Workshop on Extreme-Scale Programming Tools

18th November 2013

Supercomputing 2013

ELASTIC: Dynamic Tuning for Large-Scale Parallel Applications

Toni Espinosa

Andrea Martínez, Anna Sikora, Eduardo César and Joan Sorribes

Universitat Autónoma de Barcelona
Computer Architecture and Operating Systems Departament

Outline

①Motivation.

②Scalable Dynamic Tuning.

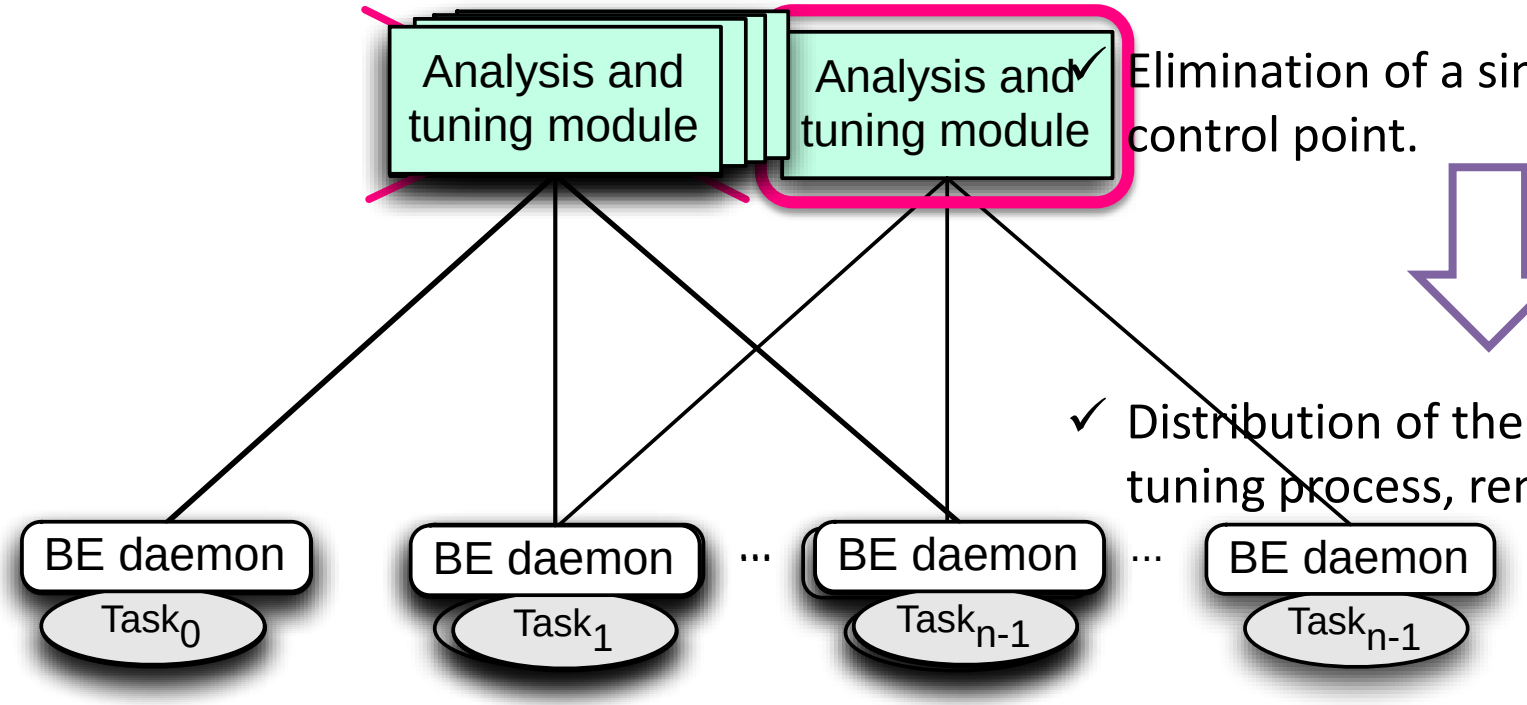
③ELASTIC.

④Experimental Evaluation.

⑤Conclusions and Future Work.

Motivation

Centralised Architecture of Tuning Tools



✓ Elimination of a single centralised control point.

✓ Distribution of the analysis and tuning process, remaining effective.

Outline

①Motivation.

②Scalable Dynamic Tuning.

③ELASTIC.

④Experimental Evaluation.

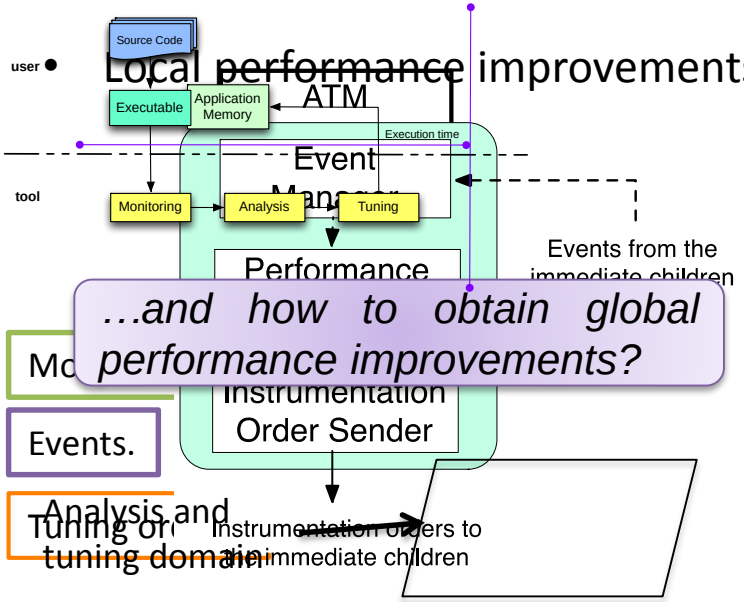
⑤Conclusions and Future Work.

Hierarchical Tuning Network

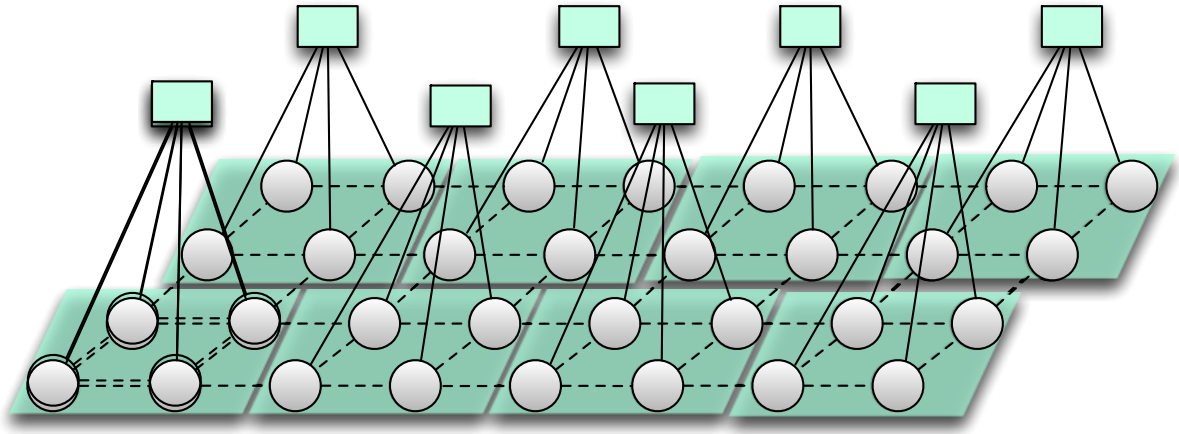
➢ ***Decompose***.

ATM

- A base level of **analysis and tuning modules** (ATM) that controls disjoint **domains** of application tasks.
- Local performance improvements are achieved.



…and how to obtain global performance improvements?

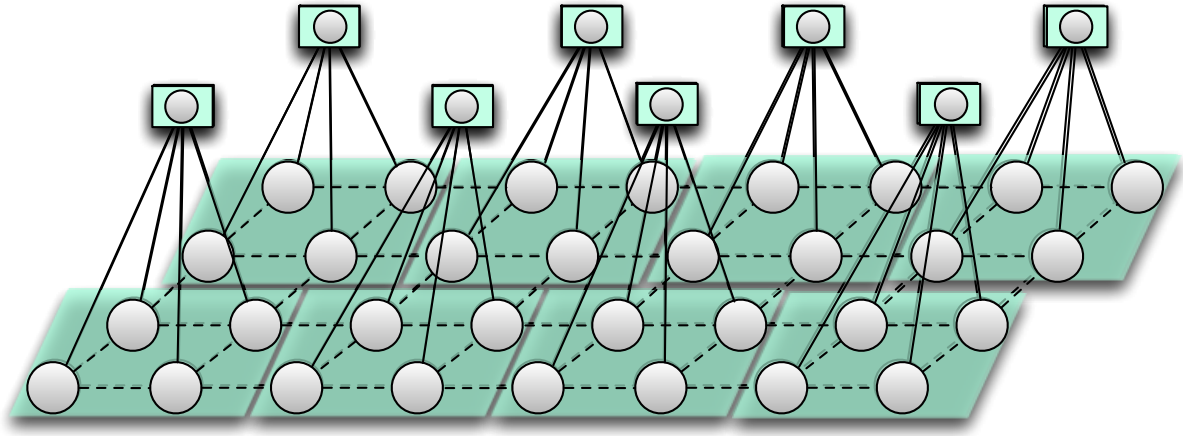


Hierarchical Tuning Network

➢ ***Abstract***.

- The **abstraction mechanism** is carried out by the ATMs.

