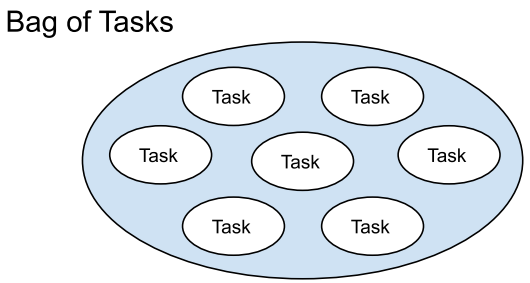
9th International Conference "Distributed Computing and Grid Technologies in Science and Education"
(GRID'2021, 5-9 July 2021)

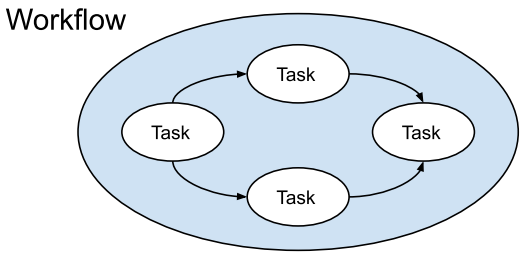
Evaluating Different Options for Scientific Computing in Public Clouds

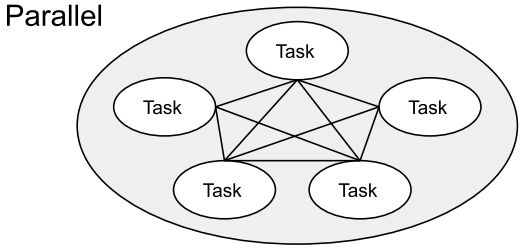
Oleg Sukhoroslov, Svyatoslav Krasnov, Yulia Tsareva

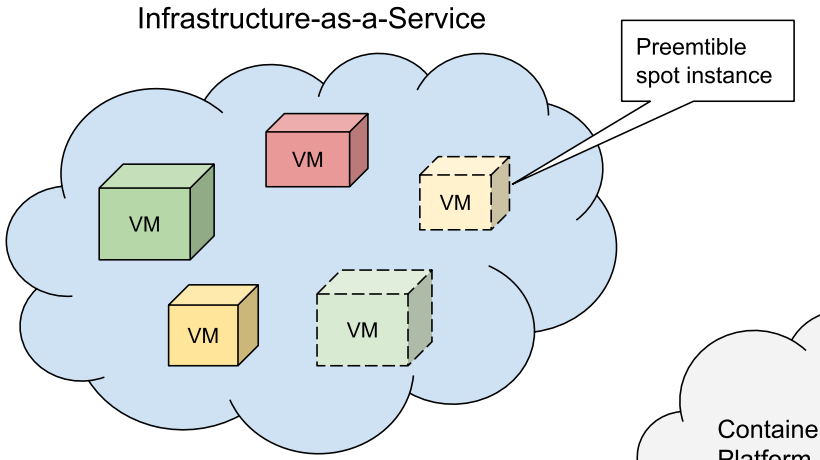
Institute for Information Transmission Problems of RAS,
National Research University Higher School of Economics

Scientific Computing in Public Clouds

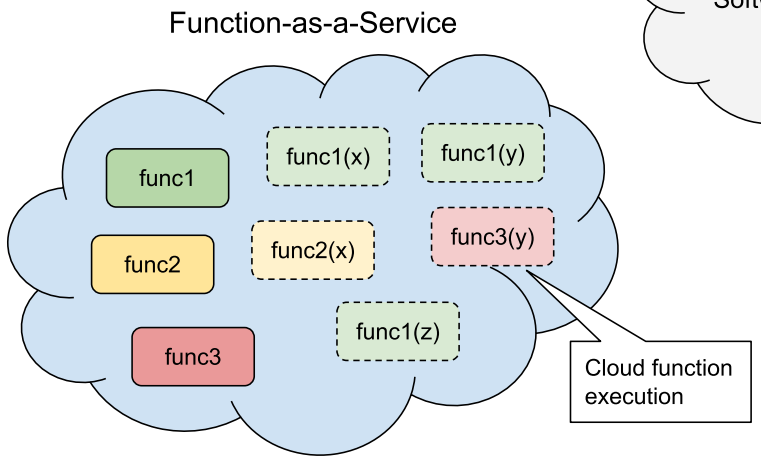








Containers-as-a-Service,
Platform-as-a-Service,
Software-as-a-Service...



Scientific Computing in Public Clouds

Pros

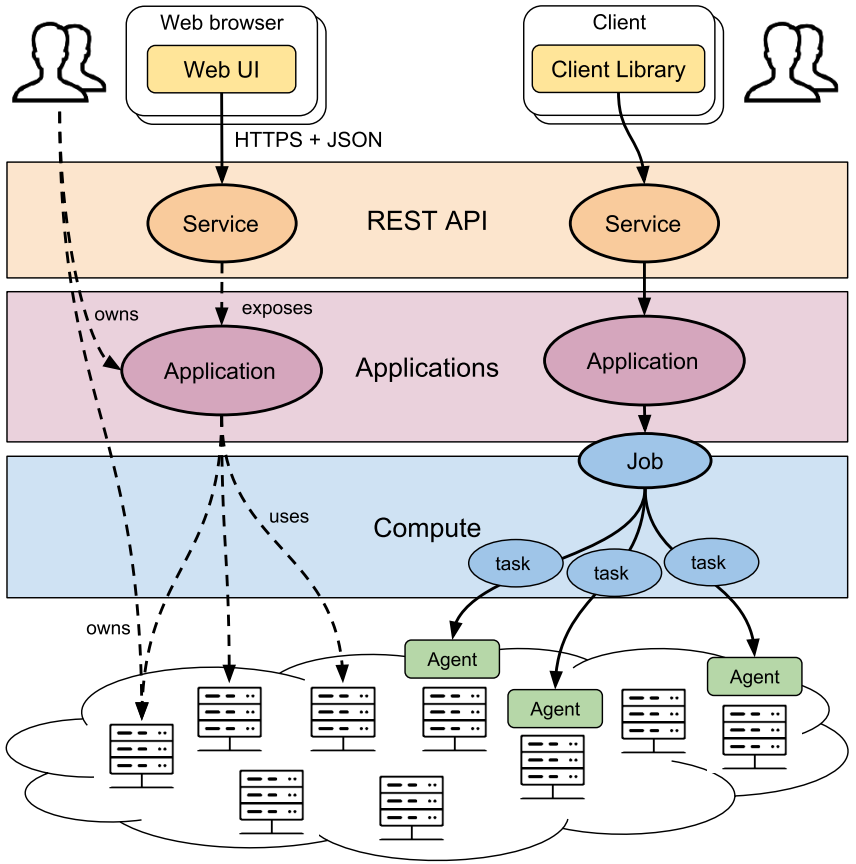
- Lower time-to-result via quick resource provision
- No need to build and operate compute cluster on-premises
- Quickly adapt to changing workloads
- Benefit from massive economies of scale

