
KubeCon
CloudNativeCon
North America 2024

KubeCon | CloudNativeCon

North America 2024

Building Resilience for Large-Scale AI Training: GPU Management, Failure Detection, and Beyond

Ganeshkumar Ashokavardhanan & Ace Eldeib

KubeCon | CloudNativeCon
North America 2024

Azure Kubernetes Service

Deploy and scale containers on managed Kubernetes

Ganeshkumar Ashokavardhanan

Software Engineer

Microsoft Azure

KubeCon | CloudNativeCon

North America 2024

cohere

We build LLMs!

Ace Eldeib

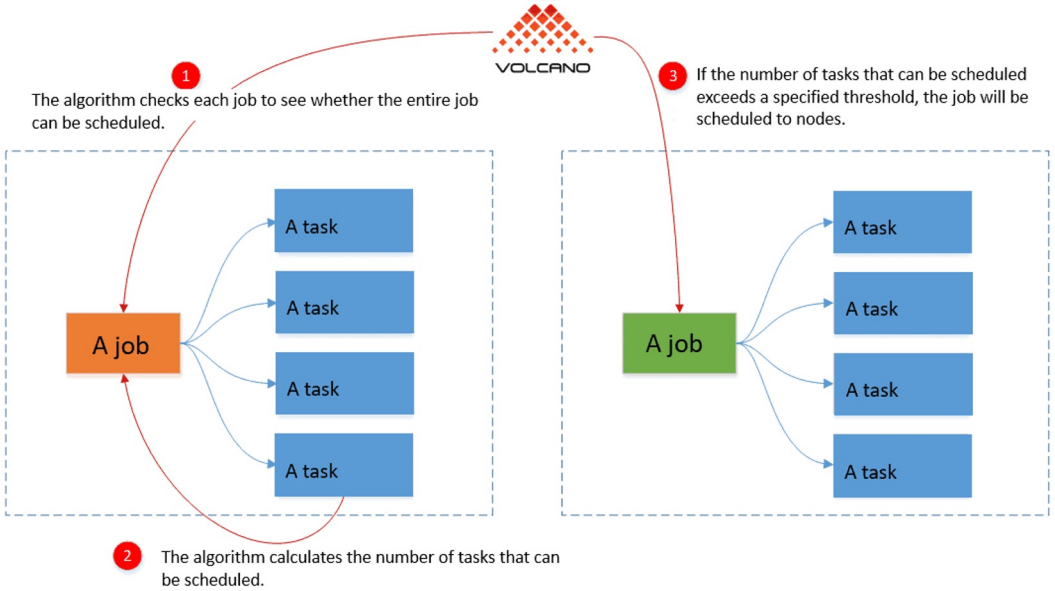
Infrastructure Engineer

Agenda

- Kubernetes + AI/ML background
- Job communication
- What goes wrong?
- Managing failure (demo!)
- Node Problem Detector for GPUs
- Remedy controllers for mitigating actions
- Demo - detecting and remediating failure
- Advanced developments and ecosystem gaps

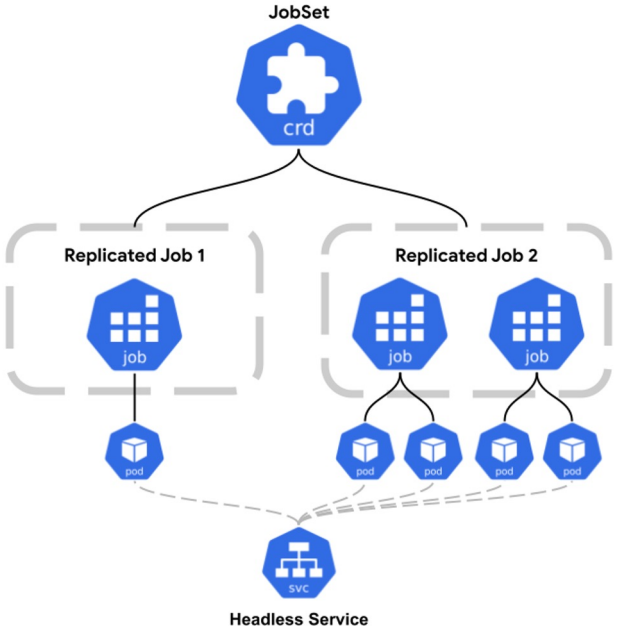
Background: K8s + ML/HPC

- AI/ML: Bigger models, more data, more compute
- Jobs running across multiple pods, multiple nodes
- Scheduled as a unit - Volcano, Kueue



