天数智芯

测试工具使用指南

版本：V4.0.0-MR

日期：2024.4.3

适用产品：智铠 50| 智铠 100

1 声明

1.1 版权声明

1.2 免责声明

天数智芯可以随时对本文档或本文档中描述的产品进行改进和/或更改。本文档包括与天数智芯产品有关的信息，作为说明典型应用的一种方式, 因此，不一定提供足以进行生产设计的完整信息。对于本文档中内容的准确性或完整性，天数智芯不做任何陈述或保证。

1.3 联系方式

地址：上海闵行区陈行公路 2168 号 3 幢

电话：021-68886607

网址：www.iluvatar.com

Contents

2 测试工具使用指南

2.1 修订记录

- COREX01-MR400-UG06-00：2024/4/3

  文档本次发布内容与 V3.2.1-MR 文档相比 (COREX01-MR321-UG06-00)，有以下更新：

  - 更新测试工具默认安装路径为 `/usr/local/corex-{v.r.m}/extras/test_demo/`
  - 更新 bandwidthTest 带宽测试工具 的描述，并提供更详细的带宽测试方法
  - 更新矩阵算力测试方法，支持 BF16 数据类型
  - 新增通讯库性能测试工具，用于验证 all gather、all reduce、all to all、broadcast、reduce scatter 等多卡通信 OP 的功能和性能
  - 新增GDS 基准测试工具，用于验证 GDS 功能是否可用

2.2 工具概览

天数智芯测试工具将随天数智算软件栈同时安装。您可以使用以下测试工具：

工具	描述
带宽测试工具	bandwidthTest 工具提供 GPU memcpy 带宽以及 PCIe 中的 memcpy 带宽数据，可以测试 host 和 device，以及 device 间的传输带宽。
矩阵算力测试工具	gemm_perf 工具执行用户指定数据类型、循环次数和矩阵大小的 GEMM 运算。
P2P 性能测试工具	p2pBandwidthTest 工具提供基于 DMA 的 P2P 传输性能测试。
通讯库性能测试工具	ixCCL-tests 工具用于验证 all gather、all reduce、all to all、broadcast、reduce scatter 等多卡通信 OP 的功能和性能。
GDS 基准测试工具	ixGDS 工具用于验证 GDS 功能是否可用。GDS 通过 DMA 使 GPU 与磁盘之间可以直接传输数据，避免了数据通过 CPU，可以提升数据传输带宽，降低 CPU 负载。

本文所有示例命令以测试工具默认安装路径 `/usr/local/corex-{v.r.m}/extras/test_demo/` 为例。

2.3 带宽测试工具

bandwidthTest 测试工具可提供 GPU memcpy 带宽以及 PCIe 中的 memcpy 带宽数据，它能测试 device 到 device 之间的传输带宽，也可测试分页内存和页锁定内存的 host 到 device 之间的传输带宽，以及分页内存和页锁定内存的 device 到 host 之间的传输带宽。

带宽测试主要涵盖三个测试场景：

- host to device，即 H2D，从系统内存搬运指定大小（size）的数据到天数智芯加速卡 HBM 内存，循环 100 次。记录总时间，用 size * 100 / 总时间，计算带宽。
- device to host，即 D2H，从天数智芯加速卡 HBM 内存搬运指定大小的数据到系统内存，循环 100 次。计算方式相同。
- device to device，即 D2D，从天数智芯加速卡 HBM 内存地址 A 搬运指定大小的数据到天数智芯加速卡 HBM 内存地址 B。计算方式相同。

各场景可对应如下内存模式：

场景	支持分页内存模式（pageable）	支持页锁定内存模式（pinned）
device to host	支持	支持
host to device	支持	支持
device to device		支持

您可以选择测试模式，包括 quick，range 和 shmoo，不同模式测试的内存范围不同。详情参考下方参数描述。

针对不同内存模式，测试工具分别调用如下函数：

- 分页内存模式：`cudaMemcpy`
- 页锁定内存模式：`cudaMemcpyAsync`

2.3.1 带宽测试方法

命令格式：

```
$ bandwidthTest < option >...
```

以下命令分别测量 host to device，device to host 和 device to device 的带宽，size 范围为 10485760 字节 (start) 至 104857600 字节 (end)，以 10485760 字节为 size 增量 (increment)。

```
$ cd /usr/local/corex-{v.r.m}/extras/test_demo/
$ ./bandwidthTest --memory=pinned --mode=range \
--start=10485760 --end=104857600 --increment=10485760 --htod
$ ./bandwidthTest --memory=pinned --mode=range \
--start=10485760 --end=104857600 --increment=10485760 --dtoh
```

```
$ ./bandwidthTest --memory=pinned --mode=range \
--start=10485760 --end=104857600 --increment=10485760 --dtod
```

start 和 end 不一样，表示 size 逐渐递增，每次 size 是从 start 开始，每次以 increment 递增取 size，每个 size 都计算一次带宽。所以第一次搬运的 size 是 10485760 (start)，第二次是 20971520 (start+increment)，第三次是 31457280 (start+increment*2)，以此类推。您可以自行设定搬运的数据量 (size)，测试时长将随数据量的增加而增加。

