



**HAL**  
open science

# Experiments for Assessing Computation/Communication Overlap of MPI Nonblocking Collectives

Alexandre Denis, Julien Jaeger, Emmanuel Jeannot, Florian Reynier

► **To cite this version:**

Alexandre Denis, Julien Jaeger, Emmanuel Jeannot, Florian Reynier. Experiments for Assessing Computation/Communication Overlap of MPI Nonblocking Collectives. [Research Report] RR-9367, Inria & Labri, Univ. Bordeaux; CEA, CEA/DAM/DIF, Bruyères-le-Châtel, France. 2020, pp.39. hal-03012097

**HAL Id: hal-03012097**

**<https://inria.hal.science/hal-03012097>**

Submitted on 18 Nov 2020

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial - NoDerivatives 4.0  
International License



# Experiments for Assessing Computa- tion/Communication Overlap of MPI Nonblocking Collectives

Alexandre Denis Julien Jaeger Emmanuel Jeannot Florian Reynier

**RESEARCH  
REPORT**

**N° 9367**

October 2020

Project-Team TADaam





# Experiments for Assessing Computation/Communication Overlap of MPI Nonblocking Collectives

Alexandre Denis\* Julien Jaeger† Emmanuel Jeannot\* Florian  
Reynier†

Project-Team TADaaM

Research Report n° 9367 — October 2020 — 36 pages

**Abstract:** We present experimental results for evaluating non-blocking MPI collectives. We compute several metrics to assess the efficiency of the overlap for different MPI library, with different configurations and for different hardware.

**Key-words:** MPI nonblocking collectives, evaluation metrics, experiments

---

\* Inria Bordeaux — Sud-Ouest

† CEA/DAM/DIF

**RESEARCH CENTRE  
BORDEAUX – SUD-OUEST**

200 avenue de la Vieille Tour  
33405 Talence Cedex

## Expériences pour évaluer le recouvrement calcul/communication des collectives non-bloquantes de MPI

**Résumé :** Nous présentons les résultats d'expériences pour l'évaluation des collectives non bloquantes dans MPI. Nous calculons différentes métriques pour évaluer l'efficacité du recouvrement pour différentes bibliothèques MPI avec différentes configurations et pour différents types de matériels.

**Mots-clés :** MPI, Collectives non bloquantes, métriques d'évaluation, expériences

## 1 Introduction

In order to execute efficiently parallel applications on large-scale supercomputers, such applications are often programmed using the Message Passing Interface (MPI) standard. MPI uses the SPMD (Simple Program Multiple Data) model to describe how each process composing the applications are communicating. MPI features diverse primitives such as point-to-point communication or collective operations. Collective operations are a set of abstract routines enabling a group of processes to execute in an efficient and concise way a sequence of point-to-point calls. Since the version 3.0 of the MPI standard, all the collective operations of MPI have a nonblocking version. This means that, when a nonblocking collective communication call is made, the data exchanges can continue in the background after the return of the call; the application can continue its execution until an explicit call to a completion function.

Using nonblocking collectives allows the application to execute independent computations between the collective initiation call and the completion call (*e.g.* `MPI_Wait`). Thanks to this, it should be possible to hide the communication behind computation and to save time through computation/communication overlap. However, for such overlap to effectively happen, it is required that the communications *progress* in the background while the application continues to execute its computation. Without this background progress, the MPI library is not able to execute the chosen collective algorithm and the data transfer can only happen when the application explicitly calls other MPI routines.

However, implementing a progression engine for collective operation is much more difficult than for point-to-point communication as such collective may require to implement a complex algorithm (*e.g.* diffusion tree) involving computation (*e.g.* for reducing values). This also requires to carefully dedicate resources to execute the engine. A good balance between what is devoted to the progression and what to the application computation is key for performance. Hence, MPI library developers are struggling to implement such efficient progress engine. Therefore, in the literature, up to now, few applications have been reported to use MPI nonblocking collective.

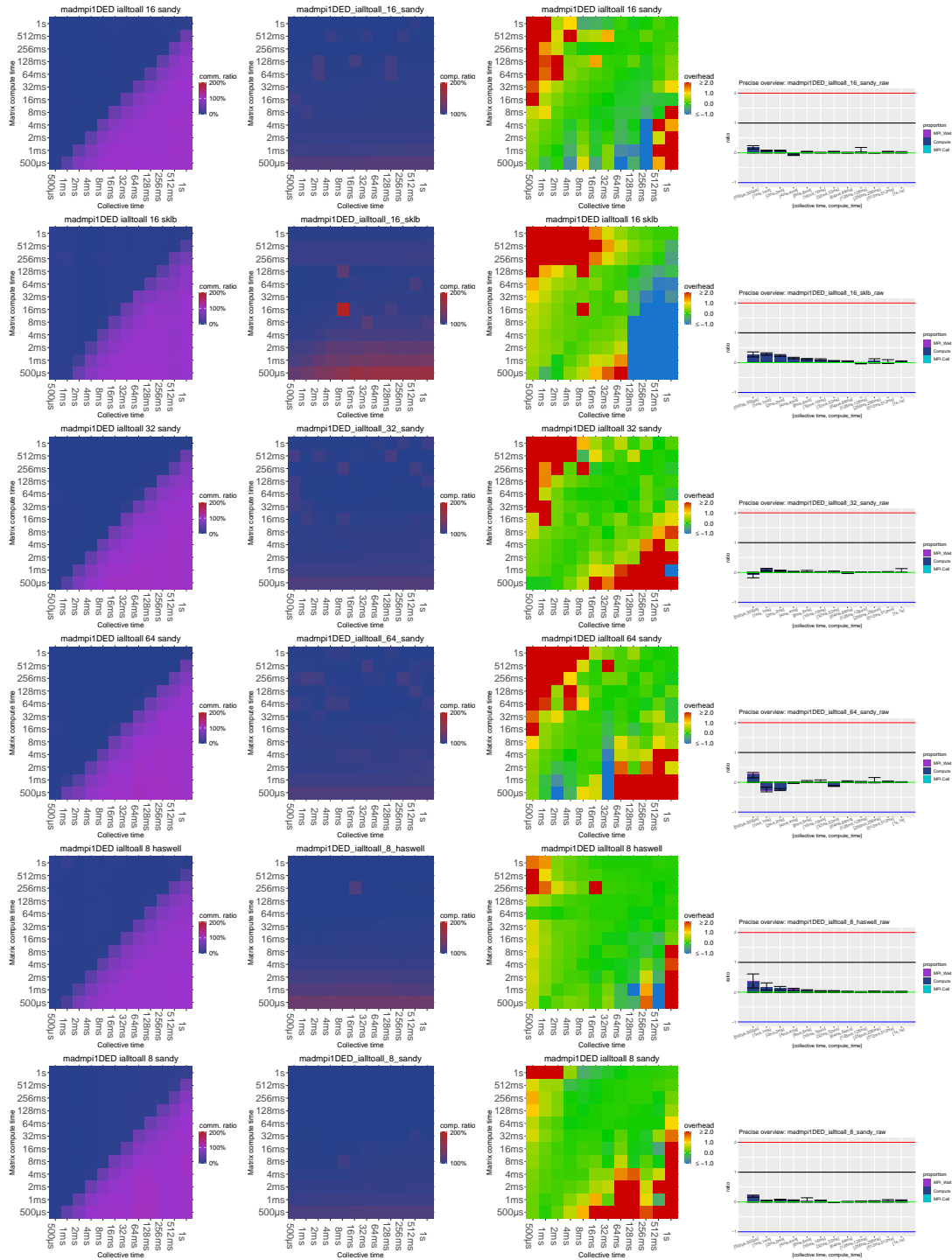
The goal of this report is actually to assess the quality of the progression engine of different MPI implementations in order to objectively determine if computation/communication overlap can efficiently be achieved by using nonblocking collective communications. Based on a set of metrics, we evaluate different MPI libraries on different nonblocking collective routine on several popular MPI implementations to evaluate how the communication/computation overlap behaves in such implementations.

This report contains experiments of of a paper submitted to IPDPS 2020. Metrics presentation and discussion of the results are done in this paper.

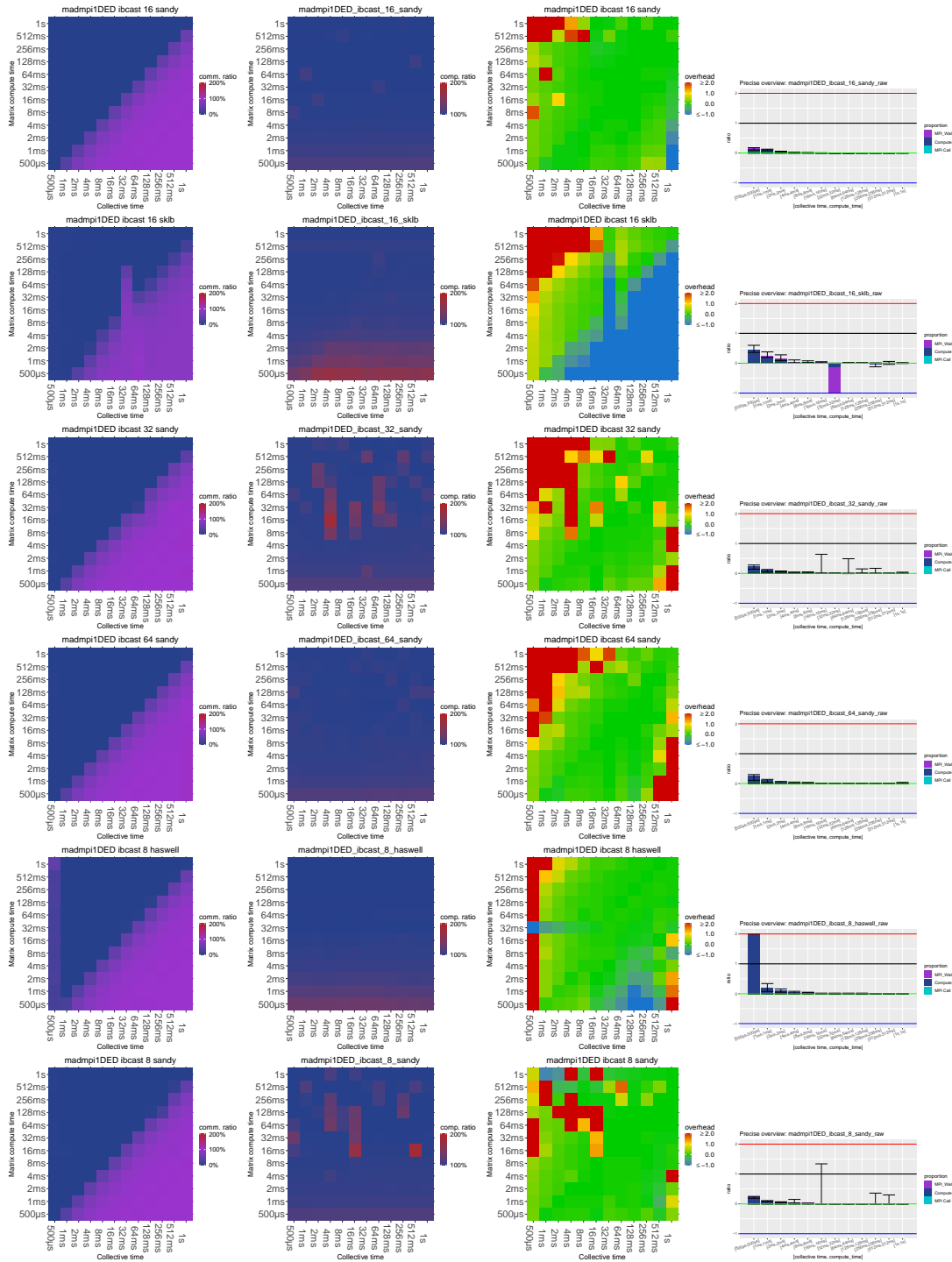
## 2 MadMPI with dedicated core

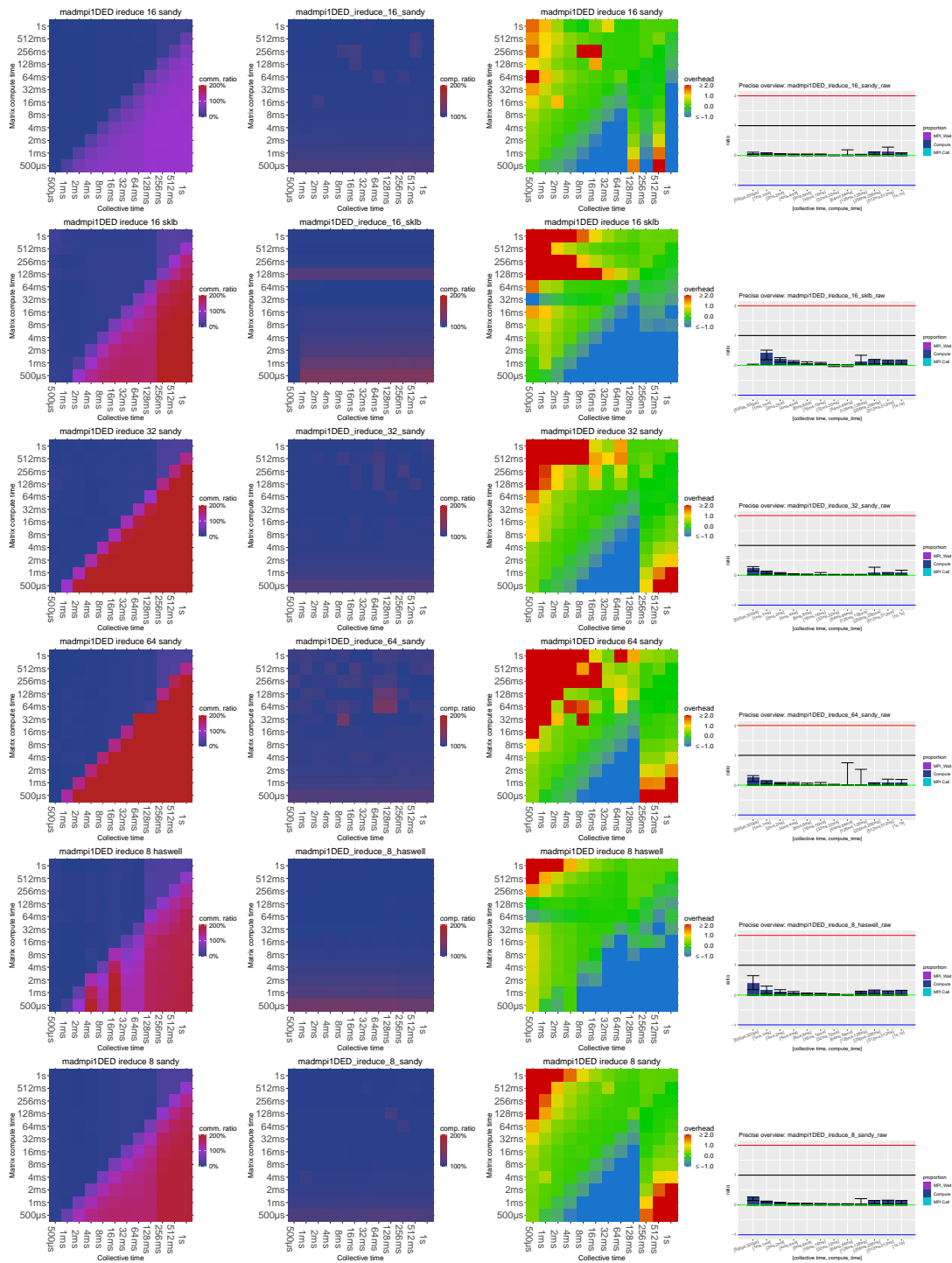
This section shows the results obtained with MadMPI using a dedicated core. The dedicated is enabled using the environment variable `PIOM_DEDICATED=1`. The network used is Infini-band.





