/本科生/在职不限

- **任务需求**

--组成原理、计算机网络

--数值计算、流体力学

--并行计算、高性能计算

--C/C++/Fortran

应用优化组

- **赛题设置**

--以行业真实应用或科研任务为出发

--专业领域：航空航天、工业大数据、城市管理等，具体领域不限

- **竞赛任务**

--程序分析→算力发掘→程序优化→实时调优

--计算速度优化

--上机（决赛）与技术答辩

赛程总览

- **即日起 7 月 15 日**

--报名及组队

--赛前培训（By往届获奖选手及特邀讲师）

- **7 月至 8 月**

--初赛任务发布及说明会

--初赛评审会（线上）

- **8 月至 9 月**

--总决赛任务发布及说明会

--总决赛（线下）评审会

竞赛福利

- **报名奖励**

--北京超级云计算中心账号

--3000 核时/人

- **奖项设置**

--初赛/总决赛一、二、三等奖

--双语证书、奖牌

--奖金池 22 万元

- **其他福利**

--进入合作单位人才库

--行业会议及特色活动

ACM中国
高性能计算专家委员会

AMD

北京超级云计算中心
BEIJING SUPER CLOUD COMPUTING CENTER

i@wu-kan.cn
https://wu-kan.cn

中山大学超算队 吴坎

中山大學 SUN YAT-SEN UNIVERSITY

国家超级计算广州中心 NATIONAL SUPERCOMPUTER CENTER IN GUANGZHOU

IPCC 介绍


ACM中国-国际并行计算挑战赛
ACM-China International Parallel Computing Challenge
超级云计算
教育基金奖


全国三等奖
全国二等奖
全国一等奖


示例：中国科学院深圳先进技术研究院
指导老师：XXX
团队成员：XXX、XXX、XXX
2020年11月4日参加第一届ACM中国-国际并行计算挑战赛全国总决赛
（应用优化赛道），荣获
一等奖
ACM中国-国际并行计算挑战赛
2020年11月
CERTIFICATE
OF APPRECIATION
EG.HUNAN UNIVERSITY
ADVISORS:XXX、XXX
CONTESTANTS: XXX、XXX、XXX
GROUP: Application Optimization
ACM（CHINA）-International Parallel Computing Challenge（National Finals）in 2020
First Prize
-The team is required to have the following SKILLS-
ACM-China International Parallel Computing Challenge
11/2020

官网注册通道

组委会联系方式

- 官网：www.paraedu.org.cn
- 邮箱：ACM_IPCC@163.com
- 电话：18310726311（余老师）
- 微信公众号：
  - 北京超级云计算中心(BJBLSC)
- 交流群：
  - 1046805935（参赛选手）
  - 1095416620（指导老师）

IPCC'21 初赛

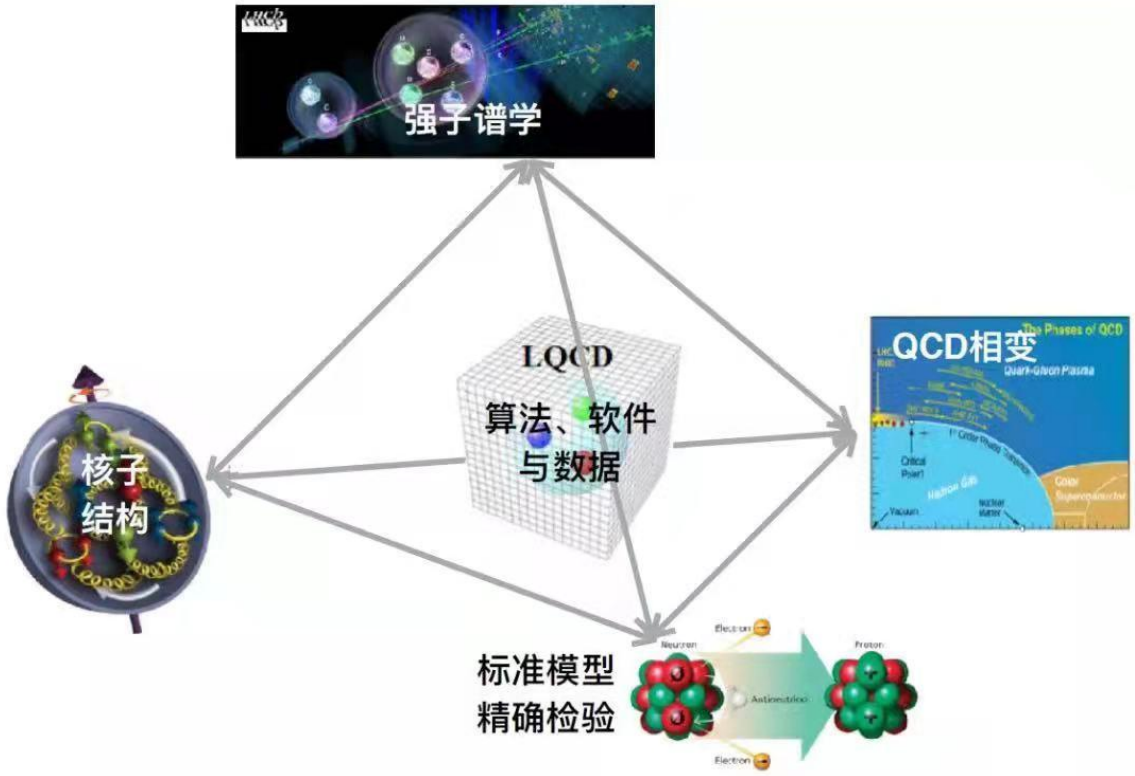
- SLIC 超像素算法优化
  - 将图像的像素依据某种相似性进行聚类，形成一个大“像素”，这个大像素可作为其他图像处理算法的基础。
  - 算法核心内容包括 RGB-Lab 空间转换、像素点数据结构重构、像素点邻域计算、迭代更新聚类中心等等。

