第二届中国研究生操作系统开源创新大赛赛题

系统创新

1.赛题题目：基于虚拟化技术的操作系统安全增强技术

赛题说明：

安全是操作系统的重要需求，通用操作系统在密码凭证、生物信息等敏感数据保护方面存在风险，现有安全保护机制依赖处理器特定的硬件机制（如TrustZone、SGX等），在部署使用存在一定的局限性。基于虚拟化进行系统安全增强（如Windows的VBS机制、Android的pKVM等）也是近年来的重要方案，此外目前基于内存安全语言Rust实现操作系统也能够规避安全风险实现安全内核，将两者结合，可以借助虚拟化技术实现双内核架构可以解决安全和生态的平衡，通过同时运行通用内核和安全内核，由安全内核运行安全应用，为通用内核提供安全服务；由虚拟化层提供内存隔离和安全通信，从而支持内存隔离、凭据保护、安全分级等需求。

赛题要求：

基于虚拟化技术构建双内核架构，包括通用内核和安全内核，其中安全内核采用Rust语言实现，从而提升自身安全性。具体要求如下：

* 虚拟化可以适配主流Hypervisor，包括但不限于Jailhouse、Rust-Shyper等；
* 支持同时运行通用内核和安全内核，其中通用内核是Linux等通用操作系统内核，安全内核是基于Rust语言实现的内核；
* 支持通用内核与安全内核的内存隔离和安全通信；
* 设计安全应用，包括但不限于密码服务、凭据保护等，实现安全内核为通用内核提供安全服务的演示；
* 任选一个处理器架构平台。

评分标准：

功能完整性（30%）：

* 基于Rust安全内核完成Hypervisor适配（40分）；
* 支持通用Linux内核和安全内核进行数据通信传输和控制交互（60分）。

应用效果（40%）：

* 设计应用场景，展示通用内核安全增强效果（60分）；
* 开展性能测试和优化，性能有明显提升的可以加分（40分）。

代码规范性（20%）：

* 代码结构清晰、可维护性高（80分）；
* 符合开源社区规范（20分）。

文档质量（10%）：

* 文档包含设计说明、部署说明、测试报告（50分）；
* 文档条例清晰、撰写规范（50分）。

赛题联系人：

马俊 majun@nudt.edu.cn

参考资料：

1. Rust内核ArceOS：<https://github.com/arceos-org/arceos>
2. Rust内核Asterinas：https://github.com/asterinas/asterinas
3. Rust内核Kycore：<https://gitee.com/openkylin/kycore>
4. Hypervisor Jailhouse：https://github.com/siemens/jailhouse
5. Hypervisor Rust-Shyper：<https://gitee.com/openeuler/rust_shyper>

参赛资源支持：无

2.赛题题目：基于Rust语言的运行态无关驱动框架设计

赛题说明：

组件化是现代操作系统开发的重要手段，尤其是目前基于Rust语言开发的新型操作系统内核利用Rust crate机制，构建操作系统无关的功能组件（如设备驱动），可面向不同应用需求快速构建不同类型的操作系统内核。目前的组件化实践主要是编译时按需编排内核组件，可运行在不同内核、不同特权态，但上述多样性在编译时确定，运行后无法按需调整。一个更为灵活的设计目标是运行时的运行态无关，即系统运行过程中可动态按需调整组件运行态，从而灵活适应用户对性能、安全等需求的灵活支撑。

赛题要求：

* 设计/改进基于Rust语言的操作系统内核，支持组件化开发（组件自身设计为独立的Rust crate）；设计运行态无关的驱动组件运行框架，支持动态切换驱动的运行态（即内核态或用户态）。具体要求如下：
* 选择一款Rust语言实现的操作系统内核进行改进或者使用Rust语言新设计实现操作系统内核；
* 实现设备驱动运行框架，驱动程序自身设计为独立的Rust crate（可复用开源代码但需清楚标注）；
* 驱动代码支持在用户态和内核态运行（需将运行态相关的代码抽象为Rust trait，驱动组件依赖上述Rust trait实现，自身不包含运行态相关的代码）；
* 操作系统内核调用驱动时根据不同的运行态实现相关trait；
* 实验环境任选一款ARM64处理器平台，支持更多平台的可根据难度加分；
* 设计测试用例，验证驱动功能。

**评分标准：**

功能完整性（50%）：

* 基于Rust内核完成驱动组件开发，包括但不限于SD卡、网卡、WIFI等设备（40分）；
* 设计运行态无关的驱动组件运行框架，支持驱动组件在用户态和内核态运行（60分）。

应用效果（20%）：

* 设计应用场景，展示运行态动态切换优势（100分）。

代码规范性（20%）：

* 代码结构清晰、可维护性高（80分）；
* 符合开源社区规范（20分）。

文档质量（10%）：

* 文档包含设计说明、部署说明、测试报告（50分）；
* 文档条例清晰、撰写规范（50分）。

赛题联系人：

马俊 majun@nudt.edu.cn

参考资料：

1. Rust内核ArceOS：<https://github.com/arceos-org/arceos>
2. Rust内核Asterinas：https://github.com/asterinas/asterinas
3. Rust内核Kycore：<https://gitee.com/openkylin/kycore>

参赛资源支持：无

3.赛题题目：类ChromeOS的轻量级操作系统设计与实现

赛题说明：

随着移动互联网和云计算的发展，用户对操作系统的需求逐渐从单一设备转向多设备协同，亟需从操作系统角度提供统一体验。本次大赛要求参赛队伍设计并实现一款类似ChromeOS的轻量级操作系统，支持传统桌面环境和移动终端设备（如平板、手机），从设计到实现上为用户在系统体验上提供一致视图。该系统应具备高效、跨平台的特点，能够无缝切换桌面与移动模式。操作系统应基于Linux内核，鼓励使用开源技术栈（可参考gecko-b2g和nutria开源项目）作为运行时系统，同时需提供简洁的用户界面和良好的用户体验。

赛题要求：

1. 系统架构：基于Linux内核和Gecko HTML5（或Blink等）打造统一桌面环境，同一份代码支持多种架构，能够在桌面和移动设备上运行。

2. 用户界面：提供桌面模式和移动模式两种界面，支持触控和键鼠操作。

3. 性能优化：系统应轻量化，启动速度快，资源占用低，适合低端设备运行。系统占用内存小于2GB，启动时间小于10s。

4.开发文档：提供详细的设计文档、用户手册和开发指南。

评分标准：

功能完整性（40%）：

* 系统是否满足跨平台特性：支持不同架构平台，支持触控和键鼠操作（60分）；
* 支持移动模式和桌面模式热切换（40分）。

用户体验（20%）：

* 界面设计是否简洁易用，操作是否流畅（50分）；
* 多分辨率屏幕自适应（50分）。

性能优化（20%）：

* 系统占用内存小于2GB，系统启动时间小于10s（50分）；
* 可在低端设备运行：内存<=4GB，CPU核心数<=4（50分）。

文档质量（20%）

* 代码结构清晰、符合开源社区规范、可维护性高（50分）；
* 设计文档、用户手册和开发指南是否清晰完整（50分）。

赛题联系人：

彭龙 penglong@nudt.edu.cn

参考资料：

1. gecko-b2g, https://github.com/kaiostech/gecko-b2g
2. nutria, <https://github.com/capyloon/nutria>

参赛资源支持：无

4.赛题题目：基于可信执行环境的机密容器实现与优化

赛题说明：

随着云原生的不断发展，容器作为一种轻量级、可移植的虚拟化技术得到广泛应用。它能够将应用程序及其运行环境打包成一个独立的容器，实现应用程序在不同环境间的一致性和可移植性。与传统的基于VMM的虚拟化不同，基于容器的虚拟化技术共享相同的主机操作系统，这种轻量级隔离机制带来了性能优势，但也引入了许多安全风险。机密计算是一种基于硬件的安全技术，其核心目标是通过使用可信执行环境，保护运行时代码和数据的机密性和完整性。本赛题要求基于TEE技术构建优化的机密容器运行时现有机密容器运行时，对其性能进行优化，实现轻量级、高效运行的目标。

赛题要求：

1. 基于AMD SEV、Intel SGX/TDX、ARM TrustZone/CCA等主流TEE技术构建优化的机密容器运行时

2. 对机密容器运行时进行优化，在保证确保机密计算的机密性、完整性的前提下，降低容器运行时开销，如减少容器启动时间、内存占用及执行标准计算任务的耗时降低等

评分标准：

功能完整性（30%）：

* 优化后的机密容器需完整支持容器生命周期管理（创建/启动/停止/删除）（40分）；
* 支持TEE的远程证明（Attestation）、内存加密等核心安全机制（30分）；
* 兼容标准容器接口（如OCI运行时规范）（20分）；
* 支持运行智能计算任务（10分）。

性能优化（50%）：

* 与机密 Kata Container 容器运行时相比，优化后的性能（包括容器启动时间、部署效率）提升至少10%（100分）。

文档质量（20%）：

* 详细说明优化思路、实验方法和性能测试结果分析（50分）；
* 文档条例清晰、撰写规范（50分）。

赛题联系人：

李小玲 lixiaoling@nudt.edu.cn

参考资料：

1. https://confidentialcontainers.org/
2. https://github.com/confidential-containers

参赛资源支持：无

5.赛题题目：基于 Rust 语言的内核组件化设计与实现

赛题说明：

Rust 语言凭借内存安全、零成本抽象及高性能特性，已成为系统编程领域的重要革新力量。本赛题聚焦内核组件化设计，要求利用 Rust 的模块化能力、所有权机制和异步编程机制等，实现内核功能解耦、可扩展架构和支持高并发执行。通过实践 Rust 内核功能组件开发，参赛者需解决跨语言接口适配、调度性能优化和资源管理优化等核心问题，并形成不同形态操作系统内核架构，包括宏内核、微内核、虚拟化软件，探索 Rust 在内核场景下的工程化应用，验证新型编程范式对系统底层的改造潜力，进一步研究内核组件故障隔离、动态加载和热升级能力，为操作系统弹性部署与漏洞修复提供新路径。推动系统级代码的高性能、安全性与开发效率双重提升。

赛题要求：

任务目标：基于在唐图社区（包括位于github上的rcore-os，arceos, arceos-hypervisor，kern-crates等）的Rust内核组件和组件框架进行改进，或进一步设计实现新的内核组件和内核组件框架，形成可灵活组合的各种异构形态和功能的内核。具体要求如下所示（可在下面不同形态的内核中选择一个或多个）：

* 改进或创新并组合组件形成支持Linux应用的组件化宏内核，可运行Linux应用测试用例；
* 改进或创新并组合组件形成支持Linux应用的组件化微内核，可运行Linux应用测试用例；
* 改进或创新并组合组件形成支持运行多个内核的组件化虚拟化软件，可运行支持Linux应用的组件化宏内核，以及支持Linux应用的组件化微内核等；
* 改进或创新并组合组件形成各种新颖功能或创新功能的组件化内核，如支持硬件新特性、故障隔离、动态加载和热升级能力等。

评分标准：

功能完整性（60%）

* 改进或创新并组合组件形成的宏内核或微内核形态可运行Linux应用测试用例，以通过的测例数量为功能完整性评价指标；改进或创新并组合组件形成的虚拟化软件可运行Guest Mode的Linux内核和某个其他内核，以支持的虚拟化功能数量为功能完整性评价指标；改进或创新并组合组件形成新型组件化内核的新颖功能，有实际测例说明组件和内核的创新性，以创新性的可量化指标为功能完整性评价指标（100分）。

应用效果（20%）

* 支持x86-64/aarch64/Risc-V64/LoongArch64 QEMU-9.2+模拟环境或物理环境中的一种处理器上运行（40分）；
* 有额外的测试用例能展示组件或组件化内核的功能、性能、安全等特征（40分）；
* 运行效果可重现（20分）。

代码规范（10%）

* 基于Rust 2024版本进行开发（40分）；
* 代码结构清晰、符合开源社区规范、可维护性高（20分）；
* 能通过Rust的代码规范性检查（40分）。

文档质量（10%）

* 文档包含设计文档、安装部署说明、测试分析报告（50分）；
* 文档条例清晰、撰写规范（50分）。

赛题联系人：

向勇 [xyong@tsinghua.edu.cn](mailto:xyong@tsinghua.edu.cn)

陈渝 [yuchen@tsinghua.edu.cn](mailto:yuchen@tsinghua.edu.cn)

参考资料：

1. <https://github.com/rcore-os>
2. [https://github.com/oscomp](https://github.com/oscomp/arceos)
3. [https://github.com/arceos-hypervisor](https://github.com/arceos-hypervisor/axvisor)
4. <https://github.com/taic-repo>
5. <https://github.com/kern-crates>

参赛资源支持：无

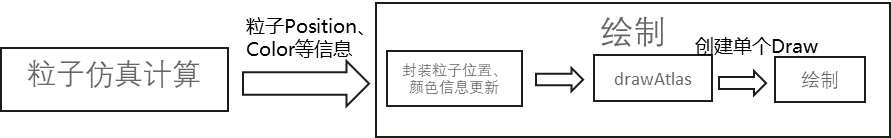
6.赛题题目：端侧操作系统图形2D粒子动效优化

赛题说明：

粒子动效提升了界面的丰富性，在大桌面转场，文件传送，扫描二维码，以及在应用的视频背景等场景中应用非常广泛。开源端侧操作系统如OpenHarmony操作系统提供了系统的2D粒子动效能力，用户可以使用开放的粒子动画接口，在应用或者大桌面场景创建UI粒子动画效果。



在动视效UX设计越来越高端精致的趋势下，设计师和应用场景提出了更多粒子数以及更多的粒子运动效果（力场作用，颜色渐变）的需求。但现有的粒子系统在终端功耗的约束下，只能支撑1000个粒子的实时(120fps)计算渲染。原因分析如下。OpenHarmony现有的2D粒子系统主要包括了粒子计算和粒子绘制两个主要的过程。粒子绘制在2D统一渲染框架中，使用Skia绘制。由于skia并不提供GPU计算管线，粒子计算当前采用了CPU计算的方式。具体流程如下图示意：

**

OpenHarmony粒子动画的主要步骤和流程

本创新大赛的赛题为：通过优化粒子计算和粒子绘制流程，支撑端侧1万粒子120帧实时仿真绘制，相同场景下功耗低于优化前的基线。优化手段可包括但不限于粒子计算和粒子绘制全GPU化、指令合批，编译优化等。

这里提供一个具体优化思路分析，做为对参赛选手的引导。

参考思路：考虑抛弃使用CPU实现粒子计算，将粒子计算和粒子绘制融合到一个流程，通过向skia提交粒子绘制指令，在GPU中完成粒子计算和结果更新最终直接完成粒子绘制结果。这个思路优点是GPU适用于大规模粒子的场景，在性能和能效上有先天的优势，需要解决的问题是，粒子动画是一个有状态流程，每帧中粒子计算的输入值取决于上一帧粒子计算的结果，如果使用全GPU解决方案，如何每帧保存这个粒子状态也是一个技术挑战。

赛题的挑战指标：5000个粒子场景实现120Hz的帧率，在相同场景下优化后的功耗要优化前的功耗降低30%。

赛题要求：

1. 需要基于开源操作系统实现效果，需要满足目标效果；

2. 若基于OpenHarmony操作系统，实现方法不得与OpenHarmony操作系统已有算法雷同或者一致；

3. 优化后在粒子规模为5000个粒子数量的条件下，各个粒子效果的场景在移动端设备上120fps满帧运行；

4. 在相同的场景下（如1000个粒子数），效果不劣化的前提下，优化后相比OpenHarmony操作系统已有算法功耗优化30%；

5. 要求在给定硬件设备上进行比测。

评分标准：

|  |  |  |
| --- | --- | --- |
| **评分项** | **占比** | **说明** |
| 目标 | 40 | 达成10000个粒子并120fps满帧，40分；达成5000个粒子并120fps满帧，35分。性能若未达标120帧，按照差距比例给分。 |
| 功耗 | 40 | 优化后对比优化前功耗优化了30%，如果没有达到30%按照达成比例给分，如果是负优化，这项不得分； |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

粆倩文 chaoqianwen@huawei.com

参考资料：

1. OpenHarmony 操作系统粒子功能介绍：<https://gitee.com/openharmony/docs/blob/d18088691f14b83e1d402abb1c8e35ce33fca3d0/en/application-dev/reference/apis-arkui/arkui-ts/ts-particle-animation.md>
2. OpenHarmony操作系统当前算法实现：

https://gitee.com/openharmony/graphic\_graphic\_2d/blob/master/rosen/modules/render\_service\_base/src/animation/rs\_render\_particle\_system.cpp

<https://gitee.com/openharmony/graphic_graphic_2d/blob/master/rosen/modules/render_service_base/src/animation/rs_render_particle.cpp>

<https://gitee.com/openharmony/graphic_graphic_2d/blob/master/rosen/modules/render_service_base/src/animation/rs_render_particle_effector.cpp>

<https://gitee.com/openharmony/graphic_graphic_2d/blob/master/rosen/modules/render_service_base/src/animation/rs_render_particle_emitter.cpp>

1. 硬件平台Laval手机信息：https://laval.csdn.net/651288d19fe5a0689ae58be6.html?login=from\_csdn

参赛资源支持：

1. OpenHarmony社区可提供：硬件开发平台https://laval.csdn.net/651288d19fe5a0689ae58be6.html?login=from\_csdn

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136093426100789248](https://www.chaspark.com/" \l "/races/competitions/1136093426100789248)

