DSC-204A: Scalable Data Systems, Winter 2024

Sky Computing

Lecturer: Ion Stoica *Scribe: Prasannakumaran Dhanasekaran, Krish Rewanth Sevuga Perumal*

1 Sky

The first internet demonstration conducted in 1977 showcased the transmission of information packets via radio transmitters from Menlo Park to USC. While seemingly straightforward, the operation intricately traversed three distinct networks: the Packet Radio Network, ARPANET (a wired network), and a Satellite Network, spanning across North America and Europe as illustrated in Figure 1. This demonstration underscored the sophisticated interplay of diverse network infrastructures, ultimately abstracted by the internet, thereby obviating the need for senders and receivers to comprehend underlying implementations.

Similarly, Sky is the internet for cloud computing, comprising a suite of software tools and services tailored to streamline the deployment and operation of applications across multiple cloud platforms.

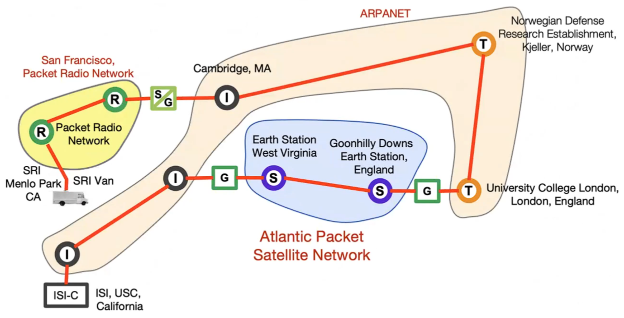

ARPANET
Norwegian Defense Research Establishment, Kjeller, Norway
San Francisco, Packet Radio Network
Cambridge, MA
Packet Radio Network
Earth Station West Virginia
Goonhilly Downs Earth Station, England
SRI Menlo Park CA
SRI Van
University College London, London, England
Atlantic Packet Satellite Network
ISI-C
ISI, USC, California

Figure 1: Architecture of Internet Demo

1.1 Why Sky?

With the increasing proliferation of regulations concerning privacy and security, navigating data processing within the confines of a specific country's jurisdiction poses considerable challenges. Operational sovereignty becomes even more stringent, particularly when a country manages the operation of data centers, creating challenges for cloud providers.

Sky aims to address multiple challenges: ensuring compliance with data and operational sovereignty regulations, leverage best-of-breed services and hardware such as Azure Confidential Computing and Google BigQuery, and facilitating resource aggregation across cloud platforms. This strategic approach is beneficial, due to the scarcity of GPUs across clouds. Sky sources GPUs from multiple cloud providers rather than relying solely on one. It also aims to mitigates costs, latency, and the risks associated with lock-in.

2 History of Sky: The Abstraction Layer

In the 2015 research paper titled ”Sky Computing: The Future of Cloud Computing” [1], the authors envisioned a novel paradigm known as Sky Computing. While the internet relies on the IP layer to abstract the network, the concept of Sky Computing proposes the implementation of a portability layer to abstract the network, enabling seamless utilization of various services. This idea parallels the analogy of the network layer in the internet and the system calls layer in operating systems, aiming to provide a unified interface for accessing cloud services across different providers.

In response, existing cloud service providers have begun offering solutions aimed at facilitating cross-platform compatibility, such as Azure ARC and Google Anthos. For instance, Anthos has the capability to run on multiple clouds and not limited to Google Cloud Platform (GCP). This abstraction layer concept holds significant similarities to the network layer in the internet and is depicted in Figure 2.

- **Portability layer is highly complex:** It is 10× more functionality than a traditional OS and similar efforts like unification of UNIX have failed in the past.
- **Portability layer is low level:** Numerous proprietary and valuable cloud services may become inaccessible. For instance, AWS alone offers over 300 services, necessitating the re-implementation of many of these services atop the portability layer.
- **Clouds have no incentives to support it:** A low-level layer would commoditize them.