Cons

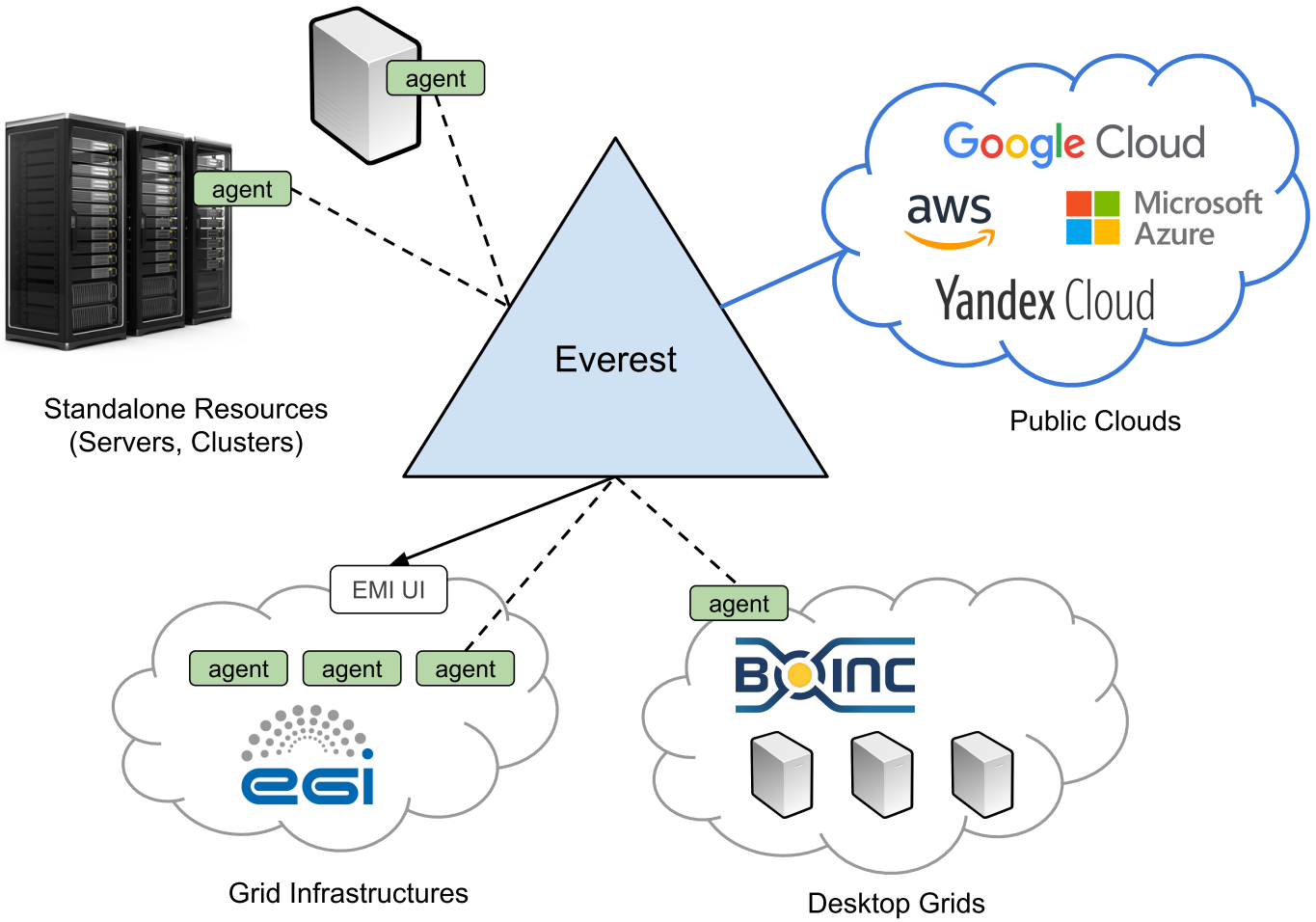
- Need to pay for resources (grants can help)
- Higher costs than on-premises cluster for some workloads
- Availability, data transfer bottlenecks, performance unpredictability
- New computing models, learning curve, provider lock-in

Everest

- Web-based platform supporting
  - Publication of computing applications as services
  - Execution of applications on computing resources
  - Sharing applications and resources with other users
- Platform as a Service
  - Multiple users, no installation
  - Access via remote interfaces (Web UI, REST API)
- Computing resources are attached by users
  - Standalone machines and clusters
  - Grid infrastructures, desktop grids, clouds
- Public instance with open registration
  - https://everest.distcomp.org/



