Automatic Characterization of HPC Job Parallel Filesystem I/O Patterns

Khorshid Biria

Seminar Supercomputer: Forschung und Innovation

Arbeitsbereich Wissenschaftliches Rechnen

Fachbereich Informatik

Fakultät für Mathematik, Informatik und Naturwissenschaften

Universität Hamburg

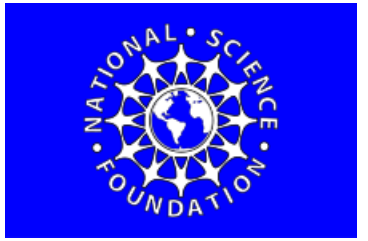
07.12.2021

Gliederung

1. Einleitung
2. Erläuterung über HPC und I/O-System
3. Ähnliche Projekte
4. Vorstellung des Systems „RUSH“
5. Vorstellung der Algorithmen für I/O-Mustererkennung
6. Implementation der Algorithmen
7. Überprüfung auf Richtigkeit der Algorithmen und die Ergebnis davon
8. Hauptergebnisse und die finale Klassifikationen
9. Vorstellung unerwarteter Ergebnisse
10. Zusammenfassung
11. Vorschläge für die zukünftigen Studien
12. Literatur

Einleitung

- XMS Projekt (XD Metrics Service)
  - Verbesserung der betrieblichen Effizienz und des Managements des XD-Netzwerks der NSF (National Science Foundation)



- In diesem Paper: Entwurf von Algorithmen zur Mustererkennung für HPC-Job-I/O
- Implementation von Algorithmen im Open XDMoD
- zur Verbesserung des HPC-Durchsatzes und deren Effizienz

Was ist ein HPC?

- HPC (High Performance Computing)
- "High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business."
- Computercluster vs. Nodes
- Der Sinn eines Hochleistungscomputers

Was ist I/O?

- I/O (Input/Output)
- Jede Eingabe führt zu einer Ausgabe.
  - Eingabe- und Ausgabegeräte von einem Computer
  - Im Allgemeinen: Lese- oder Schreibanforderung an einem Speicher
- I/O Knoten (nodes) → die Schnittstelle zwischen dem Cluster und der Außenwelt

→ Worauf ein vollwertiges Betriebssystem wie Windows Server, Linux läuft

Ähnliche Projekte

- Die Studie von Buneci und Reed
- Datenerfassungsstrategien für HPC-Job-I/O
  1. Analyse der Anwendungssoftware
  2. Analyse von Daten aus Dateisystem I/O-Knoten
  3. Analyse der Daten aus den Rechenknoten
- Unterschied zwischen I/O-nodes und compute nodes

RUSH

- Rush → Center for Computational Research's (CCR) academic HPC resource
- mit etwa 700 x86_64-Rechenknoten
- Vernetzte Knoten
- Rush hat zwei globale Dateisysteme:
  - 3PB IBM GPFS [23] high-performance parallel filesystem
  - Isilon IQ36000x Storage Arrays
- Slurm als Ressourcenmanager

Algorithmen für I/O-Mustererkennung

- Analyse von mehreren Aufträgen, die das GPFS-Dateisystem verwenden
- Suche nach Zeitreihen der I/O
- Ergebnis ⟶ begrenzte Anzahl an gängiger I/O-Muster:
  1. Haupt-I/O-Nutzung zu Beginn des Auftrags
  2. Haupt-I/O-Nutzung zum Ende des Auftrags
  3. I/O-Aktivität zu Beginn und am Ende, aber nicht während des Auftrags
  4. Geringe I/O-Aktivität zu Beginn oder am Ende, aber hohe I/O-Aktivität in der Mitte des Auftrags
  5. Aufträge mit annähernd gleichmäßiger Aktivität während der gesamten Dauer des Auftrags
  6. regelmäßige periodische I/O-Aktivität

Simple Job classifier-Algorithmus

- Algorithmus für I/O-Mustererkennung als Plugin für SUPReMM Software
- Ein Klassifizierungsalgorithmus, der die Aufträge anhand eines sehr groben Maßes kategorisierte, wann der Großteil der I/O stattfand:

Algorithm 1 Simple job classification

```
if s0 > s1 + s2 + s3 then
    if s3 > 2(s1 + s2) then
        a ← CANYON
    else
        a ← START
    end if
else if s3 > s0 + s1 + s2 then
    if s0 > 2(s1 + s2) then
        a ← CANYON
    else
        a ← END
    end if
else if s1 + s2 > 2(s0 + s3) then
    a ← HILL
else if min(s0, s3) > 2 max(s1, s2) then
    a ← CANYON
else
    a ← OTHER
end if
```

