Entwicklung und Automatisierung von HPC-Systemen im Rahmen der Praxisphase im Studiengang Ingenieurinformatik

P. Hertzel[1], T. Hilbig[1], C.Schröder[1,2]

[1] Hochschule Bielefeld – Univiersity of Applied Sciences and Arts, Bielefeld Institute for Applied Materials Research, Computational Materials Science and Engineering, Bielefeld Center for High Performance Simulation Studies, Department of Engineering Sciences and Mathematic, Germany

[2] Bielefeld University, Faculty of Physics, Germany

Einleitung:

Die dargestellten Arbeiten sind im Rahmen der Praxisphase des Studiengangs Ingenieurinformatik im Umfeld des BiCeHPSS für das Projekt PolyCycle entstanden.
Im Mittelpunkt stand die praktische Anwendung ingenieurwissenschaftlicher und informatikbezogener Methoden zur Entwicklung und Automatisierung technischer Systeme im Umfeld des High Performance Computing (HPC). Ziel der Projekte war es, bestehende Prozesse in Bezug auf Effizienz, Skalierbarkeit und Reproduzierbarkeit zu optimieren, sowie neue Automatisierungslösungen zu implementieren, die den Betrieb und die Nutzung der HPC-Systeme des BiCeHPSS vereinfachen. Im Verlauf der Praxisphase wurden mehrere Teilprojekte umgesetzt, die sich jeweils mit unterschiedlichen Aspekten der Systemkonfiguration, der Softwarebereitstellung und der Lehrunterstützung befassten.
Dazu gehörten die automatisierte Konfiguration von Serverhardware, die Entwicklung eines DevOps gesteuerten Build-Prozesses zur Erstellung von HPC-Compute-Node-Images sowie die Implementierung einer Benchmark-App für das Open OnDemand HPC-Webportal des Clusters.

Automatisierte Serverkonfiguration mittels Ansible, Rundeck und Azure-DevOps

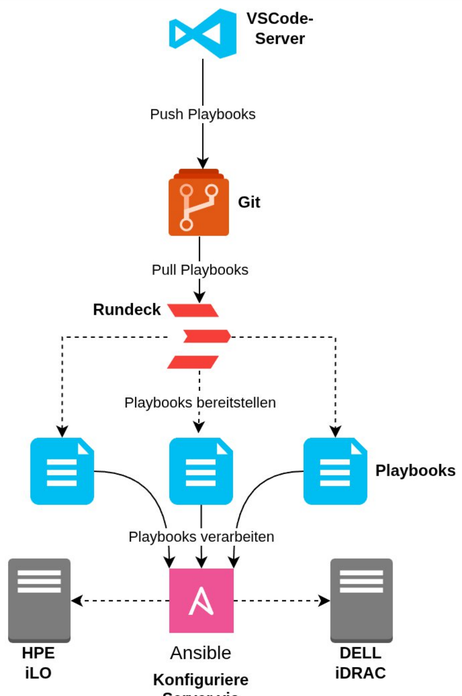

Abbildung 1: Konfigurations Workflow mit Rundeck und Ansible

In High Performance Computing Umgebungen ist die konsistente und fehlerfreie Bereitstellung einer großen Anzahl von Servern entscheidend für die Stabilität, Leistung und Wartbarkeit der Systeme.

Manuelle Konfigurationsverfahren sind dabei zeitaufwendig und fehleranfällig, was insbesondere bei unterschiedlichen Hardware-konfigurationen zu Abweichungen in den Systemeinstellungen führen kann.

Aus diesem Grund wurde eine Automatisierungsumgebung entwickelt, welche eine einheitliche Konfiguration von Servern verschiedener Hersteller ermöglicht. Dabei werden grundlegende Systemeinstellungen, wie beispielsweise BIOS-Einstellungen und Out-of-Band-Management-Konfigurationen, zentral verwaltet.

Zur Umsetzung wurde eine Umgebung auf Basis von Ansible, einer Automatisierungs-Engine und Rundeck, einer Automatisierungs-plattform, realisiert. Diese kommuniziert über die standardisierte Redfish-API direkt mit den Management-Controllern der Server (HPE iLO und Dell iDRAC).