IPCC'21 决赛

- 格点量子色动力学 LQCD 稀疏线性系统求解优化
  - 其计算热点是对四维超立方格子上的一个稀疏线性系统 Mx=b 的求解。
  - 主要难点在于对 dslash 算子的数学形式、程序框架的综合理解以及对迭代优化算法的运用以及对格点分配方式参数与优化技巧适配的优化。

IPCC 历届赛题回顾

更多学习资料与往届讲座分享，尽在比赛交流群 1046805935！



格点量子色动力学 LQCD 稀疏线性系统求解优化

SLIC 超像素算法优化

i@wu-kan.cn
https://wu-kan.cn
中山大学超算队 吴坎

IPCC'20 初赛

- 9-point stencil 算法优化
  - 矩阵计算中的基础算法。
  - 从输入的图像中读取像素点信息然后对非边界网格进行加权模糊计算，从而得出结果像素点并生成输出图像。
  - 该算法可从图像计算延伸到各种矩阵数值计算领域中。

IPCC'20 决赛

- 三维超声速欠膨胀射流数值模拟优化
  - 给定入口总温、增压比及马赫数，使用大涡模拟计算柱坐标下的非定常射流流场参数。
  - 计算使用的空间差分格式为 7 阶 weno 格式（对流项）和 6 阶中心差分格式（粘性项），时间推进格式为四步三阶的 Runge-Kutta 法。

IPCC VS ICPC

IPCC

- 比赛时间中等（五周）
- 可以对赛题进行充分优化
- 软硬件结合优化
  - 解决现实问题，贴合业界需求
- 有上手门槛，需要计算资源
- 算法层面优化，降维打击！

ICPC

- 比赛时间短（五小时）
- 易受发挥影响，结果不稳定
- 只关注软件（算法）设计
  - 通常只关注单线程算法
- 上手难度低，刷题网站多
- 掌握硬件知识，常数更小！

