Tailoring Simultaneous Inversion for preemptible Kubernetes clusters on the cloud

Malcolm Griffiths, Raafat Abdul Alim, Jaime Ramos-Martinez and N. Chemingui, PGS, Bob Clapp, Stanford University.*

Summary

We describe a cloud-native, fully fault-tolerant redesign of computationally intensive seismic high-performance computing (HPC) algorithms specifically optimized for execution across multiple Kubernetes clusters utilizing preemptible or spot market compute instances. We use the simultaneous inversion of velocity and pre-stack reflectivity algorithm as an example and demonstrate the compute and IO scalability of Kubernetes clusters when applied to high-frequency seismic inversion problems. Our strategy paves the way for advanced seismic imaging and inversion that delivers high-quality earth models in a cost-effective manner while reducing the cycle time of production projects.

Introduction

Cycle time reduction of seismic HPC applications is a continuously evolving subject. Techniques aimed at distributing the computational workload across multiple systems and multiple CPU cores utilizing OpenMP and MPI can be found in the literature from the early and mid 1990's, as illustrated in works like Amundsen et al (1996) and Ober et al (1997). The horizontal scaling of applications has historically faced limitations due to the availability of compute resources in local dedicated centers and performance constraints of I/O subsystems. The advent of publicly available cloud-based computing systems has created an alternative to traditional, privately owned HPC clusters with major cloud service companies offering Infrastructure as a Service (IaaS). Correctly leveraged, IaaS can offer an affordable alternative to the high costs of building, maintaining, and upgrading private clusters, while facilitating on-demand HPC growth and automatic rescaling of corporate HPC footprints. Work by Okita et al (2019), Witte et al (2019), and Debens et al (2022), highlight previous attempts to build and cost-optimize seismic processing algorithms across various cloud-based platforms.

In this paper we describe how we have reengineered our HPC algorithms to utilize cloud-based computing resources using Federated Kubernetes Clusters running preemptible compute instances to create a cheap and efficient alternative to dedicated on-premises clusters. We show a successful example of performing simultaneous inversion in the cloud, highlighting the intense computational needs of the process and the required architecting of the algorithm.

Kubernetes as a platform for HPC

Kubernetes is a container orchestration platform that performs many functions of a traditional job scheduler . It consists of a master control plane and then one or more worker nodes. The control plane provides automatic cluster provisioning, pod scaling, and load balancing as part of the cluster scheduler. The compute workload is executed via one or more containerized pods on the worker nodes. The worker nodes are completely configurable in terms of memory, storage, and core counts allowing the problem requirements to determine the configuration of the provisioned cluster. Individual worker nodes can be either on-demand/reserved or, the cheaper alternative, preemptible/spot instances. Successful utilization of preemptible instances requires an application update to handle the variable preemption rates.

Simultaneous Inversion of velocity and pre-stack reflectivity

Simultaneous inversion, described by the workflow in Figure 1, is a novel inversion solution for the joint estimation of velocity and angle-dependent reflectivity. The modeling engine utilizes the wave equation parameterized in terms of velocity and vector reflectivity while the inversion step utilizes an efficient scale separation of the inversion gradient via inverse scattering theory (Yang et al., 2021). The scale separation is key to minimizing the crosstalk of the velocity and the reflectivity. Vector reflectivity parameterization of the wave equation enables a fully data-driven inversion without requiring the density information. The vector reflectivity also provides the necessary information to compute pre-stack image gathers for angle-dependent reflectivity. For further details on computing the scattering angle from the gradient of the forward propagating pressure wavefield and the vector reflectivity, refer to Chemingui et al. (2023). The inversion workflow is accomplished by implementing individual forward shot modeling and adjoint-state gradient computations, horizontally scaled for efficiency, and coupled with a vertically scaled accumulation and model update stage.

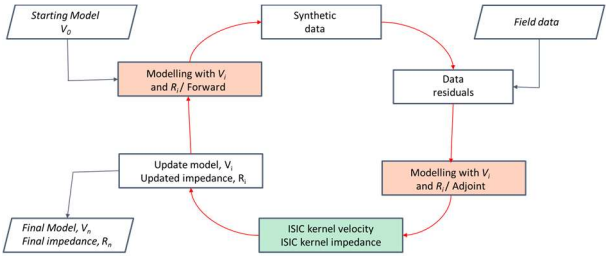

Figure 1: Simultaneous inversion schematic.

