故障定位工具介绍与使用

作者：袁江

时间：2025/12/24

CANN

目录

CANN

1. 故障定位工具介绍

故障定位工具是为开发者在**执行推理、训练过程中**可能遇到的各类异常故障，提供快速定位并解决的能力，包括：**一键式日志收集、Device状态信息展示、 AI Core Error故障信息解析**等功能。

故障定位工具
cann-oam-tools

目录

2. 工具架构&功能介绍

故障定位工具由Asys作为统一入口，用户通过Asys命令进行交互，Asys通过后端工具进行数据的采集与解析，最终由Asys完成数据汇总处理，将分析报告或定位结果展示给用户

功能	功能说明
故障信息收集	用户运行产生故障，收集**日志，软硬件信息**
文件解析	用户程序在运行过程中产生的trace文件/coredump文件/stackcore文件/coretrace文件进行解析
软硬件、Device状态信息展示	查看安装包版本，**Device温度，功率**等信息
健康检查	**可用性检测**，调用驱动接口检查所有Device或指定Device的**健康状态**，有无告警
故障检测	**可靠性检测**，对AI Core、HBM、CPU进行压力测试，观察是否出现异常
业务复跑+故障信息收集	环境中产生的日志较多，直接收集冗余信息过多，通过业务复跑，日志会重定向到用户指定路径
AI Core Error故障信息解析	日志中包含“there is an aivec error exception”或“there is an aicore error exception”，通过日志解析定位AI Core Error问题的原因

目录

CANN

3. 安装部署流程

1. 增加对软件包的可执行权限

chmod +x cann-oam-tool_<版本号>_linux-<架构>.run

2. 安装软件包（安装命令支持--install-path=*<path>*等参数，--help 命令查看命令帮助）

./ cann-oam-tool_<版本号>_linux-<架构>.run --install

如果用户未指定安装路径，则软件会安装到默认路径下，默认安装路径如下。root用户:/usr/local/Ascend，非root用户:*${HOME}*/Ascend，*${HOME}*为当前用户目录。

3. 配置环境变量，当前以非root用户安装后的默认路径为例，根据set_env.sh的实际路径执行如下命令。

source ${HOME}/Ascend/cann/set_env.sh

上述环境变量配置只在当前窗口生效，用户可以按需将以上命令写入环境变量配置文件（如.bashrc文件）。

目录

4. 重点场景- AI Core Error问题定位

现象：用户在算子应用程序报错退出，终端屏幕日志错误码为EZ9999，且日志中包含“there is an aivec error”或“there is an aicore error exception”；或者plog日志中存在报错日志“Aicore kernel execute failed”。

[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.982.539 [../../../../src/runtime/feature/src/device/device_error_core_proc.cc:403]885948 ProcessStarsCoreErrorInfo:The error from device(chipId:3, dieId:0), serial number is 54, there is an exception of aivec error, core id is 2, error code = 0, dump info: pc start: 0x12c041200000, current: 0x12c041200144, vec error info: 0xe112d1513b, mte error info: 0x302aa40, ifu error info: 0x312c041200000, ccu error info: 0x328000e9, cube error info: 0, biu error info: 0, aic error mask: 0x6500020bd00028c, para base: 0x12c100000000.
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.982.582 [../../../../src/runtime/feature/src/device/device_error_core_proc.cc:423]885948 ProcessStarsCoreErrorInfo:report error module_type=5, module_name=EZ9999
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.982.588 [../../../../src/runtime/feature/src/device/device_error_core_proc.cc:423]885948 ProcessStarsCoreErrorInfo:The extend info: errcode:(0, 0, 0) errorStr: timeout or trap error. fixp_error0 info: 0x302aa40, fixp_error1 info: 0, fsmId:0, tslot:0, thread:0, ctxid:0, blk:2, sublk:0, subErrType:2.
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.982.600 [../../../../src/runtime/feature/src/device/device_error_core_proc.cc:305]885948 AddExceptionRegInfo:add exception register for coreId=3
……
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.983.325 [../../../../src/runtime/feature/src/task/task_info/davinci_kernel_task.cc:1344]885948 GetArgsInfo:[AIC_INFO] args(20 to 31) after execute:0xa5a5a5a500000000, 0xb000000008, 0x28, 0x100000001, 0xa, 0x100000001, 0xa, 0x12c0c0026000, 0x12c0c0027000, 0, 0, 0.
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.983.330 [../../../../src/runtime/feature/src/task/task_info/davinci_kernel_task.cc:1347]885948 GetArgsInfo:tilingKey = 0, print 2 Times totalLen=(32*8), argsSize=256, blockDim=8
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.983.341 [../../../../src/runtime/feature/src/task/task_info/davinci_kernel_task.cc:1388]885948 PrintErrorInfoForDavinciTask:[AIC_INFO] after execute:args print end
[ERROR] RUNTIME(885824,add_custom):2025-12-16-11:22:10.983.393 [../../../../src/runtime/feature/src/task/task_info/davinci_kernel_task.cc:1418]885948 PrintErrorInfoForDavinciTask:[DFX_INFO]Aicore kernel execute failed, device_id=0, stream_id=47, report_stream_id=47, task_id=0, flip_num=0, fault kernel_name=AddCustom_3ee04b5d550e4239498c29151be6bb5c_0, fault kernel info ext=none, program id=0, hash=1609781249149851020.