并不相互对立，可以互相促进！

如何分工

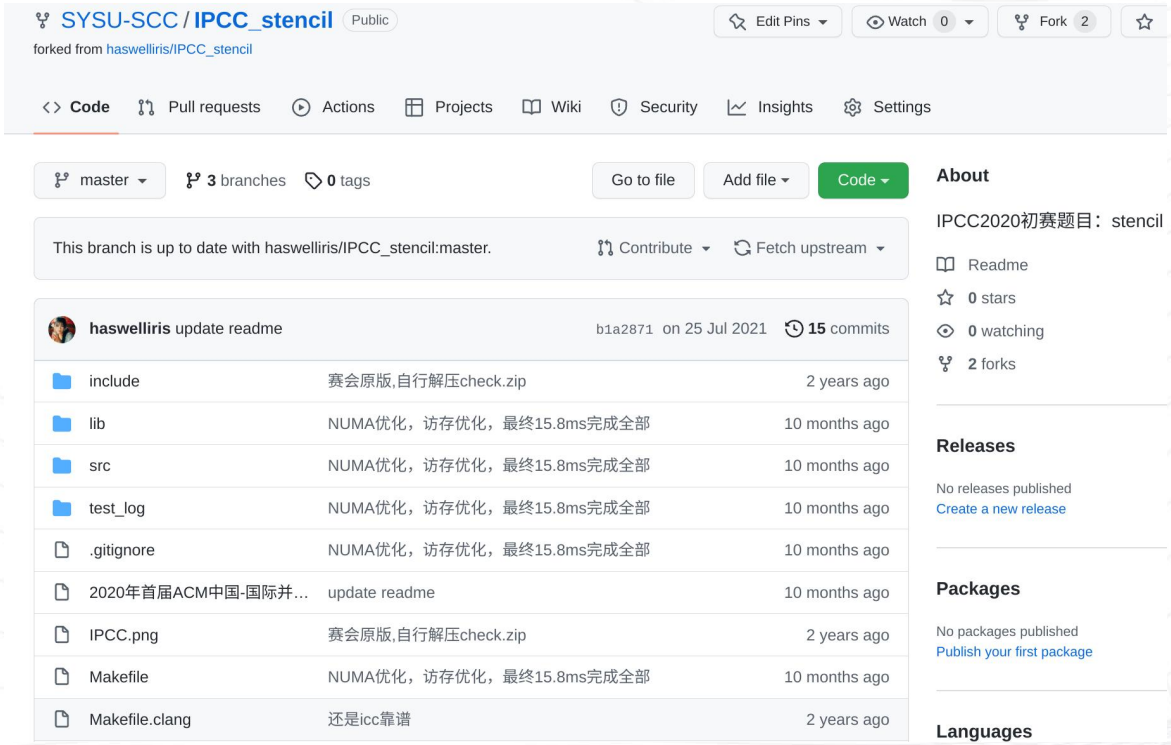
- 以 IPCC'21 中山大学_要搞快一点 为例：
  - 算法前端（我）
    - 熟悉数据结构，分析算法与机器特性，进行串行算法优化与并行算法设计
  - 数学前端（谢嘉斌）
    - 跨学科队友（中大航院本硕），熟悉 CFD，负责数值算法优化
  - 后端（梁杰鑫、冯浚轩）
    - 熟悉 X86，擅长向量化，极限压榨性能
  - 队长（我）
    - 掌握全局状态与思路，跟踪每个人的进度，适度 push，定期开会甩锅
- 前后端分离（类似于编译器设计）
  - 每个人都要专精自己的分工，也要会其他人的部分（冗余备份）

起一个队名

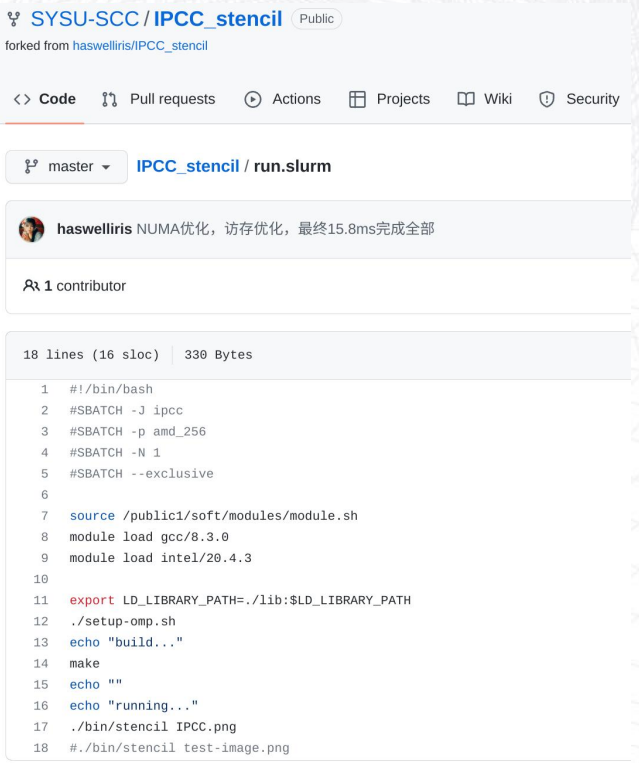
- 纯爱战士（√）
- 要搞快一点（√）
- WuK 粉丝后援团（×）

如何获得优化思路

- 比赛开始了，首先？
  - 创建代码仓库，做好分支管理，每个队员定期同步进度，方便甩锅
  - 编写一键编译&运行的脚本，非常重要，极大影响比赛体验！



来源：IPCC'20 中山大学_要搞快一点（参赛队员：张力越、谢嘉斌、梁杰鑫）代码仓库



https://github.com/SYSU-SCC/IPCC_stencil/blob/master/run.slurm

i@wu-kan.cn
https://wu-kan.cn
中山大学超算队 吴坎

- 视比赛计算资源决定是否要移植到本地或自有集群上进行调试
  - 比赛集群是否能够方便的 profile？是否连外网？
  - 比赛集群计算资源是否足够，会不会要“抢节点”？
  - 和自有机器相比，赛会提供的机器在软硬件环境上有什么特点？
  - e.g.CPC'21 决赛开放了神秘的新神威，中山大学_纯爱战士 的分工：
    - （我，算法前端）将赛题移植回“天河二号”，先进行与硬件无关的计算图优化
    - （吕天翔，数学前端，中大航院本科生）研究面向新神威的矩阵乘法
    - （冯浚轩，后端）对新神威进行一系列 benchmark，测试从核带宽、计算效率