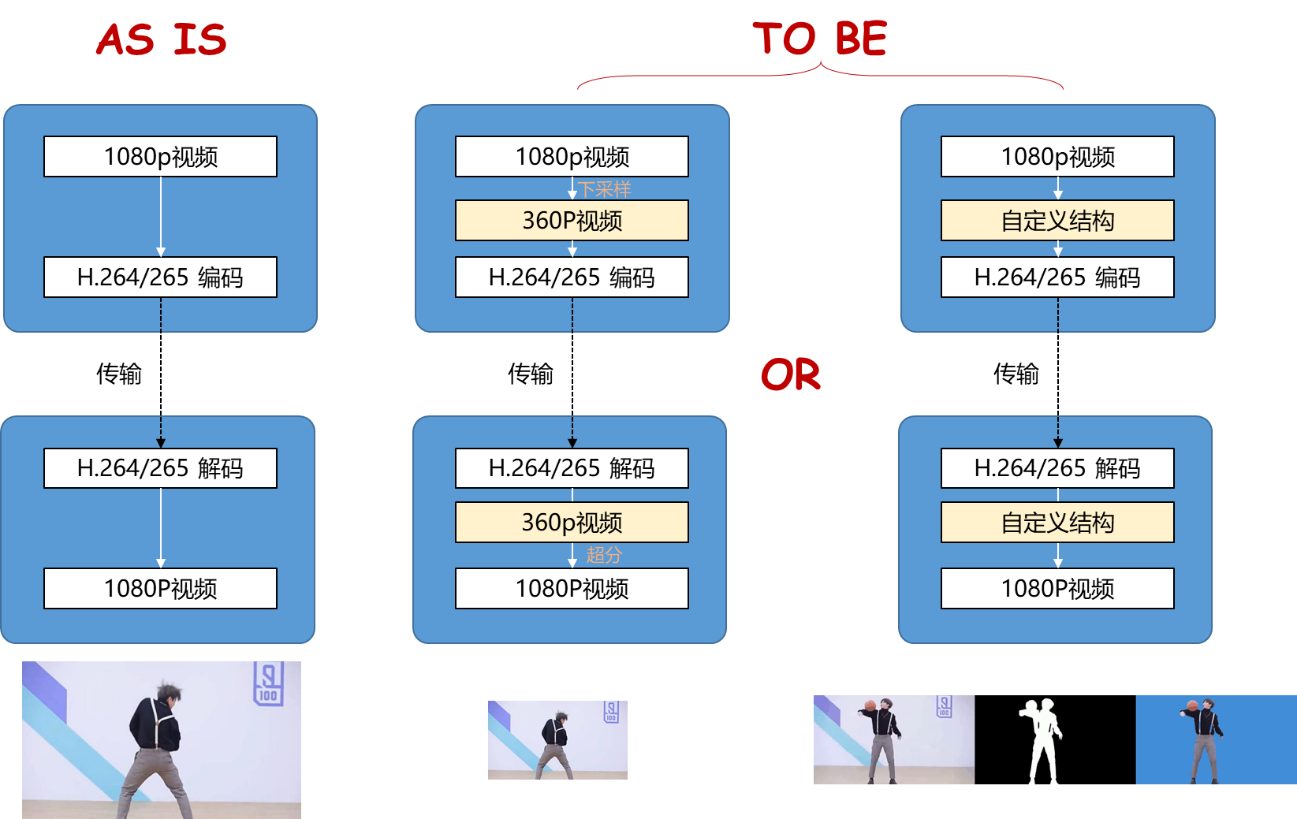
7.赛题题目：分布式硬件子系统高效视频传输

赛题说明：

当前直播话题非常火热，直播间场景的好坏直接影响用户的进入率，当前存在的问题如画面模糊、光线差、声音刺耳……极其影响用户体验。为了满足更好的直播场景效果，视频传输质量至关重要。

受到信号质量、带宽限制等因素影响，高画质高带宽视频传输成为一项重要挑战。满足用户对画质和流畅度的诉求，决定了直播场景的核心竞争力。此课题旨在分布式硬件子系统中实现一种视频传输高效高质量图像重建技术，实现对高画质高带宽视频的压力缓冲，保障用户高端精致的观赏体验。

技术路线不限，可以采用压缩感知、超分重建等技术路线，可以参考：超分辨率和背景提取的方法，压缩传输大小，通过端侧计算来换取带宽，实现高画质视频传输能力。



当前挑战：业界已具有多种超分模型和背景提取算法，但在端侧ROM和RAM资源有限情况下，需要考虑性能和效果的平衡。

1 高效的算法，超分和背景提取算法对算力要求高，需要设计高效的算法，此为该课题的主要挑战项；

2 帧率和画质性能，需要满足移动端设备的满帧运行，同时PSNR(峰值信噪比, Peak signal-to-noise ratio)和SSIM(结构相似性，structural similarity)达标。

赛题要求：

基于分布式相机模块在端侧进行原型验证。（https://gitee.com/openharmony/distributedhardware\_distributed\_camera）采用H.264视频编码方案，在dataprocessor中实现关键软件算法，实现低带宽高画质传输。具体要求如下：

1. 基于OpenHarmony操作系统实现图像重建分布式相机传输框架搭建，且必须使用图像重建技术对采集图像进行处理。

2. 基于提供不同画质打桩数据集(比赛提供少量数据集作为Sample，评审基于数据全集)，PSNR及SSIM综合指标不低于既定指标。数据集中包含大量高画质内容，90%上数据集是在低速运动或静止场景进行采集，10%数据集存在一些快速运动物体或存在相机运动情形，要求针对运动场景进行运动补偿。同一环境不同距离测试，课题采用了分布式相机框架，同时进行打桩测试，针对环境引起的变化会进行多个信号条件测试，反复测试10次求平均。最终按照数据集平均PSNR和SSIM得分进行评价。

3. 时效性：要求对整个图像重建进行日志标记，计算整个数据集条件下因图像重建算法产生的单次平均耗时。平均耗时不大于10ms（松弛度Slack为10ms，即20ms也可得分，下同），可得10分；平均耗时不大于50ms，可得8分；平均耗时不大于500ms，可得5分；平均耗时不大于1s，可得3分；平均耗时不大于3s，可得2分；平均耗时超过3s，不得分。

评分标准：

|  |  |  |
| --- | --- | --- |
| **评分项** | **占比** | **说明** |
| 功能 | 10 | 满足上述要求3（1），功能完整可得10分 |
| 性能 | 70 | 满足第三部分要求3（2）/（3），其中评分: 60 \*（0.6 \* PSNR + 0.4 \* SSIM）+ 时效性(占10分)。其中PSNR∈[0, ∞], PSNR∈[0, ∞]时得分为0；PSNR∈[20, 30]时得分为0.6；PSNR∈[30, ∞]时得分为MIN（1.0，（PSNR - 30）/ 10）。另外，SSIM∈[0, 1]，保持原值。性能得分在整体数据集上进行平均，总体得分向下取整。  代码规范性10%：代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0。 |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

董慧滨 donghuibin@huawei.com

李东华 lidonghua5@huawei.com

参考资料：

1. <https://gitee.com/openharmony/distributedhardware_distributed_camera>

参赛资源支持：

1. OpenHarmony社区可提供：硬件开发平台X2

<https://laval.csdn.net/651288d19fe5a0689ae58be6.html?login=from_csdn>

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136094700259041280](https://www.chaspark.com/" \l "/races/competitions/1136094700259041280)

8.赛题题目：基于MPAM的容器内存带宽监控管理工具

赛题说明：

MPAM（Memory system component Partitioning and Monitoring）是Arm Architecture v8.4的Extension特性。用于解决服务器系统中，混部不同类型业务时，由于共享资源的竞争（Cache，DMC，Interconnect），而带来的某些关键应用性能下降或者系统整体性能下降的问题。

MPAM应用的典型场景，例如云场景下，可针对不同业务同时作用于硬件访存路径上产生的竞争和冲突进行控制，从而帮助服务器提升利用率。

当前在主流Linux操作系统内核（如openEuler 6.6内核）上已使能MPAM的对外接口，但对于云原生容器场景，易用性不足，需要一个在K8s架构下，基于MPAM的对从节点到集群应用的内存带宽监控工具，方便运维人员在管理面对各个节点上的应用进行混部管理，保障高优业务的QoS，并能实时监测到各应用的带宽占用情况。

赛题要求：

* 带宽监控管理工具需要基于K8s管理，支持一键分发部署能力；
* 对于业务的优先级，可以用户指定或使用K8s默认的QoS level标准；
* 需要自己构造不同内存带宽诉求的应用进行混部测试，支持测试自动化执行；
* 监控结果可以在master节点收集，并提供对外接口，生成报告。

评分标准：

1. 在离线业务混部时，在线业务干扰率 α <10%。在线业务可以用带宽敏感型业务如redis、ngixn等业务模拟，离线使用大数据spark等业务模拟。

x：在线业务混部时TP99 时延

y：在线业务独立部署TP99 时延

α = (x - y) ) / x

2. 插件需要易部署并提供可观测能力，在k8s管理面可以一键分发并获取集群各个节点中业务带宽占用及竞争情况，便于运维人员进行实时分析调整。

3. 灵活的Qos调整能力，根据资源竞争对在线业务的干扰数据建模分析，灵活调整在离线业务的Qos等级、带宽限制，在在线业务干扰<10%的前提下，保障节点资源(CPU/内存等)利用率>60%。

赛题联系人：

玉凯 yukai20@huawei.com

参考资料：

1. https://gitee.com/openeuler/community/blob/f01bdb283d3860f1ffb17547b9317fdea9f26d31/sig/Kernel/mpam.md

参赛资源支持：无

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136096371816214528](https://www.chaspark.com/" \l "/races/competitions/1136096371816214528)

9.赛题题目：基于Linux操作系统的异构融合推理加速系统

赛题说明：

在算力需求爆发式增长的背景下，CPU与硬件加速器（如NPU/GPU）的深度协同成为突破计算瓶颈的核心方向。本赛题基于ARM系列处理器（例如鲲鹏920系列）和Linux系列操作系统，要求开发者构建高效异构加速系统，可通过指令集优化、任务卸载、内存共享等技术提升大模型推理整体性能，重点考察异构资源调度、跨设备通信优化等关键技术，推动自主可控的异构计算架构发展。

赛题要求：

CPU使用ARM系列处理器（例如鲲鹏920系列）；操作系统基于较新的Linux内核（6.6及以上）；可自由选择一种加速卡（昇腾/NVIDIA/天数智芯等）；

实现CPU与加速器的动态任务分配机制（如基于负载预测的任务拆分）；可以（但不强制）使用ARM NEON/SVE指令集优化CPU侧计算模块；设计CPU与加速器间的零拷贝数据传输方案；

输出成果：

* 支持异构任务调度的完整系统（含自动化部署脚本）
* 技术白皮书说明异构架构设计
* 性能分析报告（需要分析吞吐量和并发量，以及参赛选手硬件配置下的基线数据。）

|  |  |
| --- | --- |
|  | 说明（以下数据需要在加速卡上测算） |
| 基线吞吐量 token/s | 在50个batch时的最大吞吐量 |
| 基线并发量 | 单个batch下，吞吐量>10token/s时的最大并发量 |

评分标准：

* 异构调度策略：推理任务能够在CPU测、GPU测进行动态调度，能够充分利用CPU、GPU资源；需要测试在50个batch下的双侧最大吞吐量和并发量；验收数据包括：在50个batch时的最大吞吐量、保障单batch吞吐量>10tokens/s时的最大并发量；
* CPU侧优化加速效果：在CPU测的性能提升效果，相较于基线数据的性能提升效果，验收数据同上
* 工程质量：
* 代码可维护性（模块化设计、单元测试覆盖率）
* 文档完整性（部署手册、性能调优指南）

创新性提示：鼓励探索鲲鹏920的独特架构特性（如NUMA拓扑）与加速器的深度协同，允许结合openEuler的异构计算框架（如KubeEdge、Volcano）实现资源编排。

赛题联系人：

肖春阳 xiaochunyang2@huawei.com

杜开田 dukaitian@huawei.com

参考资料：

参赛资源支持：

若参赛者使用鲲鹏系列处理器+openEuler操作系统，可由华为公司负责提供相应资源

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136097463933284352](https://www.chaspark.com/" \l "/races/competitions/1136097463933284352)

10.赛题题目：AI for Compiler支持GCC CFGO实现编译选项优化

赛题说明：

GCC CFGO（Continuous Feature Guided Optimization）是指多模态(源代码、二进制)、全生命周期(编译、链接、链接后、运行时、OS/库) 的持续反馈优化；AI for Compiler通过AI成本模型替换启发式成本模型，结合编译选项调优技术，拓展编译器优化机会，提高编译优化算法泛化性，带来应用场景广泛的性能提升。

CFGO静态编译优化当前需要通过编译两次（一次采样，一次优化）进行优化使能，泛化性和易用性有待改进。通过AI for Compiler构建AI模型，在开发环境输入不同客户应用场景特征进行模型训练，在测试/生产环境为CFGO优化提供智能推理决策，减少编译次数从2次->1次，有效提升易用性和泛化性。

赛题要求：

1. 基于GCC12+AI4C框架进行使能

2. 编译器：增强或维护GCC profile\_estimate、profile优化模块等编译器优化遍，无新增功能和性能问题

3. AI4C：完成模型设计、训练算法和推理框架适配等，无新增功能和性能问题

4. 目标应用：Redis 6.0.20

评分标准：

性能提升（35%）：

* 优化后程序性能（如执行速度、资源占用）达到基线（传统两次编译反馈优化）性能的80%，且编译耗时增加不超过50%；

功能完整性（30%）：

* 实现AI模型对CFGO优化的智能决策支持，确保单次编译流程完整可用，兼容openEuler 24.03 LTS及GCC12+AI4C框架；

功能稳定性（20%）：

* 优化过程稳定，优化后程序运行时无异常；

代码规范与可维护性（10%）：

* 代码符合openEuler GCC/AI4C社区编程规范，模块解耦清晰，注释完备，无冗余功能或兼容性风险；

文档质量（5%）：

* 提供模型训练/推理流程、接口设计、性能对比数据等完整设计文档，可完整复现。

赛题联系人：

王鼎 wangding16@huawei.com

参考资料：

1. https://gitee.com/openeuler/AI4C
2. https://gitee.com/openeuler/compiler-docs/tree/master/Meeting&Summit%20Materials/2024%20openEuler%20SIG%20Gathering

参赛资源支持：无

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136098089355956224](https://www.chaspark.com/" \l "/races/competitions/1136098089355956224)

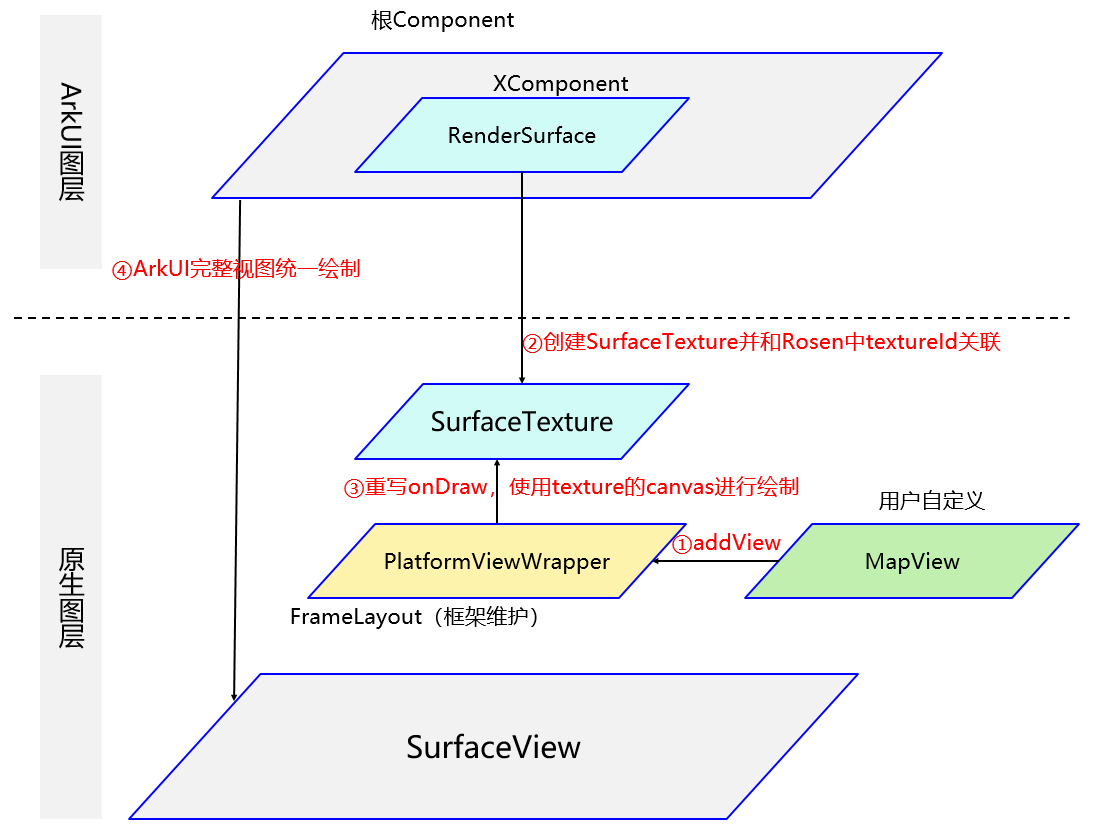
11.赛题题目：基于ArkUI-X实现高性能平台视图功能实现

赛题说明：

ArkUI-X是一个自渲染引擎的跨平台UI框架，可以让开发者使用ArkTS编写跨端UI，目前支持OpenHarmony、HarmonyOS、Android、iOS四个平台，后续会逐步增加更多平台支持。开发者基于一套主代码，就可以构建支持多平台的精美、高性能应用。

