justIN/GPUs mini tutorial + questions

Andrew McNab
University of Manchester

DUNE Collaboration Meeting, May 2025

GPUs mini tutorial + questions

- justIN recap
- GPU resources available
- Standard justIN hello world
- GPU hello world to the grid
- GPU hello world to NERSC/Perlmutter
- Questions from Computing to AI/ML user communities

justIN recap

- justIN was developed by DUNE as a workflow management system which was designed to work with MetaCat, Rucio, and GlideinWMS from day one
  - MetaCat = file catalogue: “what files are like this?”
  - Rucio = replica catalogue: “where are copies of this file?”
  - GlideInWMS = runs jobs at sites for justIN
- You ask justIN to run copies of your job to process all the files that match some query and it magically does it for you
- You don’t need to know where things run or where files are

Rucio scopes and justIN

- justIN leverages Rucio's scopes to decide who can do what
  - Each file in Rucio is in a scope so we say SCOPE:FILENAME
- justIN says that each scope is owned by a group
  - **Only people in that group can create workflows that save output files in the scope**
- Currently two active groups, more to follow:
  - /dune = everyone in DUNE
  - /dune/production = just the production team
- Scopes will let us delegate control to working groups and users

GPU resources available

- GPUs are available via justIN at
  - NERSC/Perlmutter
  - and on the grid at Manchester and QMUL in the UK
    - RAL-PPD has them online and they are being added to OSG
    - More welcome - please get in touch so we can add them
- At NERSC, DUNE has an annual allocation
  - Currently 20,000 GPU hours with A100s, but we can request more
  - But only workflows writing to /dune/production scopes currently enabled
- At the Grid sites, it's first come first served,
  - They tend to be underused and all /dune can access them
  - eg Manchester has 36 Tesla T4s = 316,000 GPU hours spread over a year

Standard justIN hello world

- If you're going to try this out, **please follow the justIN tutorial first**
  - It includes a GPU section now
  - https://justin.dune.hep.ac.uk/docs/tutorials.dune.md
- In short, you submit a workflow by giving the jobscript you have written in Bash and specifying any options to justIN
- If you have a Hello World jobscript, you can run 10 copies like this:
  - `justin simple-workflow --monte-carlo 10 --jobscript hello-world.jobscript`
- `--monte-carlo` is a fake query that just gives you the numbers 01 to 10
- The `--mql` option can be used to query MetaCat give a list of files instead

Workflow outputs

- Each job in a workflow produces
  - A jobscript log file, the tail of which can be viewed in the dashboard
  - A .logs.tgz file put in Rucio managed storage for ~1 month
    - Contains all files matching *.log including jobscript.log
  - Any output files you nominate via patterns to be put in Rucio storage
    - (Or to dCache scratch disk at Fermilab)
- You can find all these files through the justIN dashboard or in Rucio datasets
  - See the tutorial and documentation for details

GPU hello world

- Fetch `https://github.com/DUNE/dune-justin/blob/main/testing/hello-gpu.jobscript`
- Run 10 copies like this:
  - `justin simple-workflow --gpu --monte-carlo 10 --jobscript hello-gpu.jobscript`
- This does the same as Hello World, but the jobs only run on worker nodes with GPUs
  - HTCondor/GlideInWMS is told to allocate 1 GPU to the job
  - You can request CPU memory and number of CPUs, but not yet:
    - GPU memory or GPU versions

GPU environment

- justIN asks HTCondor/GlideInWMS to allocate a GPU from the pool the pilot job has got from the local batch system
- The local batch system might hide other GPUs on the worker node from us, or it might not
- You might be running alongside other GPU jobs on the worker node from other experiments or inside the DUNE pilot job
- justIN runs your jobscript inside an Apptainer container with --nv
  - This maps the /dev/nvidia devices and correct CUDA files from the worker node into the container, and sets LD_LIBRARY_PATH
  - **So we do not need to put NVIDIA licensed libraries in cvmfs**

GPU job status page (top half)

justin.dune.hep.ac.uk

justIN Dashboard Workflows Jobs AWT Sites Storages Docs amcnab@fnal.gov

Jobsub ID 401491.1@justin-prod-sched01.dune.hep.ac.uk

Jobsub ID	401491.1@justin-prod-sched01.dune.hep.ac.uk	
Workflow ID	7087	
Stage ID	1	
User name	amcnab@fnal.gov	
HTCondor Group	group_dune	
Requested	Processors	1
	GPU	Yes
	RSS bytes	2097152000 (2000 MiB)
	Wall seconds limit	80000 (22 hours)
Submitted time	2025-05-21 16:02:12	
Site	UK_Manchester	
Entry	IceCube_UK_Manchester_ce01_gpu	
Last heartbeat	2025-05-21 16:12:40	
From worker node	Hostname	wn5915090.tier2.hep.manchester.ac.uk
	cpuinfo	Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz
	OS release	Scientific Linux release 7.9 (Nitrogen)
	Processors	1
	RSS bytes	2097152000 (2000 MiB)
	Wall seconds limit	259200 (72 hours)
	GPU	Tesla T4 570.124.06 7.5 90.04.38.00.03 14913MiB
	Inner Apptainer?	True
Job state	finished	
Allocator name	justin-allocator-pro.dune.hep.ac.uk	
Started	2025-05-21 16:12:23	
Input files	monte-carlo-007087-000001	
Jobscript	Exit code	0
	Real time	0m (1s)
	CPU time	0m (1s = 99%)
	Max RSS bytes	41918464 (39 MiB)
Outputting started	2025-05-21 16:12:25	
Output files		
Finished	2025-05-21 16:12:40	
Saved logs	justin-logs:401491.1-justin-prod-sched01.dune.hep.ac.uk.logs.tgz	
List job events	Wrapper job log	