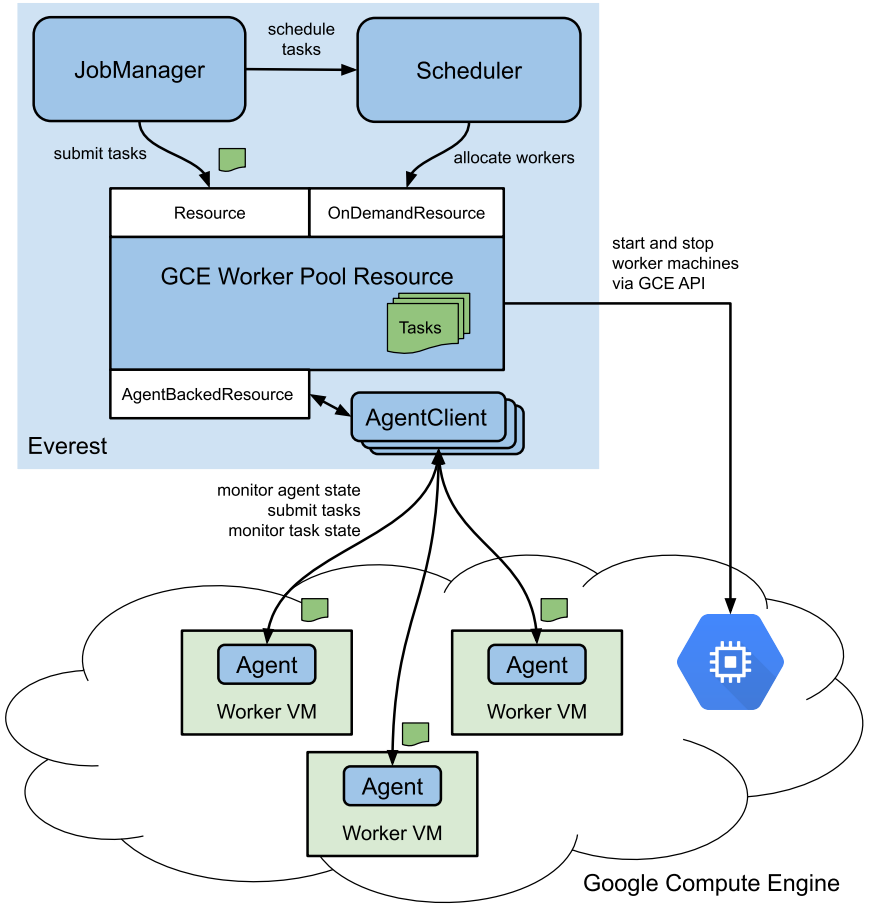
Integration with Computing Resources



Integration with IaaS Provider

- On-demand worker pool resource
  - dynamically allocates cloud resources by starting virtual machines with Everest agent when pending tasks exist
  - distributes tasks between attached agents
  - handles agent disconnections
  - releases resources by stopping VMs with idle agents after timeout
- User should define
  - VM instance type and image
  - task slots per VM
  - max number of VMs

Volkov S., Sukhoroslov O. Simplifying the Use of Clouds for Scientific Computing with Everest // Procedia Computer Science, vol. 119. 2017. pp. 112-120