平台视图是ArkUI-X跨平台框架中的将原生平台视图嵌入到ArkUI页面，解决跨平台开发过程中需要复用高性能、复杂原生组件(如地图、WebView)的一种技术。

需要在ArkUI-X跨平台框架上，实现高性能的平台视图功能，在Android、iOS平台运行。



同层渲染设计框架

赛题要求：

1. 功能性要求：

基于ArkUI-X框架现有平台视图功能优化或者使用新方案实现平台视图(同层渲染、挖洞等方案)。

支持ArkUI中嵌入原生控件(WebView、地图、视频)。Demo可参考：<https://gitcode.com/arkui-x/samples/tree/master/SuperFeature/PlatformView>。

支持交互如点击、滑动。

支持Android、iOS平台，功能表现一致。

2. 性能要求

帧率：

典型场景支持满帧运行(60帧)，例如地图、视频播放、WebView等复杂组件组件的列表滚动、动画场景。

CPU：

场景1：应用滑动、视频播放、动画场景下:

iOS平台，iphone15 Pro CPU不高于20%，手机不发热。

Android平台，Huawei Mate60 pro CPU不高于15%，手机不发热。

场景2：应用平台视图页面前台静置：

iOS平台，iphone15 Pro CPU不高于10%。

Android平台，Huawei Mate60 pro CPU不高于10%。

内存：平台视图场景下，内存增加不超过15%。

评分标准：

|  |  |  |
| --- | --- | --- |
| **评分项** | **占比** | **说明** |
| 功能完整性 | 30 | 包含创新性10分 |
| 性能优化 | 50 | iOS 性能优化40%、Android 性能优化20% |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

晏国淇 [yanguoqi1@huawei.com](mailto:yanguoqi1@huawei.com)

刘 龙 long.liu@huawei.com

参考资料：

1. ArkUI-X跨平台项目介绍：<https://gitcode.com/arkui-x/docs/blob/master/zh-cn/README.md>
2. ArkUI-X跨平台平台视图：<https://gitcode.com/arkui-x/samples/tree/master/SuperFeature/PlatformView>
3. ArkUI-X 平台视图使用指导：<https://gitcode.com/arkui-x/docs/blob/master/zh-cn/application-dev/tutorial/how-to-use-platformview-on-android.md>
4. <https://gitcode.com/arkui-x/docs/blob/master/zh-cn/application-dev/tutorial/how-to-use-platformview-on-ios.md>

参赛资源支持：

1. OpenHarmony社区可提供：硬件开发平台https://laval.csdn.net/651288d19fe5a0689ae58be6.html?login=from\_csdn

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136099074985222144](https://www.chaspark.com/" \l "/races/competitions/1136099074985222144)

12.赛题题目：基于ARM处理器实现最佳NUMA亲和应用调优

赛题说明：

ARM处理器（譬如鲲鹏920）相较于常见的x86处理器，其CPU拓扑结构与NUMA层级相对复杂。在程序启动和运行过程中，需要在调度和内存管理上充分做到NUMA亲和，才能发挥处理器的最大性能。 举例来说，鲲鹏920处理器一个CPU上有2个DIE，每个DIE上有若干个物理core，构成一个NUMA域。在每个NUMA域当中，若干个物理core又进一步组成一个cluster，进一步增强cache亲和性。此外，不同NUMA域之间访问时延也不同。再考虑到多CPU系统，会进一步加剧这种复杂性。

当前较新的Linux内核（6.12+）支持用户态调度（sched\_ext），在openEuler 24.03 SP1+上也实现了类似的功能。通过用户态调度机制，允许用户在用户空间动态加载自定义的调度策略，结合NUMA亲和的内存管理（分配、迁移、缓存等），可以对运行的应用实现最佳NUMA亲和的调度与内存管理，提升应用性能。

赛题要求：

基于较新的Linux内核（6.12+）或者openEuler 24.03 LTS SP1系统（6.6内核，已实现类似sched\_ext的可编程调度机制）和某一款ARM处理器平台作为执行与验收环境

对典型多进程/多线程业务场景，如MySQL，PostgreSQL，Nginx，Redis（6.x+），Spark，Hadoop等，进行性能优化。不对应用进行修改，只能对OS系统进行优化（如果确实涉及到对GCC、JDK等编译器/运行时的合理配置，可酌情适当进行调整）

通过上游Linux内核提供的sched\_ext接口或者openEuler提供的可编程调度接口实现用户态自定义调度器，实现业务亲和的自适应调度调优能力

改进系统自带的numa balance工具或者实现新的numa调优工具，结合上述用户态调度机制实现对应用的性能优化

可根据需要对内存管理库进行调优。譬如在glibc自带的内存管理库（ptmalloc）或者tcmalloc、jemalloc上进行改进，进一步提升内存分配、释放、缓存等管理能力

用例要包含物理机、裸机容器，虚机容器场景，要支持测试自动化执行。

需要提供完整的测试报告，包含测试用例、测试方法、调优前后数据对比和等内容

评分标准：

* 完成基本功能，包括自定义调度器以及NUMA调优服务，40%
* 性能提升超过10%为基本要求，超过15%为优秀，超过20%为杰出，占30%。特别优秀可额外加分5%~10%
* 完成glibc内存库优化可额外加分5%~10%
* 代码（含代码质量）+ 文档（设计文档、测试验收文档）占10%

赛题联系人：

陈功 [chengong15@huawei.com](mailto:chengong15@huawei.com)

王执 [wangzhi12@huawei.com](mailto:wangzhi12@huawei.com)

参考资料：

1. <https://docs.openeuler.org/zh/docs/24.03_LTS/docs/oeAware/oeAware%E7%94%A8%E6%88%B7%E6%8C%87%E5%8D%97.html>
2. <https://gitee.com/openeuler/oeAware-manager>

参赛资源支持：

1. 如使用鲲鹏处理器，可由华为公司负责提供相应资源

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136101434187567104](https://www.chaspark.com/" \l "/races/competitions/1136101434187567104)

13.赛题题目：基于AI技术的智能桌面管理系统

赛题说明：

本赛题旨在利用 AI 技术优化 Linux 桌面操作系统的操作体验，提高系统组件和应用的响应速度，以及多任务处理效率。传统的窗口管理依赖固定规则或手动配置，难以动态适应不同的应用场景。本项目利用 AI 预测用户行为，优化窗口管理、系统调度、资源分配 等核心环节，实现更流畅、更智能的桌面环境。目前国内开源操作系统已具备多窗口显示的能力，并应用Kwin-x11、wlcom等进行窗口管理，需要在此基础上进行开发工作。

赛题要求：

基于开源桌面操作系统研发及运行。

GPU负载调控

1. 监测 GPU 资源使用情况，主要针对GPU利用率，显存、温度等参数进行监控，在高负载时自动降低窗口透明度、禁用部分动画特效，确保整体流畅性。

2. 在轻负载模式下恢复完整视觉效果，保证美观性与性能平衡。

窗口自动调整

1. AI 预测用户窗口操作模式（切换、拖拽、分屏），自动调整窗口大小与排列，提升多任务处理效率。

2. 结合鼠标轨迹分析，提供智能窗口吸附功能，减少窗口调整的手动操作。

智能焦点窗口管理

1. 结合鼠标/键盘输入预测用户最关注的窗口，在非活跃窗口上自动降低透明度，提升前台窗口的视觉焦点。

2. 后台窗口节能优化：长时间未使用的窗口可降低 CPU 、GPU优先级，减少系统资源占用。

评分标准：

功能完整性（40%）

* 能否智能预测窗口使用模式，并动态调整窗口位置、焦点、透明度（40分）；
* 是否覆盖根据GPU 负载进行显示优化，如当占用率达到80%以上或者温度过高等情况，并能动态调整（20分）；
* AI 是否能够学习用户行为，提高显示优化策略的准确性（40分）。

性能优化（30%）

* 能根据GPU负载动态调整窗口和任务的资源占用（40分）；
* 优化显示流畅无卡顿、对比原系统在GPU负载过高和有长时间未使用的窗口场景下能够明显提升CPU、GPU等资源占用（40分）；
* 需明确说明方案本身的任务所占用的系统资源，如cpu、内存等情况，在满足功能基础上，减少资源占用（20分）。

代码规范性（20%）

* 结构清晰，良好的模块化设计（50分）；
* 符合开源社区开发规范，代码可读性高，易于维护和扩展（50分）。

文档质量（10%）

* 清晰描述 AI 预测模型使用、数据采集方式、调度逻辑（50分）；
* 提供详细的安装、配置、测试方法（20分）；
* 详细说明自己的设计方案及功能亮点（30分）。

赛题联系人：

李卓珩 lizhuoheng@kylinos.cn

参考资料：

1. [https://www.sysgeek.cn/check-gpu-usage-linux/](https://www.sysgeek.cn/check-gpu-usage-linux/" \t "dlt)
2. [https://rocm.docs.amd.com/projects/amdsmi/en/latest/](https://rocm.docs.amd.com/projects/amdsmi/en/latest/" \t "dlt)

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境

14.赛题题目：工业泛在场景下基于国产RISC-V的EtherNet/IP支持

赛题说明：

EtherNet/IP（EtherNet Industrial Protocol）是一种基于标准以太网的工业通信协议，属于 CIP（Common Industrial Protocol） 协议家族，广泛应用于工业自动化中，用于PLC、机器人、传感器等现场设备之间的数据交互，适用于对实时性和可靠性有一定要求的泛在计算场景。开源指令集架构的RISC-V 芯片目前在嵌入式、物联网等众多泛在计算场景得到广泛应用，CH32V307VCT6是一款有代表性的国产高性能 32 位通用 MCU（微控制器）RISC-V芯片。本赛题要求在主控芯片为CH32V307VCT6的开发板上，选定一款国产开源工业物联泛在操作系统，完成EherNet/IP协议的实现。

赛题要求：

操作系统移植成功，正常启动并通过串口打印进程或线程的状态信息，内存的信息，其他存储的信息；

开发板的网口可以正常收发数据，数据包大小以UDP协议和TCP协议支持的MTU(Maximum Transmission Unit)为准，要求持续稳定收发的时间不少于24小时；

实现工业应用层协议EtherNet/IP，分别基于UDP和TCP实现Implicit和Explicit两种通信方式，要求扫描器至少为1个，适配器至少为2个，持续稳定运行的时间不少于24小时。

评分标准：

功能完整性（30%）：

* 在推荐的开发板上完成所选操作系统适配（40分）；
* 按照赛题要求实现工业应用层协议EtherNet/IP（60分）。

应用效果（40%）：

* 设计应用场景，展示协议实现的应用效果（60分）；
* 开展性能测试和优化，稳定性和性能有明显提升的可以加分（40分）。

代码规范性（20%）：

* 代码结构清晰、可维护性高（80分）；
* 符合开源社区规范（20分）。

文档质量（10%）：

* 文档包含设计说明、部署说明、测试报告（50分）；
* 文档条例清晰、撰写规范（50分）。

赛题联系人：

薛栋梁 [xuedongliang010@163.com](mailto:xuedongliang010@163.com)

参考资料：

1. EtherNet/IP: https://www.odva.org/technology-standards/key-technologies/EtherNet-ip/
2. 推荐硬件CH32V307VCT6开发板，不进行任何商业性质推荐，参赛人员自由选择
3. 推荐工业物联泛在操作系统 xiuos <https://gitlink.org.cn/xuos/xiuos>

注意：开发分支为prepare\_for\_master，社区开发者所有操作均在该分支上进行，主分支master不接受合并请求；代码规范性要求请参赛人员联系赛题导师。

参赛资源支持：无

应用创新

1. 赛题题目：基于大语言模型的表单自动填写软件

赛题说明：

基于大模型、NLP或其他人工智能技术，通过对系统中图像、文本、音频等信息进行语义解析和总结提取从而构建用户信息体系，并实现表单的自动填写功能。充分利用操作系统的系统级能力（如文件系统、用户配置接口、设备状态监测、权限控制、安全机制等），优化模型推理性能、数据访问效率与本地化隐私保护能力，软件与操作系统之间的紧密集成与协同。

赛题要求：

支持自动收集操作系统中存储的当前用户相关信息（如个人设置、最近活动、设备状态等），对操作系统用户配置接口、系统调用、文件访问权限等机制的利用，具备一定的系统资源调度意识与采集效率优化策略；

支持基于多模态大模型技术，对操作系统中的文本、图像、音频等信息进行用户信息理解和提取，支持本地化模型部署与推理，合理管理资源使用（如内存、线程、CPU负载），结合系统进程调度机制进行推理性能优化；

支持从用户提供的pdf、doc、xls等格式文件中提取信息；

基于上述用户信息，构建用户信息体系（如姓名、性别、年龄、身份证号、兴趣爱好、个性习惯等），建议以结构化形式组织数据，支持用户信息的长期本地存储与快速调用，同时符合操作系统安全与隐私规范；

根据用户信息体系，支持对doc、xls两类格式的文件自动填充表格内容；

软件基于开源操作系统研发及运行，具备可视化图形界面，利用系统级窗口管理、文件选择、进程通信等原生能力构建高可用图形界面；

软件实现不依赖网络端工具，具备离线状态本地化处理能力，与操作系统中模型管理、缓存策略、之间的协同。

评分标准：

功能完整性（50%）：

* 支持自动收集操作系统中当前用户相关信息（20%）；
* 支持基于多模态大模型技术对文本、图像、音频等进行用户信息理解和提取（20%）；
* 支持从用户提供的pdf、doc、xls等格式文件中提取信息（20%）；
* 支持对doc、xls两类格式的文件自动填充表格内容（20%）；
* 软件实现不依赖网络端工具，具备离线状态本地化处理能力（20%）。

性能优化（20%）：

* 后台占用系统资源（CPU、内存、显存）尽量低，不影响设备正常运行（40分）；
* 系统整体响应速度快，用户操作无延迟（40分）；
* 自动填写准确率达到95%（20分）。

代码规范性（20%）：

* 代码结构清晰、可读性和可维护性高（80分）；
* 符合开源社区规范（20分）。

文档质量（10%）：

* 文档包含设计说明书（50分）；
* 提供完整的部署说明（30分）；
* 提供完成的测试报告（20分）。

赛题联系人：

孙雅彬 [sunyabin@kylinos.cn](file:///C:\\home\\fighting\\文档\\x\\)

参考资料：

1. AI SDK参考说明：【腾讯文档】AI SDK预览版说明文档 https://docs.qq.com/pdf/DTm1FTmN0U3hISnRn
2. 主流大语言模型（LLM）

|  |  |  |
| --- | --- | --- |
| **模型** | **介绍** | **链接** |
| LLaMA 2 | Meta 开源的高效基础大模型 | [https://ai.meta.com/resources/models-and-libraries/llama-downloads/](https://ai.meta.com/resources/models-and-libraries/llama-downloads/" \l "tdsub" \t "dlf) |
| ChatGLM | 清华大学推出的中文优化大模型 | [https://github.com/THUDM/ChatGLM3](https://github.com/THUDM/ChatGLM3" \l "tdsub" \t "dlf) |
| Baichuan | 百川智能推出的中文大模型 | [https://github.com/baichuan-inc/Baichuan2](https://github.com/baichuan-inc/Baichuan2" \l "tdsub" \t "dlf) |
| Mistral | 强调开源与轻量化的 LLM | [https://github.com/mistralai](https://github.com/mistralai" \l "tdsub" \t "dlf) |
| Falcon | 适用于商业用途的大模型 | [https://huggingface.co/tiiuae/falcon-180B](https://huggingface.co/tiiuae/falcon-180B" \l "tdsub" \t "dlf) |
| Gemma | Google 推出的轻量级 LLM | [https://github.com/google/gemma](https://github.com/google/gemma" \l "tdsub" \t "dlf) |

1. 主流多模态模型

|  |  |  |
| --- | --- | --- |
| **模型** | **介绍** | **链接** |
| MiniCPM-Llama3-V2.5 | 适用于多模态理解的 LLM | [https://github.com/OpenBMB/MiniCPM](https://github.com/OpenBMB/MiniCPM" \l "tdsub" \t "dlf) |
| CogVLM | 视觉-语言联合推理模型 | [https://github.com/THUDM/CogVLM](https://github.com/THUDM/CogVLM" \l "tdsub" \t "dlf) |
| BLIP-2 | 高效文本-图像理解模型 | [https://github.com/salesforce/LAVIS](https://github.com/salesforce/LAVIS" \l "tdsub" \t "dlf) |
| LLaVA | 轻量级视觉-语言对齐模型 | [https://github.com/haotian-liu/LLaVA](https://github.com/haotian-liu/LLaVA" \l "tdsub" \t "dlf) |
| GIT(Generative Image-to-Text) | 微软提出的图像到文本模型 | [https://github.com/microsoft/GIT](https://github.com/microsoft/GIT" \l "tdsub" \t "dlf) |
| Whisper (音频转文本) | OpenAI 开发的语音识别模型 | [https://github.com/openai/whisper](https://github.com/openai/whisper" \l "tdsub" \t "dlf) |

