HPC

Cloud against the storm: Clemson's 2.1 million VCPU experiment

Kevin Kissell

record for the scale of computation on a single problem in a public cloud: 2.1 million cores. It was an adventure that I think is worth sharing.

The problem: Managing evacuation routes

Evacuations from threatened coastal areas are almost entirely done by private automobiles on public roads. The evacuation routes tend to be limited in number and capacity by the terrain and the weather: Bridges and causeways are bottlenecks, and alternate routes across low-lying areas may already be flooded. Optimizing vehicular “throughput” over the available channels is incredibly important—and quite difficult.

Brandon Posey of Clemson, under the supervision of Professor Amy Apon, has been working on building traffic flow models based on streaming data from existing street-level traffic cameras. Machine vision software capable of anonymously distinguishing vehicles and vehicular motion exists, but processing and correlating the feeds from multiple cameras at the necessary scale is a herculean computational task. A single evacuation zone can have thousands of these cameras, and the full cycle of an evacuation, cleanup, and return, can take days, even weeks.

The Clemson team assumed a 10-day cycle for the experiment, and chose an evacuation area with 8500 available camera feeds, which over 10 days generate 2 million hours of video—about 210TB. TrafficVision, a commercial company that participated in the experiment, provided the software for analysis of the video segments, so that vehicle,

Find an article...

