Summarize the article

Summary of “TPU v4: An Optically Reconfigurable Supercomputer for Machine Learning with Hardware Support for Embeddings”

- **TPU v4 Overview:**
  TPU v4 is Google’s fifth-generation domain-specific architecture for machine learning (ML), and the third to reach supercomputer scale. It addresses the rapid evolution and increased scale of ML models like large language models (LLMs) and deep learning recommendation models (DLRMs).
- **Optical Circuit Switches (OCS):**
  TPU v4 introduces optical circuit switches that make interconnect topology dynamically reconfigurable, greatly enhancing scalability, reliability, availability, performance, and modularity. OCS allows the supercomputer to tolerate host failures and enables various 3D torus and twisted torus topologies—offering much higher bisection bandwidth than previous fixed electrical network designs and at a fraction of cost/power compared to alternatives like Infiniband.
- **SparseCore for Embeddings:**
  TPU v4 adds SparseCore units, specialized dataflow processors dedicated to sparse embedding operations key to recommendation models. SparseCores boost embedding model training throughput 5x–7x compared to previous approaches, while only using 5% of die area and system power.
- **Scale & Performance:**
  TPU v4 supercomputers scale up to 4096 chips, a 4x increase over TPU v3. Deployed since 2020, they provide 2.1x higher performance and 2.7x higher performance per watt over TPU v3. Compared to NVIDIA A100 (similar era, 7nm process), TPU v4 is 1.2x–1.7x faster, 1.3x–1.9x more energy efficient, and is even faster relative to Graphcore IPU Bow (4.3x–4.5x).
- **Energy, Carbon, and Efficiency:**
  TPU v4 supercomputers in Google Cloud’s energy-optimized datacenters are estimated to consume 2x–6x less energy and produce 20x less CO2-equivalent emissions than comparable DSAs in traditional on-premises datacenters.
- **Topology and Scheduling Flexibility:**
  Users can tune system topology for specific ML workloads (data/model/pipeline parallelism) to maximize job performance. Twisted torus topologies, made easy with OCS, further enhance bandwidth and minimize latency for communication-heavy models.
- **Hardware Summary:**
  Manufactured in 7nm, TPU v4 chips have double the matrix multipliers, increased clock

speed (1050 MHz), more on-chip memory (160 MB SRAM), higher memory bandwidth (1200 GB/s HBM), and more SparseCores compared to TPU v3. Each chip has 2 TensorCores and 4 SparseCores.

- **Workload Evolution:**
  The ML workload mix at Google has shifted over time: LLMs and Transformer-based models have rapidly eclipsed older RNN and CNN workloads, driving architectural changes toward bandwidth-optimized, large-scale compute fabrics.
- **Deployment and Utilization Advantages:**
  Optical switching allows incremental deployment, improved scheduling, higher supercomputer utilization, faster rollout, and better security isolation between cloud tenants.
- **Environmental Impact:**
  By combining energy-efficient architecture and sourcing ~90% carbon-free electricity, Google's TPU v4 deployments dramatically reduce the carbon footprint for ML training—critical given exploding model size and demand for sustainability.
- **Conclusion:**
  TPU v4 combines leading performance and scalable architecture with environmental sustainability and flexibility—making it the workhorse for Google's largest ML workloads, including training massive LLMs such as LaMDA, MUM, and PaLM.

Key Innovations:

- Optically reconfigurable interconnect for large ML supercomputers
- Dedicated hardware acceleration for embeddings
- Rapid, modular, and energy-efficient system deployment
- Advanced scheduling for availability and security
- Industry-leading performance and energy efficiency at scale

If you wish to dive into specific architectural diagrams, performance comparisons, or deployment scenarios, let me know.

⁂

how does the matmal FU work?

The **Matrix Multiply Functional Unit (FU)**—known as the **Matrix Multiply Unit (MXU)**—is central to the TPU v4's TensorCore. Here's how it works, based on the paper:

- **Structure:** Each TensorCore in TPU v4 contains four **128×128 Matrix Multiply Units (MXUs)**. Each chip has two TensorCores, so there are eight MXUs per chip.
- **Operation:** Each MXU takes two 128×128 input matrices and performs multiply-and-accumulate operations:
  - It multiplies the input matrices (typically BF16 or INT8 for ML workloads).
  - The operation is highly parallelized over the entire 128×128 array, so each cycle computes 16,384 (128×128) partial products.

  - The results are accumulated efficiently, reducing memory bandwidth constraints and taking advantage of data reuse in deep learning algorithms.
- **Data Flow:** Data is streamed from high-bandwidth memory (HBM) to the on-chip memory and registers, then into the MXU.
- **Purpose:** These units are designed to accelerate dense matrix multiplications:
  - Core in deep learning ops like **GEMM** (general matrix-matrix multiply), which underlie convolutions and multi-head attention in Transformers and BERT.
- **Throughput:** The architecture's use of large MXUs enables high throughput:
  - TPU v4 achieves peak performance of 275 TFLOPS (BF16/INT8), largely thanks to these MXUs.
- **Pipeline/Integration:** The MXUs are tightly pipelined with adjacent vector processing units (VPUs), which handle vector operations not suitable for the matrix mul units, and with large on-chip memories for rapid data feeding and result storage.

