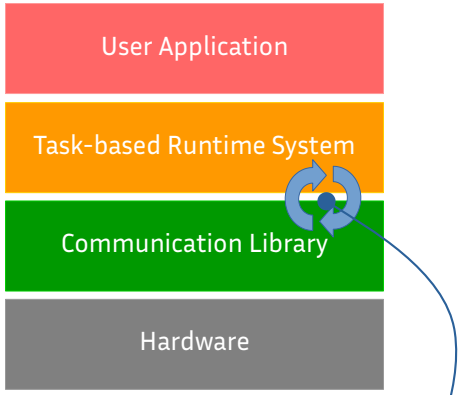
Large scale communications for task-based runtime systems

Philippe SWARTVAGHER
PhD Student at *Inria Bordeaux – Sud-Ouest*

11th JLESC Workshop

Motivation



- Emergence of **distributed task-based runtime systems**

- The runtime system knows many information about current task scheduling, future communications to perform...
- The communication library knows many information about current communications, network state...

How can **both efficiently interact** ?

Thesis topic

Interactions between task-based runtime systems and communication library

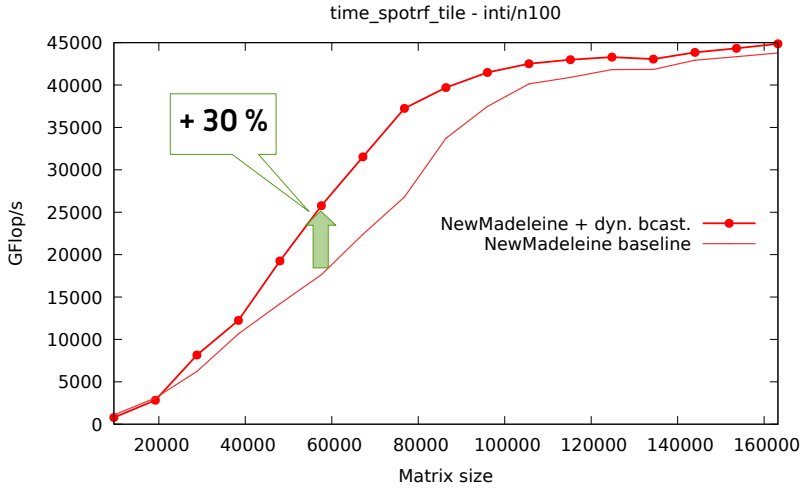
(with scalability in mind)

Supervised by Alexandre DENIS and Emmanuel JEANNOT

Inria

Efficient broadcasts

- In asynchronous dynamic task-based runtime system:
  - A data owned by a node can be needed on several other nodes: a **broadcast**
  - Only the sender node knows all recipients
  - Recipient node ignores if received data is part of a broadcast
  - → `MPI_Bcast` not usable in this case
- We developed **dynamic broacasts**:
  - Use efficient broadcast algorithms
  - Routing informations are stored in message header

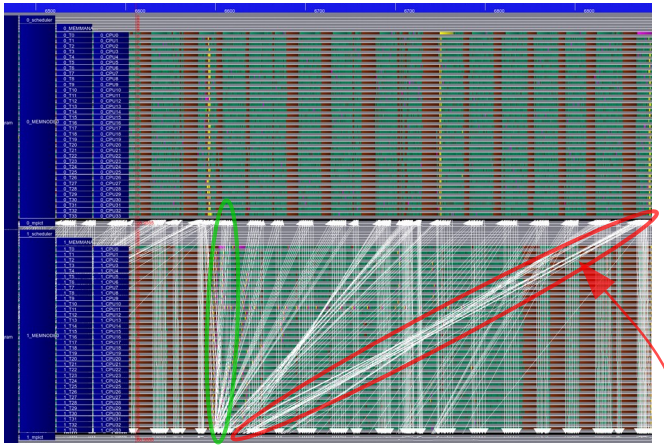


Performance gain of tiled Cholesky decomposition with our *dynamic broadcasts* on 100 nodes (1600 cores).