…representing the tasks of the virtual parallel application

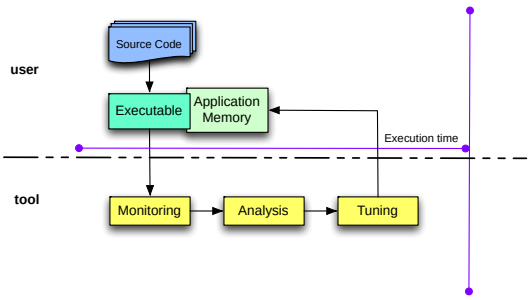


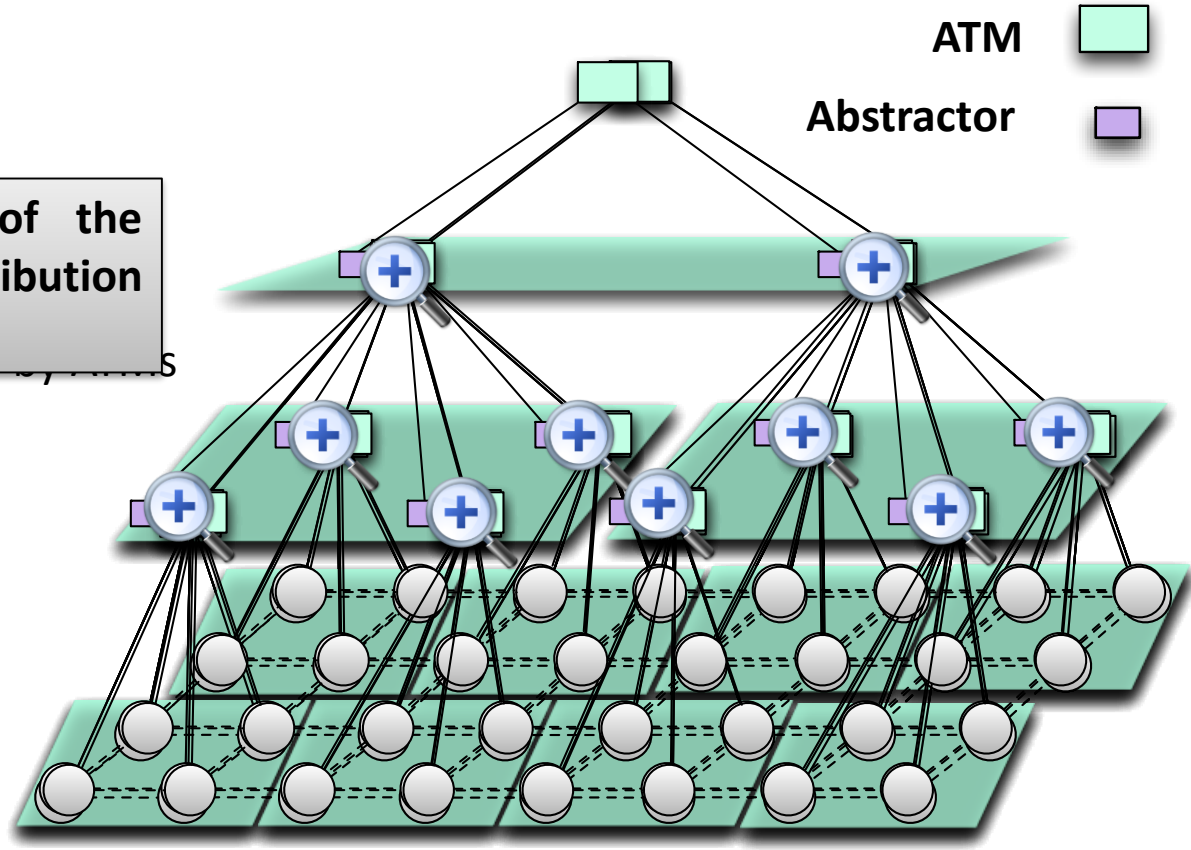
Hierarchical Tuning Network

- ***Decompose***.
- ***Abstract***.

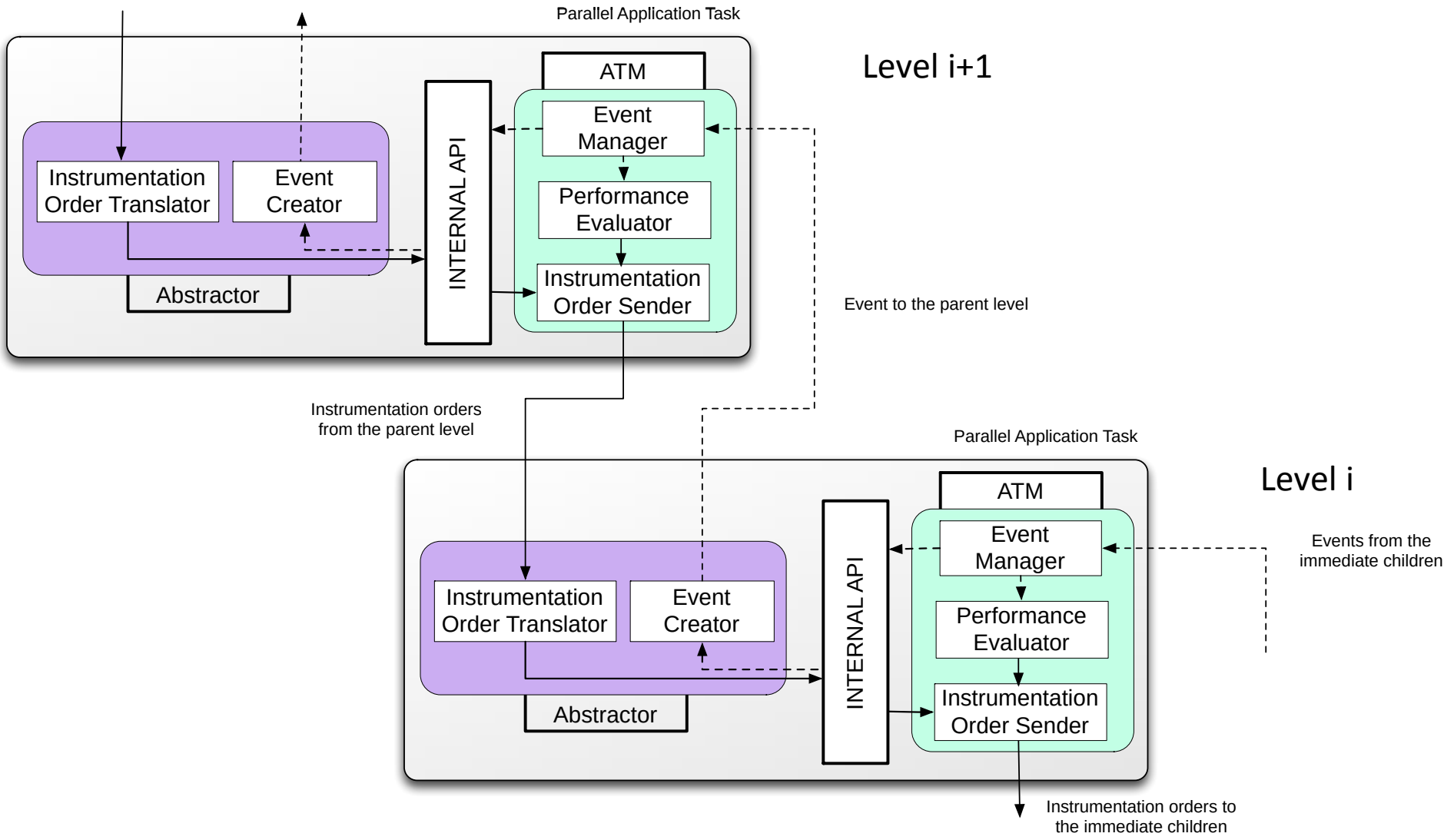
The actuation of each ATM of the network gives a hierarchical distribution of the analysis and tuning process

located at the high level.





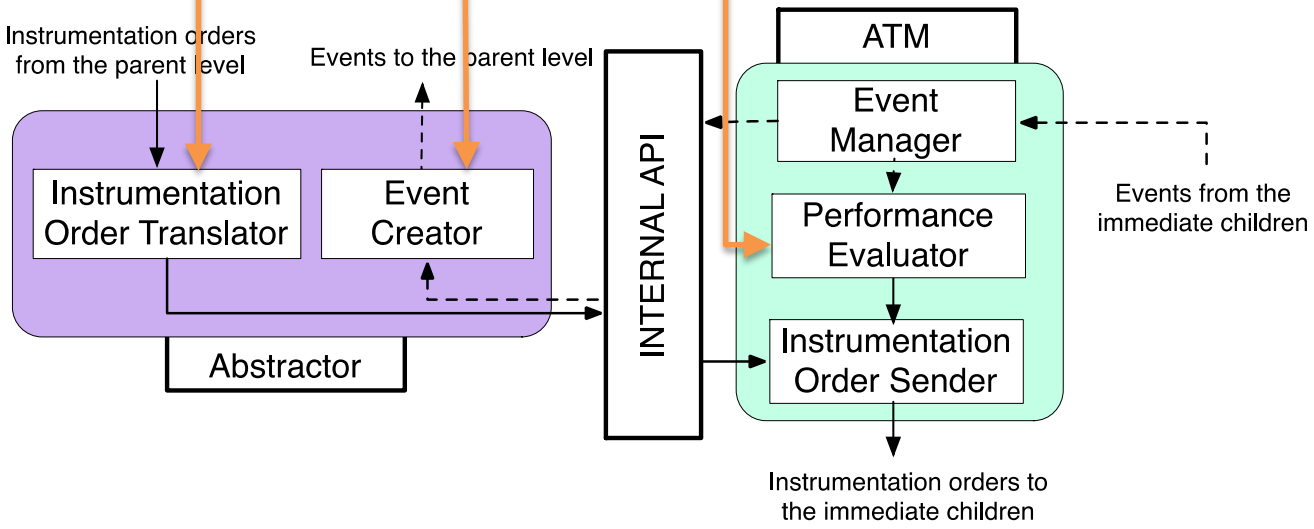
Abstraction Mechanism



Knowledge in the Tuning Network

Performance Model
Monitoring Points
Performance Expressions
Tuning Points, Actions and Synchronisation Method

Abstraction Model
How to translate a monitoring order
How to translate a tuning order
How to create a new event
How to decompose the real or virtual parallel application



Outline

①Motivation.

②Scalable Dynamic Tuning.