1. 文档解析与 OCR（文本、表格、文档处理）

|  |  |  |
| --- | --- | --- |
| **工具/模型** | **介绍** | **链接** |
| PaddleOCR | 强大的 OCR 识别框架 | [https://github.com/PaddlePaddle/PaddleOCR](https://github.com/PaddlePaddle/PaddleOCR" \l "tdsub" \t "dlf) |
| Tesseract OCR | Google 开源的 OCR 识别引擎 | [https://github.com/tesseract-ocr/tesseract](https://github.com/tesseract-ocr/tesseract" \l "tdsub" \t "dlf) |
| pdfplumber | 用于 PDF 文档解析 | [https://github.com/jsvine/pdfplumber](https://github.com/jsvine/pdfplumber" \l "tdsub" \t "dlf) |
| PyMuPDF (fitz) | 轻量级 PDF 解析库 | [https://github.com/pymupdf/PyMuPDF](https://github.com/pymupdf/PyMuPDF" \l "tdsub" \t "dlf) |
| docx (python-docx) | Word 文档解析工具 | [https://github.com/python-openxml/python-docx](https://github.com/python-openxml/python-docx" \l "tdsub" \t "dlf) |
| openpyxl | Excel 解析库 | [https://github.com/pyexcel/pyexcel](https://github.com/pyexcel/pyexcel" \l "tdsub" \t "dlf) |

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境
2. OpenHarmony社区可提供：软硬件开发环境
3. 赛题题目：操作系统源代码的组件许可证识别与合规管理工具开发

赛题说明：

开源许可证的合规性管理是保障操作系统生态健康发展的重要环节。国产开源操作系统一般基于多个上游开源项目进行二次开发，涉及大量跨许可证的代码整合。这些组件分布在不同目录层级中，涵盖许可证类型复杂多样，传统的检测方法效率较低且易遗漏潜在风险。此外，现有工具多针对单个软件项目或软件包，缺乏对操作系统特有场景的专项支持。因此，开发一款能够自动、高效、准确识别操作系统源代码中组件许可证的专用工具，并生成完整的合规报告，对提升国产操作系统的开源合规管理能力具有重要意义。

赛题要求：

本赛题要求参赛者开发一个操作系统源代码的组件许可证识别与合规管理工具，完成以下核心任务：

1. 许可证标准数据集构建：

（1）由于目前缺乏专门针对操作系统源代码组件的开源许可证标准数据集，参赛者需构建一个包含至少100个典型操作系统组件的许可证标注数据集。

（2）数据集应涵盖主流许可证类型及其变体，并标注许可证文本、类型等信息。

（3）该数据集将作为评测基准，并在比赛后开源共享给社区。

2. 许可证提取与识别：

（1）支持对操作系统中常见的组件进行解析，提取许可证相关信息。

（2）实现对主流开源许可证（GPL、LGPL、MIT、Apache等）及其变体和非标准化许可证声明的准确识别（文本级或条款级），能够处理多许可证混合声明的复杂场景（如"GPL-2.0 OR MIT"）。

3. 许可证兼容性分析：支持对不同开源许可证的兼容性进行评估（文本级或条款级），检测潜在的许可证冲突，并提供合规建议。

4. 界面设计与可视化：参赛者可根据需要设计图形化用户界面，展示许可证分析结果和兼容性评估报告，便于用户理解。

评分标准：

许可证数据集构建（15%）：

* 数据集规模（包含至少100个操作系统典型组件）（5%）；
* 数据集多样性（覆盖至少20种主流许可证及其变体）（5%）；
* 标注质量与一致性（5%）。

许可证识别准确率（40%）

* 主流许可证识别准确率≥95%（15%）；
* 非标准化声明及变体的识别准确率≥85%（10%）；
* 自定义许可证的条款级识别率≥80%（10%）；
* 多许可证混合声明处理能力（5%）。

许可证合规性分析（30%）：

* 兼容性冲突检测准确率≥90%（15%）；
* 许可证兼容性冲突条款定位准确率≥90%（10%）；
* 合规建议的可操作性与合理性（5%）。

项目规范与可视化（15%）：

* 许可证分析结果和兼容性评估信息报告完整性与可读性（5%）；
* 符合代码规范，说明文档是否清晰完整，包括工具使用方法和核心技术说明（5%）；
* 用户界面设计与易用性（5%）。

赛题联系人：

赵欣 zhaoxin10@nudt.edu.cn

参考资料：

1. openKylin开源项目：<https://gitee.com/openkylin>
2. SPDX Licenses List：<https://spdx.org/licenses/>
3. tldrlegal：<https://www.tldrlegal.com/>

参赛资源支持：无

1. 赛题题目：面向微服务基准的Trace监控数据采集与异常检测工具

赛题说明：

随着微服务架构的广泛应用，系统的复杂性和规模不断增加，传统的监控和异常检测方法难以应对微服务环境下的动态性和复杂性。本赛题要求基于OpenTracing标准的Trace采集系统（如Jaeger），结合Prometheus等性能监控组件，设计并实现一个面向TrainTicket微服务基准系统的Trace监控数据采集与异常检测系统。通过负载模拟软件（如Locust）和异常注入软件（如ChaosBlade）生成多样化的系统行为数据，利用深度学习方法对采集到的Trace数据进行分析，实现高效的异常检测，为微服务系统的稳定运行提供保障。TrainTicket是一个典型的微服务基准系统，模拟火车票售票服务，其服务间的调用链（Trace）数据包含了丰富的系统行为信息。

赛题要求：

1. 开发环境

操作系统：基于Linux内核，可选择openKylin、openEuler、Ubuntu等发行版。

硬件约束：支持分布式部署，建议使用多节点集群环境。

软件依赖：PyTorch深度学习框架、Prometheus性能监控软件、符合OpenTracing标准的Trace采集系统等；TrainTicket微服务基准系统。

2. 功能要求

* Trace数据采集模块

实现对微服务系统的Trace数据采集。

配置Prometheus，采集系统性能指标（如CPU、内存、网络延迟等）。

* 数据生成模块

模拟多种负载场景，生成不同压力下的系统运行监控数据。

注入异常（如服务超时、资源耗尽等），生成异常场景下的系统运行监控数据。

* 异常检测算法模块

基于深度学习方法（如LSTM、Transformer等），设计并实现异常检测模型。

利用采集到的系统运行监控数据，训练模型并验证其检测效果。

评分标准：

1. 功能完整性（30%）：达成本赛题基本功能要求，包括Trace采集、数据生成、异常检测等模块的实现。

2. 异常注入与数据集收集（20%）：必须包含至少2种不同类型的异常（如服务超时、资源耗尽）每种异常类型需提供明确的注入方法、参数配置和影响程度说明。异常数据需标注明确的开始时间、持续时间和影响范围。

3. 异常检测效果（30%）：以Eadro（参考资料7）论文中提供的开源TrainTicket异常数据集为标准评测数据集，异常检测模型的准确率、召回率、F1-score等指标达到较高水平。具体来说，准确率（10%），即正确识别异常的能力，不低于0.7；召回率（10%），即发现所有异常的能力，不低于0.7；F1-score（10%），即综合评价指标，不低于0.7。提供详细的模型评估报告，包括混淆矩阵、ROC曲线、AUC值等。异常检测模型的训练、验证和测试流程需清晰可复现。

4. 系统性能（10%）：数据采集的性能开销低，对微服务系统的整体影响小于10%。

5. 代码与文档质量（10%）：代码规范，文档包含系统架构图、模块说明、实验报告及结果分析。

赛题联系人：

赵欣 zhaoxin10@nudt.edu.cn

参考资料：

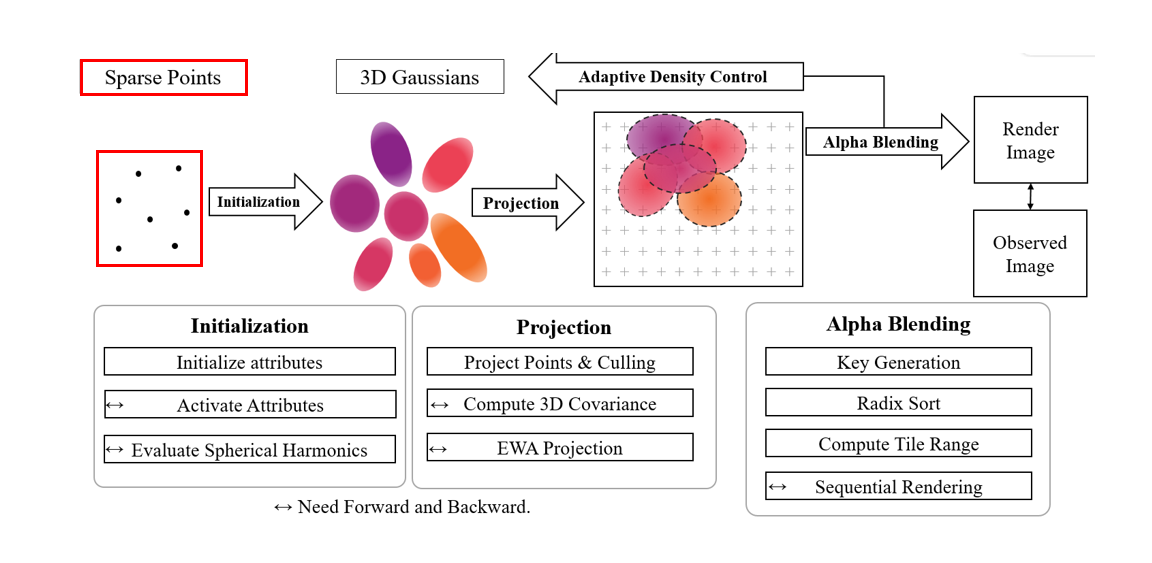
1. Jaeger官方文档（https://www.jaegertracing.io/docs/）
2. Prometheus官方文档（https://prometheus.io/docs/）
3. Locust官方文档（https://locust.io/）
4. ChaosBlade官方文档（https://chaosblade.io/）
5. PyTorch官方文档（https://pytorch.org/）
6. TrainTicket官方文档（https://github.com/FudanSELab/train-ticket）
7. Eadro论文开源的基于TrainTicket的评测数据（<https://zenodo.org/records/7615394>）

参赛资源支持：无

1. 赛题题目：端侧视觉系统点云构建

赛题说明：

实时稠密点云构建技术是计算机视觉和三维重建领域的重要研究方向，旨在通过图像数据实时生成高密度的三维点云模型，稠密点云模型可作为采集完整度的重要依据，也可以作为重要初始化数据参与后续3DGS/Mesh重建。当前已有稠密重建技术较多，基于ACMH的系列方法，基于神经网络的MVSNet系列方法均可以借鉴参考，不限制具体实现方法（鼓励使用不依赖位姿的预测方法），并在开源操作系统上实现。当前挑战是在移动端需要考虑性能与效果的平衡，鼓励充分利用GPU和NPU算力。



稠密点云构建在整个3DGS重建流程中的位置

赛题要求：

需要基于开源操作系统，利用Vulkan计算管线实现，输入相机位姿（端侧SLAM结果）和视频数据，要求在视频时长+10s内完成运算，计算过程中逐渐完善点云结果。提供多个采集数据，计算多个数据集下的平均表现，输入输出部分的文件读写操作不计入时间要求。

输入：

1. 视频文件

2. 时间戳对应的相机位姿（手机端SLAM结果）

3. 相机内参

输出：

视频时间5S后输出稠密点云；每隔5S更新点云。（模拟实时处理过程，不能使用更新时间点以后的视频帧。）

评分标准：

提供一个参考点云模型，通过计算各时间点输出点云与参考点云的Chamfer距离来评价重建质量。Chamfer距离的计算公式为：



其中：

* A是重建点云，B是参考点云。
* ∣A∣和∣B∣分别是点云A和B的点数。
* ∥a−b∥ 是点a 和点b之间的欧几里得距离。

假设视频时长是T，那么总计要求输出N=(T+10)/5（向下取整）个点云，每个点云的得分权重逐渐提高，后一个点云是相邻前一个点云权重的1.3倍。总的点云质量得分如下：

点云质量得分越低代表重建质量越好。

在提供的多个采集数据集上计算平均值，得分最低者胜出。

|  |  |  |
| --- | --- | --- |
| 评分项 | 占比 | 说明 |
| 功能 | 50 | 点云质量 |
| 创新性 | 30 |  |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

赵明伟 zhaomingwei1@huawei.com

参考资料：

1. MVSNet实现：https://github.com/YoYo000/MVSNet
2. ACMH实现参考：https://github.com/GhiXu/ACMH
3. colmap参考实现：<https://github.com/colmap/colmap>
4. 硬件平台Laval手机信息：<https://laval.csdn.net/651288d19fe5a0689ae58be6.html?login=from_csdn>

参赛资源支持：

1. OpenHarmony社区可提供：软硬件开发环境

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136104796112347136](https://www.chaspark.com/" \l "/races/competitions/1136104796112347136)

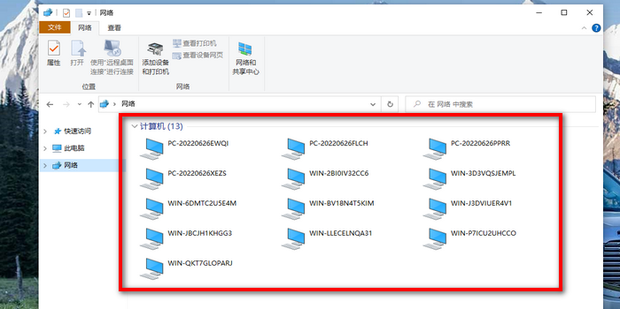
1. 赛题题目：PC网上邻居

赛题说明：

网上邻居能够方便用户轻松地在网络环境下发现和访问同一局域网内的其他设备，现已在windows PC广泛使用。这一功能让用户可以更便捷地共享文件、打印机等资源，提升了多设备协同工作的效率和便利性。这不仅简化了操作流程，也增强了用户体验。实现网上邻居功能将有助于操作系统上更好的体验。

赛题要求：

开发一款网上邻居应用，实现一种局域网内设备发现和文件共享的功能：



1. 设备发现：能够自动搜索并显示同一局域网内的其他设备，包括但不限于其他PC、手机、平板等。

2. 文件共享：允许用户将文件夹设置为共享状态，并通过网络访问其他设备上的共享资源。

3. 设备发现时延满足要求：在不考虑网络拥塞的情况下，1s内搜索到网络中的设备并正确显示设备名称和类型。

4. 设备发现数量满足要求：至少支持同时发现10台以上的设备。

评分标准：

|  |  |  |
| --- | --- | --- |
| **评分项** | **占比** | **说明** |
| 功能 | 40 | 同时满足上述具体要求一、具体要求二 |
| 性能优化 | 30 | 同时满足具体要求三，且4达到500台可获得满分，多台设备可以通过工具发包模拟。 |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

李光远 liguangyuan.li@huawei.com

李锋 lifeng@huawei.com

参考资料：

1. 【OpenHarmony PC适配】https://gitee.com/organizations/openharmony\_pc

https://www.bilibili.com/video/BV1YwCkY9Ewe/?vd\_source=b1ac0c969d08a18600e11cd96550a42c

https://www.uotan.cn/resources/openharmony-5-0-0-x86.680/

1. 【OpenHarmony 统一互联】https://gitcode.com/ohos-oneconnect

参赛资源支持：

1. OpenHarmony社区可提供：RK3568开发板或其他端侧开发板\*2

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136105551870021632](https://www.chaspark.com/" \l "/races/competitions/1136105551870021632)

1. 赛题题目：泛在计算中间件-轻量级上下文感知任务协同系统

赛题说明：

泛在计算场景下，海量异构设备（如传感器、智能终端、边缘网关）需实现高效协同，但面临异构设备管理难、动态任务分配难等问题。本赛题要求设计一个轻量级中间件系统，实现设备自主发现、设备间无缝通信、上下文感知的任务协同，减少通信延迟，降低多设备协同开发门槛。

赛题要求：

1. 基础功能：

* 设备即插即用：在资源(RAM，ROM，CPU算力)相对比较丰富，并且是长上电的设备上部署泛在计算中间件，支持局域网内BLE/Wi-Fi设备（如温湿度传感器、手环、智能灯具、智能音箱等）自动发现和注册，可实时感知设备状态。
* 场景编排与执行：无上行网络条件下，可以通过APP对场景进行编排(即IFTTT场景编排)，编排包含添加、删除、修改、使能和去使能；当用户触发设备或满足特定条件时，中间件进行决策判断，调度执行相应动作，实现上下文感知的任务协同。

2. 创新功能

* 开发上下文敏感控制面板（如靠近摄像头自动弹出手势控制界面）。实现跨设备语音指令接力（如对手机说“打开客厅灯”），实现“按需供能”，延长续航并维持用户体验等

评分标准：

功能完整性（40%）：

* 实现设备发现和注册功能，支持不少于5种设备（25分）；
* 实现场景编排与执行功能（25分）；
* 实现完整的系统，展示基础功能，体验流畅，示例应用具有创新性（50分）。

性能与异常处理（30%）：

* 设备发现速度快，延迟低，上线时延要求:<=1s（25分）；
* 设备上线成功后，满足触发条件时，触发动作时延:<=500ms（25分）；
* 场景编排最大数量 >=500（25分）；
* 能正确处理设备离线、网络异常等问题（25分）。

