



# TRITON ON ROCM



# PROGRESS UPDATE

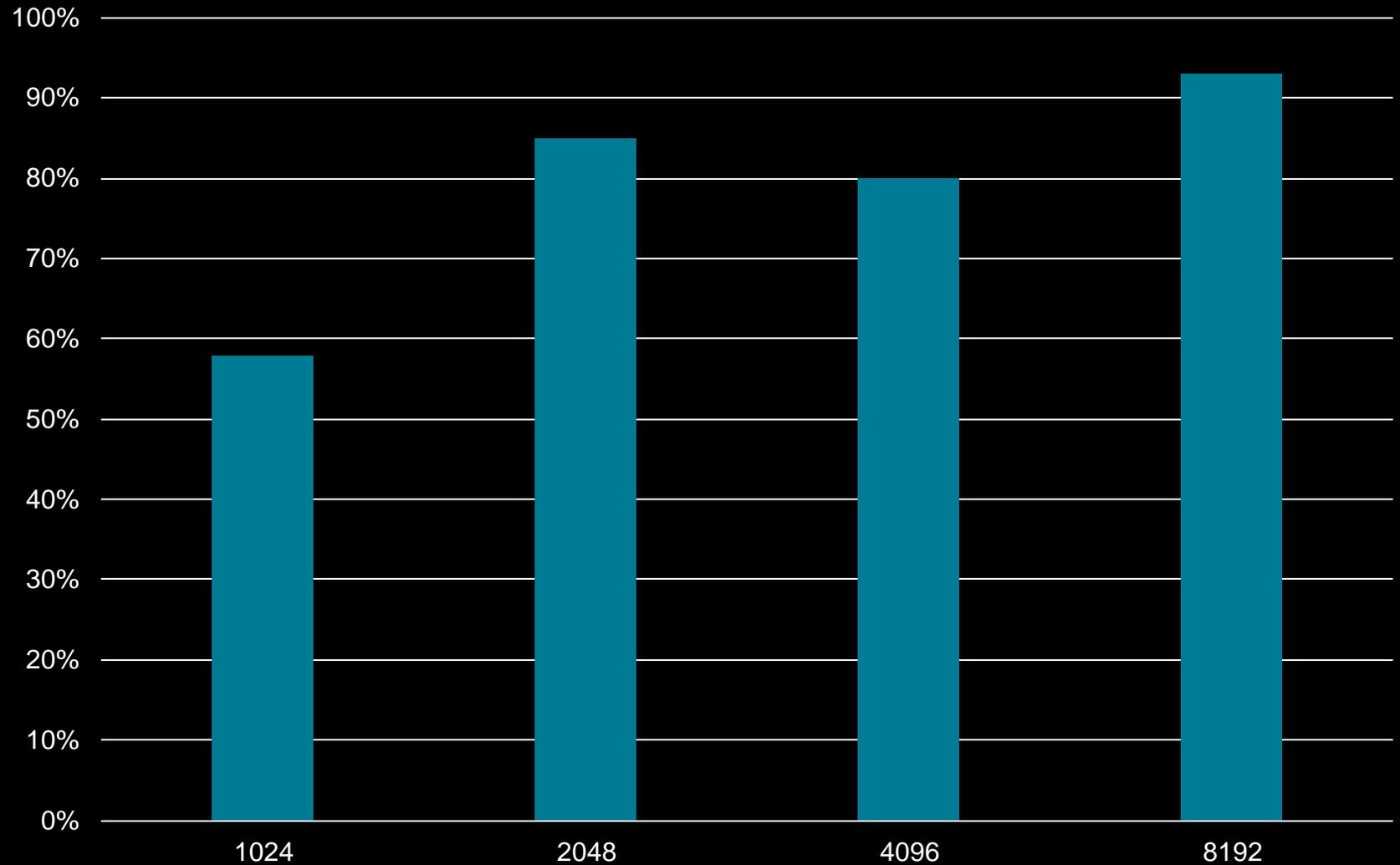
- Support began ~a year ago
  - Originally explored as dependency of DeepSpeed
  - Pytorch 2.0 accelerated our involvement
- Where is the code?
  - Some code is upstream
  - Fork: <http://github.com/ROCmSoftwarePlatform/triton> ([track](#) our progress!)
  - Soon to be an official "3rd-party backend"
- Hardware Support
  - Initial support for CDNA2 GPUs (mi100/mi200)
  - Experimental support on RDNA GPUs
  - Device detection at runtime
- Full support for `torch.compile` in Pytorch 2.0
- When calling into `triton.compile`, use `device_type="hip"`

