Title: Automatic Detection of Pathological Jobs for HPC User Support

1 Overview

Target group:	*HPC Users & HPC-performance support staff*
Project objective:	*Establish patterns for performance bugs and develop change-templates to guide HPC users towards better efficiency and performance.*
Summary:	This project will develop an automated rule-based detection system for pathological HPC jobs. Examples for pathological jobs are over-parallelization, software pipelines with a decreasing degree of parallelism, or a lack of process binding. Detection of pathological jobs will be defined by parametrizable rules, which can be extended and specified by HPC experts. The detection system will support "action templates" that describe countermeasures for a specific pathological job category. The detection system will support automated actions, such as direct notification of HPC users and forwarding of completed action templates to assist in mitigating the detected issue. **Project duration/ planned period:**
12 months/ January to December 2023	

1.1 Participating NHR-centers:

Function	NHR-Center	URL
Project leader	NHR4CES@TUDa	
Project partner	PC2	
Project partner	NHR@FAU	
Project partner	NHR@ZIB	

2 General Project description

Performance tuning requires a thorough knowledge of the application algorithm, the target processor and system architecture, the programming language, and the software ecosystem, i.e. compilers, MPI, OpenMP, and CUDA. Experts in the field are few. The bulk of research groups consists of Ph.D. students, selected senior scientific staff, and professors that focus on their research question and view HPC systems as a tool to support their research efforts. Therefore, it is not surprising, that HPC support staff has to frequently deal with recurring HPC mistakes, such as over-parallelization, lack of pinning, binding, or NUMA-aware memory allocation. If the HPC center has a job-specific monitoring system in place, such mistakes are usually detected manually by experienced support staff looking at various metrics and searching for visual patterns. Training users in HPC helps to reduce the number of performance mistakes, but students on average leave the university after a few years.

An automated approach to detect pathological jobs will reduce the workload on the HPC support staff and provide users an automated feedback and mitigation recommendation. Frameworks for job-specific performance monitoring now offer sufficient data to enalbe automatic detection of pathological jobs in production HPC environments. Examples for such frameworks are ClusterCockpit [**?**][1], PIKA [**?**] or GEOPM [**?**][2].

[1] https://github.com/ClusterCockpit

[2] https://geopm.github.io/

2.1 Goal

The project *Automatic Detection of Pathological Jobs for HPC User Support* aims to develop a configurable automatic detection engine that uses cluster metrics to identify pathological HPC jobs. This detection engine will analyze job-specific metrics and apply *performance anti-patterns* (PAP) that describe the properties of a pathological HPC job. The design of the detection engine will be agnostic of the monitoring environment providing the metric and job metadata. A possible approach is the use of a generic data structure specification and exchange format description, such as the job archive format [3] developed in the NHR monitoring project. Such a format can then be generated from any monitoring solution, such as GeoPM, ClusterCockpit, PIKA or Prometheus.

The PAP used in the detection engine should not be hard-coded, but specified externally as rules, e.g. by HPC experts, and parameterized with properties specific to a given cluster. This parametrisation should enable quick porting of an PAP to another NHR system as ideally only the cluster-specific parameter need to be updated, if the proposed detection engine is used. For an initial set of PAPs we will draw on our own experience and previous work, such as the DFG projects ProfitHPC[4] and ProPE[5], that have already defined pathological job classes and their detection. In this project, the focus will be put on providing an initial set of pathological job patterns.

Knowledge about the specific application used in a job is important to increase the accuracy of pathological job detection. Relating performance patterns and job performance statistics to an application and as a consequence also to application classes or domains offers interesting opportunities for analysis and statistics. Therefore, this project will also provide a subsystem to automatically determine the applications of a job. This information is then added to the job meta data and can be later used in job classification rules.

The detection engine will support different courses of action in response to a detected pathological job. The implementation will provide examples for typical actions but also allow to tailor and configure the response at each site in accordance to local policies. However, the detection engine should explicitly support the documentation of detected pathology, and provide the ability to parameterize an *action template* (AT). An action template is designed in conjunction with the PAP and provides a documentation of the problem, possible steps to mitigate the problem and a suggested course of action. This would allow, for example, to create an action that directly notifies a user who submitted the pathological job with this information, e.g., by email. Other actions may notify the HPC support staff, e.g., by means of a ticket-system, or documentation in an event log.

2.2 Benefit & Value

The projects detection-engine, performance anti-patterns, and action templates will provide a helpful tool for HPC support staff and HPC users. Through the automated detection of pathological jobs from monitoring data and ready to use action templates, HPC staff is able to invest the freed time on the more time-consuming and challenging performance analysis and consulting work. Users benefit from immediate performance assessments of their jobs, including actionable feedback, enabling them to improve subsequent jobs directly and ideally without interaction with support staff. A user implementing an action template should improve the runtimes and resource usage of those jobs. The applicability of the proposed approach transcends NHR sites and should be applicable to all HPC centers, thus benefiting the whole HPC community at large.

2.3 Work packages

All project partners contribute to all work-packages; each work package is lead by a project partner.