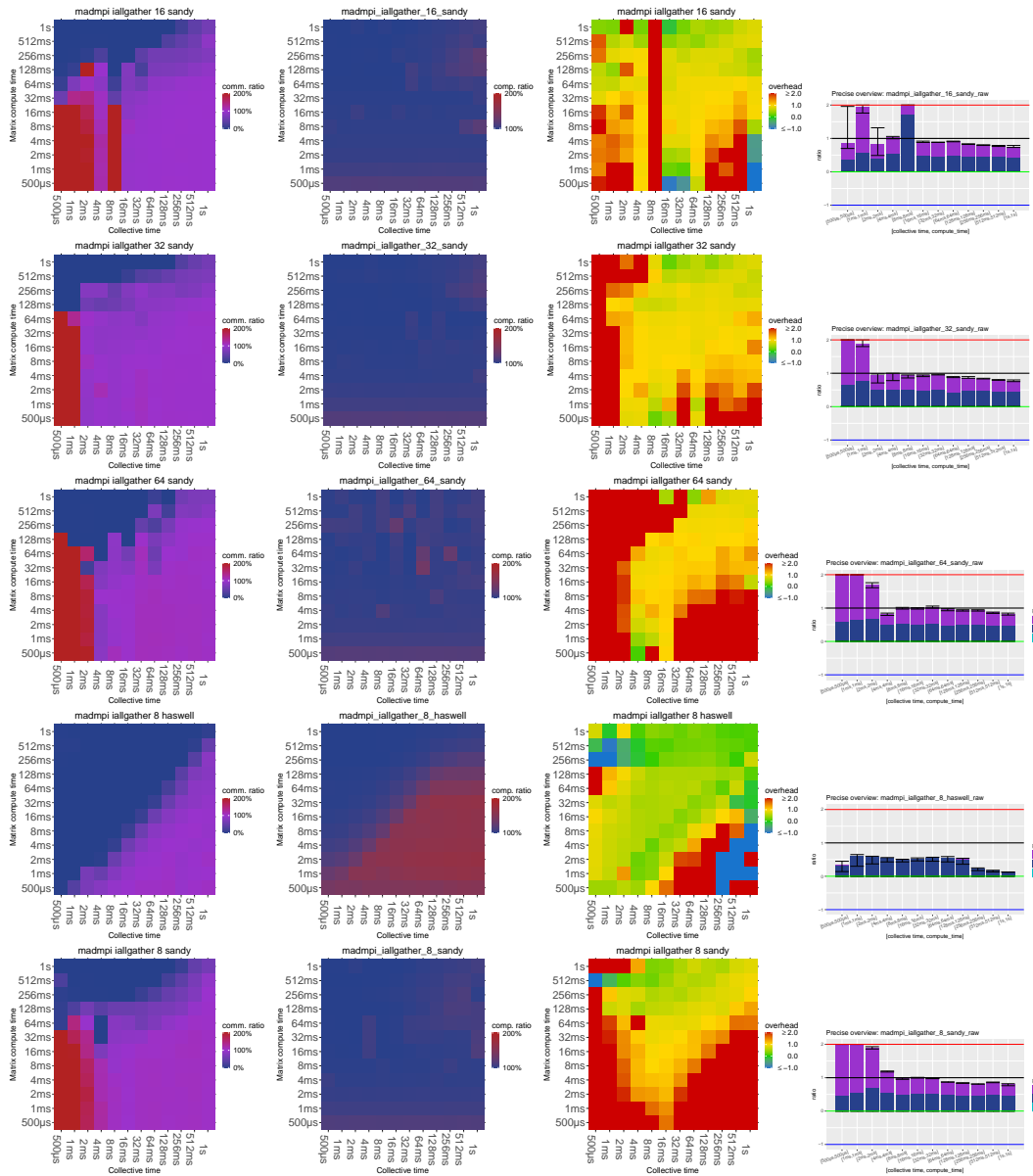


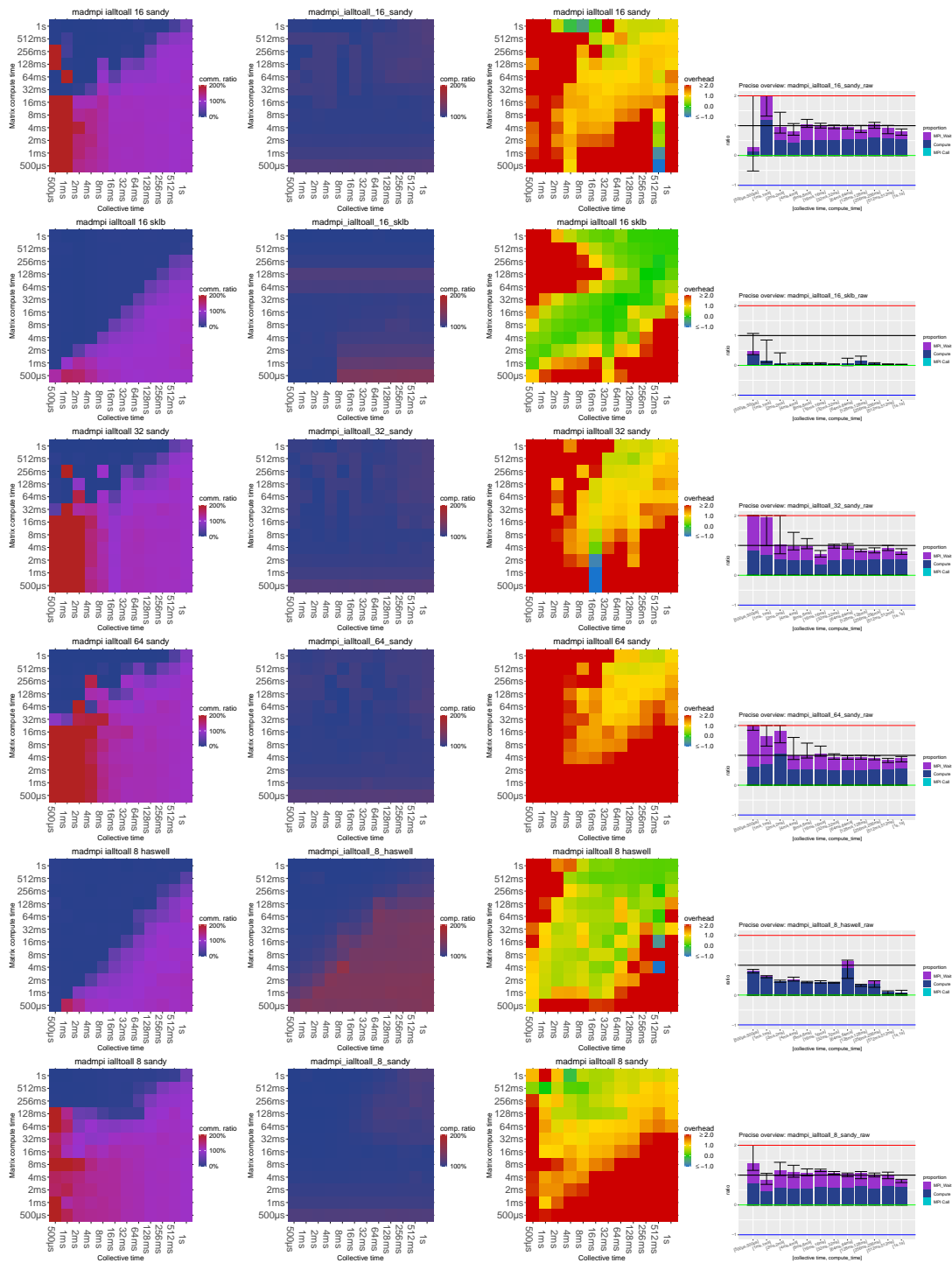


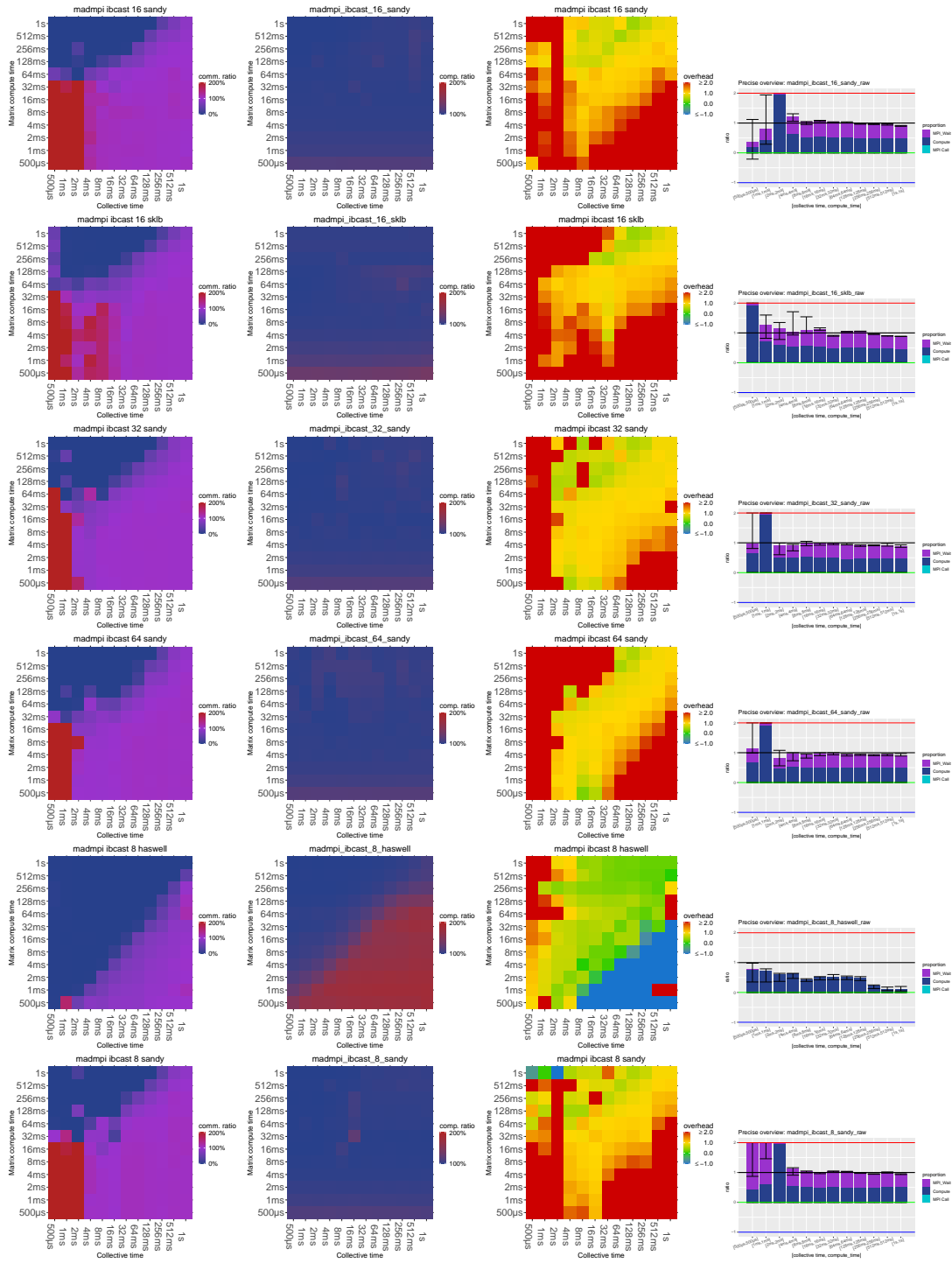


### 3 MadMPI

The following results are obtained using MadMPI with the default configuration. This is also executed on an Infiniband network.



