Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Pream: Enhancing HPC Storage System Performance with Pre-allocated Metadata Management Mechanism

Jiafan Gao
30.11.2021

Arbeitsbereich Wissenschaftliches Rechnen
Fachbereich Informatik
Fakultät für Mathematik, Informatik und Naturwissenschaften
Universität Hamburg

Gliederung

- Einführung in Hochleistungsrechnen
  - Funktion und Architektur
  - Nutzung von parallelen Dateisystemen
  - Probleme und mögliche Ansätze
- Pream
  - Aufbau
  - Funktionsweise der Komponenten
  - Ablauf
- Leistungsvergleich
- Zusammenfassung

Einführung Hochleistungsrechnen

- Computergestütztes Rechnen
- Bedarf an hoher Rechenleistung oder Speicherkapazität
- Parallelisierung von Rechenaufgaben
- Clusterbildung
- Einsatz in diversen Forschungsgebieten

Aufbau traditioneller HLR-Architektur

- Rechenknoten
  - Ausführung von Aufgaben
- Verbindungsnetz für die Verbindung zu Ressourcen
- Paralleles Dateisystem für effiziente Datenspeicherung
  - Verteilt Daten über mehrere Server (Striping)
- Metadaten Server
  - Suchoptimierung

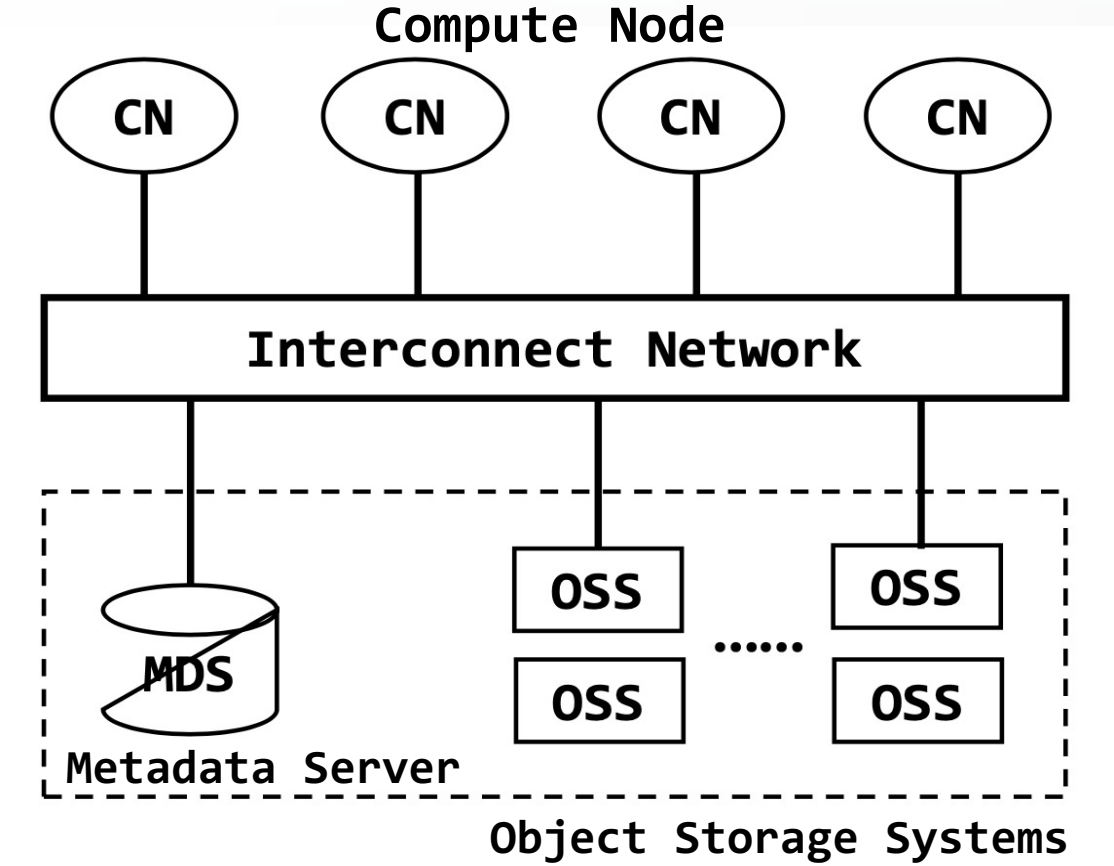


Fig. 1: A typical HPC system architecture

Bildquelle: siehe Literatur [1]

Problematik

- Striping weniger effizient für kleine Daten
- Entkopplung von Rechenknoten & Ressourcen
  - Limitierte Speichermöglichkeiten auf Rechenknoten
- Verwaltung der Daten durch parallele Dateisysteme
  - Streitfrage bei gleichzeitigen Ein- und Ausgabe Anfragen
  - Teure Metadaten Operationen
  - Beeinflussung der Leistung

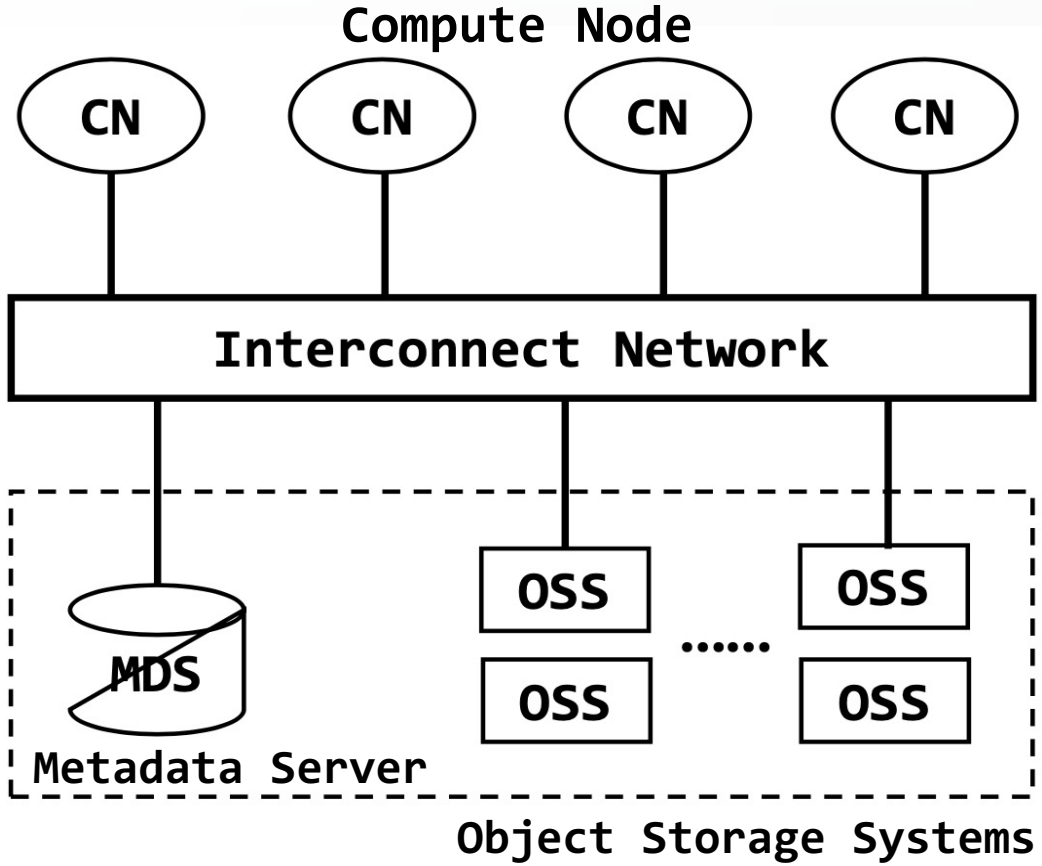


Fig. 1: A typical HPC system architecture

Bildquelle: siehe Literatur [1]