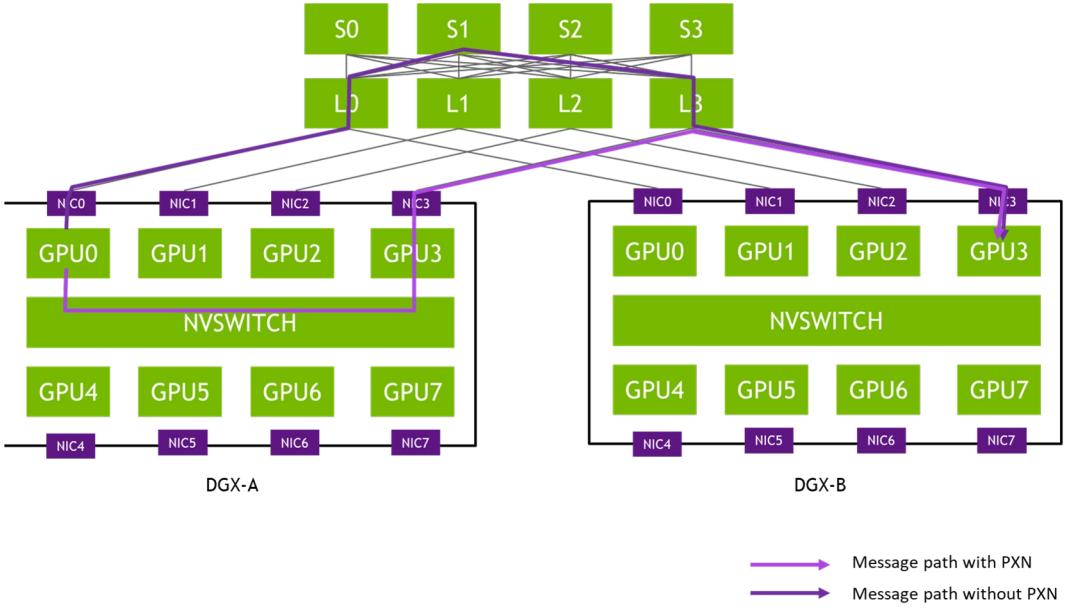
Background: K8s + ML/HPC

- Framework-specific abstractions
  - MPIJob
  - PytorchJob
- More recently: JobSet, LeaderWorkerSet
  - Startup ordering
  - Failure/retry policies
  - Predictable pod DNS



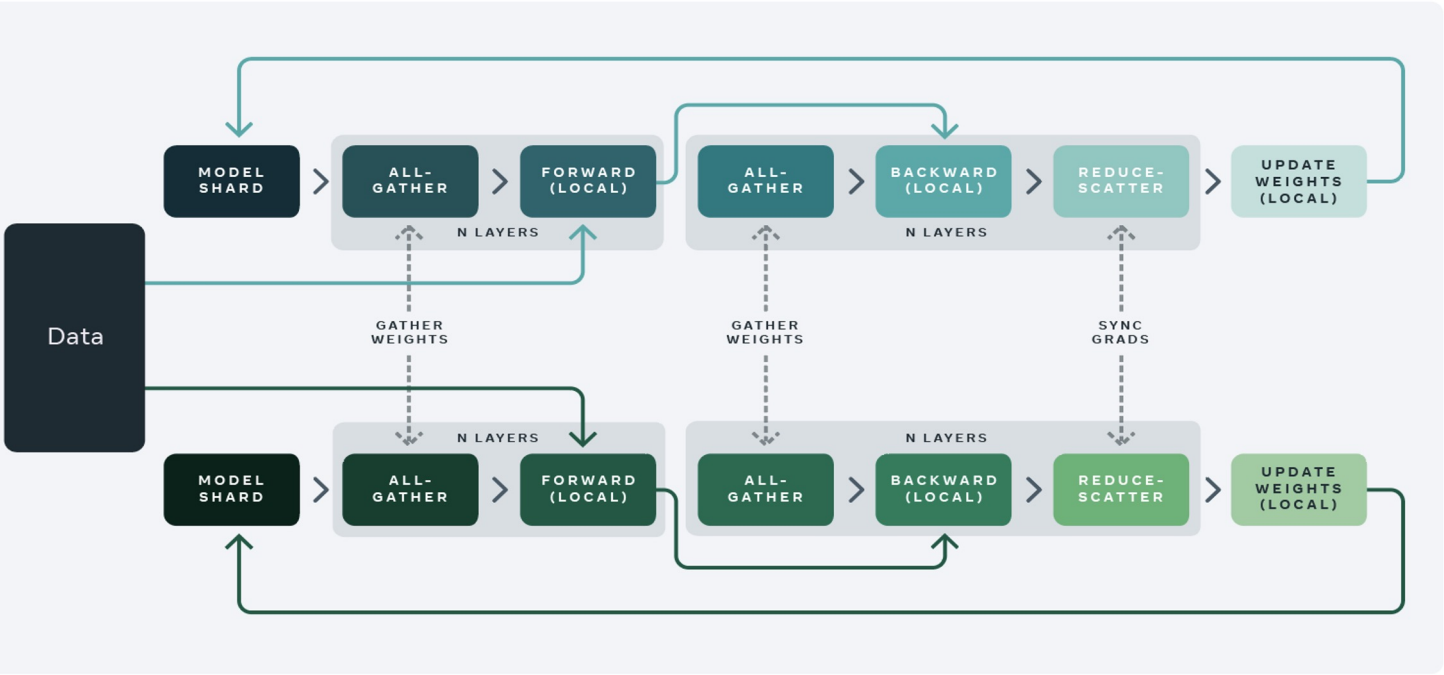
Background: K8s + ML/HPC

- LLM training: shard/replicate and parallelize
- Need to sync gradients/weights across workers between steps
- Typically use NCCL (Nvidia Collective Communications Library)