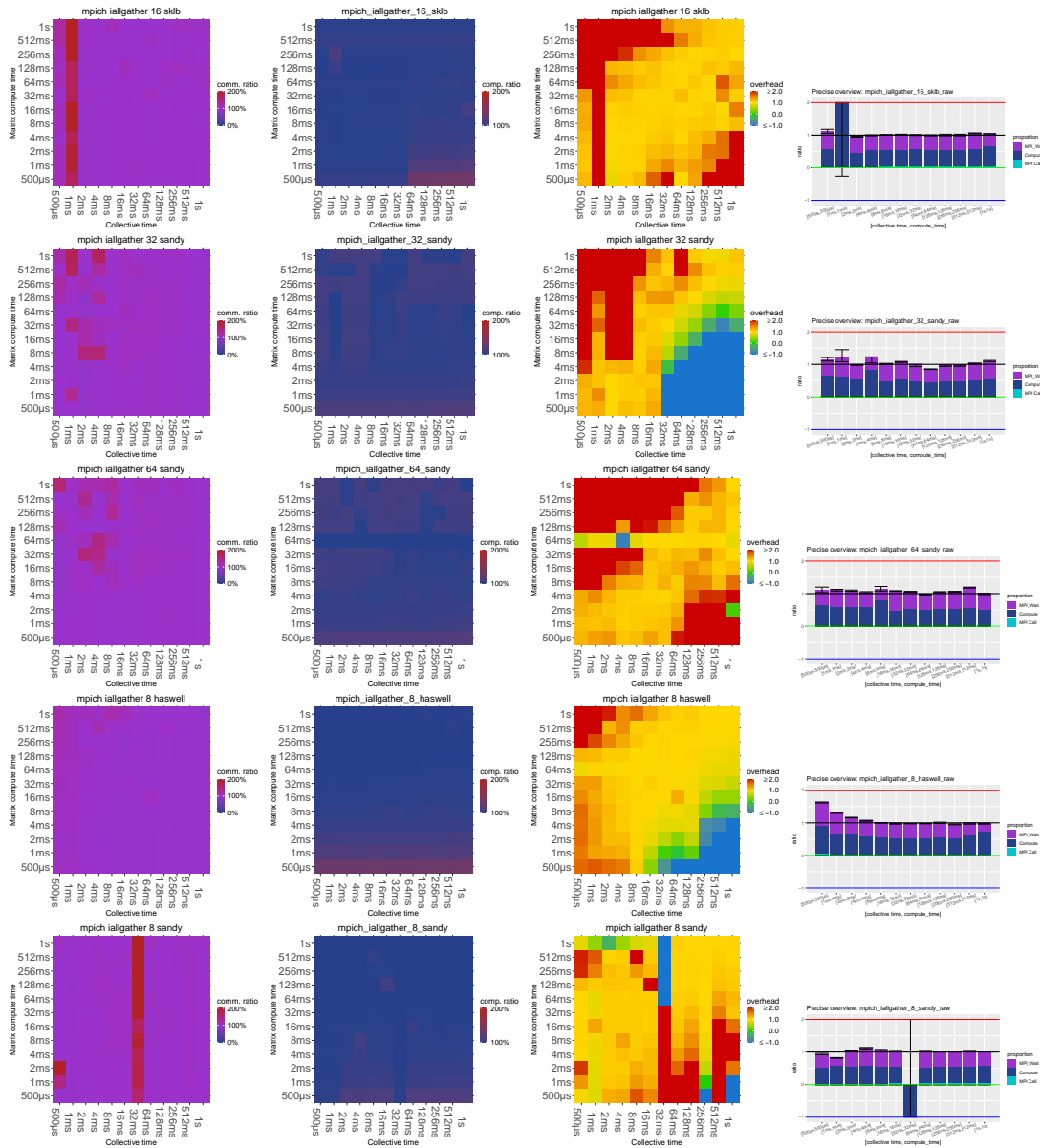


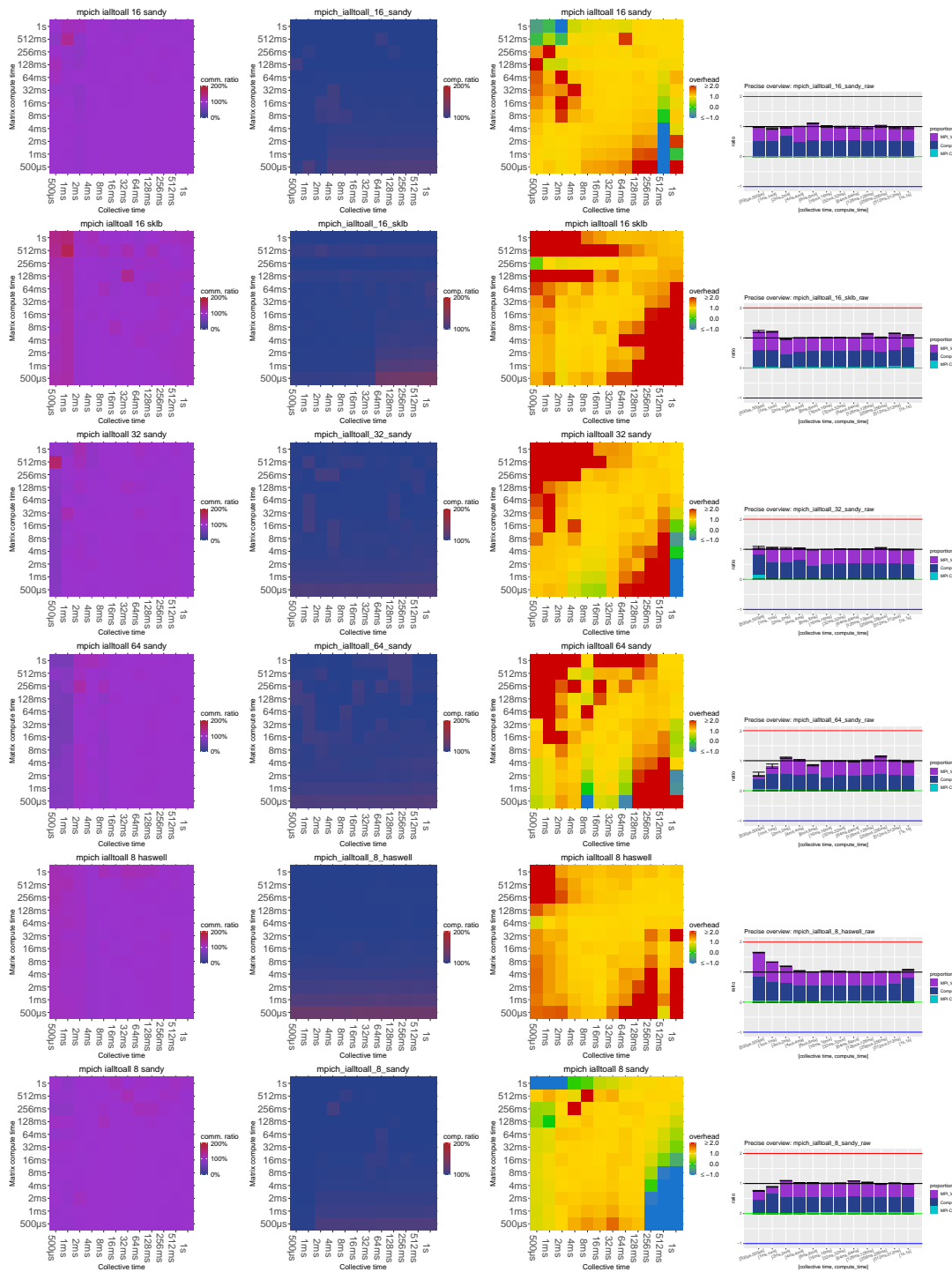




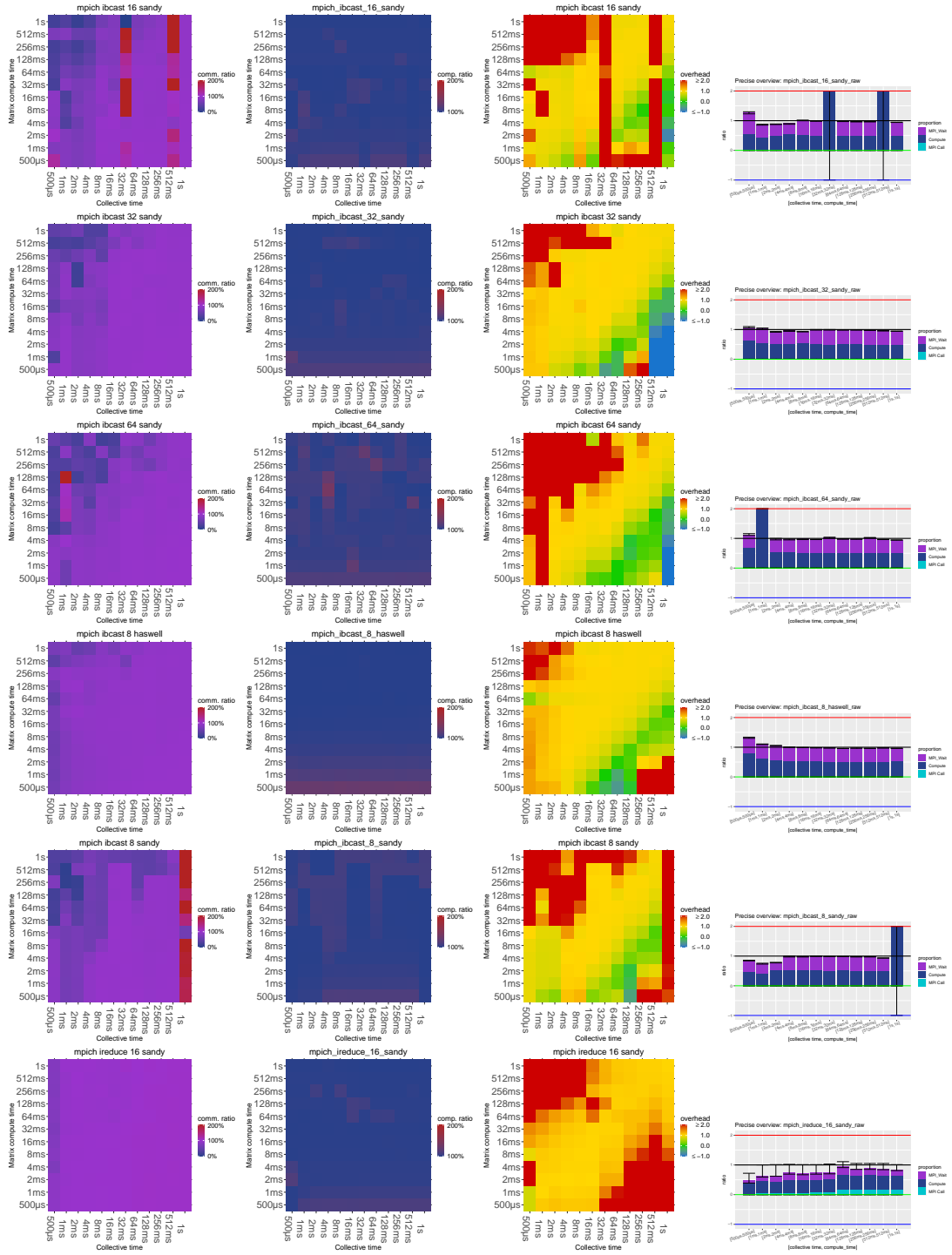
## 4 MPICH

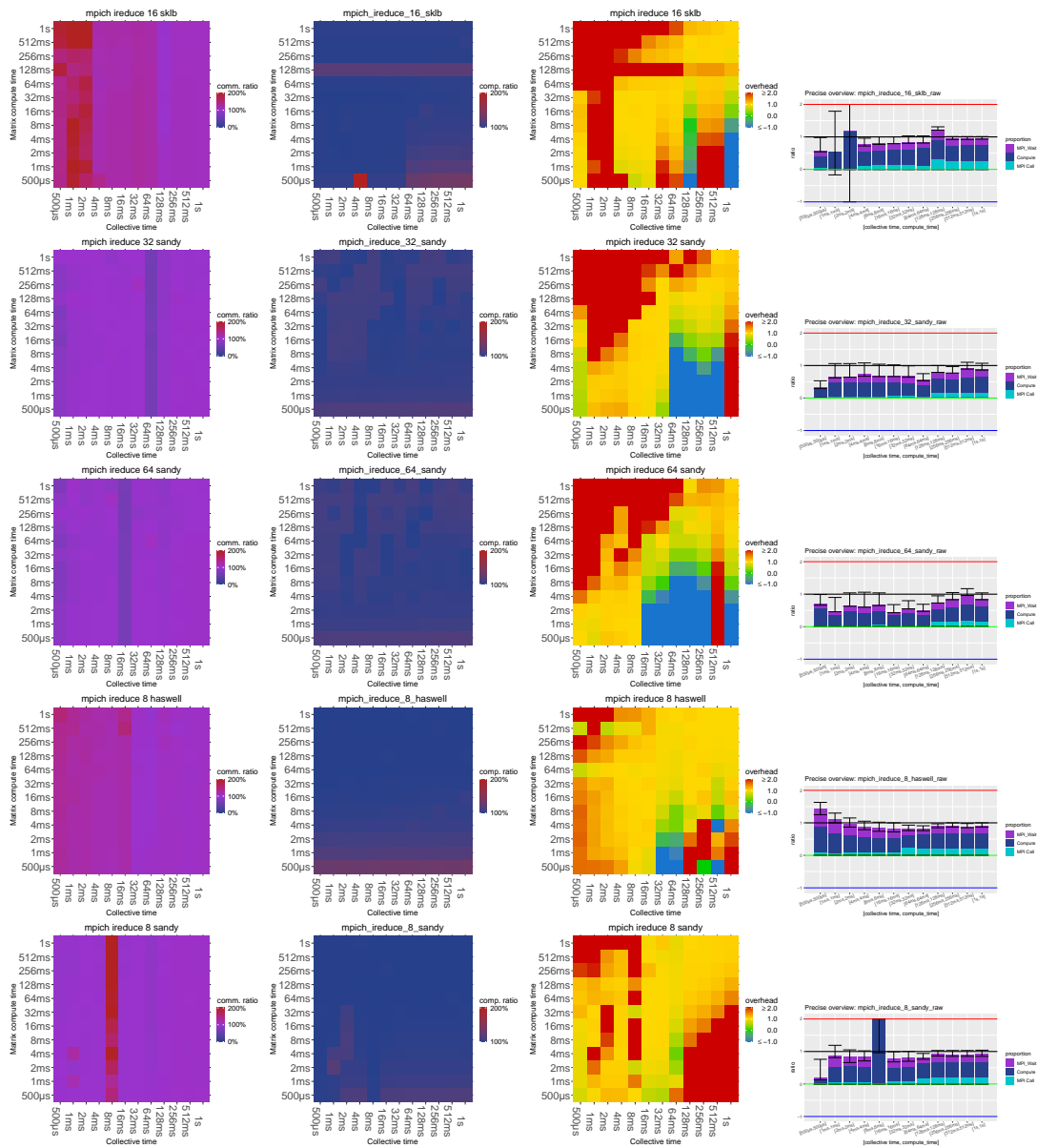
These experiments are done using MPICH 3.3 with the default configuration. The network used is Infiniband.





