Design, Implementation, and Validation of a Heterogeneous Resource for HPC

G.B. Barone[1], D. Bottalico[1], **L. Carracciuolo**[2], D. Michelino[1], G. Sabella[1]

[1]University of Naples Federico II, Italy
[2]National Research Council, Italy

Final Workshop for the Italian PON IBiSCo Project
April 18-19, 2024

Table of Contents

Table of Contents

Introduction I

COTS Computing System

In the first half of the 1990s, Thomas Sterling and Donald Becker built a cluster of networked computers, called **Beowulf** [21], as an alternative to large supercomputers. At the time, their idea of providing "**Commodity Off The Shelf (COTS)**" based systems has been a great success.

Resources for Exascale Computing

This idea is still valid and can inspire the realization of HPC computing systems, whose computational power is far from that of the most powerful computers in the world, but whose architecture is already **compliant with incoming Exascale Era systems**. Most likely, these systems will respond to the following description:

- multi-node systems, connected by high-performance networks,
- where each node will have a high level of internal parallelism which will be also made available by the *accelerators* based on technologies such as `NVIDIA` and `Intel Xe` GP-GPUs (the **accelerators**).

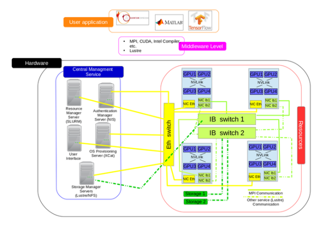
Introduction II

Description of the IBiSCo HPC resources and their Management Services

The use of heterogeneous features aims to **ensure the best use** of resources **for different scenarios applications**, such as distributed memory computing, GP-GPU accelerated workloads and their combinations.

- 128 GPUs and about 1600 physical cores distributed on 32 nodes whose connections are based on InfiniBand and NVLink technologies.
- 320 TB distributed on 4 storage nodes connected to the computing nodes by an InfiniBand network.

Some **Management Services** are configured: `NIS` for **authentication**, `Slurm` for **management/access to computing resources**, `User Interface` for **cluster front-end**, `Lustre` and `NFS` for **storage management**.



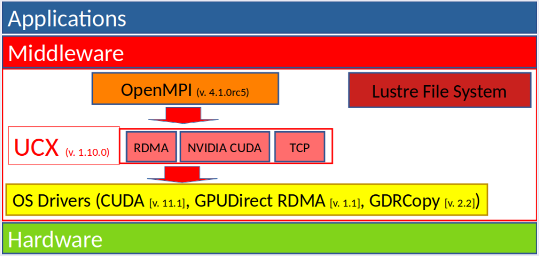
Cluster Management Services

Table of Contents

The Architecture of the Heterogeneous High Performance Computing Cluster I

Layered architecture

The architecture of this cluster is depicted as a set of multiple layers. The highest layer of the architecture consists of the application layer. The lowest one consists of the hardware resources.



The Layered Cluster Architecture

The Architecture of the Heterogeneous High Performance Computing Cluster II

The middleware components of IBiSCo HPC cluster

The efficient use of cluster technologies is made possible by a software layer (**based on "Open Source" solutions**) interposed between the lowest and the highest levels, namely **the middleware**, which is based on a combination of the following technologies:

1. **OpenFabrics Enterprise Distribution (OFED)** [14] for drivers and libraries needed by the Mellanox InfiniBand network cards .
2. **CUDA Toolkit** [13] for drivers, libraries, and, development environments, enables NVIDA GP-GPU .
3. **MPI-CUDA aware** [2] implementation of OpenMPI [15] through the UCX open-source framework [16].
4. **Lustre file system** [19] a distributed, parallel, and open source file system - provides high-performance access to storage resources.

The Architecture of the Heterogeneous High Performance Computing Cluster III

The process communication sub-layer

Bandwidth and latency in message exchange among processes are some of the issues preventing the full exploitation of GP-GPU potential. In this regard, NVIDIA introduced

- `CUDA Inter-Process Copy (IPC)` [10] and `GPUDirect Remote Direct Memory Access (RDMA)` [7] technologies for intra- and inter-node GPU process communications for InfiniBand-based clusters.
- `gdrcopy` to optimize inter-node GPU-to-GPU communications for small messages. [17].

To combine these technologies with communication libraries (i.e., OpenMPI), the *Unified Communication X (UCX)* open-source framework is used.

