Generating and Scaling a Multi-Language Test-Suite for MPI

Julien Adam, Jean-Baptiste Besnard, Paul Canat, Hugo Taboada,
Adrien Roussel, Marc Pérache, Julien Jaeger, Sameer Shende

30th European MPI Users' Group Meeting
September 11-13, 2023
Bristol, UK

EUROMPI 23

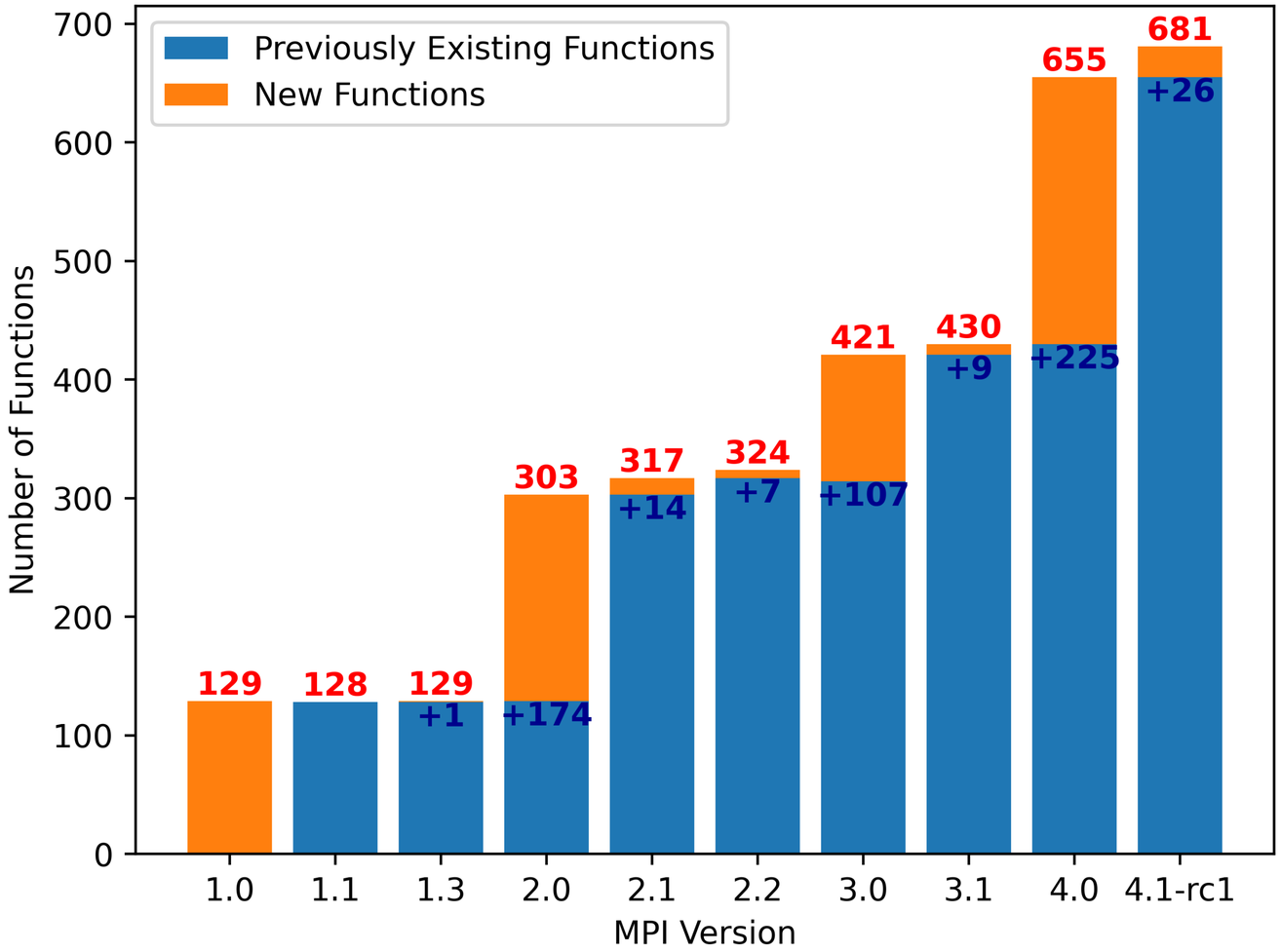
MPI & Testing

- Testing is not new
- Scalability tests are a challenge on their own. Many parameters to consider:
  - Interconnect
  - Number of MPI processes
  - Affinity.
- => **Validation != « `mpirun -np 2 ./IMB-MPI1 MPI_AllReduce` »**
- => Many custom-made test-suites are available and broadly used and often require modifications to fit the testing environment

Changes from the MPI Standard

- MPI Standard is growing… fast
- +30% functions with MPI 4.0
- Mostly induced by Large count
- LaTex bindings became a hassle

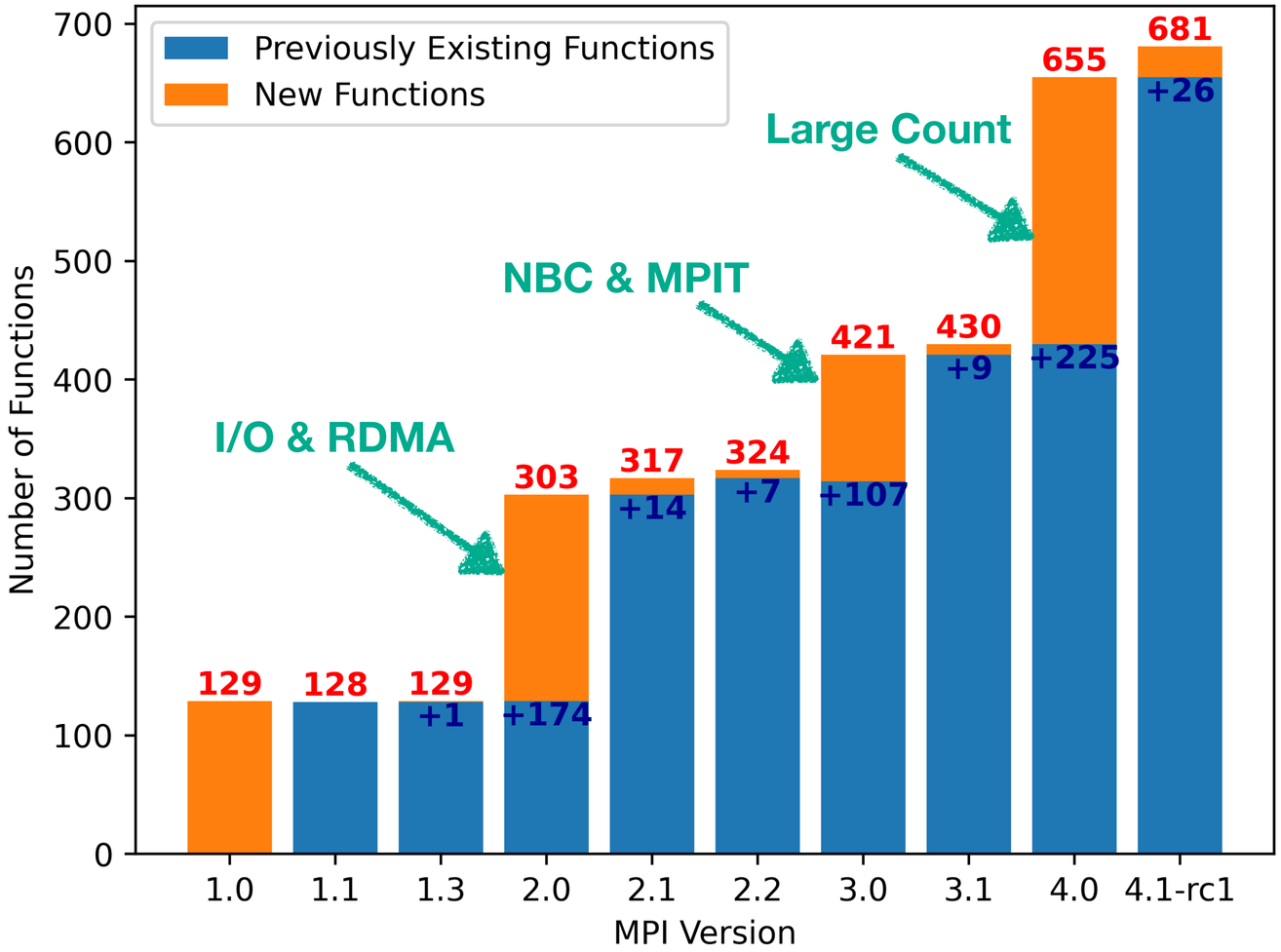
- => « ***Pythonization*** »



