Development and evaluation of a Kubernetes cluster simulator based on Batsim

Presented by: Théo Larue
Supervised by: Olivier Richard & Michael Mercier

Université Grenoble Alpes

August 31, 2020

Table of contents

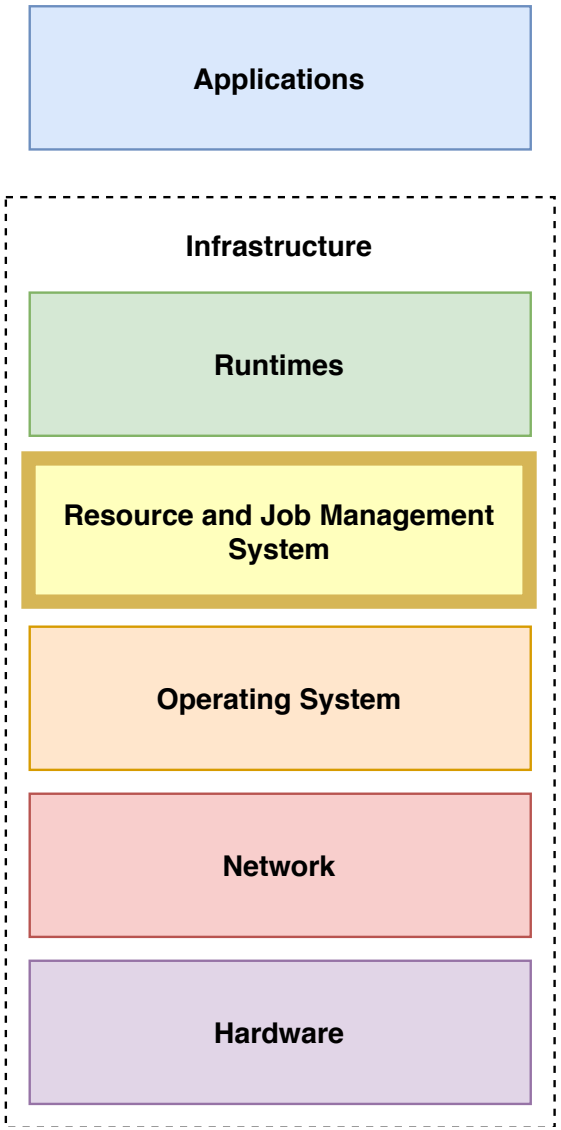
Introduction

Resource and Jobs Management System

The RJMS is at the core of the cluster.

Examples of RJMS

- OAR
- SLURM
- HadoopYARN
- Apache Mesos

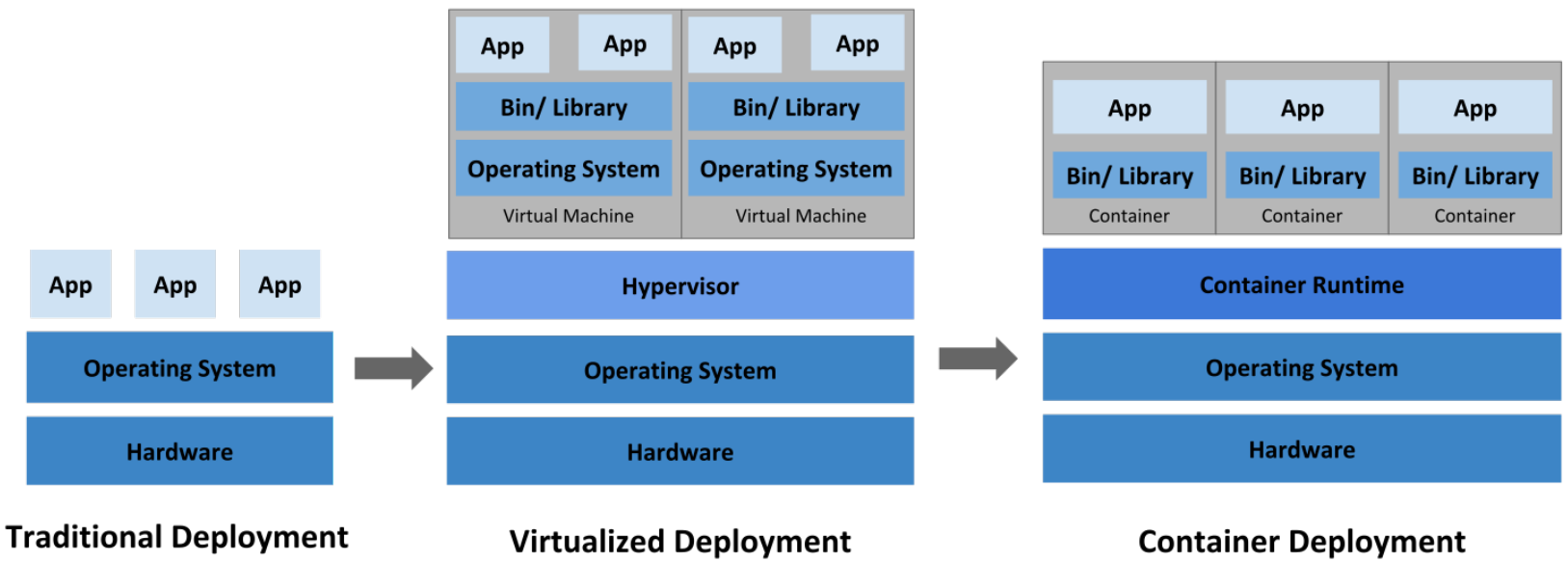


Kubernetes

Kubernetes in a nutshell

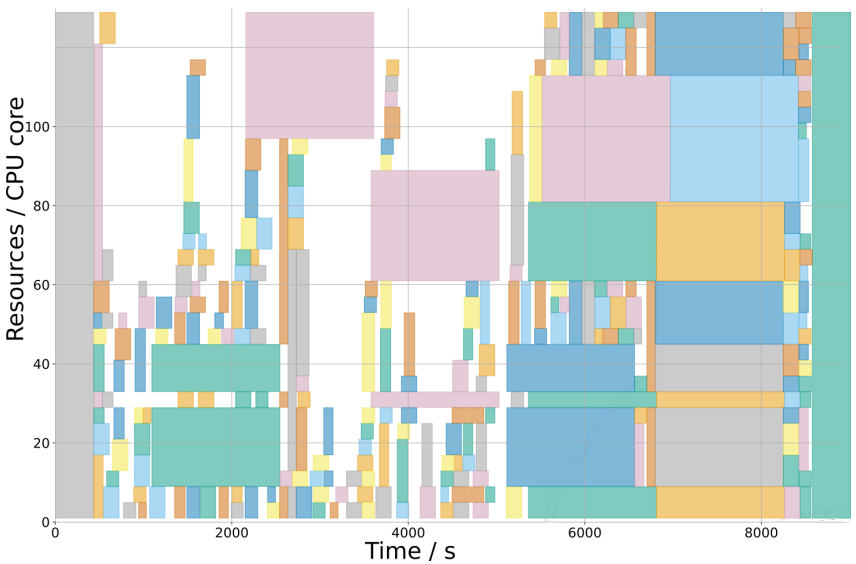
- Open source resource manager for containerized applications
- About 2M lines of code
- 2.8k contributors

kubernetes



source: `https://kubernetes.io/docs/`

A component of the RJMS: the scheduler



Scheduling is the act of allocating tasks to resources.

