

Práctica 9

Introducción a MPI Investigación y Preparación del Entorno

Jordi Blasco Lozano
Computación de alto rendimiento
Grado en Inteligencia Artificial

Índice:

Índice:	2
0. Antes de empezar	2
Plan de investigación	3
1. Investigación Guiada sobre MPI	3
1.1 ¿Qué es MPI?	3
1.2 ¿MPI lenguaje o biblioteca?	4
1.3 ¿Quién desarrolla y mantiene MPI?	4
1.4 Versiones de MPI	5
1.5 Principales usos de MPI	5
2. Preparación de Entornos	8
Instalación y Configuración Detallada en Linux (con Docker)	8
Alternativas para Otros Lenguajes	10
Conclusiones	10
Referencias	11

0. Antes de empezar

Para la primera parte de la práctica quise utilizar la herramienta gratuita de Deep Research que nos proporciona Google con su modelo de inteligencia artificial de razonamiento. Esta herramienta es super útil en entornos de investigación avanzada ya que nos proporciona un plan de investigación detallado con todo lo que el modelo va a investigar, este plan lo podemos modificar a nuestro antojo. Esta práctica será más larga de lo normal ya que explicaré paso a paso como fue la investigación, haré referencia a la cadena de pensamiento y incluiré todas las referencias usadas al final de la práctica.

Plan de investigación

En primer lugar, al usar nuestro prompt con la herramienta Deep Research activada, Gemini nos proporciona un plan de investigación modificable. En mi caso el plan de investigación que se formula es el siguiente:

- (1) Investiga y define qué es MPI (Message Passing Interface), su propósito principal en la computación paralela y distribuida, y su relevancia en entornos de alto rendimiento.
- (2) Determina si MPI es un lenguaje de programación o una biblioteca. Explica su compatibilidad con lenguajes como C, C++, Fortran, Python y Java, detallando cómo se integran las bibliotecas correspondientes en cada caso.
- (3) Investiga quién desarrolla y mantiene MPI, destacando el papel del MPI Forum en su estandarización y evolución. Identifica organismos, empresas y universidades participantes, proporcionando ejemplos de colaboraciones relevantes, como las de centros de supercomputación y multinacionales tecnológicas.
- (4) Describe la evolución de las versiones de MPI (MPI-1, MPI-2, MPI-3, MPI-4), resaltando las mejoras clave introducidas en cada una y el estado actual del estándar en términos de adopción.
- (5) Investiga y documenta al menos tres áreas de aplicación principales donde se utiliza MPI, explicando cómo contribuye a la resolución de problemas complejos en cada una de ellas. Considera ejemplos como supercomputación y simulaciones científicas, inteligencia artificial y machine learning distribuido, análisis de datos a gran escala o modelado de fluidos.
- (6) Identifica al menos tres organizaciones o instituciones relevantes que utilicen MPI en sus sistemas de computación de alto rendimiento, como la NASA, el CERN o el Barcelona Supercomputing Center.
- (7) Para cada organización identificada en el paso anterior, describe brevemente un caso de uso o aplicación concreta de MPI.
- (8) Explica por qué MPI es la tecnología adecuada para los casos de uso identificados, detallando sus ventajas frente a otros modelos o paradigmas de programación paralela. Presenta esta información en formato APA.

No modificaremos el plan de investigación y continuaremos con la primera parte de la práctica.

1. Investigación Guiada sobre MPI

1.1 ¿Qué es MPI?

La Interfaz de Paso de Mensajes (MPI, por sus siglas en inglés) es un estándar para la comunicación entre múltiples computadoras que ejecutan un programa paralelo en un entorno de memoria distribuida. En la computación paralela, los múltiples computadores, o incluso los múltiples núcleos de procesador dentro de un mismo computador, se denominan nodos. El propósito principal de MPI es definir un modelo de computación paralela donde cada proceso paralelo tiene su propia memoria local, y los datos deben compartirse explícitamente mediante el envío de mensajes entre procesos. MPI permite que los programas se escalen más allá de los procesadores y la memoria compartida de un solo servidor, hacia la memoria distribuida y los procesadores de múltiples servidores combinados. Su relevancia en entornos de alto rendimiento radica en su capacidad para facilitar la creación de aplicaciones paralelas portátiles y de alto rendimiento, esenciales para la mayoría de los sistemas de memoria distribuida.

