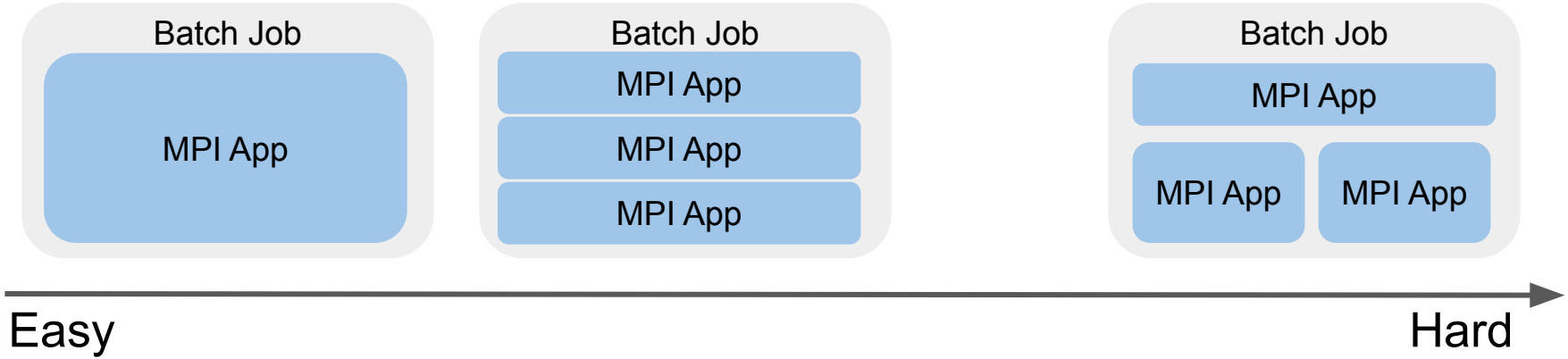
MPI Support in Parsl

and Globus Compute*

Yadu Babuji

* in due course, at the appropriate juncture, in the fullness of time

MPI, why is it so hard!



Proposed approach - Application side

bash_app.py

```
@bash_app
def my_mpi_application(epochs: int, resource_specification: Dict):
    # PARSL_DEFAULT_PREFIX will resolve to `mpiexec -n 4 -ppn 2 -hosts NODE001,NODE002`
    return f"$PARSL_DEFAULT_PREFIX my_mpi_simulation {epochs}"

# Resources in terms of nodes and how ranks are to be distributed are set on a per app
# basis via the resource_spec dictionary.
resource_spec = {
    "NUM_NODES": 2,
    "RANKS_PER_NODE": 2,
}
future = my_mpi_application(epochs=10, resource_specification=resource_spec)
```

Proposed approach - Runtime configuration

config.py

Raw

```
nodes_per_task = 2
tasks_per_block = 16
config = Config(
    executors=[
        HighThroughputExecutor(
            label='mpiapps',
            enable_mpi_mode=True,
            address=address_by_hostname(),
            start_method="fork",            # Needed to avoid interactions between MPI and os.fork
            max_workers=tasks_per_block,
            cores_per_worker=1e-6,          # Prevents limiting workers to cores on one node
            provider=PBSProProvider(
                ...
                launcher=SimpleLauncher(),  # Launches only a single executor per block
                select_options="ngpus=4",
                nodes_per_block=nodes_per_task * tasks_per_block,
            ),
        ),
    ]
)
```

What we are missing

- Fault handling: Reclaiming nodes when a launcher/MPI fails
  - Often a failure of a node triggers a shutdown of the entire batch job
  - Launcher failures often hang (eg, theta with aprun)
- Resource specification covers the most minimal set of descriptors
  - GPU/Accelerator binding like with `srun --gpu-bind=<map_gpu/mask_gpu/...>`
  - Specify memory per rank/cpu like: `srun --mem=<size>`
- Configuration cleanliness around specifying the right launcher from (aprun/srun/mpirun/mpiexec…) for the specific target config.
- Potentially a new **`mpi_app`** ?

Please help!

- Read the MPI Functions doc for background, add comments
- Try the experimental feature branch with your MPI application
- Tell us what worked vs what didn’t
- If you have ideas:
  - Add an issue OR
  - Push code over a PR :)