Numerous factors

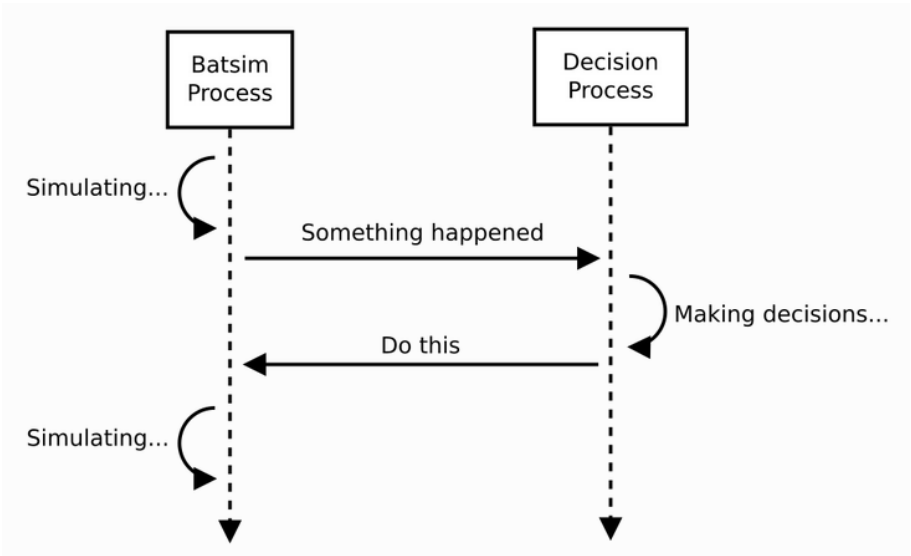
- Workloads
- Applications
- System size
- Network topology
- Energy consumption
- Scheduling policies

Complex implementations: Kubernetes default scheduler weighs **47k lines of code**.

Studying RJMS

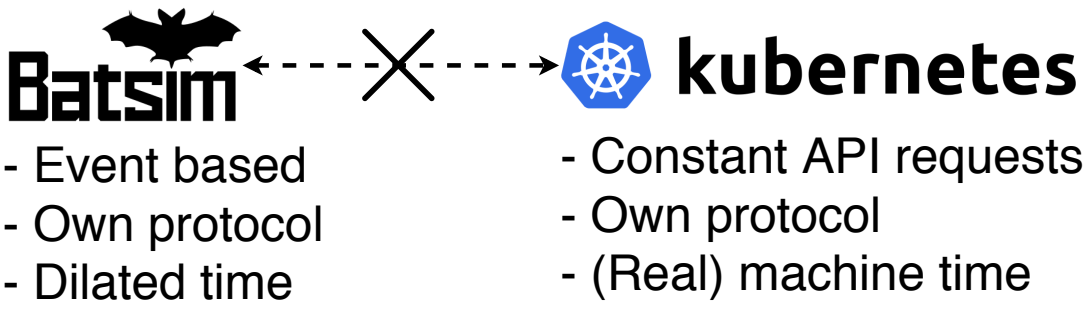
Different approaches

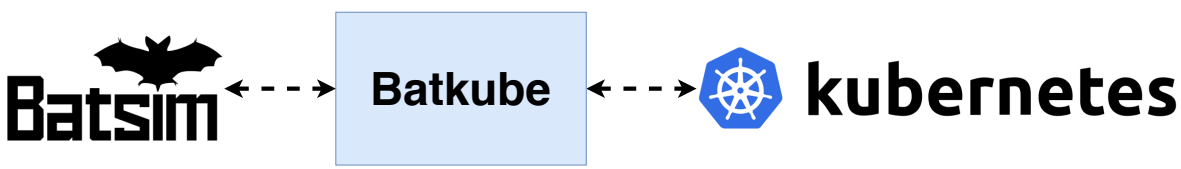
- Analytical study
- Real experiments
- Emulation
- **Simulation**



Batsim, an infrastructure simulator aimed at studying RJMS.

Contribution





Batkube supports

- Any Go scheduler
- Any cluster size
- Resource requests
- Non parallel tasks

Literature review

Infrastructure simulators

	Grid	HPC	Cloud	P2P	Volunteer
SimGrid	✓				
GridSim	✓				
LogGOPSim		✓			
BigSim		✓			
CloudSim			✓		
GroudSim	✓		✓		
PeerSim				✓	
OverSim				✓	
SimBA					✓
SimBOINC					✓

Domain specific simulators

SimGrid

- Framework for building simulators
- Versatile, accurate and scalable
- 20 years of experience
- Simple analytical models

Simulators for the study of RJMS

Often *ad hoc* simulators

"Publish and perish" - Milian Poquet

Some active projects

- YARNSim
- SLURM simulator

	Scheduler	Platform	Job model
Accasim	Internal	Ad hoc	Static duration
Alea	Internal	GridSim	Static duration
Batsim	Custom Protocol	SimGrid	SimGrid models

Simulation of RJMS

Kubernetes cluster simulation

Kubernetes simulation projects

	joySim	k8s-cluster-simulator
Origin	private (JD.com)	student project
Availability	closed source	open source
Focus	service	batch processing
Scheduler	any	user implementation
Models	mock nodes	static job durations
Capabilities	fully fledged simulator	time dilation
	monitoring tools	raw metrics

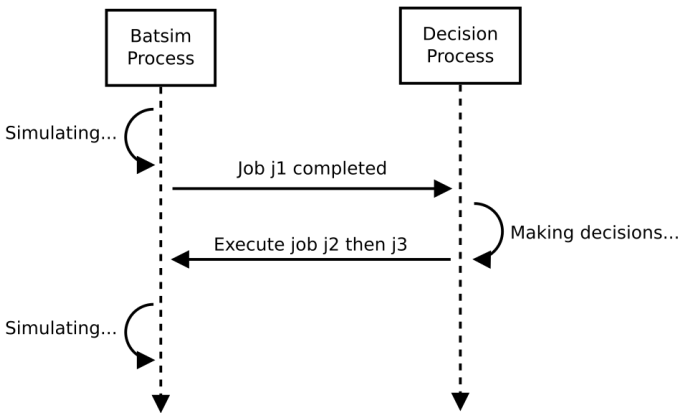
Notable Kubernetes schedulers

- kube-batch

- Poseidon (Firmament)
- **kube-scheduler**

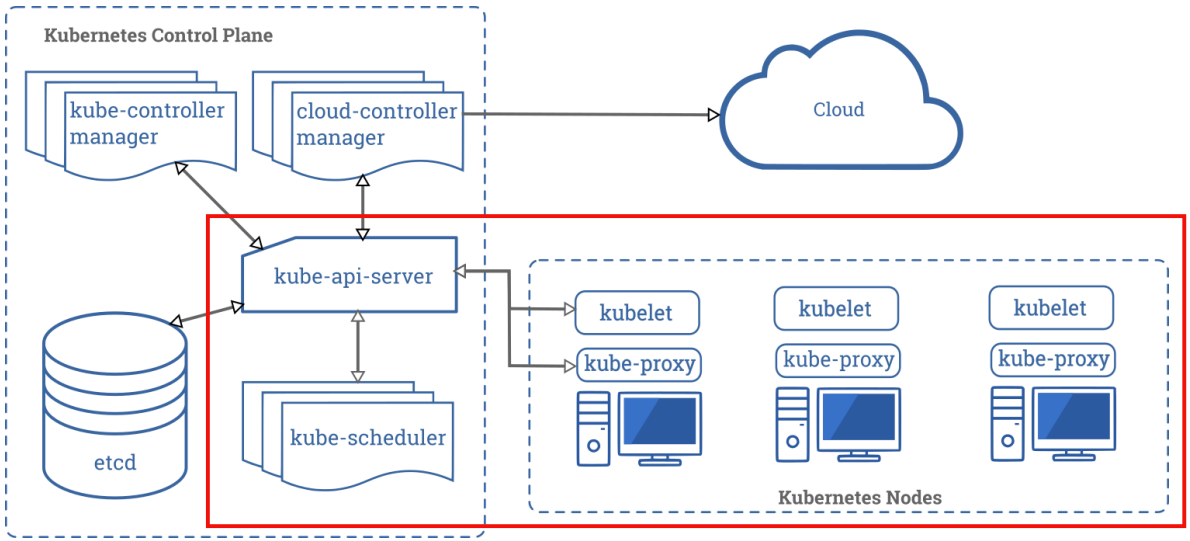
Integrating Kubernetes schedulers to Batsim