1.2 ¿MPI lenguaje o biblioteca?

MPI no es un lenguaje de programación en sí mismo, sino una especificación para una interfaz de biblioteca estandarizada y portátil capaz de soportar comunicaciones y operaciones relacionadas útiles para la programación en memoria distribuida. Define la sintaxis y la semántica de las rutinas de biblioteca que son útiles para una amplia gama de usuarios que escriben programas portátiles de paso de mensajes en lenguajes como C, C++ y Fortran. La mayoría de las implementaciones de MPI consisten en un conjunto específico de rutinas directamente invocables desde estos lenguajes, constituyendo una interfaz de programación de aplicaciones (API). Si bien el estándar proporciona principalmente enlaces para C, C++ y Fortran, existen enlaces no oficiales y bibliotecas de envoltura que extienden el soporte de MPI a otros lenguajes como Python y Java. Por ejemplo, Python soporta MPI a través de la implementación MPI4Py, que se basa en los enlaces C++ del estándar MPI-2. En el caso de Java, existen bibliotecas como OpenMPI Java y FastMPJ que proporcionan funcionalidades MPI. Estas integraciones a menudo aprovechan las implementaciones subyacentes de MPI en C/C++ para ofrecer capacidades de computación paralela en otros lenguajes.

1.3 ¿Quién desarrolla y mantiene MPI?

El desarrollo y mantenimiento del estándar MPI están a cargo del MPI Forum, un grupo amplio que incluye proveedores de computadoras paralelas, desarrolladores de bibliotecas y especialistas en aplicaciones. El MPI Forum se estableció alrededor de 1993 con el objetivo de crear un estándar para la programación paralela mediante paso de mensajes. Este foro abierto continúa reuniéndose regularmente para considerar correcciones y extensiones al estándar, asegurando su evolución y adaptación a las necesidades de la comunidad de computación de alto rendimiento. La especificación de MPI es impulsada por este foro, que consiste en desarrolladores, proveedores y usuarios de MPI. La colaboración es fundamental, y el proyecto Open MPI, por ejemplo, es desarrollado y mantenido por un consorcio de socios académicos, de investigación e industriales. Entre los contribuyentes a las implementaciones de MPI se encuentran numerosas universidades (como Indiana University y University of Tennessee), laboratorios de investigación (como Oak Ridge National Laboratory y Lawrence Livermore National Laboratory) y empresas tecnológicas (como Nvidia, Oracle, IBM e Intel). Esta amplia participación garantiza que MPI se beneficie de diversas experiencias y se mantenga relevante para el ecosistema de HPC. Las colaboraciones entre centros de supercomputación como el Barcelona Supercomputing Center (BSC) y multinacionales tecnológicas como Lenovo son ejemplos concretos de cómo se impulsa el avance de MPI y las tecnologías relacionadas. Estas asociaciones permiten la investigación conjunta en áreas como la medicina de precisión y la eficiencia energética en supercomputación.

1.4 Versiones de MPI

La evolución del estándar MPI ha pasado por varias versiones desde su lanzamiento inicial. MPI-1.0 se publicó en mayo de 1994, estableciendo el modelo fundamental de paso de mensajes con comunicaciones punto a punto, colectivas y tipos de datos básicos. Posteriormente, se lanzaron versiones menores como MPI-1.1 y MPI-1.2, que incluyeron correcciones y aclaraciones. MPI-2.0, publicado en 1997, introdujo funcionalidades significativas como la entrada/salida paralela, el acceso remoto a memoria (RMA), la gestión dinámica de procesos y los enlaces para C++. MPI-3.0, lanzado en 2012, añadió importantes mejoras como las operaciones colectivas no bloqueantes, las operaciones colectivas de vecindad, mejoras en la comunicación unilateral, una nueva interfaz de herramientas y enlaces para Fortran 2008. La versión más reciente, MPI-4.0, publicada en junio de 2021, se centra en la adaptación del estándar a entornos modernos con gran cantidad de subprocesos y aceleración por GPU, introduciendo características como sesiones, comunicaciones particionadas y mejoras en la resiliencia. La versión actual del estándar es MPI-4.1, lanzada en noviembre de 2023, que contiene principalmente correcciones y aclaraciones a la versión 4.0. MPI sigue siendo el modelo dominante en la computación de alto rendimiento, y la mayoría de las aplicaciones científicas paralelas utilizan MPI, incluyendo los sistemas de exaescala del Departamento de Energía de EE. UU... Implementaciones como MPICH y Open MPI buscan la conformidad total con el estándar MPI-4.1.