应用程序运行的硬件环境和软件环境——硬件信息

CPU	Intel 10875H	AMD EPYC 7452	Intel Xeon Glod 8180
Core(s) per socket	8	32	28
Thread(s) per core	2	1	2
Sockets (numa)	1	2	2
Frequency	2.3 ~ 5.1(4.3 all) GHz	2.35 ~ 3.35 GHz	2.5 ~ 3.8 GHz
L1d/L1i cache	256KB/256KB	32KB/32KB	32KB/32KB
L2 cache	2MB	512KB	1MB
L3 cache	16MB	16MB	38MB
AVX2-GFLOPS	< 700	2406.4*2	2240*2
stream	24.1GB/s	244.9GB/s	137.4GB/s
Max bandwidth	45.8 GB/s	400 GB/s	250 GB/s
	开发/编译平台	运行平台	参考参数

开发/编译平台：INTEL i7-10875H
信息简述：8核16线程，开发及初步调优使用

运行平台：AMD EPYC 7452
信息简述：NUMA架构，32核x2，无超线程，支持 fma，avx2，比赛实际运行平台

应用程序运行的硬件环境和软件环境——软件信息

```
No LSB modules are available.
Distributor ID: Ubuntu
Description:    Ubuntu 20.04.2 LTS
Release:        20.04
Codename:       focal
```

```
LSB Version:    :core-4.1-amd64:core-4.1-noarch:cxx-4.1-amd64:cxx-4.1-noarch:desktop-4.1-amd64:desktop-4.1-noarch:languages-4.1-amd64:languages-4.1-noarch:printing-4.1-amd64:printing-4.1-noarch
Distributor ID: CentOS
Description:    CentOS Linux release 7.9.2009 (Core)
Release:        7.9.2009
Codename:       Core
```

开发/编译平台：
系统：WSL2 Ubuntu 20.04 LTS
Gcc： 9.3.0
Glibc：2.31

合适的编译环境依赖于：
gcc>=9，glibc>=2.30

运行平台：
系统：CentOS Linux 7.9
Gcc： 4.8.5， icpc：19.1.3.304
Glibc：2.17(本机自带版本) 2.31(静态链接版本)

在没有合适编译机的情况下，可以直接使用平台上的 icpc 编译，此时与左边情况性能差距(降低)约为 5% - 30%。

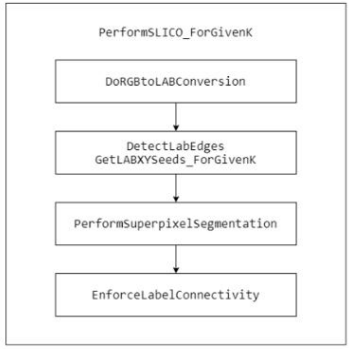
来源：《IPCC圆桌派丨初赛优化作品分享》

中山大学超算队 吴坎
i@wu-kan.cn
https://wu-kan.cn

- 尝试理解代码逻辑——局部的、全局的
  - 全局不强求，有时很困难——但更深的理解总能带来更多的优化空间
- 观察代码，找到代码热点，决定后续优化方向
  - 手动插入计时代码：详细、准确、影响小，但 log 可能会很长
    - 比赛后期，最好精确到每个函数的每个主要 for 循环
  - 借助 gprof、vtune、ncu 等一系列工具：直观，但不一定与实际运行情况完全等价，且通常需要管理员权限

应用程序的代码结构

ACM中国 国际并行计算挑战赛



计时区仅有 SLIC::PerformSLICO_ForGivenK 一个函数

主要流程分为左图的四步：

1. 将 ubuff 中以 RGB 存储的像素转换为 LAB 格式
2. 在 LAB 空间做边缘检测，生成种子
3. 运行 K 聚类算法，迭代多次
4. 运用 BFS 算法遍历各连通聚类，重新标记。

优化方法 基准建立

ACM中国 国际并行计算挑战赛

1. 以初赛赛题为基准，以平台默认方式为准，编译及运行方法如赛题所示，其中版本为 g++==4.8.5，glibc==2.17

程序使用方法

1）参考编译命令：g++ -std=c++11 SLIC.cpp -o SLIC

2）运行命令：srun –p amd_256 –N 1 ./SLIC

```
[sca3207@ln121%bscc-a5 SLIC_BK]$ srun -p amd_256 -N 1 -t 10 ./SLIC.out
Computing time=25263 ms
There are 0 points' labels are different from original file.
```