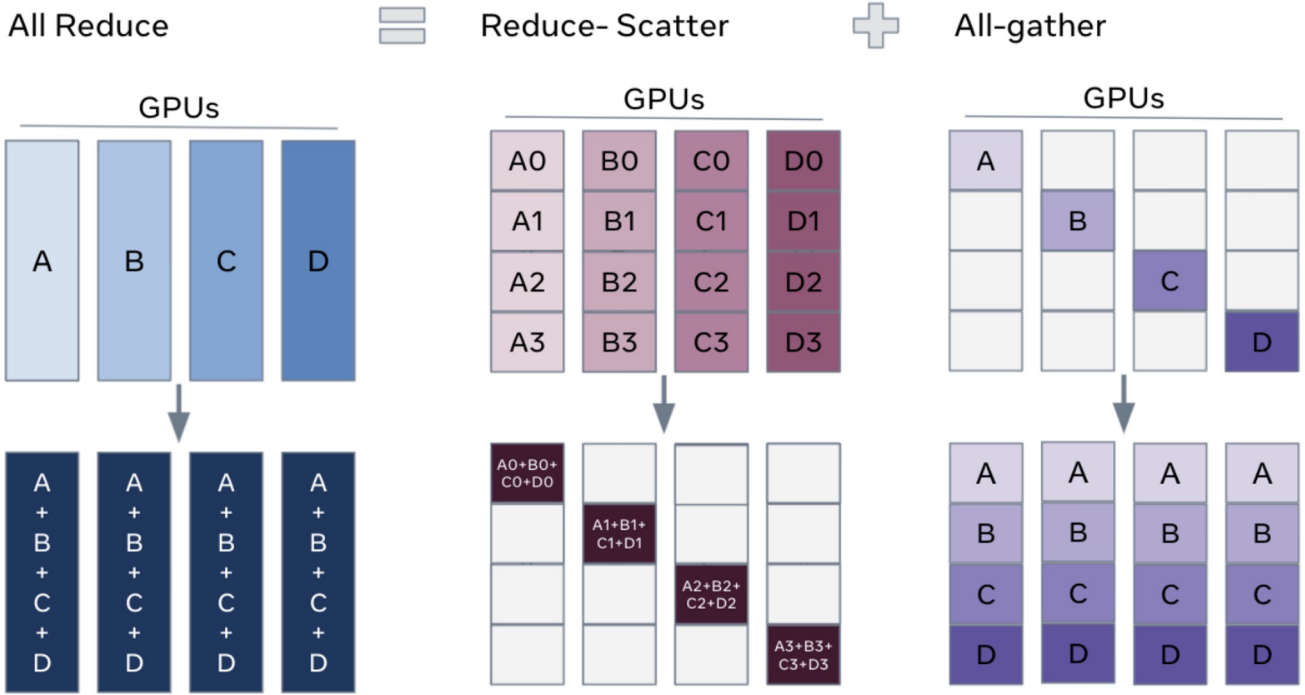
Job Communication

Fully sharded data parallel training



https://engineering.fb.com/2021/07/15/open-source/fsdp/

Job Communication



https://engineering.fb.com/2021/07/15/open-source/fsdp/

What goes wrong?

- “Hard” failures
  - Immediate job failure
  - GPU memory failures
  - GPU driver errors
  - NCCL/networking errors (NVIDIA Collective Communications Library)
- “Soft” failures
  - Performance degradation, e.g. low network throughput
- Somewhere in between
  - Job hangs due to CUDA or NCCL errors
  - NaN / incorrect computation results
  - May require application code to handle without intervention

What goes wrong?

Failure Symptoms	Failure Domain			Likely Failure Cause
	User Program	System Software	Hardware Infra	
OOM	✓	✗	✗	User Bug
GPU Unavailable	✗	✓	✓	PCIe error, Driver/BIOS, thermals
GPU Memory Errors	✗	✗	✓	Thermal Noise, Cosmic Rays, HBM Defect or Wear
GPU Driver/Firmware Error	✗	✓	✗	Outdated Software, High Load
GPU NVLink Error	✗	✗	✓	Electro/Material Failure, Switch
Infiniband Link	✗	✗	✓	Electro/Material Failure, Switch
Filesystem Mounts	✗	✓	✗	Failed Frontend Network, Drivers in D State, Storage Backend
Main Memory Errors	✗	✗	✓	Circuit Wear, Thermal Noise, Cosmic Rays
Ethlink Errors	✗	✗	✓	Electro/Material Failure, Switch
PCIe Errors	✗	✗	✓	GPU Failure, Poor Electrical Contacts
NCCL Timeout	✓	✓	✓	Userspace Crash, Deadlock, Failed HW
System Services	✓	✓	✓	Userspace Interference, Software Bugs, Network Partition

Llama 3: https://arxiv.org/pdf/2410.21680v1

Example: NCCL_CROSS_NIC

- Increase in step time latency
- Only above specific job size (thousands of GPUs)
- Isolate and reproduce -> NCCL tests
- Topology detection/graph search suboptimal for large trees

Fixed Issues

The following issues have been resolved in NCCL 2.23.4:

- Fixed GPU Direct RDMA check on linux kernels 6.6+.
- Fixed performance regression when mixing small and large operations.
- Fixed crash in topology detection when devices have a NUMA ID of -1.
- Fixed Tree graph search when NCCL_CROSS_NIC is set to 1..
- Fixes for IB operation on multi-node NVLink systems.

Job Communication

Bus bandwidth

While the algorithm bandwidth makes sense for point-to-point operations like Send/Receive, it is not always helpful to measure collective operations speed, since the theoretical peak algorithm bandwidth is not equal to the hardware peak bandwidth, usually depending on the number of ranks. Most benchmarks only provide time measurements, which is hard to interpret for large sizes. Some others also provide algorithms bandwidth, but see that depending on the number of ranks, that bandwidth varies (and decreases as the number of ranks increase).

To provide a number which reflects how optimally the hardware is used, NCCL tests introduce the notion of "Bus Bandwidth" ("busbw" column in the tests output). This number is obtained applying a formula to the algorithm bandwidth to reflect the speed of the inter-GPU communication. Using this bus bandwidth, we can compare it with the hardware peak bandwidth, independently of the number of ranks used.

