in der Cloud

Hintergrund

- Effizienz hat hohe Priorität in der Cloud
  - Ziel: Kosten Minimieren, Leistung Maximieren
- x64-CPUs verbrauchen viel Energie
- ARM Prozessoren bieten mögliche Effizienzsteigerung
  - Reduced Instruction Set CPU (RISC)
  - Weniger Transistoren und explizites Design für Energie-Effizienz
  - Mehr Register für effiziente Verarbeitung

Problemstellung

- Welche Cloud-Provider bieten bereits ARM basierte Services an?
- Welche Services werden auf ARM-Architektur angeboten?
- Welchen Performance pro Kosten / Watt kann man im Vergleich zu x32/x64 CPUs erwarten?
- Welche Einschränkungen gibt es durch z.B. die geringere Anzahl an Instruktionen?

Unterstützung der Provider

Google Cloud

Compute Engine

- 2 Varianten

Kubernetes Engine

Azure

Virtual Machine

- 4+ Varianten

Kubernetes Service

AWS

EC2

- 3 Varianten

Aurora
Batch
CodeBuild / CodeCatalyst
DocumentDB
RDS
ECR / ECS / EKS
Lambda
……

Verwendete Instanzen

Google Cloud

- c4a-standard-4
- 3 GHz [15]
- 0,1725€ / h

- c4-standard-4 [14]
- 3.1 GHz / 4 GHz
- 0,1898€ / h

Azure

- Standard_B4pls_v2
- 3 GHz
- 0,114€ / h

- Standard_B4als_v2
- 3.5 GHz
- 0,128€ / h

AWS

- m7g.xlarge
- 2.7 GHz
- 0,1567€ / h

- m7i.xlarge
- 3.2 GHz
- 0,1936€ / h

PerfKitBenchmarker

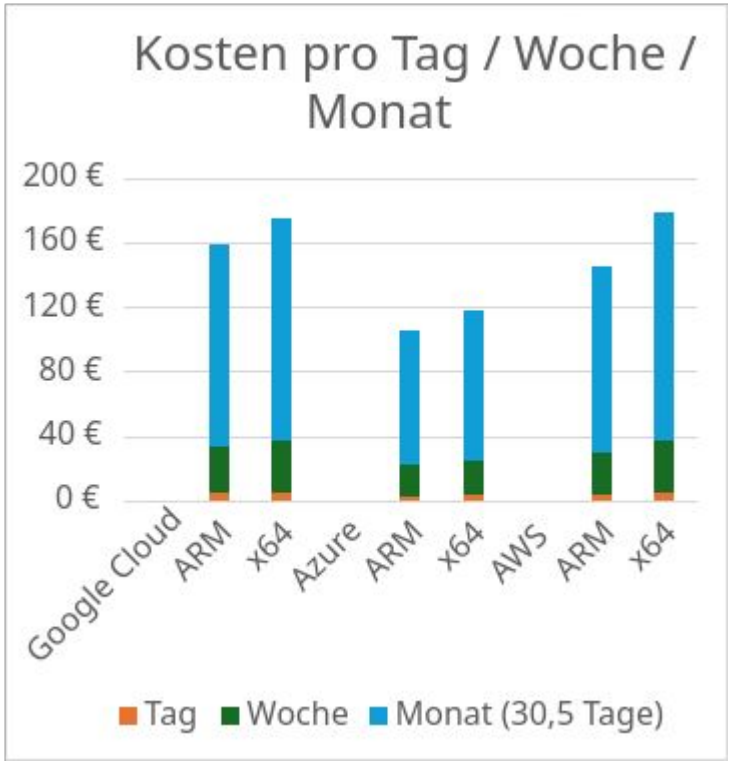
- Cloud Benchmarking Tool von Google
- Unterstützt vielzahl an Providern
  - Google Cloud, Azure, AWS, Alibaba, IBM, Openstack, …..
- Unterstützung für verschieden Arten von Benchmarks
  - CPU, Disk, RAM, Netzwerk, NoSQL / Database Auslastung, Web Traffic, …..
- Automatisiert Provisionierung und Durchführung
  - Aufsetzen und Teardown von Netzwerk, Firewall, VM-Instanzen, SSH
  - Ergebnisse werden an lokales Gerät zurückgesandt