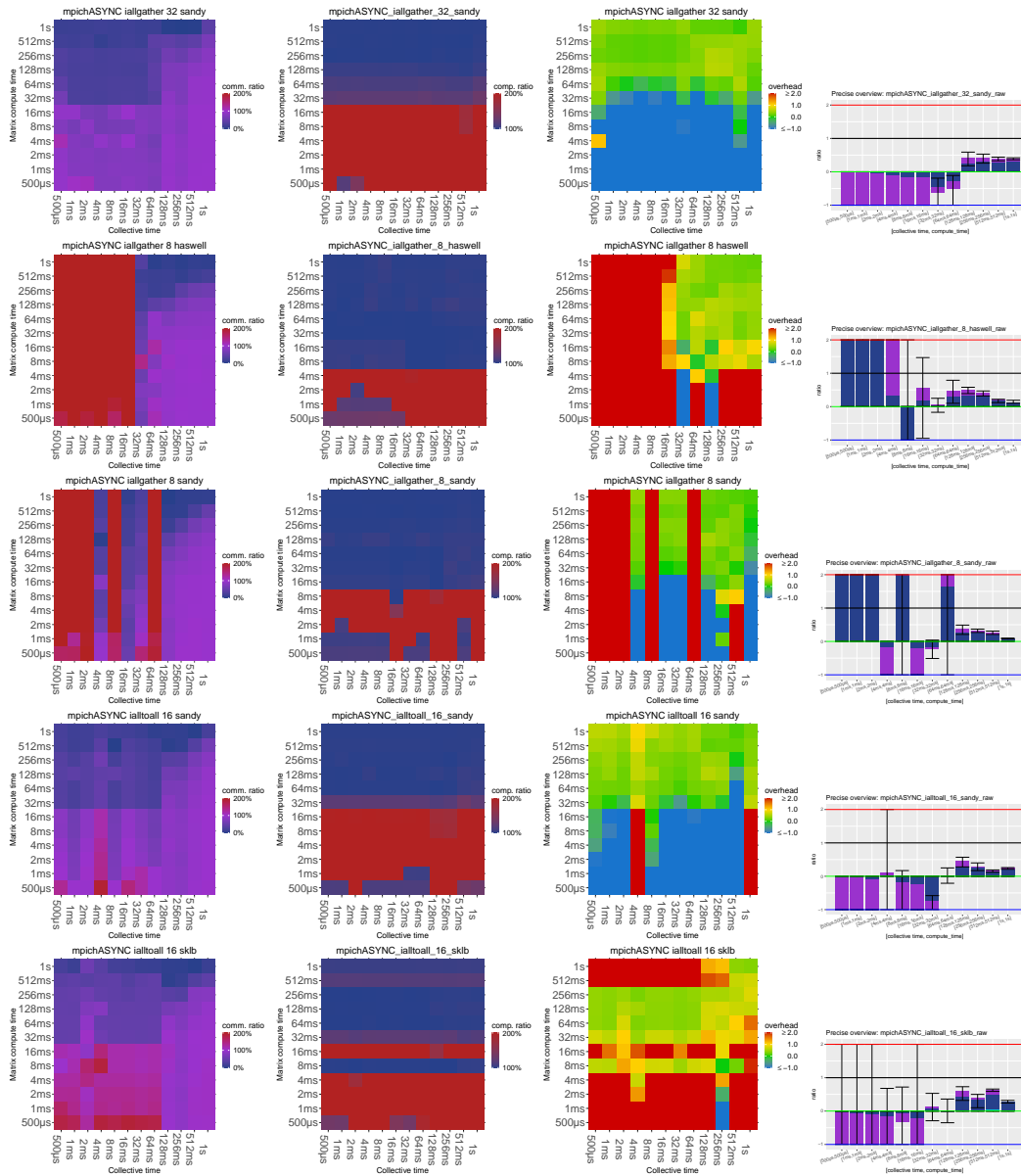


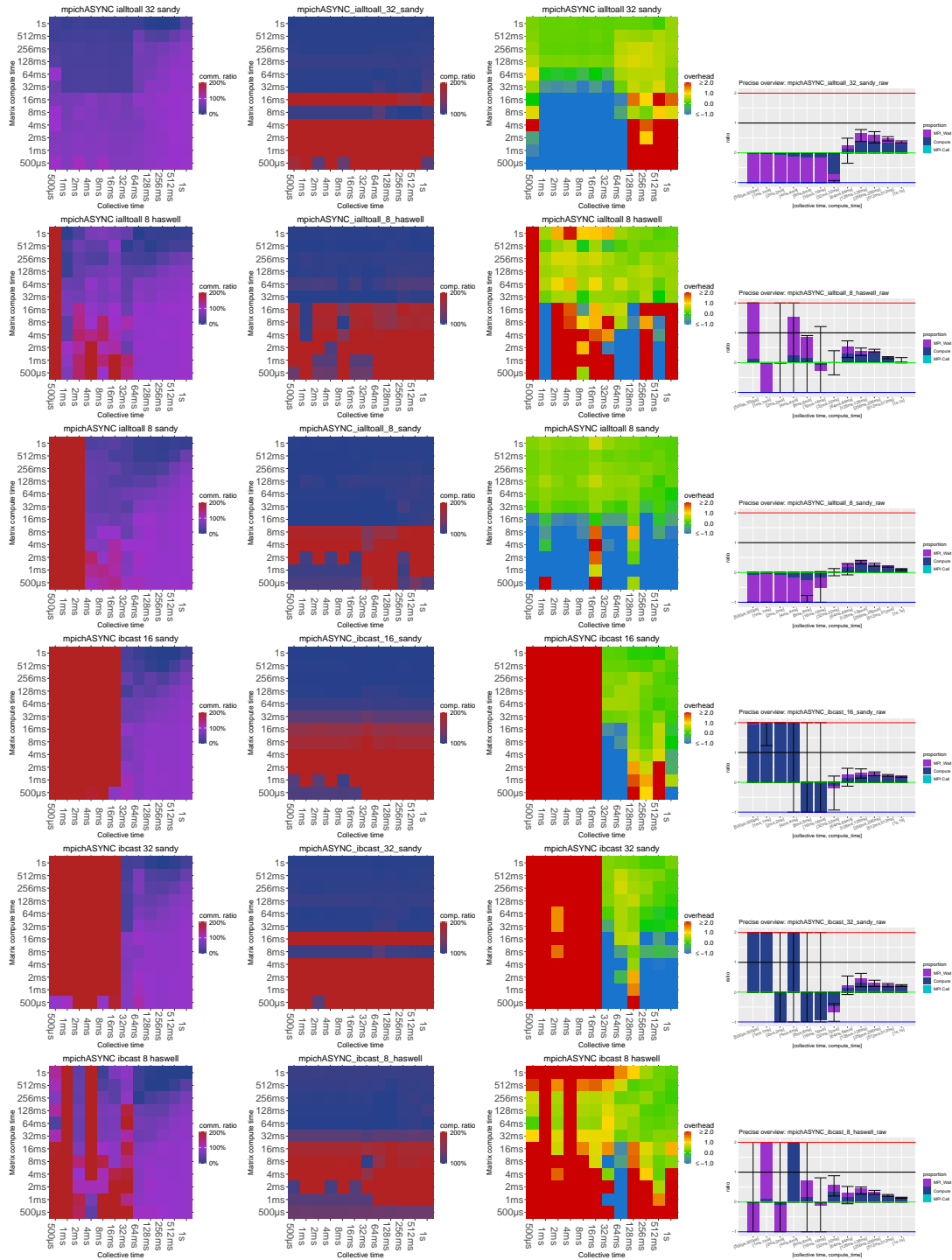


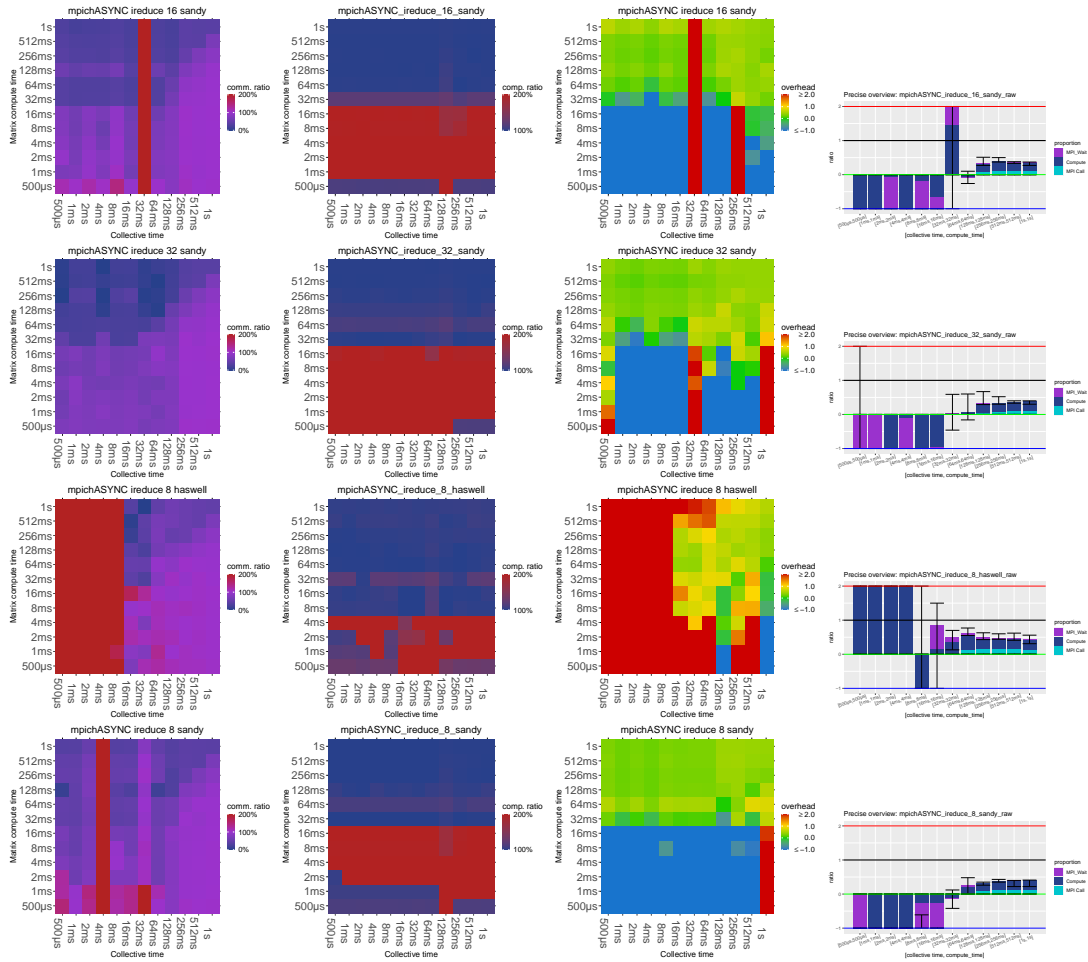


## 5 MPICH with progress thread

The same version (MPICH 3.3) and same network(Infiniband) is used here. We enabled the progress thread using `MPICH_ASYNC_PROGRESS=1`. No core is dedicated to the progress thread.