Latest stories

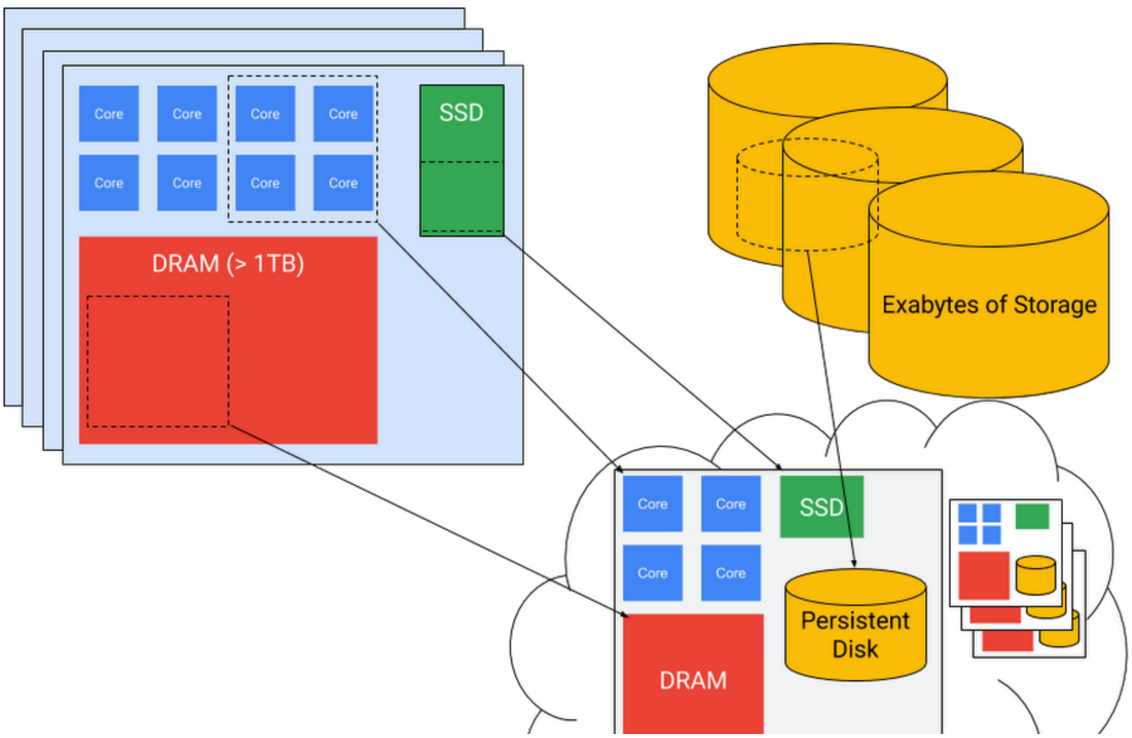
Products

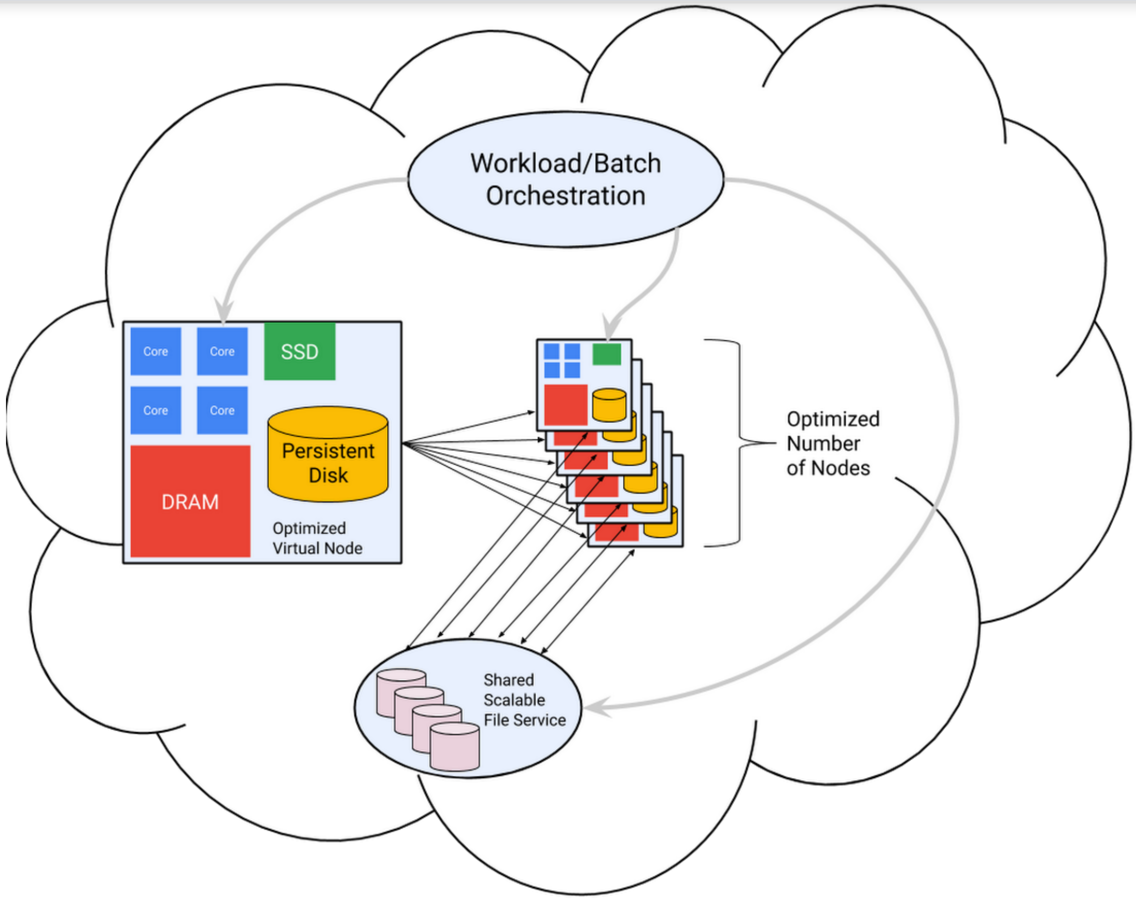
Topics

About

RSS Feed

accelerators. When these can't be computed from first principles, they are generally straightforward to determine experimentally. And once it has been determined, Google Cloud Platform provides the capability to generate an optimal custom node type that can be carved out of the Google infrastructure, as shown in the diagram below. The customer pays only for the resources actually deployed. The sweet spot for the TrafficVision application was 16 virtual CPUs and 16GB of memory per instance.





The Clemson team started with this node-out approach. The processing of individual video segments by the TrafficVision software runs most cost-effectively on small virtual

But while the nodes of the virtual supercomputer are small, and potentially ephemeral, infrastructure with CloudyCluster workflow tooling, which had already been demonstrated to work at a million-core scale.

Public cloud services like Google Cloud manage very large pools of resources for large pools of clients. Resource management must necessarily be parallel and distributed to operate at scale. This distribution and parallelism is normally transparent to users, but at the scale of this experiment, it was necessary to understand and adapt to the underlying design.