1.5 Principales usos de MPI

MPI se utiliza ampliamente en diversas áreas donde la computación de alto rendimiento es esencial para resolver problemas complejos. A continuación, se presentan tres áreas principales de aplicación y organizaciones relevantes que utilizan MPI, junto con un análisis de por qué MPI es la tecnología adecuada en estos casos.

Supercomputación y Simulaciones Científicas

MPI es fundamental en el campo de la supercomputación y las simulaciones científicas. Actúa como el modelo de programación de facto para la computación científica a gran escala y está disponible en todos los sistemas de alto rendimiento. Científicos e ingenieros de todo el mundo emplean MPI en miles de aplicaciones, incluyendo la investigación de fuentes de energía alternativas y la simulación meteorológica. Códigos para astrofísica, dinámica molecular y modelado climático son ejemplos típicos de aplicaciones que utilizan MPI.

Un ejemplo de una organización que utiliza MPI en este ámbito es la NASA. La NASA utiliza MPI en su División de Supercomputación Avanzada (NAS) en el Ames Research Center (ARC) y en el Centro de Simulación Climática de la NASA (NCCS) en el Goddard Space Flight Center (GSFC). Las aplicaciones incluyen simulaciones de hipersónicos y soluciones para la COVID-19. MPI es adecuado para las aplicaciones de la NASA debido a su capacidad para manejar las enormes demandas computacionales de las simulaciones aeroespaciales, el modelado climático y el análisis de grandes conjuntos de datos generados por instrumentos como el lidar. Su escalabilidad y portabilidad permiten a la NASA ejecutar simulaciones complejas en sus sistemas de computación de alto rendimiento.

Inteligencia Artificial y Machine Learning Distribuido

MPI está ganando importancia en el campo de la inteligencia artificial (IA) y el aprendizaje automático (ML) distribuido. Open MPI, una implementación específica de MPI, se utiliza ampliamente en aplicaciones de computación de alto rendimiento, incluyendo IA, ML y ciencia de datos, donde el procesamiento de datos a gran escala y los cálculos complejos son comunes. Open MPI se emplea para distribuir los procesos de entrenamiento a través de múltiples GPU o nodos, lo que reduce significativamente el tiempo necesario para entrenar redes neuronales complejas. Frameworks como TensorFlow y PyTorch pueden aprovechar Open MPI para el entrenamiento distribuido.

El Barcelona Supercomputing Center (BSC) es una institución relevante que utiliza MPI en este contexto. Aunque el material proporcionado no detalla específicamente el uso de MPI en IA/ML en el BSC, su colaboración con empresas como Lenovo para avanzar en la investigación en supercomputación, incluyendo la medicina de precisión, sugiere que MPI podría ser una tecnología subyacente para distribuir las cargas de trabajo computacionales intensivas en el análisis de grandes conjuntos de datos médicos y genómicos, tareas comunes en el ML distribuido. La capacidad de MPI para escalar a través de múltiples nodos y su eficiencia en la comunicación entre procesos lo hacen adecuado para este tipo de aplicaciones.

Análisis de Datos a Gran Escala y Modelado de Fluidos

MPI también juega un papel crucial en el análisis de datos a gran escala y el modelado de fluidos. En ciencia de datos, Open MPI se puede utilizar para paralelizar tareas de procesamiento de datos, como transformaciones a gran escala e ingeniería de características, mejorando la eficiencia y la escalabilidad. mpiBLAST, por ejemplo, mejora el rendimiento de NCBI BLAST en varios órdenes de magnitud al escalar a cientos de procesadores, utilizando eficientemente los recursos computacionales distribuidos a través de la fragmentación de bases de datos, la segmentación de consultas, la programación inteligente y la E/S paralela. En el modelado de fluidos, códigos de simulación como flowCart se ejecutan en paralelo tanto en memoria compartida (OpenMP) como en memoria distribuida (MPI) con excelente escalabilidad. CRUNCH CFD es otro solucionador de flujo de malla no estructurada con capacidades de simulación multifísica que utiliza MPI para la ejecución paralela.

