Microsoft

HPC ClusterLevel Validation & Performance Benchmarking Superbench, NCCL, and Linpack Methodology

By: M. EL AZZOUZI

Scope: This section defines the **clusterwide validation and benchmarking phase** after individual node acceptance. It ensures the cluster as a whole meets or exceeds design targets in GPU, network, and systemlevel response performance.

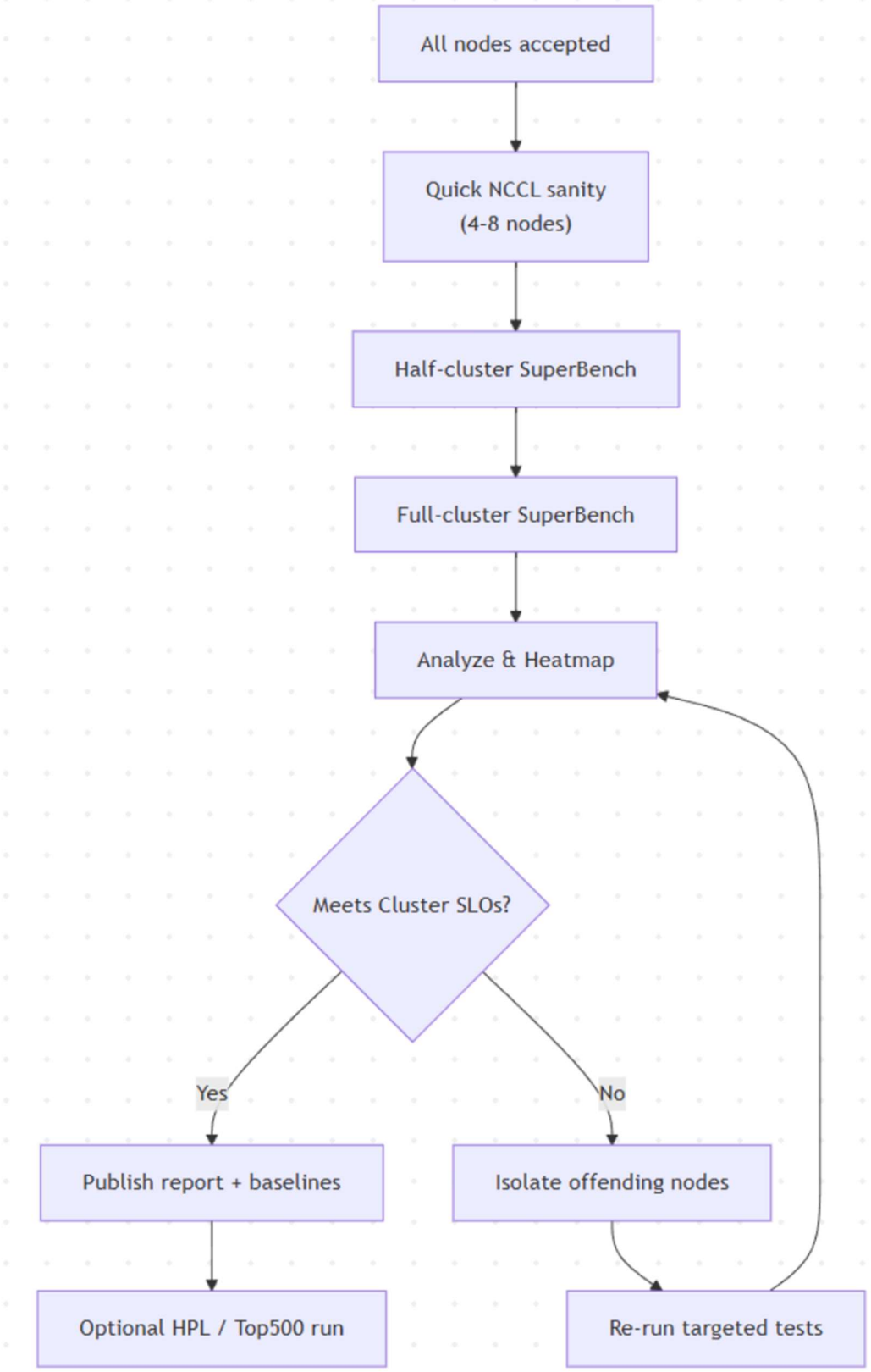
1) Objectives

- Verify **GPU fabric coherence** across the cluster (intranode and internode).
- Validate **NCCL, RDMA, and IB fabric health** at scale.
- Benchmark **endtoend performance** using **SuperBench** (synthetic + deep learning workloads).
- Detect topology or imbalance issues (e.g., link skew, GPUtoGPU latency asymmetry, NUMA misbinding).
- Generate an **automated health & performance heatmap** for management and future regression comparison.
- (Optional) Conduct **Top500/HPL benchmark** to rank the cluster performance against public metrics.