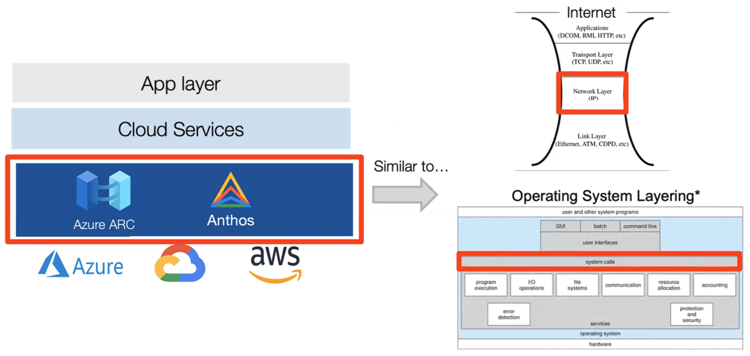


Figure 2: Portability Layer

3 Market-based Approach

In contrast to portability layers, Sky computing adopts a market-based approach to cloud services. Sky plans to align application demand with services offered by various cloud providers, thereby offering users a variety of choices. This approach allows for trade-offs to be made based on factors such as cost, performance, or locality.

3.1 Intercloud Broker

The intercloud broker functions by gathering user input, where users submit job specifications alongside desired criteria such as cost, performance, and location. Concurrently, it collects information on the services offered by various clouds. Subsequently, the intercloud broker distributes jobs across different clouds based on specified criteria and availability. It then matches application requirements, oversees computations, and handles restarts in case of failures. With multiple cloud services available, depicted in Figure 3, the broker selects the optimal service instance based on factors like cost and performance, making these services easily accessible to users.

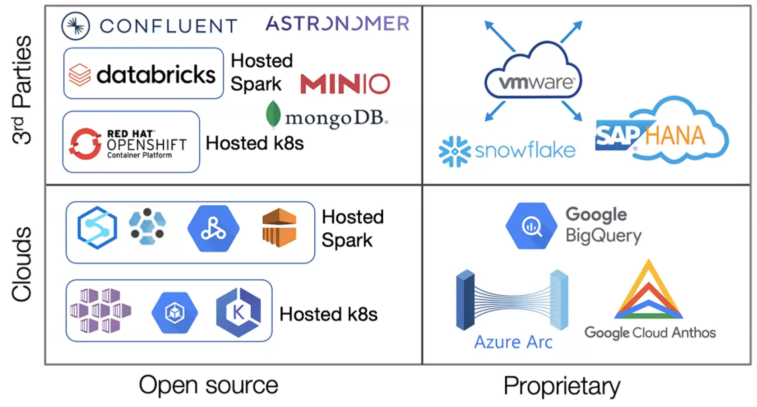


Figure 3: Available Cloud Services

The user's billing can be managed in two primary ways within the intercloud broker framework. Firstly, users may have individual accounts with different clouds, with all incurred charges directed to these respective accounts. Alternatively, the intercloud broker itself can maintain accounts with various clouds, allowing the clouds to charge the broker directly, after which the intercloud broker invoices the user accordingly. The architecture of the intercloud broker is showcased in Figure 4.

