

# 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



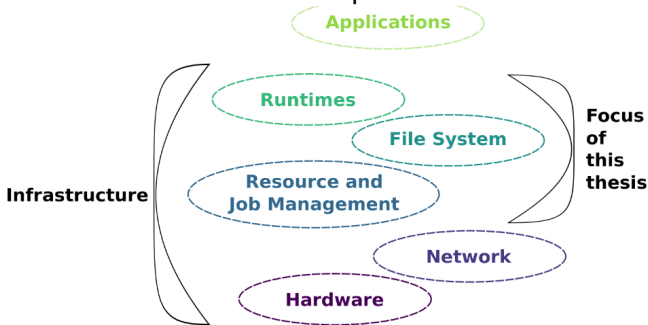
# Table of contents

- 1 Introduction
- 2 Literature review
- 3 Integrating Kubernetes schedulers to Batsim
- 4 Study of the simulator
- 5 Discussion and future work

# Introduction

# Computer infrastructure

TODO: definition of a computer infrastructure



Placeholder (from michael thesis). TODO: my own illustration

# Resource and Jobs Management System

TODO: the RJMS is at the core of the hpc cluster.  
def or diagram illustrating a RJMS (without much details).  
Show that the scheduler is part of the RJMS.  
Examples: OAR, Apache Mesos, hadoop YARN, SLURM

# Kubernetes

Explain containers real quick.

Container orchestration software, description based.

# Studying computer infrastructures

## Objects of studies

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

## How ?

- Analytical study
- Real experiments
- Emulation
- Simulation

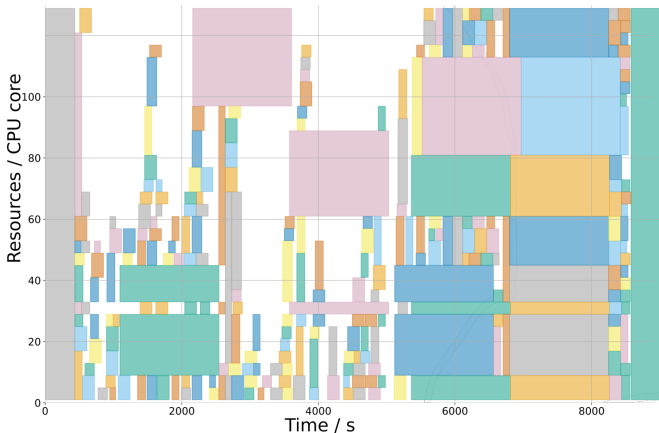
# Batsim

Aimed at studying RJMS.

Strong decoupling decision process / simulator.



# The scheduling problem



**Scheduling** is the act of allocating tasks to resources.

# Contribution

TODO: Our contribution is Batkube, an interface between Batsim and Kubernetes schedulers

# Literature review

# Domain specific simulators

refs on domain specific simulators

# Software specific simulators

YARNSim, SLURM simulator

# Publication specific simulators

“Publish and perish” - Milian Poquet

# SimGrid

SimGrid: Versatile, scalable, accurate.

Cpu = a computation speed.

Storage = a seek time and a data transfert rate.

Network = a flow model, modeling bandwidth sharing behaviors.

Simple models but thoroughly validated.

# Related work

## GridSim

Alea: modular, extensible.

Accasim: supports additional information (temperature, power consumption). Very efficient in terms of simulation time and memory usage.



# Kubernetes cluster simulation

k8s-cluster-simulator: open source, student project, delay jobs.

Schedulers provided via a Go interface.

joySim: closed-source, fully fledged kubernetes cluster simulator, service oriented (mock nodes).

# Integrating Kubernetes schedulers to Batsim

# Technical challenges

## Challenges to tackle

- 1 Integration with Kubernetes.