Suboptimale Ansätze im parallelen Dateisystem

- Zwischenspeicherschicht (BurstBuffer)
  - Zwischen Rechenknoten und Dateisystem
  - Mehr Netzwerkaufwand für Rechenknoten
  - Nicht hilfreich für kleine Daten
- Erstellung von vorübergehenden Dokumenten
  - Herausforderung für Zugang zu Metadaten in den Systemen
- SSD auf dem Rechenknoten
  - Aufwändige Konfiguration der Hardware

Lösungsvorschlag Pream

- Nutzung von Vorabzuweisung und Proxy Server
- Reduzierung der Verzögerung von Metadaten Operationen
- Verbesserung der Leistung der Dateisystemen

- Einfache Integration
  - In parallelen Dateisystemen
  - In anderen verteilten Dateisystemen als Zwischenspeicher (Cache Schicht)

Pream – Bestandteile und Aufbau

- Client
- Proxy Server
  - Vermittler zwischen Client und Servern
- Objekt Speicher
  - Paralleles Dateisystem
- Metadaten Server

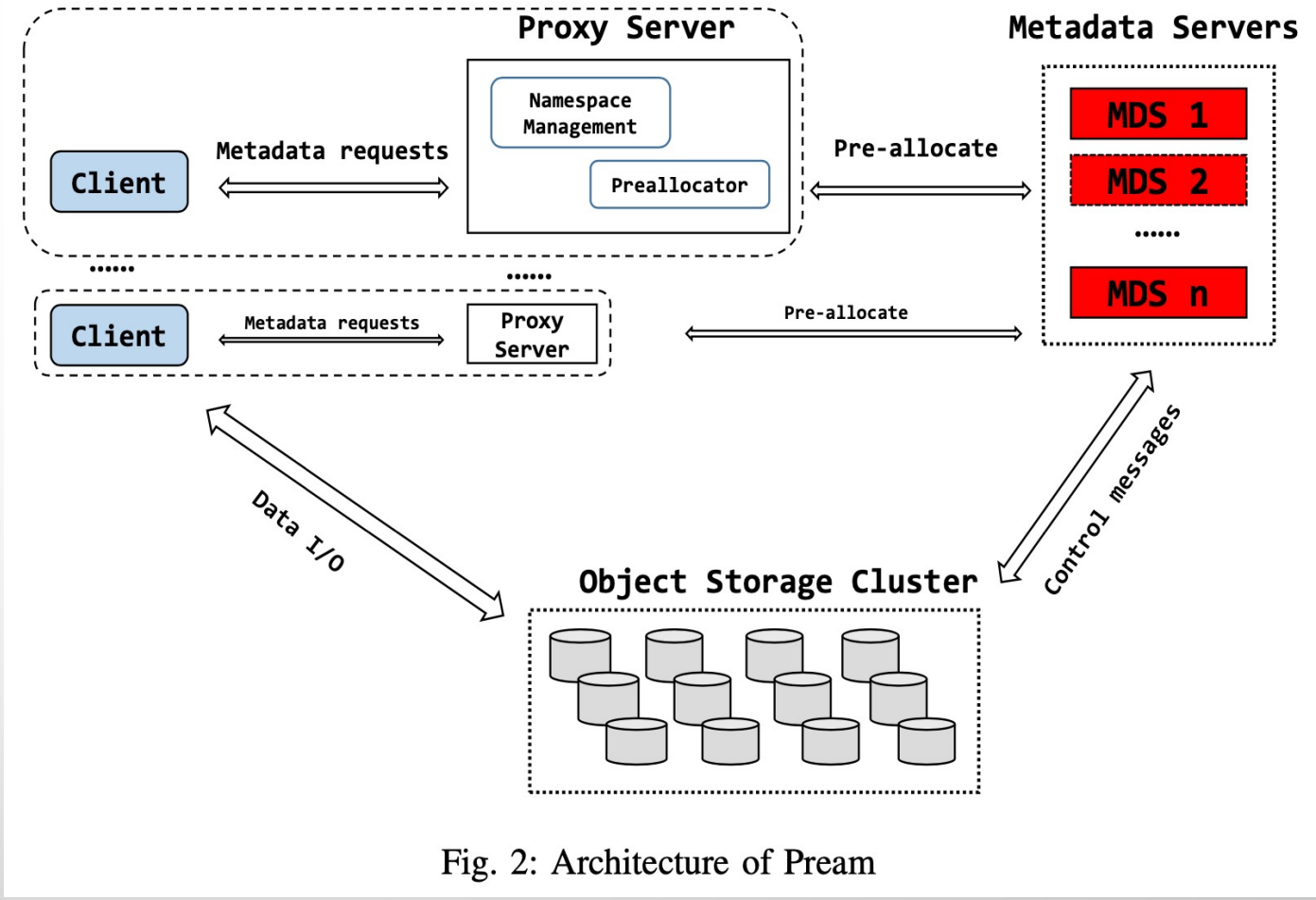


Fig. 2: Architecture of Pream

Bildquelle: siehe Literatur [1]

Pream – Bestandteile und Aufbau

- Ein- und Ausgabe über Proxy Server
  - Speicherservice
- Mechanismus für Vorabzuweisung von Metadaten
  - ➢ ”Preallocator”
- Abbildung der Namensräume
  - ➢ “Namespace Management”
- Verwaltung der Metadaten