Changes from the MPI Standard

- MPI Standard is growing… fast
- +30% functions with MPI 4.0
- Mostly induced by Large count
- LaTex bindings became a hassle

- => « ***Pythonization*** »



Changes from the MPI Standard

```
\begin{funcdef}{MPI\_SEND(buf, count, datatype, dest, tag, comm)}
\funcarg{\IN}{buf}{initial address of send buffer (choice)}
\funcarg{\IN}{count}{number of elements in send buffer (non-negative integer)}
\funcarg{\IN}{datatype}{datatype of each send buffer element (handle)}
\funcarg{\IN}{dest}{rank of destination (integer)}
\funcarg{\IN}{tag}{message tag (integer)}
\funcarg{\IN}{comm}{communicator (handle)}
\end{funcdef}

\cdeclmainindex{MPI\_Comm}%
\mpibind{MPI\_Send(const~void*~buf, int~count, MPI\_Datatype~datatype, int~dest, int~tag, MPI\_Comm~comm)}

\mpifnewbind{MPI\_Send(buf, count, datatype, dest, tag, comm, ierror) \fargs TYPE(*), DIMENSION(..),
INTENT(IN) :: buf \\ INTEGER, INTENT(IN) :: count, dest, tag \\ TYPE(MPI\_Datatype), INTENT(IN) :: datatype \\
TYPE(MPI\_Comm), INTENT(IN) :: comm \\ INTEGER, OPTIONAL, INTENT(OUT) :: ierror}
\mpifbind{MPI\_SEND(BUF, COUNT, DATATYPE, DEST, TAG, COMM, IERROR)\fargs <type> BUF(*) \\ INTEGER  COUNT,
DATATYPE, DEST, TAG, COMM, IERROR}
\mpicppemptybind{MPI::Comm::Send(const void*~buf, int~count, const MPI::Datatype\&~datatype, int~dest,
int~tag) const}{void}
```

```
\begin{mpi-binding}
    function_name('MPI_Send')

    parameter('buf', 'BUFFER', desc='initial address of send buffer', constant=True)
    parameter('count', 'POLYXFER_NUM_ELEM', desc='number of elements in send buffer')
    parameter('datatype', 'DATATYPE', desc='datatype of each send buffer element')
    parameter('dest', 'RANK', desc='rank of destination')
    parameter('tag', 'TAG', desc='message tag')
    parameter('comm', 'COMMUNICATOR')
\end{mpi-binding}
```

Motivation

- Proper testing in (HPC) is a challenge, especially from library/tool perspective
- MPI is dense, moving forward before features are fully supported among runtimes
- Due to increasing complexity, implementations cannot be on par in every aspect of MPI.
- => How to exploit this « pythonized » interface to build a proper MPI validation tool…
  - … Scaling to the test environment?
  - … Reconfigurable to the targeted runtime?
  - … Without any user intervention?

Motivation

Data from the MPI Forum website:
https://www.mpi-forum.org/implementation-status/

MPI 4.0

Feature	MPICH	Open MPI
Large Counts	✅	✅
Partitioned Communication	✅	✅
Sessions	✅	✅
MPI_T Events	✅	✅
Error Handling	✅	✅
Non-blocking SENDRECV	✅	✅
Persistent Collectives	✅	✅
New Split Types	✅	✅
MPI_COMM_DUP info	✅	✅
Info Assertions	✅	✅
Memory Alignment	✅	✅
MPI_INFO_CREATE_ENV	✅	✅

- Under development + - Partly done

MPI 3.1

Feature	MPICH	MVAPICH	Open MPI	Cray	Tianhe	Intel MPI	IBM (BG/Q - Legacy)	IBM (PE - Legacy)	IBM (Spectrum)	HPE	Fujitsu	Microsoft	MPC	NEC	Sunway	RIKEN	AMPI
Non-Blocking Collectives	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
Neighborhood Collectives	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	❌	✅	✅	✅	✅	✅
RMA	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	+
Shared Memory	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	#
MPI_T	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	#	✅	✅	✅	✅	Q1 2019
`MPI_COMM_CREATE_GROUP`	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	#	✅	✅	✅	✅	✅
F08 Bindings	✅	✅	✅	✅	✅	✅	✅	❌	✅	✅	#	❌	✅	✅	✅	✅	Q2 2019
New Datatypes	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
Large Counts	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
`MPI_MProbe`	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅	✅
NBC I/O	✅	✅	✅	✅	❌	✅	❌	❌	✅	✅	#	❌	#	✅	❌	✅	Q2 2019

- Under development + - Partly done

mpicheck

MPI CHECK

- Build test-suites to assess coverage from MPI implementations
- 1 scenario = 1 MPI function call and its derivatives (PMPI, large count…)
- Focused on function semantics, not functional testing (yet?)
- Generated from standard's sources
- Support language bindings C & Fortran(s)
- Classifier to annotate tests
- Compliant with PCVS, and easily portable to any test framework format