2) Tools and Frameworks

Component	Purpose	Notes
NCCL Tests	Validate allreduce, alltoall, broadcast, and reduce operations over IB/NVLink	Included in `nccl-tests` container or SuperBench suite
SuperBench	Endtoend benchmark suite for AI/HPC systems	Supports GPU, CPU, and network tests; JSON output with visualization; official NVIDIA + Microsoft collaboration
DCGM Exporter	Health metrics (temps, clocks, power, ECC, XIDs)	Should be enabled clusterwide during tests
ibdiagnet / ibqueryerrors	IB topology & counter verification	Part of MOFED; run before and after performance tests
Grafana / Prometheus dashboards	Visualization and anomaly detection	Used to produce performance heatmaps and live monitoring

HPL (Top500)	Optional largescale Linpack benchmark	To measure theoretical peak TFLOPS and compare against rankings


All nodes accepted
Quick NCCL sanity
(4-8 nodes)
Half-cluster SuperBench
Full-cluster SuperBench
Analyze & Heatmap
Meets Cluster SLOs?
Yes
No
Publish report + baselines
Isolate offending nodes
Optional HPL / Top500 run
Re-run targeted tests

Microsoft

3) PreValidation Steps

1. **Ensure all nodes are accepted** from the nodelevel playbook.
2. Confirm:
   - Fabric Manager, DCGM, and IB interfaces are healthy on all GPU nodes.
   - Kubernetes node labels reflect correct `gpu.count` and `ib.ports`.
   - RDMA is enabled: `/dev/infiniband/uverbs*` visible inside GPU containers.
   - NCCL P2P and socket networks are configured:
   - `export NCCL_DEBUG=INFO`
   - `export NCCL_IB_HCA=mlx5_0,mlx5_1`
   - `export NCCL_SOCKET_IFNAME=eth0`
   - `export NCCL_NET_GDR_LEVEL=2`
3. Sync all system clocks and NTP across nodes.
4. Record cluster topology:
5. `ibnetdiscover > ib_topo.txt`
6. `ibdiagnet -r > ib_route_health.txt`

4) Quick NCCL Fabric Sanity Test

Run a short distributed NCCL test job across representative nodes.

```
# 4node NCCL test example
kubectl create -f nccltest.yaml
```

Sample `nccl-test.yaml`:

```
apiVersion: batch.volcano.sh/v1alpha1
kind: Job
metadata:
  name: ncclsanity
  namespace: gcrdiag
spec:
  queue: gcrdiag
  minAvailable: 4
  plugins:
    ssh: []
  tasks:
  - replicas: 4
    name: ncclsanity
    template:
      spec:
        containers:
        - name: nccl
          image: nvcr.io/nvidia/nccl-tests:2.23.4-cuda12.4
          command: ["/bin/bash", "-c"]
```

```
    args:
      - >
        mpirun -np 32 --allow-run-as-root -bind-to none \
        ./build/all_reduce_perf -b 8 -e 1G -f 2 -g 8
  restartPolicy: Never
```

Expectations:

- All ranks initialize within seconds.
- Throughput within ±10% of lab baseline (e.g., ≥ 360 GB/s nodetonode on HDR400).
- No NCCL warnings about topology mismatch or IB channel fallback.

Outcome: A quick confidence check that GPUs, IB fabric, and NCCL runtime are coherent.

5) SuperBench Cluster Validation

SuperBench provides **modular validation and benchmarking** for both singlenode and fullcluster runs.