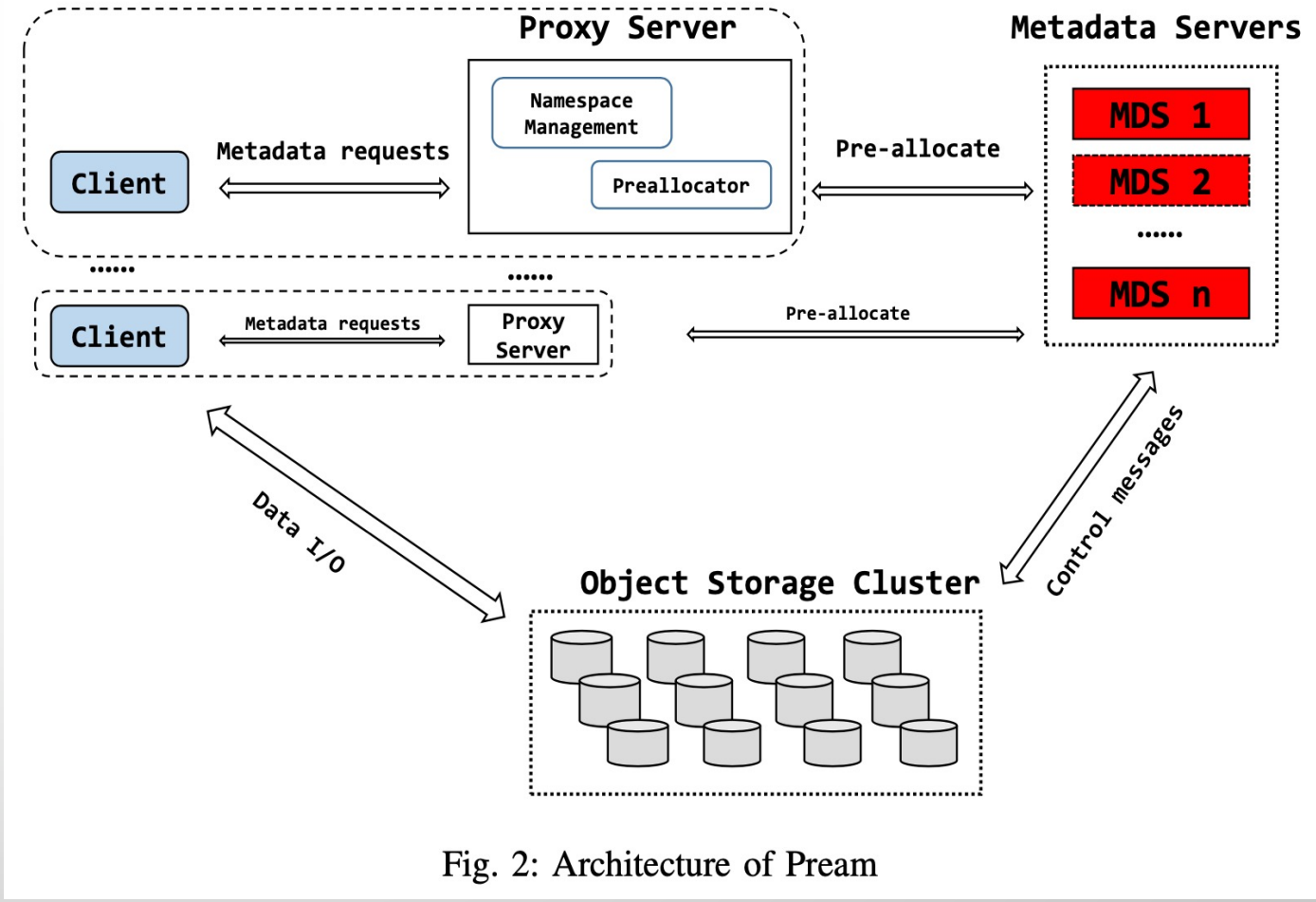
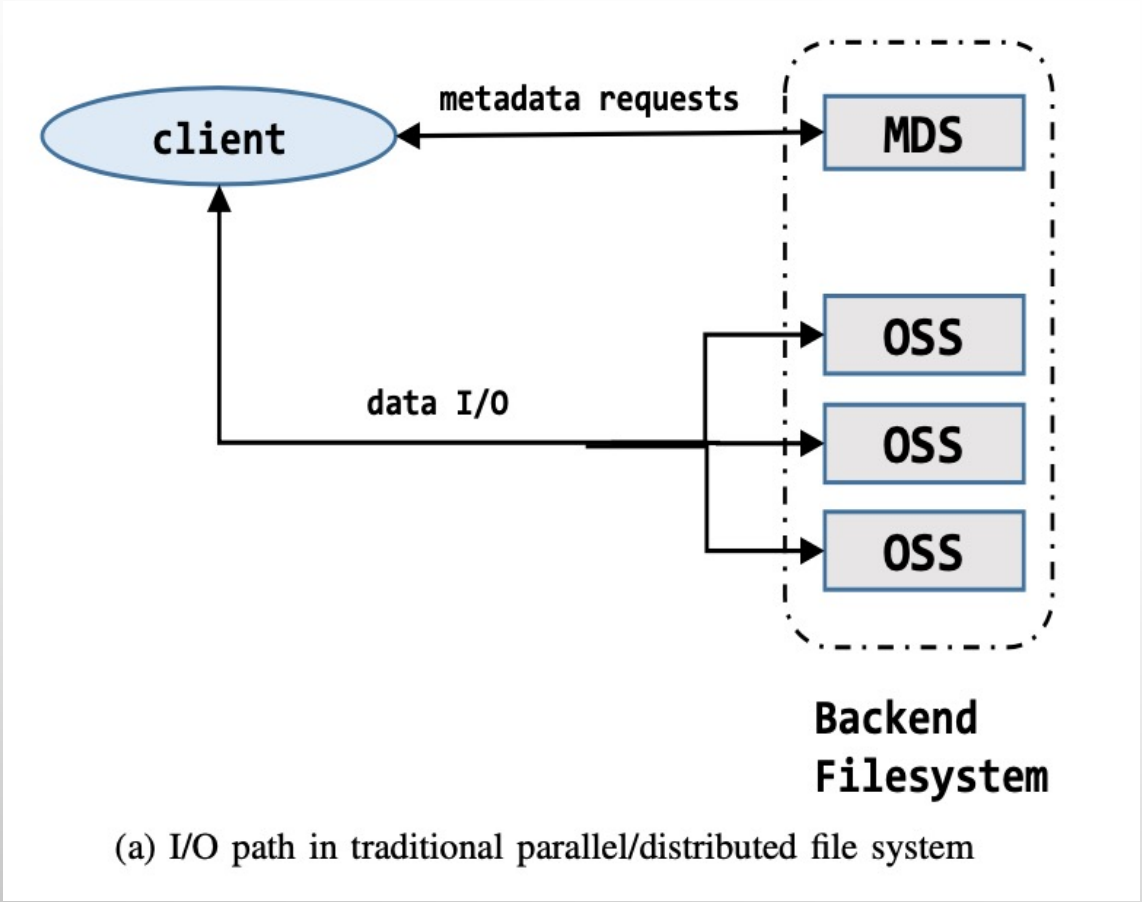


