RUTGERS

THE STATE UNIVERSITY
OF NEW JERSEY

Hadoop on HPC: Integrating Hadoop and Pilot-based Dynamic Resource Management

Andre Luckow, Ioannis Paraskevakos, George Chantzialexiou and Shantenu Jha

Overview

- Introduction and Motivation
- Background
- Integrating Hadoop/Spark with RADICAL-PIlot
- Experiments and Results
- Discussion
- Conclusion
- Future work

Introduction and Motivation

- The characteristics of Data-Intensive application are fairly distinct from HPC applications
- There are applications that cannot be easily characterized either as Data-Intensive or Compute-Intensive
  - Biomolecular Dynamics Analysis tools (e.g. MDAnalysis, CPPTraj) have characteristics of both
- The challenge for these tools is to scale to high data volumes as well as to couple simulation with analytics
- To the best of our knowledge, there is no solution that provides the capabilities of Hadoop and HPC jointly
- We explore the integration between Hadoop and HPC to allow applications to manage simulation (HPC) and data-intensive stages in a uniform way

Background

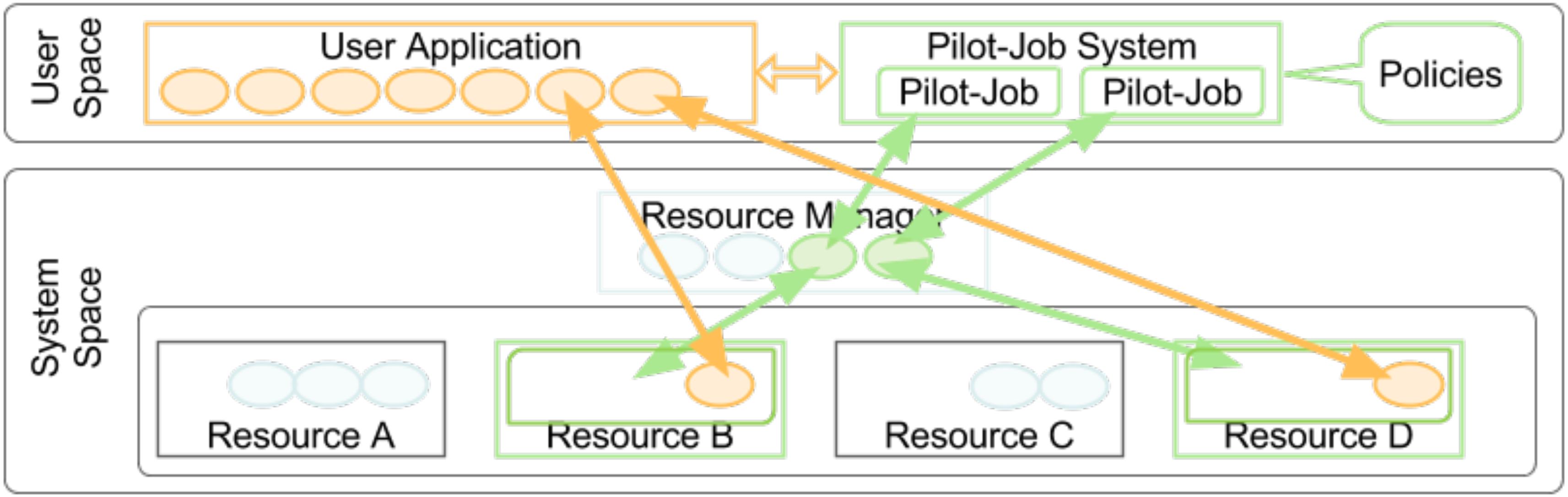
- HPC and Hadoop: Compute-Intensive applications vs Data-Intensive
- HPC uses parallel filesystems, Hadoop distributes the filesystem to the node's local hard drives
- Hadoop's scheduler YARN is optimized for data-intensive applications in contrast to HPC schedulers, like SLURM
- The complexity of creating sophisticated application lead to the creation of higher level abstractions.
- Many systems that run Hadoop on HPC exist
  - Hadoop on Demand
  - MyHadoop
  - MagPie
  - MyCray

Challenges

- How to achieve interoperability between HPC and Hadoop:
  - Challenge 1: Choice of storage and filesystem backend
    - Although Hadoop prefers local storage, many parallel filesystems provide special client library which improves interoperability
  - Challenge 2: Integration between HPC and Hadoop Environments
    - The Pilot-Abstraction can play the role of a unifying concept.
    - By utilizing the multi-level scheduling capabilities of YARN, the Pilot-Abstraction can efficiently manage Hadoop
  - Challenge 3: While keeping the generality, we try to keep the API as simple and unchanged as possible