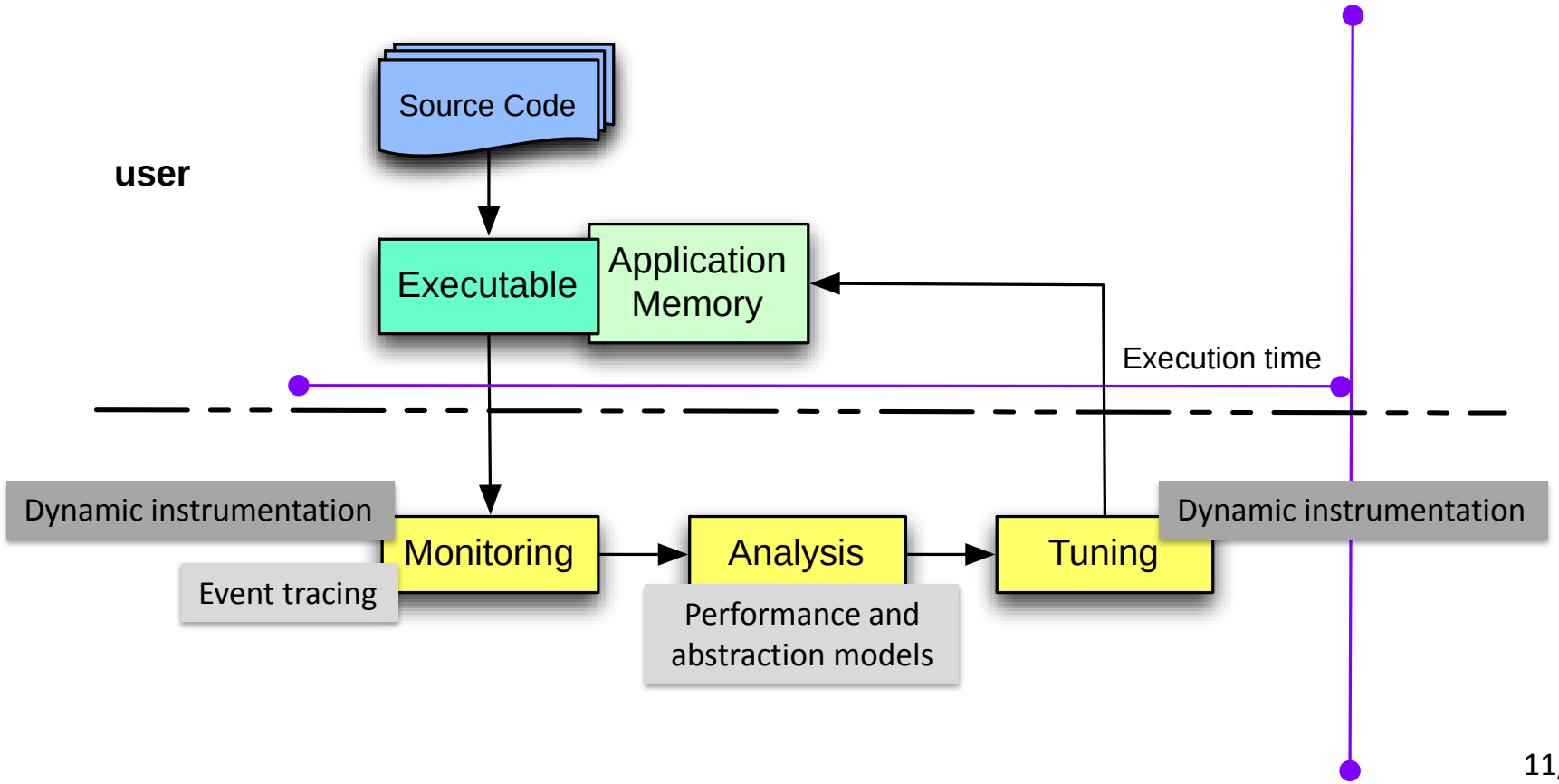
③ELASTIC.

④Experimental Evaluation.

⑤Conclusions and Future Work.

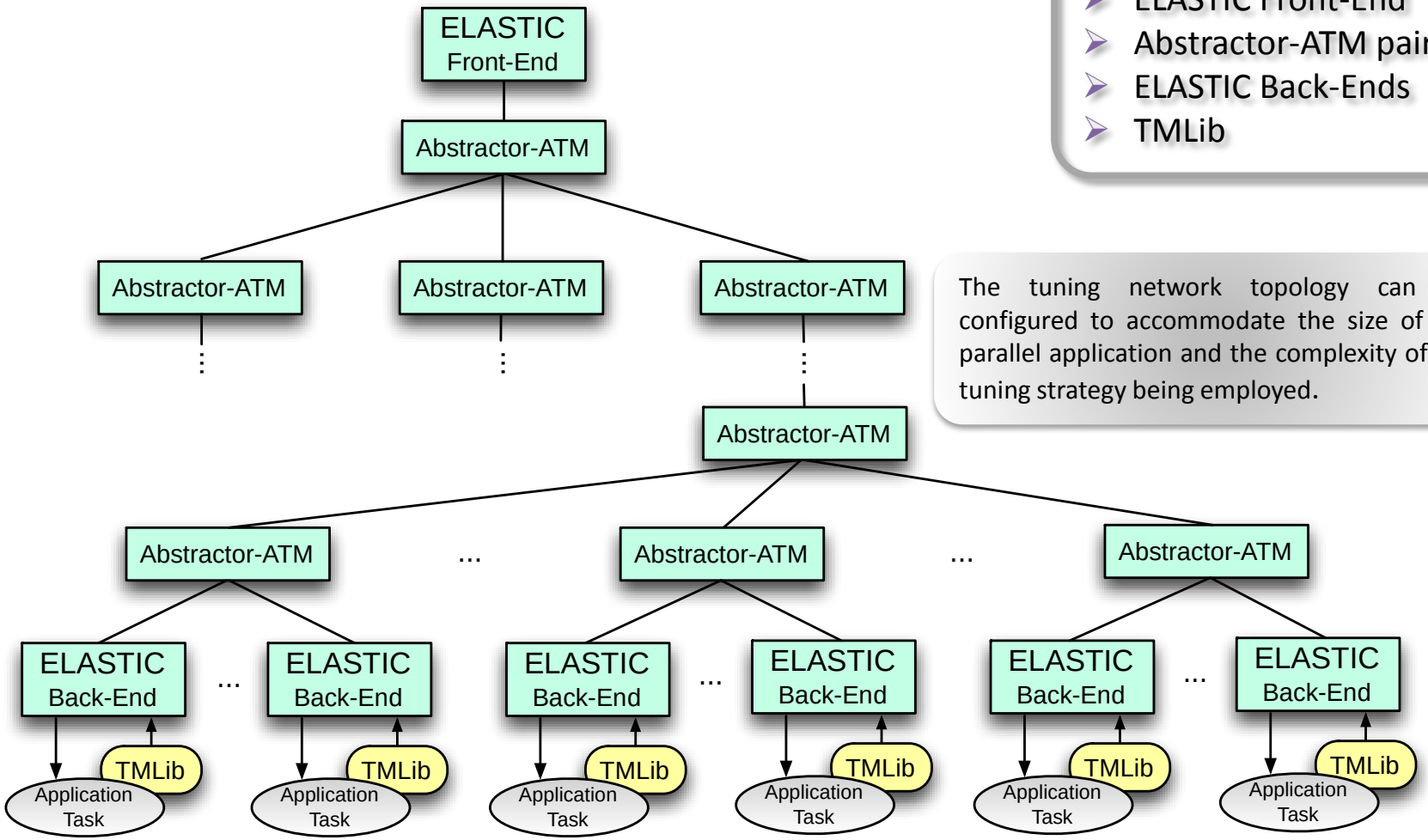
ELASTIC

- Prototype implementation in C++.
- For MPI parallel applications.
- Target systems: UNIX based supercomputers.



ELASTIC: Architecture

- ELASTIC Front-End
- Abstractor-ATM pair
- ELASTIC Back-Ends
- TMLib



The tuning network topology can be configured to accommodate the size of the parallel application and the complexity of the tuning strategy being employed.

ELASTIC Package

ELASTIC Package

Set of code and configurations that implements the performance and abstraction model.

TUNING AND ABSTRACTION API

Performance Model
Monitoring Points
Performance Expressions
Tuning Points, Actions and Synchronisation Method

Abstraction Model
How to translate a monitoring order
How to translate a tuning order
How to create a new event
How to decompose the real or virtual parallel application

```
vector<Monitoring Order> PerformanceEvaluator::InitialMonitoringOrders()

bool PerformanceEvaluator::NewEvent(Event *e)

vector<Order> PerformanceEvaluator::EvaluatePerformance()
```

```
vector<Monitoring Order> InstrumentationOrderTranslator::
                TranslateMonitoringOrder(MonitoringOrder *mo)

vector<Tuning Order> InstrumentationOrderTranslator::
                TranslateTuningOrder(TuningOrder *to)
```

```
bool EventCreator::NewEvent(Event *e)

vector<Event> EventCreator::CreateEvent()
```

ELASTIC Package

PLUGIN ARCHITECTURE

- Codification of the ELASTIC Package based on subclassing Abstractor-ATM components.

This plugin architecture converts ELASTIC into a general purpose tuning tool and gives it the flexibility to tackle a wide range of performance problems

Outline

①Motivation.

②Scalable Dynamic Tuning.

③ELASTIC.

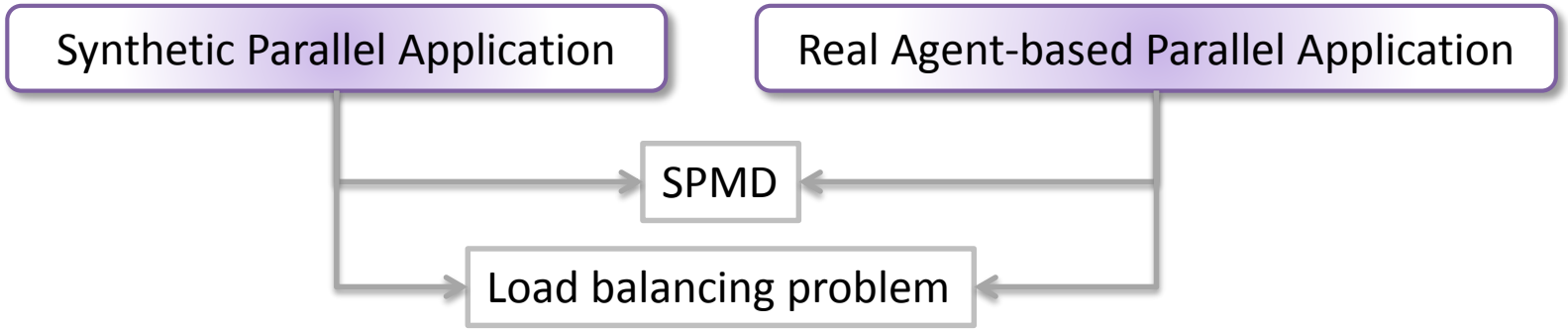
④Experimental Evaluation.

⑤Conclusions and Future Work.

Experimental Evaluation

The evaluation consists of

Executing a parallel application which presents a specific performance problem and using ELASTIC to dynamically detect and resolve the problem.