Labelling Standard Levels

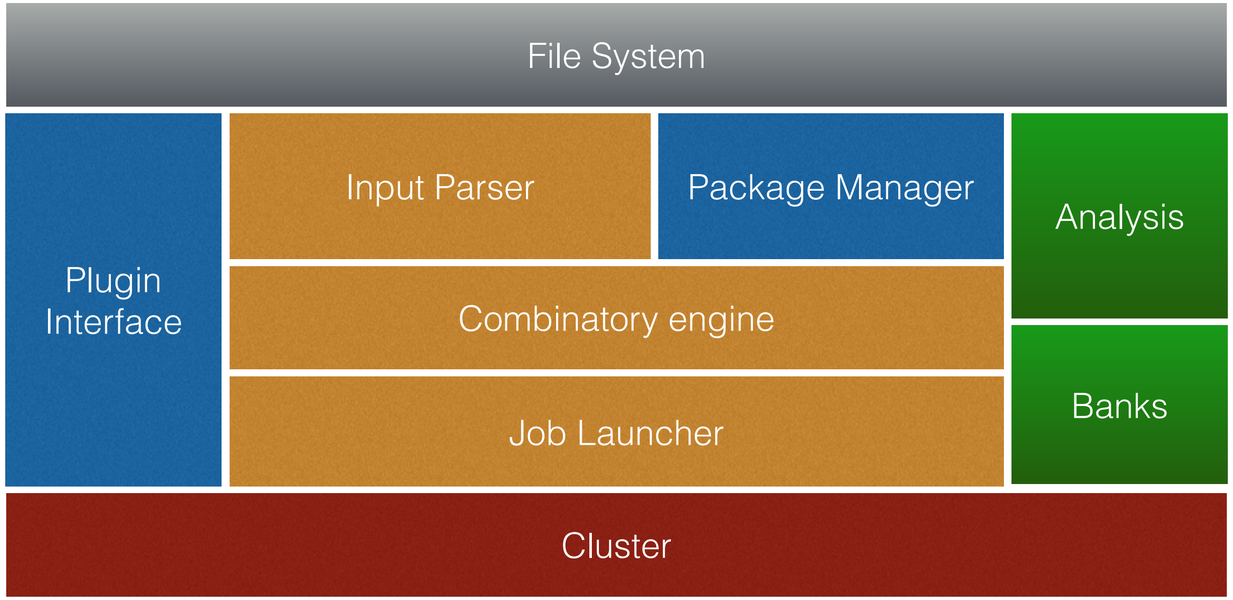
MPI Function Table

MPI Functions	1.0	1.1	1.3	2.0	2.1	2.2	3.0	3.1	4.0	4.1*
MPI_Abort	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
MPI_Accumulate	N/A	N/A	N/A	✔	✔	✔	✔	✔	✔	✔
MPI_Accumulate_c	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	✔	✔
MPI_Add_error_class	N/A	N/A	N/A	✔	✔	✔	✔	✔	✔	✔
MPI_Add_error_code	N/A	N/A	N/A	✔	✔	✔	✔	✔	✔	✔
MPI_Add_error_string	N/A	N/A	N/A	✔	✔	✔	✔	✔	✔	✔
MPI_Address	✔	✔	✔	⚠	⚠	⚠	✖	✖	✖	✖
MPI_Aint_add	N/A	N/A	N/A	N/A	N/A	N/A	N/A	✔	✔	✔
MPI_Aint_diff	N/A	N/A	N/A	N/A	N/A	N/A	N/A	✔	✔	✔
MPI_Allgather	✔	✔	✔	✔	✔	✔	✔	✔	✔	✔
MPI_Allgather_c	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	✔	✔
MPI_Allgather_init	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	✔	✔
MPI_Allgather_init_c	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	✔	✔

Test Engine: PCVS

https://pcvs.io/

- Parallel Computing Validation System
- CLI-based Job Engine, compatible with batch-managers.
- Written in Python, inputs in YAML
- Compatible with many build systems
  (Make, CMake Autotools)
- Decouples benchmarks & tests from testing environments
  - Making test design portable
  - Retargeting apps to different runtimes
  - Deploy test-suites to various test architecture
    (workstations, large clusters)
  - Build execution timeline to keep track of progession/regression



Test Engine: PCVS

- Benchmarks expose resource requirements
- Environment provides resources
- => the intersection constitutes the combinatory matrix
- Test workload depends on both these information
- Distributed over computing resource to minimize time to result
- Over multiple scheduling policies

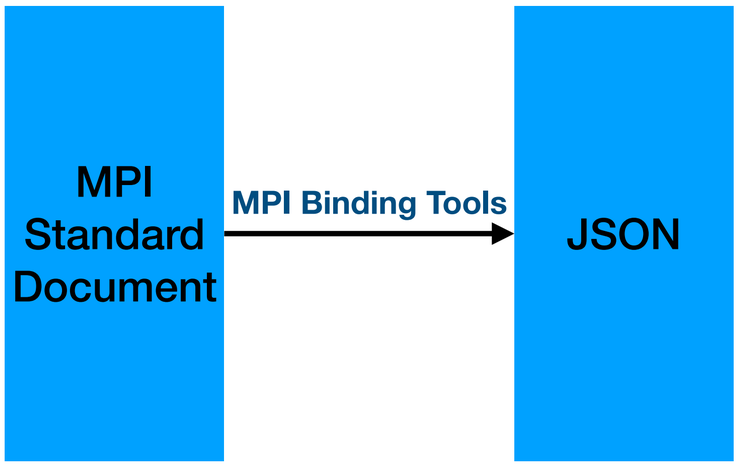
```
compiler:
  cc: {program: mpicc}
  fc: {program: mpifort}
  variants:
    openmp:
      args: [-fopenmp]
criterion:
  n_mpi: {values: [1, 2, 3, 4]}
machine:
  concurrent_run: 4
  nodes: 25
runtime:
  criterions:
    n_mpi: {option: '-np '}
  program: mpirun
```