4. 重点场景- AI Core Error问题定位

定位思路：

1. 首先通过Asys执行一键收集日志，收集定位问题需要的信息，执行如下命令：

asys collect --output=

2. 输出

```
.
├── dfx
│   ├── atrace
│   │   └── trace_97203_97450_20251215213521461084
│   ├── bbox
│   │   ├── device-0
│   │   └── device_info.txt
│   ├── log
│   │   ├── device
│   │   └── host
│   └── stackcore
│       └── dev-os-0
├── hardware_info.txt
├── health_result.txt
├── software_info.txt
└── status_info.txt
```

日志及软硬件信息

文件	解释
atrace	trace 落盘信息
bbox	Device 侧的黑匣子信息
log/device	Device 侧日志
log/host	Host侧日志
stackcore	报错触发coredump时的core文件
hardware_info.txt	host侧和device侧的硬件信息
software_info.txt	安装包版本、环境变量
health_result.txt	device侧的健康信息
status_info.txt	device侧的信息，包含芯片信号，AI core利用率等

文件对应说明

4. 重点场景- AI Core Error问题定位

定位思路：

1. 通过Asys执行检查所有Device或指定Device的健康状态，执行如下命令：

asys health -d=

2. 输出：若设备正常，则继续排查软件故障；若设备异常，参考《黑匣子异常错误码信息列表》和《健康管理故障定义》排查硬件故障。

```
asys health
 +-----------------------+-----------------------------+
 | Group of 2 Device     | Overall Health: Healthy     |
 +=======================+=============================+
 | Device ID: 0          | Healthy                     |
 +-----------------------+-----------------------------+
 | Device ID: 1          | Healthy                     |
 +-----------------------+-----------------------------+
```

正常设备

```
asys health -d=0
 +-------------------+-----------------------------+
 | Device ID: 0      | Overall Health: Warning     |
 |                   | ErrorCode Num: 1            |
 +===================+=============================+
 | 0xa419321c        | lp pmbus error              |
 +-------------------+-----------------------------+
```