Different communication paradigms



source: `https://batsim.readthedocs.io`

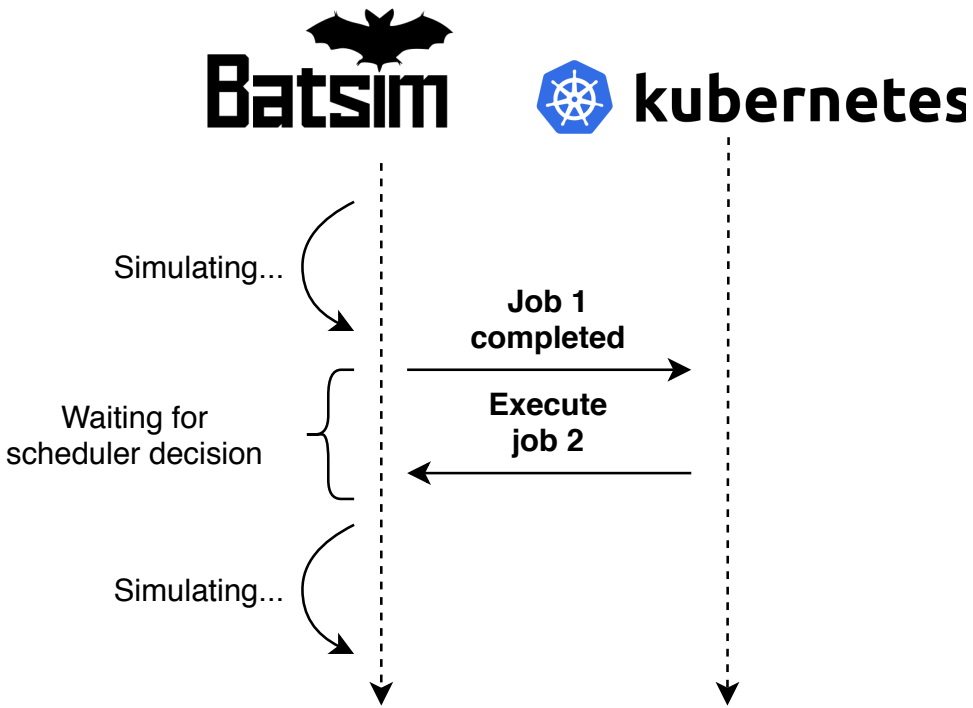
- Event based
- Simulation time



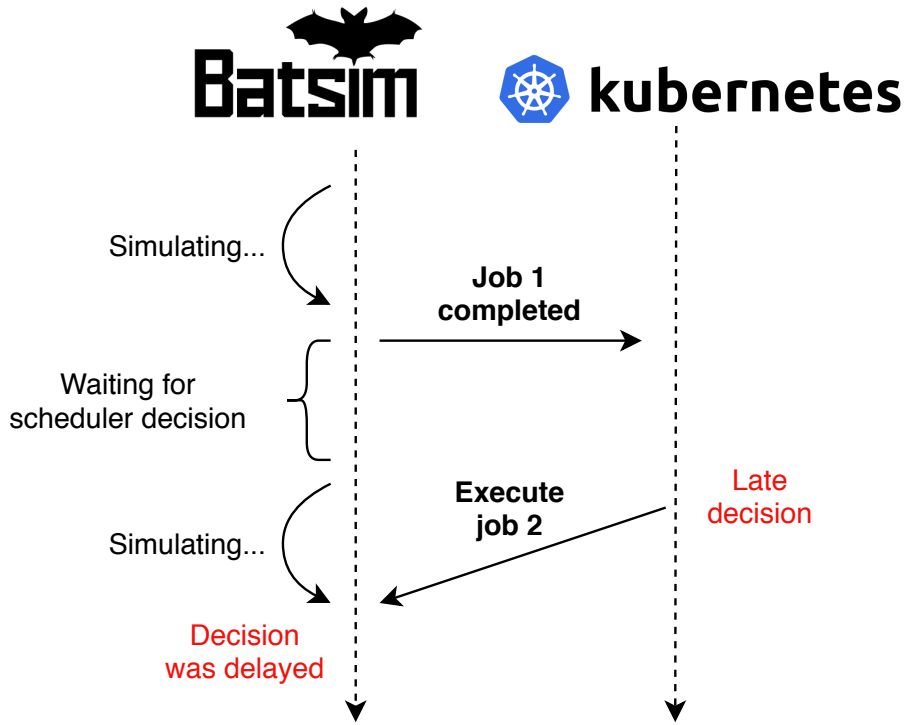
source: `https://kubernetes.io/docs/concepts/overview/components/`