Fig. 2: Architecture of Pream

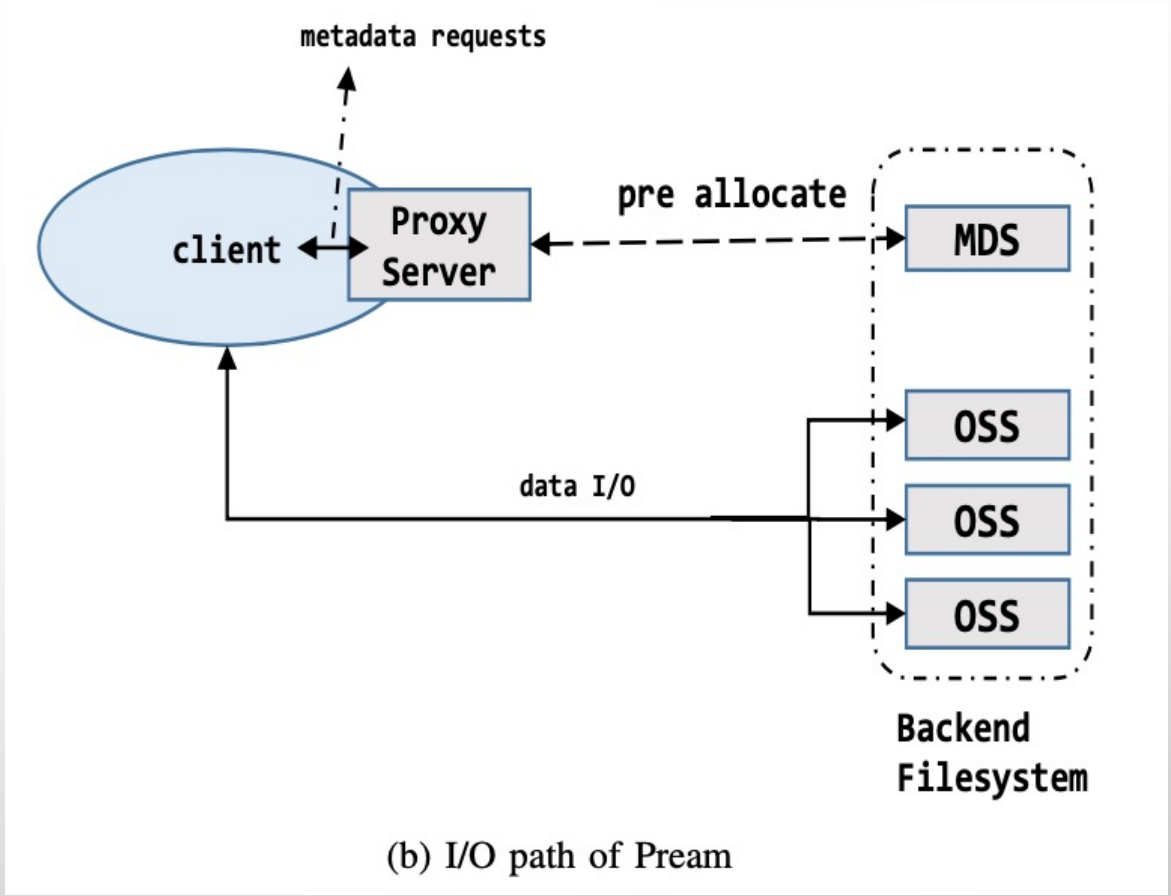
Bildquelle: siehe Literatur [1]

Unterschied Eingabe/Ausgabe Pfad



(a) I/O path in traditional parallel/distributed file system

Bildquelle: siehe Literatur [1]

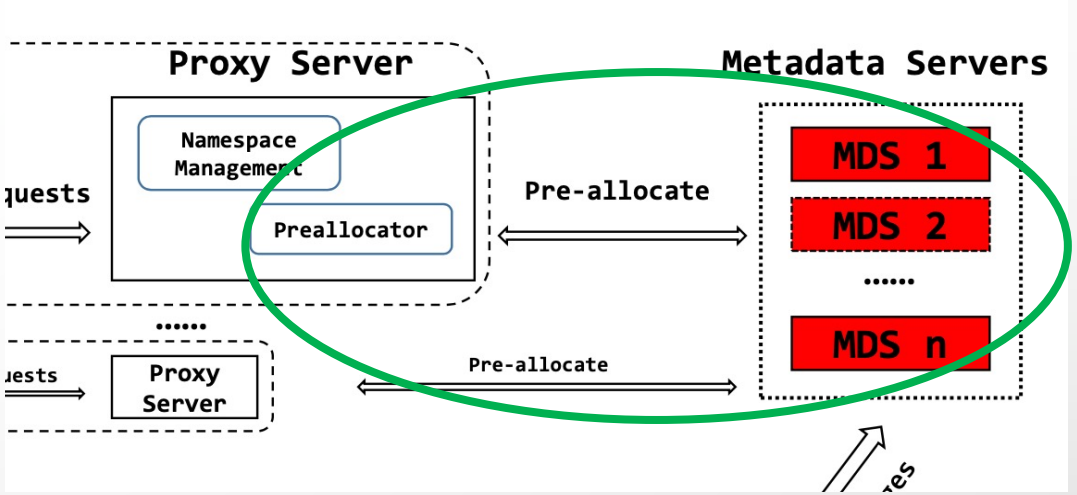


(b) I/O path of Pream

Bildquelle: siehe Literatur [1]

Pream – Vorabzuweisung mit Preallocator

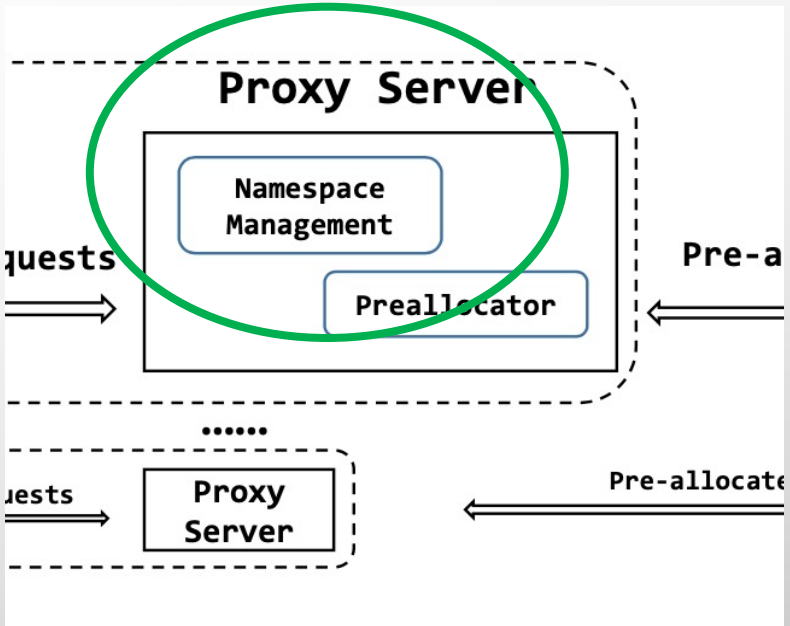
- Ein Thread im Hintergrund
- Allokiert Metadaten Speicher bereits vor der Anfrage vom Client
- Metadaten nur für Erstellen von Dateien
- Sendet einen Stapel von Anfragen an Vorab-zuweisung für Dateien
- Anzahl der Anfragen abhängig von Nutzlast der Clients
- Vordefinierter Schwellenwert für ungenutzte Metadaten
  - Preallocator wieder aktiv



Bildquelle: siehe Literatur [1]