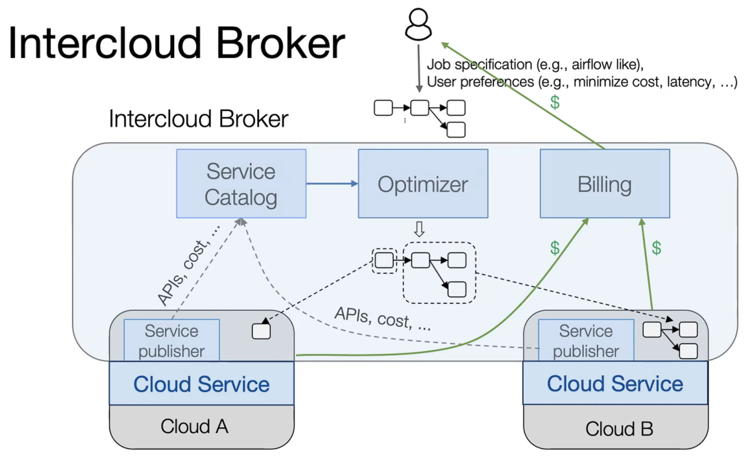


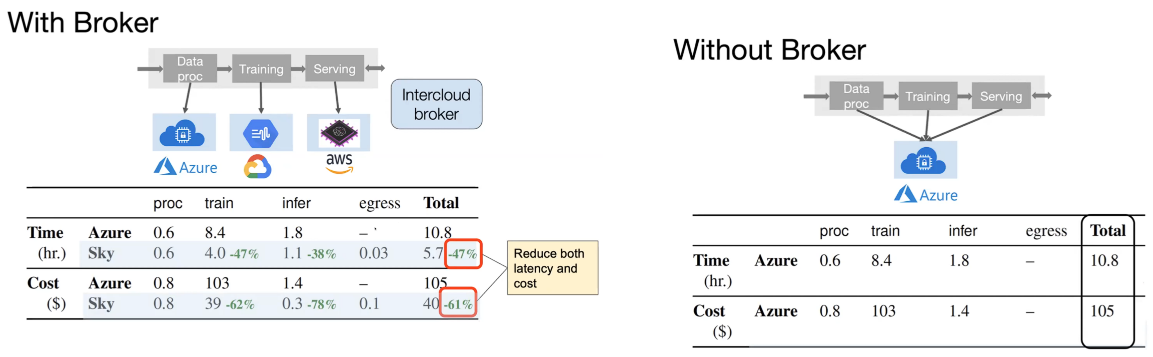
Figure 4: Intercloud Broker Architecture

3.2 Implementation of an ML Pipeline

Given the requirement for a 3-stage ML pipeline involving data processing, training, and serving, with an emphasis on cost-efficiency, a strategic approach involves leveraging different cloud platforms. For instance, utilizing Azure for data processing, transitioning to GCP for training to benefit from TPUs, and finally using AWS for serving and leveraging its inferential engine can lead to cost and time reductions.

Under this approach, the time breakdown would be as follows: 0.6 hours for data processing on Azure, 8.4 hours for training on GCP, and 1.8 hours for serving on AWS, totaling 10.8 hours. In terms of costs, it would amount to $0.8 for data processing on Azure, $103 for training on GCP with TPUs, and $1.4 for serving on AWS, resulting in a total of $105. This is illustrated in Figure 5.

This strategic distribution of tasks across different cloud platforms minimizes both time and expenditure, with negligible transfer costs incurred during the transitions between clouds.



		proc	train	infer	egress	Total
Time	**Azure**	0.6	8.4	1.8	–	10.8
(hr.)	**Sky**	0.6	4.0 -47%	1.1 -38%	0.03	5.7 -47%
Cost	**Azure**	0.8	103	1.4	–	105
($)	**Sky**	0.8	39 -62%	0.3 -78%	0.1	40 -61%

		proc	train	infer	egress	**Total**
Time (hr.)	**Azure**	0.6	8.4	1.8	–	10.8
Cost ($)	**Azure**	0.8	103	1.4	–	105

Figure 5: Cost and time of training on multiple clouds

4 Why sky can grow?

There are three main conjectures why sky can grow.

- **Compatability set is growing quickly:** Open-source software plays a significant role as a driver in the software industry, with its dominance across multiple layers of the software stack and compatibility with almost all cloud platforms.

- **Third parties provide multi-cloud services:** Third-party providers offer multi-cloud services to improve their competitiveness against established cloud platforms.

- **Clouds themselves want it:** Third-party providers not only offer hosted versions of open-source software projects like Kubernetes (k8s), Apache Spark, and Apache Kafka but also provide their own stack on other clouds, such as Azure ARC and Google Anthos. Furthermore, they support competitor's APIs on their cloud platforms to facilitate winning over competitor's customers.

Sky doesn't rely on assistance from established cloud platforms. Once it enters the market, market forces will drive its growth. Early applications will seek compatibility across all platforms, leading to the creation of interfaces for multiple cloud platforms. This, in turn, will increase the range of services and expand compatibility sets across different platforms.