- Central API
- Real time

Time synchronization



Scenario 1: correct synchronization



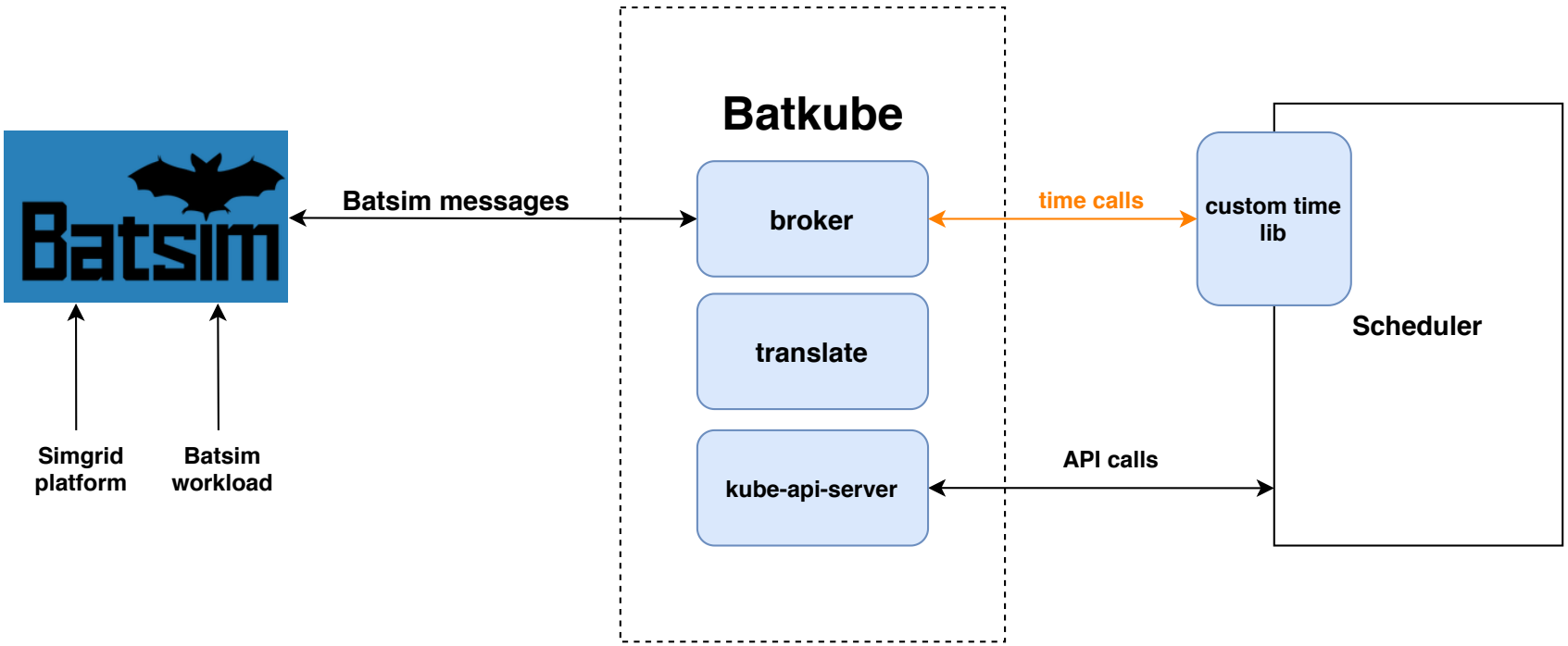
Scenario 2: delayed decision

Technical challenges

Challenges to tackle

1. Integration with Kubernetes
2. Scheduler time interception
3. Time synchronization

Architeture of Batkube

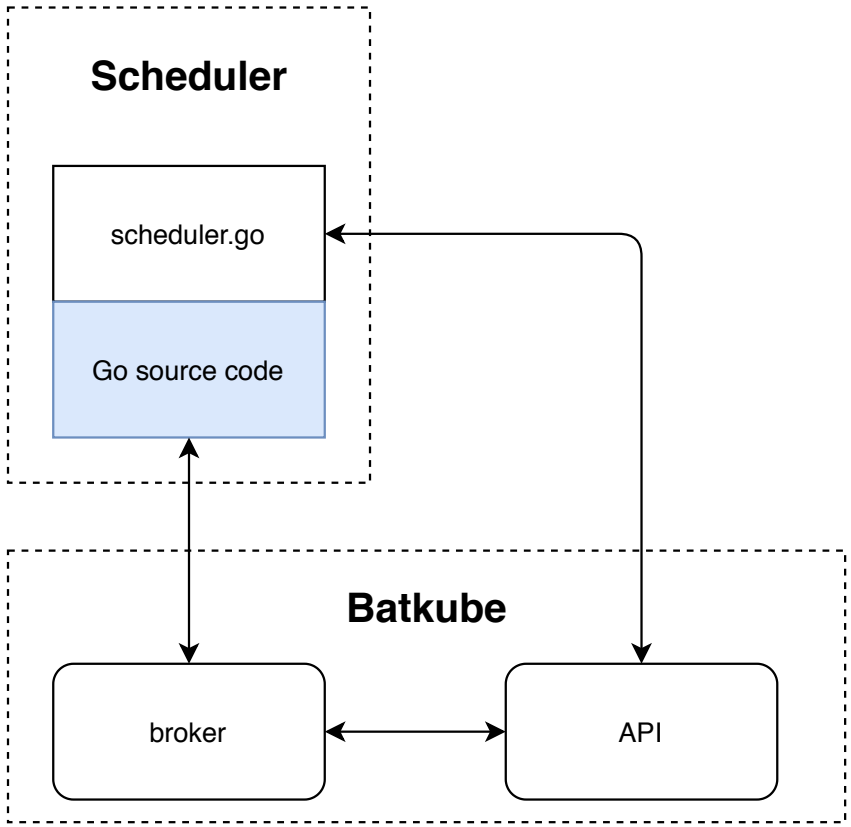


Global architecture of Batkube.

Time interception

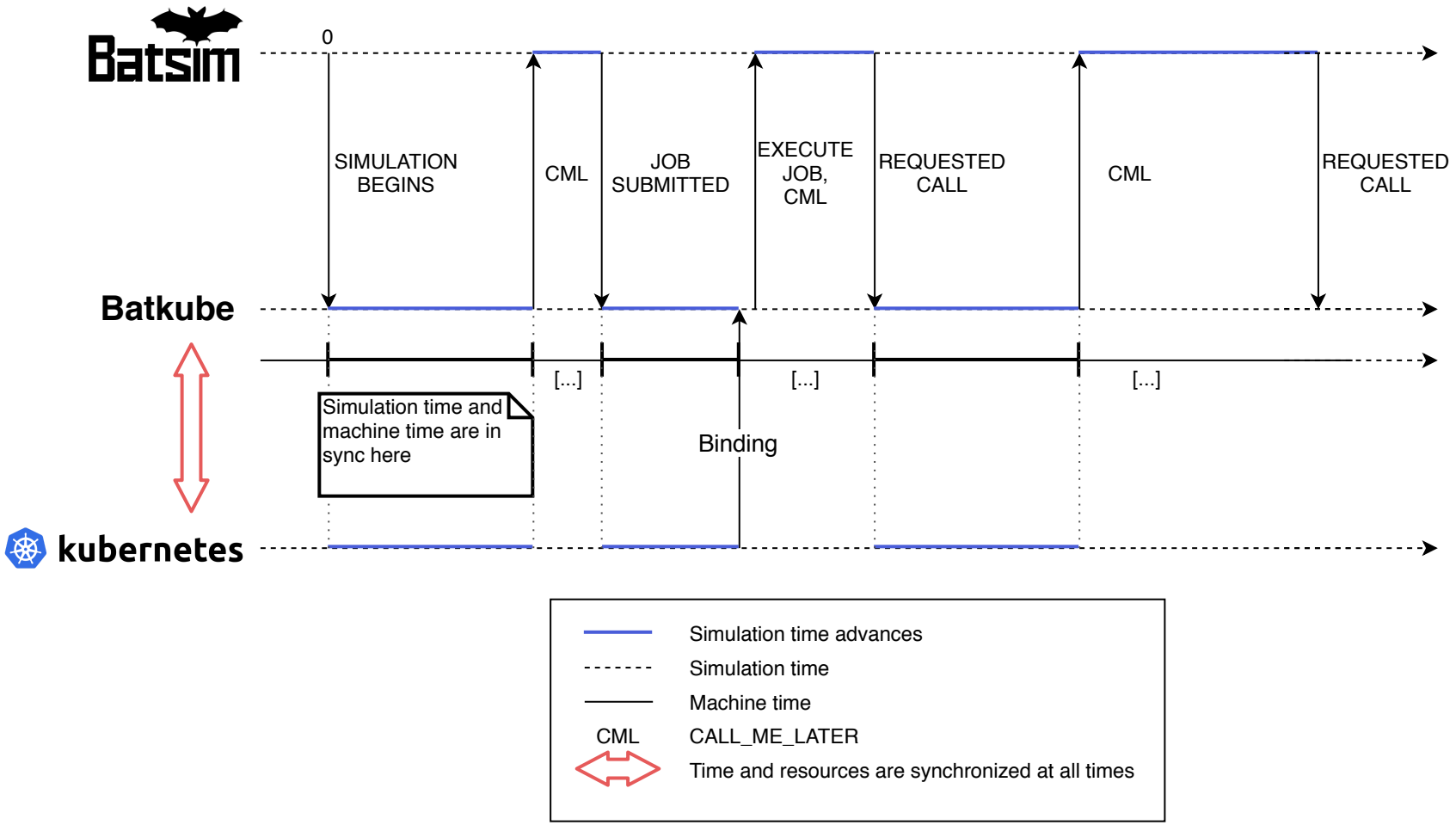
Redirection of time calls

- Specific functions are redirected
- Automatic source code manipulation using AST
- Ensured compatibility with the rest of the code