[3] https://github.com/ClusterCockpit/cc-specifications

[4] https://profit-hpc.de/

[5] https://blogs.fau.de/prope/

WP1) **Monitoring Setup/Deployment & Output Normalization (TUDa):**

Description: To address the heterogeneous monitoring environments used at project partners sites, and in general at NHR sites, this work package will provide a suitable interchange format, capable of storing the metrics and meta-date required. WP1 will survey and assess existing solutions and if necessary augment or develop required functionality; a likely candidate is the job-archive format developed in the NHR monitoring project. This work package will also develop the necessary export tools to extract metrics from:

- GEOPM, used at TU Darmstadt,
- ClusterCockpit, used at FAU and Paderborn University, and
- Prometheus[6], used at ZIB.

To ensure all rules operate on the same metrics with identical meaning, a normalization pass will be developed, e.g. to make the output of ClusterCockpit, Prometheus and GEOPM comparable in terms of nomenclature and measured data. **Tasks:**

- survey, choose or develop intermediate storage format
- develop data exporter for monitoring solutions used by the project partners
- develop a normalization for all required metrics

WP2) **Application Detection (ZIB):**

Description: Supporters can make a better assessment of job performance when they know which application was used. Then, observed performance can be compared to the expected behavior of that application. Common configuration errors may be detectable from performance metrics, when the application is known. The ability to recognize the running applications and exposing that information as meta data will enable the design of specialized rules for specific applications and enhance the automated detection of performance issues and pathological jobs.

Tasks:

- choose and collect needed data of running processes
- design and implement the automated application detection framework
- author and apply rules

WP3) **Data Analysis / Identification of Relevant Performance Patterns (NHR@FAU):**

Description: This work package will define a set of pathological job patterns. To this end, it will build on previous work in this field and develop a job pattern repository, where for every pattern the aspects general description, symptoms, detection, possible fixes or mitigation, and examples, are described. The pattern specification mechanism should cover the majority of relevant pathological job (jobs that require immediate action) and should enable detection with sufficient accuracy using a minimum set of data sources.

Tasks:

- review of existing work regarding job patterns and classification
- develop and refine existing and new pathological job patterns and document them in a job pattern repository
- validate the job patterns and provide examples for illustration

[6] https://prometheus.io/

WP4) Development of Rule Format / Rule Evaluation Method (PC2):

Description: The performance patterns identified in WP3 need to be automatically detected with appropriate rules in the collected job metric data. This work package will develop and define a sufficiently expressive rule format for this purpose. One important aspect of the rule definition is that the cluster-specific parameters must be accounted for appropriately. Additionally, the rules should be able to generate a human-understandable output explaining why a certain rule has become active for a job so that HPC-support and users can understand it and judge if it's a false positive.

In addition to a rule format, an evaluation engine for the rules that can be easily integrated into existing job-monitoring systems will be implemented. The result of the detection engine will be passed to activities such as e-mail notifications or be transferred to an existing informational system.

Tasks:

- formalize the performance patterns found in WP3
- review existing rule definition formats
- define a portable rule format
- implement a rule-evaluation engine

WP5) Templates for Action Plans (NHR@FAU):

Description: This work package investigates and documents systematic action strategies triggered by detected pathological jobs, resulting in a general terminology and action plan framework for all pathological job patterns This involves the definition of the severity (either because of wasted resources or because other users are also affected), available communication channels, optimization or mitigation strategies, and knowledge management with documentation of cases, feedback processes, and statistical evaluation.

Tasks:

- develop general action plan workflow
- develop specific parameterized action plans for every pathological job pattern
- describe workflows for knowledge management of the actionable cases

2.4 Planned Communication Structure / Collaboration within the Project

Within the project a common room in Matrix Chat will be used for interactive communication. Monthly coordination calls will be used for progress-tracking and overall coordination. For this project, a kick-off meeting directly at the start of the project, as well as a mid-way meeting in summer 2023 are planned to forester collaboration. All work products, such as the detection engine, rule format, and action plan workflows are developed by all project partners collaboratively. The project will use GitLab for providing the necessary distributed development support, feature- and bug-tracking mechanisms.

2.5 Deliverables & Dissemination

Deliverables:

- A transfer process to convert a HPC system specific monitoring output into the input-format used for the detection engine.
- An application detection process, capable of clustering binaries according to relation to applications.
- An automated detection engine applying performance anti-patterns to extracted monitoring data.

- A pathological job pattern repository in a simple, portable format as, e.g., JSON files.
- Contribution of performance anti-patterns developed to the HPC-Wiki (https://hpc-wiki.info).
- Action templates including text templates and action plans as part of the pathological job pattern repository.

Dissemination:
All results will be released to the public, if possible as an publication to an appropriate venue, such as Supercomputing (SC) or International Supercomputing Conference (ISC). In particular, the developed *action templates* will be added to the HPC wiki, as well as documentation on how to setup and configure the detection framework on HPC-Systems. The source codes for all software artifacts, such as the converters, detection engine, and pathological job patterns will be release on an appropriate platform, such as GitHub.

2.6 Project Schedule

	Q1			Q2			Q3			Q4		
	1	2	3	4	5	6	7	8	9	10	11	12
WP1	■	■	■									
WP2	■	■	■	■	■	■	■	■	■			
WP3	■	■	■	■	■	■	■	■	■	■	■	
WP4			■	■	■	■	■	■	■	■		
WP5	■	■	■	■	■	■	■	■	■	■	■	■