5.1 Deployment

Install via Helm:

```
helm repo add superbench https://superbench.github.io/charts
helm upgrade --install superbench superbench/superbench \
  --namespace superbench --create-namespace \
  --set cluster.mode=volcano --set benchmark.mode=distributed
```

5.2 Configuration

Edit `values.yaml` to specify tests and node pools:

```
superbench:
  enable: true
  mode: distributed
  node_selector:
    k8s.lambda.ai/node-type: gpu
  benchmarks:
    - name: gpu-burn
    - name: nccl-allreduce
    - name: memory-bandwidth
    - name: ib-loopback
    - name: kernel-launch
    - name: disk-bw
  duration: 1800
  output:
    format: json
```

```
save_path: /results/superbench-$(date +%F).json
```

5.3 Execution Modes

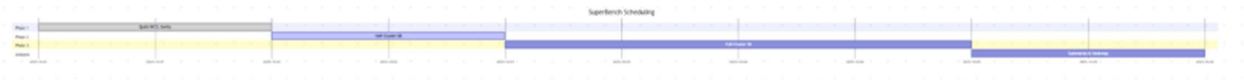
Mode	Description	Expected Runtime
Halfcluster test	Run on 50% of GPU nodes (balanced across racks)	~1 hour
Fullcluster validation	Run on all GPU nodes simultaneously	2–4 hours depending on scale

5.4 PostProcessing

```
superbench analyze result --input /results/*.json --output
/results/summary.xlsx
```

- Generates **performance summaries per node** (latency, bandwidth, FLOPS, memory BW, etc.).
- Integrate results into **Grafana heatmap dashboard**:
  - Xaxis: node index or hostname
  - Yaxis: test type (NCCL, IB, GPU, CPU)
  - Color: relative performance (green ≥ baseline, yellow = warning, red = regression)

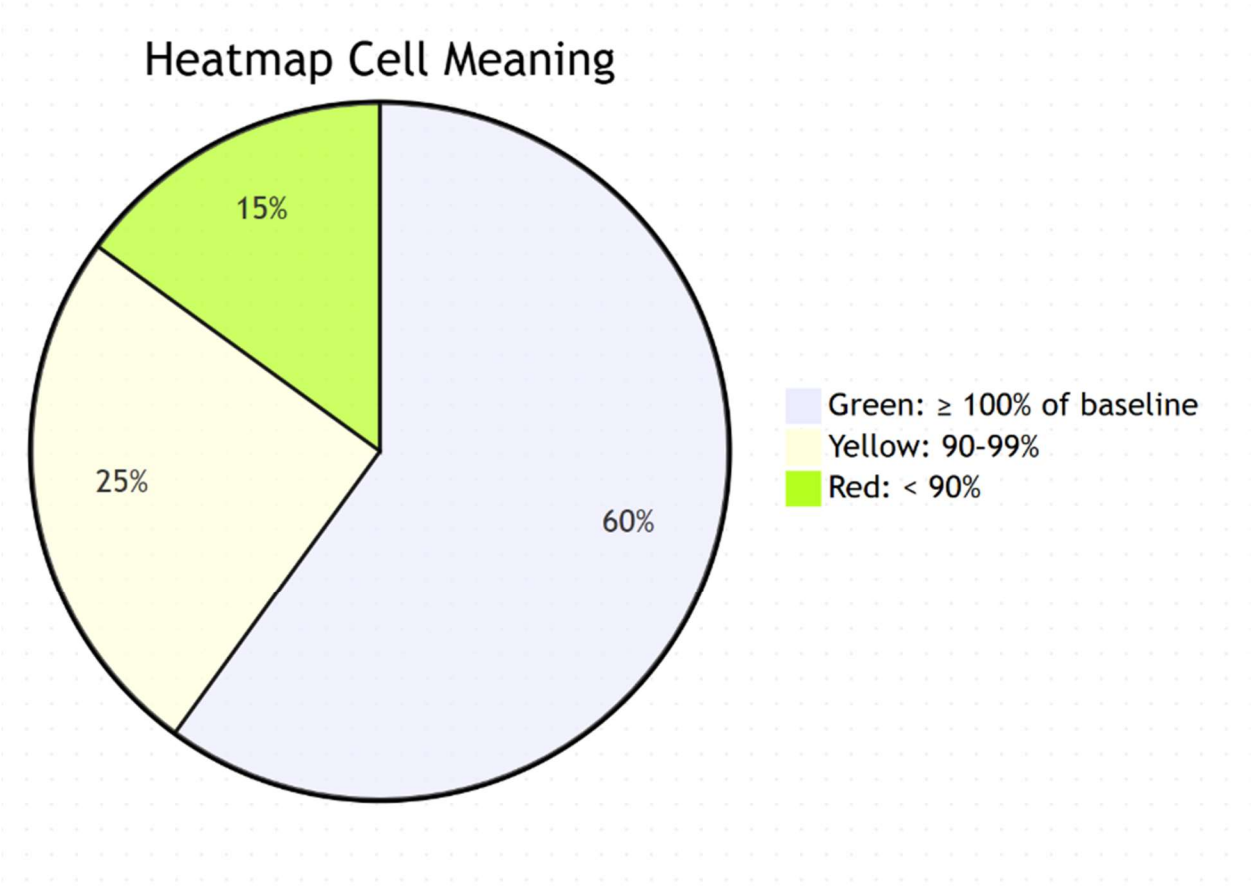
Deliverable: A visual **cluster health/performance heatmap** + Excel summary stored under `/validation_reports/<date>/`.