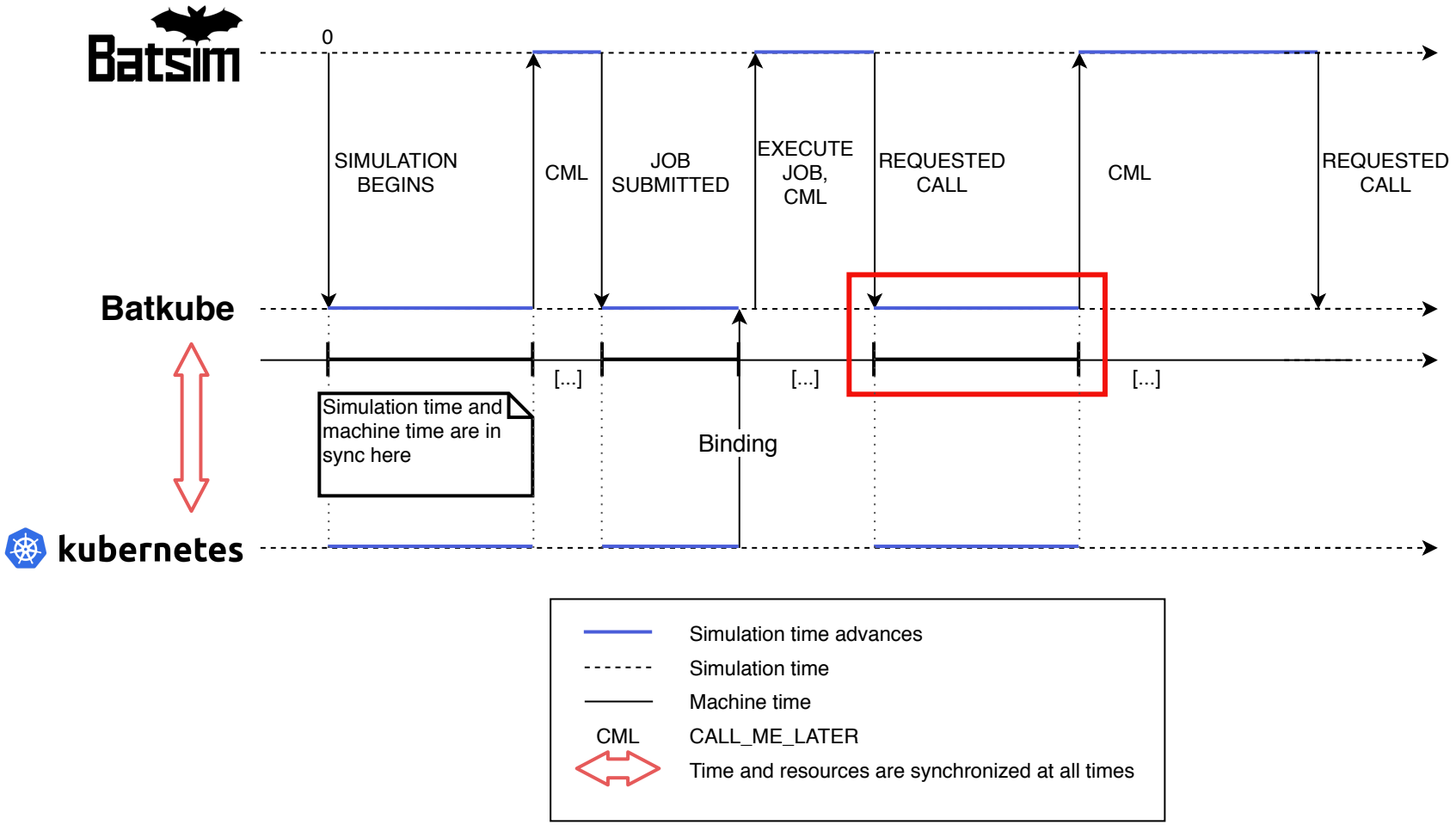
Schedulers are patched to redirect their time.

Time synchronization



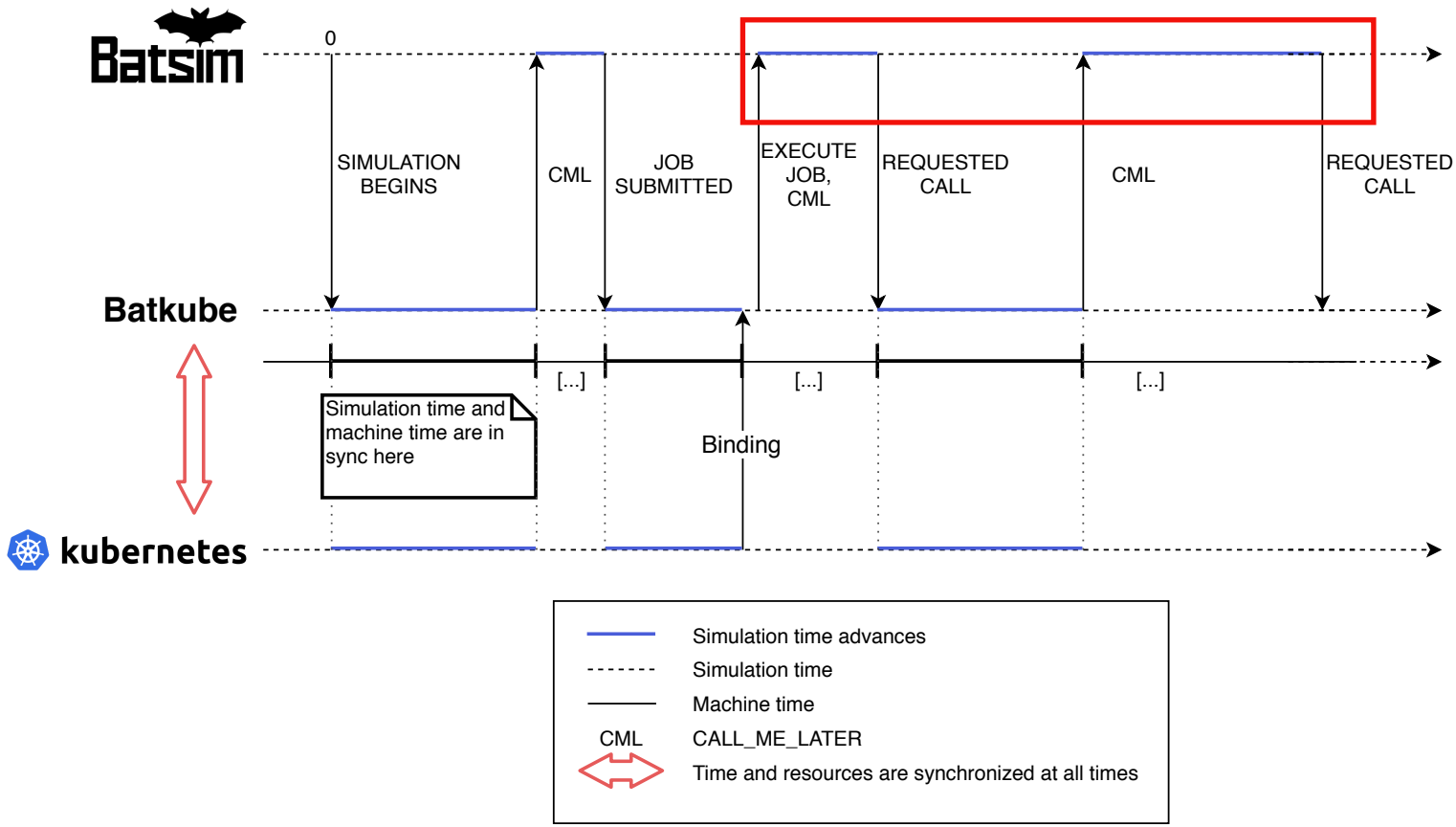
Time synchronization between Batsim and the scheduler