2.3.2 带宽测试工具功能列表

bandwidthTest 工具的可选项如下：

命令参数	命令描述
--help	Display this help menu
--csv	Print results as a CSV
--device= < deviceno >	Specify the device to be used **all**: Compute cumulative bandwidth on all the devices **0,1,2,...,n**: Specify any particular device to be used
--memory= < memmode >	Specify which memory mode to use: **pageable**: Use pageable memory **pinned**: Use non-pageable system memory
--mode= < mode >	Specify the mode to use: **quick**: Perform a quick measurement, the default size is 32M **range**: Measure a user-specified range of values **shmoo**: Perform an intense shmoo of a large range of values
--htod	Measure host to device transfers
--dtoh	Measure device to host transfers
--dtod	Measure device to device transfers
--wc	Allocate pinned memory as write-combined
--cputiming	Force CPU-based timing always

命令参数	命令描述
--start= < size > --end= < size > --increment= < size >	(For range mode only) The starting transfer size in bytes (For range mode only) The ending transfer size in bytes (For range mode only) Increment size in bytes

2.4 矩阵算力测试工具

天数智算软件栈提供矩阵算力测试工具 gemm_perf：

算力测试项	工具	说明
矩阵算力	gemm_perf	用于统计计算的 GPU 总用时以及 TFlops/s。使用 gemm_perf 矩阵算力测试工具，可以指定需要测试矩阵乘的 MNK 大小，转置属性，循环次数，以及矩阵元素的数据类型。

2.4.1 矩阵算力测试方法

1. 执行测试程序，可显示对应参数的详细说明，具体介绍如下：

```
$ cd /usr/local/corex-{v.r.m}/extras/test_demo/
$ ./gemm_perf --help

Iluvatar GEMM performance benchmark.
Usage: ./gemm_perf [OPTIONS]

Options:
-h,--help Print this help message and exit
--m UINT Specify GEMM M size.
--n UINT Specify GEMM N size.
--k UINT Specify GEMM K size.
--ta BOOLEAN Specify whether transpose matrix A.
--tb BOOLEAN Specify whether transpose matrix B.
--d UINT Specify data type of input matricies. 0 for FP32, 1 for FP16, 2 for INT8, 3 for
 ↪  BF16.
--l UINT Specify number of loops calculating GEMM.
--nd Use normal distribution to generate matrices, default using uniform distribuition
--limit FLOAT Use limit value to set the fixed range, by default it is 0
--i INT ... Specify devices with index, 0,1,2...
```

例如，执行 100 次 FP16 矩阵相乘（不指定 m，n，k，ta，tb 等参数时默认会选择较优配置）：

```
$ ./gemm_perf --i 0 --d 1 --l 100
```

Important

- **--i** 是必要参数，表示指定卡。0 表示第一张卡，1 表示第二张卡，以此类推。
- 如需指定某张卡，例如指定第三张卡，则使用 **--i 2** 表示。
- 如需指定多张卡，例如指定前三张卡，则使用 **--i 0,1,2** 表示。

2. 算力结果输出如下方所示：

```
Benchmarking GEMM: M = 1536 N = 2048 K = 20480 Transpose A: 0 Transpose B: 1 LOOPS: 100
↪  Data type: FP16
Specified device: 0
Test results:
avg elapse: 263.737947 ms, TFLOPs: 48.854941
↪
```

2.5 P2P 性能测试工具

p2pBandwidthTest 工具提供基于 DMA 的天数智芯加速卡之间 P2P 传输性能测试。

2.5.1 P2P 性能测试方法

执行以下命令：

```
$ cd /usr/local/corex-{v.r.m}/extras/test_demo/
$ ./p2pBandwidthTest
```