Tailoring Simultaneous Inversion for Kubernetes

Tailoring High-Performance Compute algorithms for preemptible Kubernetes clusters

An embarrassingly large number of seismic HPC algorithms can be divided into a series of group-independent, modular, linear workflows. For simultaneous inversion this translates into shot-based forward modelling, residual calculation, and adjoint modelling. Completing the iterative process requires a linear accumulation of the individual shot gradients, and the inversion / model updating step.

Partitioning monolithic applications like simultaneous inversion into group-independent linear workflows enables application execution across one or many task-specific node configurations with the Group controlling the highest level of parallelism. Identifying the modular yet linear aspects of the application facilitates a natural means of performing roll-back recovery and implementing coarse-grained preemption support in the event of pod eviction/preemption.

A key component of the redesigned HPC algorithms is the job orchestration layer. This is a low compute, low memory, application responsible for monitoring, distribution, and scheduling the main application's compute elements. A separate Redis server serves as an intermediary between the orchestration layer and the Kubernetes scheduler with Kubernetes being responsible for the compute node provisioning and the task execution.

Limitations in the VPC address space, capacity restrictions, and spot instance availability place an upper most limit on the eventual Kubernetes cluster size in terms of total core count and available worker nodes. Using shot groups as the highest level of parallelism for simultaneous inversion the maximum number of simultaneous compute tasks currently peaks at 10,000 to 15,000, without resorting to additional vertical scaling. For very large production jobs involving tens to hundreds of thousands of individual compute tasks it becomes beneficial to automatically distribute the work across multiple Kubernetes clusters or what is commonly referred to as a Federation of clusters.

Preemption support is achieved by periodically saving the application metadata, which includes information on the runtime status of the job and/or individual tasks, as well as additional application checkpointing to global persistent storage. The checkpointing includes any non-persistent arrays and output buffers for each task. The rate and complexity of the checkpointing is balanced between the recompute costs and the associated IO costs of the preemption support. This checkpointing places an enormous burden on the IO layer and associated IO components with moderatly sized inversion problems generating upwards of 1 to 2 Petabytes of temporary data per iteration. A situation that would be untenable in traditional private data centers. It does, however, open many possibilities for data analytics and enhanced data processing of intermediate products before the accumulation and inversion phase.

Figure 2 shows a schematic of a redesigned simultaneous inversion application with each independent group being assigned to an individual compute instance. The application and preemption support datasets are written to various cloud storage containers all of which are accessible to the accumulation and inversion phase.

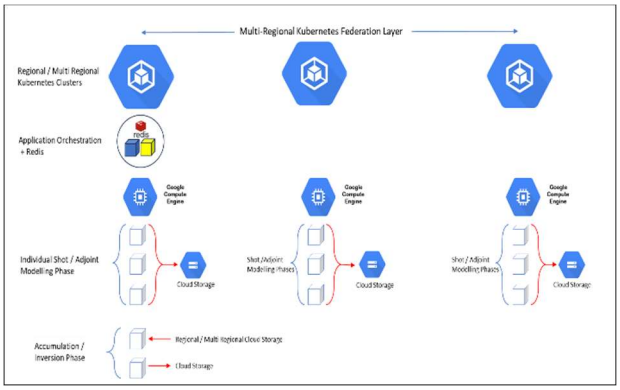

Figure 2: Redesigned simultaneous inversion schematic for multi-cluster exection on preemptable compute instances.

Application to Field Data

Our restructured seismic inversion application was applied to field data acquired in the Cape Anguille/Orphan Basin region offshore Canada. This is a narrow azimuth, multi-sensor dataset consisting of 16, 8km cables with a 1.6km cross spread. A total of 32440 shots were selected for the inversion giving a spatially even coverage. A total of 3 iterations were run at a maximum of 40Hz. Initial analysis of the inversion parameters determined the optimum system configuration for the forward and adjoint modeling tasks to be 32vCpus and 128Gb of memory. The accumulation and inversion phases used a secondary resource pool consisting of 64vCpu and 512Gb VMs. These node pools were provisioned across a federation of 4 separate GKE clusters. Figures 3a to 3d show the VM usage (running and pending) for each cluster. The blue-shaded areas highlight periods where individual clusters are starved for capacity. Cluster federation helps to buffer the effects of localized availability issues by automatically redistributing the workload.