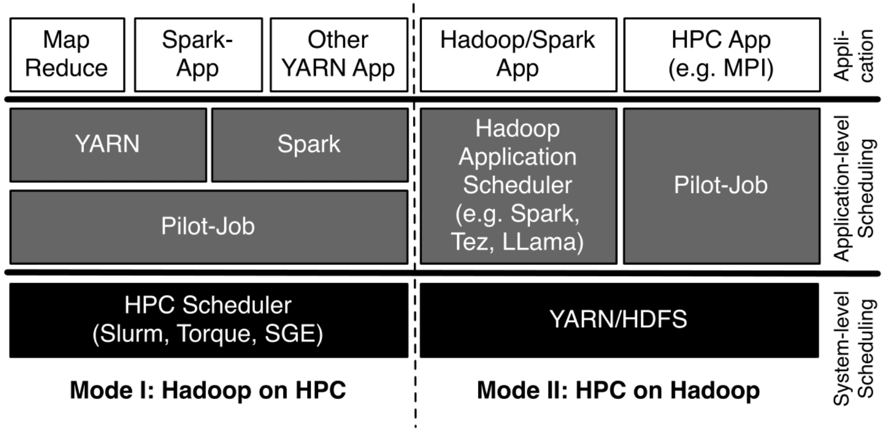
Pilot - Abstraction

- Defines and provides the following entities:
  - Pilot-Job: is a placeholder that is submitted to the management system representing a container for a dynamically determined set of compute tasks.
  - Pilot-Compute: allocates and manages a set of computational resources
  - Compute-Unit: a self-contained piece of work represented by an executable



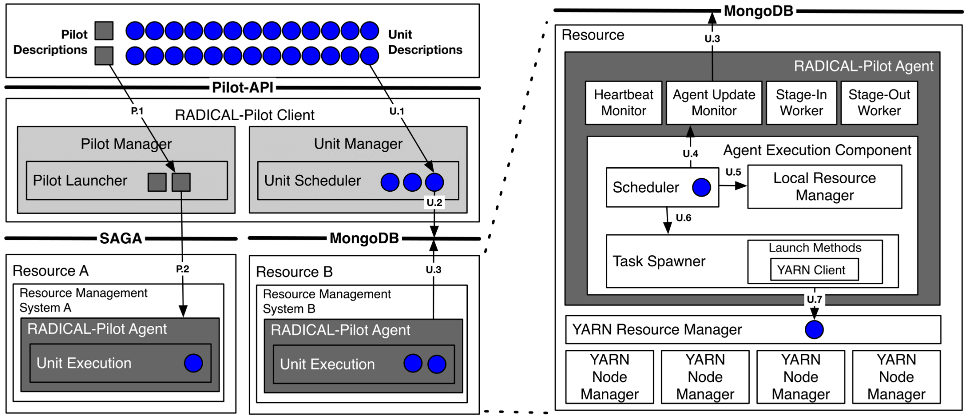
Integrating Hadoop/Spark with Pilot-Abstraction

- Two basic modes of integration:
  - **Mode I:** Running Hadoop/Spark applications on HPC environments:
    - RADICAL-Pilot-YARN
    - RADICAL-Pilot-Spark
  - **Mode II:** Running HPC on YARN clusters