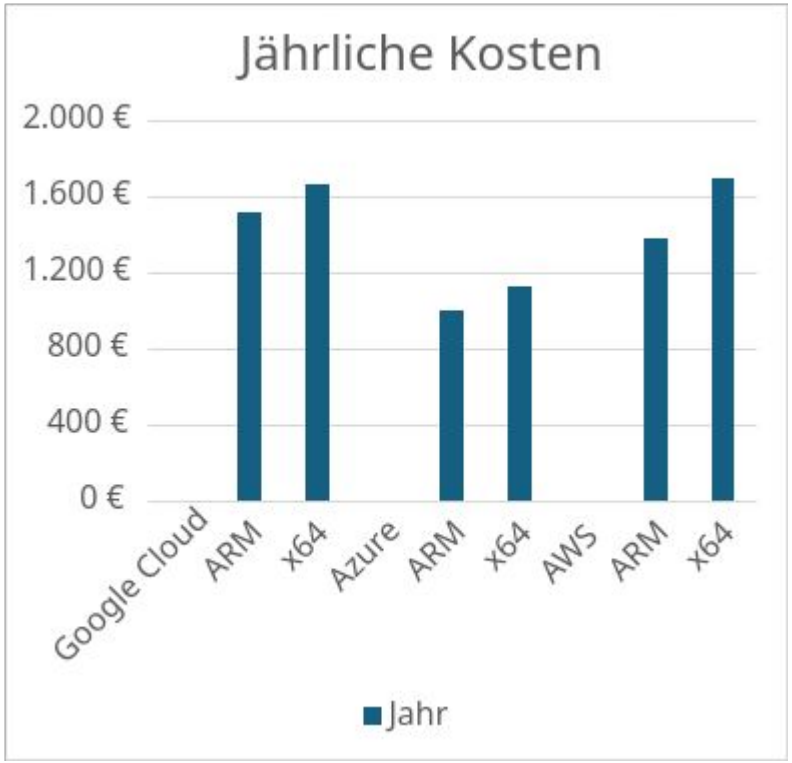
https://github.com/GoogleCloudPlatform/PerfKitBenchmarker

Ausgeführte Benchmarks

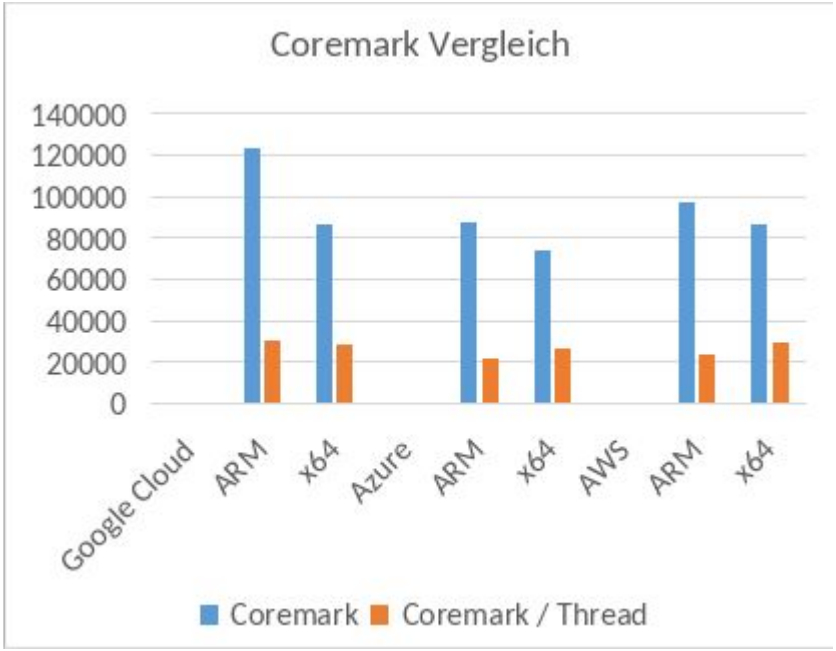
- coremark
  - Ebenfalls von Google für Tests der Instanzen verwendet
  - Verschiedene realistische Operationen für Anwendungen
    - Listen-Operationen, Matrix-Berechnung, State-Machine
  - Programmiert in C
- scimark2
  - Verschiedene Wissenschaftliche Berechnungen zur Schätzung der Mflops (Million flops / s) eines Systems
  - Implementiert in Java
- Durchläufe: 1x coremark, 2x scimark2 auf je 2 Instanzen