Pream – Namespace Management im Proxy Server

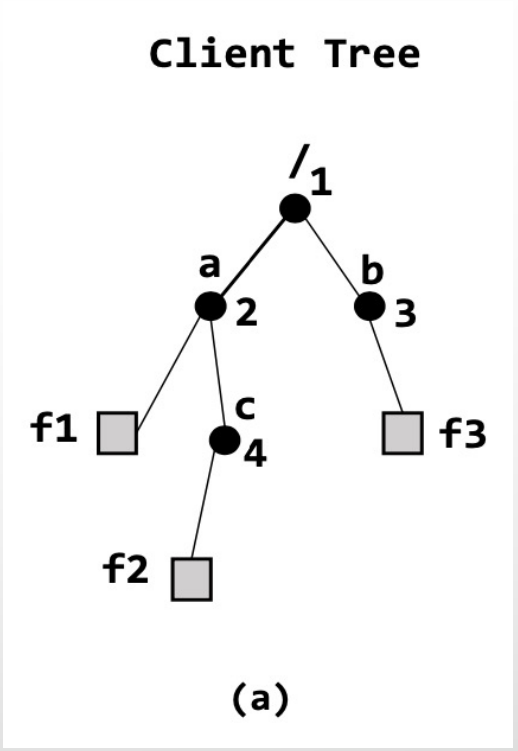
- Erstellen der Metadaten Informationen für Verzeichnisse
- Abbildung Namensraum des Client und der Vorabzuweisung
  - Verwaltung der Verzeichnisse
- Speicherung der Metadaten als Listen
  - Ungenutzte Liste
  - Schmutzige Liste
- Zusammenfassung anhand einer Zurodnungstabelle



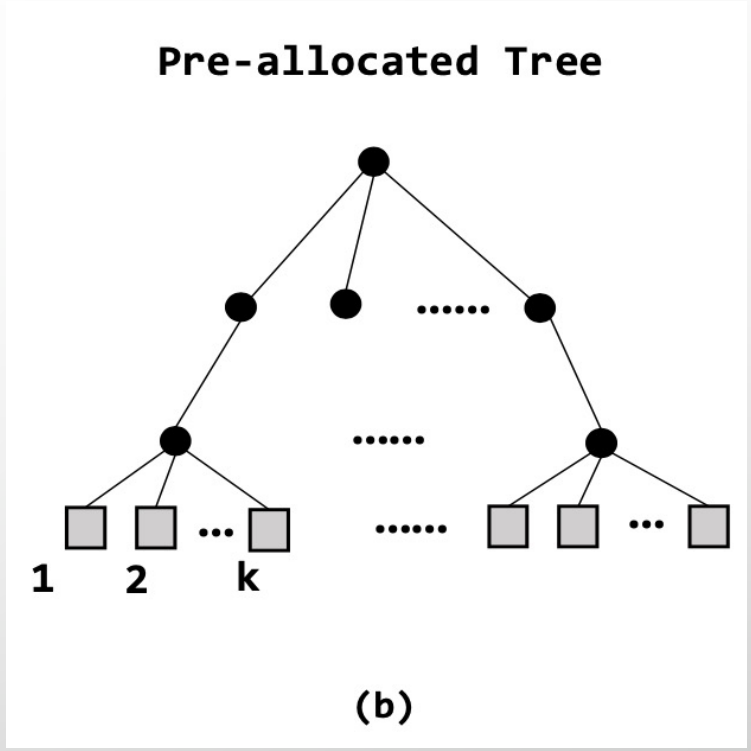
Bildquelle: siehe Literatur [1]

Abbildung des Namensraums

- Punkt: Pfad, Quadrat: Datei
- Namensraum des Clients (a)
  - i.d.R. unbekannt
- Namensraum im Vorfeld definieren
- In (b) zweistufiges Verzeichnis
  - Jede Ebene 128 Unterverzeichnis
  - Nutzen von String hash
  - Verhindert große Verzeichnisse
  - Dokumente {1,…,k}



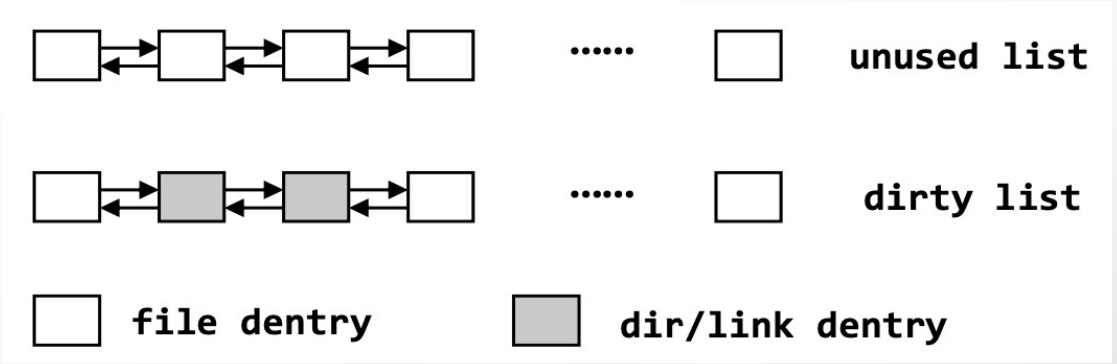
Bildquelle: siehe Literatur [1]



Bildquelle: siehe Literatur [1]

Metadaten Management – Speichern in Listen

- Metadaten für Dateien als Listen gespeichert
- “Dentry” Struktur repräsentiert Metadaten
  - Verzeichniseintrag
  - *fid*, `struct` *stat* etc.
- “unused list” für ungenutze Metadaten
  - Preallocator wieder aktiv, wenn zu wenig ungenutzte Metadaten vorhanden
- “dirty list” für genutzte Metadaten und Pfade
  - Nach Zuordnung in “dirty list” geschoben
- Zuordnung durch eine Tabelle festgehalten



Bildquelle: siehe Literatur [1]

Metadaten Management – Zuordnungstabelle

- Speicherung von Zuordnungen in “MapTable”
- Key/Value Paare in der Tabelle
- Key:
  - Indexnummer vom Elternknoten + Dateiname
- Value:
  - “Dentry” Struktur
  - *fid* 0 repräsentiert Pfadinformation

MapTable

key	value
<0,/>	0,struct stat,…
<1,a>	0,struct stat,…
<1,b>	0,struct stat,…
<2,c>	0,struct stat,…
<2,f1>	1,struct stat,…
<4,f2>	2,struct stat,…
<3,f3>	3,struct stat,…

Bildquelle: siehe Literatur [1]