During the initial shot and adjoint modelling phase all 4 clusters were fully utilized with approximately 8000 tasks (shots) being assigned to each cluster or the equivalent of around 1 to 1.1million vCpu's. The Kubernetes scheduler automatically removes idle instances resulting in the characteristic rapid upscaling and gradual downscaling

pattern seen in the utilization graphs. This ensures that costs are only incurred while the VM's are active. The IO generated for each shot modeling task was approximately 3.2TBytes of mostly temporary data. Figures 4a and 4b show the cumulative IO throughput for all clusters with peaks above 1.4TBytes/sec. Each iteration involved the generation of over 100 Petabytes of temporary data. Figure 5 shows a sample inline section for the different quantitative imaging volumes: high-resolution velocity model, angle-dependent reflectivity, full and partial stacks and relative density. Notice the overall flatness of the gathers that are used for quantitative interpretation.

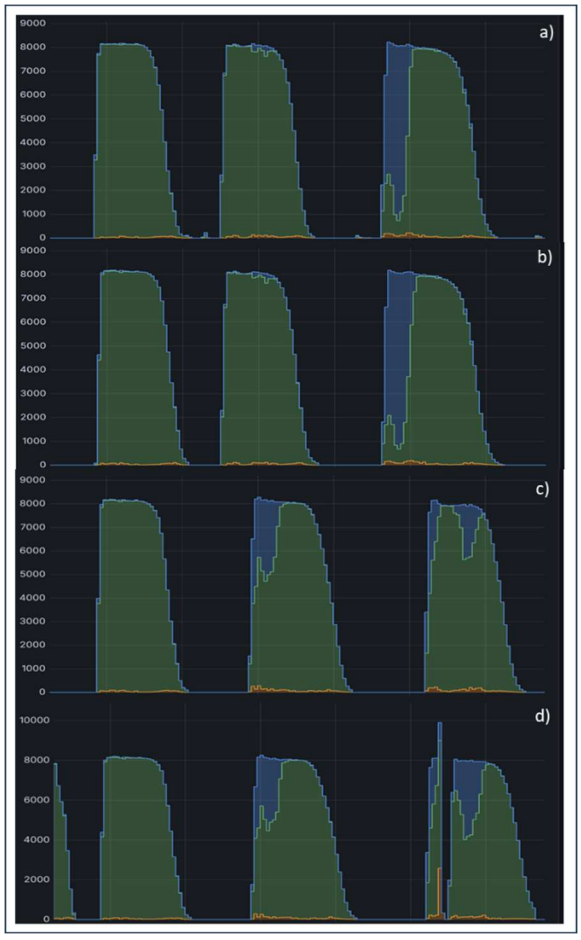

Figure 3: VM utilization across each of four Kubernetes Clusters. Instances were 32vCpus, 128GB. a) US-West-1, b) US-West-2, c) US-West-3, d) US-West-4.

Conclusions

This work demonstrates a cost-effective and fault-tolerant solution for high frequency simultaneous inversion and shows how the CPU and IO requirements can be met through a combination of horizontally scaled Kubernetes clusters and massively parallel cloud storage. Our cloud-native strategy paves the way for advanced seismic imaging and inversion that delivers high-quality earth models in a cost-effective manner while reducing the cycle time of production projects. Future work will focus on the utilization of the intermediate products generated from the individual adjoint modeling phases for the purpose of enhanced imaging.

Acknowledgments

We would like to thank PGS for permission to show this work, PGS MutiClient for the access to the field data, and the Google support teams for their ongoing assistance and feedback. We thank Sean Crawley, Guanghui Huang, Mariana Gherasim and Elena Klochikhina for their contribution to this work.

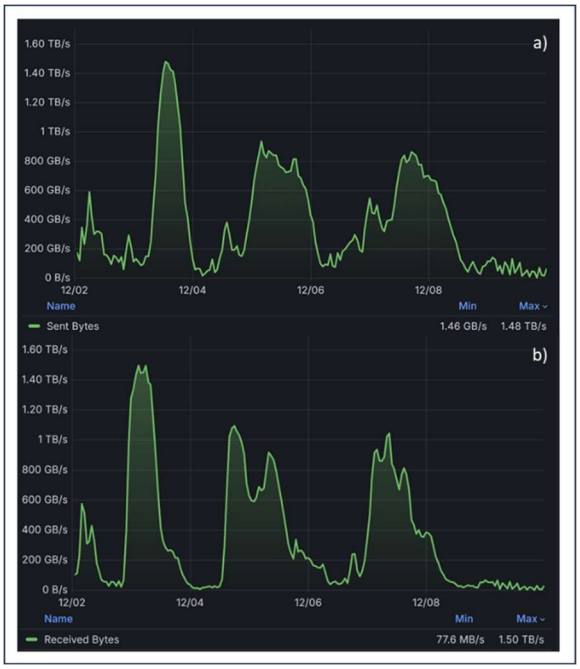

Figure 4: Cumulative IO throughput for all clusters. a) Read throuput (peak=1.48TB/s), b) Write throughput (peak=1.5TB/s).