El CERN (Organización Europea para la Investigación Nuclear) es una institución que utiliza MPI en el contexto del análisis de datos a gran escala. El CERN utiliza una infraestructura de lotes Linux para aplicaciones HPC (por ejemplo, simulaciones de ingeniería y física) que requieren clústeres MPI. Los experimentos en el Gran Colisionador de Hadrones (LHC) generan cantidades masivas de datos que requieren un análisis computacional intensivo. MPI permite al CERN distribuir el análisis de estos datos a través de grandes clústeres, lo que les permite extraer información significativa de complejas interacciones de partículas. La necesidad de capacidades avanzadas de computación distribuida en el CERN, evidenciada por su colaboración con empresas de TI para construir una red de datos gigante, subraya la importancia de MPI en su infraestructura.

Ventajas de MPI

La idoneidad de MPI para estos casos de uso se deriva de sus ventajas fundamentales. MPI permite que los programas se escalen más allá de los límites de un solo servidor, utilizando la memoria distribuida y los procesadores de múltiples servidores. Ofrece portabilidad, lo que permite que los programas se ejecuten en diversas configuraciones de hardware y red sin cambios significativos en el código. MPI ha demostrado un rendimiento superior en comparación con otros entornos de programación paralela. Proporciona comunicación asíncrona, gestión eficiente de búferes de mensajes, grupos eficientes y una rica funcionalidad. Incluye un amplio conjunto de operaciones de comunicación colectiva, topologías virtuales y diferentes modos de comunicación. Además, MPI admite bibliotecas de terceros y redes heterogéneas. En el contexto de los sistemas HPC, MPI es el estándar de facto y, a menudo, la única forma disponible para programar clústeres. Ofrece un control granular sobre la gestión de procesos y la comunicación de datos, y fomenta la localidad de la memoria, lo que resulta ventajoso en arquitecturas NUMA. Sus tipos de datos derivados mejoran la portabilidad y la eficiencia al permitir la comunicación de datos no contiguos con una sola llamada.

Conclusiones

MPI se ha consolidado como una tecnología esencial para la computación de alto rendimiento, facilitando la resolución de problemas complejos en una amplia gama de disciplinas. Su capacidad para escalar aplicaciones paralelas en entornos de memoria distribuida, junto con su portabilidad y rico conjunto de características, lo convierten en una herramienta indispensable para la supercomputación, la inteligencia artificial distribuida, el análisis de datos a gran escala y el modelado de fluidos. Organizaciones líderes como la NASA, el CERN y el Barcelona Supercomputing Center confían en MPI para impulsar sus investigaciones y avances tecnológicos, lo que subraya la relevancia y el impacto continuo de MPI en la comunidad científica y de ingeniería.

2. Preparación de Entornos

Sistema Operativo	Entorno MPI en C/C++	Alternativas para otros lenguajes
Windows	MS-MPI o MPICH (instalable con Cygwin)	Python: mpi4py Java: MPJ Express
Linux	Open MPI MPICH o mpi-default-dev	Python: mpi4py Fortran: soporte nativo en MPICH/Open MPI Java: MPJ Express
MacOS	Open MPI o MPICH (usando Homebrew o MacPorts)	Python: mpi4py; Java: MPJ Express

Instalación y Configuración Detallada en Linux (con Docker)

He elegido Linux como el sistema operativo de referencia para el desarrollo detallado por varias razones:

- **Facilidad de configuración:** La existencia de repositorios robustos y documentación extensa.
- **Entorno homogéneo:** Uso de Docker para garantizar que el entorno sea consistente en cualquier máquina (Windows, Linux o MacOS) sin depender de la configuración local.
- **Ligereza y herramientas:** La imagen basada en Debian 11 (Bullseye) es ligera y permite instalar herramientas útiles para administrar, depurar y monitorear aplicaciones MPI.

Recursos Necesarios

- **Docker:** Instalado en el sistema anfitrión.
- **Imagen Base:** python:3.10-bullseye (que utiliza Debian 11).
- **Paquetes del Sistema:** psmisc, bc, y mpich (este último para MPI en C/C++).
- **Librerías y Herramientas para Python:** En caso de usar MPI en Python, se puede instalar mpi4py a través de un archivo de requerimientos (requirements.txt).