Fairness is an important property of a multi-user operating system, and Google Cloud enforces it in various ways. One of those is limiting the rate at which one user can make resource requests of the shared infrastructure. With no limit, a malicious (or more likely simply erroneous) script could saturate the resource management system, acting as a de facto denial-of-service (DOS) attack. Google Cloud solves this problem by putting quotas on the number of GCP REST API requests that can be made per 100 seconds. The default is 2000. At a minimum, an API request is made to create each VM instance, and another is required to delete it when it is no longer needed.

2000 API calls per 100 seconds, 20 per second, is more than enough for most applications, but at the scale of Clemson's experiment, 93,750 virtual machine instances, it would take more than an hour just to provision or free all the worker nodes. The system supports a tripling of the quota to 6000 per 100 seconds, but even at this level, spinning down after a run would take 26 minutes, during most of which tens of

Find an article...

Latest stories

Products

Topics

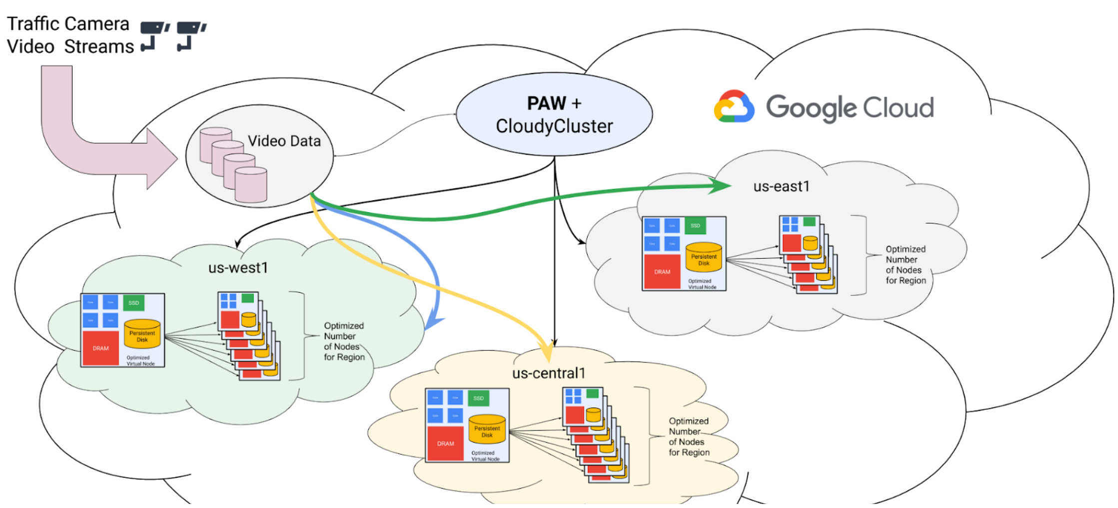
About

RSS Feed

scale of the experiment. With provisioning running open-loop in parallel in multiple zones.

To regulate the flow of creation requests, the Clemson team used batching of GCP API requests. Instead of instance creation running open-loop at the aggregate maximum rate, instances were created in bursts of no more than 5000, with randomized pauses between batches. This allowed administrative connection setup to make enough forward progress to avoid timeouts.

Clemson's final operational model was something like the diagram below. They spread the work across six Google Cloud geographical regions (and the administrative zones within the regions), to parallelize the provisioning process and get beyond the API rate limitations. This also allowed them to cast a wide net when looking for available preemptible cores.



While some ingenuity was required, the experiment was a success. Using Google Cloud, Posey and the Clemson team showed it is possible to draw on worldwide spare capacity to deploy very large scale emergency computations in the interest of public safety. Most hurricane evacuations won't require the full 2-million-plus virtual CPUs used here to process evacuation data in real time, but it's reassuring to know that it's possible. And a source of some pride to me that it's possible on Google Cloud. Kudos to Brandon Posey and Amy Apon of Clemson University, Boyd Wilson and Neeraj Kanhere of TrafficVision, Dan Speck of the Burwood Group, and my Google colleagues Vanessa July, Wyatt Gorman, and Adam Deer.

POSTED IN: HPC—GOOGLE CLOUD PLATFORM

RELATED ARTICLES

New Compute Engine A2 VMs—first NVIDIA Ampere A100 GPUs in the cloud

Google Cloud and NVIDIA's enhanced partnership accelerates computing workloads