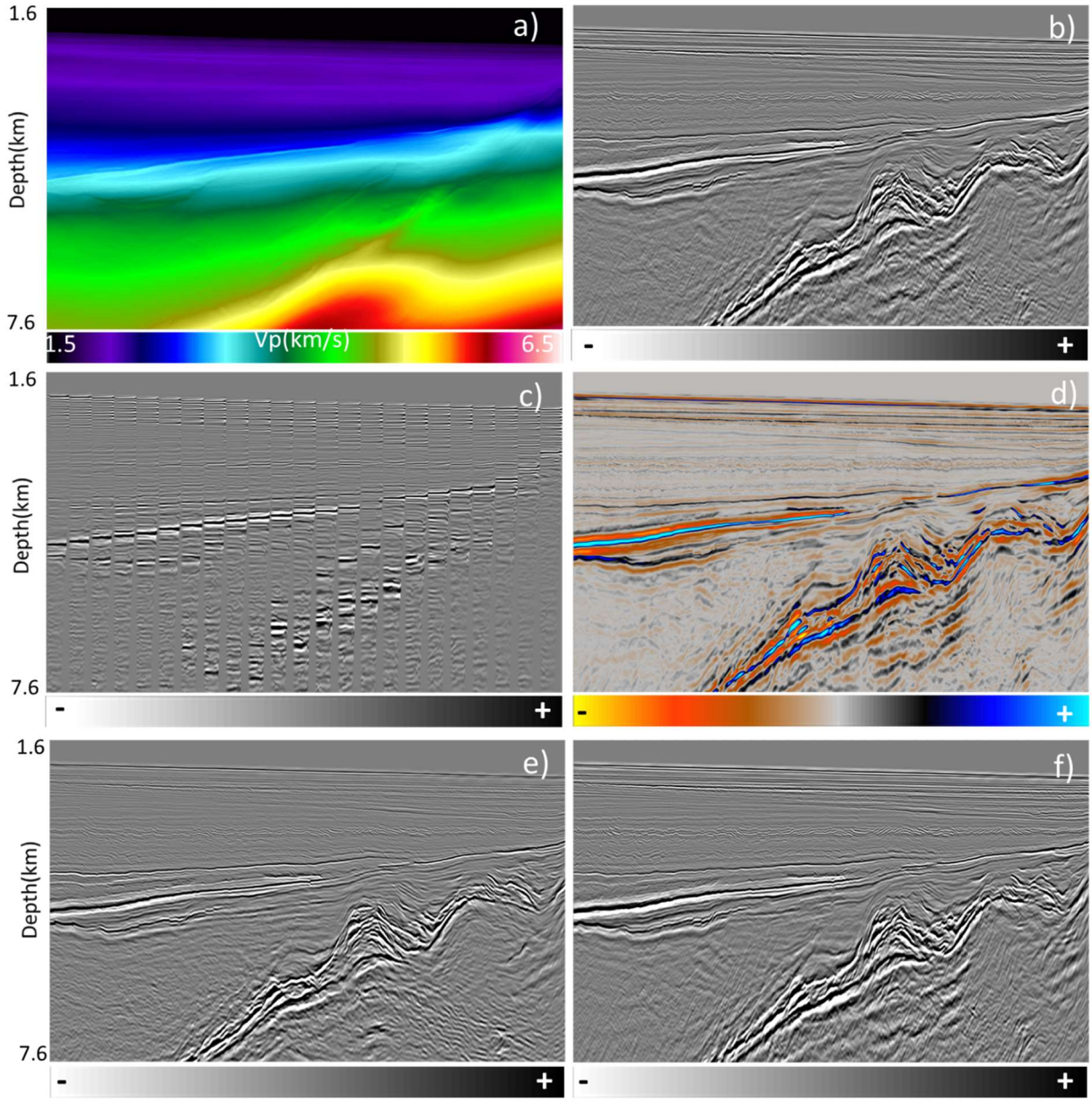

Figure 5. Cape Anguille field data example. Inline section for the a) inverted velocity model, b) full reflectivity stack, c) angle gathers, d) relative density, e) near and f) far angle stacks