代码规范性（20%）：

* 代码模块化，逻辑清晰，易于维护（50分）；
* 提供简化的部署流程，减少部署复杂度（50分）。

文档质量（10%）：

* 文档涵盖所有关键步骤和使用说明（50分）；
* 文档语言简洁，结构清晰，易于理解（50分）。

赛题联系人：

蒋炎岩 jyy@nju.edu.cn

胡孝东 [huxiaodong@huawei.com](mailto:huxiaodong@huawei.com)

参考资料：

1. 统一互联：https://gitcode.com/ohos-oneconnect
2. 智慧场景：https://consumer.huawei.com/cn/wholehome/intelligent-scenes/
3. 赛题硬件：rk3568开发板（部署中枢）、hi3863开发板（设备感知、执行设备）
4. 示例工具链

设备发现：BlueZ（BLE）、Avahi（mDNS）

UI开发：openKylin官方QT/GTK组件

压力测试：使用openKylin内置的stress-ng模拟高负载场景

https://github.com/home-assistant/core

https://github.com/openhab/openhab-core

<https://github.com/mozilla-iot/gateway>

参赛资源支持：

1. 麒麟软件公司提供rk3568开发板（部署中枢）、hi3863开发板（设备感知、执行设备）
2. OpenHarmony社区可提供：rk3568开发板（部署中枢）、hi3863开发板（设备感知、执行设备）

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136106641680224256](https://www.chaspark.com/" \l "/races/competitions/1136106641680224256)

1. 赛题题目：端边协同AI聊天机器人

赛题说明：

AI技术快速发展，给人们带来了诸多好处：既提高了生产效率，又丰富了娱乐生活。由于AI对硬件的算力有一定要求，部分设备不具备端侧AI能力。请基于端边协同，将一个无算力端侧设备和一个有算力边缘设备结合，使无算力设备能够借用有算力设备的算力，完成端侧AI语音聊天功能。

赛题要求：

1. 实现一个端侧AI聊天功能，要求有文字和语音输入&输出。

2. 端侧设备实现交互，边缘设备提供算力。两设备间使用无线连接。

3. 边缘设备可以部署开源模型，端侧交互代码自主实现。

评分标准：

|  |  |  |
| --- | --- | --- |
| **评分项** | **占比** | **说明** |
| 功能 | 30 | 满足上述具体要求前两项，中文语音识别准确，要求干净环境词错误率<5%，响应时延<500ms。语音输出采用流式输出，边生成边播放。 |
| 性能优化 | 30 | 前后端设备都未休眠的情况下，语音聊天端到端响应时延小于2秒；后端设备休眠的情况下，设备唤醒时间不大于1秒； |
| 稳定性 | 20 | 设备在一般噪声环境（15dB SNR），连续运行3小时不断线。 |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

刘小飞 splane.liu@huawei.com

徐梓荐 [xuzijian1@huawei.com](mailto:xuzijian1@huawei.com" \t "_blank)

参考资料：

1. 分布式硬件子系统https://gitee.com/openharmony/docs/blob/master/zh-cn/readme/分布式硬件子系统.md
2. 分布式硬件管理框架https://gitee.com/openharmony/distributedhardware\_distributed\_hardware\_fwk

参赛资源支持：

1. OpenHarmony社区可提供：Rk3568开发板+Laval开发者手机

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136107327893524480](https://www.chaspark.com/" \l "/races/competitions/1136107327893524480)

1. 赛题题目：实时视频直播跨平台应用开发

赛题说明：

ArkUI-X是一个自渲染引擎的跨平台UI框架，可以让开发者使用ArkTS编写跨端UI，它具备简洁自然的UI信息语法、丰富的UI组件、多维的状态管理，以及实时界面预览等相关能力，帮助提升应用开发效率，并能在多种设备上实现生动而流畅的用户体验。目前支持OpenHarmony、HarmonyOS、Android、iOS四个平台，后续会逐步增加更多平台支持。开发者基于一套主代码，就可以构建支持多平台的精美、高性能应用。

实时视频流播放在直播、安防等场景下应用非常普遍，本赛题要求参赛者使可以使用Xcomponet和ffmpeg开发一个基于ArkUI-X跨平台框架的实时视频直播应用，直播的视频流来源不限制，需要使用如RTP/RTSP协议接收。

赛题要求：

1. 功能性要求：

参赛者可以基于Xcomponet和ffmpeg实现实时视频码流播放。

视频要求必须是实时视频，分辨率720P/1080P，需要使用RTP/RTSP等传输协议。

2. 性能要求

实时视频帧率在15-30FPS。

视频播放时CPU低于30%。

评分标准：

|  |  |  |
| --- | --- | --- |
| **评分项** | **占比** | **说明** |
| 功能 | 30 |  |
| 性能 | 30 | 流畅度 |
| 跨平台代码复用 | 20 |  |
| 代码规范性 | 10 | 代码符合OpenHarmony社区规范，按照规范类型扣分点，每条减少1分，直至为0 |
| 文档质量 | 10 | 文档能将从架构和技术实现角度，说明架构和技术竞争力。 |

赛题联系人：

晏国淇 [yanguoqi1@huawei.com](mailto:yanguoqi1@huawei.com)

刘 龙 long.liu@huawei.com

参考资料：

1. ArkUI-X跨平台项目介绍：https://gitcode.com/arkui-x/docs/blob/master/zh-cn/README.md
2. XComponent介绍：https://developer.huawei.com/consumer/cn/doc/harmonyos-references-V5/ts-basic-components-xcomponent-V5

参赛资源支持：

1. OpenHarmony社区可提供软硬件开发资源

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136107857761329152](https://www.chaspark.com/" \l "/races/competitions/1136107857761329152)

1. 赛题题目：基于AI的系统参数智能调优技术

赛题说明：

Linux 系统参数调优是提升系统性能、降低资源消耗的重要手段，然而，面对复杂且多样化的工作负载，手动调优过程耗时且依赖于个人经验，难以快速适配动态变化的需求。近年来，人工智能技术为系统参数优化提供了新的思路，能够基于历史数据和实时监测信息，动态调整系统关键参数，提升系统的稳定性与执行效率。本赛题旨在探索基于AI的 Linux 系统参数自动调优技术，推动国产操作系统与 AI 技术的深度融合，助力系统性能的智能化提升。选手需针对特定的系统参数（如 CPU 调度策略、内存管理参数、I/O 队列深度、网络协议栈参数等）进行智能优化，开发一个能实时监控、分析工作负载并生成最优参数组合的调优系统。

赛题要求：

基于开源操作系统研发及运行。

动态监控与分析：实时采集和分析CPU 相关参数（如调度策略、CPU 亲和性、CFS 调度器参数）、内存管理参数（如虚拟内存回收策略、内存页大小、缓存管理）、I/O 相关参数（如 I/O 队列深度、磁盘读写策略）、网络参数（如 TCP 拥塞控制算法、socket 缓冲区大小）等关键指标，识别当前工作负载类型。

智能参数优化：基于AI技术，结合历史数据和实时反馈，动态生成并应用最优系统参数组合，支持对多种工作负载场景（如CPU密集型任务、数据密集型任务、IO密集型任务等）进行适配和调优；

用户友好界面：提供可视化配置界面，支持一键优化、参数自定义调节及调优前后效果对比等功能。

可扩展性：方案应具备良好的扩展性，方便新增系统参数或优化策略。

评分标准：

功能完整性（40%）：

* 监控指标是否全面与准确（50分）；
* 界面设计的是否直观与易用（30分）；
* 方案是否可扩展（20分）。

性能优化（30%）：

* 性能提升幅度：通过关键指标（如 CPU 利用率、内存使用率、I/O 吞吐量、网络延迟等）对比，优化后性能提升幅度至少达到 10%（70分）；
* 适配多样工作负载场景：针对不同工作负载（如计算密集型、I/O 密集型、混合负载）均需展示明显的性能改善，至少覆盖两种负载类型（30分）。

代码规范性（20%）：

* 代码结构清晰优雅，代码模块化设计（70分）；
* 可维护性高，可扩展性好（30分）。

文档质量（10%）：

* 给出需求分析、设计说明文档。描述作品背景、应用领域、架构设计思路、数据采集方式、AI模型使用、模块功能描述等（60分）；
* 给出测试用例文档。结合不同性能优化指标检测方案测试优化率（20分）；
* 给出软件使用指南文档（20分）。

赛题联系人：

余烁奇 yushuoqi@kylinos.cn

参考资料：

1. [https://www.oschina.net/news/162574](https://www.oschina.net/news/162574" \t "dlt)
2. [https://qp6kkktqa2.feishu.cn/wiki/AgsvwQSxlinIBHkgB2VcpUf3nAe](https://qp6kkktqa2.feishu.cn/wiki/AgsvwQSxlinIBHkgB2VcpUf3nAe" \t "dlt)
3. [https://my.oschina.net/emacs\_8898893/blog/17538774](https://my.oschina.net/emacs_8898893/blog/17538774" \t "dlt)
4. [https://zhuanlan.zhihu.com/p/652434521](https://zhuanlan.zhihu.com/p/652434521" \t "dlt)

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境
2. OpenHarmony可提供软硬件开发资源
3. 赛题题目：基于大模型的操作系统智能健康检查与自动修复

赛题说明：

利用大模型构建智能操作系统的健康检查和故障自动修复机制，基于本地知识库构建智能诊断系统，自动检测操作系统及行业应用（如 OA 系统）运行状态，分析故障原因，并尝试修复或提供优化建议。

赛题要求：

* 推荐硬件环境：

联想M90F（CPU: D3000内存：16G独立显卡：Arise-GT10C0t硬盘：512G）

联想M90H（CPU: Hygon C86-3G (OPN:3350) 内存：16GB独立显卡：Arise-GT10C0t硬盘：512GB）；

* 自动收集操作系统日志或行业应用运行日志；
* 基于大语言模型对系统日志和应用日志进行分析，实时识别系统或应用的运行状态，发现系统或应用故障；
* 构建操作系统和行业应用运行状态相关的本地知识库；
* 使用大模型，根据故障日志和本地知识库，输出修复方案；
* 调用系统或应用API自动完成故障修复；

评分标准：

功能完整性（40%）：

* 软件基于开源操作系统研发及运行（20分）；
* 实时对系统日志和至少两款应用日志进行分析，识别系统和应用的运行状态，发现系统或应用故障，准确率90%以上（20分）；
* 使用向量化模型及向量数据库完成操作系统和行业应用运行状态相关的本地知识库构建（20分）；
* 可根据系统状态和故障，结合本地知识库给出修复方案，方案正确率90%以上（30分）；
* 80%以上故障可调用系统或应用API自动完成修复（10分）。

注：标准评测集和测试用例短时间内无法提供，预计提供1000条评测集和测试用例，需要3周时间完成评测集和测试用例的设计和构建。

性能优化（30%）：

* 系统日志和应用日志分析速度快，不影响用户体验，问题故障分析报告输出延迟小于1分钟（30分）；
* 整合系统资源（CPU、内存、显存）占用低，整体占用50%以下，不影响设备正常运行（40分）；
* 系统整体响应速度快，用户操作无延迟（30分）。

代码规范性（20%）：

* 代码结构清晰、符合开源社区规范、可维护性高（80分）；
* 符合开源社区规范（20分）。

文档质量（10%）：

* 文档包含概要设计说明书，详细设计说明书，以及部署说明（50分）；
* 提供完整的部署说明（30分）；
* 提供完成的测试报告（20分）。

赛题联系人：

孙雅彬 [sunyabin@kylinos.cn](file:///C:\\Users\\jenny\\Desktop\\操作系统大赛题目-2025\\2025.3.31赛题更新\\0411\\sunyabin@kylinos.cn" \t "dlt)

参考资料：

1. [https://zhuanlan.zhihu.com/p/675509396](https://zhuanlan.zhihu.com/p/675509396" \t "dlt)

**注：**标准评测集和测试用例短时间内无法提供，预计提供1000条评测集和测试用例，需要3周时间完成评测集和测试用例的设计和构建。

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境
2. OpenHarmony可提供软硬件开发资源
3. 赛题题目：视频字幕AI自动识别显示工具

赛题说明：

随着短视频、直播、在线教育等场景的爆发式增长，视频内容的可访问性和跨语言传播需求日益增强。传统字幕制作依赖人工转录，效率低且成本高。近年来，基于深度学习的语音识别（ASR）、自然语言处理（NLP）显著提升了字幕生成的自动化水平。AI字幕技术通过自动识别音频、生成精准字幕并支持多语言翻译，正在重塑视频创作与传播方式。

本赛题旨在推动AI技术在视频处理领域的创新应用，优化音视频播放逻辑，同时探索跨语言适配、实时处理等关键技术难点。该方向与“智能技术应用与创新”赛道目标高度契合，可促进教育、娱乐、跨国协作等场景的智能化升级。

赛题要求：

* 软件基于开源操作系统研发及运行；
* 核心功能：基于操作系统桌面接口或SDK开发工具实现视频的语音识别、字幕自动生成及时间轴同步，支持中英等多语言翻译切换；
* 性能要求：字幕生成准确率≥90%，支持实时或准实时处理（延迟<3秒），对字幕获取处理流程输出系统资源使用情况；
* 扩展功能：支持字幕样式自定义（字体、颜色、位置）、多说话人分离、背景噪音过滤。

评分标准：

功能完整性（40%）：

* 基于操作系统桌面接口或SDK开发工具实现视频播放（20分）；
* 可以显示视频字幕（30分）；
* 可以显示多种语言字幕（30分）；
* 支持多种扩展功能（字幕字体，颜色，位置调整，多说话人分离等）（20分）。

性能优化（30%）：

* 字幕识别准确率超过90%（55分）
* 准确率低于60%（0分）；
* 准确率高于60%，低于70%（35分）；
* 准确率高于70%，低于80%（45分）；
* 准确率高于80%，低于90%（55分）。
* 字幕识别延迟小于1s（35分）
* 字幕识别延迟大于1s，小于2s（35分）；
* 字幕识别延迟大于s2，小于3s（25分）；
* 字幕识别延迟大于3s（0分）。
* 对字幕获取处理流程输出系统资源使用情况（10分）
* 输出字幕获取过程的系统资源情况，内存、CPU、显存等信息（10分）。

代码规范性（20%）：

* 代码目录结构清晰，易读，可维护性强（50分）；
* 符合开源社区规范（50分）。

文档质量（10%）：

* 概要设计说明书、测试设计说明书（50分）；
* 申报书、用户手册和安装说明（50分）。

赛题联系人：

于恒 yuheng@kylinos.cn

参考资料：

1. FFmpeg多媒体处理框架

官网：[https://github.com/linyqh/NarratoAI](https://github.com/linyqh/NarratoAI" \t "dkey)[https://github.com/mpv-player/](https://github.com/mpv-player/" \t "dkey)org

核心功能：音视频流提取、格式转换与时间轴同步；支持多轨道合成（字幕、配音、视频），为AI字幕工具提供底层处理能力

1. MPV播放器矿框架

项目地址：https://github.com/mpv-player/

核心功能：集成音视频播放能力，便于开发；

1. NarratoAI（智能视频解说与字幕生成）

项目地址：https://github.com/linyqh/NarratoAI

相关技术：基于Whisper的语音识别与时间轴同步技术；

多语言翻译（支持Qwen2-VL模型）与字幕-视频自动合成（FFmpeg集成）；

实时处理优化方案，支持GPU加速降低延迟。

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境
2. 赛题题目：泛在开源操作系统第三方软件包自动化升级工具

赛题说明：

当前的操作系统功能日益复杂，为了实现快速迭代和创新，操作系统在开发过程中高度依赖大量第三方开源软件包，形成了复杂的供应链网络。为了提供更完善的功能、修复潜在的安全漏洞，这些第三方软件包不断进行版本更新。因而，要求操作系统必须及时跟进更新所依赖的第三方软件包，以确保系统的安全性与功能性。然而，在第三方软件包的更新过程中，可能会出现API变更，即所谓的breaking changes，这可能导致操作系统与所依赖的软件包之间出现兼容性问题。目前，第三方软件包的更新工作主要依靠手动完成，这种方式不仅效率低下，而且容易导致更新滞后，增加了系统面临安全风险的可能性。为此，本赛题的目标是开发一款能够自动更新第三方软件包的工具，支持主流开源操作系统的软件包升级，并将工具集成到开源代码托管平台中，从而为用户提供自动化生成的包更新Pull Request（PR），旨在解决现有更新流程中的滞后性和效率低下的问题，提高系统的安全性和功能性。

**赛题要求：**

* **目标平台**：基于开源代码托管平台，实现对开源操作系统的第三包的自动化升级。
* **第三方软件包自动化更新功能设计**：
* 给定目标操作系统项目的版本库地址，自动获取项目代码；
* 提取目标操作系统项目所依赖的第三方软件包列表；
* 构建目标操作系统项目与第三方软件包的API使用调用图；
* 分析第三方软件包更新过程中产生的breaking changes，识别升级过程中的API兼容性问题（如参数错误、错误处理缺失、版本兼容性问题）；
* 给出API修复建议或代码补丁，并支持人工确认/自动应用。
* 与代码托管平台集成：
* 将上述工具以SoftBot形式集成到GitLink代码托管平台上，支持为GitLink平台上托管的各类操作系统项目自动生成第三方软件包升级Pull Request。

评分细则：

|  |  |  |
| --- | --- | --- |
| **项目** | **分值** | **说明** |
| 功能完整性 | 40 | 包括 API 图构建、误用检测、修复建议生成； |
| 应用效果 | 30 | 提供具体修复示例、内核稳定性提升、对比实验等；支持的项目数越多越能得高高分。 |
| 代码规范性 | 20 | 模块结构清晰、易扩展，遵循开源社区规范； |
| 文档质量 | 10 | 包括设计说明、使用说明、修复示例和评估报告； |

* 额外加分项（每项加3分）：
* 将上述工具以SoftBot形式集成到GitLink开源代码托管平台；
* 实现对GitLink平台泛在操作系统专区（泛在操作系统）项目的支持。

赛题联系人：

高祥 [xiang\_gao@buaa.edu.cn](xiang_gao@buaa.edu.cn" \t "dkey)

王涛 taowang2005@nudt.edu.cn

参考资料：

1. SoftBot开发使用指南：https://help.gitlink.org.cn/Bot%E5%B8%82%E5%9C%BA/
2. Laerte Xavier, et.al. Historical and impact analysis of API breaking changes: A large-scale study, Saner 2017.
3. Xiang Gao, et.al. APIfix: output-oriented program synthesis for combating breaking changes in libraries. OOPSLA, 2021.