2. 原代码开启-O3：

```
g++ -std=c++11 -O3 SLIC.cpp -o SLIC.out
[sca3207@ln121%bscc-a5 SLIC_BK]$ srun -p amd_256 -N 1 -t 10 ./SLIC.out
Computing time=8520 ms
There are 0 points' labels are different from original file.
```

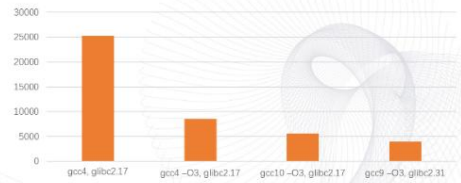
3. 原代码开启-O3，更换平台上的 gcc10：

```
g++ -std=c++11 -O3 SLIC.cpp -o SLIC.out
[sca3207@ln121%bscc-a5 SLIC_BK]$ srun -p amd_256 -N 1 -t 10 ./SLIC.out
Computing time=5635 ms
There are 0 points' labels are different from original file.
```

4. 原代码开启-O3，更换为本地的 gcc9，静态链接 glibc2.31：

```
[sca3207@ln121%bscc-a5 SLIC_BK]$ srun -p amd_256 -N 1 -t 10 ./SLIC.out
Computing time=3920 ms
There are 0 points' labels are different from original file.
```

gcc4, glibc2.17	gcc4 –O3, glibc2.17	gcc10 –O3, glibc2.17	gcc9 –O3, glibc2.31
25263 ms	8520 ms	5635 ms	3920 ms



由于基准 1 性能较差，不利于优化指标参考，在优化方法章节，以上述基准4为优化基准作为对比。
之后，将以基准1为原始基准，基准4为优化基准。

来源：《IPCC圆桌派丨初赛优化作品分享》

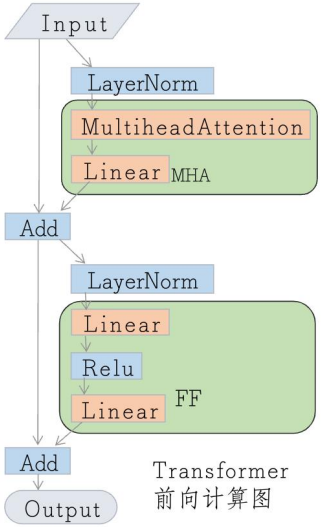
中山大学超算队 吴坎
i@wu-kan.cn
https://wu-kan.cn

如何获得优化思路

- 计算理论算力、内存带宽与程序实际达到的效率
  - 程序的不同部分，是计算密集还是访存密集？
  - 该部分的优化是否已近逼近极限，有/没有必要继续？
- 增加代码局部性：循环融合、循环次序调整、算子融合

算子分析

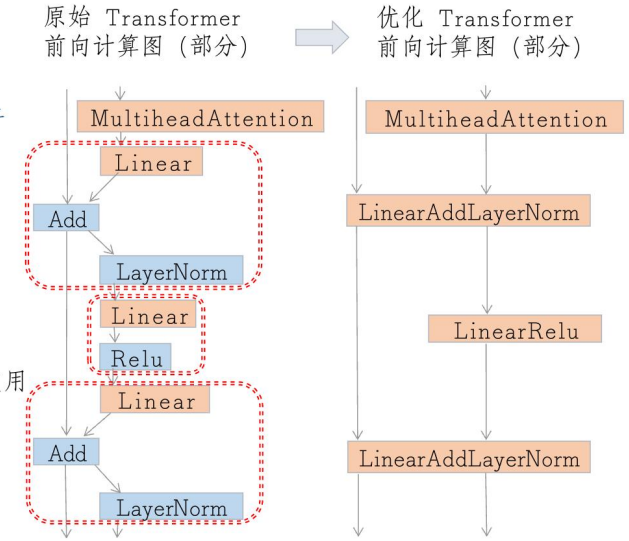
- 单个 Transformer 包含两个子层
  - MHA（Multi-Head Attention）
  - FF（Feed Forward）
  - 层与层之间有 Add & Layer Normalization
- 复杂度分析与测量实验的结果表明
  - 计算密集算子与访存密集算子交替出现
  - MHA 与 FF 占据总计算时间的 80% 以上
- 难点与挑战：
  - 对于计算密集算子，如何充分逼近理论算力？
  - 对于访存密集算子，如何高效利用内存带宽？



Transformer 前向计算图

我们的工作

- 计算图优化
  - 核融合（Kernel Fusion）
    - 合并相邻的计算密集算子与访存密集算子
      - LinearAddLayerNorm
      - LinearRelu
    - 降低访存压力
    - 减少多次启动计算核的开销
  - 层融合（Layer Fusion）
    - 实际模型常将多层 Transformer 堆叠使用
    - 针对相邻 Transformer 的连接处优化
      - 可以执行更多核融合！