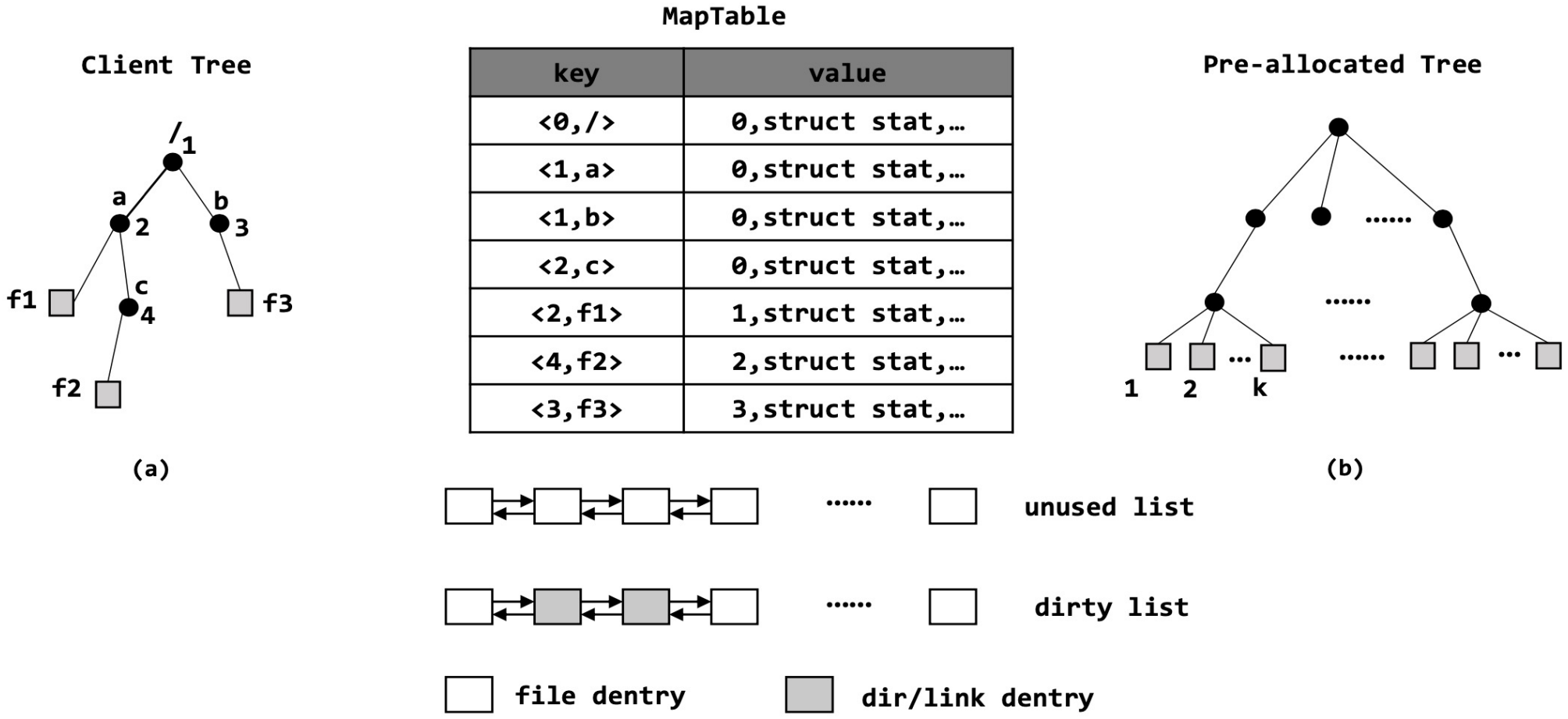
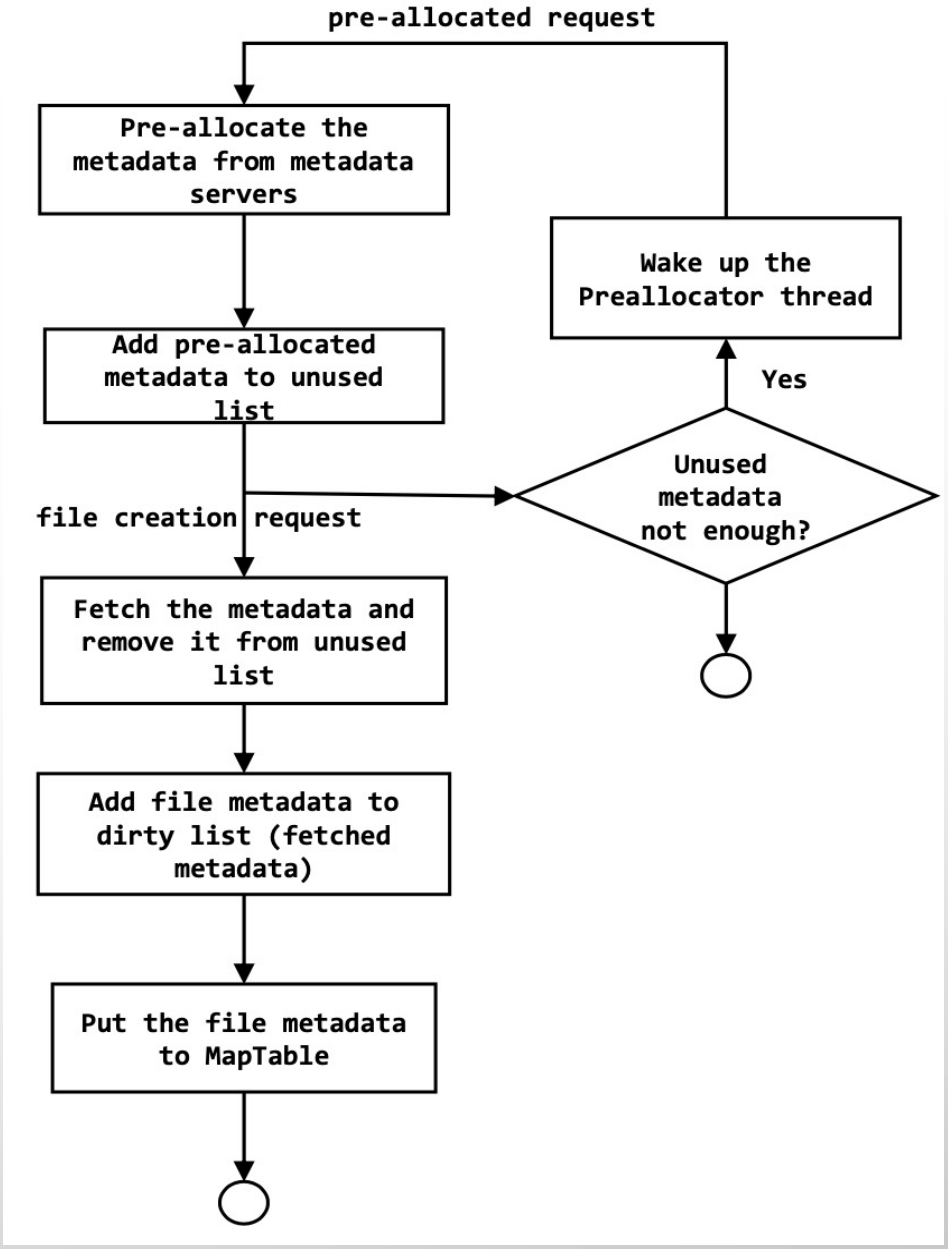


Fig. 4: Namespace management of Pream

Bildquelle: siehe Literatur [1]

Ablauf in Pream

- Anfrage für Vorabzuweisung der Metadaten
- Hinzufügen in die Liste “unused list”
- Anfragen für Erstellung von Dateien
- Von der unbenutzten Liste entfernen
- Hinzufügen in die Liste “dirty list”
- Hinzufügen in die Mapping-Tabelle “MapTable”
- Beim Schwellenwert neue Vorabzuweisungen anfragen



Bildquelle: siehe Literatur [1]

Evaluierung der Leistung

- Vergleich mit Lustre (ein bekanntes paralleles Dateisystem)

- Vergleichstest eines wiederholbaren Programmteils
  - Leistung bei Dateierstellung
- Vergleichstest auf der Serverebene
  - Ladezeit der Server
  - Arbeitsleistung der Server
- Vergleichstest Anwendungsbereich

