H2020 FETHPC-1-2014

An Exascale Programming, Multi-objective Optimisation and Resilience Management Environment Based on Nested Recursive Parallelism

Project Number 671603

D4.3 – AllScale runtime system monitoring infrastructure

WP4: Unified Runtime System for Extreme Scale Systems

Version: 0.1
Author(s): Thomas Heller (FAU)
Date: 28/03/18

Due date:	PM30
Submission date:	30/04/2017
Project start date:	01/10/2015
Project duration:	36 months
Deliverable lead organization	FAU
Version:	0.1
Status	Draft
Author(s):	Thomas Heller (FAU)
Reviewer(s)	Xavier Aguilar(KTH), Kiril Dichev (QUB)

Dissemination level	
PU/PP/RE/CO	*PU - Public*

Disclaimer

This deliverable has been prepared by the responsible Work Package of the Project in accordance with the Consortium Agreement and the Grant Agreement Nr 671603. It solely reflects the opinion of the parties to such agreements on a collective basis in the context of the Project and to the extent foreseen in such agreements.

Acknowledgements

The work presented in this document has been conducted in the context of the EU Horizon 2020. AllScale is a 36-month project that started on October 1st, 2015 and is funded by the European Commission.

The partners in the project are UNIVERSITÄT INNSBRUCK (UBIK), FRIEDRICH-ALEXANDER-UNIVERSITÄT ERLANGEN NÜRNBERG (FAU), THE QUEEN'S UNIVERSITY OF BELFAST (QUB), KUNGLIGA TEKNISKA HÖGSKOLAN (KTH), NUMERICAL MECHANICS APPLICATIONS INTERNATIONAL SA (NUMEXA), IBM IRELAND LIMITED (IBM).

The content of this document is the result of extensive discussions within the AllScale Consortium as a whole.

More information

Public AllScale reports and other information pertaining to the project are available through the AllScale public Web site under http://www.allscale.eu.

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	28/03/18	Frist draft	Thomas Heller
1.0	17/04/18	Incorporating Reviews	Thomas Heller

Table of Contents

1 Table of Contents

Executive Summary

The AllScale runtime system offers a comprehensive and complete mechanism to allow monitoring of non-functional application metrics. This feature is provided by two means: 1) emitting signals from within AllScale specific events and 2) HPX performance counters. This allows for a generic approach to deliver the system status to the monitoring component.

2 Introduction

One of the key aspects of the AllScale runtime system is the interaction between the monitoring and the runtime component for non-functional parameters of an application. In order to accomplish goals like dynamic load balancing, a monitoring infrastructure needs to be provided.
As the AllScale runtime system is built on top of the HPX runtime system, it makes use of the performance counter infrastructure provided by HPX. Furthermore, the monitoring is extended to provided discrete signals of AllScale specific events to give the monitoring component the possibility to react on given key aspects of the AllScale runtime, such as the execution start of a work item.

3 Implementation

3.1 The HPX performance counter framework

The HPX performance counter framework is a powerful facility to support runtime adaptivity. As such, it provides a rich set of predefined counters. Those counters can be queried by a given name and can be retrieved for all running localities or threads (where applicable). The list of currently implemented counters can be seen here: https://stellar-group.github.io/hpx/docs/html/hpx/manual/performance_counters/counters.html.
The performance counters can be queried either from the command line, directly from the application or within the monitoring or resilience components. This allows for maximum flexibility as well as a foundation for experiments to see the effect of various different possible implementations on the actual runtime.

3.2 The AllScale monitoring events

The monitoring events are the following:

- work_item_enqueued
  - This signal is emitted whenever a Work Item is being spawned and put into the schedulers queue
- work_item_dequeued
  - This signal is emitted when a work item is dequeued and prepared for execution by the scheduler
- work_item_split_execution_started/work_item_process_execution_started
  - This signal is emitted when the work item split or process variant execution starts, that is when all transfers and dependencies have been completed
- work_item_split_execution_finished/work_item_process_execution_finished
  - This signal is emitted when the execution of the split or process work item variant is done
- work_item_result_propagated

    - This signal is emitted once the result of the work has been propagated to the parent.
- work_item_first
    - This signal is emitted to mark the begin of a iteration
- work_item_dispatched
    - This signal is emitted to mark a migration of a work item from one locality to another

The events can be subscribed to, to allow the other components to react whenever such an event occurs. The AllScale monitoring events are purely local and it is up to the listener to further process the specific performance counters for an event and make it available to other components.

4 Conclusions and Future Work

The infrastructure and implementation of various performance counters is in place and already used extensively by the AllScale monitoring component. The discrete, AllScale specific events as well as the available performance counters are continuously improved and adapted to arising use cases.