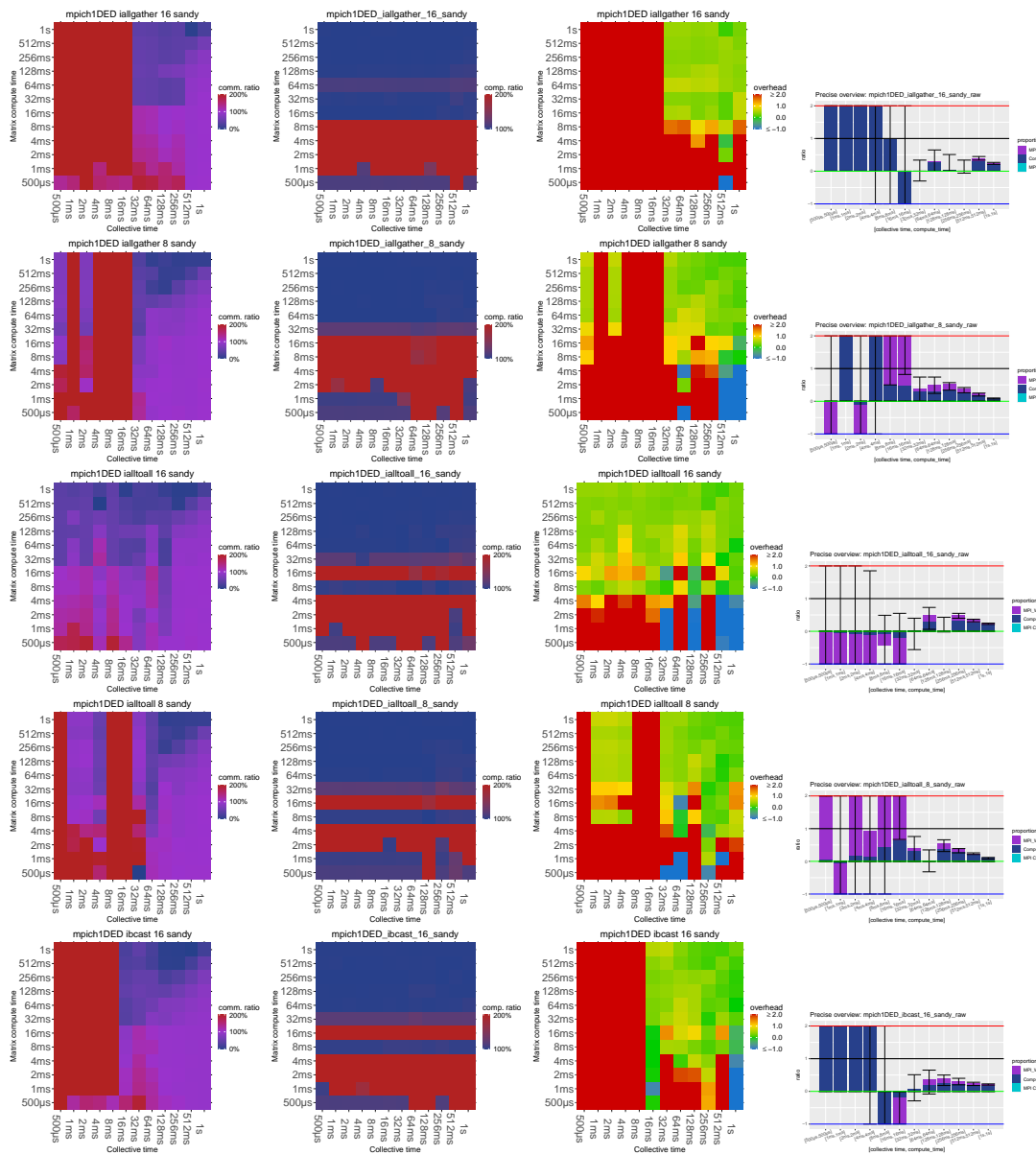


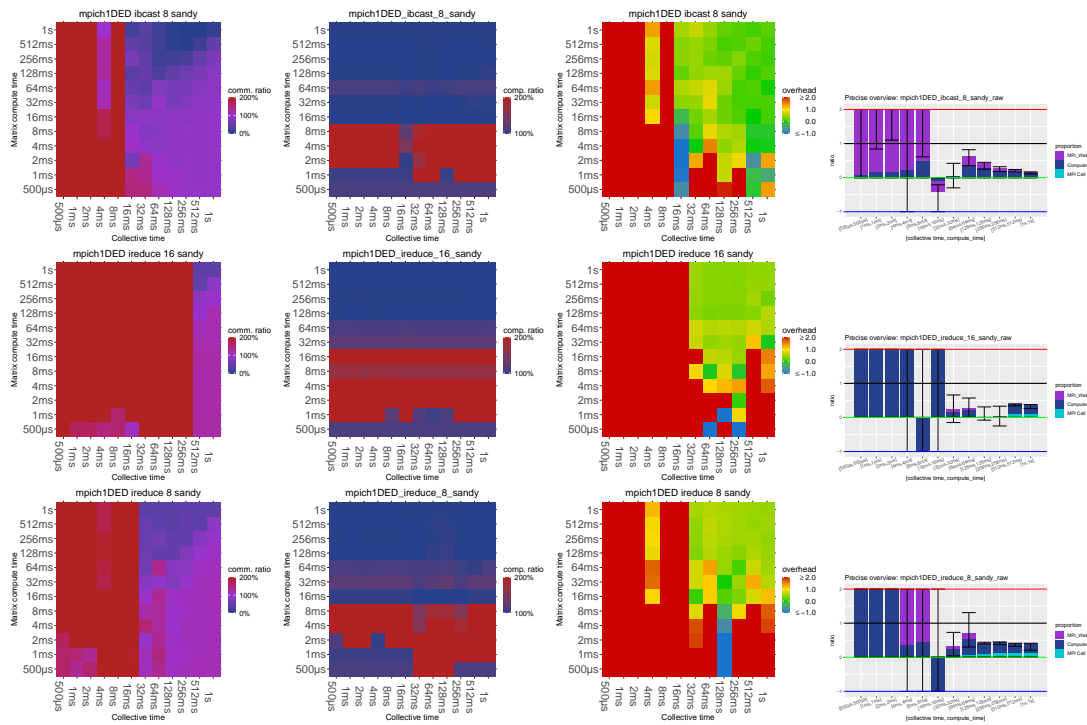




## 6 MPICH with dedicated core

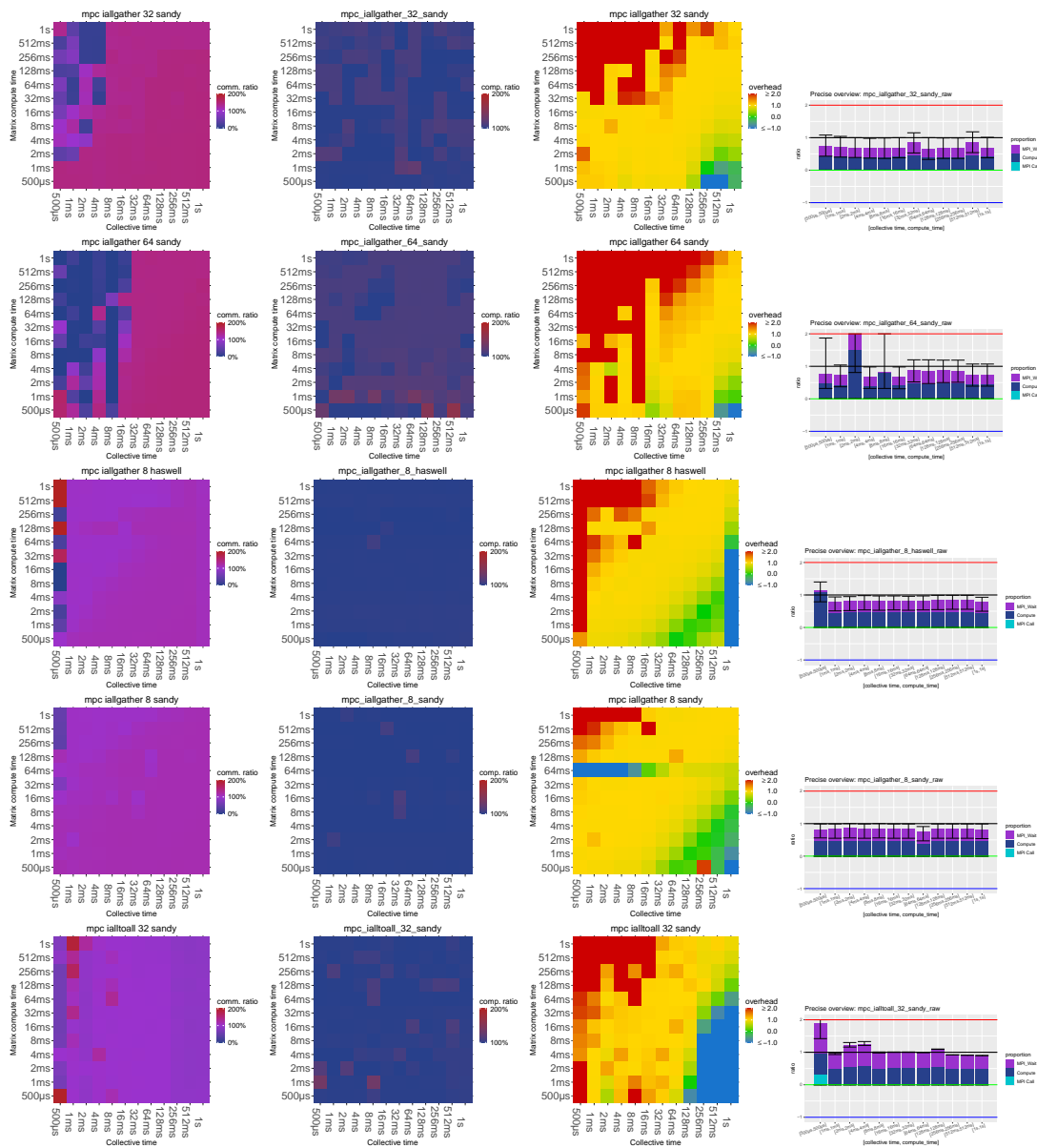
The same version (MPICH 3.3) and same network(Infiniband) is used here. We enabled the progress thread using `MPICH_ASYNC_PROGRESS=1`. We dedicate a core for the progression. The progress thread placement is done by MPICH.



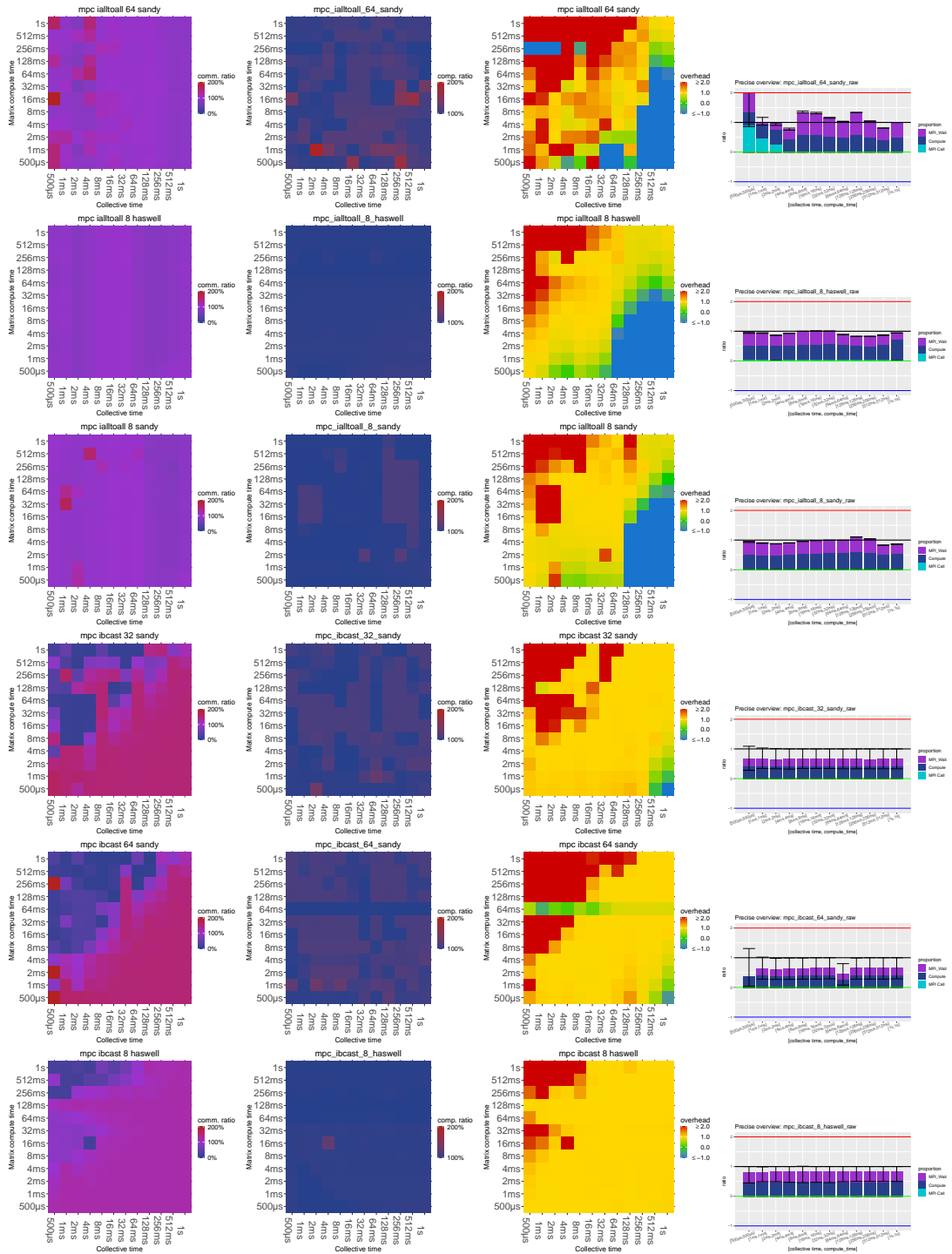


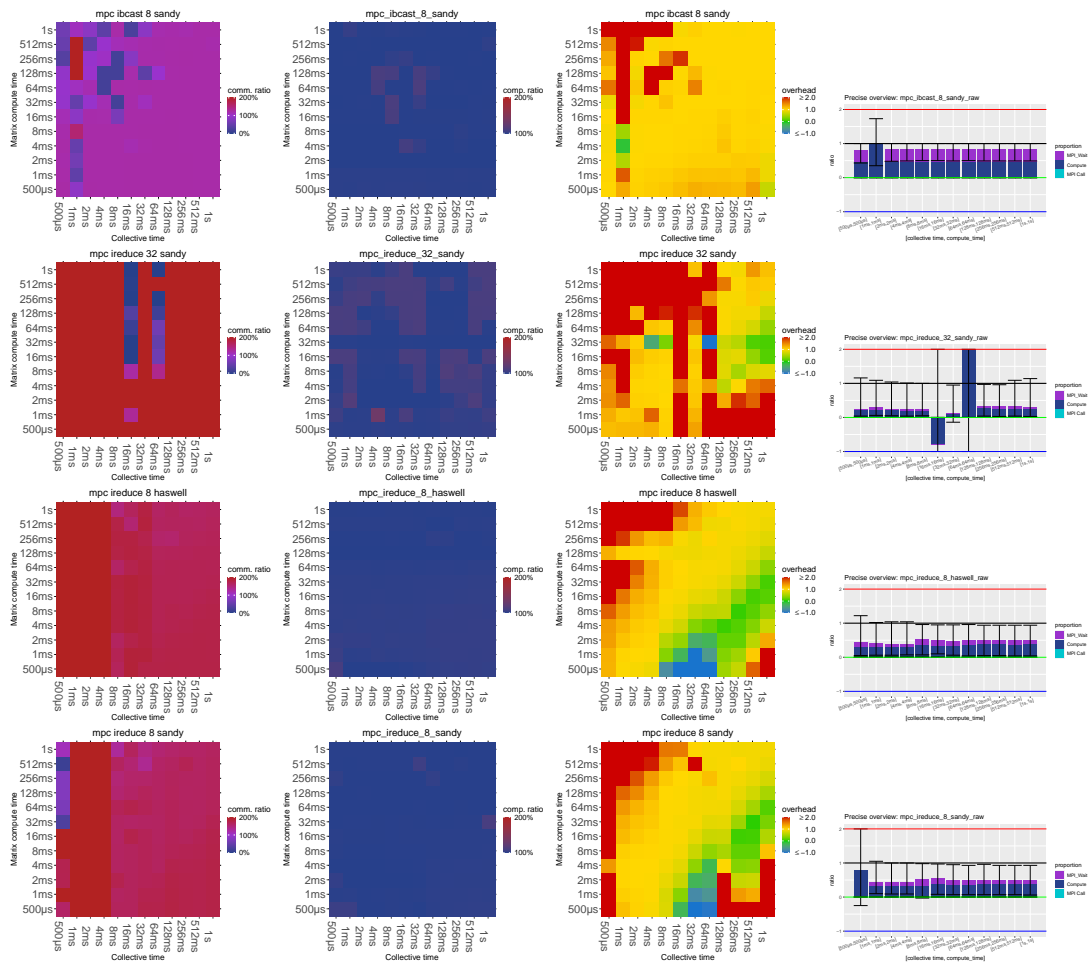
## 7 MPC

These results are done using MPC on an Infiniband network. We used the default configuration.