异常设备

4. 重点场景- AI Core Error问题定位

定位思路：

1. 在收集到故障定位信息并且排除硬件故障后，我们可以通过Asys的AI Core Error故障信息解析功能，辅助定位AI Core问题：

asys analyze -r=aicore_error --path

2. 输出：打屏日志中，提示Asys分析的结果在info.txt中。

```
2025-12-16 12:25:02 (1311057) - [INFO] Total device count: 1
2025-12-16 12:25:02 (1311057) - [INFO] Valid device_id 0
2025-12-16 12:25:02 (1311057) - [INFO] Check the validity of the input and output paths for file parsing.
2025-12-16 12:25:02 (1311057) - [INFO] ******************Collection******************
2025-12-16 12:25:02 (1311057) - [INFO] Step 1. Check key information in the log and copy the log.
2025-12-16 12:25:02 (1311057) - [INFO] Step 2. Obtain the name and path of the flushed data file from the log.
2025-12-16 12:25:02 (1311057) - [INFO] Step 3. Obtain the operator name from the log.
2025-12-16 12:25:02 (1311057) - [INFO] Step 4. Obtain the compilation file based on the operator name.
2025-12-16 12:25:02 (1311057) - [INFO] Step 5. Start to collect compile file.
2025-12-16 12:25:02 (1311057) - [INFO] Step 6. Collect Graph Engine files.
2025-12-16 12:25:02 (1311057) - [INFO] ******************Analysis******************
2025-12-16 12:25:02 (1311057) - [INFO] Step 1. Extract operator information, including registers, tiling, and operator files.
2025-12-16 12:25:02 (1311057) - [INFO] Step 2. Create a directory for storing parsing result files.The directory is /home/[redacted]/msaicerr_exception/info_2
2502/aicerror_0_20251124105454
2025-12-16 12:25:02 (1311057) - [INFO] Step 3. Extract the node information of an operator in the GE graph.
2025-12-16 12:25:02 (1311057) - [INFO] Step 4. Extract and compare the data between 'args before' and 'args after'.
2025-12-16 12:25:02 (1311057) - [INFO] Step 5. Decompile the operator file, which triggers an instruction.
2025-12-16 12:25:02 (1311057) - [INFO] Step 6. Check whether memset or atomic_clean is correctly inserted before the operator in the graph.
2025-12-16 12:25:02 (1311057) - [INFO] Step 7. Parse dump data and check whether flushing data to disk is normal.
2025-12-16 12:25:03 (1311057) - [INFO] Parse dump file finished, result path is: /home/[redacted]msaicerr_exception/info_20251216122502/collection/dump
2025-12-16 12:25:03 (1311057) - [INFO] Step 8. Check whether the pointer tensor exists.
2025-12-16 12:25:03 (1311057) - [INFO] Step 9. Verify a single operator.
2025-12-16 12:26:27 (1311057) - [INFO] Successfully reproduced the AI Core exception by running the single-operator test case.
2025-12-16 12:26:27 (1311057) - [INFO] Step 10. Verify the environment using the sample operator.
2025-12-16 12:27:06 (1311057) - [INFO] Step 11. Write the parsing result. The result file is saved in the /home/[redacted]/msaicerr_exception/info_20251216122
tory.
2025-12-16 12:27:06 (1311057) - [INFO] Analysis info is saved in /home/[redacted]/msaicerr_exception/info_20251216122502/aicerror_0_20251124105454/info.txt
2025-12-16 12:27:06 (1311057) - [INFO] The summary info is saved in /home/[redacted]msaicerr_exception/info_20251216122502/README.txt
2025-12-16 12:27:07 (1311057) - [INFO] Analysis finished, please check /home/[redacted]aicerr_exception/info_20251216122502, you can view README.txt first.
```

4. 重点场景- AI Core Error问题定位

定位思路： 开始 → 收集故障信息 → 排查硬件故障 → 排查软件故障 → 结束

Info.txt 文件中包含5部分内容，通过第2部分内容中发现，问题原因是数据搬运时访问越界

1. 报错芯片和算子名称

```
*****1. Basic information*************
error time        : 2025-11-24-10:54:5
device id         : chipId:3, dieId:0
core id           : 2
task id           : 0
stream id         : 3
node name         : None
kernel name       : te_gatherv2_9921e3
flip num          :
kernel file       : /home/msaicerr_exc
json file         : /home/msaicerr_exc
cce file          : /home//msaicerr_ex
rts_block_dim     : 40
driver_aicore_num : 36
```

2. AI Core 错误码及报错解读

```
*****2. AI Core DFX Register*****
AIC_ERROR        : (0x800000, 0, 0)
MTE_ERR_INFO : 0x302aa40
mte_err_type bit[26:24]=011
meaning:read decode error, 越界, 读请求访问的目的地址不在

mte_err_addr bit[22:8]=000001010101010
meaning:MTE Error Address [19:5]  approximate:0x5540
```

3.报错的操作指令所在位置

```
***3. Operator Error Line Number***
start pc   : 0x124000000000
current pc: 0x1240000001a0
Error occurred most likely at line: 100
```

4. 算子执行前后，输出地址的内容

```
***4. Operator Input/Output Memory*****
args before execution: []
args after  execution: [[20616935636992],[20616943318016],[2
```