Pasos de Instalación y Configuración

Preparación del Dockerfile

Se crea un archivo practica7-dockerfile que define la imagen personalizada con todos los recursos necesarios:

```
practica7 - dockerfile
1 FROM python:3.10-bullseye
2 WORKDIR /workdir
3
4 # Copiamos el archivo de requerimientos
5 COPY requirements.txt .
6
7 # Actualizamos e instalamos las dependencias básicas, MPI (MPICH) y otros paquetes necesarios
8 RUN apt-get update && \
9     apt-get install -y psmisc bc mpich && \
10    pip install --no-cache-dir -r requirements.txt
11
12 ENTRYPOINT ["/bin/bash"]
13
```

Construcción y Ejecución del Contenedor

Se utiliza un script para construir la imagen y montar el directorio de trabajo, de forma que los cambios realizados en el sistema local se sincronicen con el contenedor:

```
practica7 - buildUbuntu
1 #!/bin/bash
2
3 # Construir la imagen de Docker
4 docker build -t miimagen .
5
6 # Iniciar el contenedor y montar la carpeta local
7 docker run -it -v ${PWD}:/workdir miimagen
8
```

Nota: En Windows, se recomienda ejecutar el script utilizando Git Bash

Configuración del Entorno en el Contenedor

Una vez iniciado el contenedor, se tiene acceso a un entorno basado en Debian 11, donde se encuentra instalado MPICH para desarrollar aplicaciones en C o C++. Si se trabaja con Debian localmente, bastaría con ejecutar:

```
practica7 - .bash
1 apt-get install -y psmisc bc mpich
```

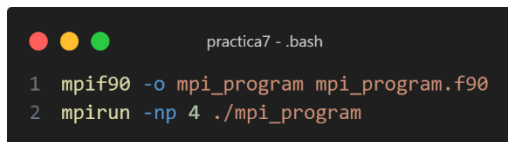
Alternativas para Otros Lenguajes

Python

Se puede instalar mpi4py para implementar aplicaciones MPI en Python.

Fortran

Tanto Open MPI como MPICH proporcionan el compilador mpif90 para Fortran. Se puede compilar un programa MPI de Fortran con:



```
practica7 - .bash
1 mpif90 -o mpi_program mpi_program.f90
2 mpirun -np 4 ./mpi_program
```

Java

MPJ Express es una de las opciones más utilizadas para implementar MPI en Java. Se debe descargar y configurar MPJ Express, compilar y ejecutar programas MPI en Java usando sus scripts de ejecución.

Conclusiones

Elección del Sistema

He optado por desarrollar el entorno en Linux utilizando Docker debido a la facilidad de replicar el entorno de desarrollo en cualquier sistema operativo. Esto permite cumplir con los objetivos de la práctica sin tener que lidiar con las diferencias propias de cada sistema.

Flexibilidad

La imagen Docker basada en Debian 11 simplifica la instalación y configuración de MPI (usando MPICH), mientras que se dejan abiertas alternativas para otros lenguajes como Python (con mpi4py), Fortran y Java.

Gestión y Monitoreo

La inclusión del paquete psmisc permite administrar, depurar y monitorear de forma efectiva los procesos MPI, garantizando una mayor estabilidad y facilidad de uso en el entorno.

