UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

Job-Monitoring @ PC2

Robert Schade, Michael Schwarz

Paderborn University, Germany
Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Paderborn Center for Parallel Computing

HPC Infrastructure and History

Photo: Jens Simon

Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Noctua 1 (seit 2019): Cray CS500

- 272 Nodes 2xIntel Skylake SP 6148
- Consumer GPUs (NVIDIA 2080ti,…)
- Omnipath 100

Noctua 2 (seit 2022): Sequana XH2000

- 1056 Nodes 2xAMD Milan 7763
- 32 Nodes 2xAMD Milan 7763+4x NVIDIA A100
- 16 Nodes 2xAMD Milan 7713+3xXilinx Alveo U280 FPGA
- 16 Nodes 2xAMD Milan 7713+2xIntel Stratix 10 GX 2800 FPGA
- Infiniband HDR100/200, optical Switch for FPGAs

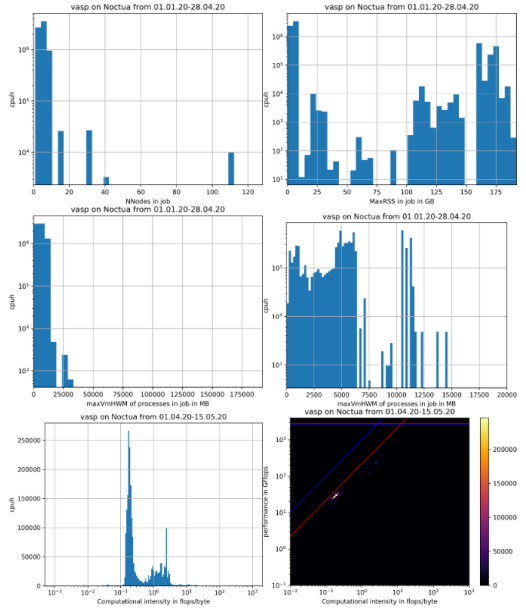
History of Job-Monitoring:

End 2019	Installation of ClusterCockpit for Noctua 1
2020-2021	Extension with recommendation rules, job detection
Mid 2022	Installation of ClusterCockpit for Noctua 2
Autumn 2022	Migration of Noctua 1 to Noctua 2 ClusterCockpit

Job-Monitoring

Past Uses of Monitoring Data:

- User support
- Jobmix definition
- Acquisition requirements (interconnect bandwidths, …)
- ARM workload exploration



Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Additional Future Use of Monitoring Data:

- Evaluation of adaptive CPU throttling (AMD HSMP)
- More recommendations

Current Challenges:

1. SLURM connection (preferred: squeue --json)
2. Node sharing
3. GPUs
4. FPGAs
5. Useful recommendations
6. Application detection

Job-Meta- und Metrikdaten
Tag 2: 8:30-9:00

NHR Monitoring Workshop 2022

Paderborn University, Germany
Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Paderborn Center for Parallel Computing

Überblick

- Metadaten
  - Notwendig:
    - jobId, user, project, cluster, subCluster, partition, numNodes, exclusive, startTime, stopTime, jobState, duration
    - Resources: CPU-core Allokationsinformation, GRES-Device Alloaktionsinformation (z.B. GPUs)
    - Statistics (mem_used, cpu_load, flops_any, mem_bw, net_bw, file_bw)

  - Optional/interessant:
    - Job's scontrol show job (insb. Working directory)
    - Tags
    - Jobskript
    - Job environment
    - genutzte Programme (als Tags)
    - geladene Umgebungsmodule (als Tags)

  - Weitere Vorschläge?

Siehe auch https://github.com/ClusterCockpit/cc-specifications/blob/master/datastructures/job-meta.schema.json

Überblick Metrikdaten

	Granularität				
	Node	Socket	Memory Domain	core	hwthread
flops_any	X	x	x	x	x
mem_bw	X	x	x		
net/file_bw	X				
ipc	X	x	x	x	x
cpu_used	X	x	x	x	x
cpu_load	X	x	x	x	x
flops_dp/sp	X	x	x	x	x
vectorization_ratio	X	x	x	x	x
cpu/mem_power	X	x			
acc_utilization	X				
acc_mem_used	X				
clock	X	x	x	x	x
eth_read/write_bw	X				
pfs_read/write_bw/req...	X				
fs_read/write_bw	X				
ic_read/write_bw	X				

Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Siehe auch https://github.com/ClusterCockpit/cc-specifications/blob/master/datastructures/job-metric-data.schema.json

Metrikdaten: Offene Fragen/Probleme

- Wie weit wollen wir in der Spezifikation noch gehen?