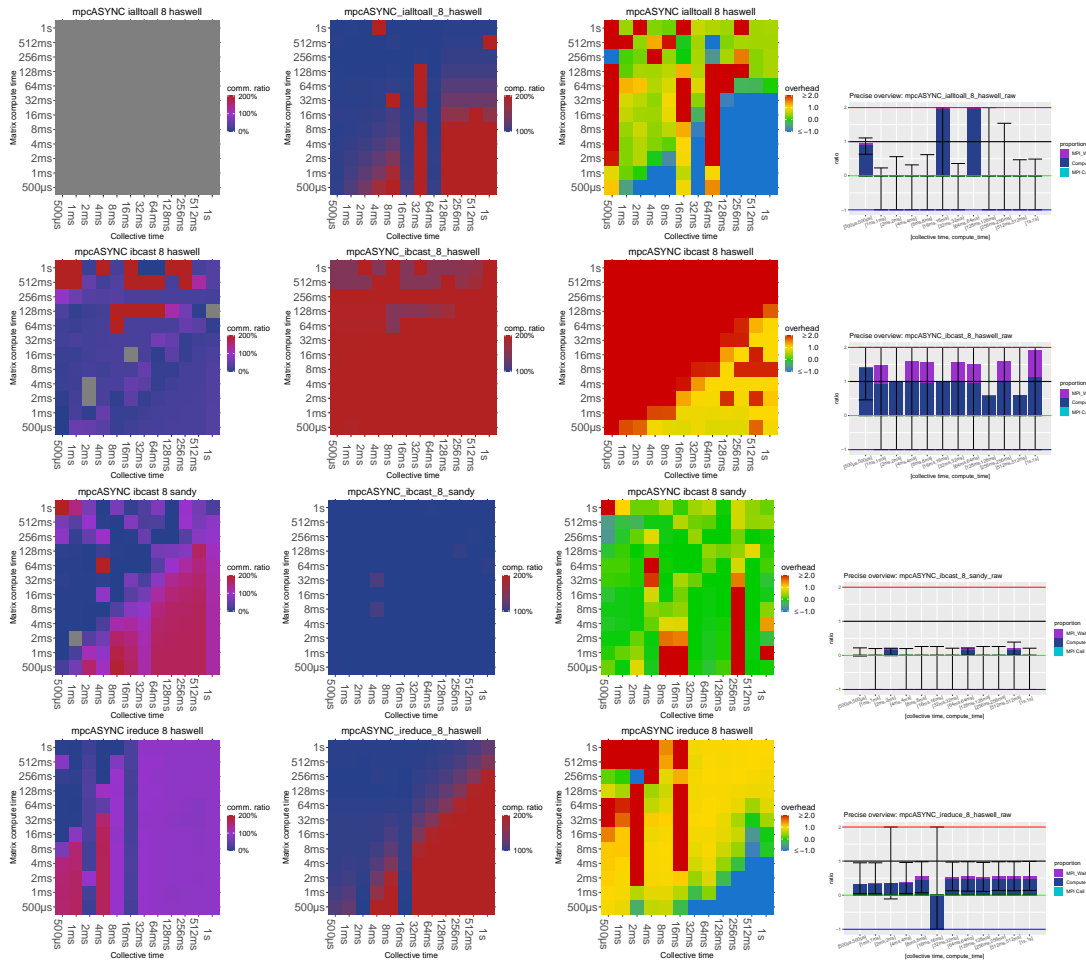






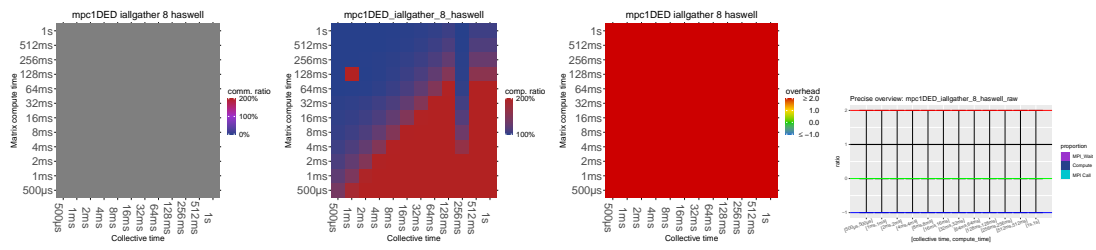
## 8 MPC with progress thread

These results are done using MPC on an Infiniband network. We enabled the progress thread. We didn't dedicate core for the progress thread.



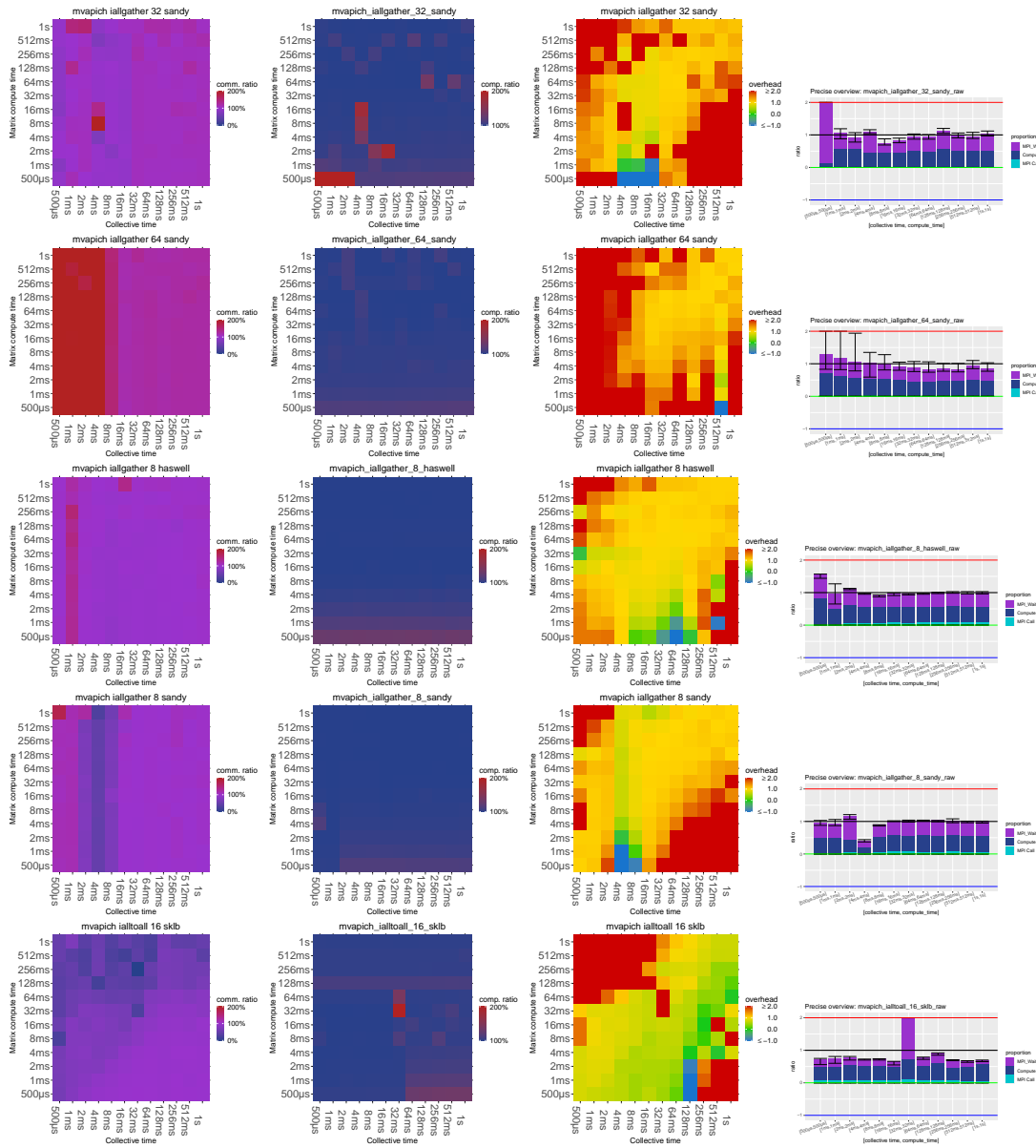
## 9 MPC with dedicated core

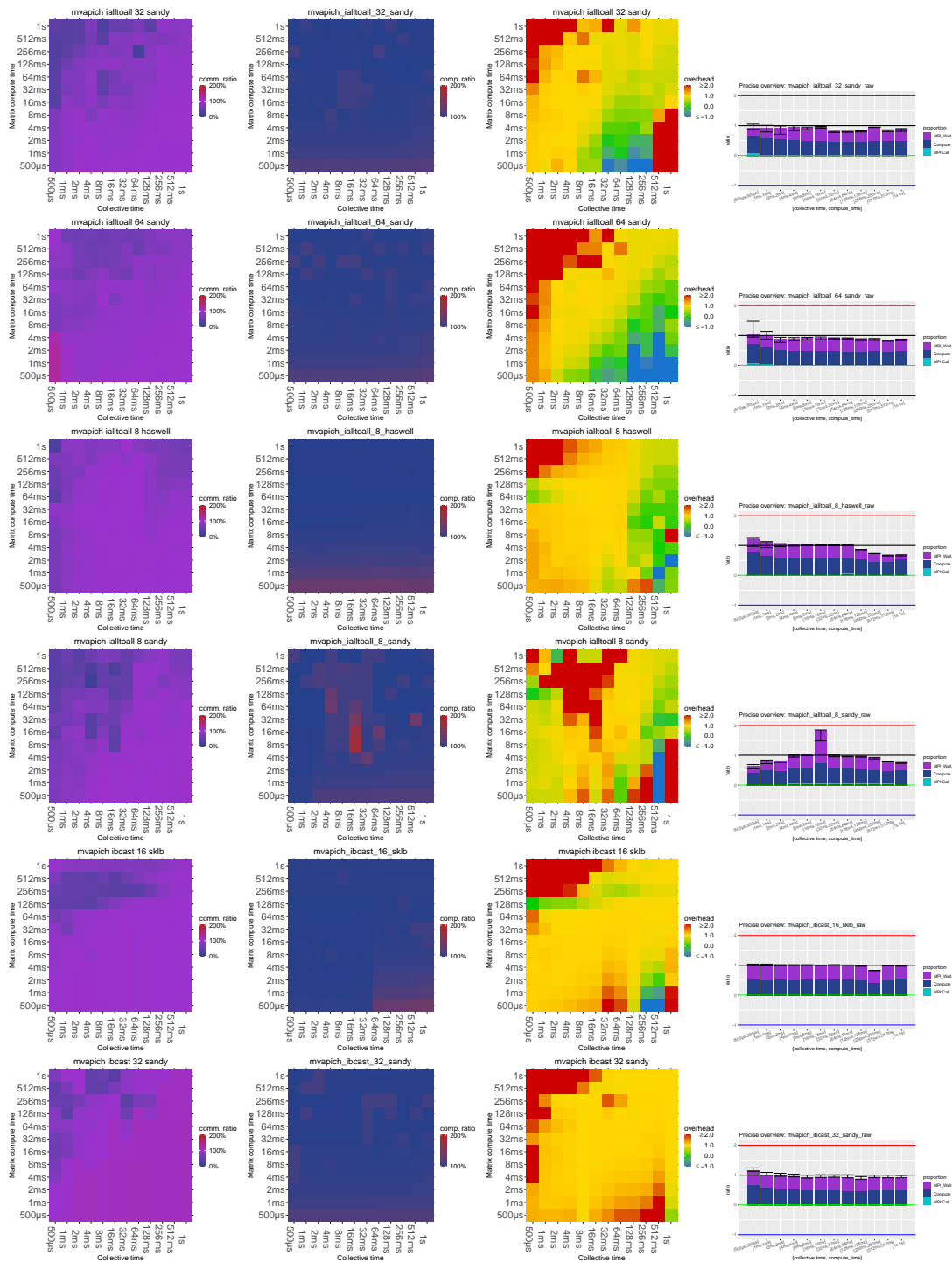
These results are done using MPC on an Infiniband network. We enabled the progress thread. A core is dedicated for the progress thread. The thread is manually bound on this core.

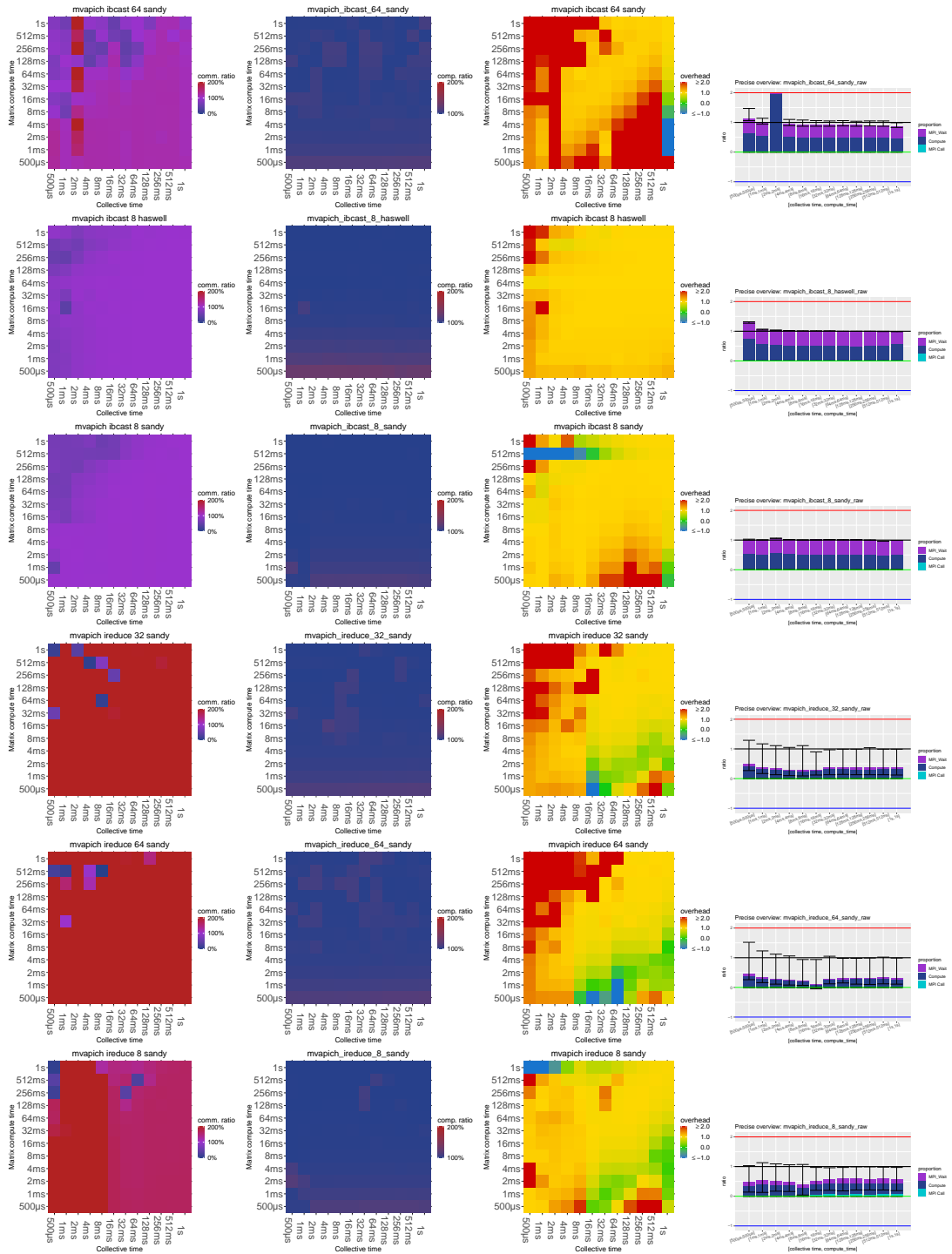


# 10 MVAPICH

These experiments have been done with MVAPICH 3.2.1 on an Infiniband network. We used the default configuration.