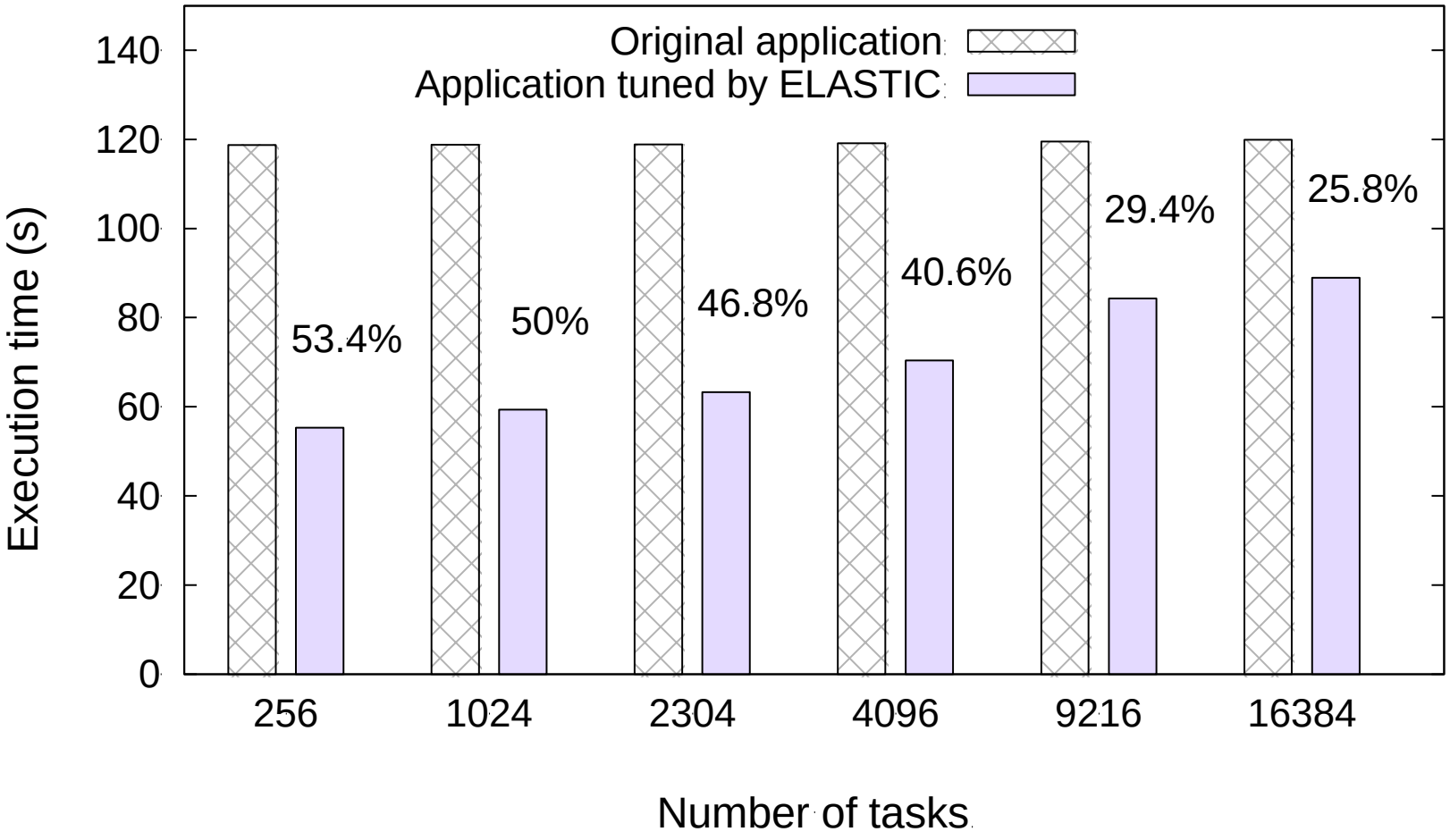
Execution Environment: **Supercomputer SuperMUC at LRZ**.

- 9400 compute nodes (155656 cores).
- Each node has 2 8-core 2.7 GHz Intel Xeon processors and 32GB main memory.
- SuSe Linux.

Experimental Evaluation

Results

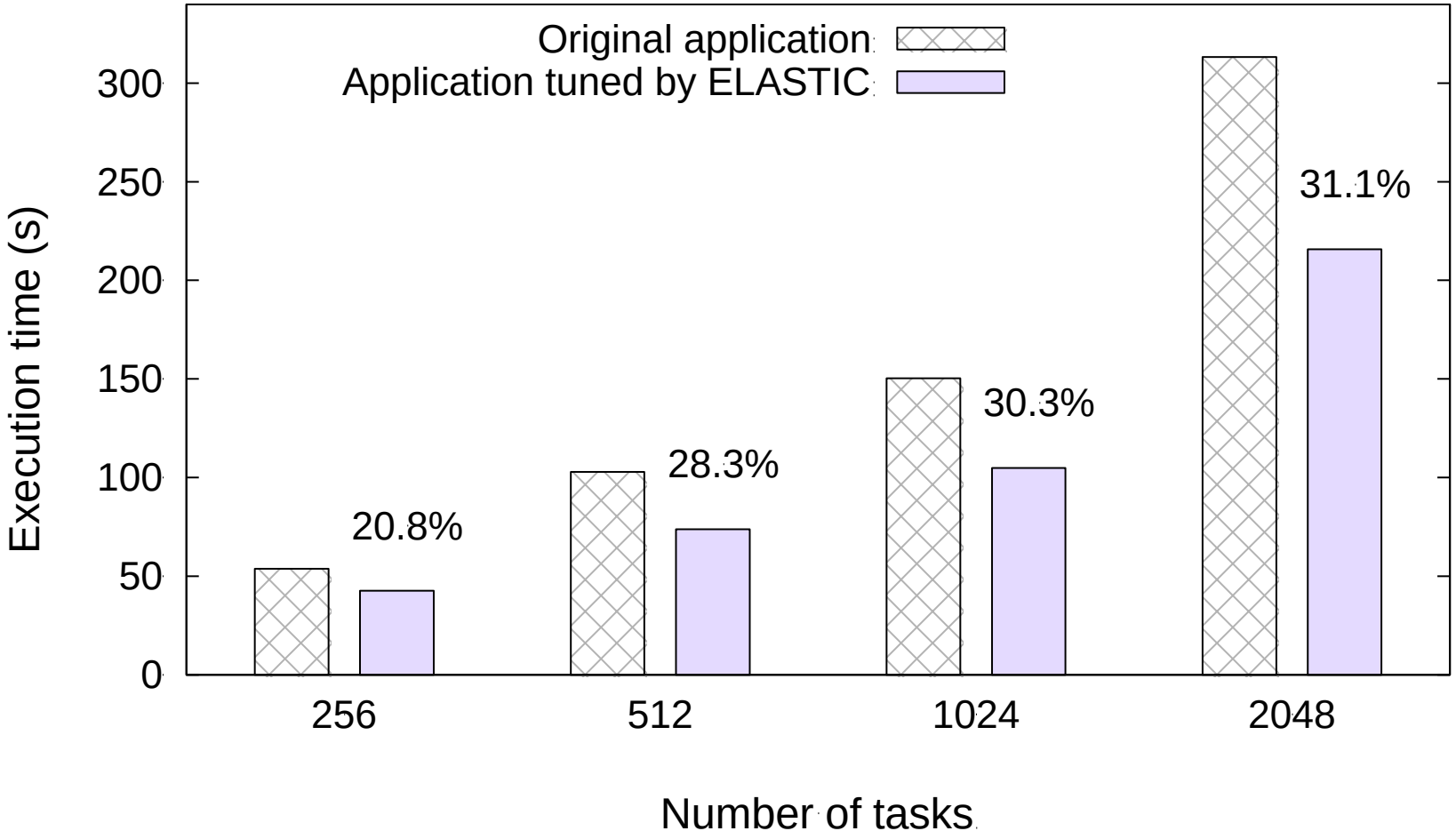
SYNTHETIC APPLICATION



Experimental Evaluation

Results

AGENT-BASED APPLICATION



Outline

①Motivation.

②Scalable Dynamic Tuning.

③ELASTIC.

④Experimental Evaluation.

⑤Conclusions and Future Work.

Conclusions

- ✓ The distribution of the dynamic tuning process through a hierarchical tuning network of analysis modules has been defined.

- ✓ ELASTIC, a tool that implements the proposed design, has been developed.
  - ➢ It offers dynamic tuning through dynamic monitoring, automatic performance analysis and dynamic modifications.
  - ➢ It presents an adaptable topology and a plugin architecture.

- ✓ The encouraging results obtained from the experimental evaluation using ELASTIC show that our approach is effective for large-scale dynamic tuning.

Future Work

- ✓ Creation of general ELASTIC Packages which solve a given performance problem.
  - ➢ It would be required a small adaptation to applied them to specific parallel applications.
- ✓ Combine our approach with the one implemented under the AutoTune project.

Workshop on Extreme-Scale Programming Tools

Supercomputing 2013

UAB

ELASTIC: Dynamic Tuning for Large-Scale Parallel Applications

Thank you

Toni Espinosa, Andrea Martinez, Anna Sikora, Eduardo César and Joan Sorribes

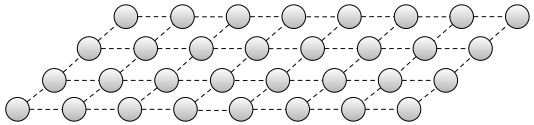
Universitat Autónoma de Barcelona

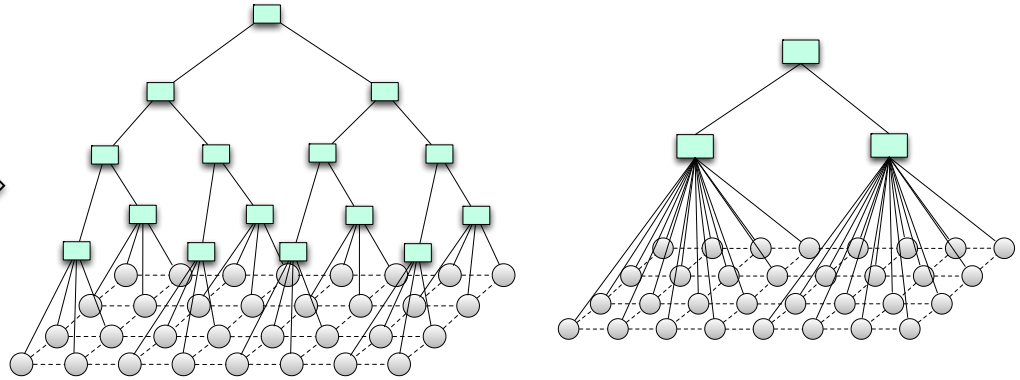
Computer Architecture and Operating Systems Departament

Tuning Network Topology

HOW DO WE SELECT A TUNING NETWORK TOPOLOGY?