6) Additional Network & System Tests

Test	Tool	Command / Example	Pass Criteria
IB port error check	`ibqueryerrors`	`ibqueryerrors -v`	Zero critical counters before/after test
Fabric routing check	`ibdiagnet`	`ibdiagnet -r`	No invalid or missing routes
Latency per GPU pair	`nvsysdiag` or `nvlink_bandwidth`	`nvidia-smi nvlink -s`	No disabled NVLinks
Collective sync test	`alltoall_perf`	`mpirun -np <n>`	Stable mean latency
CPU/RAM imbalance	`numactl --hardware` + `stream`	`stream --benchmark`	Balanced throughput per socket
Filesystem IO	`fio`	See nodelevel tests	IO bandwidth ≥ baseline

Thermal drift	`dcgmi dmon`	During SuperBench run	Temp deltas < 10°C per node



7) Reporting Template

Section	Description	Data Source
Hardware Summary	GPU/CPU/RAM/IB per node	Inventory database
NCCL Quick Test	Sanity throughput (GB/s)	`nccl-tests` logs
SuperBench Summary	Pertest and pernode metrics	`superbench analyze` output
Heatmap	Visualized node health	Grafana or Excel heatmap
IB Topology	Connectivity map	`ibdiagnet` results
Regression Detection	Δ vs previous validation	SuperBench JSON diff
Anomalies	Xid, ECC, link errors	DCGM + ibqueryerrors logs
Recommendation	Accept / Rerun / Quarantine nodes	Engineer decision

8) Optional: Top500 / HPL Benchmark (If Time Allows)

Running a **High Performance Linpack (HPL)** test provides an approximate **Top500 ranking**.

8.1 Toolchain

- Container: `nvcr.io/nvidia/hpc-benchmarks:latest`
- Scheduler: run under Volcano with full GPU access.
- Config file: define problem size `N`, process grid `PxQ`, and block size `NB`.

8.2 Example Job

```
mpirun -np 256 --map-by ppr:8:node \
  ./xhpl
```

8.3 Output

- `Rmax` = measured TFLOPS
- `Rpeak` = theoretical TFLOPS
- Compute **Efficiency = Rmax/Rpeak** (target ≥ 85%)
- Compare against **Top500 list** of similar hardware generations.

8.4 Usefulness

- Validates CPU–GPU–network orchestration at extreme scale.
- Serves as a singlenumber metric for management and marketing.
- Archive the run under `/benchmarks/top500/<date>/`.

9) Acceptance Criteria (Cluster Level)

Category	Pass Condition	Action on Fail
GPU/NCCL	≥90% of baseline throughput; all ranks initialize	Investigate failing nodes; rerun after cordon
IB Fabric	0 critical errors; all ports at full speed	Replace cable/transceiver; revalidate
SuperBench summary	≥95% green cells in heatmap	Analyze yellow/red nodes
System stability	No node crash, Xid, or fabric reset during test	Quarantine failing nodes
Optional HPL	≥80% efficiency vs Rpeak	Log and compare vs historical

10) Deliverables

1. **SuperBench Report Bundle:** JSON + XLSX summaries.
2. **Cluster Heatmap:** PNG or Grafana dashboard snapshot.
3. **NCCL & IB logs:** archived under `/validation_reports/<date>/logs/`.

4. **Linpack Top500 report (if executed).**
5. **Executive Summary:** key scores, bottlenecks, actions.

11) Future Enhancements

- Integrate automated NCCL & SuperBench validation into CI/CD for new nodes.
- Baseline heatmap autopublish in Grafana after each intake.
- Autodetect regressions (e.g., via zscore anomaly detection on throughput).
- Use collected data for **capacity planning** and **vendor performance SLAs.**

End - Part 2