参赛资源支持：无

1. 赛题题目：开源操作系统软件仓库依赖关系检测分析工具

赛题说明：

开源已成为全球主流的软件开发模式之一，全球超96%的代码库中包含开源软件。目前国产操作系统多采用开源的Linux技术体系，然而由于开源软件质量参差不齐，供应链安全风险日益突出，而依赖关系是软件供应链的一种关键影响因素。开源开发模式下，操作系统开发者多通过使用已有的第三方组件对系统进行扩展和改进，而第三方组件也会依赖其他的组件，因此形成了复杂的依赖链条。由于操作系统的版本演化以及各个组件的不断更新升级，开源软件的规模不断扩大，以依赖关系为基础的供应链分析也越来越复杂。本赛题聚焦开源操作系统软件仓库的依赖关系检测分析，要求参赛者基于开源操作系统发行版的软件仓库，构建覆盖软件包、文件和调用接口等多级关联信息的全量知识图谱，并开发自动化分析工具对软件仓库中的依赖关系、函数调用关系以及各对象之间包含关系等进行分析，找出潜在的兼容性风险和依赖关系缺陷。

赛题要求：

选择一个开源操作系统发行版（如openKylin、openEuler等），选择至少2个主要版本的软件仓库，覆盖软件包数量不少于3万个；

建立面向软件仓库的动态库接口的全量知识图谱，图谱中至少包含软件包、动态库文件和函数接口等实体，包含软件包依赖、函数接口调用、实体间包含等关系；

开发依赖关系分析工具，工具能够通过函数接口的调用关系推理软件包的依赖关系是否正确、完备；检测操作系统升级后软件包是否兼容，分析内容包括正向检测（软件包A在升级后的操作系统版本中依赖关系是否完整）和反向检测（软件包A升级后依赖软件包A的其他软件包是否兼容）。开发语言不限。

提供可视化的分析结果展示，支持分析报告的导出。

评分标准：

技术完整性（30）：

支持RPM包/DEB包的格式解析、依赖关系可视化完整度，是否具有跨架构检测能力；

实用性（30）：

工具的性能（单个软件包分析时效性）、工具的易用性；

数据的展示型（20）：

可视化效果；

报告质量（20）：

文档的完整度、逻辑清晰性

赛题联系人：

王静 wangjing@nudt.edu.cn

参考资料：

1. https://www.debian.org/doc/debian-policy/ch-relationships#binary-dependencies-depends-recommends-suggests-enhances-pre-depends

参赛资源支持：无

研究创新

1. 赛题题目：面向大语言模型高效推理框架的系统级优化与实现

赛题说明：

随着大语言模型在边端侧的应用不断拓展，推理服务的重要性日益凸显，其效率直接关乎用户体验，因此，迫切需要对边端侧大模型的推理效率进行优化。目前，已有不少推理框架与架构从不同维度对大模型推理展开优化工作。比如，vLLM运用PagedAttention技术，显著提升了显存利用率；llama.cpp借助量化技术、硬件加速及跨平台兼容性，有效降低了大模型的部署门槛。然而，鉴于边端侧环境普遍存在端到端延迟敏感、资源受限等特性，现有系统仍需围绕这些痛点，开展更具针对性的优化举措。本赛题要求基于开源技术栈，构建具备端到端优化能力的大模型推理框架。

赛题要求：

完成面向大语言模型推理框架的系统级端到端推理效率优化。可选vLLM、llama.cpp等主流推理框架。硬件不做固定要求，但要求优化前后保持相同硬件配置。使用开源大模型（如MiniCPM、Qwen等）作为测试基准并提供可复现的Benchmark测试方案及性能对比报告。同时要求不损失模型推理精度，优化前后软件环境也需保持一致。具体优化推荐但不限于以下方向：

* 推测解码技术：采用小型draft model生成候选token，通过target LLM并行验证加速解码。
* 注意力计算加速技术：优化注意力机制的计算效率，通过算法改进或数据布局优化提升推理速度。
* 内存管理技术：实现显存的智能分配与回收机制，提升资源受限环境下的内存利用率。

评分标准：

功能完整性（30%）：

* 完整实现至少一个推理框架的适配与优化（60分）；
* 支持多模型部署能力（40分）。

应用效果（40%）：

* 端到端推理延迟降低幅度（40分）；
* 能够在不同模型或框架的适配（40分）；
* 切实解决边端侧场景的典型问题（20分）。

代码规范性（20%）：

* 代码结构清晰、符合开源规范（80分）；
* 测试覆盖完备（20分）。

文档质量（10%）：

* 包含技术方案、部署指南、测试报告（50分）；
* 文档逻辑清晰、格式规范（50分）。

赛题联系人：

张建锋 zhangjianfeng@nudt.edu.cn

参考资料：

1. <https://docs.vllm.ai/en/stable/index.html>
2. <https://github.com/vllm-project/vllm>
3. <https://github.com/ggerganov/llama.cpp>
4. <https://www.modelscope.cn/models/OpenBMB/MiniCPM3-4B>

参赛资源支持：无

1. 赛题题目：面向Linux的分布式异构GNN负载均衡优化

赛题说明：

随着图神经网络（GNN）在社交网络、推荐系统等领域的广泛应用，分布式训练已成为处理大规模图数据的必要手段。然而，分布式异构环境中节点性能差异显著，传统负载均衡算法难以有效适应异构算力下GNN计算特性，导致计算资源浪费和效率低下。本赛题要求基于Linux系统，结合PyTorch与图计算框架（PyG/DGL），设计并实现一种面向GNN任务的负载均衡调度优化方法，通过评估各节点的系统性能指标（如计算能力、内存、网络带宽等），解决异构环境下GNN任务分配与资源动态适配的难题，提升整体计算效率，为智能时代的高效分布式计算提供技术支持。

赛题要求：

1. 开发环境

操作系统：基于Linux内核，可选择openKylin、openEuler、Ubuntu等发行版。

硬件约束：分布式环境由算力异构设备组成。

软件依赖：使用PyTorch 2.0+，PyG 2.4+或DGL 1.0+。

GNN模型：GraphSAGE。

数据集：ogbn-products。

集群规模：要求集群规模至少为两个节点，且存在算力异构，算力异构差值不低于40%（异构差值以执行基线标准计算，增加节点加分，见评分标准6）。

异构算力平台：CPU、GPU均可，使用国产硬件平台加分（见评分标准6）。

2. 功能要求

针对分布式GNN异构算力节点负载不均衡的问题，提出一种解决方案，以优化GNN在分布式环境下负载均衡水平，提高计算效率。

基线标准：2节点（节点之间异构算力差距明显）+ 模型GraphSAGE（2层采样，两层邻居采样数设置为“10,10”） + 数据集ogbn-products + 图划分算法使用METIS，节点子图采用均分+运行50个epoch所取的每个节点时间和全局时间的平均值。

泛化性：支持多节点集群规模拓展，模拟环境或真实集群上进行实验，验证所设计算法的性能和效果，并提供实验数据和分析报告。

评分标准：

评分将从以下五个维度进行综合评估：

1. 功能完整性（25%）：评估参赛队伍开发的系统是否达成本赛题的基本功能要求。系统应具备全面的功能模块，涵盖核心功能与辅助工具，以确保在实际应用中能够高效、准确地完成任务。此外，还应包含必要的错误处理和日志记录功能，以保障系统的稳定性和可维护性。

2. 负载均衡效果（25%）：此标准旨在确保集群中各节点的工作负载均匀分布，避免部分节点过载而其他节点闲置，从而提高整体运行效率与响应速度，保障系统稳定运行，提升用户体验。（与所数据集节点数平均分配相比）

具体评分标准为：

0%≤节点间负载标准差＜5%：负载极为均衡，各节点工作负载分配几乎完全一致，集群效能达到最优状态，得25分；

5%≤节点间负载标准差＜15%：负载较为均衡，各节点工作负载分配基本均匀，集群效能良好，得20分；

15%≤节点间负载标准差＜25%：负载有一定差异，但整体仍处于可接受范围内，集群效能中等，得15分；

25%≤节点间负载标准差＜35%：负载差异较为明显，部分节点可能面临一定压力，集群效能稍受影响，得10分。

节点间负载标准差≥35%：负载极不均衡，部分节点过载而其他节点闲置，严重影响集群效能和系统稳定性，得0分。

3. 计算效率提升（25%）：要求集群平均资源利用率相较于原始状态有显著改善，通过优化资源调度策略、合理分配计算、存储和网络资源，减少资源浪费，确保资源能被充分利用，同时满足不同任务的需求，提升系统的整体效益。（与所数据集节点数平均分配相比）

具体标准为：

整体计算效率提升≥40%：计算效率大幅提升，优化效果显著。通过优化资源调度策略、合理分配计算、存储和网络资源，任务执行时间大幅缩短，系统整体效益显著提高，得25分；

30%≤整体计算效率提升＜40%：计算效率提升较高，优化效果良好。资源调度较为合理，任务执行时间明显缩短，系统整体效益有较大提升，得20分；

20%≤整体计算效率提升＜30%：计算效率有一定提升，优化有一定效果。资源调度基本合理，任务执行时间有所缩短，系统整体效益有所改善，得15分；

10%≤整体计算效率提升＜20%：计算效率有少量提升，优化效果一般。资源调度存在改进空间，任务执行时间略有缩短，系统整体效益略有改善，得10分；

整体计算效率提升＜10%：计算效率提升不明显，优化效果不显著。资源调度策略改进有限，任务执行时间缩短不明显，系统整体效益提升不明显，得0分。

4. 可扩展性（15%）：评估系统架构的灵活性与适应性，强大的可扩展性使系统能适应功能增长与变化。系统需支持接入多种图划分策略，以便根据实际需求灵活调整图的划分方式，优化数据分布与处理流程，从而提高系统在处理大规模数据和复杂任务时的性能与效率。

具体评分标准为：

在PyG 或 DGL中集成图划分算法调用接口，可以将设计的图划分算法集成于PyG 或 DG中，通过简单的接口调用，即可灵活选择和应用多种图划分策略，显著提升了系统的可扩展性和易用性，得15分；

支持图划分算法调用接口，但不完整，功能有限，虽然有一定的可扩展性，但在灵活性和易用性上不如完全集成的情况，得10分；

不支持图划分算法调用接口，需自行实现：系统未集成任何图划分算法调用接口，用户需要自行实现图划分算法或手动集成外部库，系统的可扩展性较差，开发和维护成本较高，得0分。

5. 代码与文档质量（10%）：代码的规范性和可读性是项目长期维护与迭代的关键，良好的文档则有助于用户理解和使用系统。代码应遵循开源社区的代码风格要求，结构清晰、注释详尽，便于后续的维护与功能扩展。文档需包含系统架构图，直观展示系统各组件的结构与交互关系；使用说明应详细准确，指导用户顺利安装、配置和操作系统的各项功能。

6. 加分项：

使用国产硬件算力平台加3分；

最低2个节点，每增加1个节点加1分，最多不超过5分。

赛题联系人：

赵欣 zhaoxin10@nudt.edu.cn

参考文献：

1. PyTorch官方文档（https://pytorch.org）
2. PyG官方教程（https://pytorch-geometric.readthedocs.io/en/latest/index.html）
3. DGL官方教程（https://www.dgl.ai/）
4. 开源代码：参考PyG与DGL开源示例代码。
5. 论文搜索关键字：建议GNN（图神经网络）、graph partitioning（图划分） 和 heterogeneous（异构）等。

参赛资源支持：无

1. 赛题题目：智能体操作系统中的多智能体协作技术

赛题说明：

多智能体协作系统借鉴了人类社会的协作原则，通过分工合作，结合各个智能体的优势，完成一些超出单个智能体能力范围的任务。例如，多智能体协作系统可以用于模拟和优化交通、能源、物流等领域的复杂系统，也可以用于设计和实现智能家居、智能城市、智能工厂等应用场景。随着多智能体协作系统的愈发复杂，在多智能体协作系统中实现高效的任务调度与协作，成为推动通用人工智能落地的重要课题。为此，AIOS应运而生。AIOS 是一个开源的大模型智能体操作系统，包含专门设计的大模型内核、智能体调度器、上下文管理器、内存与存储管理器、工具管理器以及访问控制机制，可有效管理大模型及外部工具等系统资源。基于此架构，AIOS 能显著提升智能体的任务执行效率，实测可达最高 2.1 倍的性能提升。通过对资源分配和大模型上下文切换的优化，AIOS 不仅在当前环境中展现出高效率和高可扩展性，也为未来实现更高水平的通用人工智能奠定了基础。

赛题要求：

本赛题基于 “Linux + AIOS + 端侧大模型（7B或8B）” 的环境，要求参赛队伍在 AIOS 框架内实现对多智能体协作的支持。具体要求如下：

1. 参赛队伍自行拟定单智能体无法完成的复杂任务，并围绕该任务使用 AIOS-Agent SDK 设计多智能体协作系统，应包含不少于4种可协作的智能体。

2. 复杂任务应可分解为不少于4个可并行执行的子任务，根据子任务的特性，多智能体协作系统应将子任务自动分配给不同的智能体完成子任务。

3. 设计合理的多智能体协作模式，实现对复杂任务的自动化执行。协作流程中，不少于3对不同类型智能体之间应存在双向交互。

4. 多智能体协作过程中应考虑并满足子任务执行的先后顺序及依赖关系的约束，确保整体流程有序进行。

评分标准：

功能完整性（50%）：

* 自行拟定的复杂任务合理，满足赛题对任务场景的要求（40分）；
* 多智能体的实现及整体协作流程设计满足赛题要求（60分）。

应用效果（50%）：

* 与多智能体串行完成任务相比，在智能体操作系统中并发完成任务的总时间应减少不少于20%（50分）；
* 多智能体协作应能正确完成任务（50分）。

赛题联系人：

纪斌 jibin@nudt.edu.cn

涂宏魁 tuhkjet@foxmail.com

刘美燕 liumeiyan@kylinos.cn

参考资料：

1. AIOS开源代码库：https://github.com/agiresearch/AIOS
2. 论文：《AIOS: LLM Agent Operating System》（https://arxiv.org/abs/2403.16971）
3. 论文：《Multi-Agent Collaboration Mechanisms: A Survey of LLMs》（https://arxiv.org/pdf/2501.06322）
4. 开源代码库：Optimized Workforce Learning for General Multi-Agent Assistance in Real-World Task Automation（https://github.com/camel-ai/owl）

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境
2. 赛题题目：面向开源操作系统的定向模糊测试框架开发与漏洞挖掘

赛题说明：

随着开源软件在各个领域的广泛应用，其安全性问题日益受到关注。开源操作系统通常具有复杂的代码结构和多样化的功能模块，容易隐藏各种安全漏洞。模糊测试作为一种有效的漏洞检测技术，通过向软件输入随机或畸形数据，能够高效地发现潜在的安全问题。然而，传统的模糊测试方法往往缺乏针对性，效率较低。定向模糊测试通过结合软件的特定功能模块和已知漏洞模式，能够显著提高漏洞挖掘的效率和准确性。本赛题旨在开发一个针对开源操作系统的定向模糊测试工具，通过对软件的特定功能模块进行精准测试，挖掘潜在的安全漏洞。参赛者需要结合软件的实际应用场景，设计高效的测试用例生成策略和漏洞检测机制，为开源软件的安全性提升提供支持。该赛题将推动参赛者在软件安全、模糊测试技术和漏洞分析等领域进行深入研究和实践，具有重要的技术创新价值和实际应用意义。

赛题要求：

1. 测试环境以开源操作系统的关键组件（例如openKylin操作系统的kylin-photo-viewer 、kylin-os-manager、glibc、ukui-control-center、libukcc-dev 、kylin-connectivity等组件）作为研究对象。

2. 代码分析：能够对以开源操作系统的关键组件的代码结构进行分析，识别关键功能模块。

3. 测试用例生成：根据目标模块的输入格式和已知漏洞模式，生成针对性的测试用例。

4. 测试执行与监控：将测试用例输入到目标模块中，并实时监控模块的执行状态，检测异常行为（如崩溃、内存泄漏、非法访问等）。

5. 漏洞定位与报告：当检测到异常行为时，能够快速定位到可能的漏洞位置，并生成详细的漏洞报告。

评分标准：

功能完整性（40%）：

* 具备模块分析功能（25分）；
* 具备测试用例生成功能（25分）；
* 具备测试执行与监控功能（25分）；
* 具备漏洞定位与报告功能（25分）。

漏洞挖掘能力（40%）：

* 复现2023年4月以后国产开源操作系统中的报告的CVE漏洞，每复现1个得分10分，最高50分；
* 发现严重漏洞1个得分20分，发现重要漏洞1个10分，发现低危漏洞1个5分，最高50分。