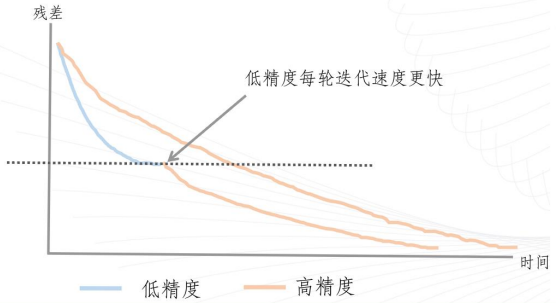
来源：《CPC'21 中山大学_纯爱战士 决赛作品优化报告》

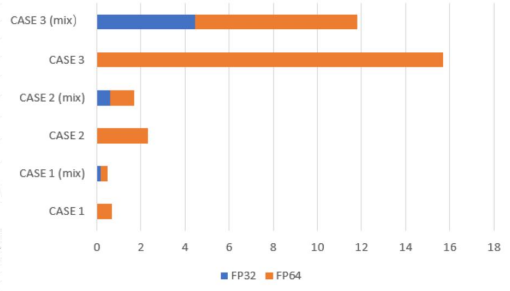
中山大学超算队 吴坎
i@wu-kan.cn
https://wu-kan.cn

- 是否可以使用混合精度？
  - 访存密集应用，尝试使用低精度存储、高精度运算
  - 计算密集应用，尝试使用高精度存储、低精度运算
  - 迭代算法，尝试使用先低精度、后高精度
    - e.g.IPCC'21 决赛，先低后高可使运行时间减少四分之一

我们的工作 中山大学_要搞快一点

- 混合精度算法
  - 算法的初期阶段，对精度要求较低!
  - 先通过单精度快速得到一个较低精度的解
    - 实验中发现单精度通常收敛在 0.001，以此为单精度迭代的结束阈值
  - 在此基础上使用双精度迭代，直至符合要求





来源：《IPCC'21 中山大学_要搞快一点 决赛作品优化报告》

4.5 混合精度优化

先用单精度迭代求解到指定精度，再改用双精度求得最终解

迭代步数↓/Case→	Case-1	Case-2	Case-3
单精度迭代步数	11	12	12
双精度迭代步数	9	10	10
总迭代步数	20	22	22

	Case-1	Case-2	Case-3
基准性能/s	46.995	103.603	550.487
优化后耗时/s	0.523	1.409	11.224
加速比	89.93	73.54	49.05

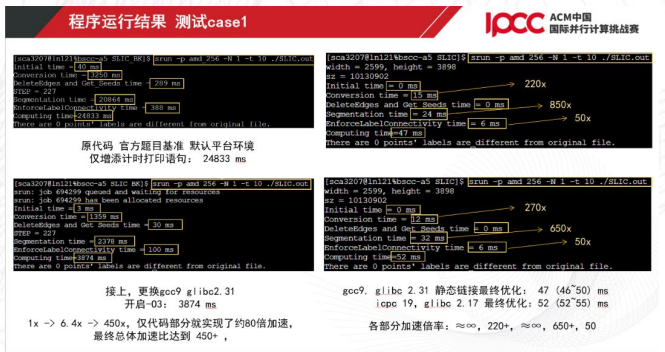
编译优化 | 消除冗余 | 舒尔补预处理 | BiCGSTAB | 混合精度 | 访存优化 | 调整运行参数

来源：《IPCC'21 SJTU_CHPC 决赛作品优化报告》

- 掌握硬件特性
  - 多进程多线程计算时是否绑核？如何分配？NUMA 的影响是怎样的？
    - 本次全明星赛即是一个非常经典的案例！
  - 编译选项是否贴合硬件？该设备支持怎样的向量化指令？
    - AVX-512 指令一定比 AVX2 高效吗？-O3 一定比 -O2 快吗？
  - 将在后续分享中详细展开，敬请关注！
- 从编译器获得灵感
  - 不同的编译器，对程序的不同部分，各自会有什么影响？
  - 为什么？可以在另外一个编译器上达到同样的效果吗？

- aocc/icc/icx/clang 一定比 gcc 好吗？
  - 例如，日常使用中发现，icc@19 后，很多旧应用编译失败…
  - IPCC'21 初赛应用 SLIC，直接使用 icc 编译会快三倍…但是结果错误！
- 新版本编译器一定比旧版本好吗？
  - 例如，日常使用中发现，gcc@8 后编译的 GPGPU-Sim 应用出现段错误…
  - gcc@9 及之后与 cuda@10 及之前的版本不兼容…
  - icc@17 编译的 SLIC 运行时间与 icc@16，icc@18，icc@19 差异很大