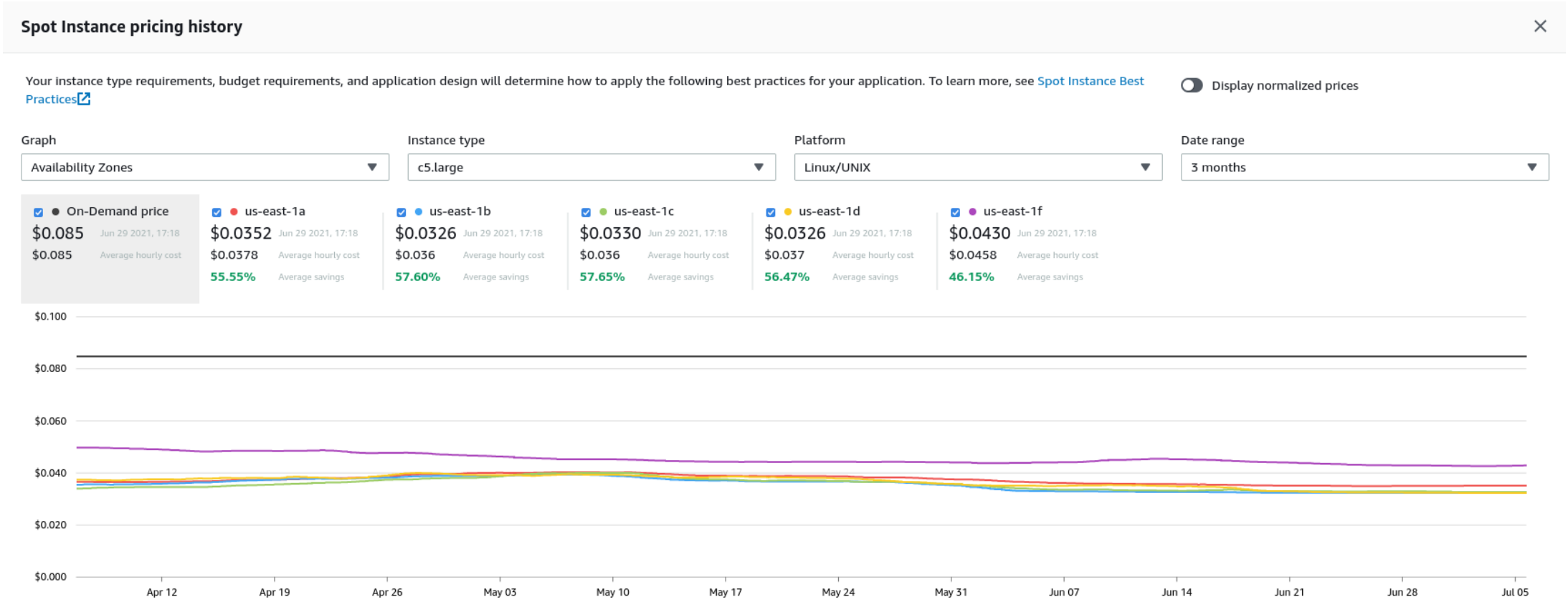


Leveraging Spot Instances

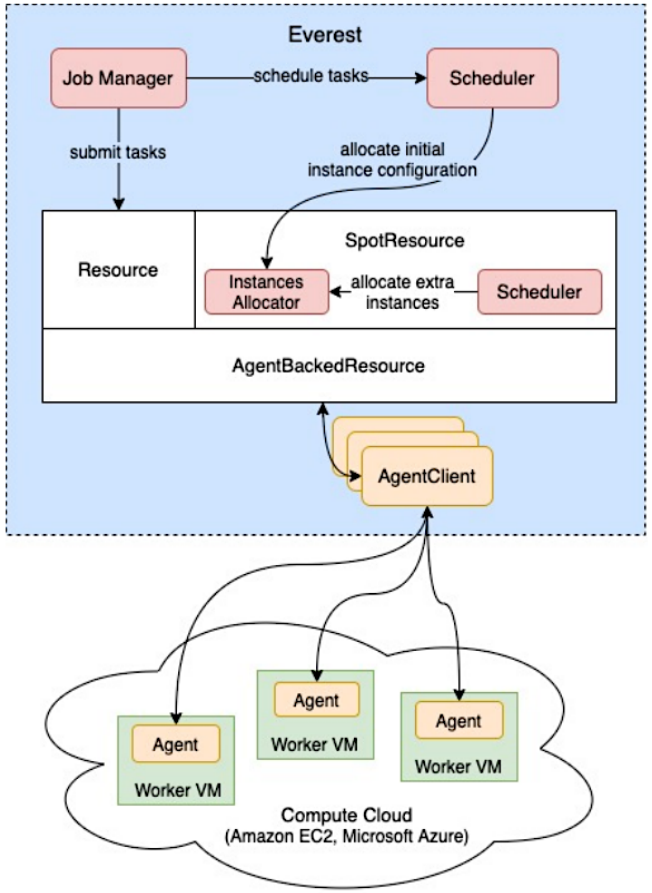
- All major providers offer preemptible/spot instances
  - a way to sell the spare compute capacity
- Much lower price than regular on-demand instances
  - the price can change dynamically based on demand
  - a user can specify the maximum price he is willing to pay
- Spot instance can be interrupted by provider when
  - a capacity is needed for regular instances
  - the spot price is above the user limit
- Challenges
  - finding the best instances configurations
  - task scheduling and handling instance interruptions

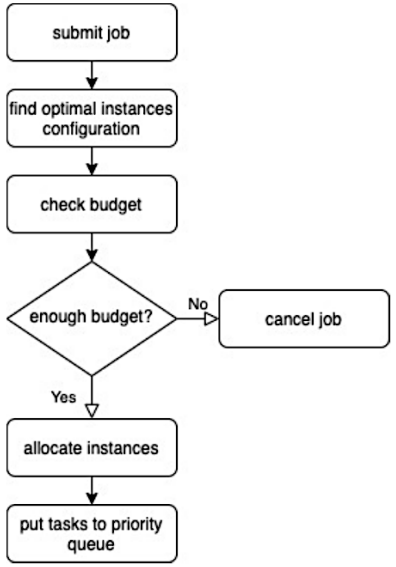
Instance	On-demand	Spot
c5.large	$0.085	$0.0325
c5.xlarge	$0.17	$0.0648
c5.2xlarge	$0.34	$0.1574
c5.4xlarge	$0.68	$0.2591