Die sog. Playbooks (Beschreibungen der Konfiguration) werden dabei in Visual Studio Code entwickelt und anschließend mittels Git versioniert. Rundeck übernimmt daraufhin die automatisierte Bereitstellung und Ausführung dieser Playbooks. Dabei werden die Skripte aus dem Repository abgerufen, von Ansible verarbeitet und an die Zielsysteme verteilt.

Automatisierter Build- und Bereitstellungsprozess für HPC-Compute-Node-Images

Für den stabilen und effizienten Betrieb eines HPC-Clusters ist es erforderlich, dass alle Compute-Nodes (Rechenknoten) auf einer identischen und reproduzierbaren Systemumgebung basieren.

Um die Softwarebasis der Compute-Nodes zu vereinheitlichen und den manuellen Aufwand bei der Bereitstellung des Systems zu reduzieren, wurde ein automatisierter Prozess zur Erstellung und Bereitstellung von Compute-Node-Images entwickelt. Diese Images werden zentral verwaltet und versioniert. Das Cluster-Management-System Warewulf verwendet diese anschließend für die automatisierte Provisionierung.

Zur Umsetzung wird eine Azure-DevOps-Pipeline genutzt, die den gesamten Build-Prozess automatisiert steuert. Grundlage hierfür sind die im Git abgelegten Dockerfiles, die den Inhalt der System-Images enthalten.

Ein Azure-Agent auf einem Hypervisor mit Rocky Linux 9 führt den Build-Prozess aus und erstellt daraus mehrere Docker-Images. Anschließend werden die erzeugten Images auf einen Netzwerk-Share des Clusters übertragen.

Von dort aus ruft Warewulf die aktuellen Images ab und stellt sie den Compute-Nodes per PXE-Boot zur Verfügung. Dadurch ist gewährleistet, dass alle Compute-Nodes konsistent auf derselben Systemumgebung basieren und die Provisionierung der Systeme standardisiert und reproduzierbar erfolgt.

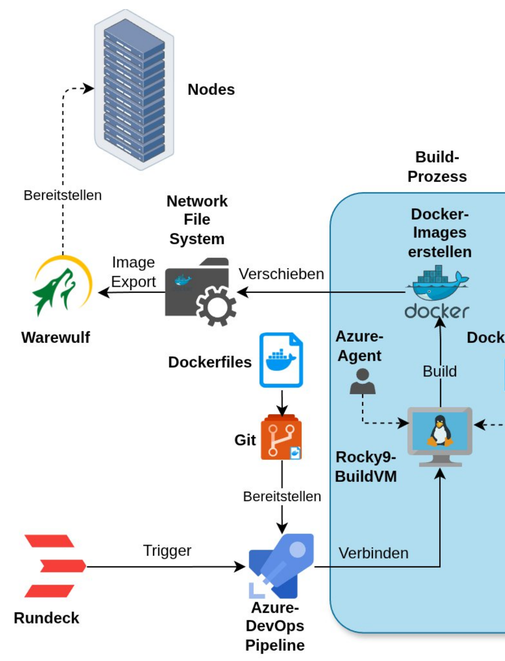

Abbildung 2: Ablauf des Build- und Bereitstellungsprozesses

Entwicklung einer Open OnDemand (HPC-Webportal) basierten Webanwendung zur Auswertung von HPC-Jobs am Beispiel der Monte-Carlo-π-Berechnung