输出示例如下：

```
[P2P (Peer-to-Peer) GPU Bandwidth Latency Test]
Device: 0, Iluvatar MR-V100, pciBusID: 39, pciDeviceID: 0, pciDomainID:0
Device: 1, Iluvatar MR-V100, pciBusID: 3d, pciDeviceID: 0, pciDomainID:0
Device=0 CAN Access Peer Device=1
Device=1 CAN Access Peer Device=0
P2P Connectivity Matrix
D\D 0 1
0 1 1
1 1 1
Unidirectional P2P=Disabled Bandwidth Matrix (GB/s)
D\D 0 1
0 N/A 1.02
1 1.55 N/A
```

```
Unidirectional P2P=Enabled Bandwidth Matrix (GB/s)
D\D 0 1
0 N/A 12.32
1 12.45 N/A
```

结果矩阵分别为:

- 多卡连接性
- 多卡间不打开 P2P 的带宽
- 多卡间打开 P2P 的单向读写带宽

2.6 通讯库性能测试工具

ixCCL-tests 工具作为通讯库性能测试工具，用于验证 all gather、all reduce、all to all、broadcast、reduce scatter 等多卡通信 OP 的功能和性能。

ixCCL-tests 是基于 NCCL-tests 在天数智算软件栈环境中重新编译获得，具体的使用方法也可以参考 NCCL-tests。

2.6.1 开始之前

使用 ixCCL-tests 工具前，您需要确保如下操作已完成：

- 如果需要使用 IB/ROCE 网卡，您需要确保网卡驱动已安装、RDMA 互通测试正常。使用 Docker 容器环境进行测试时，需要在创建 Docker 容器时加上 **--network=host** 参数。
- 您需要确保参与测试的机器都已正确安装天数智算软件栈，并可以运行基本的 GPU 程序。如果需要使用 GDR 作为机间的通信方式，则需要确保已加载 itr_peer_mem_drv 模块，您可以通过 **lsmod | grep itr_peer_mem_drv** 命令进行确认。
- 进行多机通信测试，需要配置 SSH 免密登录。在执行测试前，需要先互相 SSH 登录一次，用于确保 known-hosts 之间已保存了各自的 list。如果环境在 Docker 容器内，则需要确认 SSH 的监听端口，并使用 mpirun 的 MCA 参数 **-mca plm_rsh_args** 来指定，详情请参考MCA 参数说明。

  注：配置多机的 Docker 容器，如果是非 root 用户想要使用 RDMA，则需要修改 memlock unlimited，即在启动容器时加上 **--ulimit memlock=-1** 参数；如果是主机环境，则需要使用 **ulimit -l unlimited** 命令。
- 请您先设置环境变量 **export NCCL_USE_DIRECT=1** (NCCL_USE_DIRECT 的默认值为 0)。

2.6.2 参数说明

以下是使用 ixCCL-tests 工具进行测试时涉及的参数说明。

2.6.2.1 mpirun 重要参数说明

- **--allow-run-as-root**：允许 mpirun 在 root 用户执行时运行 (mpirun 默认在以 root 用户身份启动时中止)
- **--prefix <dir>**：指定 MPI 的安装路径
- **-np <number of processes>**：指定并行进程的数量
- **-x <variable>**：将环境变量传递给远程主机
- **-H, -host, --host <host1,host2,...,hostN>**:指定要在其上启动进程的主机名,形如:0.113.2.20:4,10.113.2.21:4，":" 后紧跟着的"4" 每个节点启动的进程的上限
- **-mca <key> <value>**：指定 MCA 参数

2.6.2.2 NCCL 环境变量说明

- `NCCL_NET`：指定多机通讯的方式可以为 Socket 或者 IB，指定为 IB 的同时，加载了 itr_peer_mem_drv 驱动就会通过 GDR 进行通信；没有加载该驱动则会通过 host IB 的方式进行通信
- `NCCL_SOCKET_IFNAME`：指定 Socket 通信使用的网卡
- `NCCL_IB_DISABLE`：为 1 则表示禁用 IB 网卡

2.6.2.3 MCA 参数说明

- **-mca plm_rsh_args**：指定启动 MPI 进程是传递给 SSH 或者 RSH 的参数，该参数是个字符串，形如：**-mca plm_rsh_args '-p 55555'**，其中的"55555" 是 ssh_config 配置的 Port
- **-mca btl ^openib**：MPI 进程将使用除 OpenIB 之外的所有可用通信模块进行通信
- **-mca btl_tcp_if_include bond0** (网卡)：指定在启动 MPI 进程时要使用的通信模块。"btl" 表示 Byte Transfer Layer，是 OpenMPI 中的一个组件，用于在 MPI 进程之间传输数据；"tcp" 表示使用 TCP/IP 协议进行通信；"if_include" 表示只使用指定的网络接口进行通信