Vergleich – Dateierstellung

- In Abhängigkeit von Anzahl an Dateien
- Input/Output Operations per Second (IOPS)
  - Je höher, desto besser
- Pream eindeutig effizienter
  - Ruft nur die Struktur in der “ungenutzten Liste” ab
- Lustre
  - Erstellung der Metadaten erst bei Anfrage

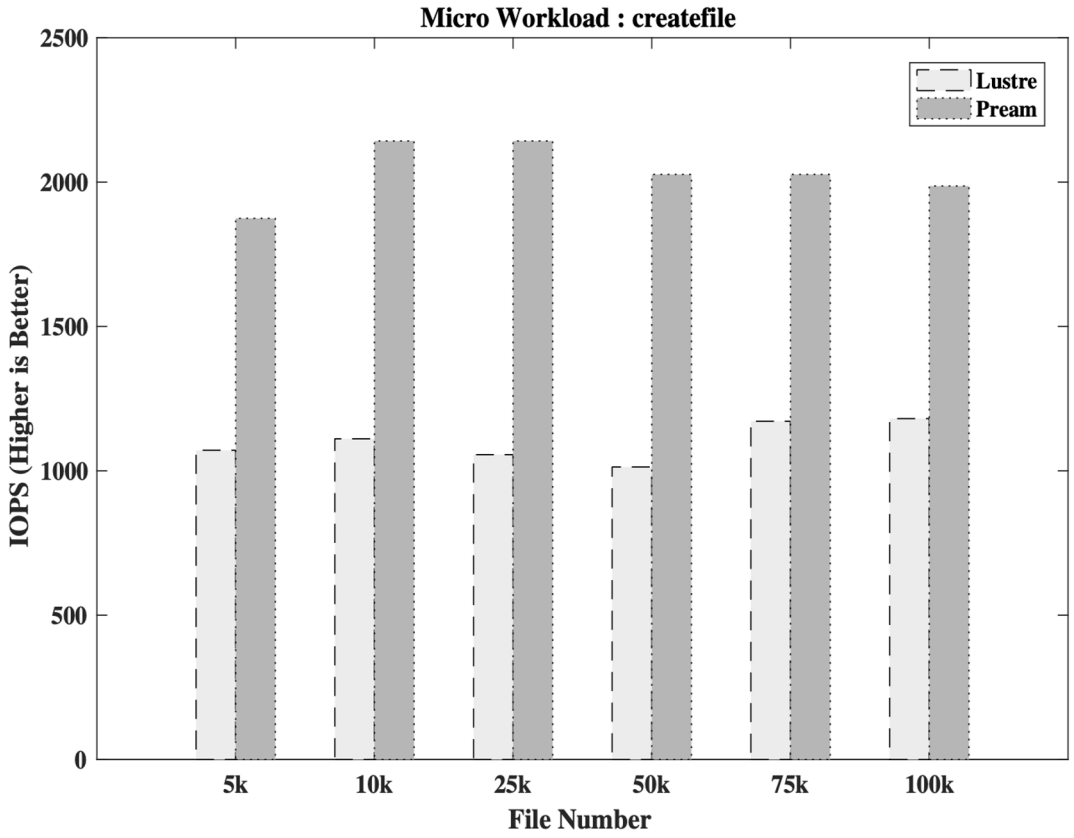


Fig. 6: Performance of file creation

Bildquelle: siehe Literatur [1]

Vergleich – Auslastung der Dateisysteme

- Ladezeit von Dateiserver in realer Umgebung
  - Erstellen/Löschen
  - Schreiben/Lesen
  - Öffnen/Schließen
  - Etc.
- Untersuchung für kleine Dateien
- Sinkende Leistung bei steigender Anzahl an Dateien
- Pream effizienter

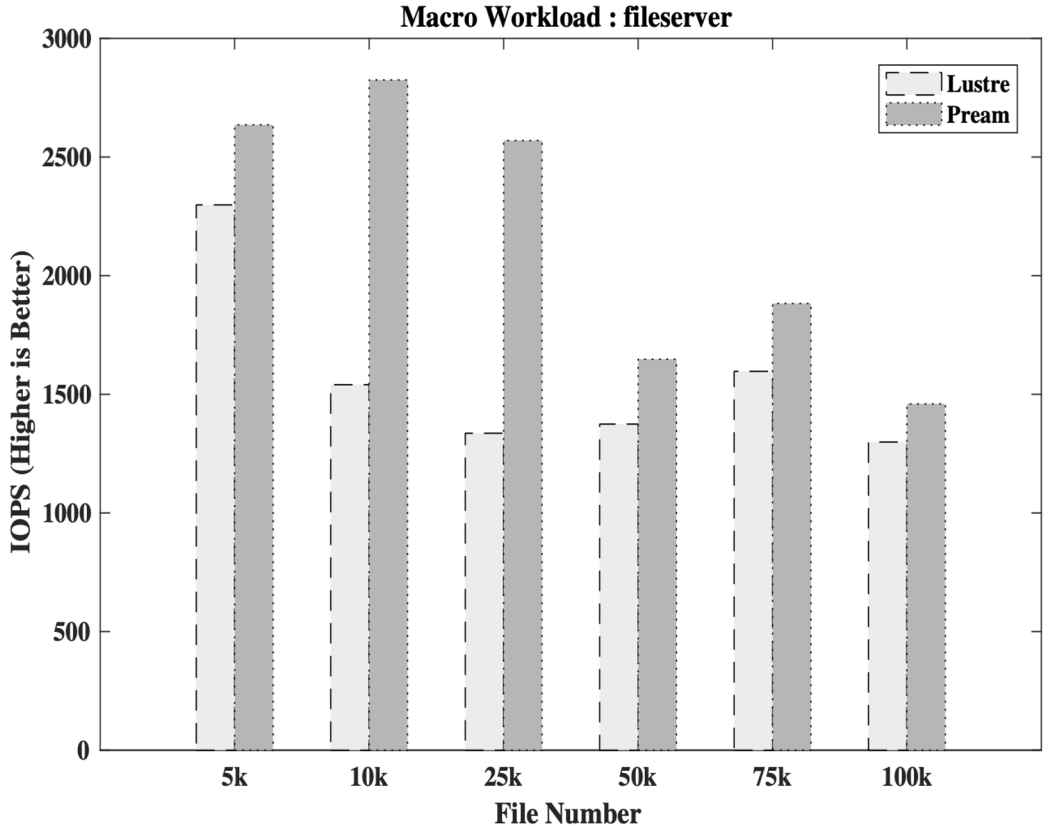


Fig. 7: Performance of fileserver workload

Bildquelle: siehe Literatur [1]

Vergleich – Auslastung der Dateisysteme

- Ladezeit von Webserver in realer Umgebung
  - Lesen
  - Öffnen
  - Schließen
- Pream fast 5-fach effizienter
- Keine Schreiben-Operation
- Daten im Cache, daher keine großen Veränderungen