# RECENT COMPLETED ITEMS

- Configurable warp size/wavefront size for Triton
  - CDNA2 devices use 64 thread wavefronts [\[1\]](#)
- MatrixCore enablement (MFMA instructions)
  - 32x32 fp16/bf16 initially, other variants coming [\[1\]\[2\]](#)
- Flash attention Support [\[1\]](#) [\[2\]](#)
  - Blocksize 64
  - Keep resultant tensors in registers [\[1\]](#) [\[2\]](#) [\[3\]](#)
- Shared memory bank conflict mitigation for dot operations [\[1\]](#) [\[2\]](#)
- Adjustable workgroup-size to optimally use VGPRs [\[1\]](#)
- Vector loads [\[1\]\[2\]](#)

# GEMM PERFORMANCE

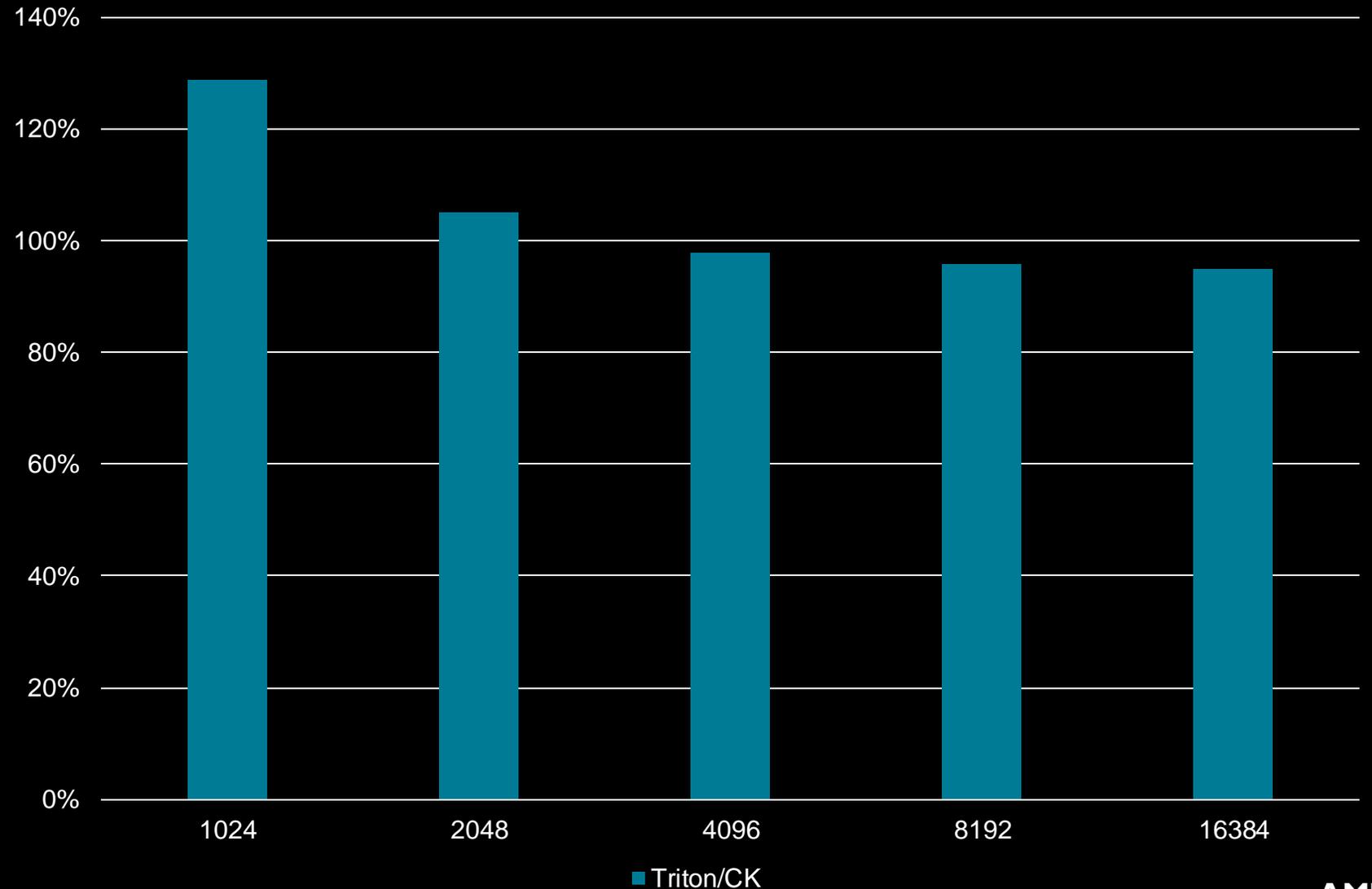
- $M=N=K=1024 - 8192$



■ Triton/rocBLAS

# FA FWD PERFORMANCE

- Embedding size = 3072
- Batch = 4
- seqLen = 1024 - 16384



**THANK YOU!**