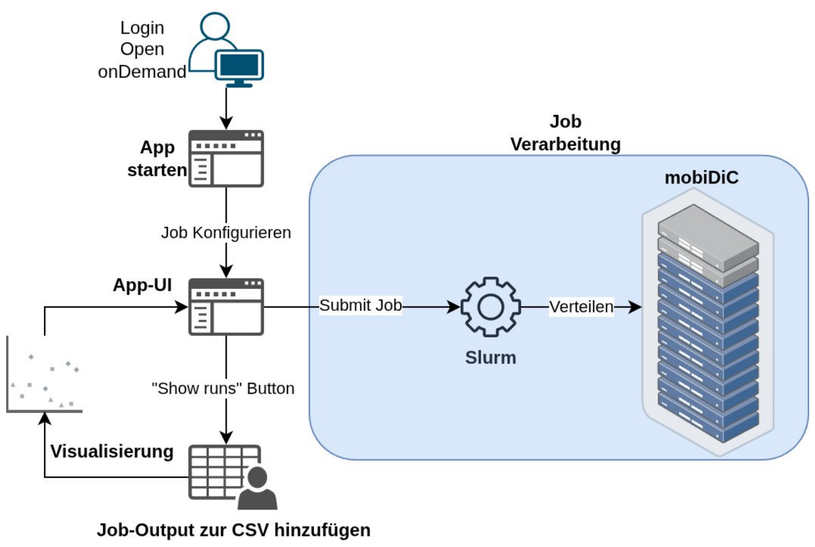

Abbildung 3: Passenger-App Workflow

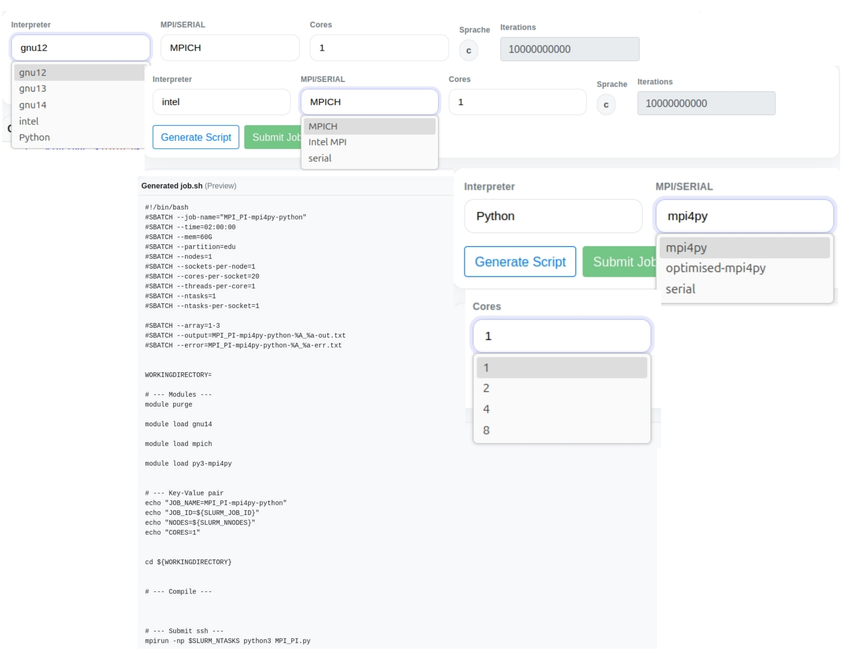

Abbildung 4: UI und Skript-Vorschau der App

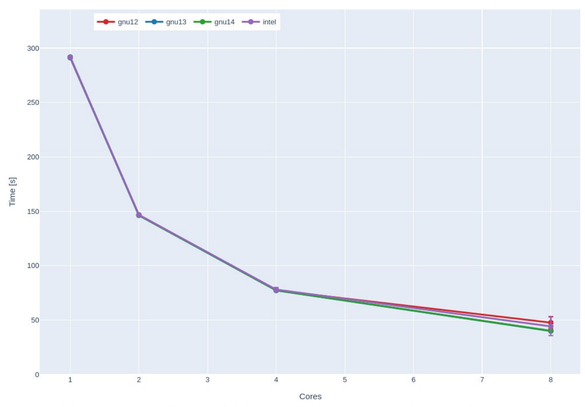

Abbildung 5: Compiler Vergleich verschiedener MPI Implementationen

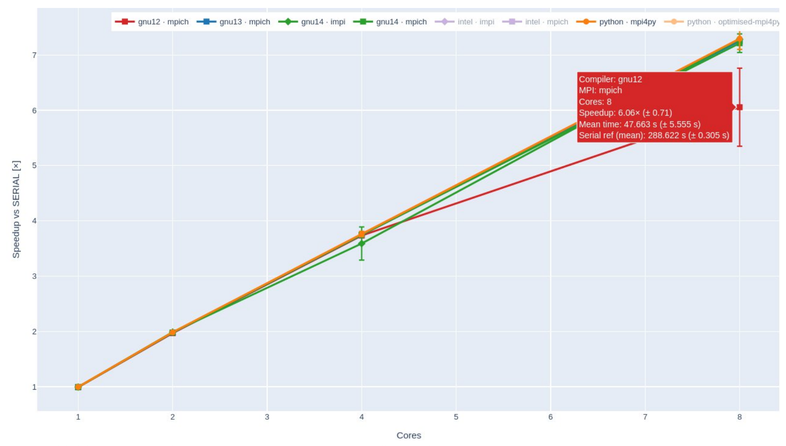

Abbildung 6: Speedup Diagramm zur eigenen Seriellen-Referenz

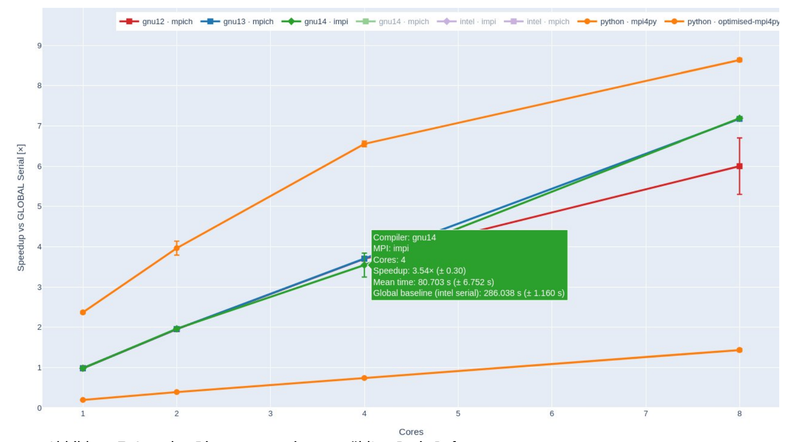

Abbildung 7: Speedup Diagramm zu einer gewählten Basis-Referenz

Im Modul „High Performance Computing“ besteht die Herausforderung häufig darin, Studierenden praxisnahe Einblicke in die Themen Parallelisierung, Compiler-Optimierung und den Einsatz unterschiedlicher MPI-Implementierungen zu vermitteln. Der direkte Umgang mit der komplexen Clusterumgebung ist jedoch oft mit hohen Einstiegshürden verbunden.

Zur Vereinfachung des Zugangs wurde eine Webanwendung auf Basis des Python-Frameworks Flask entwickelt, die es ermöglicht, HPC-Jobs über eine Benutzeroberfläche zu erstellen, auszuführen und auszuwerten. Das Ziel besteht darin, Lehrveranstaltungen durch eine interaktive und visuell unterstützte Umgebung zu ergänzen, in der Studierende verschiedene Job-Konfigurationen experimentell vergleichen können.

Die Anwendung ist als sogenannte Passenger-App in die bestehende Open OnDemand (OOD, HPC-Webportal) Umgebung integriert und bildet typische Aufgaben wie die Einreichung von Jobs und die Ergebnisvisualisierung ab.

Als Beispiel dient die Monte-Carlo-Berechnung von π, die sich leicht parallelisieren lässt und gut zur Demonstration von Performance-Unterschieden eignet.