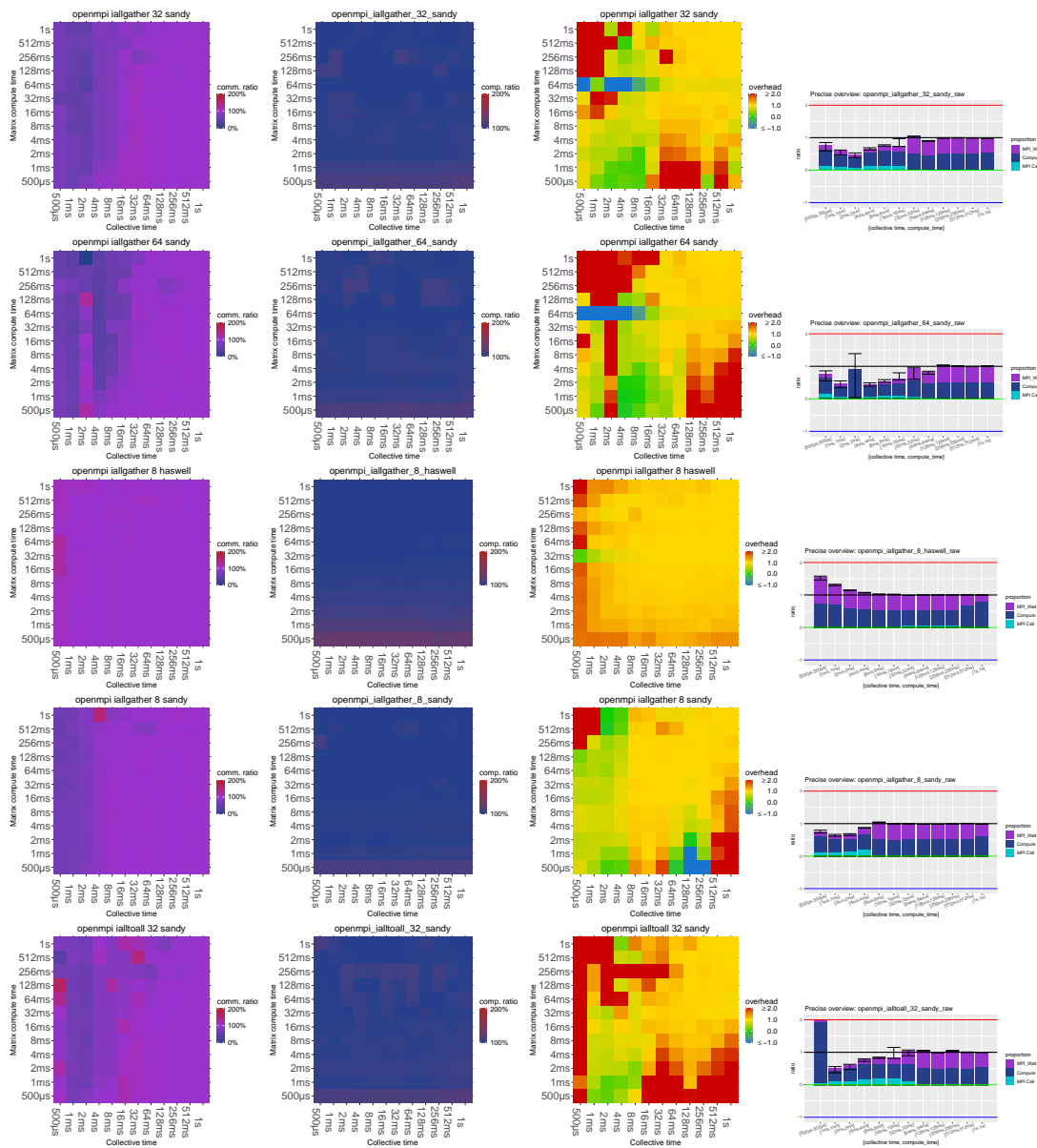




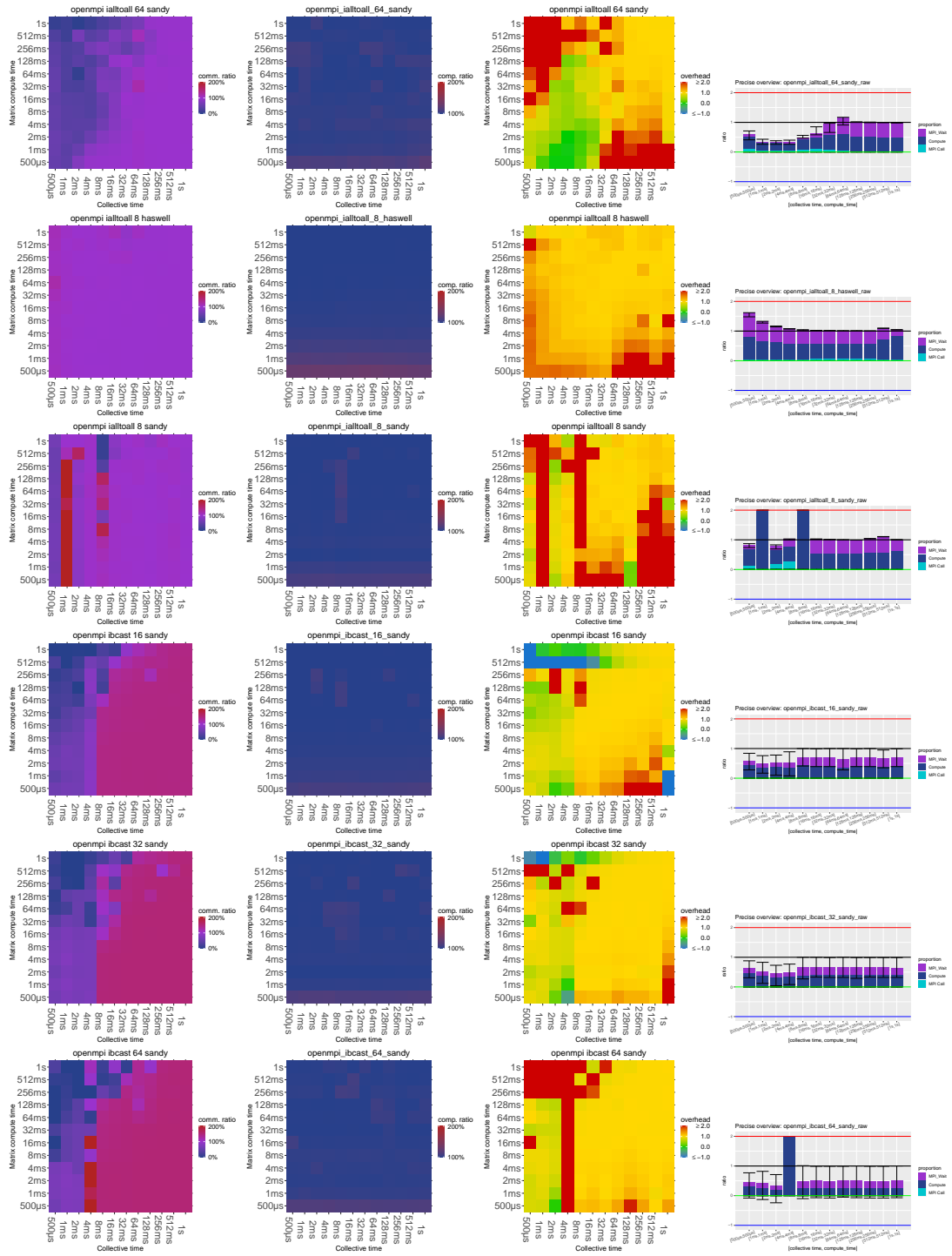


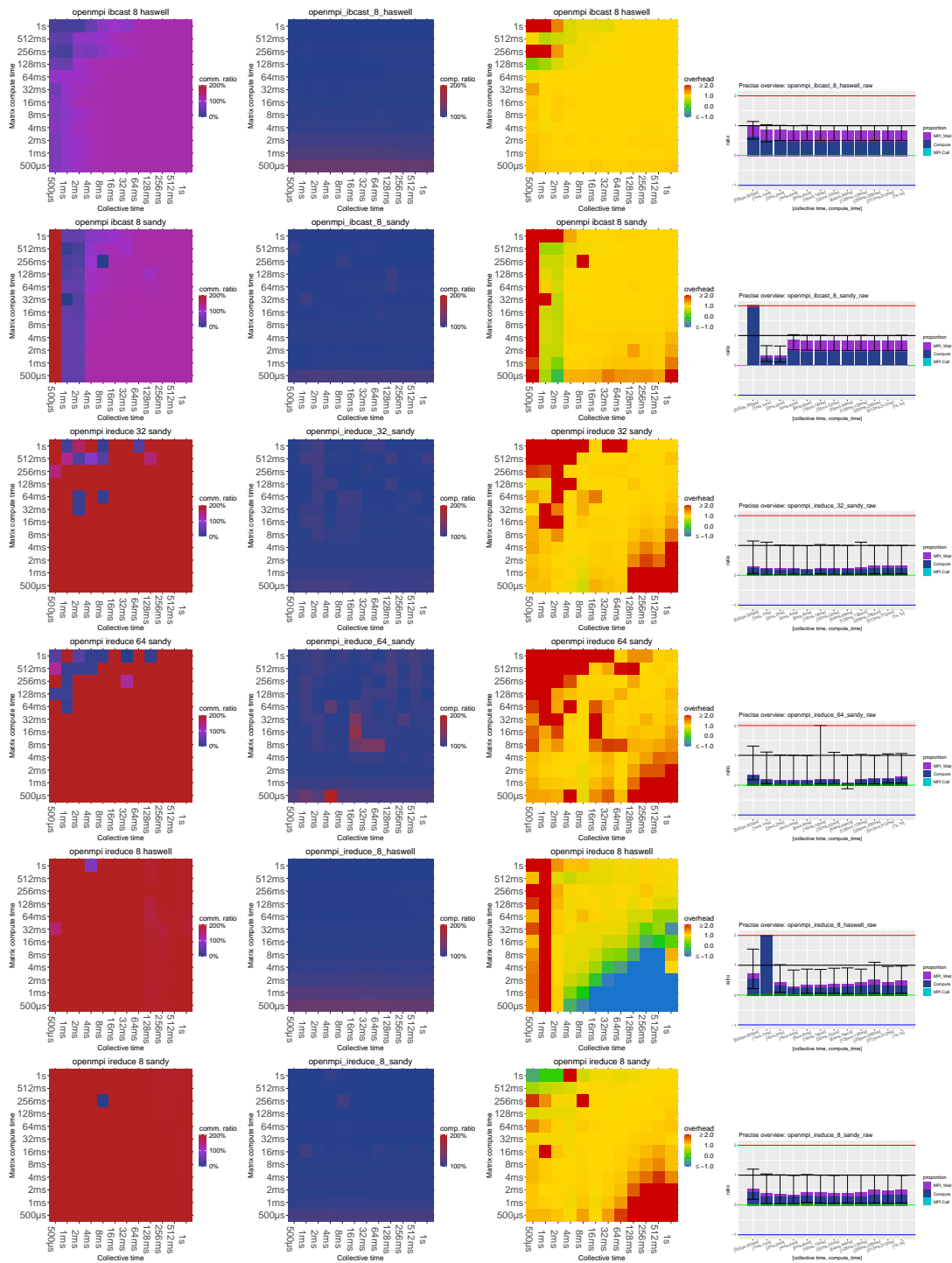
# 11 Open MPI

The MPI runtime used here is Open MPI 4.0.2. We ran the benchmark on an Infiniband network. We used the default configuration.