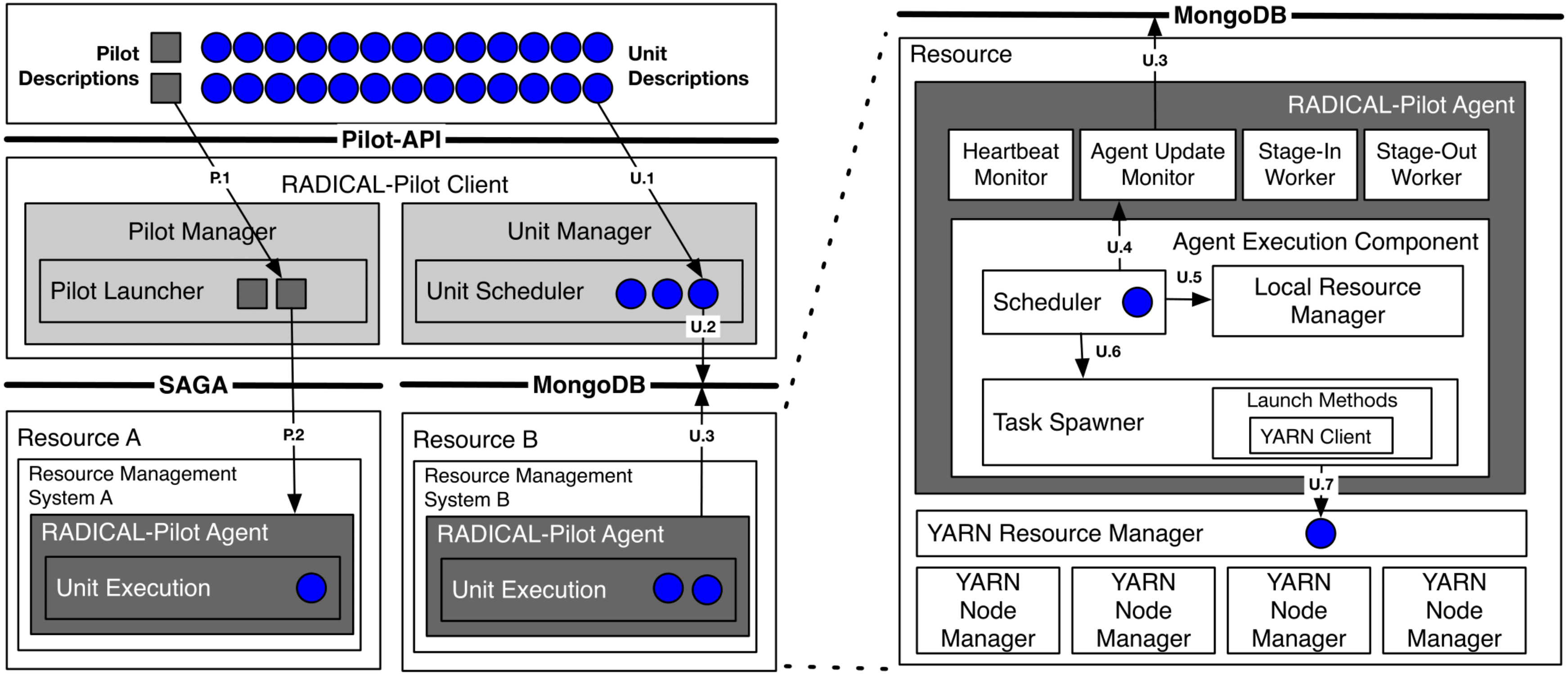
Integrating Hadoopk with RADICAL-Pilot

- RADICAL-Pilot consists of:
  - A client module with the Pilot-Manager and the Unit-Manager
  - An Agent (RADICAL-Pilot Agent) running on the resource
- The RADICAL-Pilot Agent consists of:
  - Heartbeat Monitor
  - Stage In/Out Workers
  - Agent Update Monitor
  - Agent Executing Component:
    - Local Resource Manager
    - A Scheduler
    - Task Spawner
    - Launch Method