Referencias

- ¹ What is Message Passing Interface (MPI)? - TechTarget. (n.d.). Retrieved from [https://www.techtarget.com/searchenterprisedesktop/definition/message-passing-interface-MPI#:~:text=The%20message%20passing%20interface%20\(MPI\)%20is%20a%20standardized%20means%20of,same%20computer%20%E2%80%93%20are%20called%20nodes.](https://www.techtarget.com/searchenterprisedesktop/definition/message-passing-interface-MPI#:~:text=The%20message%20passing%20interface%20(MPI)%20is%20a%20standardized%20means%20of,same%20computer%20%E2%80%93%20are%20called%20nodes.)
- ² Introduction to MPI - Research Computing Services. (n.d.). Retrieved from <https://carleton.ca/rcs/rcdc/introduction-to-mpi/>
- ³ What is MPI? - Cornell Virtual Workshop. (n.d.). Retrieved from <https://cvw.cac.cornell.edu/mpi/intro/what-is-mpi>
- ⁴ Message Passing Interface. (2024, December 2). In **Wikipedia**. Retrieved from https://en.wikipedia.org/wiki/Message_Passing_Interface
- ⁵ What Is MPI? - CIQ. (n.d.). Retrieved from <https://ciq.com/wiki/mpi-message-passing-interface/>
- ⁶ Introduction to MPI - Research Computing Services. (n.d.). Retrieved from <https://carleton.ca/rcs/rcdc/introduction-to-mpi/#:~:text=The%20Message%20Passing%20Interface%20>
- ⁷ What is MPI and why do we use it? I'm a starting fortran programmer. : fortran. (n.d.). Retrieved from https://www.reddit.com/r/fortran/comments/1158z1s/what_is_mpi_and_why_do_we_use_it_im_a_starting/
- ⁸ What's the purpose of MPI? : HPC. (n.d.). Retrieved from https://www.reddit.com/r/HPC/comments/10eat9s/whats_the_purpose_of_mpi/
- ⁹ Introduction to MPI. (n.d.). Retrieved from <https://hpc.nmsu.edu/discovery/mpi/introduction/>
- ¹⁰ Kimpe, D., & Ross, R. (2015). MPI as a Network Transport for Distributed Services.
- ¹¹ MPI - HPC Wiki. (n.d.). Retrieved from <https://hpc-wiki.info/hpc/MPI>
- ¹² Open MPI: Open Source High Performance Computing. (n.d.). Retrieved from <https://www.open-mpi.org/>
- ¹³ MPI - Message Passing Interface - LRZ. (n.d.). Retrieved from <https://doku.lrz.de/mpi-message-passing-interface-11481695.html>
- ¹⁴ Álvarez, C., Pena, T. F., & Tourino, J. (2024). MPI4All: Towards Easy MPI Bindings Generation for Any Programming Language.
- ¹⁵ Armstrong, R., скулъ, D. B., & Viswanathan, V. (n.d.). **Language Interoperability in Scientific Computing: The Babel Approach**. Retrieved from <https://www.osti.gov/servlets/purl/7805>
- ¹⁶ Is C++ the only language that works with MPI? - Stack Overflow. (n.d.). Retrieved from <https://stackoverflow.com/questions/34301431/is-c-the-only-language-that-works-with-mpi>
- ¹⁷ Introduction — mpi4py 3.1.5 documentation. (n.d.). Retrieved from <https://mpi4py.readthedocs.io/en/stable/intro.html>
- ¹⁸ MPI Forum. (n.d.). Retrieved from <https://www.mpi-forum.org/>
- ¹⁹ MPI | Meeting Professionals International. (n.d.). Retrieved from <https://www.mpi.org/>
- ²⁰ ULFM Fault Tolerance — Open MPI v5.0.x documentation. (n.d.). Retrieved from <https://docs.open-mpi.org/en/main/features/ulfm.html>
- ²¹ Gropp, W., Lusk, E., & Skjellum, A. (n.d.). **Using MPI: Portable Parallel Programming with the Message-Passing Interface**.
- ²² Initialisation of MPI is too restrictive · Issue #103 · mpi-forum/mpi-issues. (n.d.). Retrieved from <https://github.com/mpi-forum/mpi-issues/issues/103>
- ²³ Master Planning for Innovation (MPI) | APQC. (n.d.). Retrieved from <https://www.apqc.org/what-we-do/education-k-12-services/master-planning-innovation>
- ²⁴ Funders | Migration Policy Institute. (n.d.). Retrieved from <https://www.migrationpolicy.org/about/funders>
- ²⁵ The Open MPI Development Team — Open MPI v5.0.x documentation. (n.d.). Retrieved from <https://www.open-mpi.org/about/members/>
- ²⁶ Who We Are | MPI | Meeting Professionals International. (n.d.). Retrieved from <https://www.mpi.org/about/who-we-are>
- ²⁷ Homepage | OPHI. (n.d.). Retrieved from <https://ophi.org.uk/>
- ²⁸ MPCDF: Max Planck Computing and Data Facility. (n.d.). Retrieved from <https://www.mpcdf.mpg.de/>
- ²⁹ LLSC 5th Anniversary Booklet_010322_0.pdf. (n.d.). Retrieved from https://www.ll.mit.edu/sites/default/files/rdgroup/doc/2022-01/LLSC%205th%20Anniversary%20Booklet_010322_0.pdf
- ³⁰ HLRS - High-Performance Computing Center Stuttgart. (n.d.). Retrieved from <https://www.hlrs.de/>
- ³¹ Bridges-2 - Pittsburgh Supercomputing Center. (n.d.). Retrieved from <https://www.psc.edu/resources/bridges-2/>
- ³² The Barcelona Supercomputing Center and Lenovo announce partnership to significantly advance R&D in supercomputing technology | BSC-CNS. (2023, November 21). Retrieved from <https://www.bsc.es/news/bsc-news/the-barcelona-supercomputing-center-and-lenovo-announce-partnership-significantly-advance-rd>
- ³³ MPI-4.1 Standard. (n.d.). Retrieved from <https://www.mpi-forum.org/docs/mpi-4.1/mpi41-report/mpi41-report.htm>
- ³⁴ MPI-4.0 Standard. (n.d.). Retrieved from <https://www.mpi-forum.org/docs/mpi-4.0/mpi40-report.pdf>
- ³⁵ MPI Advanced. (n.d.). Retrieved from http://www.unixer.de/teaching/mpi_tutorials/ppopp13/2013-02-24-ppopp-mpi-advanced.pdf
- ³⁶ L02-MPI-Evolution.pdf. (n.d.). Retrieved from <http://www.archer.ac.uk/training/course-material/2018/04/adv-mpi-exeter/Slides/L02-MPI-Evolution.pdf>
- ³⁷ MPI Updates - Exascale Computing Project. (n.d.). Retrieved from <https://www.exascaleproject.org/highlight/mpi-updates/>