5. Dump文件解析：算子输入输出数据

```
***********5. Operator Dump File Parsing***
Original file: /home/msaicerr_exception/inf
after convert:
shape: (1, 1, 120000, 16) size: 3840000 dty
/homemsaicerr_exception/info_20251216122502
    If dtype is float32, summary is: Max: 2
    If dtype is float16, summary is: Max: 5
    If dtype is bfloat16, summary is: Max:
    If dtype is int32, summary is: Max: 115
    If dtype is int64, summary is: Max: 497
shape: (1, 120000) size: 480000 dtype: unkn
/home/msaicerr_exception/info_2025121612250
    If dtype is float32, summary is: Max: 1
    If dtype is float16, summary is: Max: 3
    If dtype is bfloat16, summary is: Max:
    If dtype is int32, summary is: Max: 100
```

6. 执行单个算子结果，测试设备是否正常

```
****6. Execution Result of the Single-Operator Test Case**
Single-operator test case success to be executed.
```

目录

CANN

5. 后续规划

1、关键竞争力

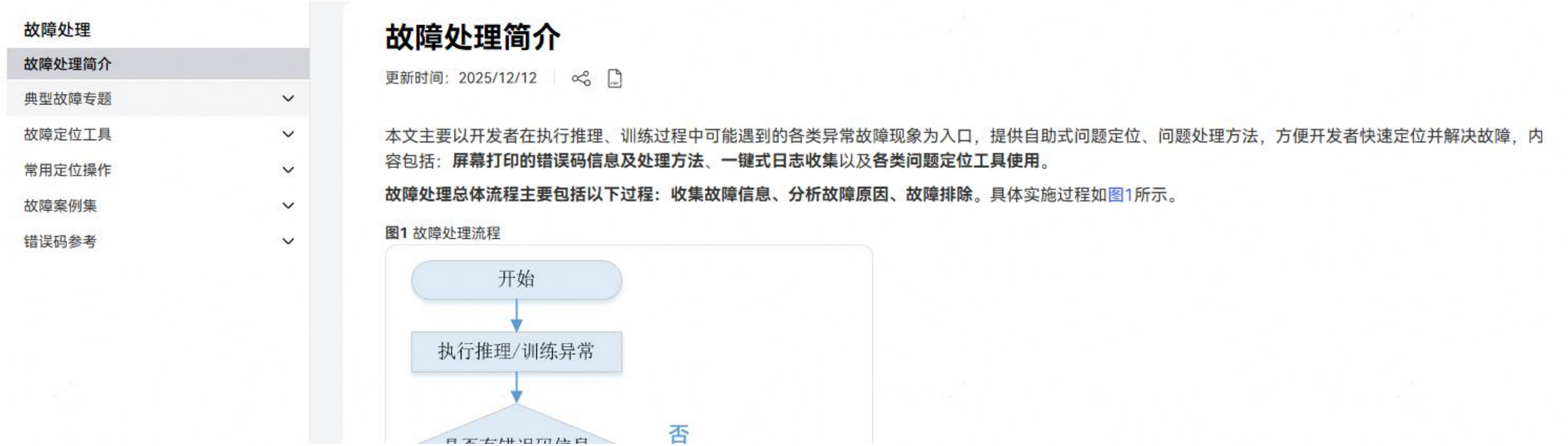
竞争力	关键指标	25年基线	26年目标
生态开放接口	统一数据标准 支持三方生态工具	• 不统一 • 不支持	• 支持业界可观测统一标准 • 支持Prometheus生态
全链路可视	功能完备性（对标业界Leader） 易用性	• 基础拓扑可视能力 • 易用性评分XX	• 支持全链路拓扑（H2D、D2D多场景） • 易用性评分 XX

2、关键能力里程碑

目录

6. 开源介绍

- 开源代码仓：https://gitcode.com/cann/oam-tools
- 开源社区文档：https://www.hiascend.com/document/detail/zh/CANNCommunityEdition/850alpha001/maintenref/troubleshooting/troubleshooting_0001.html



目录

7. 参与活动

AMCT仓，欢迎提交issue/PR

欢迎通过SIG联系我们

Oam-tools仓PR提交

Oam-tools仓Issue提交

Thank you.

社区愿景：打造开放易用、技术领先的AI算力新生态

社区使命：使能开发者基于CANN社区自主研究创新，构筑根深叶茂、跨产业协同共享共赢的CANN生态

Vision: Building an Open, Easy-to-Use, and Technology-leading AI Computing Ecosystem

Mission: Enable developers to independently research and innovate based on the CANN community and build a win-win CANN ecosystem with deep roots and cross-industry collaboration and sharing.

上CANN社区获取干货

关注CANN公众号获取资讯

https://gitcode.com/cann

CANN