UCX is a

- *"... a lightweight exascale-ready communications framework ... "* optimized for modern, high-bandwidth, low-latency networks.
- It **exposes a set of abstract communication primitives** that **automatically choose the best available hardware resources**. Supported technologies include `RDMA` (both `InfiniBand` and `RoCE`), NVIDIA GP-GPU `NVLink`, and shared memory.

The Architecture of the Heterogeneous High Performance Computing Cluster IV

The distributed, parallel file system sub-layer

The implementation adopted in the IBiSCo cluster is based on **Lustre**, a **high-performance, parallel, and distributed file system**. High performance is guaranteed by Lustre's flexibility in supporting multiple storage technologies, from the common ones based on Ethernet and TCP/IP to those with high-speed and low latency such as InfiniBand and RoCE. **Storage nodes host the OSTs** (The Lustre Object Storage Targets (OST) are the block devices on which data is distributed) for the **two Lustre exposed file systems**:

- The **home file system** is characterized by **large disk space needs and fault tolerance**, therefore it is made up of **RAID-5 SAS HDD array**.
- The **scratch area** needs **fast disk access** times and no redundancy requirement, hence it is hosted on **SATA SSD disks**.

Table of Contents

Cluster Validation I

Benchmarks for Cluster Performance Validation

In the HPC context, it is a common practice to **evaluate performance** (in terms of speedup, throughput, I/O speed, etc.) as a **response to the HPC workload** [9] by mean of **appropriate benchmarks**. The most common benchmarks in HPC context (i.e., see [4, 8, 18]) use one of three possible strategies:

- **high-level** benchmarks evaluate performance by testing the application-level components;
- **low-level** benchmarks test low-level system functions (i.e., network bandwidth and latency).
- **hybrid** benchmarks evaluate performance from both low and high levels.

The strategy we use is a *"hybrid"* approach: the tests evaluate the performance of the highest level components (**"macro benchmark tests"**), which can be considered tests from *"the applications point of view"*; down to the evaluation of the lowest level components (**"micro benchmark test"**).

Cluster Validation II

Benchmarks gools

A set of micro- and macro-benchmarks are used to study communication and access to resources. Benchmark results are provided which should be useful for:

1. filling the lack of deep **understanding** on how modern GP-GPU can be connected and **the actual impact** of *"state-of-the-art"* hardware/software **technologies** on **multi-GPU application performance**;
2. **evaluating** the **usage of parallel file systems** in applications with **intensive parallel data access**.

Communication and computation - Micro-benchmark I

Communication and computation - Micro-benchmark description

We evaluate, by the **CUDA-aware version of MPI OSU Micro-Benchmarks** [3], the basic characteristics of the **GPU interconnections** focusing on both **MPI Peer-to-Peer (P2P)** and **MPI Collective (CL) GPU-TO-GPU communication patterns**.

- All the tests are conducted to evaluate the performance of **intra-** and **inter-node communications** where different combinations of `RDMA`, `IPC`, and `gdrcopy` are used.
- All plots use a logarithmic scale with base 2 and 10 respectively for the x and y coordinate axis.

Communication and computation - Micro-benchmark II

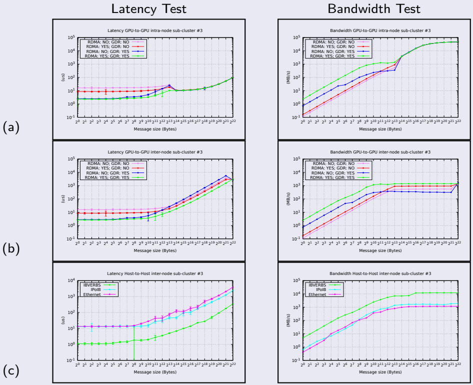
Communication and computation - P2P Micro-benchmark description

The latency and bandwidth of P2P tests are evaluated as follows:

- **Latency Test**: the latency tests are performed in a ping-pong fashion, by using blocking versions of the MPI functions (`MPI_Send` and `MPI_Recv`).
- **Bandwidth Test**: Non-blocking versions of the MPI functions (`MPI_Isend` and `MPI_Irecv`) are used in this case. The sender sends a fixed number of consecutive messages to the recipient and waits for its reply.