Parallel Application
of N tasks



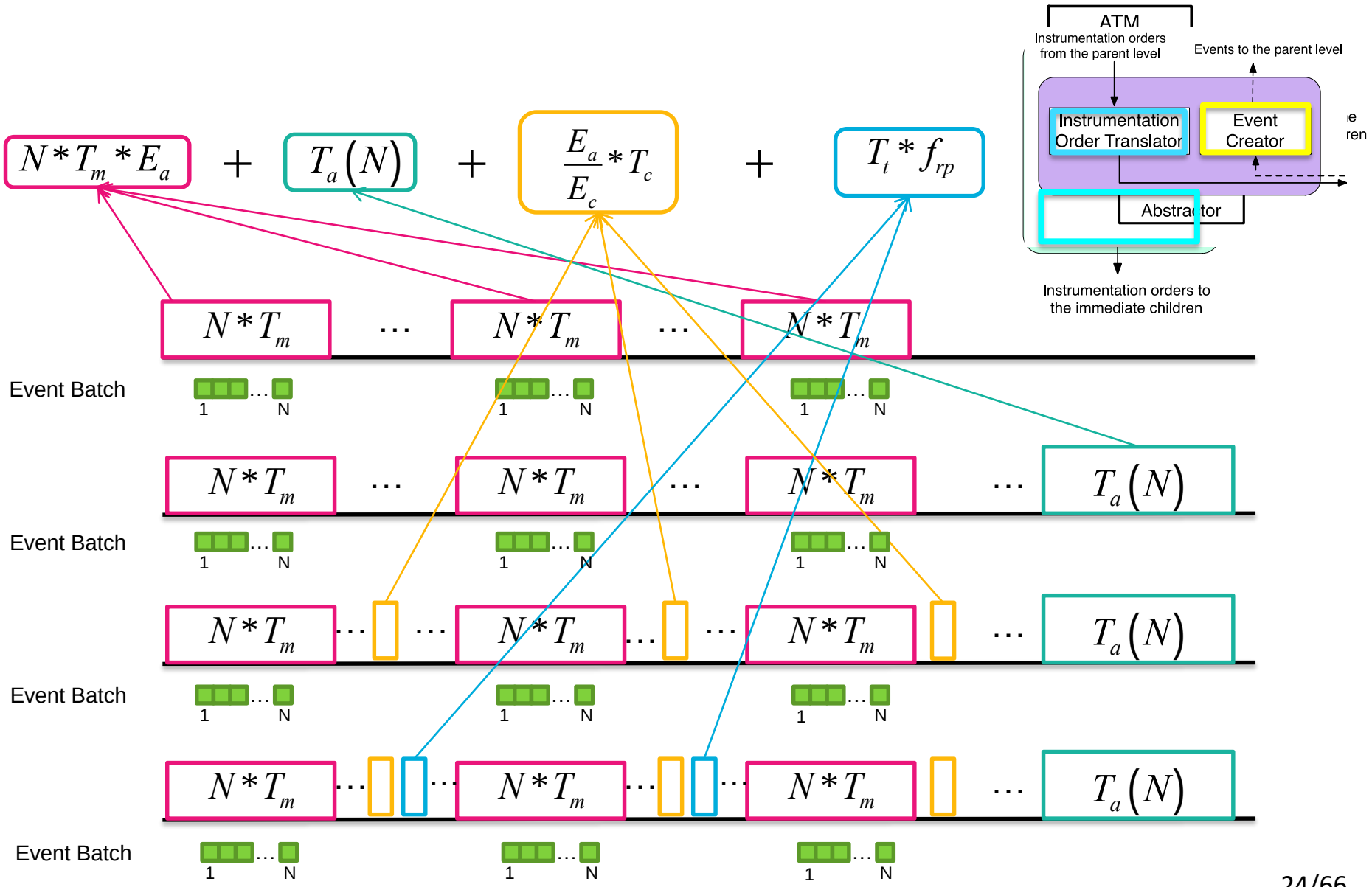


The structure of the topology will depend on the number of levels in the hierarchy and the number of Abstractor-ATM pairs in each level

The use of tuning networks composed of the minimum number of non-saturated Abstractor-ATM pairs.

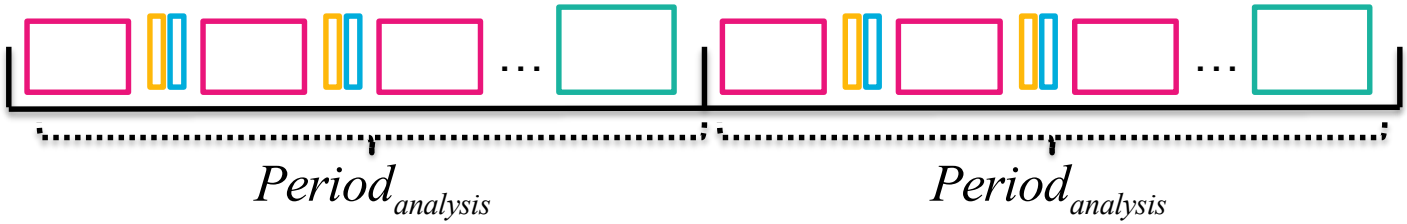
The maximum domain size that an Abstractor-ATM pair can manage without becoming saturated.

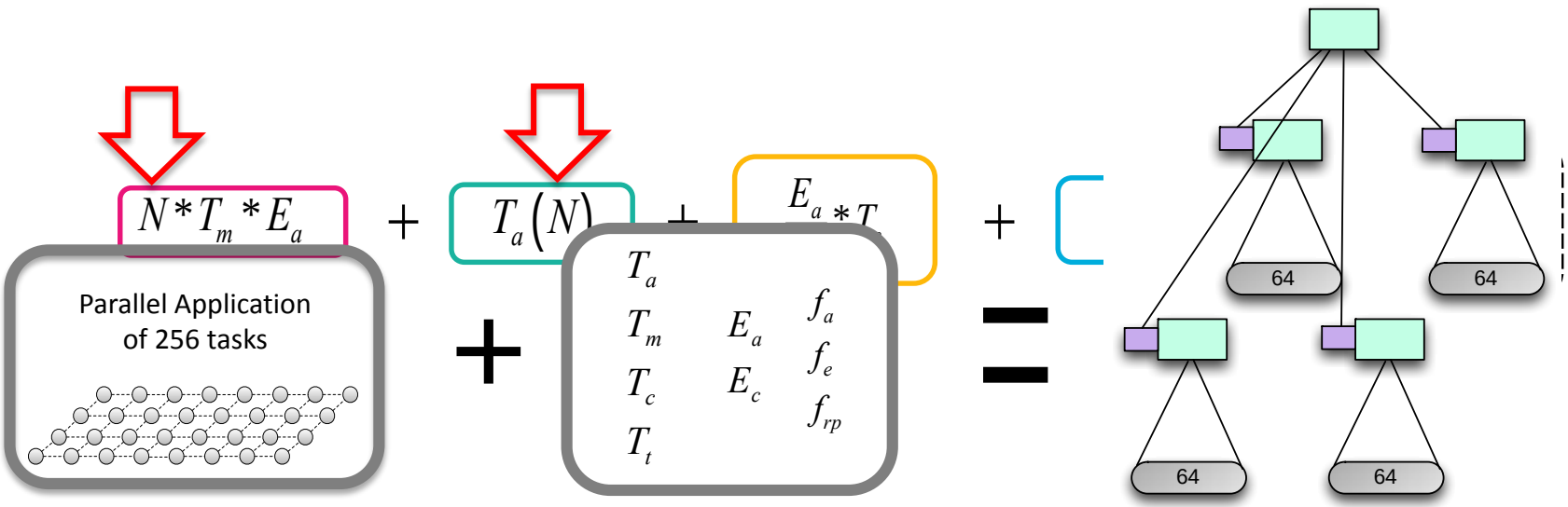
Modelling an analysis and tuning process



Calculating the number of tasks that an analysis module can manage without becoming saturated

$$f_{analysis} = \frac{f_e}{E_a}$$

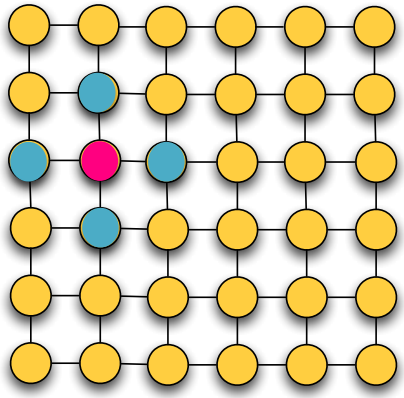


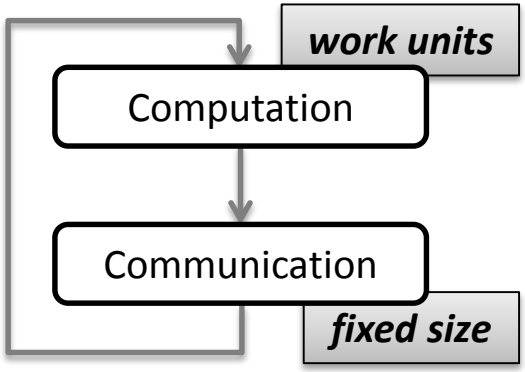


Experimental Evaluation

Application and performance problem

- Logical layout: 2D grid.
- Iteration pattern:
  - Computation.
  - Communication.

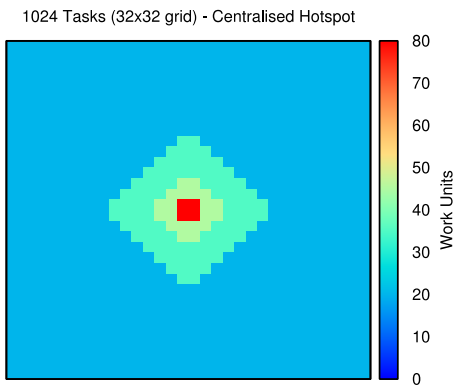


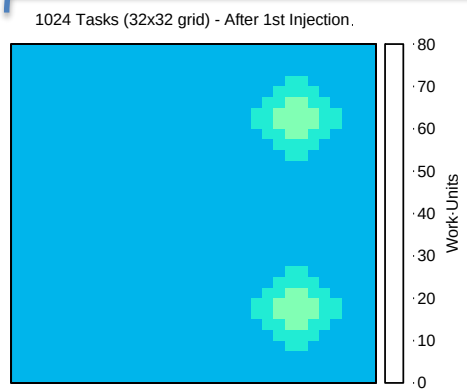


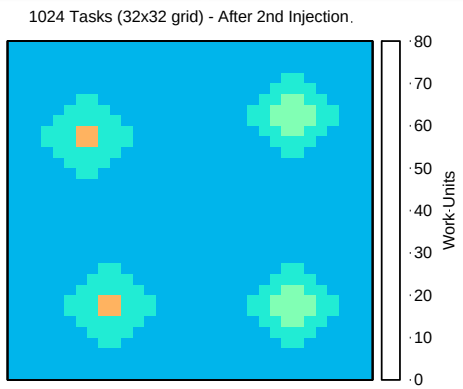
- **Load imbalance**: hotspots of additional workload were introduced into the application at runtime.