Parameters of the synchronization I



Timeout value

Parameters of the synchronization II



Simulation time step $\in$ [`base-simulation-timestep`, `max-simulation-timestep`]

Multiplying factor: `backoff-multiplier` (default $= 2$)

Study of the simulator

Study of the simulator parameters

Scheduler kube-scheduler

Workloads

- *burst*: 200x170s, 1 cpu, at time zero
- *spaced*: 200x170s, 1 cpu, every ten seconds
- *realistic*: 49 jobs, between 0 and 6 cpu, between 0 and 1 hour duration, extracted from the KIT ForHLR II system.

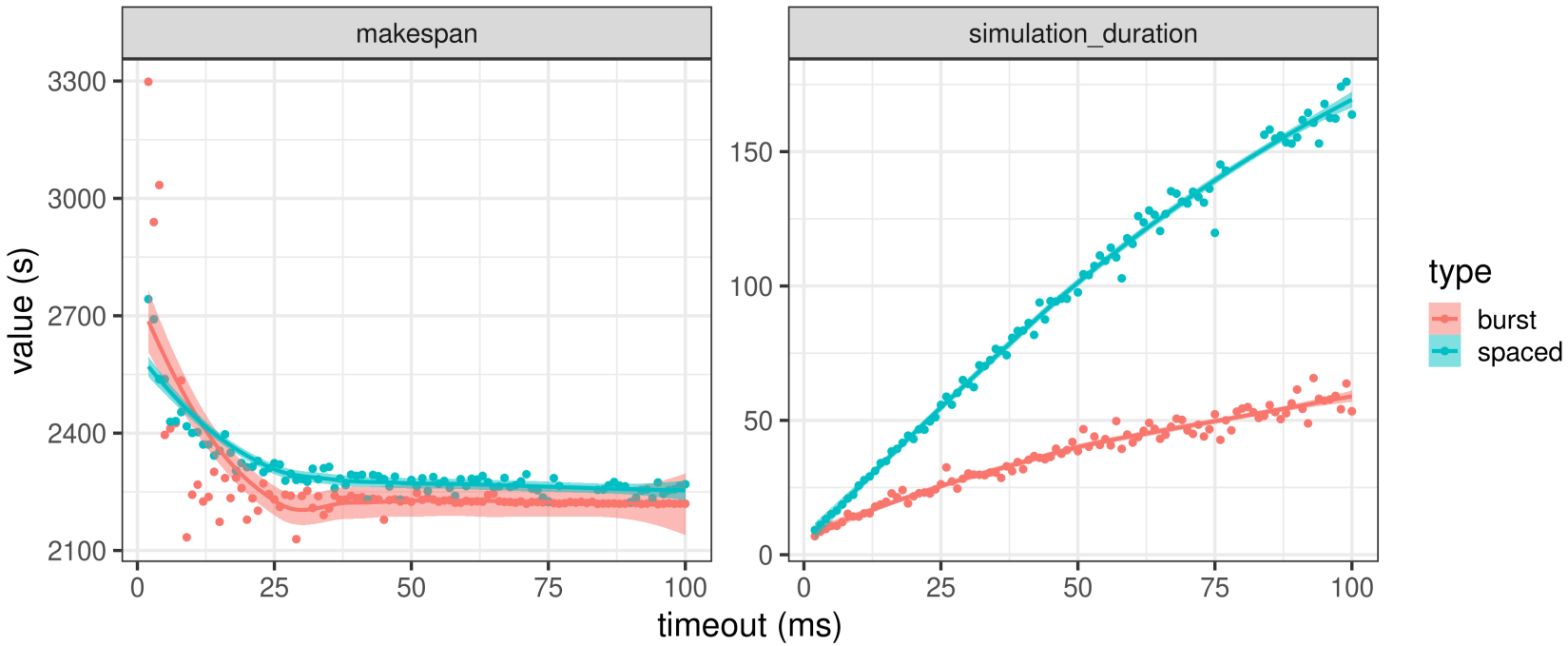
Platforms

- *burst* and *spaced*: 16 nodes x 1 cpu
- *realistic*: 1 node x 6 cpu

Metrics

- Makespan = simulated length of the simulation
- Simulation duration = real execution time