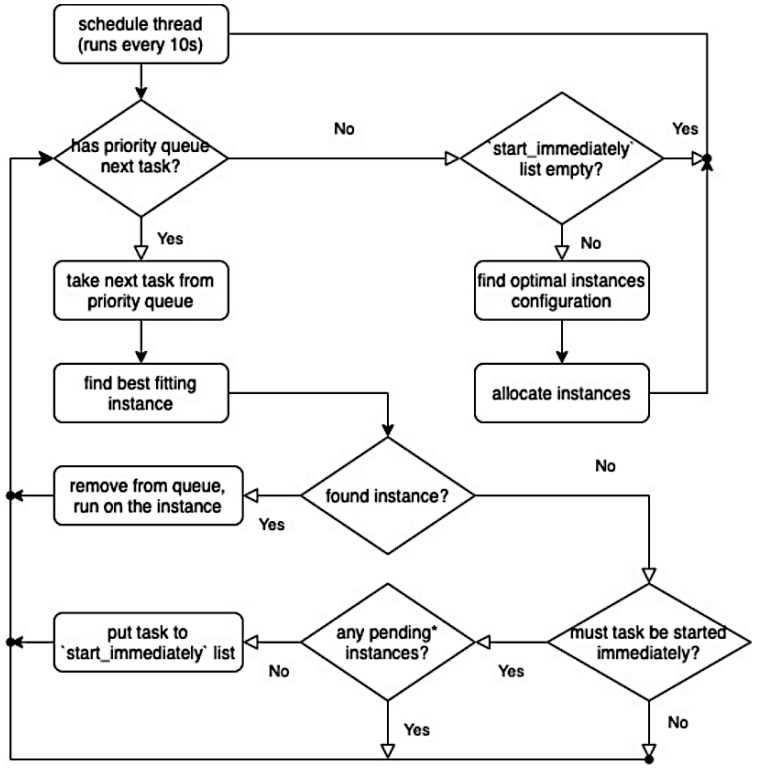
EC2 Spot Instance Pricing History



Using EC2 and Azure Spot Instances via Everest



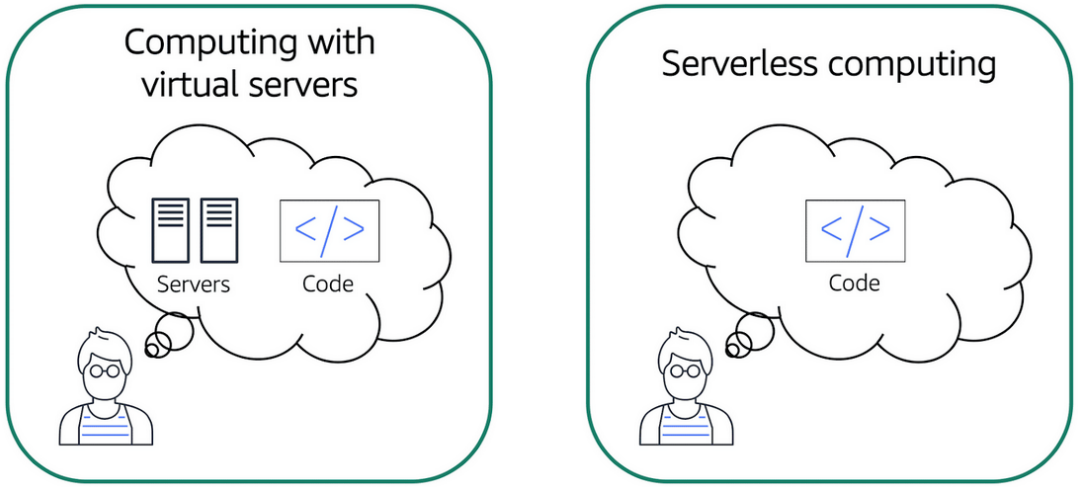




Experiments with Spot-based Resources

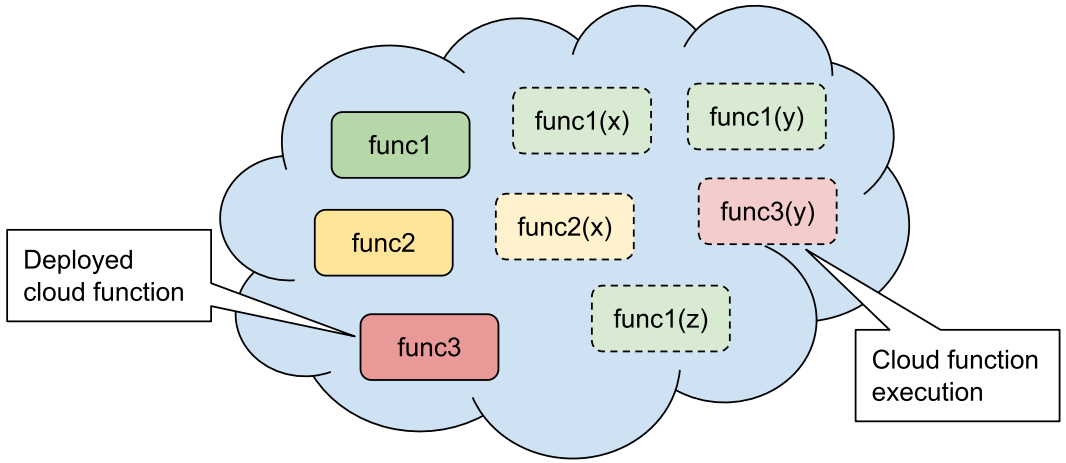
	Amazon EC2	Microsoft Azure
Molecular docking application (100 tasks)		
Run time	38m 14s	37m 15s
Used instances (price per hour)	c5a.large x 3 ($0.0324)	F4s v2 ($0.095892) F2s v2 ($0.047946)
Cost	$0.06012	$0.0822
Geophysical application (670 tasks)		
Run time	5h 9m 5s	5h 21s 16s
Used instances (price per hour)	c5a.large x 11 ($0.0324)	F4s v2 x 5 ($0.095892) F2s v2 x 1 ($0.047946)
Cost	$1.812	$2.779

Serverless Computing



- Management of the underlying infrastructure is abstracted away from the user
- Allocation and deallocation of resources is performed automatically
- Computing is done on demand in short bursts with the results persisted to storage
- When a cloud application is not used, it does not consume computing resources

Cloud Functions (Function as a Service)

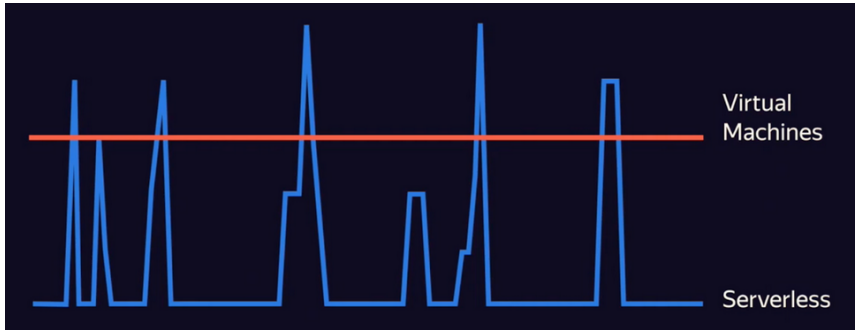


A compute model wherein a piece of code (function) is deployed and then run in response to incoming events (e.g. user requests)

- each event causes a single execution of the function to run and act on the event
- many function executions can run concurrently (automatic elastic scaling)
- charges are based on resources consumed by function executions