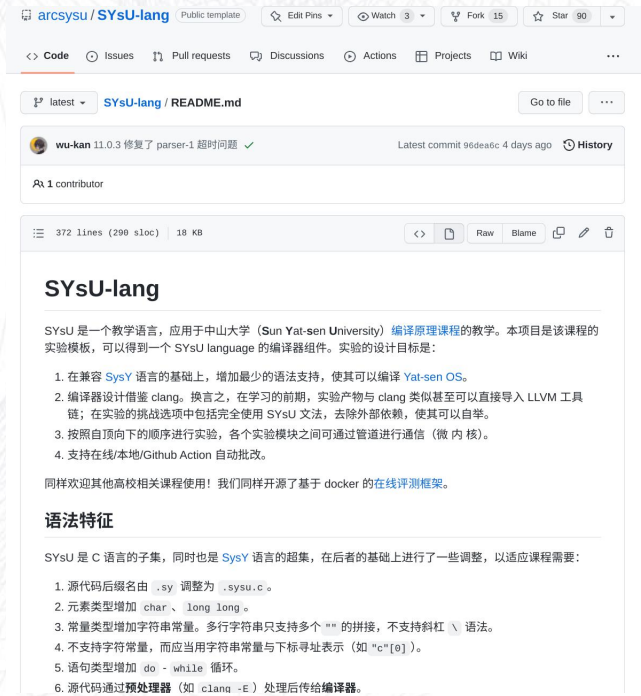
- 我们的策略
  - 以一个稳定版本的 gcc 作为主力编译器，此处推荐 gcc@7.5.0
    - gcc@7 后开始支持 C++17，同时对旧应用的兼容性也较好
  - 用多版本 aocc/icc/icx/clang/gcc 编译测试（可以编写脚本批处理）
  - 比较不同编译器不同代码段的运行时间（包括比赛集群和自有集群）
  - 探究该编译器优化的原理/结果错误的原因，然后在 gcc@7.5.0 上复现
- e.g.IPCC'21 初赛第一名为 46ms，仍有提高空间
  - 比赛时发现 icc 编译的 SLIC 变快，但结果错误
  - 探究后查明 icc 对其进行了查表优化
  - 我们在 gcc 上手动实现查表，实现更优的 RGB 转换
    - 6ms/12ms



来源：《IPCC圆桌派丨初赛优化作品分享》

干货：编译器使用策略

- 了解编译器所做的工作很有用
  - 能让你从前端的语法维度、代码优化的维度、与硬件结合的维度几个方面，加深对计算机技术的理解，提升自己的竞争力
  - 无论你是前端工程师、后端工程师，还是运维工程师
  - 在知道底层原理的情况下，对上层的描绘会更加写实
- 欢迎试用编译原理实验 SYsU-lang
  - 不超过三千行，实现一个精简 C 语言编译器
  - 帮助理解：机器是怎样认识我们写的代码？
  - 有助于我们写出更高效的代码！



https://github.com/arcsysu/SYsU-lang

龙书《编译原理》

扫码提交问卷调查

组委会联系方式

- 官网： www.paraedu.org.cn
- 邮箱： ACM_IPCC@163.com
- 电话： 18310726311（余老师）
- 微信公众号：
  - 北京超级云计算中心(BJBLSC)
- 交流群：
  - 1046805935（参赛选手）
  - 1095416620（指导老师）

- 自行搭建 wiki/飞书，所有队员都需要写总结
- 在内部培训中分享参赛经历与心得
- 日常练习一定要自带计算资源（超算集群）才可以吗？
  - 并不是！超算竞赛总是建立在对硬件特性的掌握能力上的
  - 掌握“掌握硬件”的能力，比掌握特定的硬件本身更重要
  - “图吧垃圾佬”，甚至有一块树莓派都可以！只要：
    - 掌握硬件的理论算力、带宽
    - 计算你实际达到的计算效率，并为之优化！
- 更多学习资料与往届讲座分享，尽在比赛交流群 1046805935！