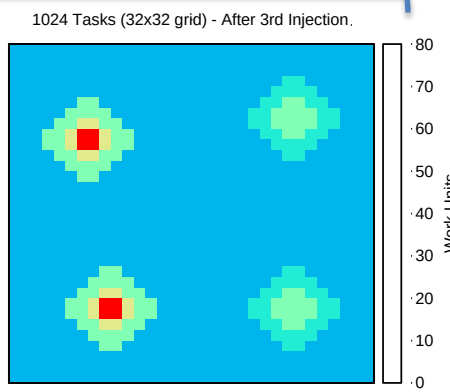
Scenario 1 **10%**

Scenario 2 **21%**









Experimental Evaluation

ELASTIC Package to balance the work units

Performance Model

Abstraction Model

Performance Expressions

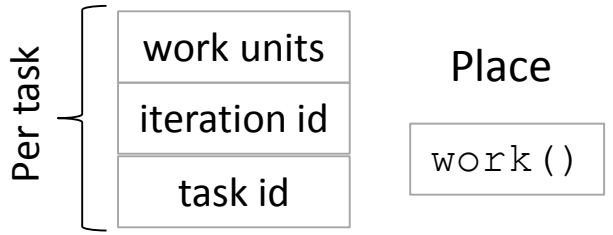
LOAD BALANCE ALGORITHM

Constraint: the work units only can be moved between neighbouring tasks

Input: `work units.`

Output: `work units to send.`

Measurement Points



Tuning Points, Action and Synchronisation

- Points: [`send_north, send_south, send_east, send_west`]
- Action: set the value of these variables.
- Synchronisation: at the beginning of the migration phase.
- Migration function

Experimental Evaluation

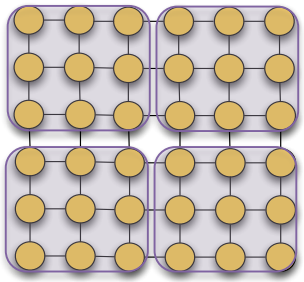
ELASTIC Package to balance the work units

Performance Model

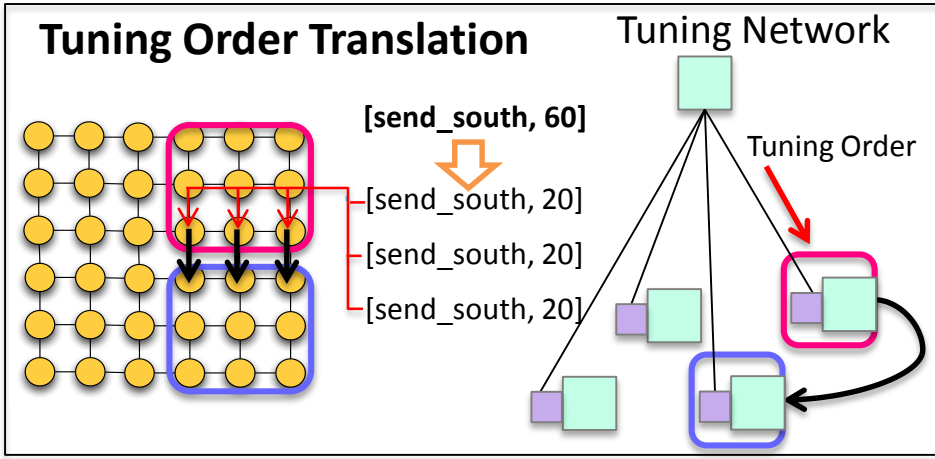
Abstraction Model

Decomposition Scheme Constraint: communication pattern between tasks

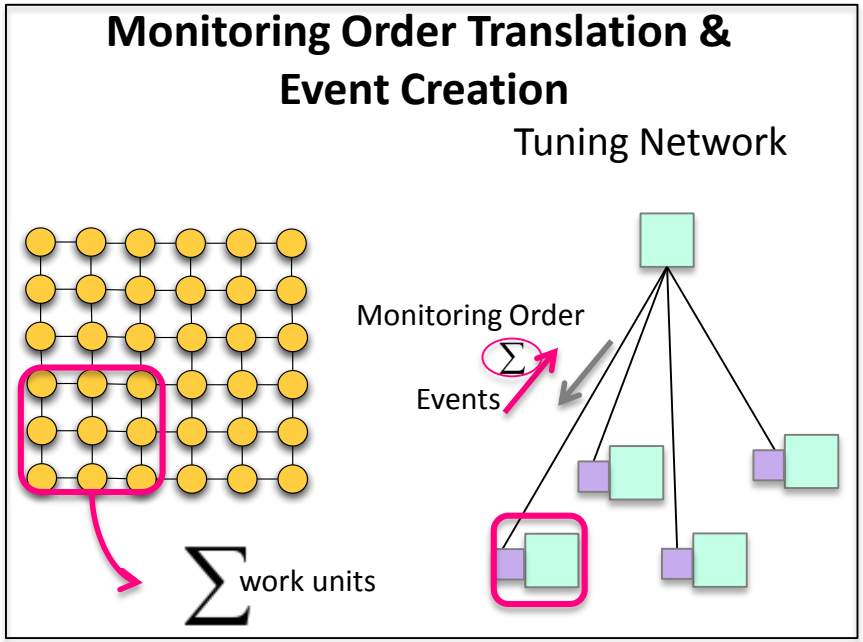
Real/Virtual Application



Tuning Order Translation



Monitoring Order Translation & Event Creation



Experimental Evaluation

Experimentation Plan

For the two scenarios

Maximum domain size = 314

Number of Application Tasks	Level 0 Number of ATMs	Level 1 Number of ATMs
256		
1 024		
2 304		
4 096		
9 216		
16 384		

- 100 iterations.
- 20 work units per task.
- The additional load is proportional to the size of the parallel application.

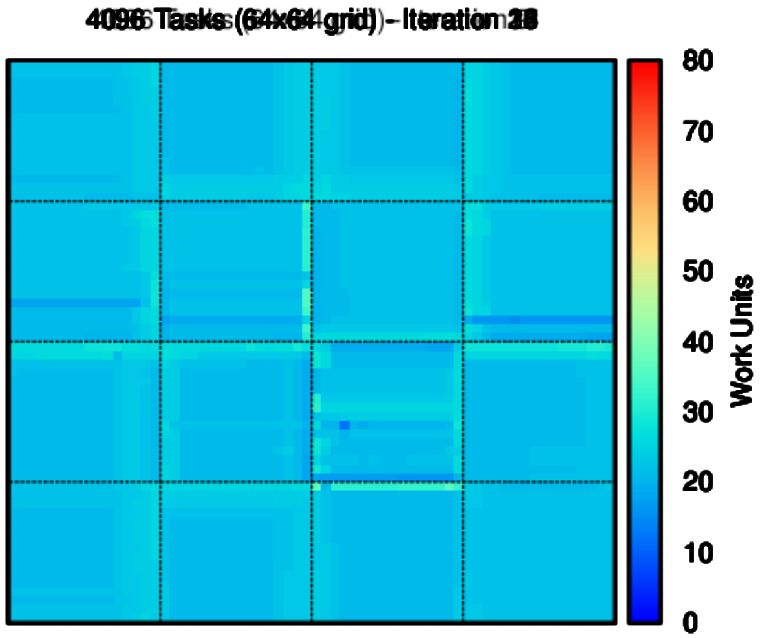
Experimental Evaluation

Results

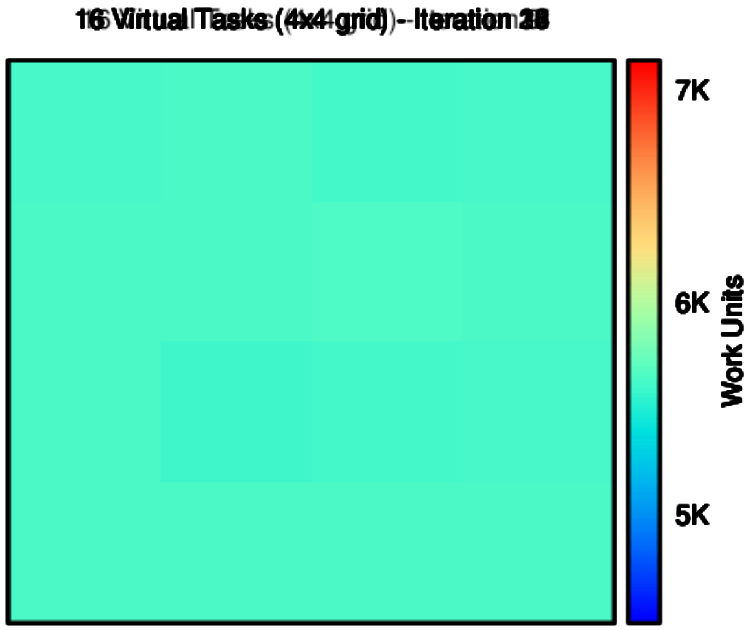
SCENARIO 1

4096 tasks parallel application (64x64 grid)