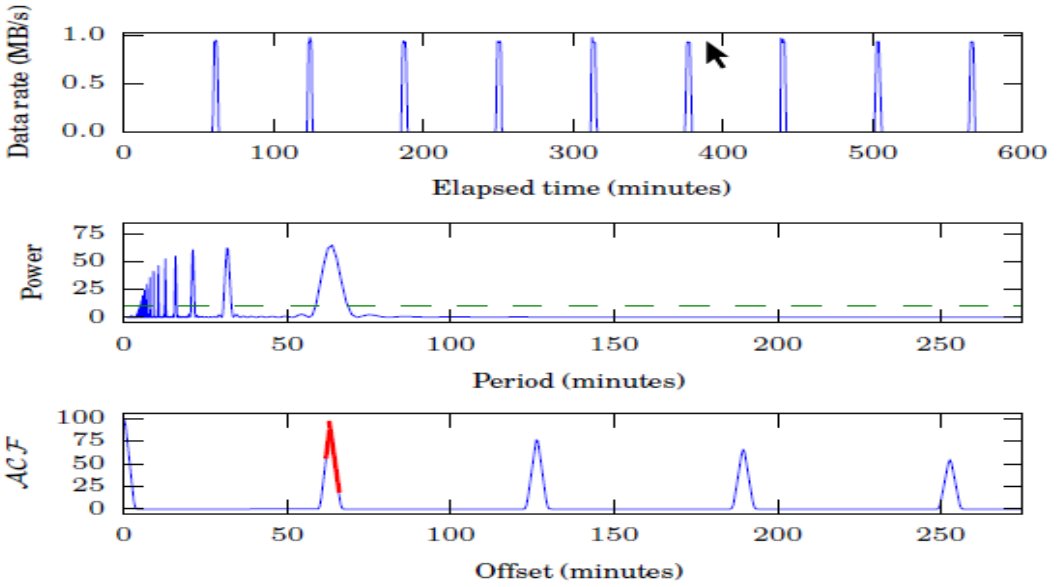
Periodicity Detection-Algorithmus

- Für die Suche nach periodischem I/O-Verhalten durch Umsetzung von AUTOPERIOD Algorithmus
- Ein zweistufiger Ansatz, der die Informationen sowohl im Periodogramm als auch in der autocorrelation function (ACF) berücksichtigt

Grafische Darstellung

Periodicity Detection-Algorithmus

- Arbeitsweise:
  1. Das Periodogramm der Daten nach der Lomb-Scargle Methode berechnen
  2. Die statistisch signifikanten Zeiträume bestimmen. Wenn es keine gibt, ist der Job nicht periodisch.
  3. AutoCorrelation Function (ACF) berechnen
  4. Von den Kandidatenperioden mit signifikanter Spektralleistung im Periodogramm prüfen, ob die Periode auf einem Hügel oder Tal der ACF liegt
  5. Den Kandidatenzeitraum mit der höchsten Leistung wählen, der sich auf einem Hügel des ACF liegt. Dies ist die Periode der Stelle. Wenn es keine Spitzen auf einem Hügel der ACF gibt, ist der Auftrag nicht periodisch.



Überprüfung auf Richtigkeit

- Durch Verwendung äquivalenter Daten aus dem Google-Trends-Dienst, um die Ergebnisse aus dem Originalpaper zu reproduzieren
  - → Einigermaßen korrekter Algorithmus
  - → Während der ersten Tests: einige Fälle gefunden, in denen der Algorithmus ein periodisches Verhalten identifizierte, die I/O-Daten jedoch nicht periodisch zu sein schienen oder zwar periodisch waren, aber eine andere Periode als die ermittelte Hauptperiode aufwiesen
- Lösung: eine einfache webbasierte Anwendung als manuellen Ansatz, der eine schnelle manuelle Überprüfung ermöglicht
- Gleiche Idee für den ersten Algorithmus

Hauptergebnisse und die finale Klassifikationen

- Die hier vorgestellten Ergebnisse stammen von GPFS-Dateisystemdaten, die auf der akademischen HPC-Ressource Rush im CCR gesammelt wurden
- Analyse für eine Untergruppe von Aufträgen
- Ausgeschlossene Aufträge:
  1. Shared-Node-Jobs
  2. Aufträge kürzer als zehn Minuten

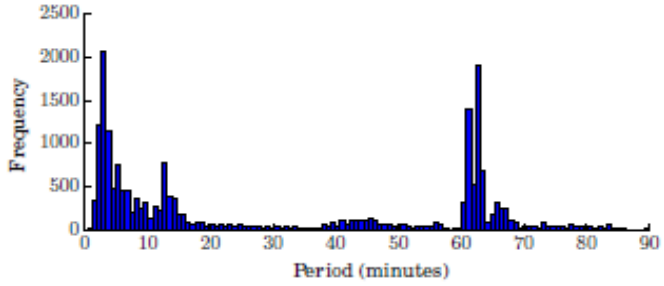
Simple classifier Algorithmus

Hauptergebnisse und die finale Klassifikationen

- Die Ergebnisse von dem ersten Algorithmus:

read \ write	HILL	START	END	~UNIFORM	OTHER	CANYON	NO USAGE
HILL	1166	652	170	30	131	116	138
START	218	4829	2053	7694	2010	2642	2356
END	3366	4395	4263	518	8899	1840	397
~UNIFORM	106	621	521	9554	1336	241	545
OTHER	240	837	713	2874	9731	833	112
CANYON	1389	730	2208	1130	2793	11397	847
NO USAGE	1468	775	32885	7077	2234	2824	492980