Site	UK_Manchester	
Entry	IceCube_UK_Manchester_ce01_gpu	
Last heartbeat	2025-05-21 16:12:40	
From worker node	Hostname	wn5915090.tier2.hep.manchester.ac.uk
	cpuinfo	Intel(R) Xeon(R) Silver 4210R CPU @ 2.40GHz
	OS release	Scientific Linux release 7.9 (Nitrogen)
	Processors	1
	RSS bytes	2097152000 (2000 MiB)
	Wall seconds limit	259200 (72 hours)
	GPU	Tesla T4 570.124.06 7.5 90.04.38.00.03 14913MiB
	Inner Apptainer?	True
Job state	finished	
Allocator name	justin-allocator-pro.dune.hep.ac.uk	

GPU job status page (bottom half)

- This is the output of the GPU Hello World jobscript
- It's mostly just these two commands:

```
# Check the GPU environment
printenv | grep -i cuda
nvidia-smi
```

- CUDA_VISIBLE_DEVICES is set to tell you which of the GPUs you can use if more than one are visible
  - Depends on the local set up

Jobscript log (last 10,000 characters)

```
CUDA_VISIBLE_DEVICES=GPU-373e15de-673e-812f-10f4-6547aef25ae2
Wed May 21 17:12:23 2025
+-----------------------------------------------------------------------------------------+
| NVIDIA-SMI 570.124.06             Driver Version: 570.124.06     CUDA Version: 12.8     |
|-----------------------------------------+------------------------+----------------------+
| GPU  Name                 Persistence-M | Bus-Id          Disp.A | Volatile Uncorr. ECC |
| Fan  Temp   Perf          Pwr:Usage/Cap |           Memory-Usage | GPU-Util  Compute M. |
|                                         |                        |               MIG M. |
|=========================================+========================+======================|
|   0  Tesla T4                       On  |   00000000:3B:00.0 Off |                    0 |
| N/A   41C    P8             10W /   70W |     117MiB /  15360MiB |      0%   E. Process |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   1  Tesla T4                       On  |   00000000:5E:00.0 Off |                    0 |
| N/A   28C    P8              9W /   70W |      27MiB /  15360MiB |      0%   E. Process |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   2  Tesla T4                       On  |   00000000:60:00.0 Off |                    0 |
| N/A   27C    P8              9W /   70W |      67MiB /  15360MiB |      0%   E. Process |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   3  Tesla T4                       On  |   00000000:86:00.0 Off |                    0 |
| N/A   26C    P8              9W /   70W |      91MiB /  15360MiB |      0%   E. Process |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   4  Tesla T4                       On  |   00000000:AF:00.0 Off |                    0 |
| N/A   26C    P8              9W /   70W |       3MiB /  15360MiB |      0%   E. Process |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+
|   5  Tesla T4                       On  |   00000000:D8:00.0 Off |                    0 |
| N/A   29C    P8             13W /   70W |       3MiB /  15360MiB |      0%   E. Process |
|                                         |                        |                  N/A |
+-----------------------------------------+------------------------+----------------------+

+-----------------------------------------------------------------------------------------+
| Processes:                                                                              |
|  GPU   GI   CI              PID   Type   Process name                        GPU Memory |
|        ID   ID                                                               Usage      |
|=========================================================================================|
|  No running processes found                                                             |
+-----------------------------------------------------------------------------------------+
Hello gpu 000001
```

justIN time: 2025-05-21 16:12:46 UTC **justIN version: 01.03.01**

GPU at NERSC hello world

- Run 10 copies like this:
  - `justin simple-workflow --gpu --monte-carlo 10 --jobscript hello-gpu.jobscript --scope testpro --site US_NERSC-GPU`
- This forces the jobs to run with a GPU each at NERSC/Perlmutter
  - HTCondor / GlideInWMS / HEPCloud does some special magic!
  - You can get the outputs as normal
- We currently have to give a Production scope (eg testpro)
  - Which means you need to be in the Production team
  - Since we have a finite amount of NERSC quota, we need ensure it does not get all used up accidentally in one go
    - With Grid you have more time to spot a mistake and kill a bad workflow

Questions

- What do **you** need in terms of **limiting** matches to
  - GPUs by GPU memory
  - GPUs by GPU hardware model
    - How to express that?
  - Other GPU/driver specific things?
  - This gets you **less** GPUs of course!
- Who needs to be able to use GPUs at NERSC?
  - Production team vs working groups' production teams vs individual users

Summary

- We have useful amounts of GPU capacity available
  - At multiple Grid sites
  - At NERSC/Perlmutter
- All of which can be accessed through the standard justIN interface with a trivial change to the submission command
- We will be able to use the existing justIN/Rucio mechanisms for controlling access via groups to limit usage of the DUNE NERSC quota
- **But Computing needs help in gathering requirements and use cases to determine what other features are needed**