Summary:
The MXUs act like giant fused multiply-accumulate (FMA) arrays: they take matrix tiles from memory, deeply pipeline the computation across a large array, and output the result, all optimized for low latency and high reuse—crucial for deep learning workloads.

If you want more architectural or microarchitectural detail (e.g., data tiling strategies, scheduling, or how these units compare to GPU tensor cores), let me know.

⁂

what it the torus interconnect pattern

The **torus interconnect pattern** in TPU v4 is a network topology used to connect thousands of TPU chips together with high bandwidth and low latency for ML workloads.

What is the torus interconnect?

- A **torus** is a multidimensional (2D or 3D, sometimes called "mesh with wraparound") network where each node (TPU chip or board) connects to its nearest neighbors along each dimension.
- In a 2D torus, edges "wrap around" at the borders—so the network is topologically equivalent to a surface of a donut.
- In a 3D torus (used by TPU v4), each node connects to its neighbors in the X, Y, and Z dimensions, again with wraparound at the boundaries.

How is it used on TPU v4?

- TPU v4 typically uses a **3D torus** interconnect to connect up to 4096 TPU chips.
- Each chip connects directly to 6 neighbors (positive and negative directions in X, Y, Z).
- **Optical circuit switches (OCS)** allow the interconnect wiring to be **dynamically reconfigured** at hardware level—enabling flexibility for workload demands (e.g., forming

twisted tori, healing around failed chips).

Advantages for ML supercomputers:

- The torus offers **high bisection bandwidth** (the ability to move a lot of data between halves of the system), critical for large parallel training tasks.
- The wraparound connections eliminate network “edges,” reducing bottlenecks and improving job placement flexibility.
- The topology naturally “scales out” for larger supercomputers, while reconfigurability (thanks to OCS) means the network can be optimized for different ML jobs or fault tolerance needs.

Summary Table

Topology	Description	Used In	Key Features
2D/3D Mesh	Edges do not wrap around	Older TPUs	Simple, less fault tolerant
2D/3D Torus	Edges wrap, nodes link to other side	TPU v3 & v4	High bandwidth, no “edges”
Twisted Torus	Special wrap & linking for more bandwidth	TPU v4 (OCS)	Optimized, reconfigurable

TPU v4’s optical torus interconnect makes it possible to efficiently train huge ML models using thousands of chips with minimal communication overhead. If you want a diagram or further details about twisted torus or OCS fault recovery, let me know!

⁂

how does the TPU handle flexible sized matmul?

TPU handles **flexible-sized matrix multiplication (matmul)** primarily through **tiling, partitioning, and dataflow orchestration**:

- **Tiling to Fixed MXU Size:**
  The Matrix Multiply Units (MXUs) in TPU are designed for efficient 128×128 block matrix multiplies. For matrices larger or smaller than 128×128:
    - **Large matrices:**
      These are partitioned into 128×128 tiles. Each tile is dispatched and calculated independently in MXUs, and results are accumulated.
    - **Small matrices:**
      The TPU pads or groups smaller matrices into full 128×128 blocks internally, or, for batching, packs multiple small matmuls together.
- **Software Scheduling:**
  XLA (Accelerated Linear Algebra) compiler and runtime handle matrix partitioning, padding, and scheduling across the chip’s MXUs, making the process transparent to DL programmers and frameworks (like TensorFlow or JAX).
- **Input Packing/Batching:**
  For workloads with many small matmul ops (such as attention heads in NLP), the TPU

runtime will **batch and pack** these into full MXU blocks to maximize hardware utilization.

- **On-chip Buffering & Dataflow:**
  The design includes large on-chip memory buffers; partial products or edge cases from irregular tiling are stored and handled efficiently.
- **Special Cases:**
  If matrix shapes are highly irregular or batch size is small, the compiler can split work across several TPU chips and aggregate results post-multiply.
  Sparse or non-dense matmuls (as in embedding lookups) are routed to **SparseCores**, not MXUs.
- **High Utilization:**
  All these strategies—tiling, packing, and batching—keep TPU MXUs well utilized, even for non-ideal or unusual matrix sizes typical in real ML workloads.

Summary:
TPU v4's MXUs operate most efficiently with 128×128 dense matrices, but flexible tiling, batching, and compiler-level optimizations allow it to transparently support arbitrary matrix sizes with high performance. Sparse/irregular cases are handled in dedicated hardware units (SparseCores).

⁂

why did they compare to a100 and not h100?