# Technical challenges

## Challenges to tackle

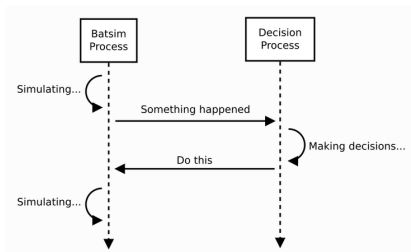
- 1 Integration with Kubernetes.
- 2 Intercepting scheduler time.

# Technical challenges

## Challenges to tackle

- 1 Integration with Kubernetes.
- 2 Intercepting scheduler time.
- 3 Time synchronization between Batsim and the scheduler.

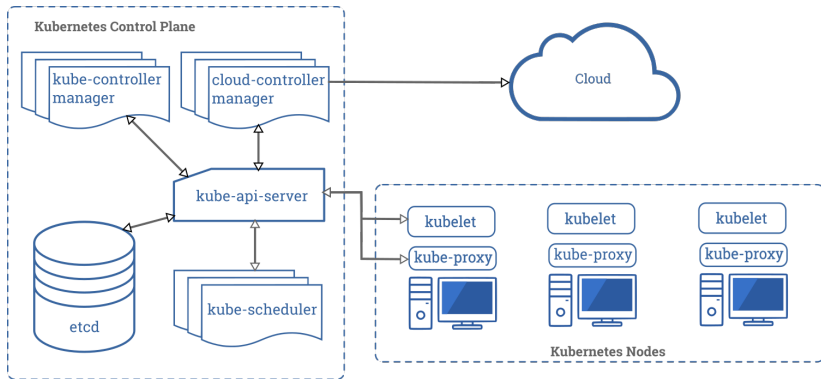
# Batsim concepts



source <https://batsim.readthedocs.io>

Batsim events and protocol.  
User defined workloads.  
(insert json examples?)

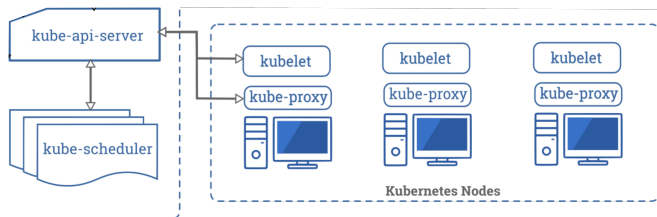
# Kubernetes concepts



source: <https://kubernetes.io/docs/concepts/overview/components/>

Kubernetes components.

# Kubernetes concepts



source: <https://kubernetes.io/docs/concepts/overview/components/>

Kubernetes components.



# Different paradigms

Batsim: event based, simulation time.

Kubernetes scheduler: asynchronous calls to the API, machine time.

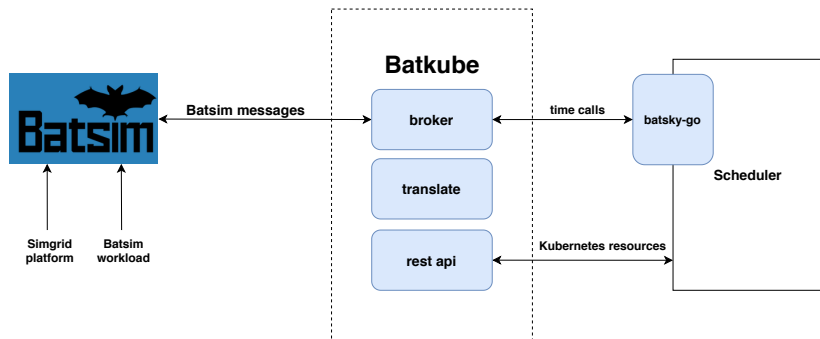
The goal is to make the scheduler event based and relying on simulation time for Batsim, and make Batsim a kube-api-server to the scheduler.

