PARALLEL THINKING

How a 1983 Boston startup named Thinking Machines invented the architecture that led to modern day AI data centers

Lew Tucker, Ph.D.

ZKM CM Symposium, March 2026

The Connection Machine: A Radical Bet

65,536

Tiny processors, all working at once

Each simpler than a pocket calculator — power came from sheer numbers

2,900×

Faster than the best computers of the era

For the right problems — ones where you do the same thing to a lot of data

$5M

Price tag — bought by national labs and Wall Street

NSA, Sandia, Morgan Stanley, and MIT were early customers

> *"Instead of making a single computer go faster, why not build thousands of simple ones and have them all work on the same problem at the same time?"*
>
> — Danny Hillis, founder of Thinking Machines, age 26

CM-2: Second Generation (1987)

→ Retained the 65,536 processor count but added floating-point hardware — every 32 processors shared a Weitek FPU chip.

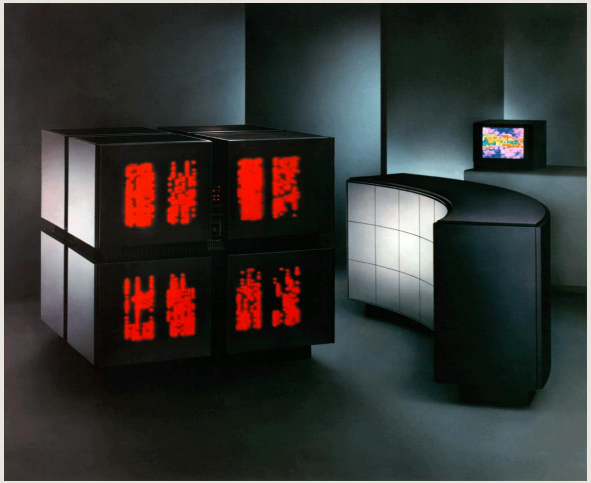
→ Dramatically improved scientific computing: physics simulations, fluid dynamics, and climate modeling became practical workloads.

→ Introduced the Data Vault — a parallel disk array streaming data at up to 1 GB/s, unprecedented for the era. First RAID system.

→ CM Fortran enabled scientific programmers to migrate existing codebases without learning an entirely new paradigm.

CM-1 vs CM-2

Feature	CM-1	CM-2
Processors	65,536	**65,536**
Float HW	None	**Weitek FPU**
Year	1985	**1987**
Languages	*Lisp	***Lisp, C*, CMF**
Disk I/O	Limited	**DataVault 1GB/s**



Four Ways to Speed Up Computation

Serial Processing

Make CPU faster through exotic materials

Traditional Supercomputers used exotic materials and vector processing to speed up calculation.

Supercomputing

SIMD

Single Instruction, Multiple Data

Process multiple streams or arrays of data at the same time in parallel.

CM-1 & CM-2

MIMD

Multiple Instructions, Multiple Data

Independent job processing by multiple CPUs, tightly connected through messaging.

CM-5, SIMD/MIMD GPUs

Cluster Computing

Racks of servers

Thousands of servers with high speed networks in large data centers.

AI Datacenters

CM-5: Learning to Be More Flexible

	CM-1 & CM-2 (1985–87)	CM-5 (1993)
Architecture	65,536 tiny 1-bit processors — all marching in lockstep	**Thousands of full SPARC processors — each fully independent**
Mode	Pure SIMD — one command, everyone obeys simultaneously	**MIMD — each processor can run completely different code**
Flexibility	Low — struggles with branching logic	**High — handles complex, varied workloads**
Still ran	CM Lisp, C* (custom languages)	**All the same data-parallel programs from CM-1/2**
Analogy	An orchestra — one conductor, everyone plays the same beat	**A jazz ensemble — each musician improvises, but in harmony**

Why This Matters

The CM-5 proved you could build a machine that was both flexible and massively parallel — as long as the programming model hid the complexity from the programmer.