Load State



Original application

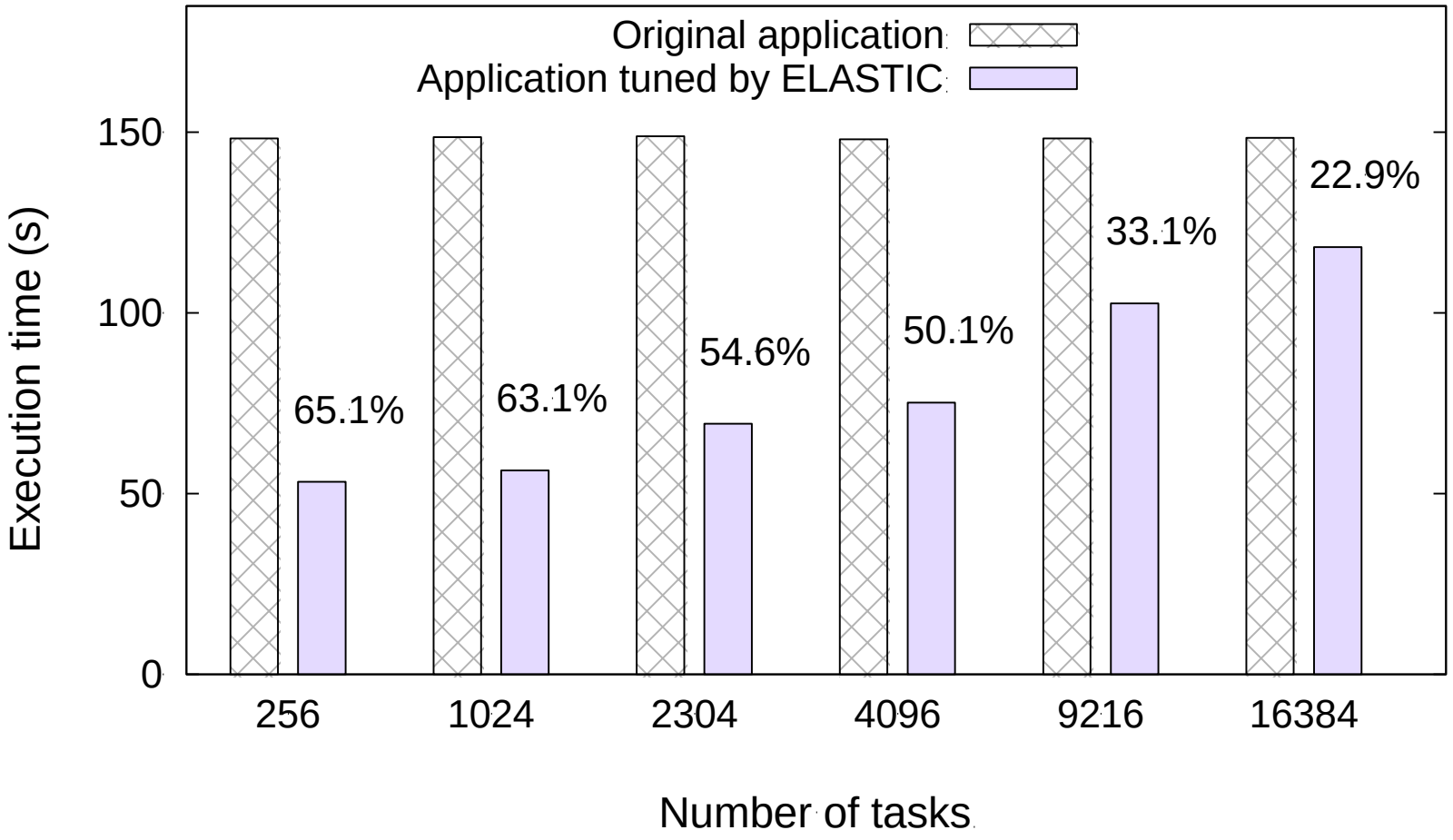


Virtual application

Experimental Evaluation

Results

SCENARIO 1

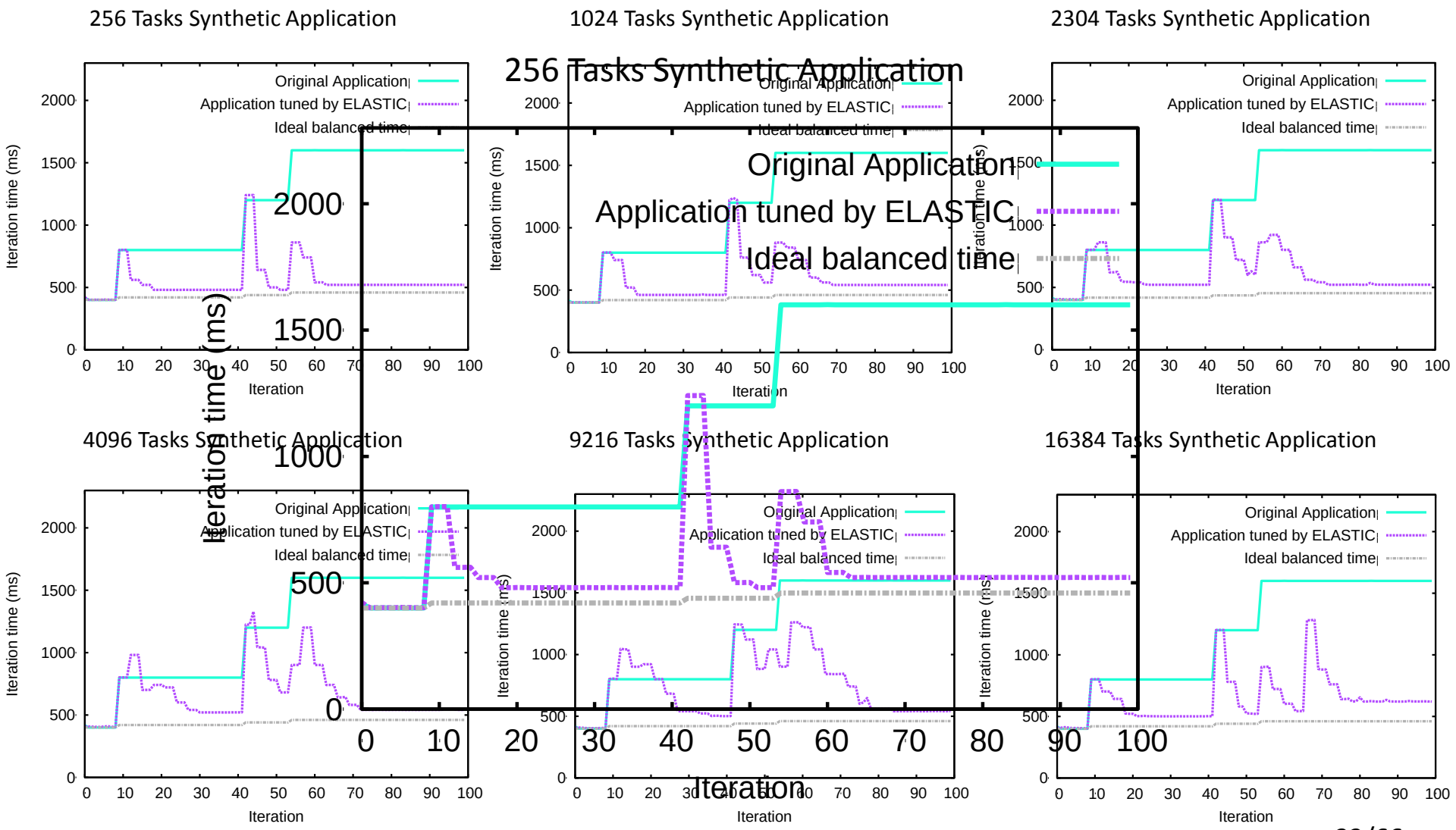


Experimental Evaluation

SCENARIO 2

Results

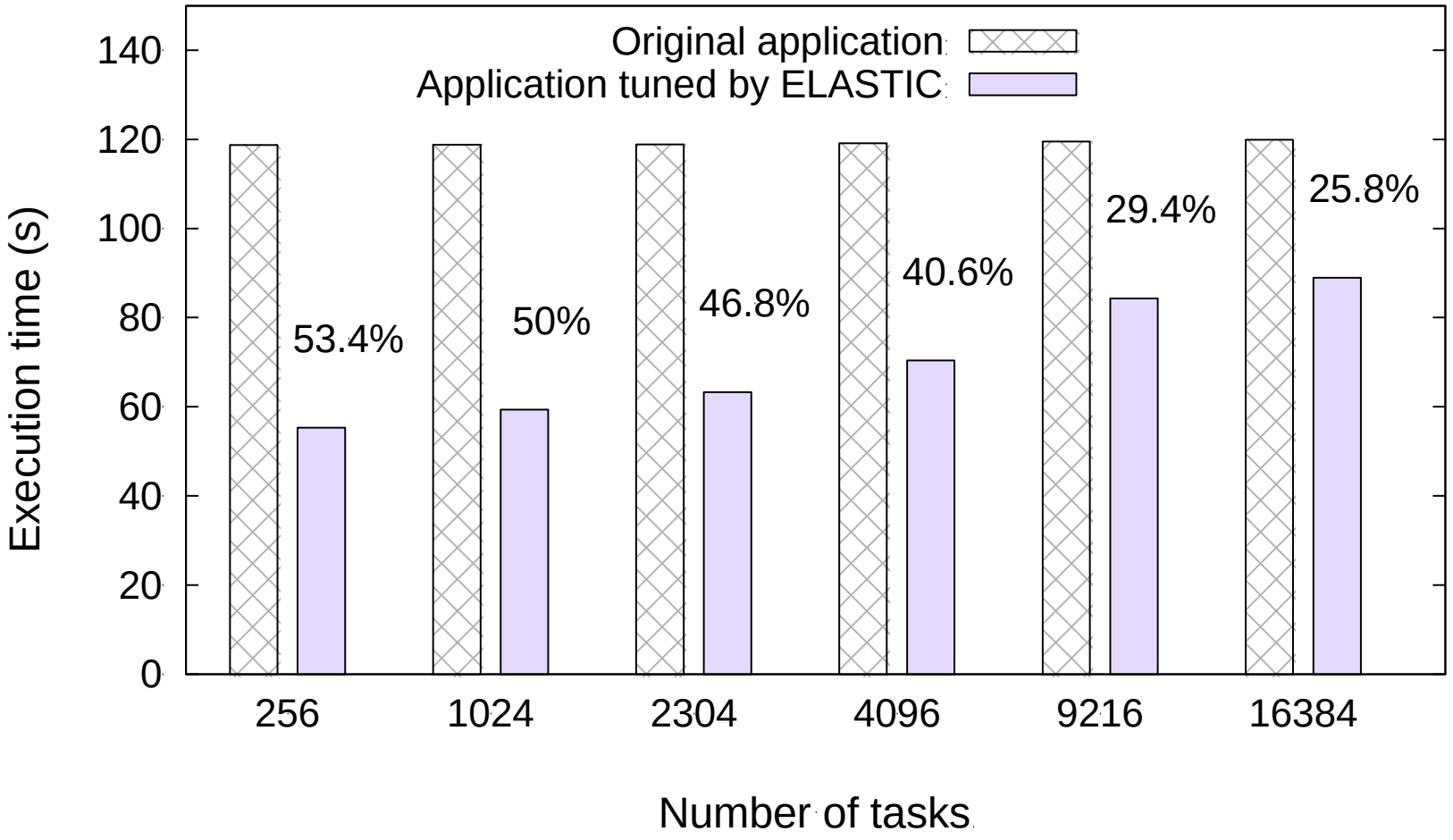
Iteration Time



Experimental Evaluation

Results

SCENARIO 2



Experimental Evaluation

Synthetic Parallel Application

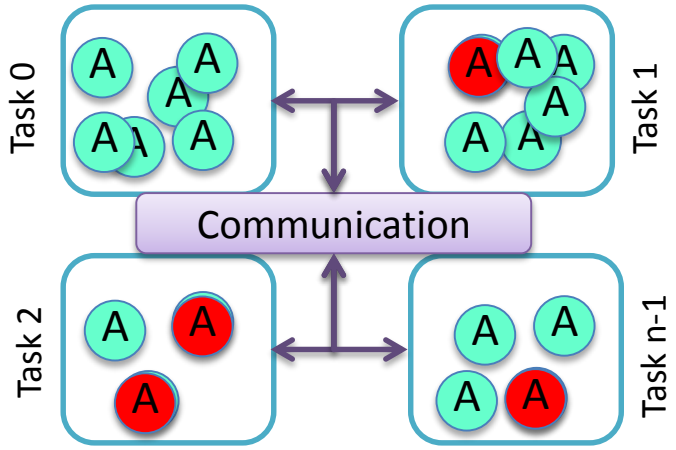
Real Agent-based Parallel Application

- Application and performance problem.
- ELASTIC Package developed.
- Experimentation plan.
- Results.

Experimental Evaluation

Application and performance problem

- Large-scale agent-based simulation.
- Simulates an epidemic model.
- Communication pattern: any-to-any



- **Load imbalance** problem due to the dynamic behaviour of the agents:
  - Births and death.
  - Time required to process an agent is not uniform.