# Batkube integration with Kubernetes



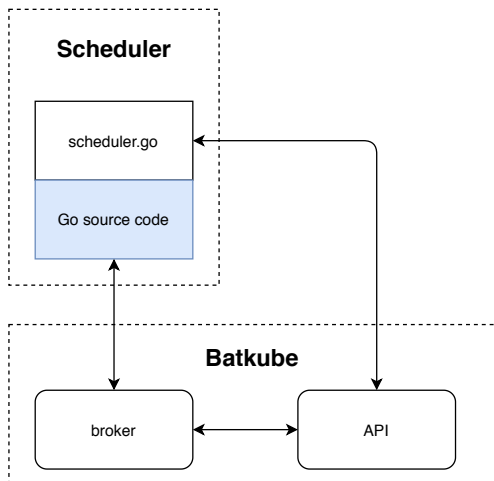
Reimplementation of a custom API.

# Architecture of Batkube



## Global architecture of Batkube.

# Time interception

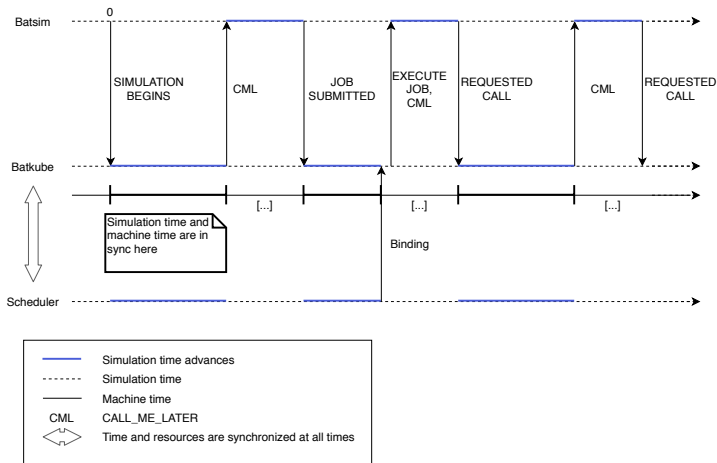


Schedulers are patched to redirect their time.

# Time synchronization I

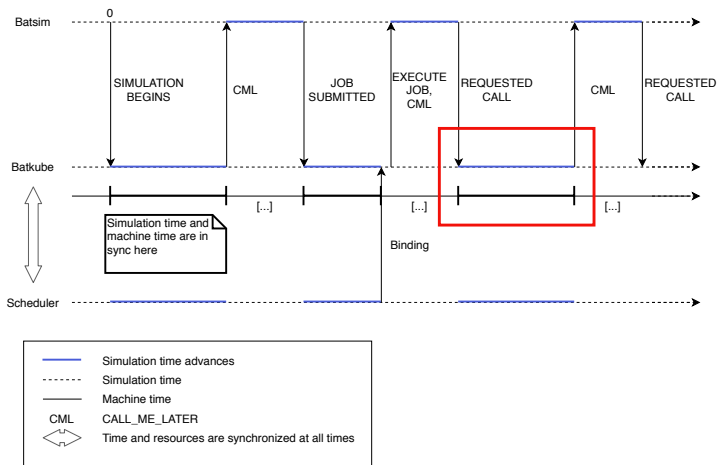
TODO: explain CML

# Time synchronization II



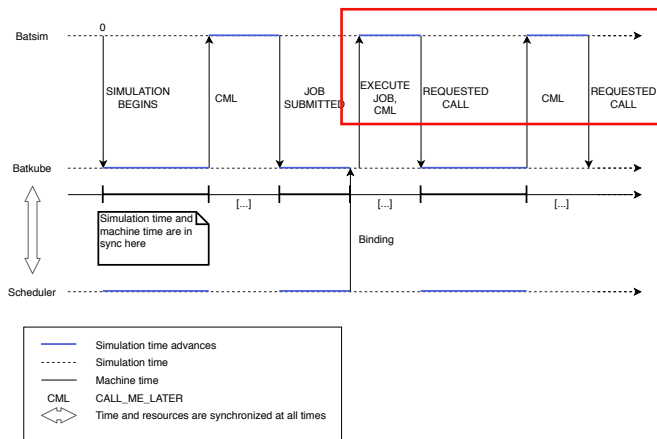
Time synchronization between Batsim and the scheduler