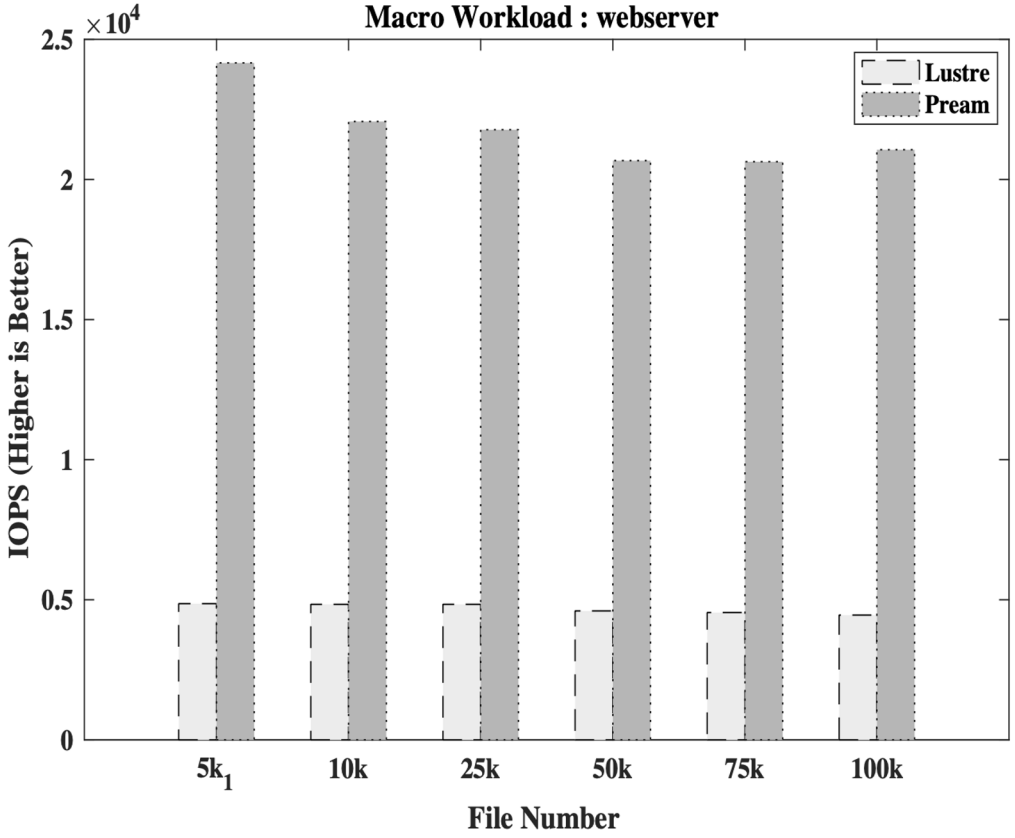


Fig. 8: Performance of webserver workload

Bildquelle: siehe Literatur [1]

Vergleich Auslastung in Anwendung

- DenseIO
  - Kontrolliert die Anzahl von temporären Dateien
- Ausführungszeit in Abhängigkeit von Anzahl der Dateien
- Ergebnis:
  - Steigende Ausführungszeit bei steigender Anzahl an Dateien
  - 5-fache Ausführungszeit bei Lustre bei 1.000 Dateien

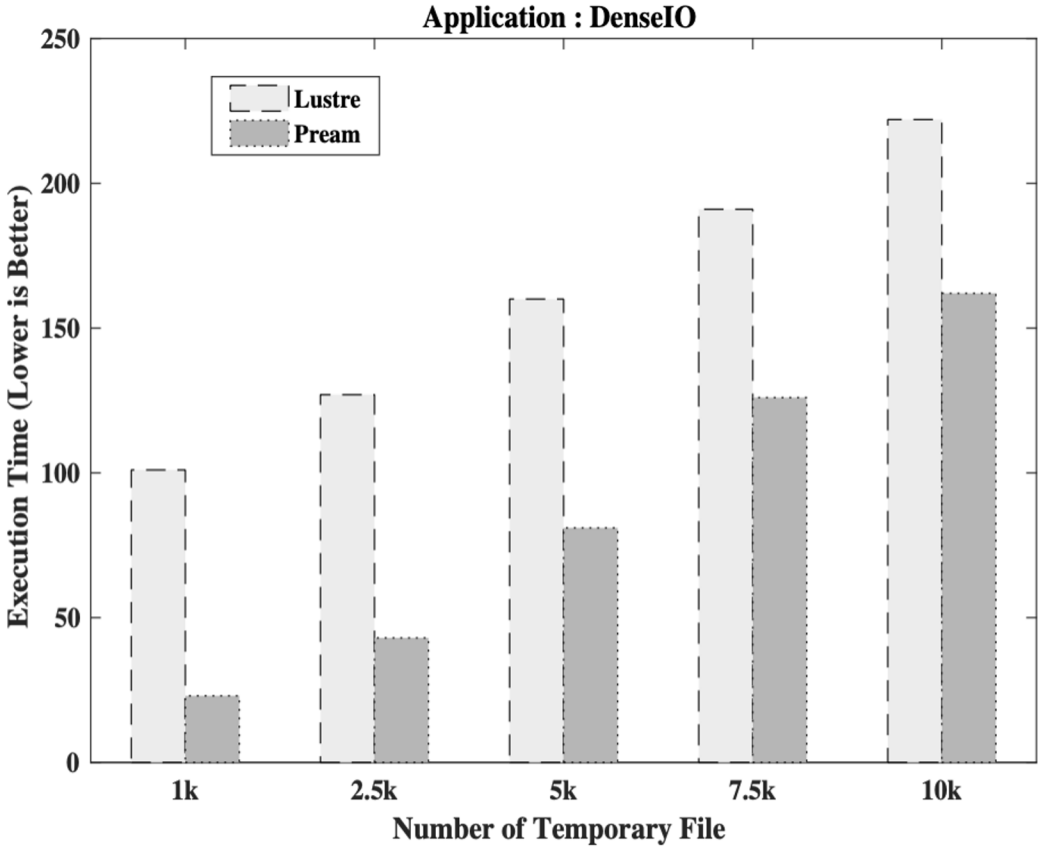


Fig. 9: Performance of DenseIO workload

Bildquelle: siehe Literatur [1]

Zusammenfassung

- Hochleistunsrechnen für aufwändige Berechnungen
- Teure Metadaten als Flaschenhals
- Optimierung der Metadaten für temporäre Dateien
  - Insbesondere bei kleinen Dateien
- Effiziente Metadaten-Speicherung durch Pream
- Vorabzuweisung der Metadaten mithilfe von Proxy Server
- Bearbeitung der Anfragen vom Client lokal im Proxy Server
- Verbesserung der Leistung des Dateisystems
- Reduzierung der Latenzzeit

Literatur

- [1] Pream: Enhancing HPC Storage SystemPerformance with Pre-allocated Metadata Management Mechanism (Li et al.)
- [2] Hochleistungsrechnen Vorlesungsaufzeichnung 2020/21 (Prof. Dr. Thomas Ludwig) besucht am 27.11
- [3] http://books.gigatux.nl/mirror/kerneldevelopment/0672327201/ch12lev1sec7.html (Robert Love) besucht am 22.11
- [4] https://tecadmin.net/what-is-inode-number-in-linux/ (Rahul) besucht am 22.11
- [5] https://www.forschungsdaten.info/praxis-kompakt/glossar/#c269911 besucht am 24.11