Timeout



`max-simulation-timestep = 20s`

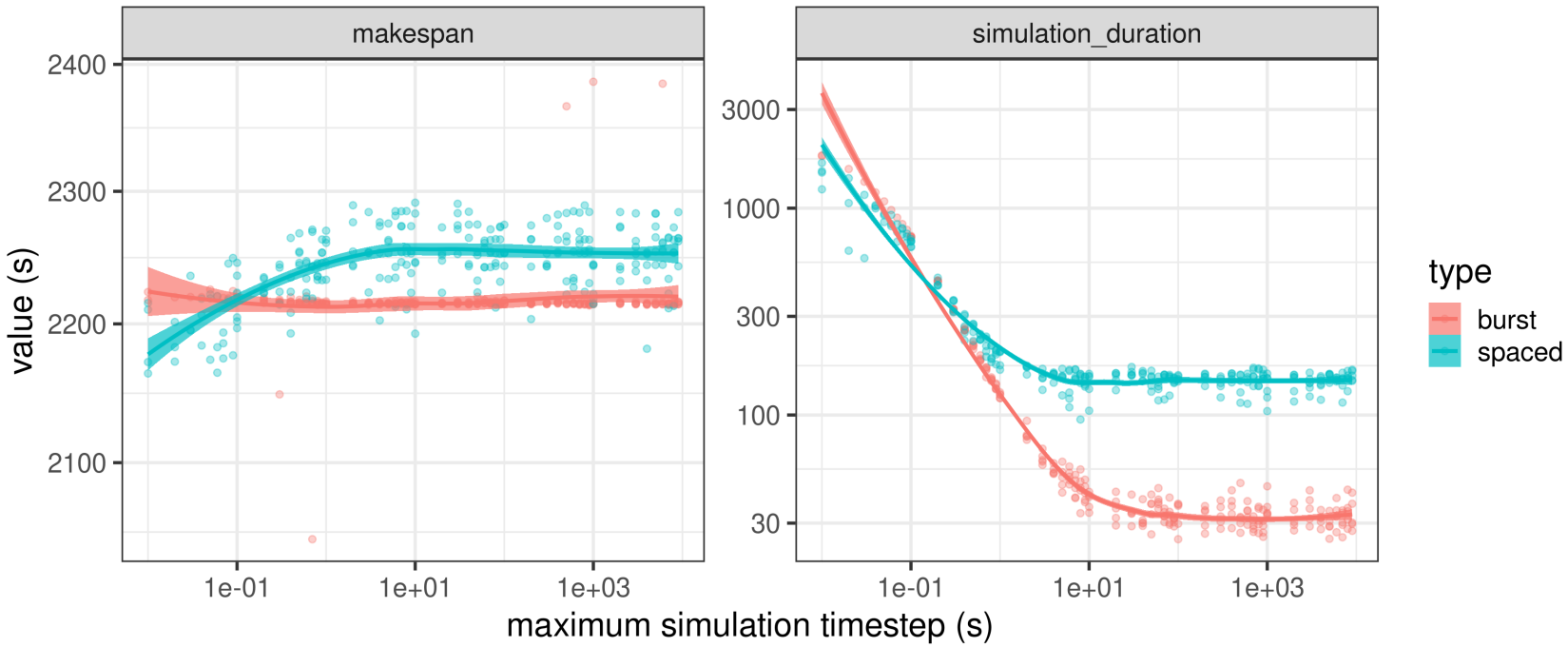
`base-simulation-timestep = 100ms`

`backoff-multiplier = 2`

→ Critical value in the **accuracy** / **scalability** tradeoff

→ An optimal value can be measured

Maximum simulation timestep



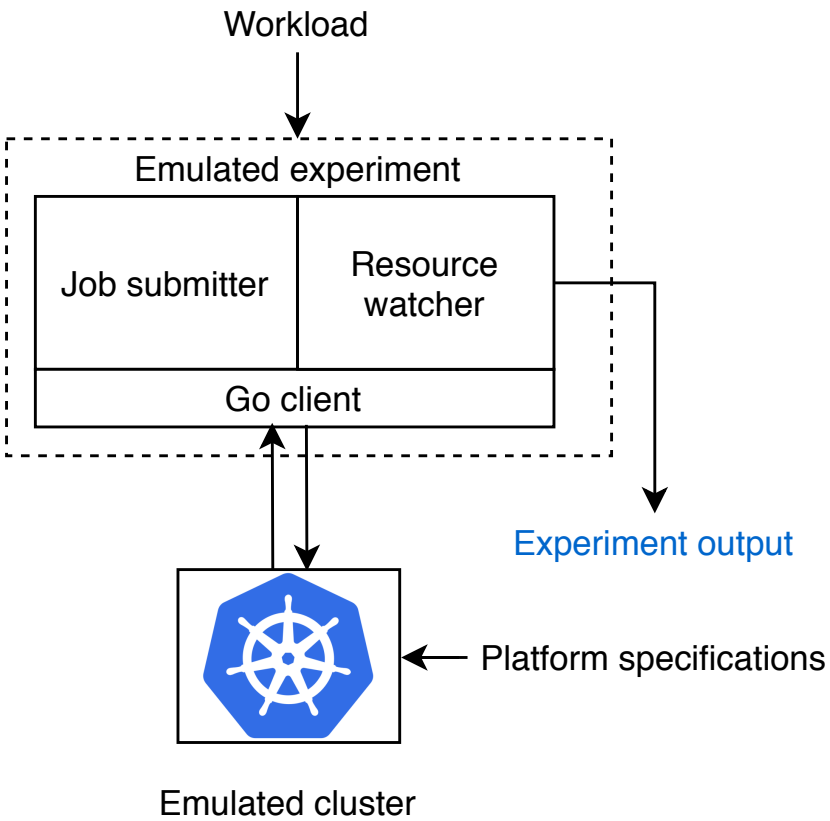
`timeout-value = 50ms`

`base-simulation-timestep = 100ms`

`backoff-multiplier = 2`

→ Experiments too simple to draw any conclusion (only decision = schedule a job)

Experimentation on a real cluster



Workloads are run 10 times each (except *realistic*)

workload	**makespan**			
	emulated		simulated	
	μ	σ	μ	σ
burst	2467	28.3	2215 (-252)	0.508
spaced	2468	5.14	2257 (-211)	16.9
realistic	32556	-	32555 (-1)	1.30

workload	**mean waiting time**			
	emulated		simulated	
	μ	σ	μ	σ
burst	1077	10.6	970 (-107)	12.6
spaced	146	1.67	48.1 (-97.9)	9.44
realistic	2884	-	2020 (-864)	950

Causes to this deviation

- Scheduler over allocating when simulated
- Incomplete simulation models

Discussion and future work

Capabilities and limitations of Batkube

Capabilities

- Can patch **any kubernetes scheduler** written in Go **without any modification**
- Static duration model for jobs
- Cpu and memory requests
- Supports the default scheduler

Limitations

- Incomplete models
- Some misbehavior of the scheduler
- Not scalable

Perspectives for future work

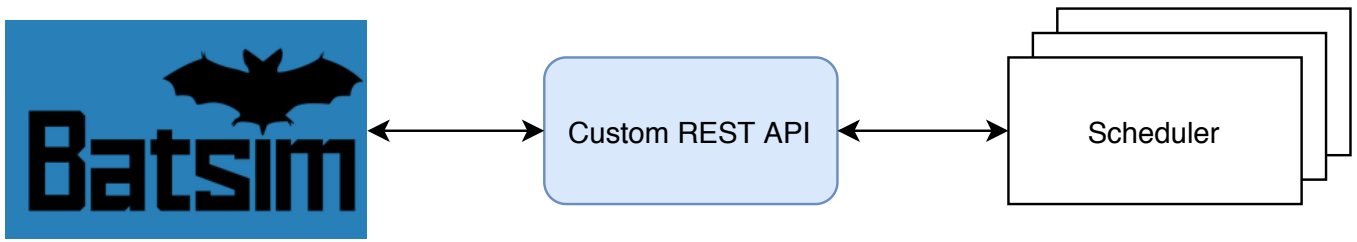
- More complete models for Kubernetes resources
- Support for IO intensive jobs
- Support for parallel jobs
- Support for other schedulers
- More extensive experiments to work on scalability

References I

Any questions?