Google compared **TPU v4 to NVIDIA A100 and not H100** for these main reasons:

- **Timeframe and Fairness:** TPU v4 and A100 were both launched and deployed around 2020, using a similar 7nm fabrication process. H100 is a newer chip (launched in 2022) and uses a more advanced 4nm process, so direct comparisons would not be fair to the TPU v4's generation.
- **Availability:** At the time TPU v4 was designed, deployed, and benchmarked, H100 GPUs were not available in the cloud, at scale, or in production environments at AWS, Azure, or Google Cloud. Benchmarking against widely accessible hardware ensures reproducibility and relevance.[5] [6]
- **Academic Convention:** Retrospective design/performance papers typically compare to contemporaries deployed at scale—TPU v4's natural peer is A100, not H100, which belongs to the next generation.[6]
- **Successor Comparison:** Any direct comparison to H100 would be more appropriate for Google's next TPU generation, which also uses more advanced technology nodes and is designed to match or exceed the latest NVIDIA hardware.

Summary: TPU v4's paper compared against A100 because both are same-era chips widely available at cloud scale when TPU v4 was introduced. Comparing to H100 awaits future TPUs on similar fabs and deployment timelines.[5] [6]

⁂

what do they think about peak FLOPS performance?

Google’s TPU v4 team is **critical of using peak FLOPS/second as a performance proxy**. They emphasize:

- **Peak FLOPS/second is not a reliable predictor of real ML workload performance.**
    - *Example:* TPU v4 is 4.3×–4.5× faster than Graphcore IPU Bow and 1.15×–1.67× faster than Nvidia A100 for MLPerf benchmarks—even though the A100 has a slightly higher peak FLOPS/second.
- **Roofline Model Importance:**
    - True performance depends on **operational intensity** (ratio of FLOPS to memory operations) and **memory bandwidth**, not just theoretical compute peak.
- **System Balance Matters:**
    - Compute, memory, and interconnect must be balanced for efficient ML training. Bottlenecks in memory or data communication often prevent systems from reaching their peak FLOPS.

In short:
TPU v4’s designers argue that **real-world throughput is what matters most**. System architecture, memory bandwidth, and ML workload patterns determine actual performance far more than peak theoretical FLOPS alone. This is supported by their benchmarking and roofline modeling.

⁂

1. https://arxiv.org/pdf/2304.01433.pdf
2. https://arxiv.org/pdf/2304.01433.pdf
3. https://arxiv.org/pdf/2304.01433.pdf
4. https://arxiv.org/pdf/2304.01433.pdf
5. https://www.hpcwire.com/2023/04/06/google-claims-its-tpu-v4-outperforms-nvidia-a100/
6. https://news.ycombinator.com/item?id=35460219
7. https://www.cloudexpat.com/blog/comparison-aws-trainium-google-tpu-v5e-azure-nd-h100-nvidia/
8. https://www.nextplatform.com/2022/10/11/deep-dive-on-googles-exascale-tpuv4-ai-systems/
9. https://www.gigenet.com/blog/gpu-vs-tpu/
10. https://www.artech-digital.com/blog/energy-efficient-gpu-vs-tpu-allocation
11. https://en.wikipedia.org/wiki/Tensor_Processing_Unit
12. https://engineering.miko.ai/navigating-the-ai-compute-maze-a-deep-dive-into-google-tpus-nvidia-gpus-and-llm-benchmarking-5332339e4c9b
13. https://www.reddit.com/r/singularity/comments/1ac0ax9/googles_latest_ai_chip_is_up_to_28_times_faster/
14. https://www.usenix.org/system/files/nsdi24-zu.pdf
15. https://www.chaotropy.com/why-googles-tpus-are-no-match-for-nvidias-gpus/

16. https://arxiv.org/abs/2304.01433
17. https://cloud.google.com/blog/products/compute/performance-per-dollar-of-gpus-and-tpus-for-ai-inference
18. https://dl.acm.org/doi/10.1145/3579371.3589350
19. https://arxiv.org/pdf/2304.01433.pdf
20. https://www.reddit.com/r/MachineLearning/comments/zil35t/d_does_google_tpu_v4_compete_with_gpus_in/
21. https://www.cloudoptimo.com/blog/tpu-vs-gpu-what-is-the-difference-in-2025/
22. https://www.datacamp.com/blog/tpu-vs-gpu-ai
23. https://www.techmonitor.ai/cloud/google-ai-supercomputer-nvidia-h100/
24. https://massedcompute.com/faq-answers/?question=How+do+the+power+consumption+and+performance+of+NVIDIA's+H100+GPU+compare+to+Google's+TPU+v4%3F
25. https://massedcompute.com/faq-answers/?question=What+are+the+advantages+of+using+Google+TPU+v4+over+NVIDIA+A100+in+terms+of+cost+and+performance%3F
26. https://arxiv.org/pdf/2304.01433.pdf