Communication and computation - Micro-benchmark III

Communication and computation - P2P Micro-benchmark results



Communication and computation micro-benchmarks results. Latency and bandwidth of P2P GPU-TO-GPU intra-node (a) and inter-node (b), and of Host-to-Host (c) communications

Communication and computation - Micro-benchmark IV

Communication and computation - Comments to P2P Micro-benchmark results

- Appreciable differences can be found between the performance of intra- and inter-node P2P communications. The **intra-node communication** seems to **reach the maximum bandwidth performance** of $50GB/s$, guaranteed by the NVLink technology, already with **medium-sized messages**. The same behavior cannot be witnessed during **inter-node communication** since **the performance** (about $10GB/s$) is **comparable to the peak performance of the InfiniBand technology** only by **transmitting large-sized messages**.
- The use of `gdrcopy` technology (see blue and green lines of all the plots) **significantly improves the performance of P2P communications with small messages**. A combination of `gdrcopy` and `GPUDirect RDMA` technologies seems to be the best choice to improve performance in all the tested configurations: it is more noticeable in P2P inter-node communications.
- All the configurations show equivalent performance when P2P intra-node communication uses large messages.
- The sustainable **performance** values for bf GPU-TO-GPU inter-node communications seem to be, in most cases, **about a tenth** of the value measured for **Host-to-Host communications**, which reach the **InfiniBand peak performance**.

Communication and computation - Micro-benchmark V

Communication and computation - CL Micro-benchmark description

The latency of collective communications is measured via the following procedure:

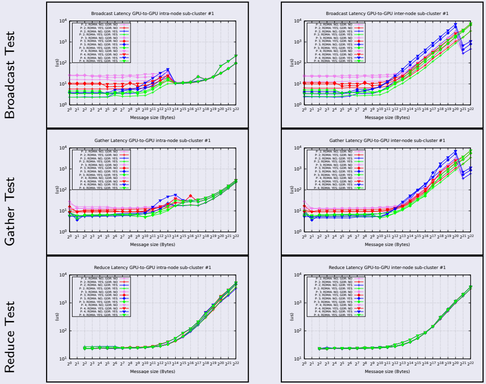
- fixing a message size, many calls of `MPI_BCast`, `MPI_Gather`, `MPI_Reduce` (with `MPI_SUM` operation type) functions are carried out to compute time spent in a single call. All those time values are averaged to compute the latency number of the **Broadcast**, **Gather**, and **Reduce** tests respectively for each considered message size.

In the case of intra-node collective communications,

- all the tasks are spawned on a single node. Conversely, when inter-node collective communication is considered one task is spawned on a single node.
- Tests are performed with different task numbers P. Lines in the plots representing tests executed on $P = 2, 3, 4$ are marked respectively with ■, ◆ and ▼ symbols.

Communication and computation - Micro-benchmark VI

Communication and computation - CL Micro-benchmark results



(a) (b)

Communication and computation micro-benchmarks results. Latency of GPU-to-GPU collective communications on the cluster: intra-node (a) and inter-node (b) communications

Communication and computation - Micro-benchmark VII

Communication and computation - Comments to CL Micro-benchmark results

- **No** particularly **perceptible changes** can be observed in the **Collective Reduce** test if different combinations of `RDMA`, `IPC`, and `gdrcopy` are used. These differences seem more noticeable in inter-node communications
- In the **other Collective Tests** some differences in results can only be **found for small message sizes** when different combinations of `RDMA`, `IPC`, and `gdrcopy` are used.

Communication and computation - Macro-benchmark I

Communication and computation - Macro-benchmark description

- To evaluate how the implemented multi-GPU heterogeneous computational resource responds to **a typical parallel workload from Scientific Computing**, the *CUDA-Aware* version of the **High Performance Linpack (HPL) Benchmark** is used.
- The HPL benchmark [1] is a software package that **solves a (random) dense linear system** in double precision arithmetic on **distributed-memory architectures** and is currently **used to compile the Top500 list** of the most powerful computers in the world [20].
- The **CUDA-Aware HPL benchmark** [6] uses **CUDA libraries to accelerate** the HPL benchmark on heterogeneous clusters, where **both CPUs and GPUs are used** with minor or no modifications to the source code of HPL. A host library intercepts the calls to BLAS *DGEMM* and *DTRSM* procedures and executes them simultaneously on both GPUs and CPU cores. However, the benchmark has a **limit: all communications to and from GPU devices are performed using the PCI channel**.