Profile

Benchmark configuration

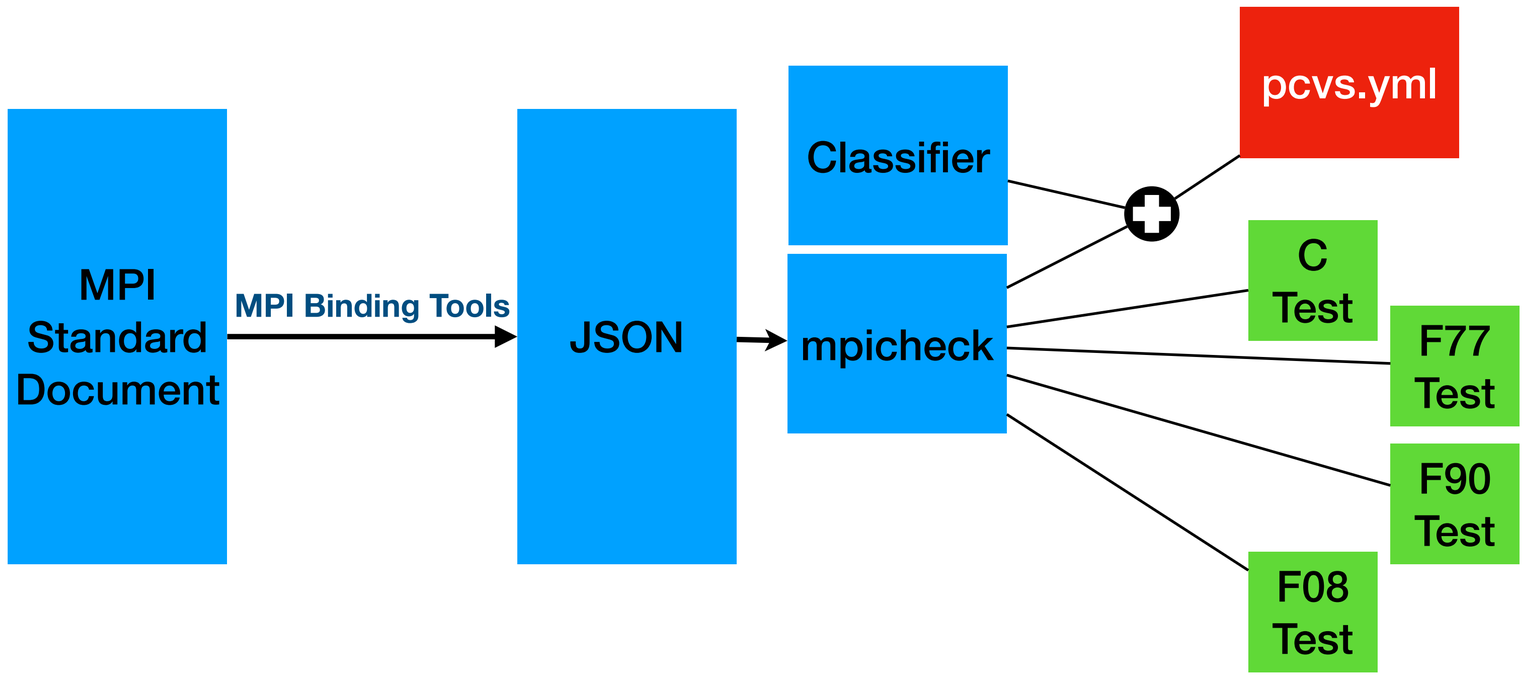
```
lulesh:
    build:
        variants: [ 'openmp' ]
        files: '@BUILDPATH@/Makefile'
        make:
            target: 'all'
    run:
        program: 'lulesh2.0'
        iterate:
            n_node:
                values: [1, 2, 4]
            n_mpi:
                values: {op: 'powerof', of: 3}
```

Workflow




```
{
    "mpi_send": {
        "attributes": {
            "c_expressible": true,
            "callback": false,
            "capitalized": false,
            "deprecated": false,
            "execute_once": false,
            "f08_abstract_interface": true,
            "f08_expressible": true,
            "f90_expressible": true,
            "f90_index_overload": null,
            "f90_use_colons": false,
            "index_upper": false,
            "lis_expressible": true,
            "not_with_mpif": false,
            "predefined_function": null,
            "proxy_render": false
        },
        "name": "MPI_Send",
        "name_f90": null,
        "parameters": [...]
        "return_kind": "ERROR_CODE"
    }
}
```

Workflow

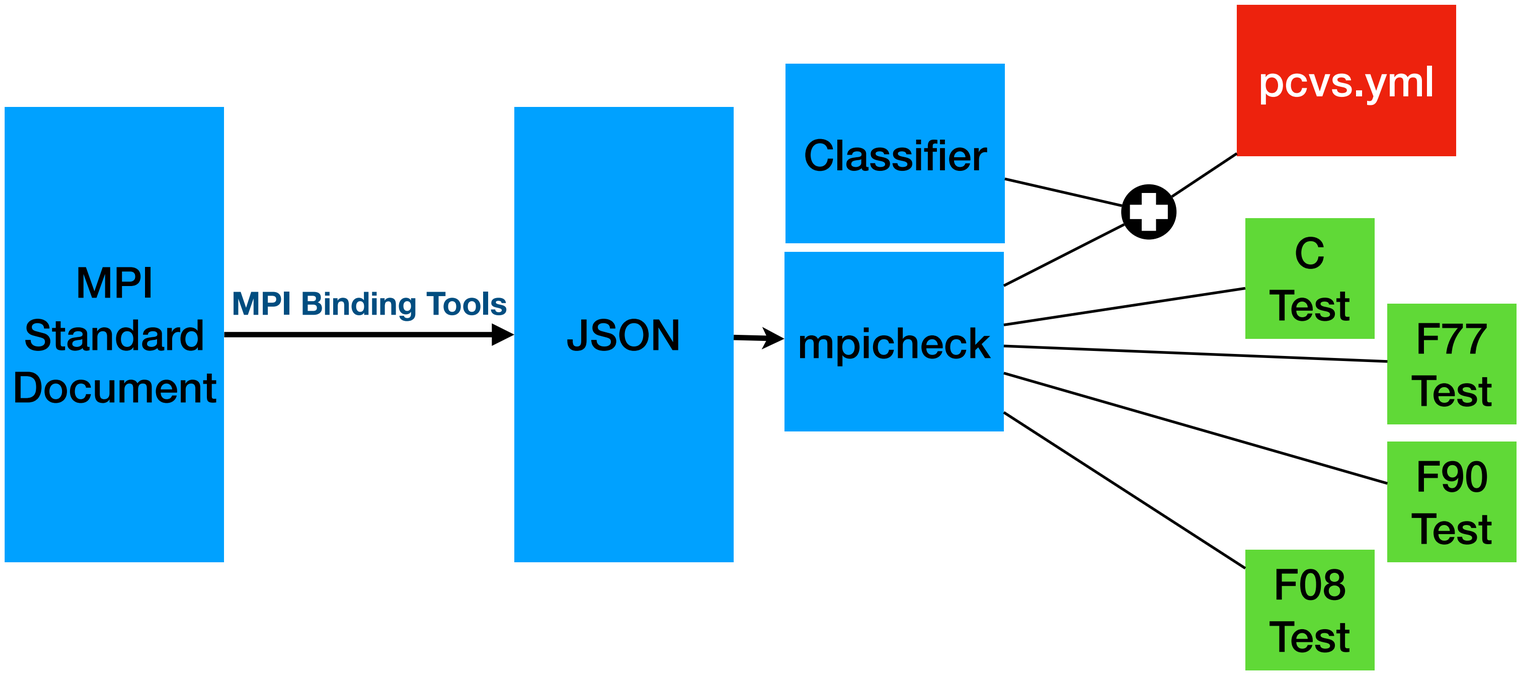



```c
#include <mpi.h>
int main(char argc, char**argv)
{
    /* vars */
    const void *var_0;
    int var_1;
    MPI_Datatype var_2;
    int var_3;
    int var_4;
    MPI_Comm var_5;
    int ret;
    /* calls */
    ret = MPI_Send(var_0, var_1, var_2, var_3, var_4, var_5);
    ret = PMPI_Send(var_0, var_1, var_2, var_3, var_4, var_5);
    return 0;
}
```

```fortran
    program main
    include 'mpif.h'

    TYPE(INTEGER) var_0(10)
    INTEGER var_1
    INTEGER var_2
    INTEGER var_3
    INTEGER var_4
    INTEGER var_5
    INTEGER var_6
    call mpi_send(var_0, var_1, var_2, var_3, var_4, var_5, var_6)
    call pmpi_send(var_0, var_1, var_2, var_3, var_4, var_5, var_6)
    end program main
```