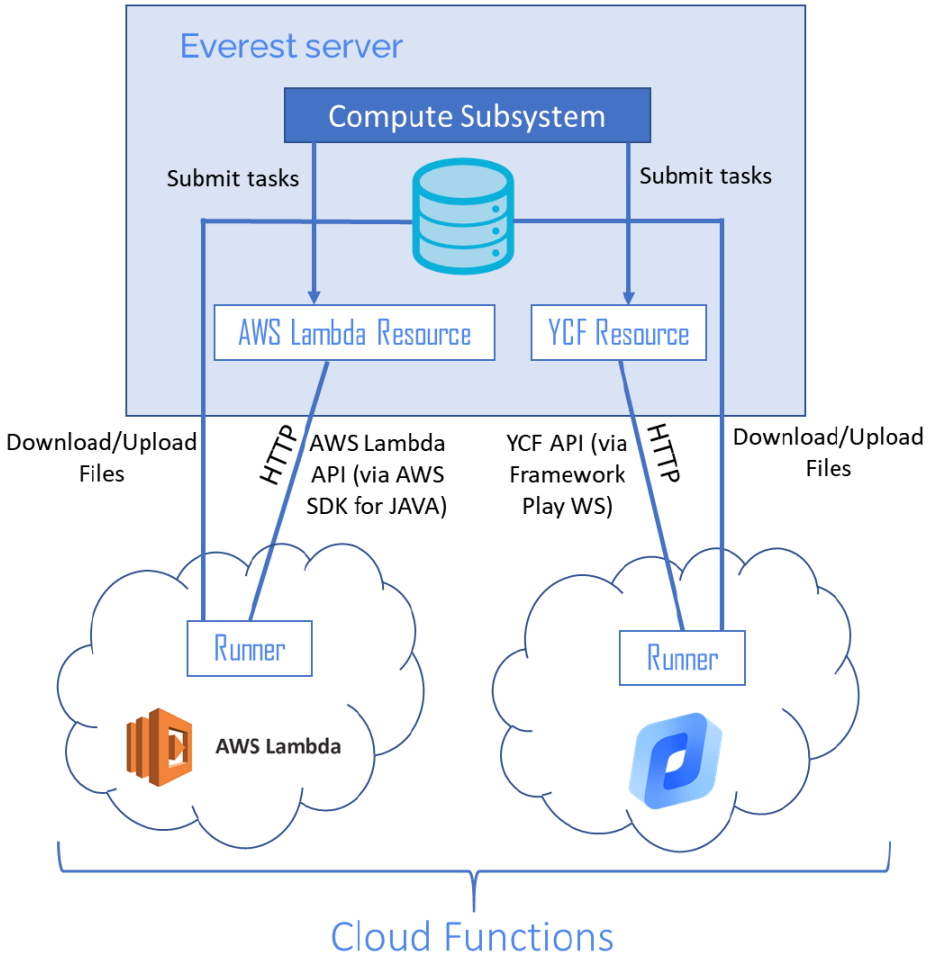
Cloud Functions

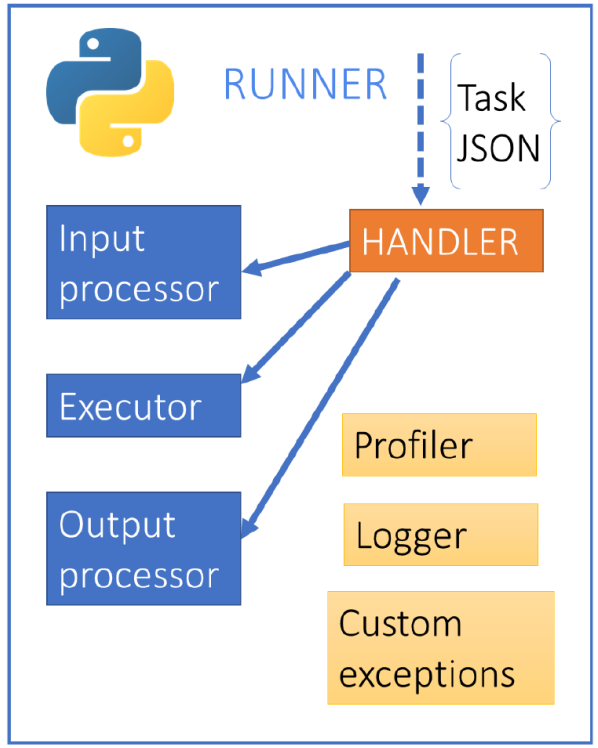
- Pros
  - Transparent cost model (no pay for idle time)
  - Faster deployments (~1 second, no need to provision VMs)
  - Inherent elasticity (no need for autoscaling policies)
  - Free usage tier
- Cons
  - Execution time and resource limits
  - Limited execution environments
  - Most expensive on a per-compute-cycle basis
  - Dependence on specific provider
- Is it suitable for scientific computing workloads?
  - Ability to run unmodified application binaries
  - Ability to use custom environment with dependencies



	AWS Lambda	Yandex Cloud Functions
Execution time limit	900s	600s
Memory size	128MB – 10GB	128MB – 2GB+
Request size limit	6MB	3.5MB
Temporary storage size	512MB	512MB
Concurrent executions	1000+	10+
Price per GB*hour	$0.06	3,42₽ ($0.047)
Free tier (per month)	400,000 GB-seconds	10 GB-hours

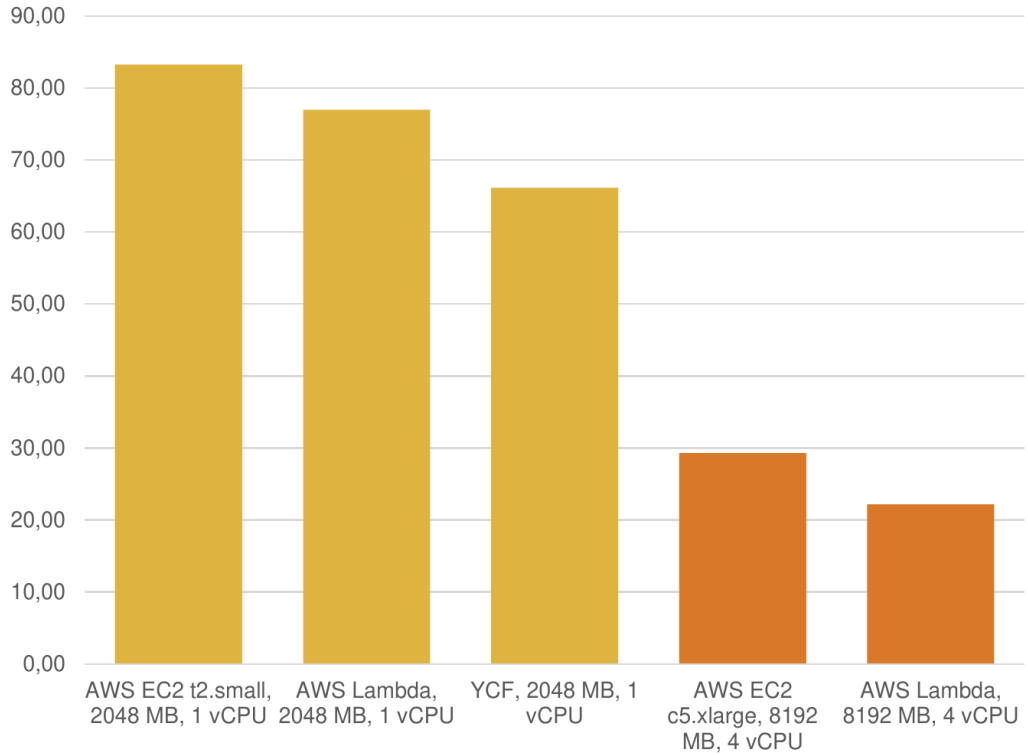
Integration of Everest with FaaS Providers