Über die Weboberfläche können Studierende verschiedene Kombinationen aus Compilern (z. B. GCC, Intel) und MPI-Implementierungen (z. B. MPICH, IntelMPI) auswählen sowie die Anzahl der Rechenkerne (Cores) festlegen, die für die Berechnung verwendet werden.

Vor dem Absenden des Jobs wird automatisch ein Batch-Skript generiert und eine Vorschau erzeugt, sodass der Aufbau und die verwendeten Parameter nachvollzogen werden können.

Nach Abschluss der Berechnungen werden die Ergebnisse verarbeitet und in der Weboberfläche grafisch dargestellt. Zur Analyse stehen verschiedene Visualisierungen zur Verfügung, wie beispielsweise ein Speedup-Diagramm zu einer gewählten Referenz (siehe Abbildung 7), Laufzeitdiagramme unterschiedlicher Compiler/MPI-Implementierungen (siehe Abbildung 5) oder ein Speedup-Diagramm aller Rechnungen mit jeweils eigener Referenz (siehe Abbildung 6).

Alle Diagramme enthalten Fehlerbalken, die die statistischen Schwankungen zwischen mehrfachen Jobausführungen abbilden. Dadurch ist eine aussagekräftigere Bewertung der Messergebnisse möglich.

Damit bietet die Anwendung eine praxisorientierte Lernumgebung, in der Studierende eigenständig HPC-Konzepte testen, Ressourcenparameter einstellen und verschiedene Diagramme analysieren können.

Computational Science Lab

Center for High Performance Simulation Studies

Bielefelder Institut für Angewandte Materialforschung der HSBI

Hochschule Bielefeld
University of Applied Sciences and Arts