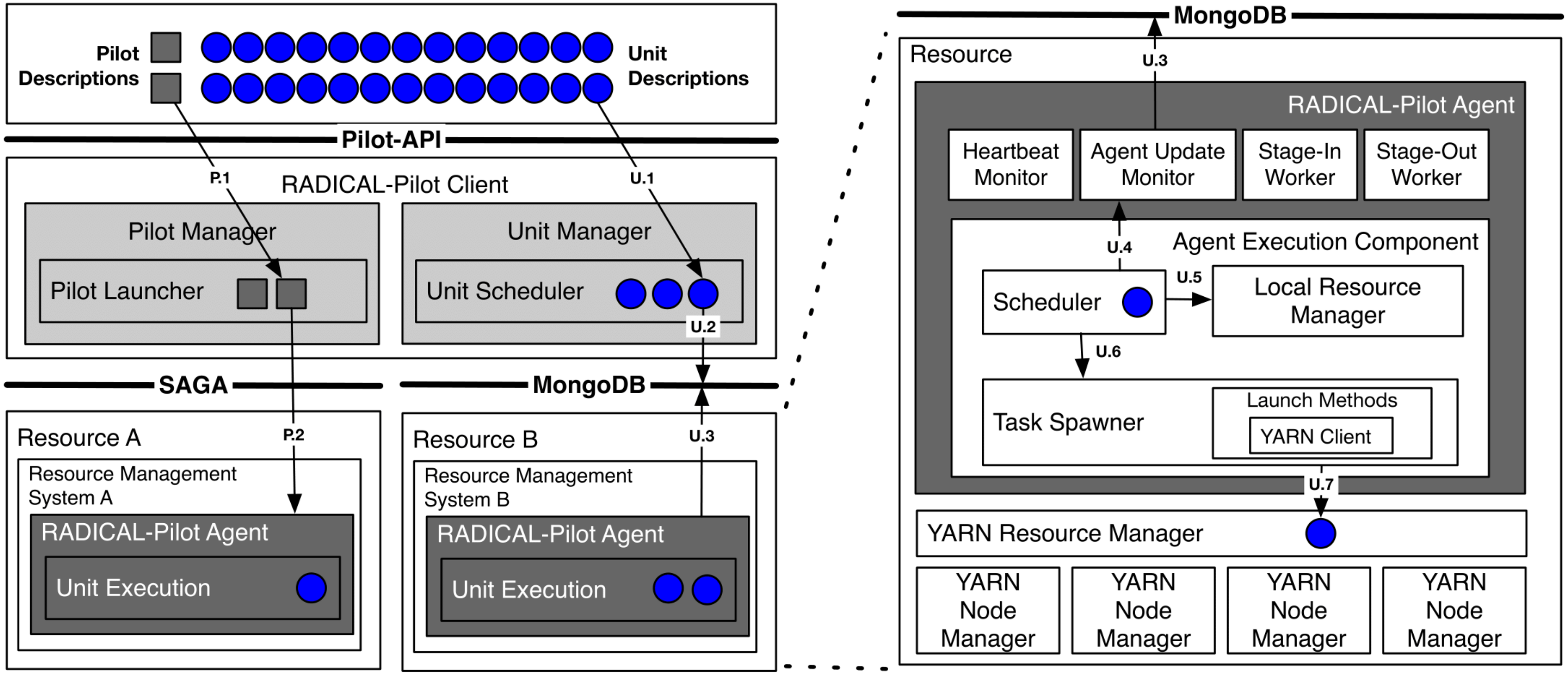
Integrating Hadoop with RADICAL-Pilot

- Agent Executing Component Extension:
  - Local Resource Manager: provides an abstraction to local resource details
    - In Mode I: Setups the Hadoop cluster
    - In Mode II: Collects the cluster resource information



Integrating Hadoop with RADICAL-Pilot

- Agent Executing Component Extension:
  - Scheduler: The scheduler uses YARN's REST API to get information about the cluster's utilization as Units are scheduled
  - Task Spawner: manages and monitors the execution of a compute unit
  - Launch Method: creates the yarn command based on the requirements (cpu, memory) of each compute unit