https://github.com/NVIDIA/nccl-tests/blob/master/doc/PERFORMANCE.md

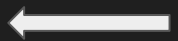
Example: MPIJob + NCCL test


```yaml
apiVersion: kubeflow.org/v2beta1
kind: MPIJob
metadata:
  name: nccl-test-64-h100
spec:
  mpiReplicaSpecs:
    Launcher:
      replicas: 1
      template:
        spec:
          containers:
            - command: ["/bin/bash", "-c"]
              args: [
                  "mpirun \
                  -np 64 \
                  -bind-to none \
                  -x LD_LIBRARY_PATH \
                  -x NCCL_SOCKET_IFNAME=eth0 \
                  -x NCCL_IB_HCA=ibp \
                  -x UCX_NET_DEVICES=ibp0:1,ibp1:1,ibp2:1,ibp3:1,ibp4:1,ibp5:1,ibp6:1,ibp7:1
\
                  -x SHARP_COLL_ENABLE_PCI_RELAXED_ORDERING=1 \
                  -x NCCL_COLLNET_ENABLE=0 \
                  /opt/nccl_tests/build/all_reduce_perf -b 512M -e 8G -f 2 -g 1
                  ",
                ]
              ...
    Worker:
      replicas: 8
      template:
        spec:
          containers:
            - resources:
                requests:
                  cpu: 110
                  memory: 960Gi
                  nvidia.com/gpu: 8
                  rdma/ib: 1
                limits:
                  memory: 960Gi
                  nvidia.com/gpu: 8
                  rdma/ib: 1
          ...
```

Example: MPIJob + NCCL test

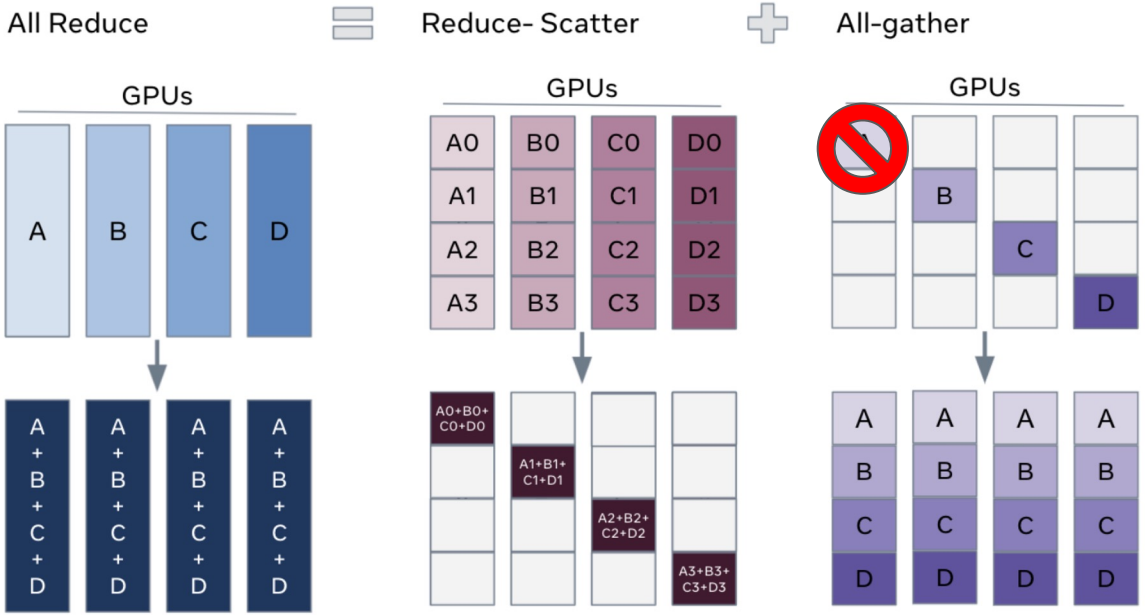

```
nThread 1 nGpus 1 minBytes 536870912 maxBytes 8589934592 step: 2(factor) warmup iters: 5 iters: 20 agg iters: 1 validation: 1 graph: 0
Using devices
 Rank  0 Group  0 Pid     22 on ace-nccl-test-64-h100-worker-0 device  0 [0x1a] NVIDIA H100 80GB HBM3
 Rank  1 Group  0 Pid     23 on ace-nccl-test-64-h100-worker-0 device  1 [0x40] NVIDIA H100 80GB HBM3
 Rank  2 Group  0 Pid     24 on ace-nccl-test-64-h100-worker-0 device  2 [0x53] NVIDIA H100 80GB HBM3
 Rank  3 Group  0 Pid     25 on ace-nccl-test-64-h100-worker-0 device  3 [0x66] NVIDIA H100 80GB HBM3
...
 Rank 25 Group  0 Pid     23 on ace-nccl-test-64-h100-worker-3 device  1 [0x40] NVIDIA H100 80GB HBM3
 Rank 26 Group  0 Pid     24 on ace-nccl-test-64-h100-worker-3 device  2 [0x53] NVIDIA H100 80GB HBM3
 Rank 27 Group  0 Pid     25 on ace-nccl-test-64-h100-worker-3 device  3 [0x66] NVIDIA H100 80GB HBM3
 Rank 28 Group  0 Pid     26 on ace-nccl-test-64-h100-worker-3 device  4 [0x9c] NVIDIA H100 80GB HBM3
 Rank 29 Group  0 Pid     27 on ace-nccl-test-64-h100-worker-3 device  5 [0xc0] NVIDIA H100 80GB HBM3
 Rank 30 Group  0 Pid     29 on ace-nccl-test-64-h100-worker-3 device  6 [0xd2] NVIDIA H100 80GB HBM3
 Rank 31 Group  0 Pid     31 on ace-nccl-test-64-h100-worker-3 device  7 [0xe4] NVIDIA H100 80GB HBM3


                                                     out-of-place                       in-place
      size         count      type   redop    root     time   algbw   busbw #wrong     time   algbw   busbw #wrong
       (B)    (elements)                               (us)  (GB/s)  (GB/s)            (us)  (GB/s)  (GB/s)
 536870912     134217728     float     sum      -1   3106.8  172.80  334.81      0   3105.9  172.85  334.90      0
1073741824     268435456     float     sum      -1   5417.2  198.21  384.03      0   5420.3  198.10  383.81      0
2147483648     536870912     float     sum      -1    10708  200.54  388.55      0    10705  200.60  388.66      0
4294967296    1073741824     float     sum      -1    21445  200.28  388.04      0    21401  200.69  388.83      0
8589934592    2147483648     float     sum      -1    42636  201.47  390.35      0    42633  201.49  390.38      0
Out of bounds values : 0 OK
Avg bus bandwidth    : 377.237
```



not bad!