Experimental Evaluation

ELASTIC Package to balance the computation time

Performance Model

Abstraction Model

Performance Expressions

LOAD BALANCE ALGORITHM

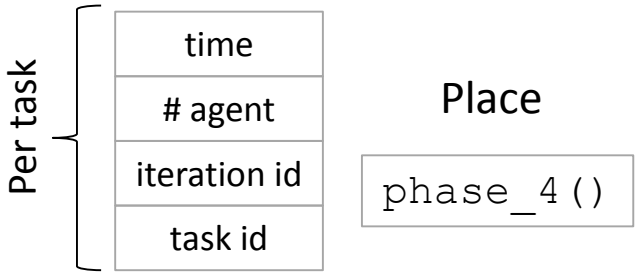
Marquez et al. 2013

Migrations can be between any two tasks in the analysis and tuning domain

Input: `#agents and computation time.`

Output: `#agents to send to each task.`

Measurement Points



Tuning Points, Action and Synchronisation

- Points: [intradomain_migrate[], interdomain_migrate[]]
- Action: set the value of these variables.
- Synchronisation: at the beginning of the migration phase.
- Migration functions.

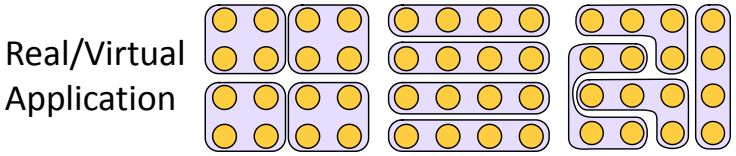
Experimental Evaluation

ELASTIC Package to balance the computation time

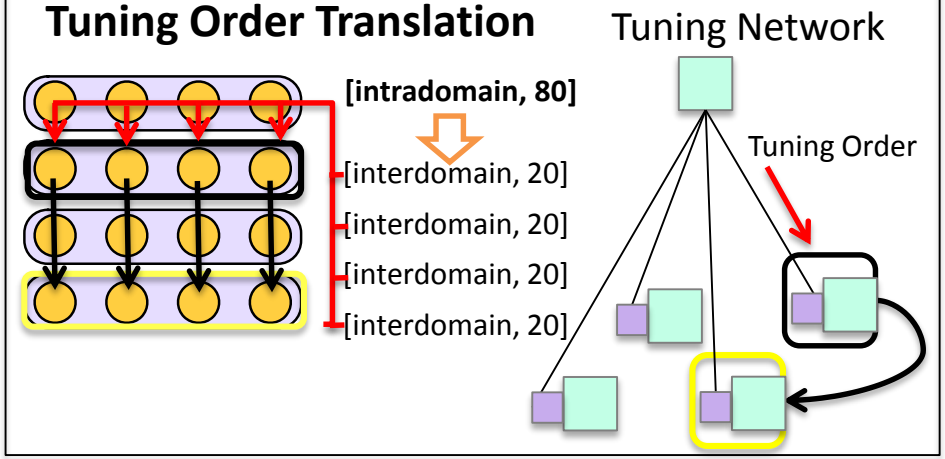
Performance Model

Decomposition Scheme

Constraint: domains with the same size

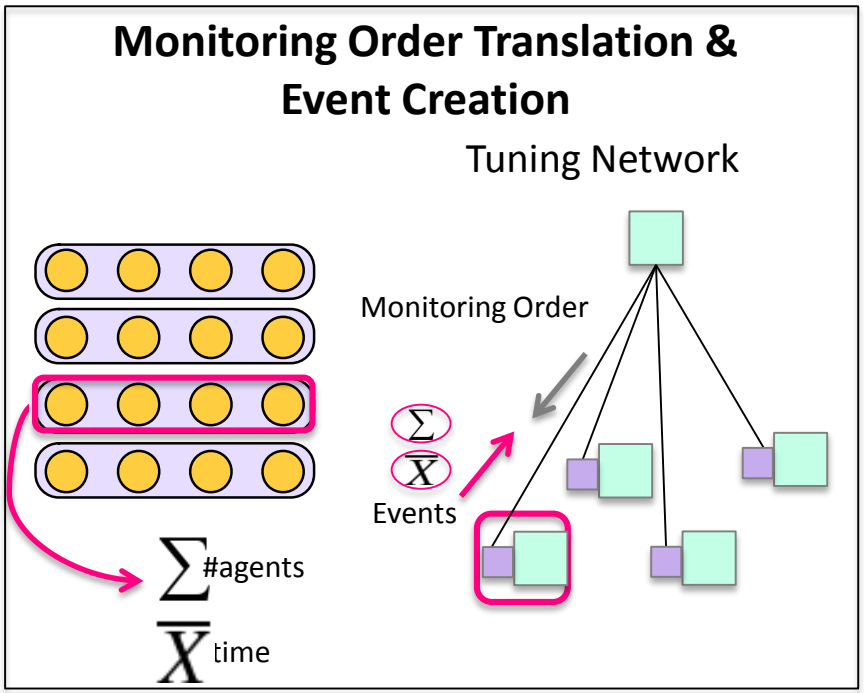


Tuning Order Translation



Abstraction Model

Monitoring Order Translation & Event Creation



Experimental Evaluation

Experimentation Plan

4 SIMULATION SCENARIOS

- Scale the number of agents.
- Scale the simulation space.

Number of Application Tasks	Number of Agents	Simulated Space Size
256	37 500	1020×1020
512	75 000	1440×1440
1 024	150 000	1800×1800
2 048	300 000	2240×2240

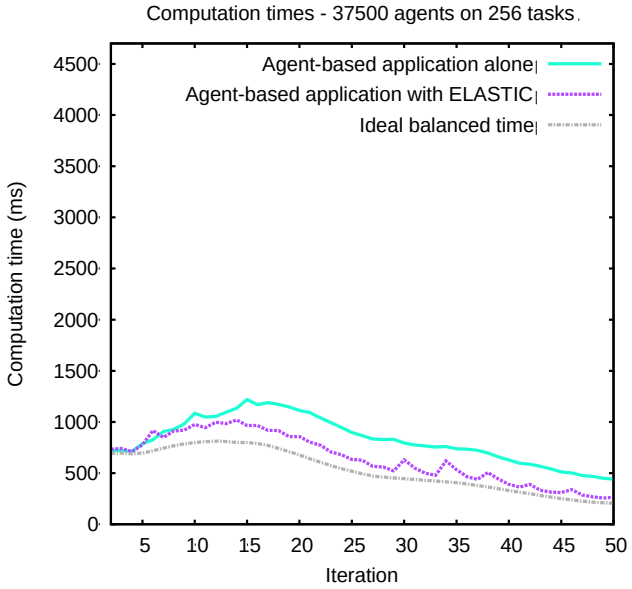
Domain size = 512

Number of Application Tasks	Level 0 Number of ATMs	Level 1 Number of ATMs
256	1	-
512	1	-
1 024	2	1
2 048	4	1

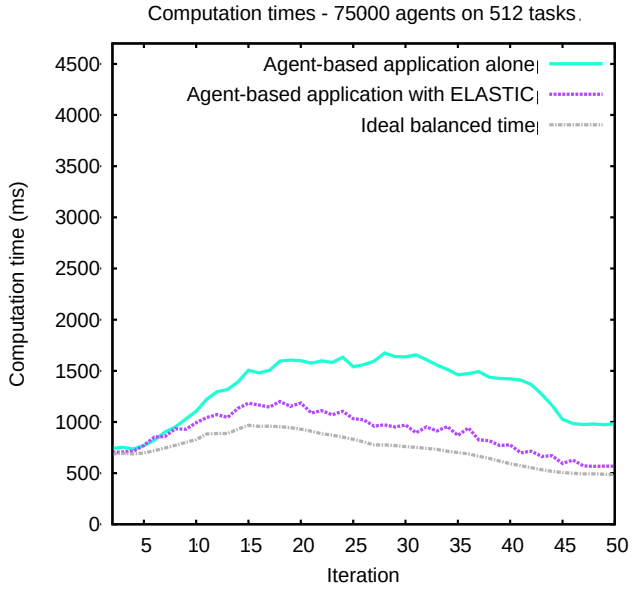
Experimental Evaluation

Results

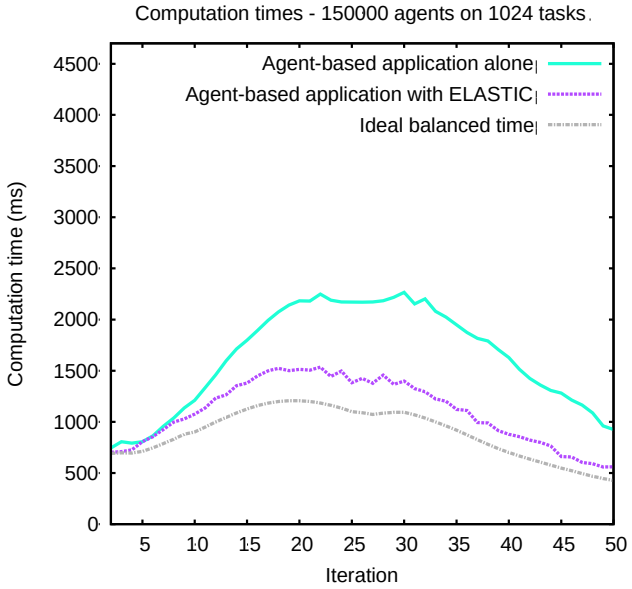
256 tasks



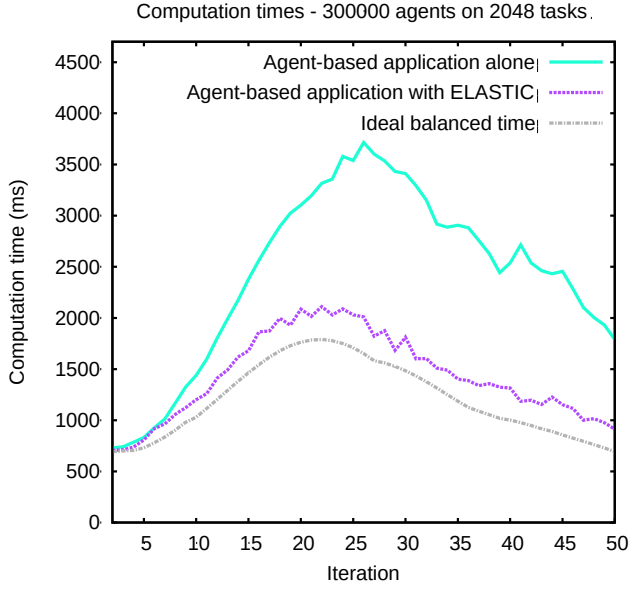
512 tasks



1024 tasks



2048 tasks



Experimental Evaluation

Results