2.6.2.4 CUDA 环境变量说明

- `CUDA_DEVICES_ORDER=PCI_BUS_ID`：要求运行时设备查询按照 PCI_BUS_ID 的顺序索引，从而使得设备 ID= 物理 ID，保证 CUDA 应用按期望使用指定设备

2.6.2.5 ixCCL-tests 可执行程序参数说明

- **-b,--minbytes <min size in bytes>**：起始值，默认值为 32M
- **-e,--maxbytes <max size in bytes>**：最大值，默认值为 32M
- **-f,--stepfactor <increment factor>**：增量因子，大小之间的乘法因子，默认禁用
- **-g,--ngpus <GPUs per thread>**：每个线程的 GPU 数量，默认值为 1

2.6.3 通讯库性能测试方法

场景一：单机双卡

执行以下命令：

```
$ cd /usr/local/corex-{v.r.m}/extras/test_demo/
$ mpirun --allow-run-as-root --report-bindings -tag-output --prefix /usr/local -np 2 --bind-to
↪  none --map-by node -mca btl ^openib ./all_reduce_perf -b 8 -e 1G -f 2 -g 1
```

场景二：单机 8 卡

执行以下命令：

```
$ cd /usr/local/corex-{v.r.m}/extras/test_demo/
$ mpirun --allow-run-as-root --report-bindings -tag-output --prefix /usr/local -np 8 --bind-to
↪  none --map-by node -mca btl ^openib ./all_reduce_perf -b 8 -e 1G -f 2 -g 1
```

场景三：双机 8 卡

执行以下命令：

```
$ cd /usr/local/corex-{v.r.m}/extras/test_demo/
$ mpirun --allow-run-as-root --report-bindings -tag-output --prefix /usr/local -np 8 -H
↪  10.113.2.20:4,10.113.2.21:4 --bind-to none --map-by node -x NCCL_NET=IB -x
↪  NCCL_SOCKET_IFNAME=bond0 -x NCCL_IB_DISABLE=0 -x NCCL_ALGO=Ring -x
↪  CUDA_DEVICES_ORDER=PCI_BUS_ID -x LD_LIBRARY_PATH -mca plm_rsh_args '-p 55555' -mca pml ob1 -
↪  mca btl ^openib -mca btl_tcp_if_include bond0 ./all_reduce_perf -b 8 -e 1G -f 2 -g 1
```

2.7 GDS 基准测试工具

GDS (GPUDirect Storage) 通过 DMA 使 GPU 与磁盘之间可以直接传输数据，这样可以避免数据通过 CPU，从而提升数据传输带宽并降低 CPU 的负载。天数智算软件栈提供验证 GDS 功能是否可用的工具 ixGDS。该工具将随天数智算软件栈同时安装在默认目录 /usr/local/corex-{v.r.m}/extras/gds/ 下。

2.7.1 开始之前

使用 ixGDS 工具前，您需要做以下准备工作。

2.7.1.1 NVMe 磁盘与驱动

- NVMe 磁盘

```
# 检查环境中有无 NVMe 磁盘
$ lsblk |grep nvme
nvme0n1              259:0    0 465.8G  0 disk
├─nvme0n1p1          259:1    0   512M  0 part
└─nvme0n1p2          259:2    0 465.3G  0 part /mnt/nvmetest
```

```
# 如果磁盘没有挂载，则需要进行挂载操作
$ mount -o data=ordered /dev/nvme0n1p2 /mnt/nvmetest
```

- NVMe 驱动

```
# 检查环境中有无符合要求的 NVMe 驱动
$ lsmod |grep nvme
grep 'nvme_v1_register_nvfs_dma_ops' /proc/kallsyms # Linux 社区中的 NVMe 驱动没有提供
↪  nvme_v1_register_nvfs_dma_ops 函数，因此可以用这个函数来检查驱动是否满足要求
```

如果 NVMe 驱动不符合要求，可以使用安装脚本进行安装。安装脚本使用了 Linux InfiniBand Drivers (nvidia.com) v5.8 版本。如果您的环境是 Ubuntu 18.04 且内核为 4.15.0，则可以直接使用官方版本的安装包进行安装。

Note

如果当前系统安装在 NVMe 磁盘上，NVMe 驱动则需要安装到 initramfs 中。参考如下操作将 NVMe 驱动安装到 initramfs 中：

(在执行下述命令前，请您先备份原有的 initramfs，避免产生因驱动不可用导致的问题。)

- Ubuntu/Debian：

```
$ echo 'nvme' >> /etc/initramfs-tools/modules
$ update-initramfs -u -k {uname -r}

# 检查 NVMe 驱动是否加入到 initramfs 中
$ lsinitramfs /boot/initrd,img-$(uname r) | grep nvme
```