Job Communication



What happens?

Usually: job dies

On a job with thousands of GPUs: $$$ wasted

https://engineering.fb.com/2021/07/15/open-source/fsdp/

How do we manage failure?

- In the application
  - Checkpoint + restore progress after restart
  - Detect silent data corruption
- Active + passive node health checks
- Remove unhealthy nodes from job scheduling
- Repair/remediate unhealthy nodes to return to service

Demo: JobSet restart + checkpointing

~/code/demo git:(main)±3

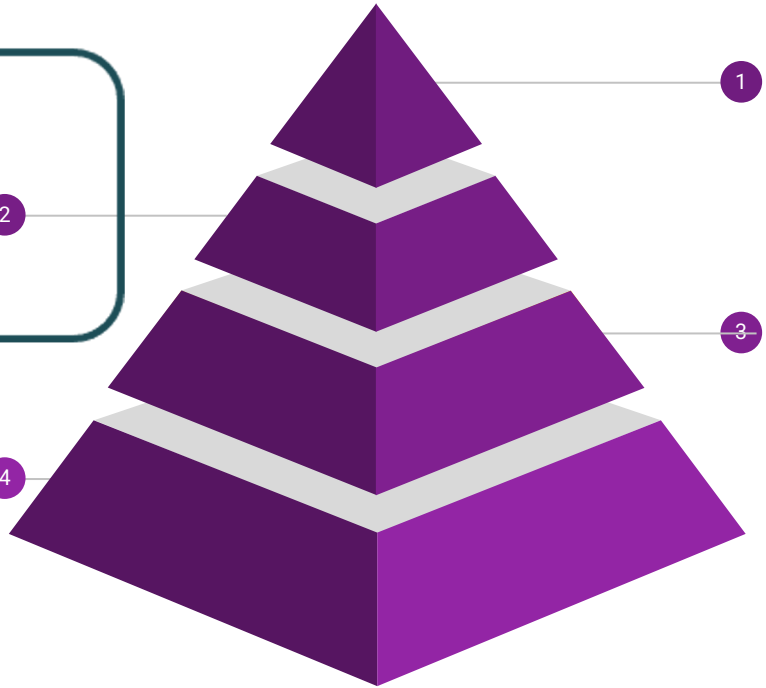
```
./demo.sh
apiVersion: jobset.x-k8s.io/v1alpha2
kind: JobSet
metadata:
  name: ace-jax-demo
  labels:
    kueue.x-k8s.io/queue-name: cw-infra-triage-queue
spec:
  network:
    enableDNSHostnames: true
  failurePolicy:
    maxRestarts: 10
  startupPolicy:
    startupPolicyOrder: InOrder
  replicatedJobs:
  - name: jax
    template:
      metadata:
        labels:
          kueue.x-k8s.io/queue-name: cw-infra-triage-queue
      spec:
        parallelism: 2
        completions: 2
        backoffLimit: 0
        template:
          metadata:
            labels:
              kueue.x-k8s.io/queue-name: cw-infra-triage-queue
          spec:
            restartPolicy: Never
            terminationGracePeriodSeconds: 1
            volumes:
            - name: jax-code
              configMap:
```


KubeCon
CloudNativeCon
North America 2024
Infra Provider
Perspective

Layers of Failure Detection

Orchestrator
Node health checks for node and GPU readiness.

Physical Clusters
Perf tests during onboarding

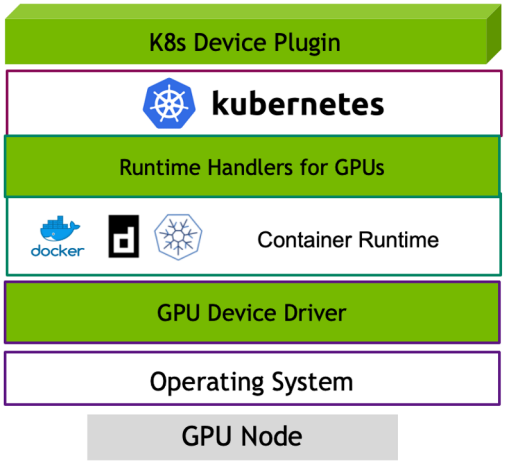


Workload
Health check before running

Virtual Machine Deployment
Prevent VMs from being allocated to the orchestrator

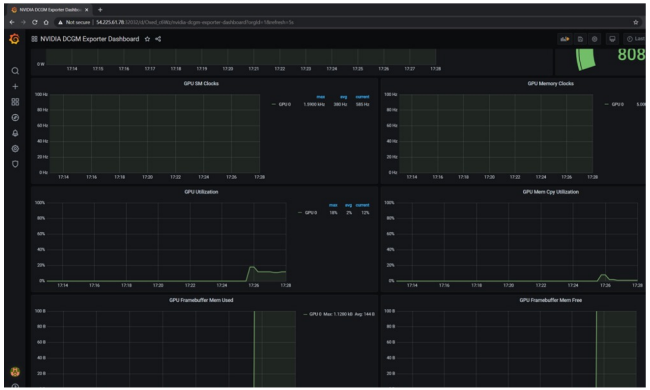
Initialization and Lifecycle Health Checks