# Parameters of the synchronization I



Timeout value

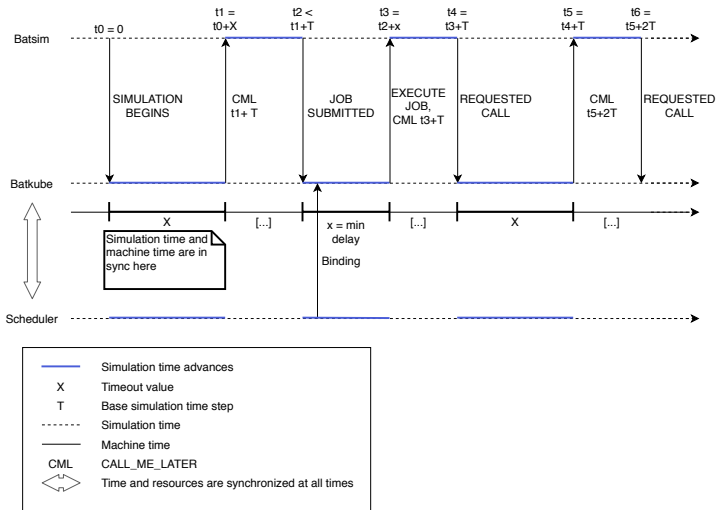
# Parameters of the synchronization II



Simulation time step  $\in [\text{base-simulation-timestep}, \text{max-simulation-timestep}]$



# Time synchronization breakdown



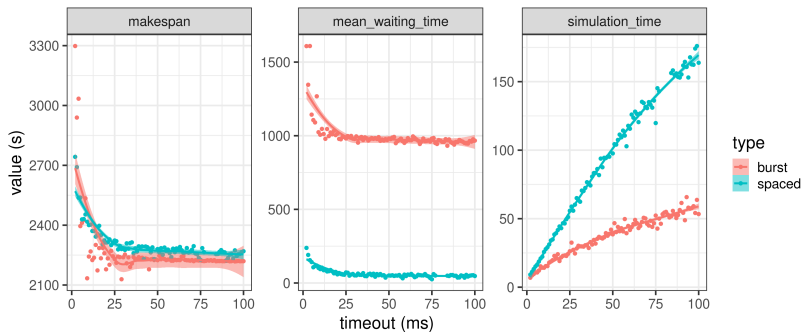
## Time synchronization between Batsim and the scheduler

## Study of the simulator

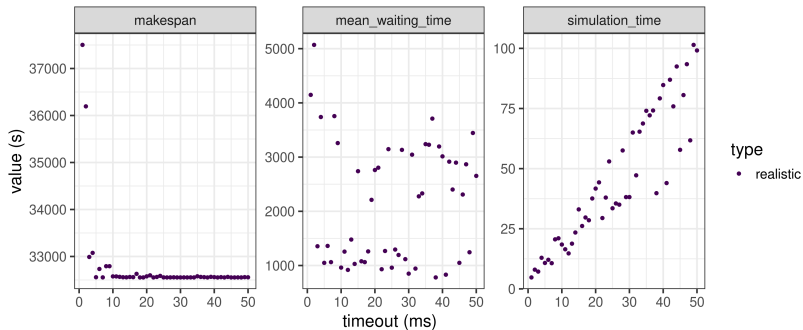
# Experimental design

TODO: Scheduler used, platforms and workloads tested, what experiments (parameters, metrics studied, repetitions)