Denis A., Jeannot E., Swartvagher P., Thibault S. (2020) **Using Dynamic Broadcasts to Improve Task-Based Runtime Performances.**
In: Malawski M., Rzadca K. (eds) Euro-Par 2020: Parallel Processing. Euro-Par 2020. Lecture Notes in Computer Science, vol 12247. Springer, Cham.
https://doi.org/10.1007/978-3-030-57675-2_28

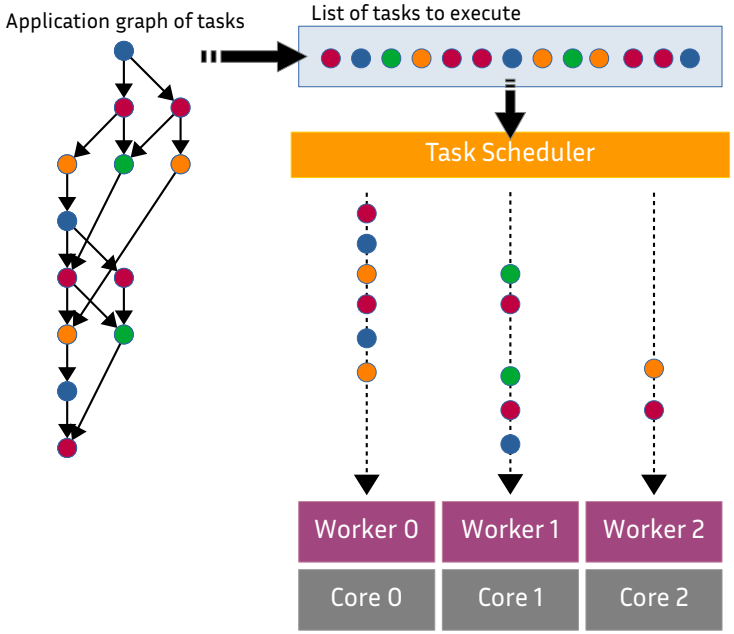
Performance analysis

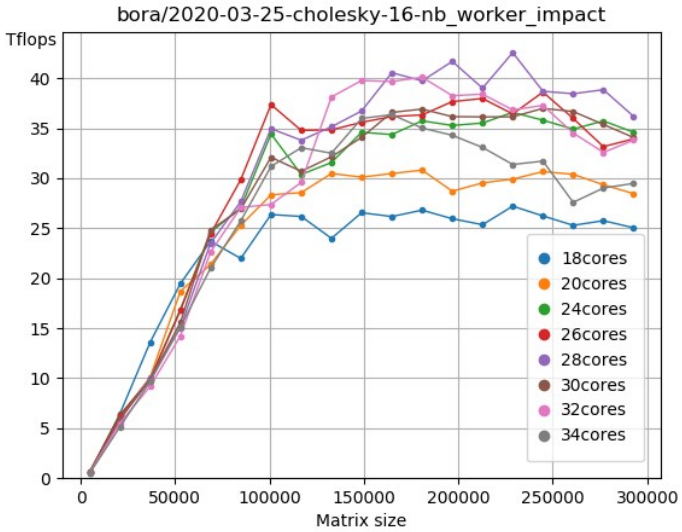
- Still some open questions:
  - > 30% performance improvement: **is it the theoretical expected gain** ?
  - > No performance improvement for big matrices: is it normal ?
- Different performance improvements according to cluster characteristics:
  - > **Which characteristic is important ?**
- Tracing tools:
  - > Can have a strong impact on performance (and thus hide some phenomena)
  - > Give a lot of metrics: which ones are relevant ?
  - > Hard to use on many nodes...
- Anyway: traces showed that, even on two nodes, during task execution, **some communications can be very looooooong**



Trace of a Cholesky decomposition on 2 nodes: some communications are longer than other

Worker scalability



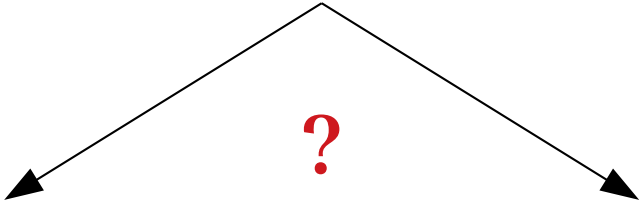


Impact of the number of workers used for Cholesky decomposition on 16 nodes with each 36 cores.

- When using many nodes:
  - > **Using all available workers does not give the best performance**
  - > Dynamic broadcasts behave better with less workers
  - > But works well on a single node → **impact of communications ?**

Competition between communication and computation ?

- During task execution, some communications can be very long
- Using a lot of workers seems to degrade communication performances



Contention with data moving between:
- memory and cores
- memory and NIC

Impact of CPU / uncore / ... frequency variations

Collaboration opportunities

- Precise performance analysis (with light overhead)
- Simulation / measure of data movement within a node
- Demystify interactions between computation and communication
- Analyze and improvement of application with challenging communication patterns

→ Feel free to contact me ! philippe.swartvagher@inria.fr