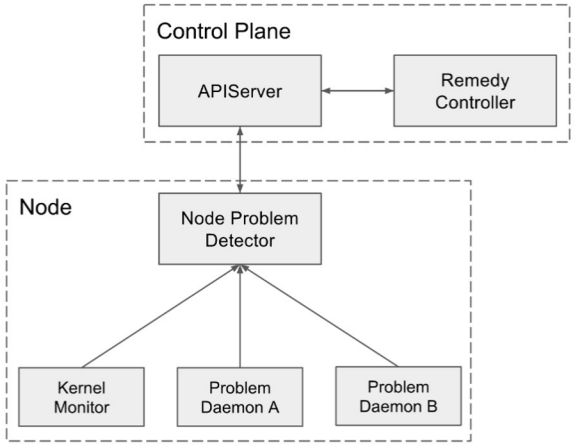
Initialization issues



Device plugin

Lifecycle issues





GPU monitoring

[kubernetes/node-problem-detector]

Node Problem Detector

Conditions

```
Conditions:
  Type                          Status  LastHeartbeatTime                 LastTransitionTime                Reason                          Message
  ----                          ------  -----------------                 ------------------                ------                          -------
  ReadonlyFilesystem            False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   FilesystemIsNotReadOnly         Filesystem is not read-only
  VMEventScheduled              False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:55:36 -0700   NoVMEventScheduled              VM has no scheduled event
  FrequentDockerRestart         False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   NoFrequentDockerRestart         docker is functioning properly
  FilesystemCorruptionProblem   False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   FilesystemIsOK                  Filesystem is healthy
  KernelDeadlock                False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   KernelHasNoDeadlock             kernel has no deadlock
  ContainerRuntimeProblem       False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   ContainerRuntimeIsUp            container runtime service is up
  FrequentUnregisterNetDevice   False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   NoFrequentUnregisterNetDevice   node is functioning properly
  FrequentContainerdRestart     False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   NoFrequentContainerdRestart     containerd is functioning properly
  KubeletProblem                False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   KubeletIsUp                     kubelet service is up
  FrequentKubeletRestart        False   Fri, 16 Aug 2024 18:58:31 -0700   Thu, 15 Aug 2024 16:54:52 -0700   NoFrequentKubeletRestart        kubelet is functioning properly
  NetworkUnavailable            False   Thu, 15 Aug 2024 16:51:53 -0700   Thu, 15 Aug 2024 16:51:53 -0700   RouteCreated                    RouteController created a route
  MemoryPressure                False   Fri, 16 Aug 2024 18:56:44 -0700   Thu, 15 Aug 2024 16:51:00 -0700   KubeletHasSufficientMemory      kubelet has sufficient memory availabl
e
  DiskPressure                  False   Fri, 16 Aug 2024 18:56:44 -0700   Thu, 15 Aug 2024 16:51:00 -0700   KubeletHasNoDiskPressure        kubelet has no disk pressure
  PIDPressure                   False   Fri, 16 Aug 2024 18:56:44 -0700   Thu, 15 Aug 2024 16:51:00 -0700   KubeletHasSufficientPID         kubelet has sufficient PID available
  Ready                         True    Fri, 16 Aug 2024 18:56:44 -0700   Thu, 15 Aug 2024 16:51:10 -0700   KubeletReady                    kubelet is posting ready status. AppAr
mor enabled
```

Default checks are not GPU specific

GPU Health Checks

Node-level health checks that can be run across the fleet

- Custom health checks:
  - Cron jobs with health checks for specific GPU and network config
- LBNL Node Health Checks (open-source)
  - Reliable (with timers to prevent hangs), extensible and reusable
  - Integrations with common HPC workload managers like SLURM and TORQUE
- Azure HPC health checks (open-source and experimental)
  - Extension of LBNL's node health check
  - Tested with multiple Nvidia A100 and H100 GPUs, and AMD MI300X

Example Health Checks

Component Tested	nd96asr_v4 expected	nd96amsr_a100_v4 expected	nd96isr_h100_v5 expected
GPU count	8	8	8
NVlink	no inactive links	no inactive links	no inactive links
GPU XID errors	not present	not present	not present
Nvidia-smi GPU health check	pass	pass	pass
GPU DtH/HtD bandwidth	23 GB/s	23 GB/s	52 GB/s
GPU Mem Errors (ECC)	20000000	20000000	20000000
GPU Throttle codes assertion	not present	not present	not present
GPU NVLink bandwidth	228 GB/s	228 GB/s	460 GB/s
IB device (GDR) bandwidth	180 GB/s	180 GB/s	380 GB/s
IB device (non GDR) bandwidth	NA	NA	NA
GPU/GPU Direct RDMA(GDR) + IB device bandwidth	18 GB/s	18 GB/s	NA
IB/GPU device topology/PCIE mapping	pass	pass	pass
IB link flap occurrence	not present	not present	not present
CPU compute/memory bandwidth	NA	NA	NA

Source (public): [azurehpc-health-checks]

Custom Plugin for NPD: GPU Count

azurehpc / experimental / aks_npd_draino / npd / deployment / **node-problem-detector-config.yaml**