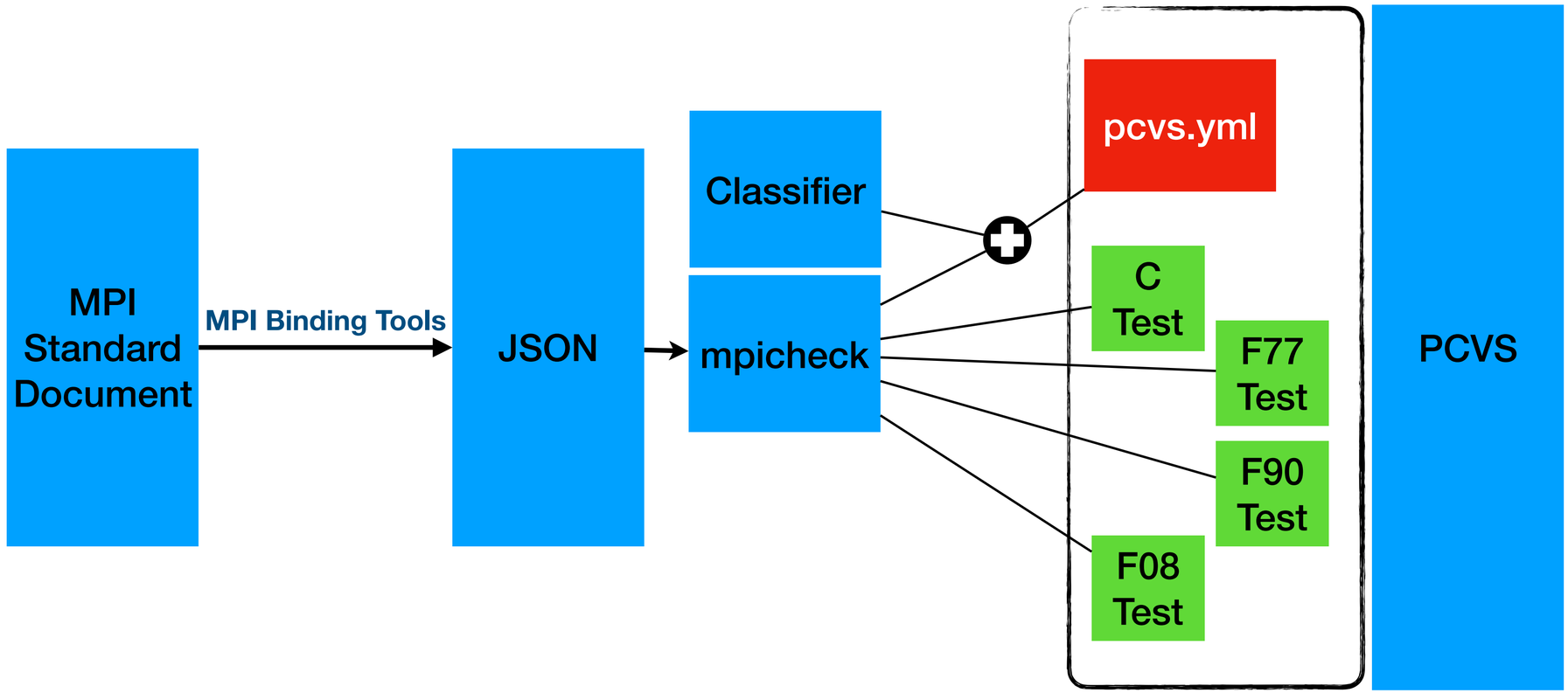
```fortran
program main
        use mpi_f08

        TYPE(INTEGER), DIMENSION(10) :: var_0
        INTEGER :: var_1
        TYPE(MPI_Datatype) :: var_2
        INTEGER :: var_3
        INTEGER :: var_4
        TYPE(MPI_Comm) :: var_5
        INTEGER :: var_6
        call mpi_send(var_0, var_1, var_2, var_3, var_4, var_5, var_6)
        call pmpi_send(var_0, var_1, var_2, var_3, var_4, var_5, var_6)
end program main
```

Workflow




```yaml
MPI_Send_langc:
  tag: [c, functions, STD:1.0]
  build:
    files:
    - MPI_Send.c
    sources:
      cflags: '-Wno-deprecated-declarations -Werror'
MPI_Send_langf77:
  tag: [f77, functions, STD:1.0]
  build:
    files:
    - MPI_Send.f
    sources:
      cflags: -Wno-deprecated-declarations -Werror -ffree-form
MPI_Send_langf90:
  tag: [f90, functions, STD:1.0]
  build:
    files:
    - MPI_Send.f90
    sources:
      cflags: -Wno-deprecated-declarations -Werror -ffree-form
MPI_Send_langf08:
  tag: [f08, functions, STD:1.0]
  build:
    files:
    - MPI_Send.f08
    sources:
      cflags: '-Wno-deprecated-declarations -Werror'
```

Workflow


MPI Standard Document
MPI Binding Tools
JSON
Classifier
mpicheck
pcvs.yml
C Test
F77 Test
F90 Test
F08 Test
PCVS

INITIALIZATION

- **Prepare environment**
  - Check whether build directory is valid
  - Ensure user-defined programs exist
  - Init & expand criterions
  - Init the global Orchestrator
  - Save Configurations into /home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/.pcvs-build
- **Load Test Suites**
  - Locate benchmarks from user directories
  - Extract tests from dynamic definitions (0 found)
  - Extract tests from static definitions (8 found)
- **===> Processing done in 3.204 sec(s)**