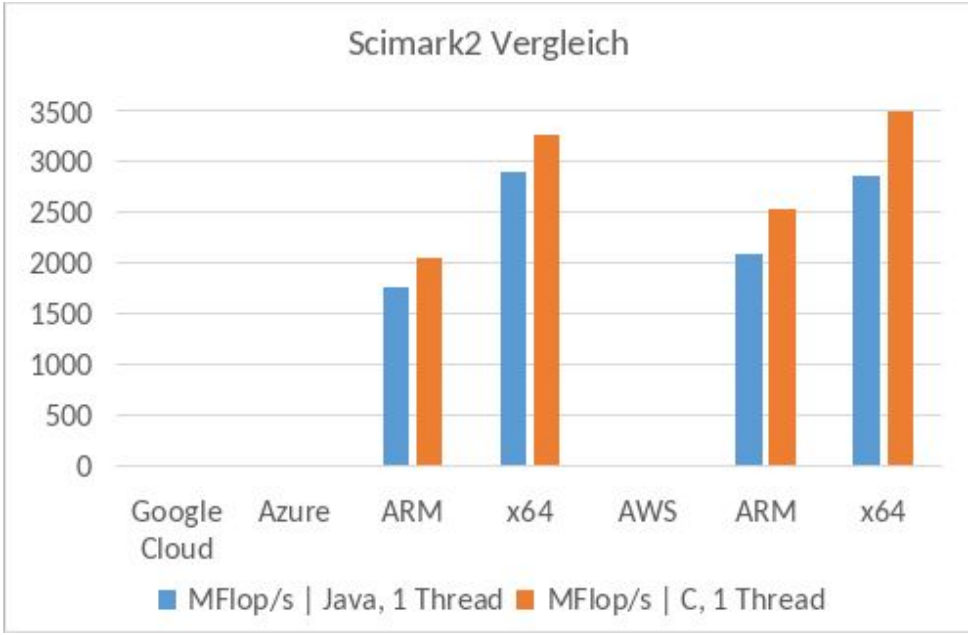
Evaluation 1 (Kosten)