- ³⁸ 2024-11-20-MPICH-BoF.pdf. (n.d.). Retrieved from <https://www.mpich.org/static/docs/slides/2024-sc-bof/2024-11-20-MPICH-BoF.pdf> ⁴ Message Passing Interface. (2024, December 2). In [Wikipedia](https://en.wikipedia.org/wiki/Message_Passing_Interface). Retrieved from https://en.wikipedia.org/wiki/Message_Passing_Interface
- ³⁹ What's New in the MPI Standard? - Argonne Leadership Computing Facility. (n.d.). Retrieved from <https://www.alcf.anl.gov/events/what-s-new-mpi-standard>
- ⁴⁰ Implementation Status - MPI Forum. (n.d.). Retrieved from <https://www.mpi-forum.org/implementation-status/>
- ⁴¹ SC24. (n.d.). Retrieved from <https://sc24.conference-program.com/presentation/?id=drs105&sess=sess810>
- ⁴² Exascale MPI (MPICH) - Exascale Computing Project. (n.d.). Retrieved from <https://www.exascaleproject.org/research-project/exascale-mpi-mpich/>
- ⁴³ Formal Analysis of MPI-based Parallel Programs. (n.d.). Retrieved from <https://www.computer.org/csdl/proceedings-article/e-science/2024/10678714/20op0AAXi5q>
- ⁴⁴ What are some applications that use MPI? : compsci. (n.d.). Retrieved from https://www.reddit.com/r/compsci/comments/2tvono/what_are_some_applications_that_use_mpi/
- ⁴⁵ A Scalable Approach to MPI Application Performance Analysis Using Automated Instrumentation and Database Management. (n.d.). Retrieved from <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=2a5eddbelaa94f90339e523e4568ebf0563efd>
- ⁴⁶ MPI Applications - ASPsys.com. (n.d.). Retrieved from <https://www.aspsys.com/mpi-applications/> ⁴⁷ Open MPI Explained - AI Jobs. (2023, September 13). Retrieved from <https://aijobs.net/insights/open-mpi-explained/>
- ⁴⁸ Machine learning and deep learning models for applications with magnetic particle imaging: A review - PMC. (n.d.). Retrieved from <https://pmc.ncbi.nlm.nih.gov/articles/PMC11324856/>
- ⁴⁹ What is Deep Learning? (n.d.). Retrieved from <https://www.youtube.com/watch?v=FrJM2dH57jo>
- ⁵⁰ Distributed Computing with MPI - Codefinity. (n.d.). Retrieved from <https://codefinity.com/blog/Distributed-Computing-with-MPI>
- ⁵¹ MPI Support in ADCME · ADCME.jl. (n.d.). Retrieved from <https://kailaix.github.io/ADCME.jl/latest/mpi/>
- ⁵² Anatomy of machine learning algorithm in mpi, spark and flink. (n.d.). Retrieved from https://infomall.org/publications/Anatomy_of_machine_learning_algorithm_in_mpi_spark_and_flink.pdf
- ⁵³ Data-flow analysis for MPI programs. (n.d.). Retrieved from https://www.researchgate.net/publication/221085145_Data-flow_analysis_for_MPI_programs
- ⁵⁴ Formal Analysis of MPI-based Parallel Programs. (n.d.). Retrieved from <https://www.anl.gov/mcs/article/formal-analysis-of-mpibased-parallel-programs>