5 SkyPilot

SkyPilot is an intercloud broker designed for AI applications, with its primary goal being to address GPU scarcity and high costs. When a user submits job requirements such as "Need 4 V100 GPUs," SkyPilot selects and runs on the most suitable cloud for the task.

The process starts by taking the user's requirements and sorting available resources by price. Next, SkyPilot fetches feasible offerings from the service catalog and optimizes them to determine the best cloud or virtual machine (VM). Subsequently, it communicates with the provisioner to allocate resources in the chosen cloud, and the executor executes the job, as illustrated in Figure 6.

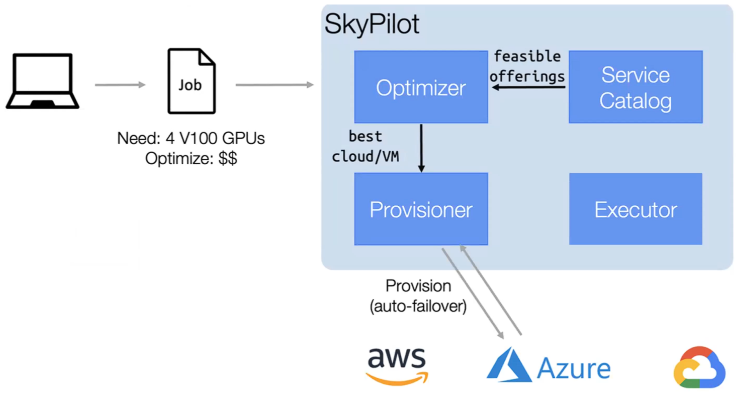


Figure 6: Job Life Cycle in Sky Pilot

SkyPilot optimizes costs by preempting and identifying instances that become cheaper as they become available. It then loads the new resource from saved checkpoints. An example of this process is depicted in Figure 7 where the cost is saved by $\sim$ 70%.

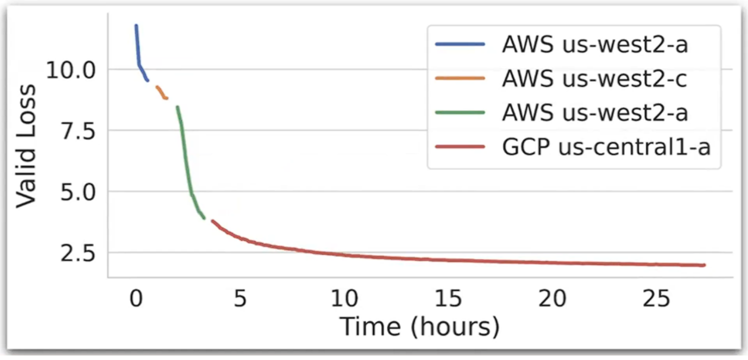


Figure 7: Saving Cost using SkyPilot

5.1 Addition of new clouds

Initially, SkyPilot supported only three cloud platforms: AWS, Azure, and GCP. Over time, more cloud support was added by the community (Figure 8), facilitated by third parties aiming to make their customers' workloads easily portable across major clouds.

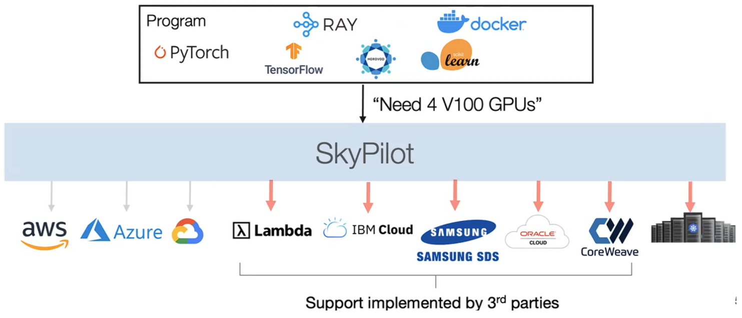


Figure 8: New clouds added to SkyPilot

References

[1] Tewari, A., Nagdev, P., & Sahitya, A. (2015). Sky computing: the future of cloud computing. IJCSIT) International Journal of Computer Science and Information Technologies, 6(4), 3861-3864.