- Wie gehen wir mit Systemen um, die diese Definitionen nicht erfüllen können?
  Beispiel flops_any:= flops_dp+0.5*flops_sp bei systemen, die nur flops=flops_dp+flops_sp counten

- Node sharing
  - cpu_load, CPU-Core-Spezifität über /proc/schedstat möglich
  - PFS
    - Möglich mit Lustre Jobstat von OSTs/MDTs (Job-spezifische IO angaben)
  - Interconnect
    - Wahrscheinlich keine job-Spezifität ohne zusätzlichen Hardwaresupport möglich

- Accelerator-Metriken
  - weitere GPU Metriken (siehe nsight-systems)

Siehe auch https://github.com/ClusterCockpit/cc-specifications/blob/master/datastructures/job-metric-data.schema.json

SLURM Anbindung
Tag 2: 11:00-12:00

NHR Monitoring Workshop 2022

Paderborn University, Germany
Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Paderborn Center for Parallel Computing

Überblick

- Anforderungen
  - Bereitstellung von Metadaten
    - Allokation von CPU-Cores/GPUs/andere GRES
    - Job Details (aka. scontrol show job)
    - Jobskript/Job environment
    - …
  - geringer bis vernachlässigbarer Impact auf SLURM Performance (insb. slurmctld und slurmdbd)
  - möglichst Unabhängigkeit von SLURM-Version
  - Vermeidung von Diskrepanzen zwischen CC und SLURM (z.B. Zwei-Wege Abgleich)
  - Eventuell einfache Portierbarkeit auf andere WLMs
  - Weitere?

- Denkbare Mechanismen

  - PREP plugin (Frank Winkler)
  - REST api (Katrin Nusser)
  - squeue --json dump (Robert Schade)
  - via SPANK plugin beim Jobstart

Zusammenfassung und Nächste Schritte

Wichtige Fragen:

- Sollte eine (oder mehrere) Slurm-Anbindung(en) bei CC mitgeliefert werden?
- Welche existierende Lösung ist am perfomantesten?
- Was sind die Caveats/Einschränkungen bei den existierenden Lösungen?
- Wie einfach deploybar sind die existierenden Lösungen schon?
  - PREP plugin
  - REST api
  - squeue --json dump
  - via SPANK plugin beim Jobstart

UNIVERSITÄT PADERBORN
Die Universität der Informationsgesellschaft

SLURM-Connector via squeue --json

Robert Schade, Michael Schwarz

Paderborn University, Germany
Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

Paderborn Center for Parallel Computing

NHR Monitoring Workshop 2022

squeue --json

- Using SLURM 21.08 (with bugfix for openapi 0.0.37 by Michael Schwarz)


```
  "job_resources": {
  "nodes": "n2cn0544",
  "allocated_cpus": 4,
  "allocated_hosts": 1,
  "allocated_nodes": {
    "0": {
      "sockets": {
        "0": "unassigned"
      },
      "cores": {
        "0": "unassigned",
        "1": "unassigned",
        "2": "unassigned",
        "3": "unassigned"
      },
      "memory": 204800,
      "cpus": 4
    }
  }
},
….
  "gres_detail": [
    "gpu:a100:2(IDX:0-1)"
  ],
```


- ~300 line Python code currently used for sync incl. job script and env
- Currently in testing

squeue --json

- Benchmark Noctua 2:

Paderborn Center for Parallel Computing

NATIONALES HOCHLEISTUNGS RECHNEN

	Number of pending jobs	Number of running jobs	Wall time squeue in s	Wall time squeue --json in s	Lines in json
Normal operation	715	336	0.250	0.304	261895
Some single-core jobs	37072	9978	0.314	1.591	1569841
Many single-core jobs	21558	27803	0.595	3.971	3950265