SUMMARY

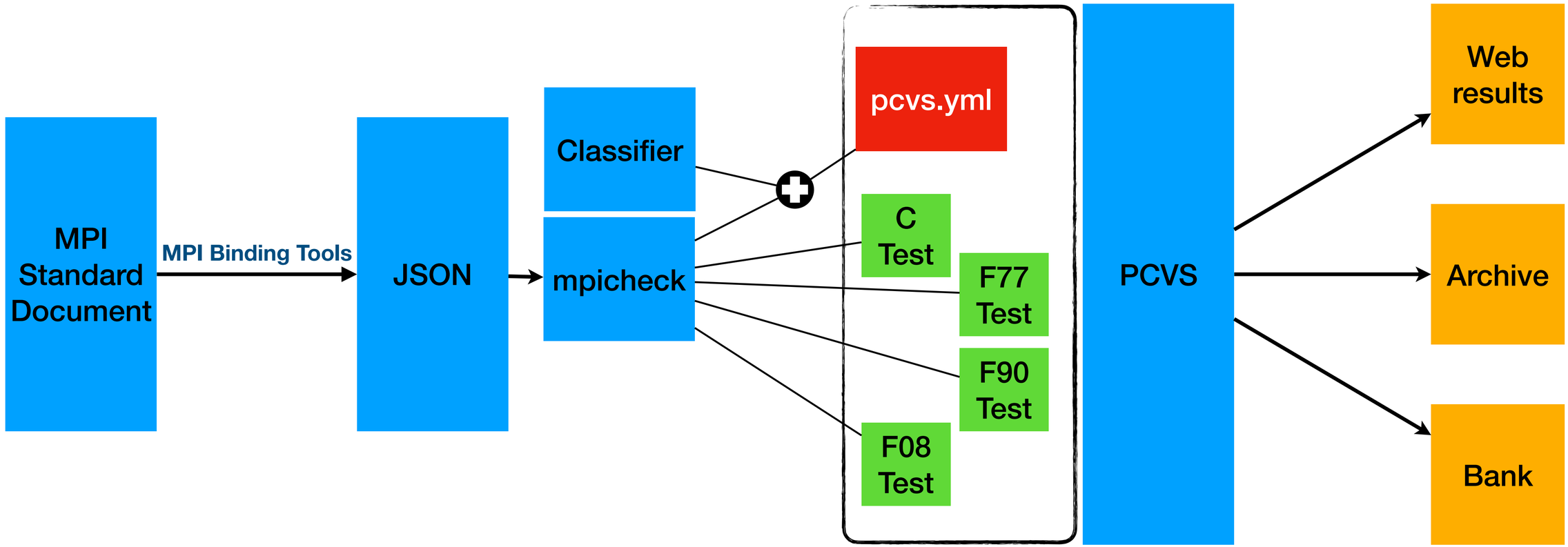
- **Global Information**
  - Date of execution: Mon May 15 14:01:21 2023
  - Run by: Julien Adam <adamj@paratools.com>
  - Active session ID: 170
  - Loaded profile: 'user.mpi'
  - Build stored to: /home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/.pcvs-build
  - Criterion matrix size per job: 1
- **User directories:**
  - MPICHECK: /prog/mpi_meta/tests/functions
- **Globally loaded plugins:**
  - TEST_RESULT_EVAL: BankValidationPlugin
- **Orchestration infos**
  - Test count: 2298
  - Max simultaneous Sets: 4
  - Resource count: 4

EXECUTION

Name	SUCCESS	FAILURE	ERROR	OTHER
mpicheck/2.2	28	0	0	0
mpicheck/4.0	355	471	0	0
mpicheck/3.1	27	0	0	0
mpicheck/2.0	568	4	0	0

0:01:30 Progress 63.2%

Workflow


MPI Standard Document
MPI Binding Tools
JSON
Classifier
mpicheck
pcvs.yml
C Test
F77 Test
F90 Test
F08 Test
PCVS
Web results
Archive
Bank

Complete examples available on:

https://mpicheck.pcvs.io/

PCVS Home About

status View -- FAILURE

Show entries

Search: MPI_Isend_c

Name	status	Elapsed time (s)
mpicheck/4.0/MPI_Isend_c_langc	FAILURE	0.05

```
mpicc -Wno-deprecated-declarations -Werror -Wno-error=line-truncation  /home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/tests/functions/4.0/MPI_Isend_c.c   -o /home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/.pcvs-build/test_suite/mpicheck/4.0/MPI_Isend_c_langc
```

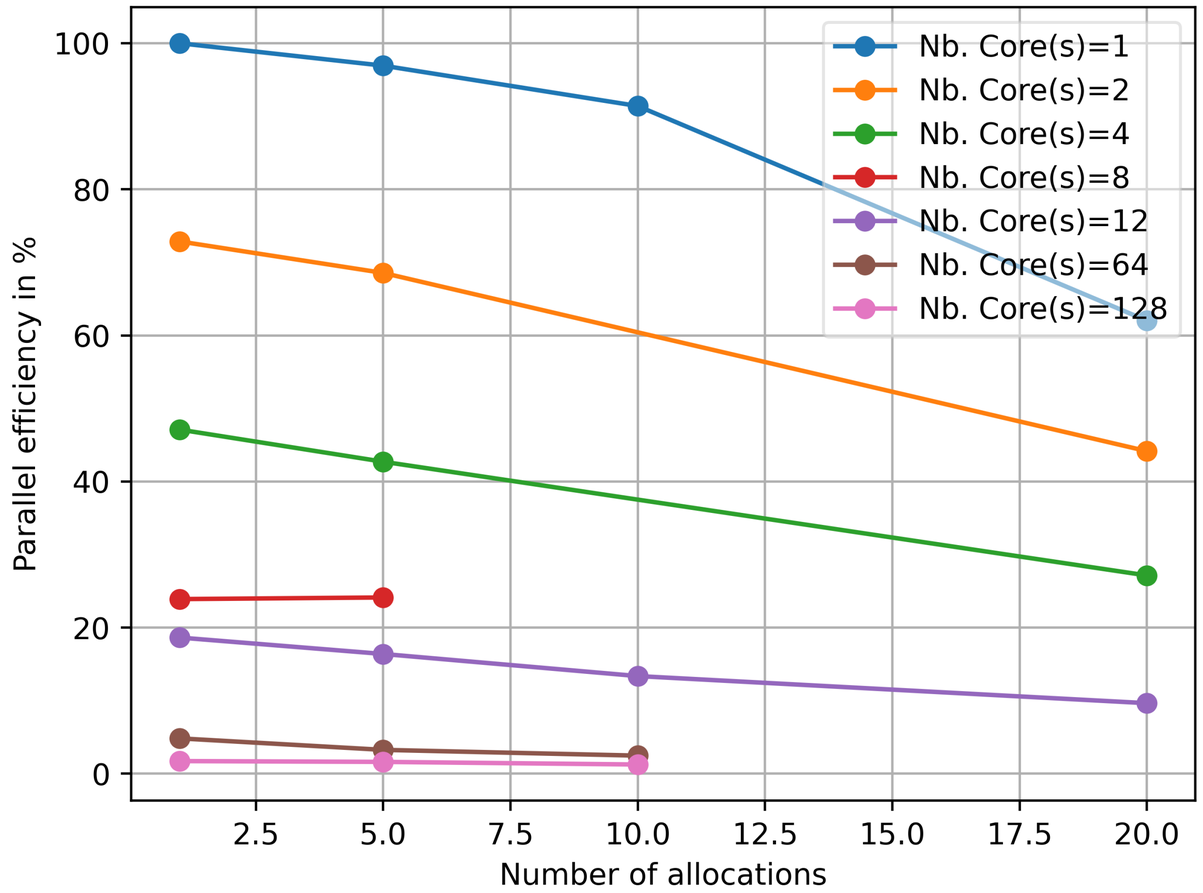
```
/home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/tests/functions/4.0/MPI_Isend_c.c: In function ‘main’:
/home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/tests/functions/4.0/MPI_Isend_c.c:14:11: error: implicit declaration of function ‘MPI_Isend_c’; did you mean ‘MPI_Isend’? [-Werror=implicit-function-declaration]
   14 |     ret = MPI_Isend_c(var_0, var_1, var_2, var_3, var_4, var_5, var_6);
      |           ^~~~~~~~~~
      |           MPI_Isend
/home/adamj/.lnk/code/pcvs-benchmarks/MPI/mpicheck/tests/functions/4.0/MPI_Isend_c.c:15:11: error: implicit declaration of function ‘PMPI_Isend_c’; did you mean ‘PMPI_Isend’? [-Werror=implicit-function-declaration]
   15 |     ret = PMPI_Isend_c(var_0, var_1, var_2, var_3, var_4, var_5, var_6);
      |           ^~~~~~~~~~~
      |           PMPI_Isend
cc1: all warnings being treated as errors
```