Communication and computation - Macro-benchmark II

Communication and computation - Macro-benchmark description

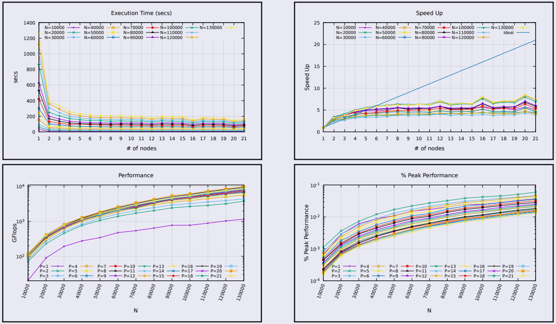
The CUDA-Aware HPL benchmark is executed on some nodes of the IBiSCo cluster: the number of total MPI tasks is $4P$ where P is the number of involved nodes. The tests are performed using different values for the problem dimension N. The graphs show:

- $T(P, N)$: The execution time of the benchmark as a function of the number P of nodes for some values of N;
- $S(P, N)$: The Speed-Up of the execution as a function of the number P of nodes for some values of N. So, $S(P, N) = \frac{T(1,N)}{T(P,N)}$;
- $SP(P, N)$: The Sustained Performance (expressed in GigaFLOPS) is obtained during the execution as a function of the problem dimension N for some values of P. It represents the number of Floating Point operations executable by an algorithm in a time range;
- $SPF(P, N)$: The fraction of Peak Performance is obtained during the execution as a function of the problem dimension N for some values of P. So, $SPF(P, N) = \frac{SP(P,N)}{PP(P)}$ where $PP(P)$ is the Peak Performance of P nodes when for each node all four GPU devices are considered, i.e., $PP(P) = (4NCores_{GPU}Clock_{GPU} + NCores_{CPU}Clock_{CPU})P$.

Communication and computation - Macro-benchmark III

Communication and computation - Macro-benchmark results

(a) (b)



(c) (d)

Communication and computation macro-benchmarks results: the CUDA-Aware HPL benchmark Execution Time $T(P)$ (a), Speed-Up $S(P)$ (b), the Sustained Performance $SP(P,N)$ (c) and the fraction of Peak Performance $SPF(P,N)$ (d).

Communication and computation - Macro-benchmark IV

Communication and computation - Comments to Macro-benchmark results

- the super linear speedup which is most remarkable for **large problems**. We think this is due to the increased time spent on CPU-GPU communications mainly as a consequence of a **saturated PCI channel** (indeed all the four GPUs of a node are involved in computations);
- the very **low scalability** of the benchmark as the number of parallel tasks increases;
- the very **small fraction of the Peak Performance** scored during executions: if we consider very large problems we get just under 10% of max computational power which can be guaranteed by the computational resources.

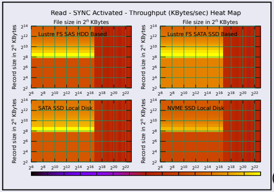
Communication and data storage - Micro-benchmark I

Communication and data storage - Micro-benchmark tests description

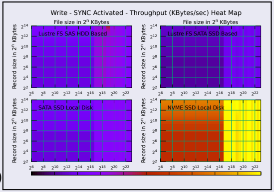
- We evaluate the basic characteristics of the implemented Lustre file systems using the **IOzone File system Benchmark** [11], which generates and measures the **time to complete** a set of file **operations as read, write**, re-read, re-write, reporting, in the plots, the **throughput performance** for the same above-mentioned operations both with and without the `SYNC` IOZone option. When this option is activated, `IOZone` will open the files with the `O_SYNC` flag **forcing all writes to the file to go completely** to disk before returning to the benchmark.
- The plots show single stream performance as a *"Heat Map"* of file size and request size for **two Lustre-based file systems** which are an **aggregation of SAS HDDs and SATA SSDs respectively** both available on storage nodes.
- In the same plots, we show, as a **term of comparison**, the results of the same test performed using two XFS file systems configured on different types of **local disks (SATA SSD and PCIe NVMe SSD)** available on computing nodes. All plots use a logarithmic scale with base 2 for the x and y coordinate axes.