中大超算队欢迎合作交流

Association for Computing Machinery

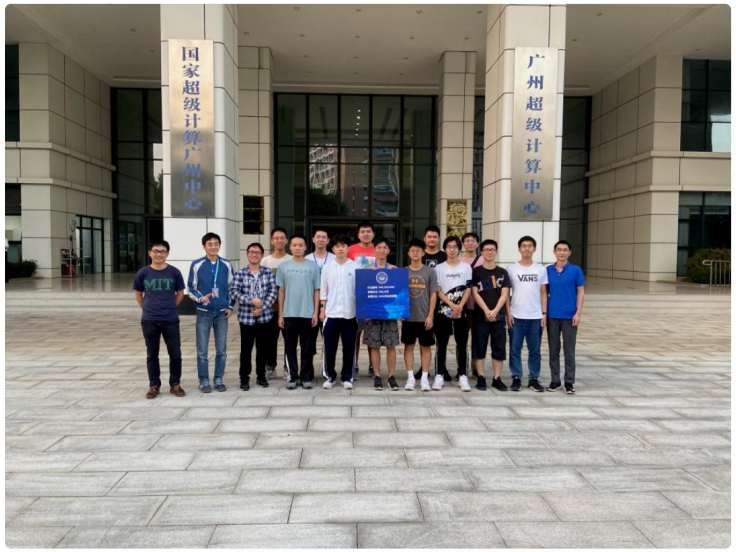
ACM中国 国际并行计算挑战赛

- 学习资料开源，友好的超算竞赛环境，你我共建:)
  - 中大超算队会公开部分内培材料与参赛代码、竞赛脚本
  - 欢迎交换友链、互相学习!

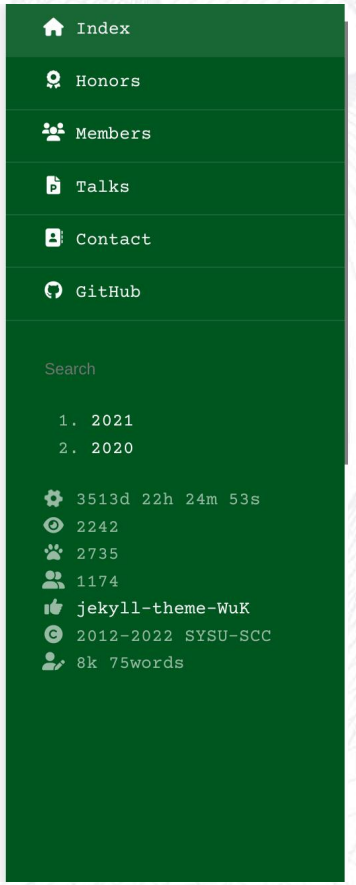
Welcome

We are Student Cluster Competition Team at Sun Yat-sen University, and conduct research in the broad area of high performance computing and cluster systems.

2021



https://scc.sysu.tech



Nov. 2021

- K. Wu. 为啥老拿第三. *29 Nov. 2021.*
- Y. Wang. 存储系统：从没入门（0）到刚入门（1）. *21 Nov. 2021.*

Oct. 2021

- X. Huang. 编译、装载与库. *11 Oct. 2021.*

Jun. 2021

- L. Zhang. 并行优化实战. *27 Jun. 2021.*
- L. Zhang. 并行开发技术概论. *06 Jun. 2021.*

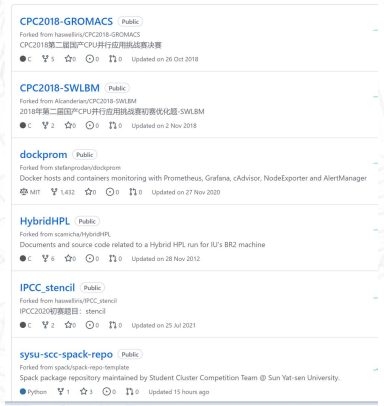
Mar. 2021

- J. Feng. CPU 内核架构：亿点点细节. *27 Mar. 2021.*

Dec. 2020

- Y. Li. AI 赛题调优经验分享. *28 Dec. 2020.*
- K. Wu. CUDA 矩阵乘法的优化. *17 Dec. 2020.*
- Y. Wang. 存储系统：从没入门（0）到刚入门（1）. *13 Dec. 2020.*
- G. Feng. InfiniBand 相关知识+MPI 使用技巧+CPC 赛题优化分享. *08 Dec. 2020.*

https://scc.sysu.tech/Talks/



https://github.com/SYSU-SCC

i@wu-kan.cn
https://wu-kan.cn
中山大学超算队 吴坎

官网注册通道

组委会联系方式

- 官网：www.paraedu.org.cn
- 邮箱：ACM_IPCC@163.com
- 电话：18310726311（余老师）
- 微信公众号：
  - 北京超级云计算中心(BJBLSC)
- 交流群：
  - 1046805935（参赛选手）
  - 1095416620（指导老师）

Thanks!

ACM 中国-国际并行挑战赛-赛前讲座2022

下期预告：超算基本原理与方法论

不要走开，精彩继续～

i@wu-kan.cn
https://wu-kan.cn

中山大学超算队 吴坎

中山大學 SUN YAT-SEN UNIVERSITY 国家超级计算广州中心 NATIONAL SUPERCOMPUTER CENTER IN GUANGZHOU