	mpicheck/4.0/MPI_Isend_c_langf08	FAILURE	0.13

Measurements

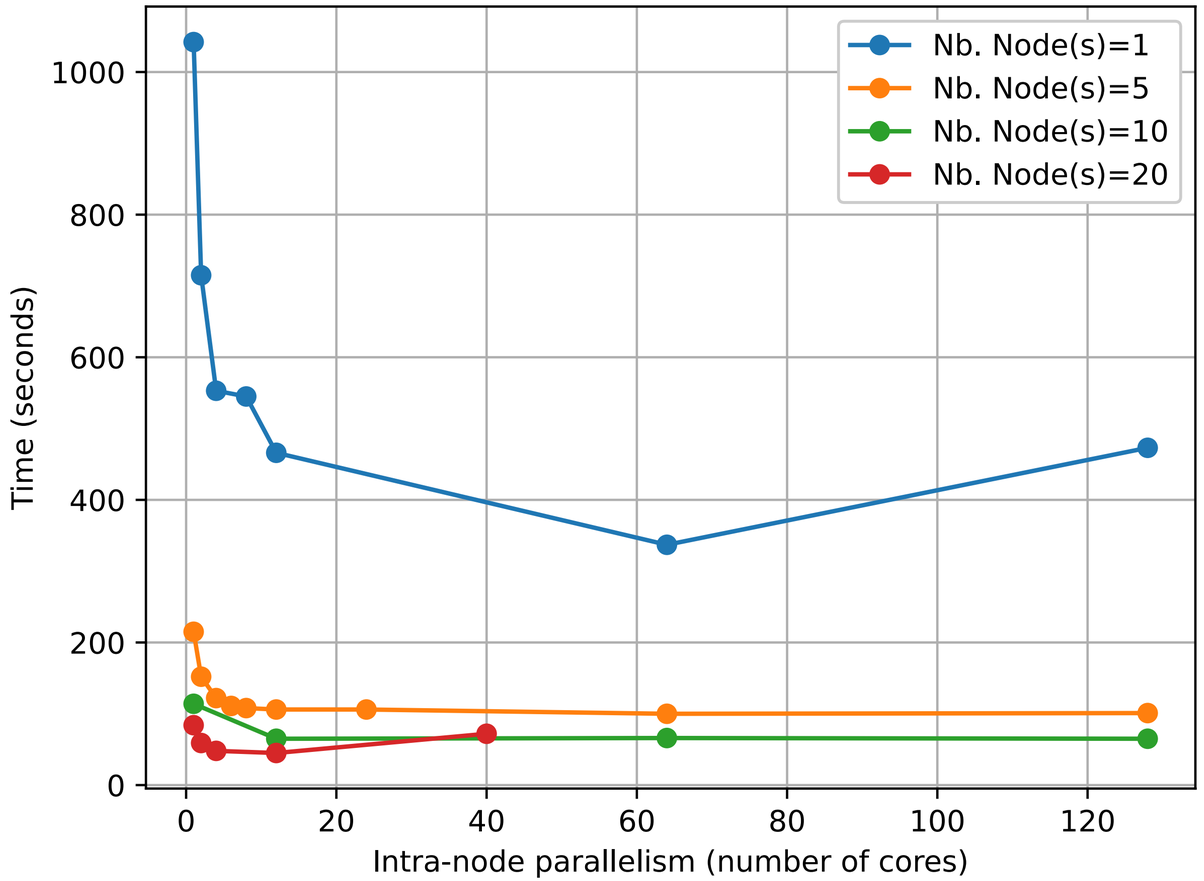
1st-degree parallelism

jobs are spread over multiple batch-manager allocations



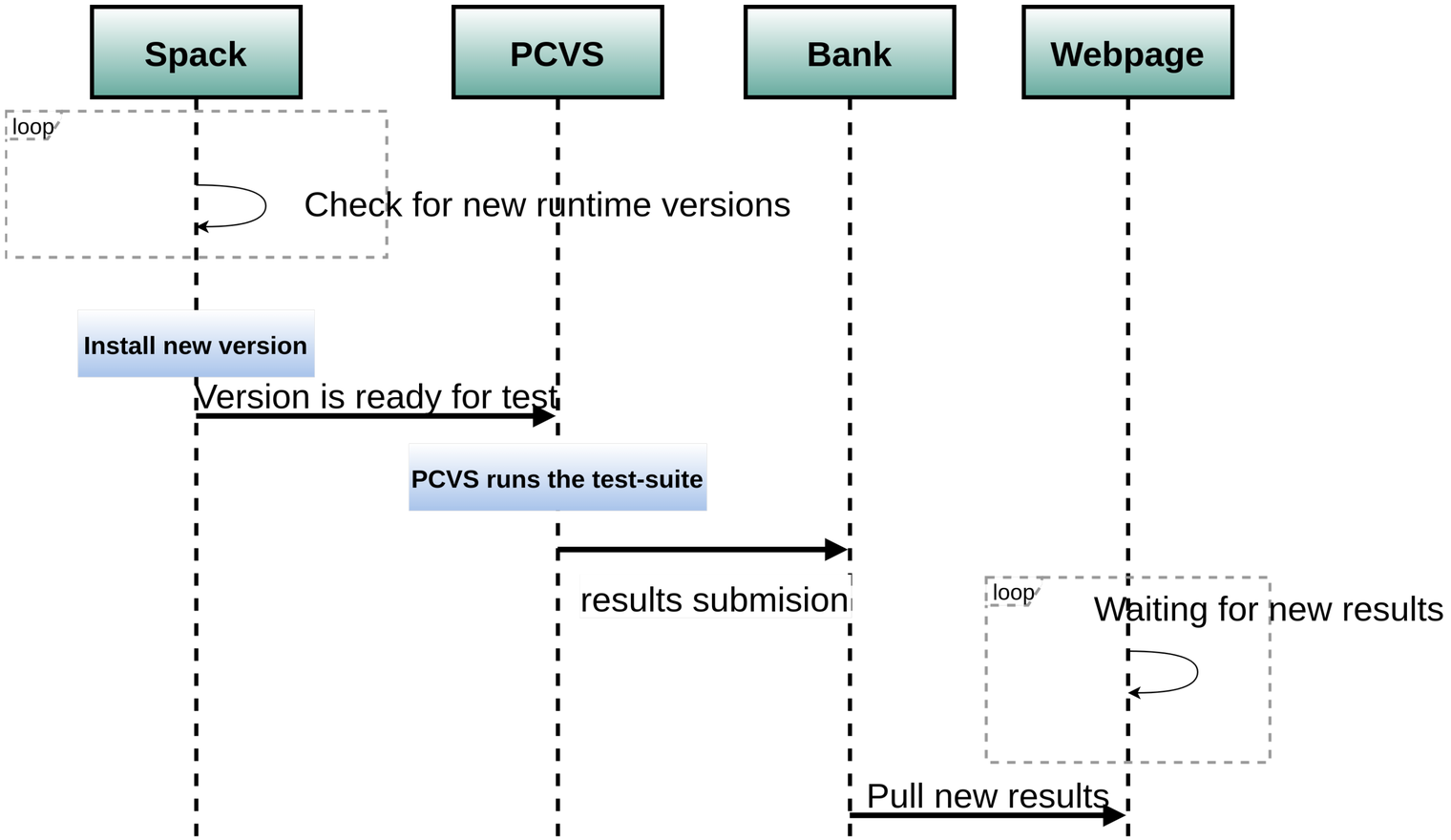
2nd-degree parallelism

Jobs are distributed over cores (1test = 1core)



Deployment: Automatic Validation

- First iteration: API availability = check if MPI function are defined.
- Relies on Spack to track new versions



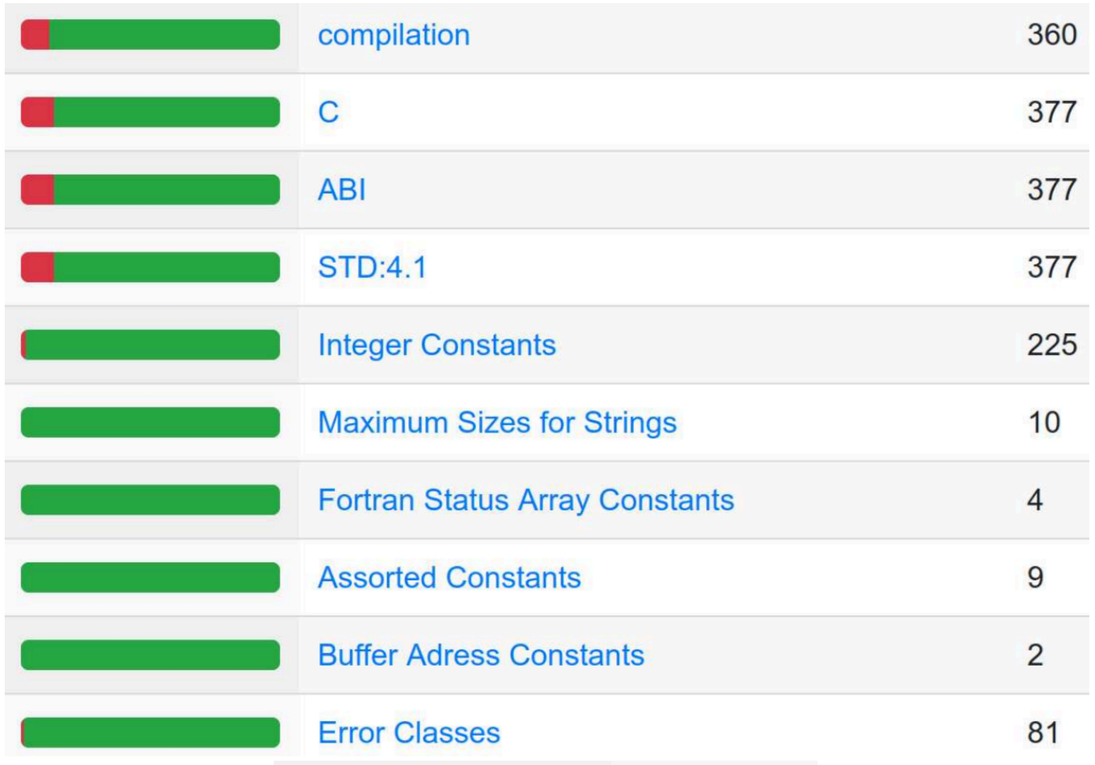
Deployment: Automatic Validation

Session ID	State	Successes	Failures	Other	Info
197	COMPLETED	1977	11	0	mpich 4.2a1 (from main, hash 0f72280c7)
72	COMPLETED	1977	11	0	mpich 4.1.1
71	COMPLETED	1977	11	0	mpich 4.1
70	COMPLETED	1976	12	0	mpich 4.0.3
67	COMPLETED	1976	12	0	mpich 4.0
69	COMPLETED	1945	43	0	mpich 4.0.2
68	COMPLETED	1945	43	0	mpich 4.0.1
219	COMPLETED	1671	317	0	MPC 4.2.0
221	COMPLETED	1652	336	0	MPC 4.1.0
227	COMPLETED	1651	337	0	OpenMPI 5.0.0rc11
78	COMPLETED	1470	518	0	mvapich2 2.3.7-1
77	COMPLETED	1470	518	0	mvapich2 2.3.7

Future Work

- More functional testing
- Exhaustive use-case scenarios (LLM?)
- Classifier Extension

- PCVS: Finer-grain job distribution
- PCVS: Integration with existing solutions (Gitlab, Jenkins…)



Thank you for your attention

Reports: https://mpicheck.pcvs.io/
Function Table: https://mpicheck.pcvs.io/std/


(demo) adamj@saturn:~/pcvs$