Code | Blame | 851 lines (787 loc) · 27.7 KB

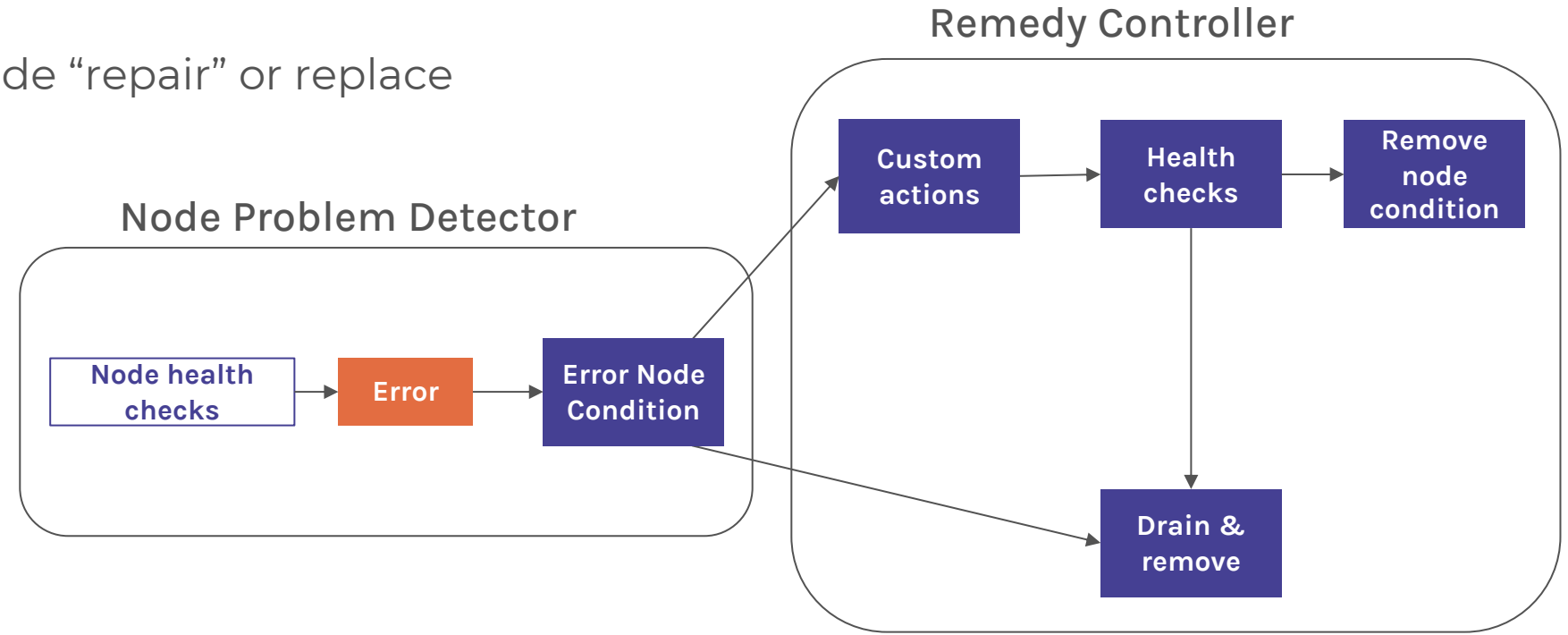
```
147       fi
148     custom-plugin-gpu-count.json: |
149       {
150         "plugin": "custom",
151         "pluginConfig": {
152         "invoke_interval": "60s",
153         "timeout": "20s",
154         "max_output_length": 80,
155         "concurrency": 3,
156         "enable_message_change_based_condition_update": false
157         },
158         "source": "custom-plugin-gpu-count",
159         "metricsReporting": false,
160         "conditions": [
161           {
162             "type": "GpuCount",
163             "reason": "GpuCountGood",
164             "message": "GPU count is correct"
165           }
166         ],
167         "rules": [
168           {
169             "type": "temporary",
170             "reason": "GpuCountBad",
171             "path": "./config/plugin/check_gpu_count.sh",
172             "timeout": "10s"
173           },
174           {
175             "type": "permanent",
176             "condition": "GpuCount",
177             "reason": "GpuCountBad",
178             "path": "./config/plugin/check_gpu_count.sh",
179             "timeout": "10s"
180           }
181         ]
182       }
183     check_gpu_count.sh: |
184       #!/bin/bash
185
186       # This plugin checks if is the VM has the correct number of GPU's
```

Custom plugin for NPD: XID error check


```
328       check_gpu_xid.sh: |
329         #!/bin/bash
330         # This plugin checks GPU XID errors
331         readonly OK=0
332         readonly NONOK=1
333         readonly UNKNOWN=2
334
335         # time threshold in hours
336         readonly time_threshold=2
337         readonly logfile="/var/log/azakslog"
338         readonly kernel_log="var/log/syslog"
339         readonly XID_EC="48 56 57 58 62 63 64 65 68 69 73 74 79 80 81 92 119 120"
340         readonly GPU_XID_TEST="GPU Xid errors detected"
341
342
343         if [[ ! -f $kernel_log ]]; then
344            echo "$kernel_log not found. Skipping GPU Xid error test."
345            exit $NONOK
346         fi
347
348         # check for any xid errors
349         grep -q "Xid" $kernel_log
350         RC=$?
351         if [ $RC == 0 ]; then
352            for XID in $XID_EC; do
353                xid_found_line=$(grep "Xid.*: $XID," $kernel_log  | tail -n 1)
354                if [ "$xid_found_line" != "" ]; then
355                   logXid=$(echo "$xid_found_line" | awk -F ',' '{print $1}' )
356                   logMsg="Found XID: $logXid"
357                   log_date="$(echo "$logXid" | awk '{print $1, $2, $3}') $(date +"%Y")"
358                   log_date=$(date -d "$log_date" +"%s")
359                   current_ts=$(date +"%s")
360                   diff=$(( (current_ts - log_date) / 3600 ))
361
362                   if [ "$diff" -le $time_threshold ]; then
363         # check if the XID has been reported in the log before
364                      if grep -qF "$logMsg" "$logfile"; then
365                         echo "This XID has been reported before: $logXid."
366                      else
367                         echo "$logMsg" >> $logfile
```