代码规范性（10%）：

* 代码结构清晰、符合开源社区规范、可维护性高（80分）；
* 符合开源社区规范（20分）。

文档质量（10%）：

* 文档包含设计说明、部署说明、测试报告（50分）；
* 文档条例清晰、撰写规范（50分）。

赛题联系人：

赵欣 zhaoxin10@nudt.edu.cn

参考资料：

1. openKylin官方网站：https://www.openkylin.top/index-cn.html
2. 国家信息安全漏洞库：https://www.cnvd.org.cn/
3. 开源漏洞披露平台：https://www.cve.org/

参赛资源支持：无

1. 赛题题目：面向GraphRAG的操作系统级优化

赛题说明：

RAG（检索增强生成）技术通过结合检索与生成模型，利用外部知识库提升生成内容的准确性和时效性，是缓解大模型“幻觉”问题的有效手段。但RAG存在显著不足：检索质量依赖片段相关性，易受关键词匹配局限，难以捕捉深层语义关联；对多跳推理（需串联多个信息点）支持较弱，且上下文窗口受限可能导致信息筛选不全。GraphRAG作为RAG的改进方案，通过构建知识图谱重构检索架构，其优势在于通过利用图结构存储实体关系，支持复杂关系推理和多跳查询，显著提升语义理解深度；此外，通过图嵌入可以实现更精准的语义检索，减少信息碎片化问题。然而，GraphRAG的系统实现面临显著的工程挑战。在存储层面，知识图谱的动态更新需频繁调用LLM生成结构化语义描述（如实体关系三元组），导致索引构建的I/O吞吐效率下降与存储成本激增（微软LazyGraphRAG研究表明，传统方案索引成本高达数百万Token）。内存管理方面，图算法（如社区检测、PageRank权重计算）的随机内存访问模式引发严重的资源争用问题，而多模态数据的混合存储进一步加剧内存碎片化风险。计算效率方面，图遍历操作的低缓存命中率与异构硬件（CPU/GPU/TPU）适配不足导致检索延迟显著高于向量检索方法。

本赛题聚焦操作系统级优化方法，旨在增强GraphRAG在实际应用场景的效果和性能。通过系统设计优化存储架构、内存调度与计算资源协同机制，以改善知识图谱的动态更新效率与大模型推理服务质量。

赛题要求：

基于现有开源GraphRAG方案（例如”GraphRAG”和”LightRAG”），进行系统级优化方案的设计、实现与验证。对硬件不做固定要求，但要求优化前后保持相同硬件环境。推荐使用最新的开源大模型，如deepseek系列和QwQ-32B等。要求通过详细实验验证系统优化效果，并开源优化方案和验证方案，同时对优化前后的系统预测效果进行比较。  
参赛者可以针对（但不限于）以下技术挑战之一（或多个）提出创新解决方案，并构建可验证的原型系统：

1. 动态存储架构优化

例如考虑设计分层存储策略，例如结合增量索引构建与日志结构化合并树（LSM-Tree）优化写入性能。

2. 内存动态分配与压缩技术

例如考虑设计实现基于三级内存池（L1缓存高频实体、L2共享中间结果、L3分布式存储）的动态资源分配机制；采用稀疏矩阵编码与边属性压缩技术，减少内存占用，并探索GPU加速图嵌入计算在多任务并发场景下的性能瓶颈。

3. 混合检索算法与硬件协同设计

例如考虑融合向量相似性初筛与图路径精修的双阶段检索策略，结合FPGA定制化指令集优化图遍历操作的缓存命中率，缩短检索延迟。

4. 知识表示形式优化

例如考虑设计能够捕获复杂实体关系与高阶结构信息的知识表示方案，如结合超图（Hypergraph）提升复杂查询与推理任务的系统效果。

评分标准：

**一、目标达成度（35%）**

**核心目标**：评估参赛方案是否针对自选技术挑战实现有效优化，目标设定合理且逻辑闭环

| **评分项** | **评分细则** | **分值** |
| --- | --- | --- |
| **优化目标明确性** | 清晰定义优化方向（如存储/内存/检索/预测效果等）并提出量化指标（如延迟降低30%、内存占用减少20%） | 40分 |
| **技术方案适配性** | 自选优化方案与目标高度匹配（例如针对存储I/O瓶颈选择LSM-Tree而非通用缓存策略），方案创新性显著 | 30分 |
| **多目标协同能力** | 若选择多个优化点，需论证协同机制（如内存压缩与GPU加速的调度策略兼容性），避免优化冲突 | 30分 |

**二、优化效果验证（40%）**

**核心目标：量化验证自选优化目标的达成度，且不引发系统关键指标（如预测准确率）显著劣化**

| **评分项** | **评分细则** | **分值** |
| --- | --- | --- |
| **自定指标达成度** | 达到预设优化目标（如延迟降低≥30%得满分，每降低5%得6分） | 60分 |
| **副作用控制能力** | 其余关键指标波动控制在阈值内（如BLEU/ROUGE下降≤3%，多跳推理准确率≥基准值95%） | 20分 |
| **横向对比优势** | 对比同类方案，在相同硬件环境下展示优化效果优势（如延迟低于基线15%以上） | 20分 |

**三、工程实践质量（15%）**

**核心目标：验证方案的可复现性、可维护性及技术前瞻性**

| **评分项** | **评分细则** | **分值** |
| --- | --- | --- |
| **模块化架构设计** | **优化模块与系统核心解耦（如通过插件化接口实现存储引擎替换）** | **40分** |
| **测试覆盖完备性** | **提供优化模块的单元测试（覆盖率≥85%）** | **30分** |

**四、文档与创新性（10%）**

**核心目标：技术方案逻辑严谨且具备推广价值**

| **评分项** | **评分细则** | **分值** |
| --- | --- | --- |
| **技术决策溯源性** | **文档需包含：优化目标选择依据（如Profiling热点分析）、方案设计动机、方案对比（至少3种候选方案优劣分析）** | **40分** |
| **创新性声明** | **明确标注技术突破点（例如新型压缩算法、硬件指令集扩展），并提供与已有专利/论文的差异性分析** | **30分** |
| **部署可操作性** | **提供自动化部署脚本** | **30分** |

赛题联系人：

郭勇 yguo@nudt.edu.cn

参考文献：

1. https://graphrag.com/concepts/intro-to-graphrag/
2. <https://github.com/HKUDS/LightRAG>

参赛资源支持：无

1. **赛题题目：WASM运行时引擎在ARMv8 CPU上的移植与性能优化**

赛题说明：

WASM（WebAssembly）是一种为堆栈式虚拟机设计的二进制指令格式，用于在Web应用、嵌入式场景与物联网场景中实现高效、安全的代码执行。

WASM沙箱引擎是为WASM程序的执行提供运行时支持的软件，它们不仅根据WASM Spec规范来加载、验证和执行WASM程序，还需要根据WASI接口标准实现相应的接口函数，从而为WASM程序提供文件系统、随机数等其它系统资源的访问能力。WASM沙箱引擎分为浏览器引擎与独立运行时引擎两类，本题主要关注后者，即独立运行时引擎。

当前，开源社区中较为知名且正不断发展的WASM独立运行时引擎有WasmEdge、Wasmer、Wasmtime等，它们都支持解释执行（Interpret&Execute）、即时编译（JIT Compile）、超前编译（AOT Compile）等执行WASM字节码的方式，其中即时编译和超前编译可极大地提高WASM运行时引擎执行字节码的性能。但总体而言，WASM应用软件性能与原生应用软件性能间仍存在不容忽视的差距。

本题旨在探索WASM运行时引擎在ARMv8 CPU上的移植与性能优化方案。参赛者应在WasmEdge、Wasmer、Wasmtime三者中选择一个或多个运行时引擎，将其移植到ARMv8 CPU平台，使用诸如但不限于代码块分割方法、字节码JIT/AOT质量、JIT/AOT缓存与自动并行化等一个或多个优化方案。

参赛者的最终目标是将所选WASM运行时引擎移植到ARMv8 CPU平台，并在性能测试集Browser-SPEC上至少取得5%的性能提升，挑战取得15%或更大的性能提升。

赛题要求：

**技术文档要求：**参赛者所交付的软件应有配套的详细**移植方案**与**优化方案**。若有参考文献或互联网资料，应尽数列出。

**软件基础要求：**参赛者所交付的软件需从WasmEdge、Wasmer、Wasmtime三者中选择**至少1个**进行移植与优化迭代。

**运行环境要求：**参赛者所交付的软件需能运行在ARMv8 CPU平台上，且运行功能测试与性能测试。其中决赛现场将使用树莓派（Raspberry Pi）4B作为统一的评测硬件环境。

**功能正确要求：**参赛者所交付的软件应**完全通过**原始软件代码仓库内的**功能测试**（例如：基于WasmEdge的交付软件应通过WasmEdge仓库内的功能测试）。若有特别情况，应在迁移或优化方案中解释无法通过功能测试缘由。

**性能提升要求：**参赛者所交付的软件在Browser-SPEC上应**至少取得5%的性能提升**，**挑战取得15%或更大的性能提升**。

评分标准：

功能完整性（40分）：

* 通过所有功能测试样例，每有1个样例不通过，扣5分，扣完为止。

性能优化（50分）：

* 在Browser-SPEC上取得5%以下性能提升（10分）；
* 在Browser-SPEC上取得5%-15%性能提升（30分）；
* 在Browser-SPEC上取得15%以上性能提升（50分）。

文档质量（10分）：

* 文档详细清晰，包含安装方式及使用方法（5分）；
* 文档应充分阐述其创新性（5分）。

赛题联系人：

沃天宇 [woty@buaa.edu.cn](file:///G:\\WeChatFiles\\WeChat%20Files\\wxid_dzt7zfbalici21\\FileStorage\\File\\2025-03\\woty@buaa.edu.cn)

参考资料：

1. WasmEdge：<https://wasmedge.org/>
2. Wasmer：<https://wasmer.io/>
3. Wasmtime：<https://wasmtime.dev/>
4. 飞腾D3000 CPU：<https://www.phytium.com.cn/homepage/production/15/>
5. 树莓派：https://www.raspberrypi.com/
6. Abhinav Jangda, Bobby Powers, Emery D. Berger, and Arjun Guha. 2019. Not so fast: analyzing the performance of webassembly vs. native code. In Proceedings of the 2019 USENIX Conference on Usenix Annual Technical Conference (USENIX ATC '19). USENIX Association, USA, 107–120. <https://www.usenix.org/system/files/atc19-jangda.pdf>

参赛资源支持：无

1. **赛题题目：操作系统SEU故障注入与容错方案设计与实现**

赛题说明：

星载操作系统作为航天器软件系统的核心，承担着资源管理、任务调度、数据通信等关键功能。随着航天技术的快速发展，航天领域对星载操作系统的可靠性提出了更高要求。

在宇宙高能粒子撞击下，航天器内CPU中的寄存器、内存等硬件元器件容易发生单粒子翻转（SEU）、单粒子闩锁（SEL）等故障。针对星载操作系统面临的硬件故障挑战，设计并实现具有高可靠性的星载操作系统至关重要。在这种场景下，高可靠星载操作系统需具备一定的容错机制，能够在硬件故障发生时通过容错（Fault Tolerance）或失效恢复（Failure Recovery）机制，确保系统稳定运行。

在本赛题的场景下，参赛者只需考虑指令可见寄存器（如CPU通用寄存器、特权寄存器）、内存上的SEU，无需考虑如Cache与指令不可见寄存器（如CPU流水线寄存器）上的故障。参赛者可选用X86、ARM、RISC-V等指令集架构实施编程，并在QEMU模拟器上验证设计与效果。参赛者可将GDB与QEMU相连接，在GDB中操纵寄存器与内存以模拟故障注入。

参赛者需结合上述所描述的航天领域特殊要求，仅依靠软件技术，使用诸如但不限于数据摘要校验、多模冗余、状态机复制等一个或多个可靠性保障方案，设计并实现一个操作系统SEU故障注入和容错方案，并尽可能对用户程序透明。故障注入在学术界没有明确公认的测试数据集，参赛者需自行设计操作系统的工作负载与故障注入测例（如：均匀分布的故障注入地址/寄存器），从而展示设计的效果。

赛题要求：

**技术文档要求：**参赛者所交付的操作系统应有配套的详细**设计方案**与**测试方案**。若有参考文献或互联网资料，应尽数列出。

**软件基础要求：**参赛者所交付的操作系统可基于现有的操作系统，也可以自行设计操作系统。要求容错机制与失效恢复机制对用户程序尽可能透明。

**运行环境要求：**参赛者所交付的操作系统可有选择地适配X86、ARM、RISC-V等指令集架构，参赛者所交付的操作系统需能在未经修改的v9.2.2版本的QEMU模拟器上运行。

**功能正确要求：**参赛者所交付的操作系统应能运行有一定规模的工作负载，且在无故障注入的情境下正确地支撑工作负载的完整生命周期。

**故障注入测试：**参赛者需准确论述所设计故障注入模型（包括故障发生位置与时间的概率分布等）的合理性，参赛者还需详细报告并分析在该故障注入模型下所交付软件的故障恢复或故障容忍能力。

评分标准：

功能完整性（20分）：

* 无故障情况下，系统可以部署运行（20分）。

SEU故障注入测试（70分）：

* 所运行的工作负载具有较强代表性，如nginx等长期运行的服务程序（15分）；
* 所设计的故障注入模型充分考虑各种故障注入情景（10分）；
* 故障注入模型（故障注入的时间、空间分布）具有合理性（15分）；
* 合理故障注入下工作负载的可用性（30分）：
* ≥20%且<40%（5分）；
* ≥40%且<60%（18分）；
* ≥60%（30分）；

其中可用性定义为：工作负载正常运行时间/测试总时间。

文档质量（10分）：

* 文档详细清晰，包含安装方式及使用方法（5分）；
* 文档应充分阐述其创新性（5分）。

赛题联系人：

沃天宇 [woty@buaa.edu.cn](file:///G:\\WeChatFiles\\WeChat%20Files\\wxid_dzt7zfbalici21\\FileStorage\\File\\2025-03\\woty@buaa.edu.cn)

参考资料：

1. QEMU：<https://www.qemu.org/>
2. QEMU与GDB连接：<https://qemu-project.gitlab.io/qemu/system/gdb.html>

参赛资源支持：无

1. 赛题题目：基于PGO的OpenHarmony动态库（.so）优化自动化检测与验证系统

赛题说明：

OpenHarmony生态应用性能优化是提升用户体验的关键环节。其中，动态库(.so文件)作为核心计算模块，其执行效率直接影响应用响应速度、电池续航和设备发热。PGO(Profile-Guided Optimization)是一种强大的编译优化技术，通过收集程序实际运行数据来指导编译器优化，可为适合的动态库带来10%-15%的性能提升。

然而，在OpenHarmony生态中，PGO技术应用面临三大挑战：

* **收益评估困难**：并非所有动态库都能从PGO中获得显著收益。计算密集型、分支逻辑复杂的库（如物理引擎、渲染器）通常收益显著，而IO绑定型或线性执行路径的库收益有限。开发者难以预先判断哪些库值得投入PGO优化成本；
* **测试用例构建复杂**：有效的PGO优化依赖高质量的profile数据，需要测试用例覆盖关键执行路径。手动构建这些测试集耗时且往往不全面，导致次优的优化效果；
* **工程成本与收益平衡**：PGO引入额外的构建步骤（插桩编译→收集profile→优化编译），增加了构建时间和CI/CD复杂度。对于收益有限的库，这种额外成本可能不值得。

本赛题要求参赛者开发一套自动化工具链，能够：

* 分析OpenHarmony应用中的动态库特征，预测PGO优化潜在收益；
* 自动识别高收益库并生成针对性测试用例，覆盖关键执行路径；
* 量化优化效果，提供性能提升报告和工程成本评估。

参赛者需在RK3568开发板上实际验证工具链效果，通过至少3个真实应用案例，证明工具能准确识别高收益库并带来≥10%的性能提升(以IPC提升、缓存命中率提高或功耗降低等指标衡量)。

赛题要求：

核心任务

开发自动化工具链实现以下功能中的一项或多项：

* PGO收益预测

1. 分析动态库的二进制特征（如分支密度、循环复杂度、间接调用比例）

2. 构建预测模型，评估库应用PGO后的潜在性能提升

3. 对OpenHarmony应用中的动态库进行排序，识别最值得优化的目标

* 测试用例管理

1. 基于提供的标准测试集，验证至少3款OpenHarmony应用中的高收益动态库

2. 测试覆盖以下关键场景：

(1)高频函数调用路径

(2)分支密集区域

(3)内存密集操作

* 优化策略推荐

1. 基于性能分析数据，生成带量化收益预测的编译参数建议（如-O级别、内联阈值等）

2. 提供优先级排序的优化建议列表

3. 输出具体的编译命令行参数

* 效果验证

1. 在RK3568开发板上实测优化效果，主要指标：

(1)CPU指令数降低比例

(2)IPC (每周期指令数) 提升

2. 参考指标（加分项）：

(1)冷启动时延

(2)帧渲染稳定性

技术要求

* **二进制分析**：提取动态库的特征（可使用任何适合的工具）
* **动态追踪**：采集性能数据，如热点函数、分支预测、缓存命中率等
* **预测模型**：训练模型预测PGO收益，输入特征可包括：
* 二进制特征：分支密度、函数大小分布等
* 运行时特征：热点函数占比、分支预测失败率等