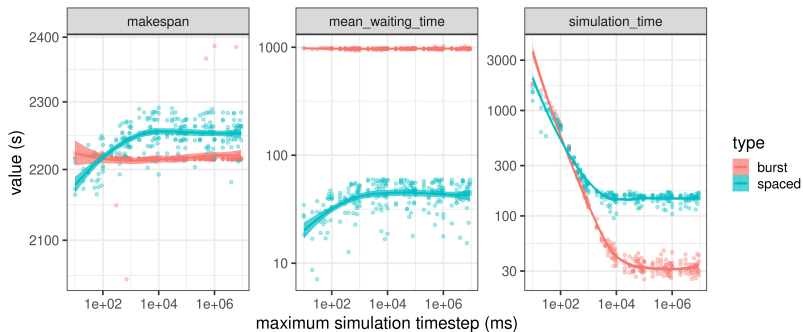
# Timeout I



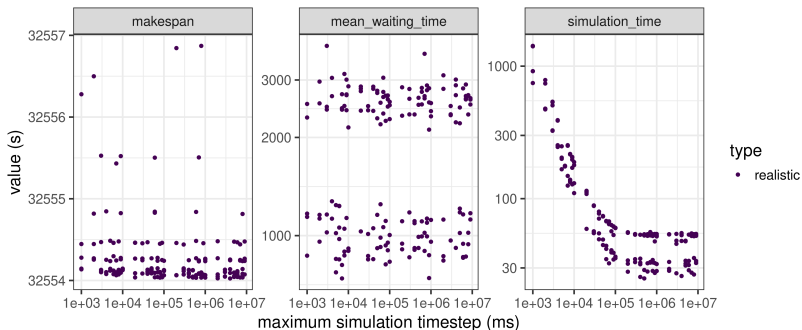
# Timeout II



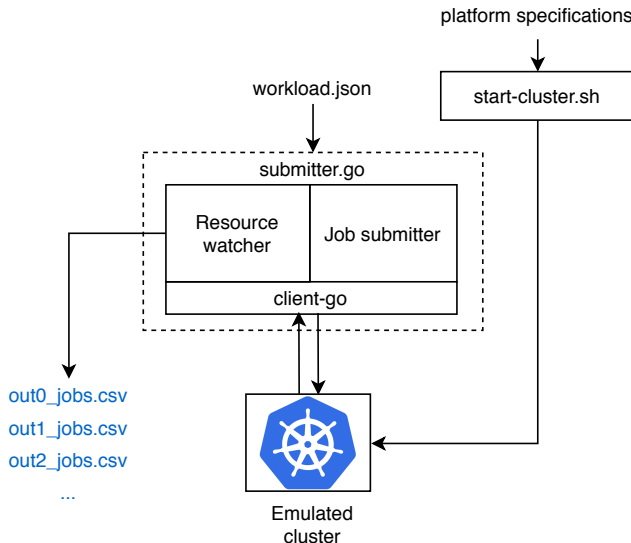
# Maximum simulation timestep I



# Maximum simulation timestep II



# Experimentation on a real cluster





# Deviation with reality

workload	makespan				mean waiting time			
	emulated		simulated		emulated		simulated	
	$\mu$	$\sigma$	$\mu$	$\sigma$	$\mu$	$\sigma$	$\mu$	$\sigma$
burst	2467	28.3	2215 (-252)	0.508	1077	10.6	970 (-107)	12.6
spaced	2468	5.14	2257 (-211)	16.9	146	1.67	48.1 (-97.9)	9.44
realistic	32556	-	32555 (-1)	1.30	2884	-	2020 (-864)	950

# Conclusion

Deviation with reality: can be fixed with some work on the api. Need experiments to measure and quantify this deviation.

max timestep: studying max timestep alone is not enough, need to study it with backoff multiplier.

base time step: need an experiment on it. Too much importance was credited to max timestep, the base timestep might have importance.

## Discussion and future work

# Capabilities and limitations of Batkube

## Capabilities

- Delay jobs
- Cpu and memory requests
- Can patch any kubernetes scheduler written in Go
- The api only supports the default scheduler

## Limitations

- Memory hungry (in fact, the scheduler is memory hungry)
- Some problems with the scheduler
- Not scalable

# Perspectives for future work

- parallel jobs
- storage
- more complete api: support for more schedulers but also tools (monitoring tools)

# References I