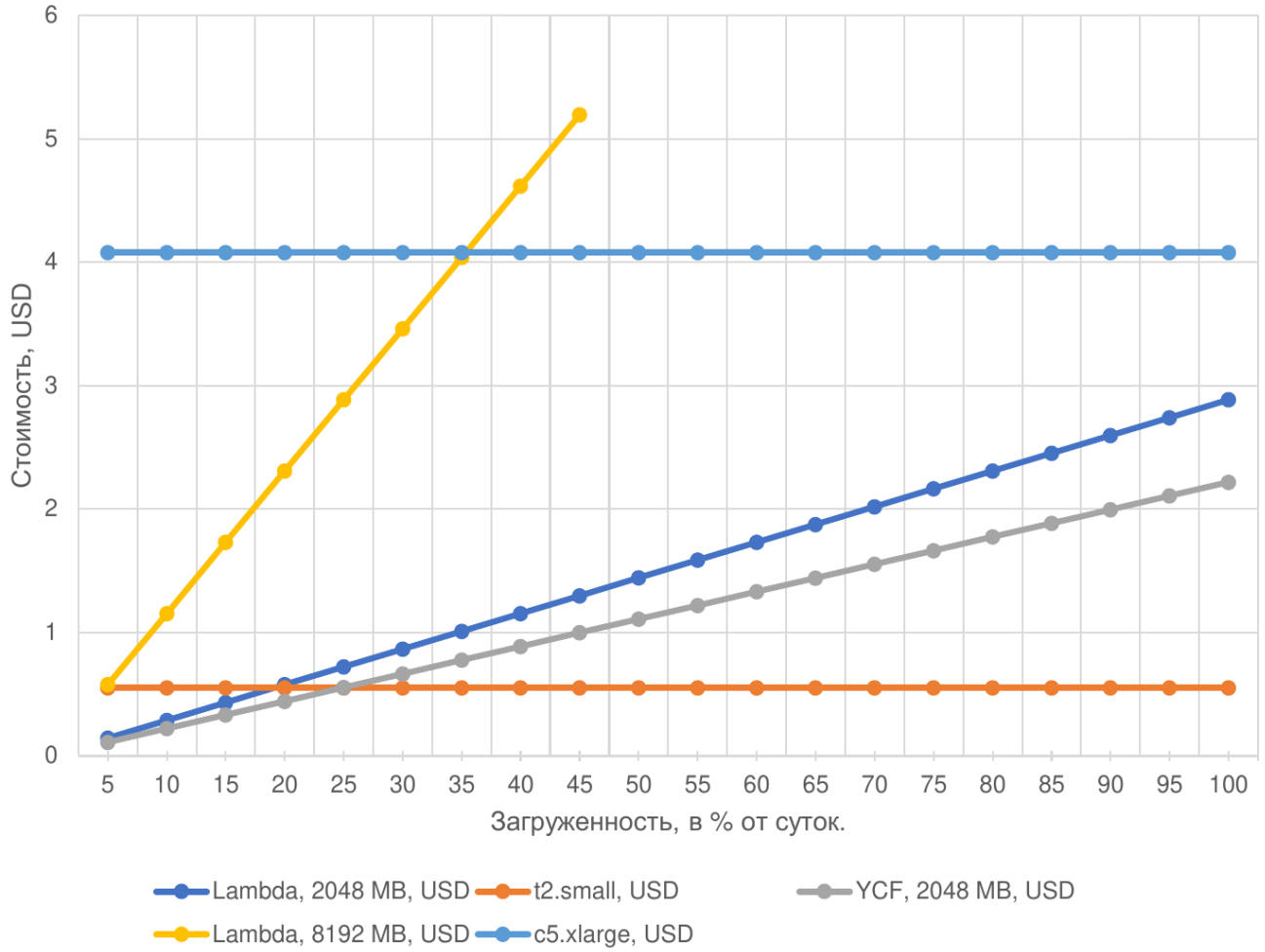
Experiments with FaaS-based Resources

Execution time and cost per task for molecular docking application



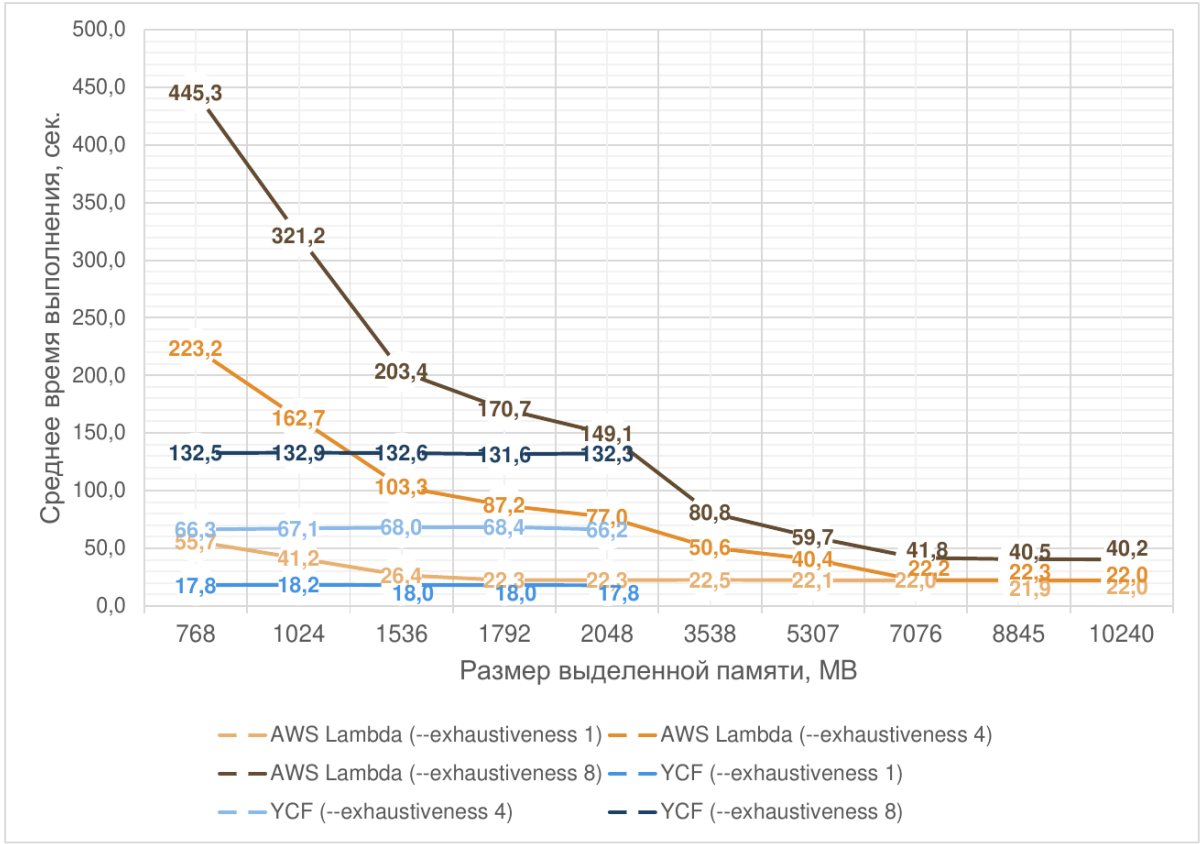
	Cost per 1K tasks	Relative cost
AWS EC2 t2.small (1vCPU 2GB)	$0.53	1
AWS Lambda (1vCPU 2GB)	$2.572	4.84
YCF (1vCPU 2GB)	$1.699	3.2
AWS EC2 c5.xlarge (4vCPU 8GB)	$1.382	2.6
AWS Lambda (4vCPU 8GB)	$2.964	5.58