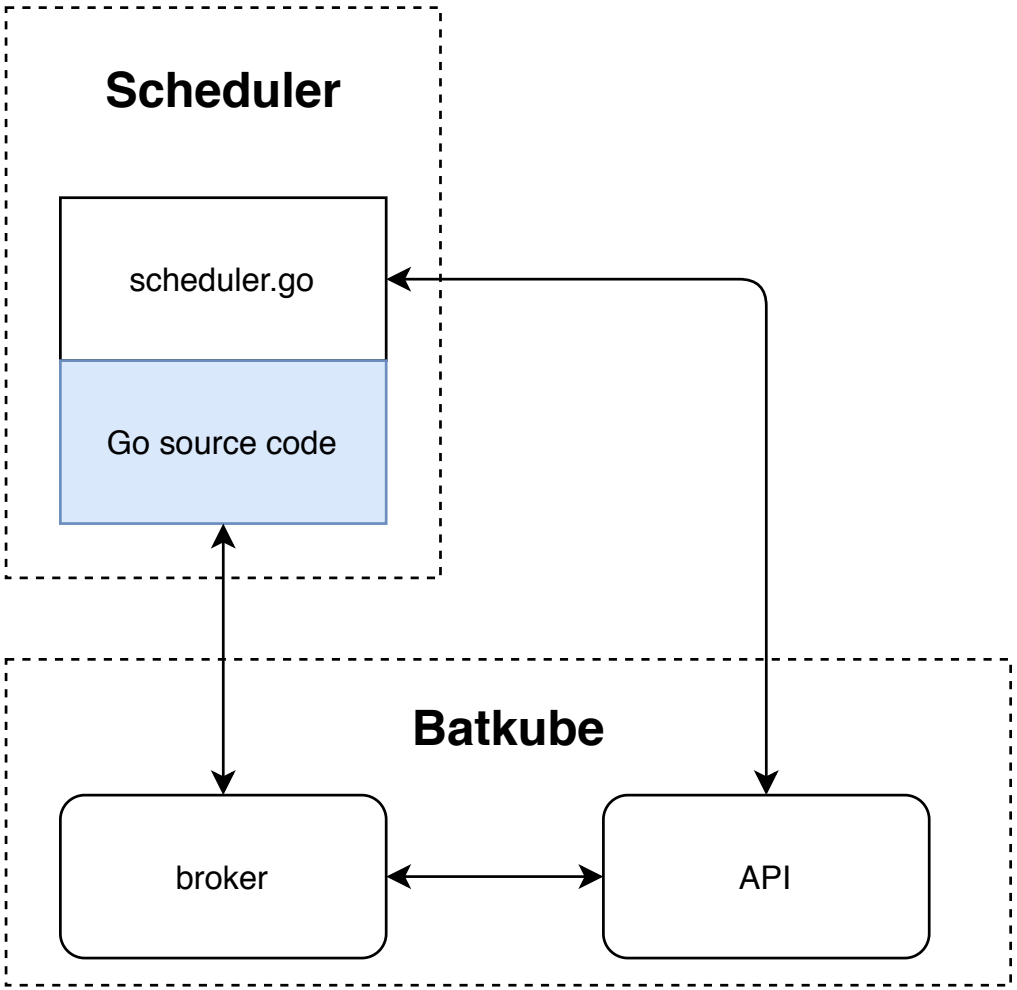
Thank you for your attention!
I am open to any questions.

Batkube integration with Kubernetes



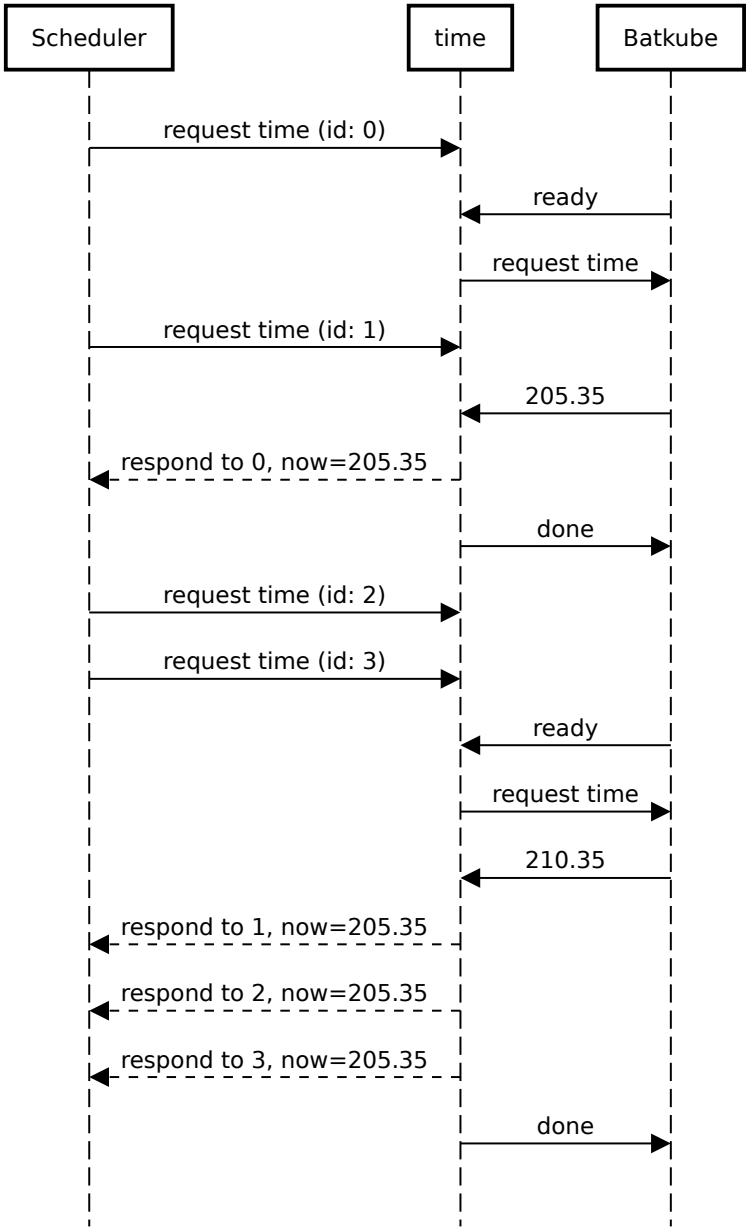
Reimplementation of a custom API.

Time interception



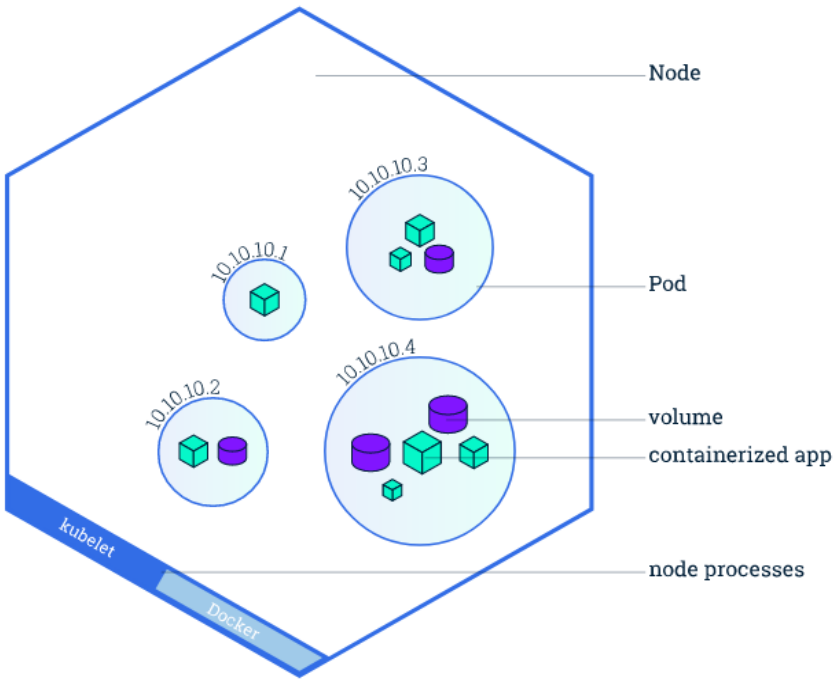
Schedulers are patched to redirect their time.

batsky-go



Exchanges between the scheduler, batsky-go (“time”) and Batsim

Similar resources



source: `https://kubernetes.io/docs/tutorials/kubernetes-basics/explore/explore-intro/`

Translation between Kubernetes and Batsim

- A Pod = a job.
- A Node = a compute resource.

TODO: results for realistic workload