交付物

* **技术报告**：详细说明技术方案、算法设计、实验结果和性能提升分析
* **可运行的技术原型**：提供完整源代码和使用说明，确保评委可复现结果
* **输出内容**：
* 优化分析报告：包含库收益预测和优先级排序
* 验证结果：通过实际编译参数调整验证优化建议，高收益库性能提升≥10%

评分标准：

|  |  |  |  |
| --- | --- | --- | --- |
| **评估维度** | **权重** | **评分细则** | **验证方法** |
| 收益预测准确性 | 35% | - 预测与实际收益偏差<5%得25分 - 偏差<3%得30分 - 偏差<1%得35分 | 对比预测收益与实际测量收益 |
| 优化效果 | 30% | - 高收益库实际CPU指令数降低≥5%：20分 - 每降低1%加2分（上限30分） - 误优化率>5%扣15分 | 对比优化前后perf report数据 |
| 工程化水平 | 10% | - 实现自动化工作流得5分 - 提供可视化分析界面得5分 | 审查工具链的易用性与分析能力 |

创新加分项（20分上限）

* **多维度优化**：除PGO外，探索其他编译优化维度（如指令集优化、LTO等）（+10分）；
* **无源码分析能力**：针对仅有二进制文件的动态库实现有效分析（+10分）。

赛题联系人：

范刚 fan.gang@huawei.com

参考资料：

1. OpenHarmony性能分析工具：<https://developer.huawei.com/consumer/cn/agconnect/huawei-smartperf/>
2. 论文《ArkAnalyzer: The Static Analysis Framework for OpenHarmony》 项目地址: <https://gitee.com/openharmony-sig/arkanalyzer>
3. 论文《Identifying Compiler and Optimization Level in Binary Code From Multiple Architectures》
4. OpenHarmony LLVM-PGO工具链文档: <https://gitee.com/openharmony/third_party_llvm-project>
5. 鸿蒙OS部分开源应用项目地址：<https://gitee.com/explore/harmonyos-app?order=recommend>

参赛资源支持：

1. OpenHarmony社区：基于Windows PC，不提供额外硬件

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136109649061687296](https://www.chaspark.com/" \l "/races/competitions/1136109649061687296)

1. 赛题题目：基于方舟分析器的OpenHarmony应用代码分析优化

赛题说明：

OpenHarmony作为分布式开源操作系统，为多设备互联提供了统一的运行环境。然而，应用程序在OpenHarmony上运行时，可能会出现代码逻辑错误、异常崩溃、内存泄漏、线程竞争、安全漏洞等问题，影响应用的稳定性和安全性。

ArkAnalyzer（方舟分析器）是OpenHarmony生态中的程序分析工具底座，提供代码静态分析和运行时分析，可检测未初始化变量、空指针引用（NPE）、数组越界、死锁、资源泄漏等问题。方舟分析器目前已在 OpenHarmony 社区 启动开源开发活动，社区开发者可以参与贡献，共同优化该工具。具体信息可访问 OpenHarmony 社区的官方仓库：

https://gitee.com/openharmony-sig/arkanalyzer

本赛题要求参赛者基于方舟分析器，优化应用代码分析的性能，提升大规模代码分析的内存利用率、执行速度和问题检测能力。参赛者可选择以下两个优化方向之一，并提供优化算法、分析报告、优化代码、可执行工具。

**方向一：优化方舟分析器的代码加载与内存管理**

**目标：**

方舟分析器在分析海量代码时，可能存在内存占用过大、加载效率低 的问题，导致处理大规模应用时出现OOM（内存溢出）或分析速度变慢。

本任务要求参赛者优化方舟分析器的代码加载方式，减少分析时的内存占用，提升大规模代码处理能力。

**评分标准：**

优化后的代码加载占用内存减少（越少得分越高）。

优化后的代码加载速度提升（越快得分越高）。

**示例优化方案：**

支持分段内存加载：避免一次性将全部代码加载到内存中，改为按需加载，减少内存占用。

优化数据结构：减少冗余数据存储，提高内存利用率。

支持并行加载与分析：利用多线程/多进程技术，使代码加载和分析并发进行，提高效率。

缓存优化：使用增量分析，避免重复分析相同代码，减少不必要的计算开销。

**示例优化结果：**

参赛者优化了方舟分析器的代码加载模块，使其单次最大内存占用减少50%，代码分析速度提高50%。

通过分片加载代码，在分析1000+ OpenHarmony 应用时，最大内存占用减少 50%。

**方向二：优化方舟分析器的代码分析算法**

**目标：**

现有的方舟分析器在处理大规模代码时，可能存在分析速度慢、误报多、检测不全等问题。本任务要求参赛者优化代码分析算法，提高检测速度，减少误报，并找到更多真实问题。

**评分标准：**

优化后的代码分析速度提升（比方舟分析器当前 API 运行更快）。

优化后的检测准确率提升（减少误报/漏报，找到更多真实Bug）。

**示例优化方案：**

并行分析优化：使用多线程/分布式分析，提升海量代码的分析速度。

智能Bug过滤：基于机器学习/大模型等识别高置信度Bug，减少误报。

增强数据流分析：改进方舟分析器的数据流分析，加快分析收敛，提高检测能力。

优化静态分析规则：增强指针分析、变量依赖分析、程序切片，让分析更精准。

**示例优化结果：**

参赛者优化了方舟分析器的数据流分析算法，使其检测时间减少50%，误报率降低50%。

通过引入分布式分析框架，使方舟分析器在10000+OpenHarmony代码文件中分析速度提升5倍。

参赛者可以选择任意一个感兴趣的方法进行探索。参赛者需提供量化优化结果（Bug 检测准确率、崩溃率降低、内存优化数据），并提交分析报告、优化代码、可执行工具。

赛题要求：

1. 必须基于方舟分析器进行OpenHarmony应用分析，并提供优化方案。

2. 分析方法不得与OpenHarmony现有的静态分析工具雷同。

3. 方案需量化评估改进效果，如Bug 检测准确率提升60%以上、内存泄漏检测覆盖率达到95%、崩溃率降低60%，并提供优化前后的数据对比。

评分标准：

本次比赛基于官方提供的OpenHarmony开源应用数据集(基于参考资料[2])，所有优化方案将在相同的测试集上运行，并与方舟分析器官方API进行对比，确保公平、公正、可量化。

参赛者需提交完整的交付物，包括：Demo 演示、评估文档，实现代码及实验结果分析报告等。评估文档需包含详细的量化分析，例如 Bug检测准确率提升50%以上、内存泄漏检测覆盖率达到95%、崩溃率降低60%等关键指标，并提供优化前后的数据对比，确保改进效果可验证。最终交付物需清晰展示方案的有效性，并证明在测试集上的性能提升。如果交付物品无法验证方案的有效性，将无法参与后续的奖项评比。

|  |  |  |  |
| --- | --- | --- | --- |
| **评分维度** | **任务 1** | **任务 2** | **权重** |
| 优化效果 | - 内存占用减少X%（对比方舟API）  - 代码加载速度提升X%（对比方舟API） | - 代码分析速度提升X%（对比方舟API）  - 发现的Bug数量增加X%（对比方舟API） | 30% |
| 准确性 | - 代码加载无错误  - 数据完整性100% | - 误报 / 漏报减少 X%  - 比方舟API更精准 | 30% |
| 创新性 | - 采用新算法/新技术（如并行加载、缓存优化） | - 提出更快的代码分析方法（如智能Bug 过滤） | 20% |
| 文档 & 代码质量 | 文档和代码符合华为规范 | 文档和代码符合华为规范 | 20% |

赛题联系人：

胡晗 hu.han@huawei.com

参考资料：

1. **ArkAnalyzer 官方文档**：<https://gitee.com/openharmony-sig/arkanalyzer>
2. OpenHarmony开源组件库：<https://gitee.com/explore/harmonyos-app>
3. **OpenHarmony 代码仓库**：[https://gitee.com/openharmony](https://gitee.com/openharmony" \t "_new)
4. **华为开发文档**：<https://developer.huawei.com/>

参赛资源支持：

1. OpenHarmony社区：基于Windows PC，不提供额外硬件

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136110656905601024](https://www.chaspark.com/" \l "/races/competitions/1136110656905601024)

1. 赛题题目：模型对算力等效建模评估

赛题说明：

随着AI模型在稠密计算、混合专家（MoE）、多模态、生成式推荐等场景的爆发式增长，算力需求的多样性与硬件资源的代际差异矛盾日益凸显。当前算力基础设施普遍面临跨代际硬件兼容性差、动态调度效率低、资源利用率不均衡等问题，亟需构建统一的算力等效建模能力，实现从模型参数（如BatchSize、序列长度）到硬件规格（如内存、带宽）的精准映射。

本赛题要求基于异构算力资源池（如GPU、NPU、FPGA等），设计一种支持多负载的算力需求表征框架，并开发动态调度工具，为模型部署提供量化依据。其意义在于突破传统算力评估的局限性，推动算力资源的高效分配与智能调度，助力智算中心实现从“盲目建设”到“按需供给”的转型。

赛题要求：

* 建模能力

1. 构建稠密、MoE、类o1、多模态及生成式推荐等多样化负载建模及算力等资源需求统一表征能力。

2. 构建模型等效算力度量工具，基于并发量、序列长度、硬件算力、模型大小，评估算力、内存、带宽的诉求。

3. 构建硬件算力的数量、型号提供相应的建议。

* 度量工具开发

构建开发动态评估工具，输入模型参数（并发量、序列长度、参数量等），输出可运行上述模型达到基线吞吐量（单一并发＞10 tokens/s）的最小算力需求（包括与硬件算力型号、显存、内存需求等）。

* 模型的输入输出参考如下：

模型输入数据参考（包括但不限于）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 负载类型 | 模型参数 | 量化方案 | 并发量 | 最大tokens数 |
| 例如 | MOE | 70B | INT8 | 50 | 8096 |

模型输出数据参考（包括但不限于）

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | 硬件算力型号 | 单卡显存 | 卡数 | 内存 | 内存带宽 |
| 例如 | 910b | 32GB | 8 |  |  |

负载类型包括：稠密模型、MoE模型、类o1模型、多模态模型、生成式推荐模型等。

硬件算力型号包括且不限制：昇腾系列NPU（910b、300I Duo等）、NVIDIA系列GPU（A100、H100等），其他系列的国产GPU也可。

评分标准：

模型覆盖度（40%）

* 支持多种模型类型，模型参数，量化方案，并发量，最大tokens数等负载建模能力（30分）；
* 支持多种硬件资源，显存，卡数等资源抽象能力（30分）；
* 建模方案。提供详细的负载、硬件资源的建模方案描述，给出度量依据、建模方法、参考文献、算法依据、计算过程等（40分）。

度量算法评估（40%）

* 算力需求预测误差：不同测试用例下，度量工具输出的模型算力需求与实际运行真值偏差（100分）。

代码、文档完整性（20%）

* 代码结构清晰，模块化设计，便于维护和扩展。重要代码注释完整。且必须完全开源（50分）；
* 文档详细清晰，包含安装方式及使用方法（50分）。

赛题联系人：

肖春阳 xiaochunyang2@huawei.com

杜开田 dukaitian@huawei.com

参考资料：

1. [https://mp.weixin.qq.com/s?\_\_biz=MzkxODcxNTExMg==&mid=2247483667&idx=1&sn=d74b229af0b38c1346492bc2ace866a2&chksm=c0443692d0a4d28cd931aa4b913c55ecebcb30bd290d673dc7f23e722dc3685ef3aeeba5f833#rd](https://mp.weixin.qq.com/s?__biz=MzkxODcxNTExMg==&mid=2247483667&idx=1&sn=d74b229af0b38c1346492bc2ace866a2&chksm=c0443692d0a4d28cd931aa4b913c55ecebcb30bd290d673dc7f23e722dc3685ef3aeeba5f833" \l "rd)

参赛资源支持：

可提供部分实验用国产算力支持。

赛题交流讨论链接：

[https://www.chaspark.com/#/races/competitions/1136112499085860864](https://www.chaspark.com/" \l "/races/competitions/1136112499085860864)

1. **赛题题目：面向大模型应用的RISC-V向量和矩阵扩展算子优化**

赛题说明：

DeepSeek等模型的兴起推动了端侧大模型的发展，而RISC-V架构凭借其开放性和可扩展性，在端侧推理终端中逐步占据一席之地。优化RISC-V端侧大模型推理技术，不仅能提升推理性能，还能推动RISC-V操作系统AI生态的完善，加速RISC-V智能操作系统的发展。

本赛题旨在通过优化大模型应用中的核心算子，在 RISC-V 架构上实现高效的推理性能，充分利用RISC-V向量和矩阵扩展指令集，以及多核计算体系架构，进一步提升大模型推理的计算速度和吞吐量。

推理框架方面，参赛者可以选择llama.cpp开源框架。操作系统环境为开源操作系统，要求参赛者基于此系统进行开发，提供全方位的优化方案。

赛题要求：

* 基于RISC-V架构的向量和矩阵扩展指令，针对llama.cpp推理框架进行大模型核心算子优化。
* 利用多核并行计算技术，从算子层面加速推理性能，提高多核架构的资源利用率。
* 基于开源操作系统上进行开发和测试，确保兼容性和稳定性。
* 提供优化前后性能对比数据，展示优化效果。
* 代码需遵循开源协议，提交至大赛指定代码仓库，并附带详细的文档说明，包括实现原理、优化思路和使用指南。

评分标准：

功能完整性（40%）：

* 提供针对性的优化方案，使用RISC-V向量以及矩阵两种扩展（30分）；
* 优化算子包括GEMM、激活函数、数学运算。算子在推理框架中能够正确执行，并输出正确的结果（20分）；
* 如优化其余算子，并保证其正确性，可额外加分（20分）；
* 利用RISC-V多核特性加速算子计算，优化技术包括多核协同计算、流水线并行等（30分）。

性能优化（30%）：

* 推理时间减少超过50%（40分）；
* 计算资源CPU占用降低超过10%（30分）；
* 吞吐量提升超过20%（30分）。

代码规范性（20%）：

* 代码结构清晰，模块化设计，便于维护和扩展，且必须完全开源（60分）；
* 注释完整性，特别是复杂逻辑部分（40分）。

文档质量（10%）：

* 文档详细清晰，包含安装方式及使用方法（50分）；
* 文档标明其创新性（50分）。

赛题联系人：

李可 [li\_ke@hl-it.cn](file:///C:\\Users\\jenny\\Desktop\\操作系统大赛题目-2025\\2025.3.31赛题更新\\0411\\li_ke@hl-it.cn" \t "dlt)

参考资料：

1. [https://github.com/XUANTIE-RV/riscv-matrix-extension-spec](https://github.com/XUANTIE-RV/riscv-matrix-extension-spec" \t "dlt)
2. [https://github.com/space-mit/riscv-ime-extension-spec](https://github.com/space-mit/riscv-ime-extension-spec" \t "dlt)
3. [https://github.com/ggml-org/llama.cpp](https://github.com/ggml-org/llama.cpp" \t "dlt)

参赛资源支持：

1. 硬件支持： LicheePi4A或LicheePi3A或OrangePi RV或Tenstorrent N300
2. 软件支持：RISC-V AI软件仓库<https://gitee.com/openkylin/community/tree/master/sig/RISC-V-AI>

12. 赛题题目：基于量化技术的AI PC推理加速

赛题说明：

在AI PC大语言模型推理场景下，因大模型任务的特殊性，端侧设备资源算力有限而无法满足大模型的需求。因此，为了解决大模型任务在端侧场景下的应用，使大模型能高效使用端侧设备上各种有限的资源，本研究期望提供一种低比特量化加速技术，通过模型的低比特量化计算，缓解大模型在端侧算力不足的问题。

赛题要求：

* 基于开源操作系统研发及运行；
* 使用开源模型,例如qwen2.5\_3b/7b；
* 提供一种4比特量化方法，可以基于开源框架中的量化工具；
* 使用4比特量化后的模型进行推理；
* 利用硬件加速或专用指令集进行低比特的计算，可以使用选择专用加速卡，赛事不提供硬件环境。

评分标准：

功能完整性（40%）：

* 量化后使用开源框架推理功能正常，推理速度高于使用开源框架中标准的q4量化方法量化的模型，且采用ceval等衡量模型推理能力指标的评估集相比原始模型分数差别不高于5%（60分）；
* 使用硬件加速功能进行计算（40分）。

性能优化（30%）：

* 有效控制精度损失范围（40分）；
* 有效降低计算资源占用（30分）；
* 有效降低存储需求量（30分）。

代码规范性（20%）：

* 代码结构清晰，模块化设计，便于维护和扩展，且必须完全开源（60分）；
* 注释完整性，特别是复杂逻辑部分（40分）。

文档质量（10%）：

* 文档详细清晰，包含安装方式及使用方法（50分）；
* 文档标明其创新性（50分）。

赛题联系人：

李真能 lizhenneng@kylinos.cn

参考资料：

1. [https://arxiv.org/abs/2401.06118](https://arxiv.org/abs/2401.06118" \t "dlt)
2. [https://arxiv.org/abs/2501.03035](https://arxiv.org/abs/2501.03035" \t "dlt)

参赛资源支持：

1. 麒麟软件有限公司可提供软件开发环境