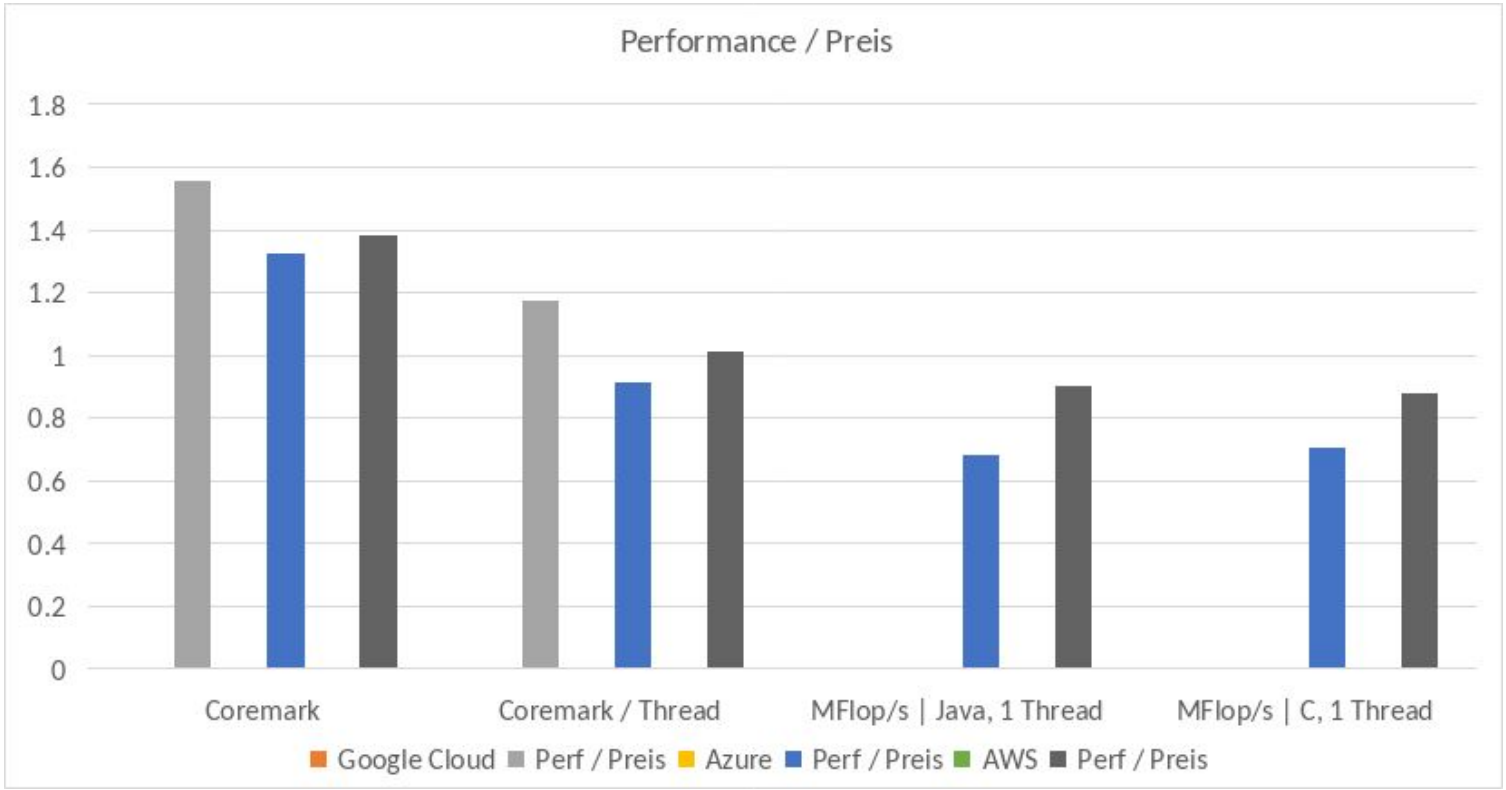


Evaluation 2 (Performance)





Evaluation 3 (Monatliche Kosten / Performance)



Einschränkungen

- Software muss explizit für Architektur gebaut werden
  - Eigene Builds / Erweitert werden umgestellt werden
  - Datenbanken, Webserver, etc. müssen ARM-Build bereitstellen
  - Gilt auch für Docker Images -> Lösung: Multi-Arch Images
- Jede vCPU entspricht genau einem Kern
  - Bei allen getesteten Providern
  - Nur 1 Thread pro Kern möglich
- Provider spezifische Einschränkungen
  - z.B. Keine verschachtelte Visualisierung, Secure-Start, Windows-Images, etc.…

Anmerkung und Konklusion

- Bis zu 40% Einsparung sind bei den Kosten möglich
  - Konkrete Anwendung muss jedoch zuvor getestet werden
- AWS sieht die Cloud-Zukunft in ARM
  - Viele SAAS Angebote bereits auf ARM verfügbare
  - Guides für Migration, Optimierung und Benchmarking
  - Neueste Maschinen-Serie (M8) aktuell nur auf ARM verfügbar
- Präsentation ist die destillierte Variante der Dokumentation
  - Setup, Skripts, gefundene Bugs in PerfKitBenchmark, etc. in Doku
  - https://github.com/Pystronic/clc3-arm-testing