Communication and data storage - Micro-benchmark II

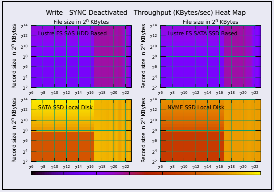
Communication and data storage - Micro-benchmark tests results



(a)



(b)



(c)

Communication and storage micro-benchmarks results: `IOZone` throughput performance (in KB/s) for read (a) and write operations with (b) and without (c) the `SYNC` options.

Communication and data storage - Micro-benchmark III

Communication and data storage - Comments to Micro-benchmark tests

- on **read operations**, all the tested file systems show **comparable performance** and **suffer from large file sizes**;
- the **Lustre** file system seems to be especially **performing on write operations** when **file size increases**. This is more noticeable if the option SYNC is activated;
- on **write operations**, the performance of **Lustre** file systems seems to be **comparable** (in terms of order of **magnitude**) with results obtained on **slow local disks** (especially if the option SYNC is disabled);

Communication and data storage - Macro-benchmark I

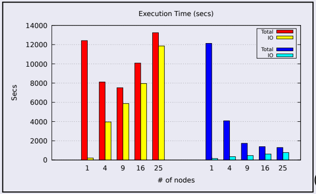
Communication and data storage - Macro-benchmark tests description

- We use a benchmark based on the **Block-Tridiagonal (BT)** problem of the **NAS Parallel Benchmarks (NPB)**[12], which is employed to test the **I/O capabilities** of high-performance computing systems, especially **parallel systems**.

  The benchmark, named `BT-IO`, is based on the **MPI I/O Application Programmer Interface** [5] which is **part of the MPI**.
- We report the results of the `BT-IO` benchmark in its *"simple"* configuration where **data, scattered in memory across the processors, are written to the same file**. What is considered here is the class *"E"* problem dimension.
- During execution, one MPI task is allocated to each node, and **both the Lustre file systems** are considered.

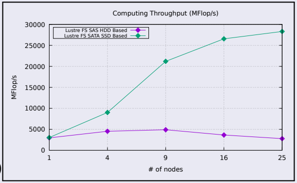
Communication and data storage - Macro-benchmark II

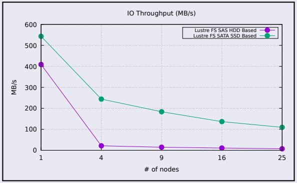
Communication and data storage - Macro-benchmark tests results



(a)

(b)





(c)

Communication and storage macro-benchmarks results. `BT-IO` results: the total time of execution versus the time spent during IO phases (a) [left and right groups of bars are related to HDD and SSD disks respectively]; the throughput of computing (b) and IO (c) stages expressed in MFlops/sec and MB/s respectively

Communication and data storage - Macro-benchmark III

Communication and data storage - Comments to Macro-benchmark tests results

- time spent during the **IO stages** might account for a **meaningful portion** (> 50%) of total execution time when the number of parallel tasks is large;
- the **write pattern** used by the tests, where **each processor writes the data** elements it is responsible for **directly** into an output file, confirms the weak performance due to a very high degree of fragmentation [22]. The Lustre file system based on **SSD disks better manages** such type of **pattern** also when the **number of processors** becomes **large**;
- IO throughput seems far from the values measured by micro-benchmarks which appear to be about a bigger order of magnitude.

Table of Contents

Conclusion I

Conclusion

- We presented the results of some **benchmarking** tests aimed at **verifying and validating** all the **solutions** implemented during the deployment of a computing cluster within the Italian National Project IBiSCo able **to satisfy the different computing needs** of the project partners.
- All the strategies implemented have been verified and evaluated by the appropriate tools used to estimate some **meaningful performance metrics** of all the **system components from a micro and macro point of view**.
- All the **macro-benchmarks confirm** that the goal of **achieving the maximum performance of IT systems is extremely demanding**.
- But we are aware that, **although useful for evaluating cluster performance** and highlighting the strengths of its resources, the **benchmarks are also intended to bring out any issues**. All of this could be spent on improving resource **users' awareness** and **resource managers' ability** to implement **increasingly effective and efficient solutions**.

References I

[1] A. Petitet et al. *A Portable Implementation of the High-Performance Linpack Benchmark for Distributed-Memory Computers*. https://www.netlib.org/benchmark/hpl/index.html (cited on page 21).