- ⁵⁵ A Scalable Approach to MPI Application Performance Analysis. (n.d.). Retrieved from https://www.netlib.org/utk/people/JackDongarra/PAPERS/2005_A%20Scalable-Approach-to-MPI-Application-Performance-Analysis.pdf
- ⁵⁶ Depth Analysis of MPI Programs. (n.d.). Retrieved from <https://www.cs.rochester.edu/u/cding/amp/papers/full/Depth%20Analysis%20of%20MPI%20Programs.pdf>
- ⁵⁷ Simpson, J., Shi, J.-J., Tao, W.-K., & Pickering, K. E. (2004). A Cumulus Parameterization Based on a Mass and Heat Flux Approach. *Monthly Weather Review*, 132(6), 1131–1158.
- ⁵⁸ Compilers - NCCS Users Portal. (n.d.). Retrieved from <https://www.nccs.nasa.gov/nccs-users/instructional/using-discover/compilation-software/compilers>
- ⁵⁹ ARM Data Archive - ARM Research Campaigns - SGP 2000 AFWEX. (n.d.). Retrieved from <https://arm.gov/research/campaigns/sgp2000afwex>
- ⁶⁰ NACS_RFP_J.1(a)_Attachment_1_SOW_20170411.pdf. (n.d.). Retrieved from [https://imlive.s3.amazonaws.com/Federal+Government/ID212655564112226265576763645925039578388/NACS_RFP_J.1\(a\)_Attachment_1_SOW_20170411.pdf](https://imlive.s3.amazonaws.com/Federal+Government/ID212655564112226265576763645925039578388/NACS_RFP_J.1(a)_Attachment_1_SOW_20170411.pdf)
- ⁶¹ NPB 2.0 Results. (n.d.). Retrieved from <https://www.davidhbailey.com/dhbpapers/npb-2.0.pdf>
- ⁶² CERN teams with leaders in Information Technology to build giant DataGrid. (n.d.). Retrieved from <https://home.cern/news/press-release/cern/cern-teams-leaders-information-technology-build-giant-data-grid>
- ⁶³ Particle Accelerators Around the World. (n.d.). Retrieved from http://www-elsa.physik.uni-bonn.de/accelerator_list.html
- ⁶⁴ MPI für Kernphysik: Startseite. (n.d.). Retrieved from <https://www.mpi-hd.mpg.de/mpi/en/>
- ⁶⁵ ISOLTRAP (CERN) - MPI für Kernphysik. (n.d.). Retrieved from <https://www.mpi-hd.mpg.de/mpi/en/research/scientific-divisions-and-groups/stored-and-cooled-ions/research/isoltrap-cern>
- ⁶⁶ HPC - CERN Service Portal. (n.d.). Retrieved from https://cern.service-now.com/service-portal?id=service_element&name=High-Performance-Computing
- ⁶⁷ Collaboration with Companies | BSC-CNS. (n.d.). Retrieved from <https://www.bsc.es/tech-transfer/collaboration-with-companies>
- ⁶⁸ Organisation | BSC-CNS. (n.d.). Retrieved from <https://www.bsc.es/discover-bsc/organisation>
- ⁶⁹ SC22 - The International Conference for High Performance Computing, Networking, Storage, and Analysis. (n.d.). Retrieved from <https://sc22.supercomputing.org/organization/?inst=6872243682651131793>
- ⁷⁰ Barcelona Supercomputing Center - BSC. (n.d.). Retrieved from <https://www.bsc.es/>
- ⁷¹ Message Passing Interface - MPI | BSC-CNS. (n.d.). Retrieved from <https://www.bsc.es/research-development/research-areas/programming-models/message-passing-interface-mpi>