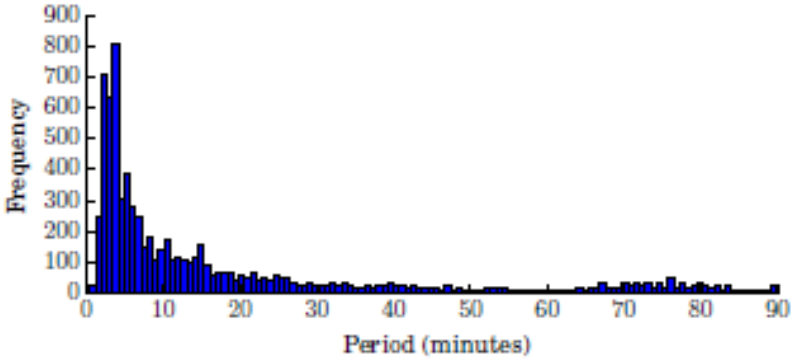
- Die am häufigsten vorkommenden Kategorien: NO USAGE Kategorie, OTHER und ~UNIFORM
- Diagonale Einträge
- Aufträge mit der Schreibklassifizierung ~UNIFORM oder OTHER von dem Simple Classifier, die außerdem als periodisch bezeichnet waren:



Simple classifier Algorithmus

Hauptergebnisse und die finale Klassifikationen

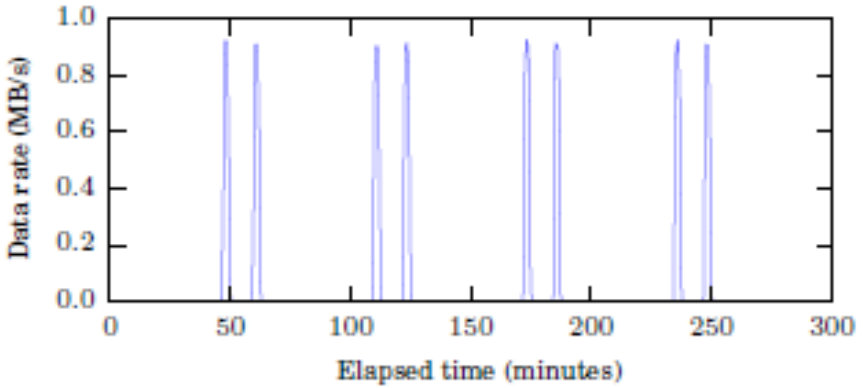
- Aufträge mit gelesener periodischer Klassifizierung und ~UNIFORM oder OTHER vom Simple Classifier:



- Sehr unterschiedlich zu dem write-Histogram
- Viele komplexe I/O-Muster können aber von dem einfachen heuristischen Klassifikator nicht gut erfasst werden
- Schwierigkeiten bei der manuellen Überprüfung der Ergebnisse: es gab einige Fälle, in denen die periodische Erkennung zu falschen oder unerwarteten Ergebnissen führte

Unerwartete Ergebnisse

- Ein Beispielauftrag mit periodischem Schreibverhalten mit zwei Hauptfrequenzen:



Zusammenfassung

- zwei komplementäre Algorithmen zur Klassifizierung von I/O-Daten von HPC-Jobs:

1. einen einfachen heuristischen Klassifikator, der auf einfachen Annahmen über typische I/O-Muster in HPC-Software beruht
2. einen Klassifikator, der periodische Muster erkennt und ihre Periode bestimmt

- Das einfache heuristische Modell funktionierte gut, aber es gab eine beträchtliche Anzahl von Stellen, die nicht in die einfachen Kategorien passten
- Motivation: die Kategorisierung von Aufträgen, um die Erkennung von Auftragsanomalien zu unterstützen.

Vorschläge für die zukünftigen Studien

- Das Plugin speichert für die Periodizitätserkennung alle I/O-Daten im Speicher.
- Das Ziel ist den Speicherbedarf für die Zusammenfassung der Periodizitätserkennung zu reduzieren

Literatur

- https://insidehpc.com/hpc-basic-training/what-is-hpc/
- https://www.dummies.com/computers/computer-networking/networking-components/explaining-io/
- https://dl.acm.org/doi/10.5555/1973333.1973349
- https://thomastechllc.com/eol_resource/isilon-iq36000x/
- https://slurm.schedmd.com/overview.html
- https://github.com/ubccr/supremm
- https://ieeexplore.ieee.org/document/7106398
- https://en.wikipedia.org/wiki/Scratch_space