Cost of Dedicated VM vs Cloud Functions

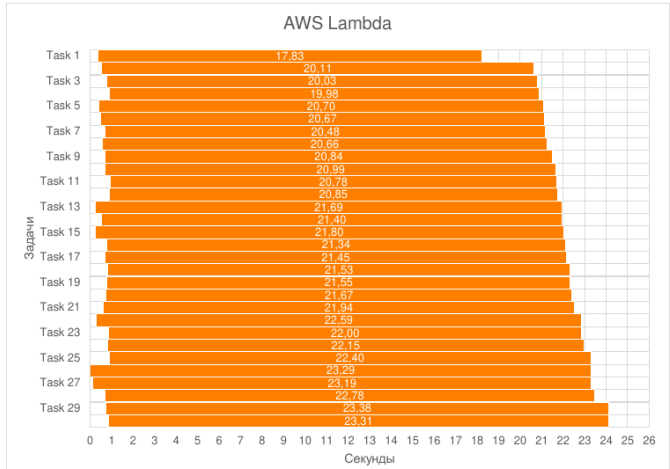


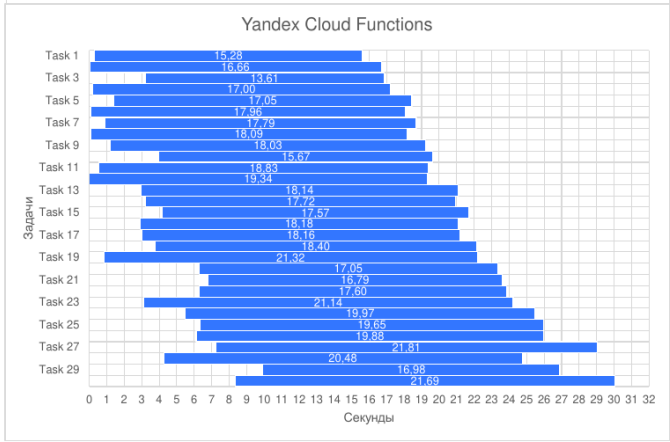
Cloud Function Performance Comparison

Task execution time as a function of allocated memory

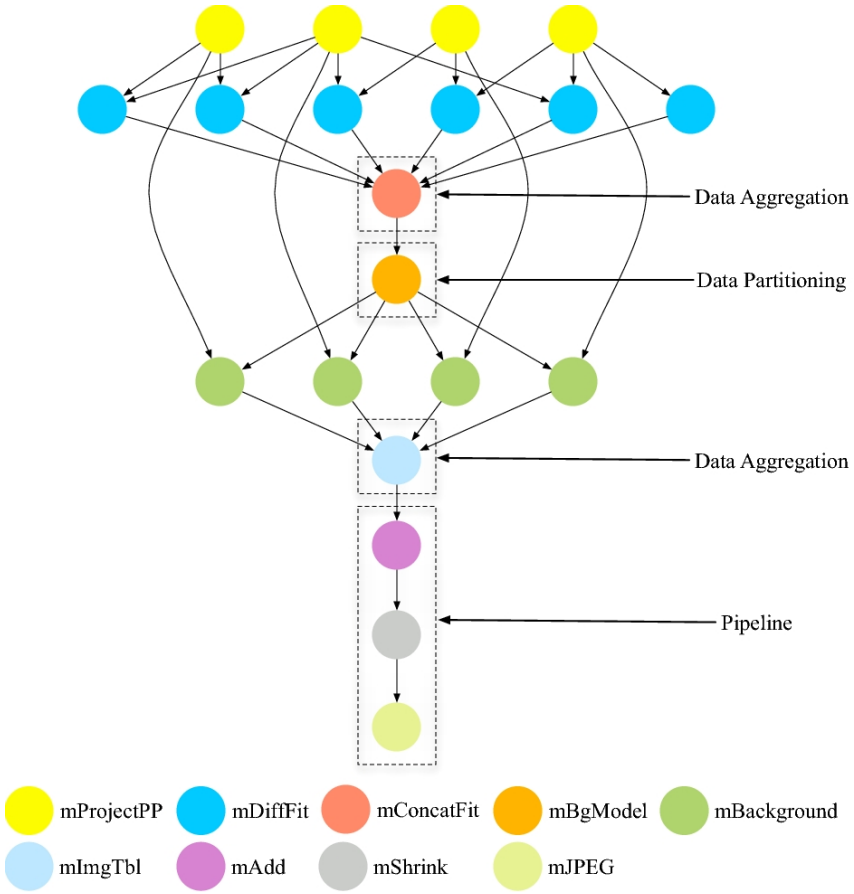


Gantt chart of task startup times





Execution Time of Montage Workflow (100 tasks)



	Montage (no data)	Montage
AWS Lambda (100 slots)	7m 38s	9m 4s
Yandex Cloud Functions (100 slots)	11m 38s	request timeouts
2 HPC Clusters (48+64 slots)	9m 55s	11m 15s

Conclusion

- Everest platform provides unified access to classic and cloud resources
  - can be used for evaluation of different options and implementation of hybrid strategies
- Task execution on spot instances (AWS EC2, Azure)
  - smart instance selection and task scheduling based on budget and time constraints
  - allows to reduce costs by 2-4 times in comparison to on-demand instances
- Task execution on cloud functions (AWS Lambda, Yandex Cloud Functions)
  - fast provisioning (seconds) and elasticity (1000s parallel executions) but with limitations
  - great for short sporadic tasks, not cost-efficient for large batch computations
- Future work
  - implement integration with other cloud services, do extensive evaluation
  - implement and study hybrid scheduling strategies combining different resources