Remedy Controller

Node “repair” or replace



Testing by simulating failures


```
root@nvidia-driver-daemonset-r8gbg:/drivers# nvidia-smi drain -p 0001:00:00.0 -m 0
Successfully set GPU 00000001:00:00.0 drain state to: not draining.
root@nvidia-driver-daemonset-r8gbg:/drivers# nvidia-smi
Fri Aug 23 06:00:31 2024
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 550.90.07              Driver Version: 550.90.07      CUDA Version: 12.4     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  NVIDIA A100-SXM4-40GB           Off |   00000001:00:00.0 Off |                    0 |
| N/A   37C    P0             60W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   1  NVIDIA A100-SXM4-40GB           On |   00000002:00:00.0 Off |                    0 |
| N/A   36C    P0             55W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   2  NVIDIA A100-SXM4-40GB           On |   00000003:00:00.0 Off |                    0 |
| N/A   36C    P0             55W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   3  NVIDIA A100-SXM4-40GB           On |   00000004:00:00.0 Off |                    0 |
| N/A   36C    P0             59W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   4  NVIDIA A100-SXM4-40GB           On |   0000000B:00:00.0 Off |                    0 |
| N/A   36C    P0             56W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   5  NVIDIA A100-SXM4-40GB           On |   0000000C:00:00.0 Off |                    0 |
| N/A   35C    P0             57W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   6  NVIDIA A100-SXM4-40GB           On |   0000000D:00:00.0 Off |                    0 |
| N/A   36C    P0             54W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
|   7  NVIDIA A100-SXM4-40GB           On |   0000000E:00:00.0 Off |                    0 |
| N/A   36C    P0             57W /  400W |       1MiB /  40960MiB |      0%      Default |
|                                         |                        |             Disabled |
+-----------------------------------------+------------------------+----------------------+
```

GPU drop testing

Network load testing

Virtual nodes: Kubernetes WithOut Kubelet

Demo: GPU Node Problem Detector + Draino + Drop GPU

```
aganeshkumar@Ganeshkumars-MacBook-Pro ~ % kubectl get nodes
```

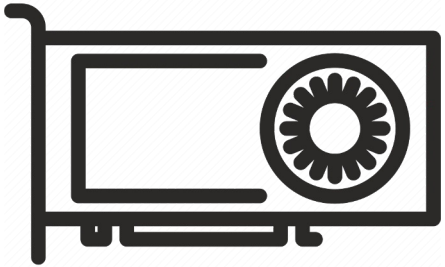
Recap: managing failure

- Workload checkpoints/restores across restarts
- Oops – a pod landed on the same bad node!
- NPD + Draino -> remove bad hardware
- App + Infra working together == Happy MLEs!

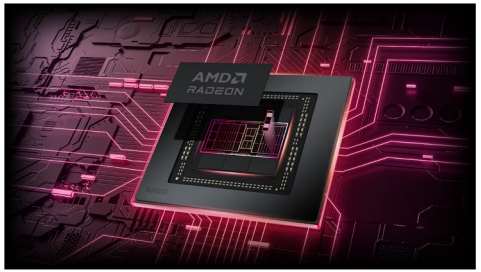
Yes, this is modern AI xD

Advanced developments and Ecosystem gaps

Gap: GPU & k8s + Heterogeneous nodes



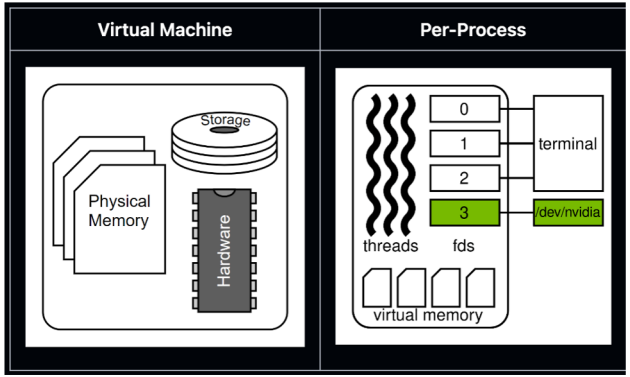
Better integrations for GPU in k8s



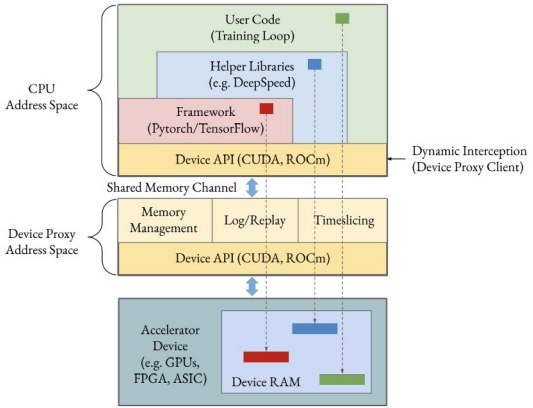
Vendor-neutral health check methods

GPU Checkpoint and Restore

Checkpoint/Restore in Userspace for GPUs



nvidia/cuda-checkpoint



Singularity: device-proxy

What else?

- Maintenance without disrupting jobs
  - Driver upgrade, node image changes
  - See Meta's maintenance trains
  - "How the Record-Breaking, Cloud Native AI Supercomputer Was Built - Peter Salanki, CoreWeave" (Kubecon 2023)
- Multi-node validation
  - NCCL tests across network/nodes with idle capacity
  - Megatron LM for real training validation
- UX for batch workloads still evolving
  - Kueue, JobSets, LWS working
  - Network/GPU topology aware scheduling


KubeCon
CloudNativeCon
North America 2024
Key Takeaways

Key Takeaways

- GPU failures are disruptive, and compound for ML training
- Multiple layers of health checks:
  - Applications - before running: Init containers and workload-specific checks
  - Orchestrator - while running: GPU node problem detector, and monitoring
- Resolution:
  - Checkpointing model weights
  - Remedy control systems - move workload
  - Prevent node reuse
- Ongoing work: making GPU health more k8s-native, vendor-agnostic health check, model checkpointing integrations, transparent GPU checkpointing


KubeCon
CloudNativeCon
North America 2024
Thank you!