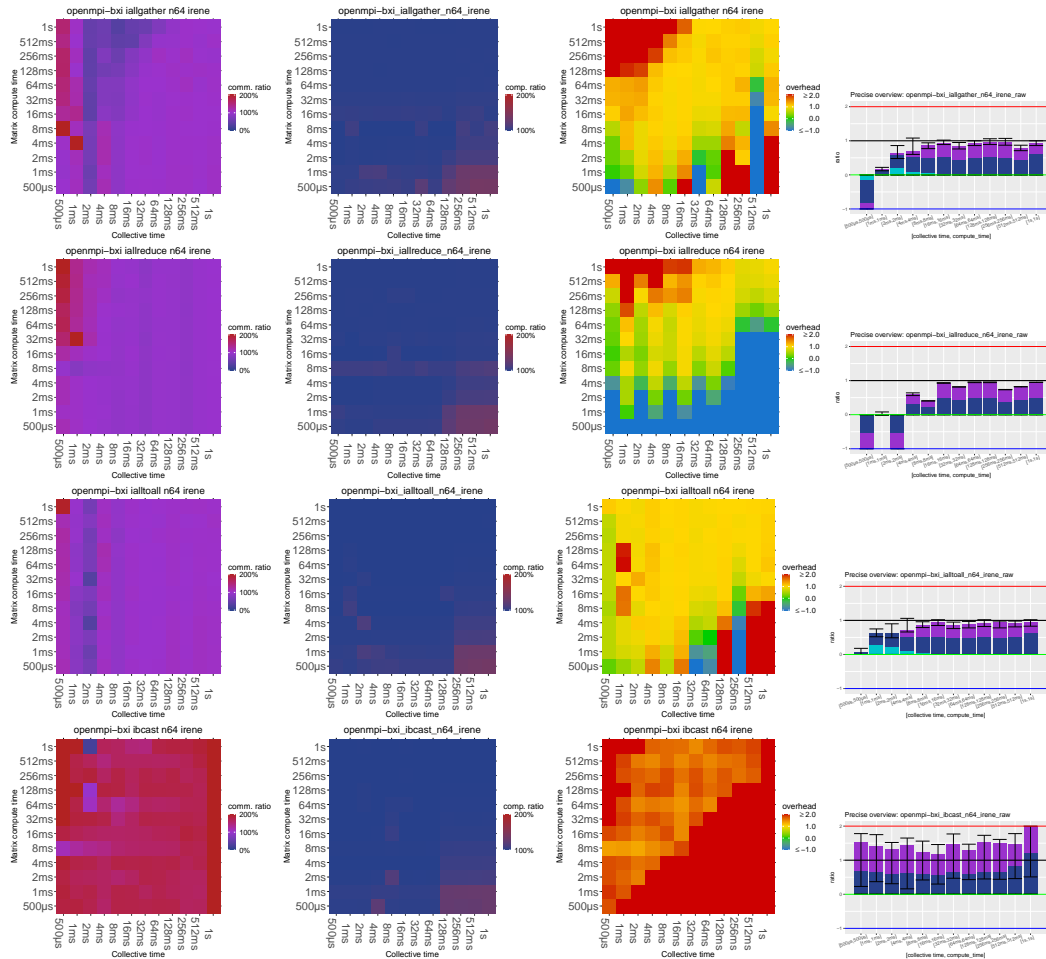






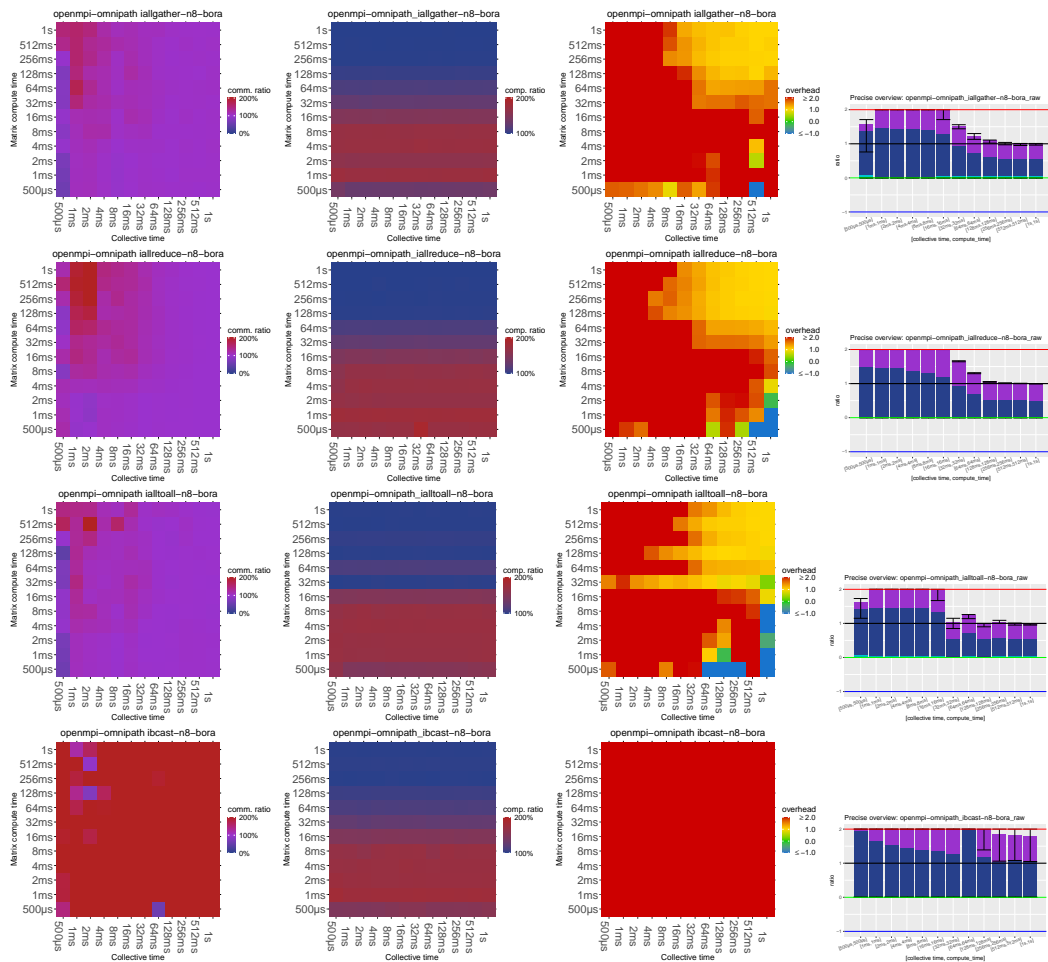
## 12 Open MPI with BXI adapter

We used the same configuration as previous section. The network adapter is a Bull BXI v1.2.



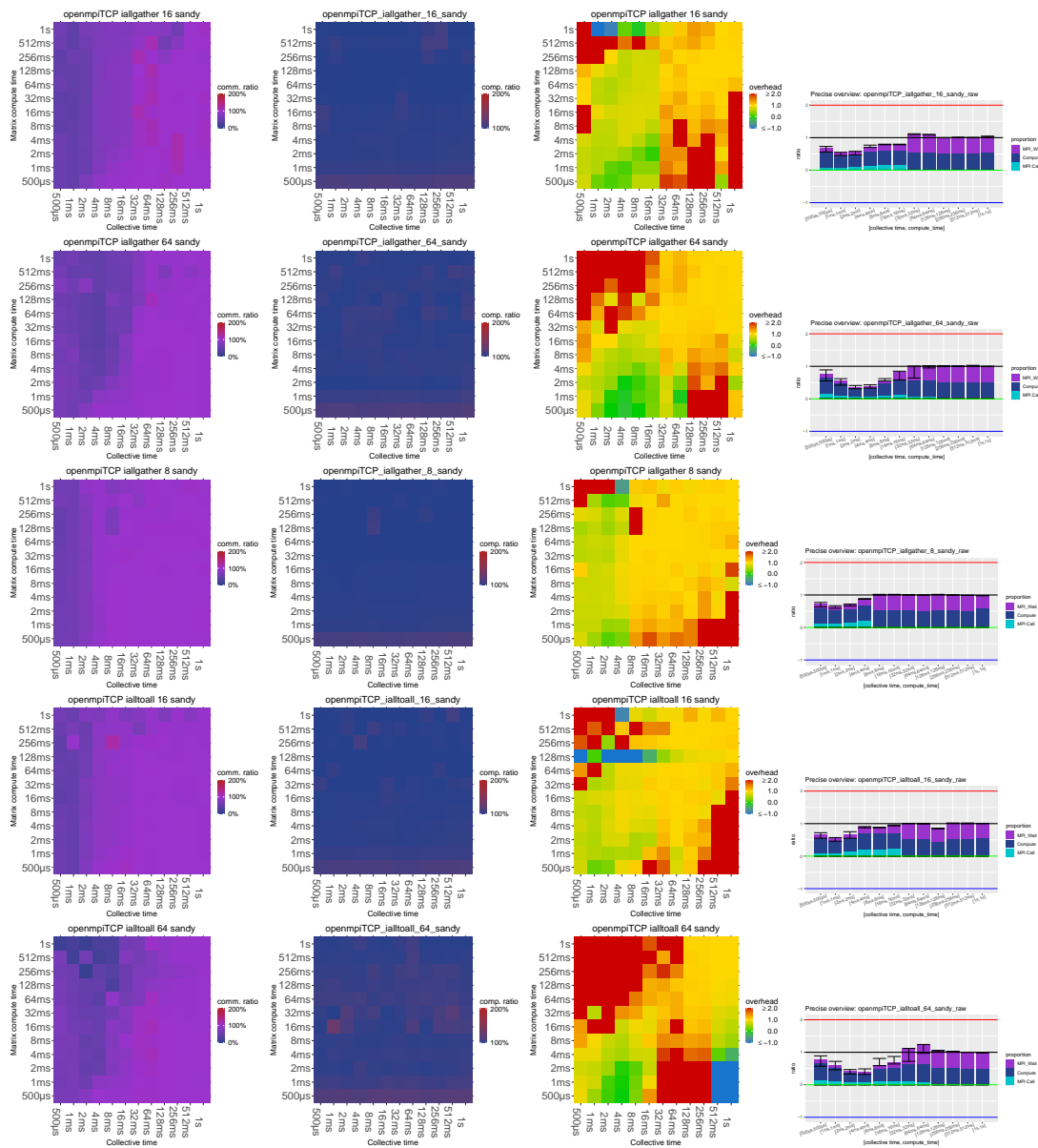
### 13 Open MPI on Omnipath network

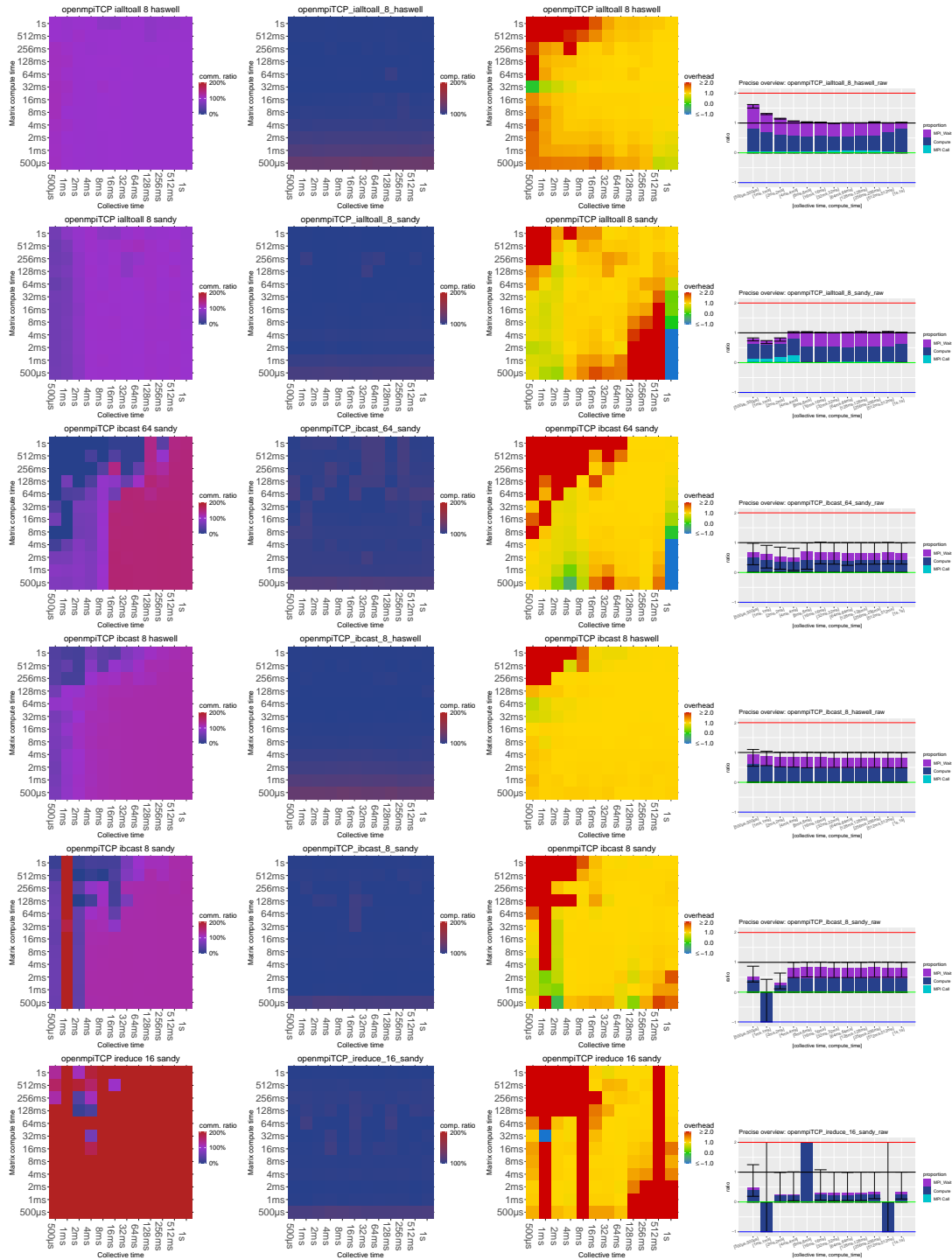
We used the same configuration as both previous sections but on an Omnipath network.

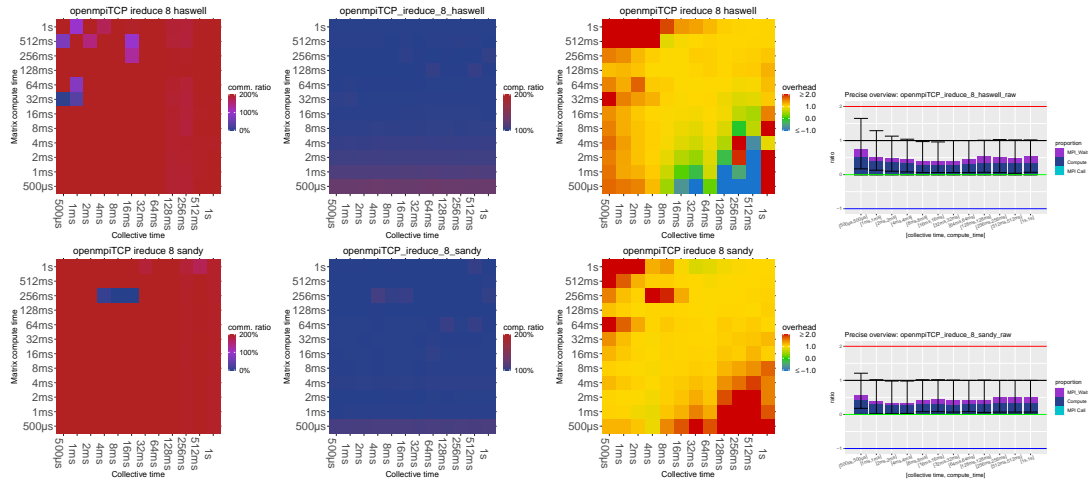


# 14 Open MPI on TCP network

We used the same configuration as three previous sections but on an TCP network.









**RESEARCH CENTRE  
BORDEAUX – SUD-OUEST**

200 avenue de la Vieille Tour  
33405 Talence Cedex

Publisher  
Inria  
Domaine de Voluceau - Rocquencourt  
BP 105 - 78153 Le Chesnay Cedex  
[inria.fr](http://inria.fr)

ISSN 0249-6399