[2] J. Kraus. *An Introduction to CUDA-Aware MPI*. https://developer.nvidia.com/blog/introduction-cuda-aware-mpi/ (cited on page 8).

[3] Bureddy, D. et al. "OMB-GPU: A Micro-Benchmark Suite for Evaluating MPI Libraries on GPU Clusters". In: *Recent Advances in the Message Passing Interface*. 2012, pages 110–120. DOI: 10.1007/978-3-642-33518-1_16 (cited on page 14).

[4] *CORAL procurement benchmarks*. https://asc.llnl.gov/sites/asc/files/2020-06/CORALBenchmarksProcedure-v26.pdf (cited on page 12).

[5] Corbett, P. et al. "Overview of the MPI-IO Parallel I/O Interface". In: *Input/Output in Parallel and Distributed Computer Systems*. Boston, MA: Springer US, 1996, pages 127–146. ISBN: 978-1-4613-1401-1. DOI: 10.1007/978-1-4613-1401-1_5 (cited on page 28).

[6] M. Fatica. "Accelerating Linpack with CUDA on Heterogenous Clusters". In: *2nd Workshop on General Purpose Processing on Graphics Processing Units*. GPGPU-2. Washington, D.C., USA: Association for Computing Machinery, 2009, pages 46–51. ISBN: 9781605585178. DOI: 10.1145/1513895.1513901 (cited on page 21).

[7] *GPUDirect RDMA - CUDA Toolkit DOC*. https://docs.nvidia.com/cuda/gpudirect-rdma/index.html (cited on page 9).

[8] *HPC Challenge Benchmark*. https://hpcchallenge.org/hpcc/ (cited on page 12).

References II

[9] Ihde, Nina et al. "A Survey of Big Data, High Performance Computing, and Machine Learning Benchmarks". In: *Performance Evaluation and Benchmarking*. 2022, pages 98–118. DOI: 10.1007/978-3-030-94437-7_7 (cited on page 12).

[10] *Interprocess Communication - Programming Guide :: CUDA Toolkit DOC*. https://docs.nvidia.com/cuda/cuda-c-programming-guide/index.html#interprocess-communication (cited on page 9).

[11] *IOzone Filesystem Benchmark*. https://www.iozone.org/ (cited on page 25).

[12] *NAS Parallel Benchmarks*. https://www.nas.nasa.gov/software/npb.html (cited on page 28).

[13] Nickolls, John et al. "Scalable Parallel Programming with CUDA". In: *Queue* 6.2 (Mar. 2008), pages 40–53. ISSN: 1542-7730. DOI: 10.1145/1365490.1365500 (cited on page 8).

[14] *NVIDIA Mellanox OFED DOC*. https://docs.mellanox.com/display/MLNXOFEDv531001/NVIDIA+MLNX_OFED+Documentation+Rev+5.3-1.0.0.1 (cited on page 8).

[15] *Open MPI: Open Source High Performance Computing*. https://www.open-mpi.org/ (cited on page 8).

[16] Shamis, Pavel et al. "UCX: An Open Source Framework for HPC Network APIs and Beyond". In: *IEEE 23rd Annual Symposium on High-Performance Interconnects*. 2015, pages 40–43. DOI: 10.1109/HOTI.2015.13 (cited on page 8).

References III

[17] Shi, Rong et al. "Designing efficient small message transfer mechanism for inter-node MPI communication on InfiniBand GPU clusters". In: *21st International Conference on High Performance Computing (HiPC)*. 2014, pages 1–10. DOI: 10.1109/HiPC.2014.7116873 (cited on page 9).

[18] *Standard Performance Evaluation Corporation*. https://www.spec.org/ (cited on page 12).

[19] *The Lustre file system*. https://www.lustre.org/ (cited on page 8).

[20] *The Top 500 list Website*. https://www.top500.org/ (cited on page 21).

[21] Thomas Sterling et al. "BEOWULF: A Parallel Workstation For Scientific Computation". In: *24th International Conference on Parallel Processing*. CRC Press, 1995, pages 11–14 (cited on page 4).

[22] Wong, Parkson et al. "NAS Parallel Benchmarks I/O Version 2.4". In: *NAS Technical Report NAS-03-002* (Jan. 2003) (cited on page 30).