- CentOS：

```
$ dracut  --add-driver 'nvme' --force

# 检查 NVMe 驱动是否加入到 initramfs 中
$ 1sinitrd /boot/initramfs-$(uname r).img | grep nvme
```

2.7.1.2 itrfs 驱动和 libcufile.so 文件

- itrfs 驱动
  - itrfs 驱动将随天数智算软件栈 (在安装驱动时) 同时安装。当 NVMe 驱动不符合要求时，安装软件栈时将不会安装 itrfs 驱动。
  - 您可以通过 **lsmod | grep itrfs** 命令查询 itrfs 驱动是否存在。
  - 在安装 itrfs 驱动时，`/etc/modules-load.d/iluvatar.conf` 中会配置自动加载 itrfs；iluvatar service 示例会出现 `itr_fs: iluvatarcorex-init@iluvatar[,itr_peer_mem_drv],itrfs` 提示。
- `libcufile.so` 文件

  软件栈安装目录中存在 `COREX_DIR/lib64/libcufile.so.1` 文件及其软链接 `libcufile.so`。

2.7.2 ixGDS 工具使用方法

ixGDS 工具提供 GDSIO 和 GDSCP 两个工具。

2.7.2.1 使用 GDSIO 工具

GDSIO 工具可以根据不同的输入生成 IO 负载。

您可以通过定义配置文件来定义需要产生的 IO 负载，详细参数文档，请参考 1. fio - Flexible I/O tester rev. 3.36 --- fio 3.36 documentation。

以下是 `libcufile-cufile.gdsio` 文件的内容参考：

```
[global]
ioengine=libcufile
directory=/mnt
gpu_dev_ids=0
cuda_io=cufile
# 'direct' must be 1 when using cuda_io=cufile
direct=1
# Performance is negatively affected if 'bs' is not a multiple of 4k.
# Refer to GDS cuFile documentation.
bs=4K
size=4g
io_size=4g
numjobs=1

# cudaMalloc fails if too many processes attach to the GPU, use threads.
thread

[read]
rw=read

[write]
rw=write
stonewall
```

此外，您也可以通过命令行的方式定义负载。上述示例使用命令行定义负载，即为：

```
$ gdsio --ioengine=libcufile --directory=/mnt --direct=1 --bs=4K --size=4g --io_size=4g --
↪  numjobs=1 --thread --gpu_dev_ids=0 --cuda_io=cufile --name=read --rw=read --name=write --
↪  rw=write --stonewall
```

您可以通过 **./gdsio --cmdhelp** 命令查看完整的 option 信息，在 cuFile 主要为：

- rw，执行读/写
- bs，每次执行的 IO 大小，需要 4K 对齐 (与 PAGESIZE 对齐)

- size，一个 job 总大小
- io_size job，实际执行的总大小 (io_size 应当不大于 size)

2.7.2.2 使用 GDSCP 工具

GDSCP 工具使用 cuFile API 实现了拷贝文件的功能，可以检查当前环境是否支持 GDS，输入文件也需要 4K 对齐。

下面是一个使用 GDSCP 工具的示例：

```
$ ./gdscp --help
./gdscp <readfilepath> <writefilepath> <gpuid>

$ ./gdscp /mnt/nvmetest/fio_test/cp_source /mnt/nvmetest/fio_test/cp_dest_0223 0
gpu md5sum:50991d699f48bc6aba25fae3e05d3329
input md5sum:50991d699f48bc6aba25fae3e05d3329
output md5sum:50991d699f48bc6aba25fae3e05d3329
MD5 matched.
```

3 商标声明

- 天数智芯、天数智芯 logo、Iluvatar CoreX 等商标、标识、组合商标为上海天数智芯半导体有限公司之注册商标或商标，受法律保护。
- 除了天数智芯的注册商标外，本内容中使用的其他产品名称及标志仅用于识别目的，该等名称及标志可能是归属于其各自公司的商标。我们否认对该等名称及标志的所有权利。
- CentOS 标识为 Red Hat 公司的商标。
- Docker 为 Docker 公司在美国和其他国家的商标或注册商标。
- Linux 为 Linus Torvalds 在美国和其它国家的注册商标。
- NVIDIA 和 CUDA 为 NVIDIA 公司在美国和/或其它国家的商标和/或注册商标。
- PyTorch 为 Facebook 公司的商标。
- TensorFlow 为 Google 公司的商标。
- Ubuntu 为 Canonical 公司的注册商标。