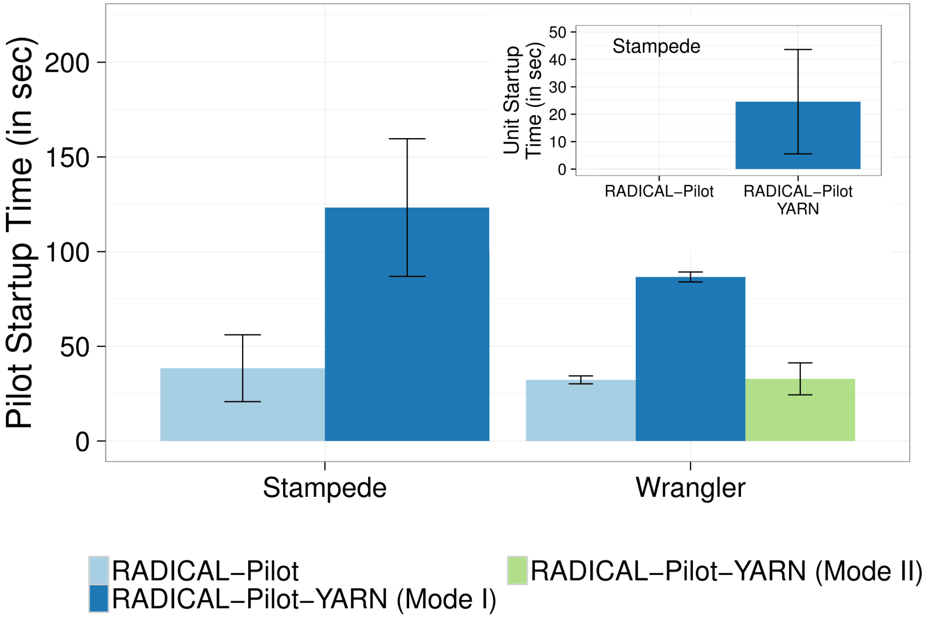


Experiments Setup

- Machines Used:
  - XSEDE/TACC Stampede: 16cores/node and 32GB/node
  - XSEDE/TACC Wrangler: 48cores/node and 128GB/node
- K-Means with 3 different senarios:
  - 10,000 points, 5,000 clusters
  - 100,000 points, 500 clusters
  - 1,000,000 point, 50 clusters
- System Configuration:
  - Up to 3 nodes
  - 8 tasks - 1node
  - 16 tasks - 2 nodes
  - 32 tasks - 3 nodes

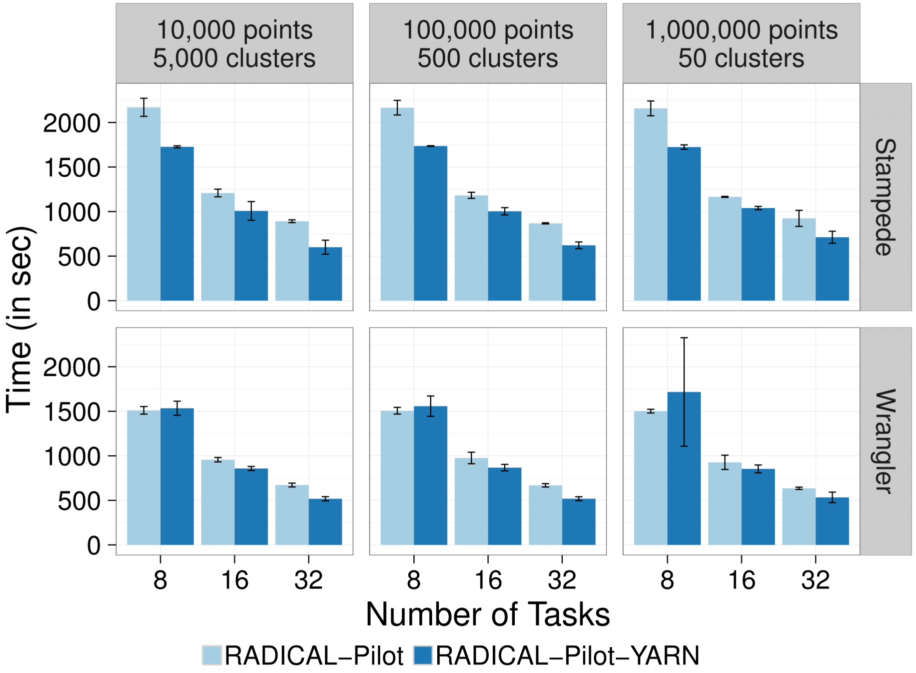
Results

- Experiment 1:
  - Start times comparison and evaluation for Pilot Startup and Compute Unit
- Mode I startup time is significantly larger both on Stampede and Wrangler.
- Mode II startup time on the dedicated Hadoop cluster that Wrangler provides is comparable to normal RADICAL-Pilot
- Inset figure shows a Compute-Unit's startup time.



Result

- K-Means Time to Completion comparison between normal RADICAL-Pilot execution and RADICAL-Pilot-YARN mode 1
- Constant Compute requirements over the 3 scenarios
- On average 13% shorter runtimes for RADICAL-Pilot-YARN
- Higher speedups on Wrangler, indicating that we saturated Stampede's RAM.



Discussion

- The pilot based approach provides a common framework for HPC and YARN applications over dynamic resources
- RADICAL-Pilot are able to detect and optimize Hadoop with respect to core and memory usage
- It is difficult to integrate Hadoop and HPC
  - Should they be used side by side?
  - Should HPC routines be called from Hadoop?
  - Should Hadoop be called from HPC?
- For which infrastructure a new application should be created? Should hybrid approaches be used?

Conclusions

- Presented the Pilot-abstraction as an integrating concept
- The Pilot-abstraction strengthens the state of practice in utilizing HPC resources in conjunction with Hadoop frameworks

Future Work

- We work with biophysical and molecular scientists to integrate Molecular Dynamics analysis
- Extending the Pilot Abstraction to support improved scheduling
- Adding support of further optimizations, e.g. in-memory filesystem and runtime

Thank you!

Any questions?!