The Lasting Blueprint

Modern AI chips do exactly this: independent processor clusters (MIMD) across a chip, each running in lockstep (SIMD) on its data. CM-5 drew this blueprint in 1993.

Programming Languages and Libraries Hide Complexity

CM Lisp & *Lisp

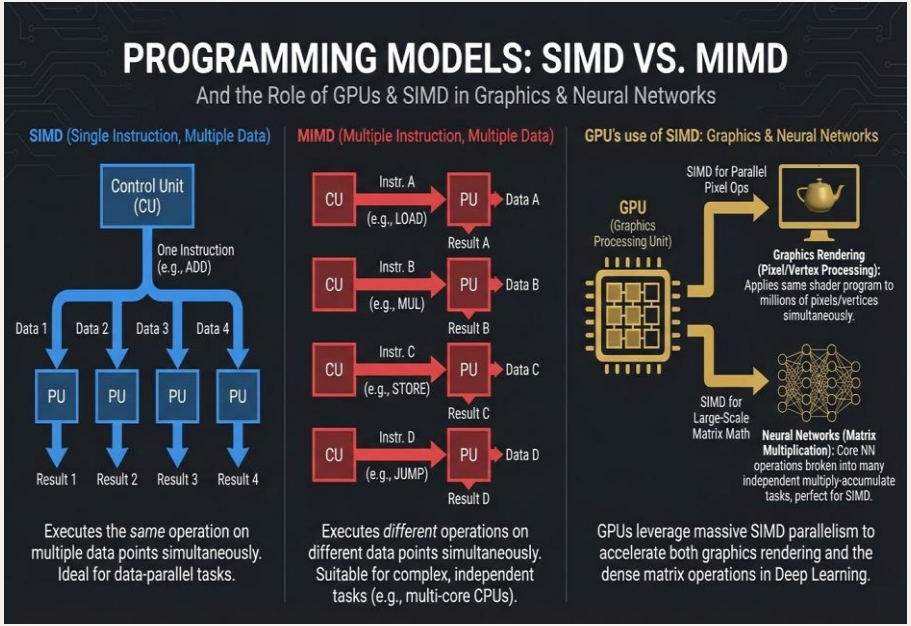
The first languages to express parallelism naturally — write an operation on a data structure and the runtime automatically distributes it across all 65,536 processors. The programmer never specifies how.

C* & CM Fortran

C* extended C; CM Fortran extended Fortran — scientists kept their familiar syntax while automatically running on thousands of processors.

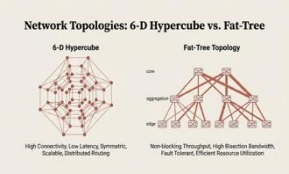
MPI & CMMD

CMMD (Connection Machine Message Passing) was TMC's inter-processor communication library — a direct architectural precursor to MPI, the standard still running on every supercomputer in the world today.



The CM's key insight: hide the complexity of massive parallelism behind a simple model — exactly what GPU computing rediscovered 20 years later.

Algorithms Invented on the Connection Machine



The CM's hypercube network alongside the fat-tree topology that replaced it — both invented to move data between thousands of processors simultaneously

Parallel Reduction

Then: Add up values across all 65,536 processors in just 16 steps — by pairing neighbors, then pairs of pairs, up a tree. Not 65,536 steps. Sixteen.

Now: MapReduce, used at Google over thousands of servers for search and GPUs for AI training.

Prefix Scan (Hillis-Steele, 1986)

Then: A seemingly impossible task — running totals across millions of entries in parallel. Hillis & Steele proved it could be done in a fraction of the expected time.

Now: Used in sorting, memory allocation, compressing data, and building search indexes. Runs billions of times per second across the internet.

Gather & Scatter

Then: The CM's wiring let any processor send data directly to any other — like a telephone exchange built into the hardware.

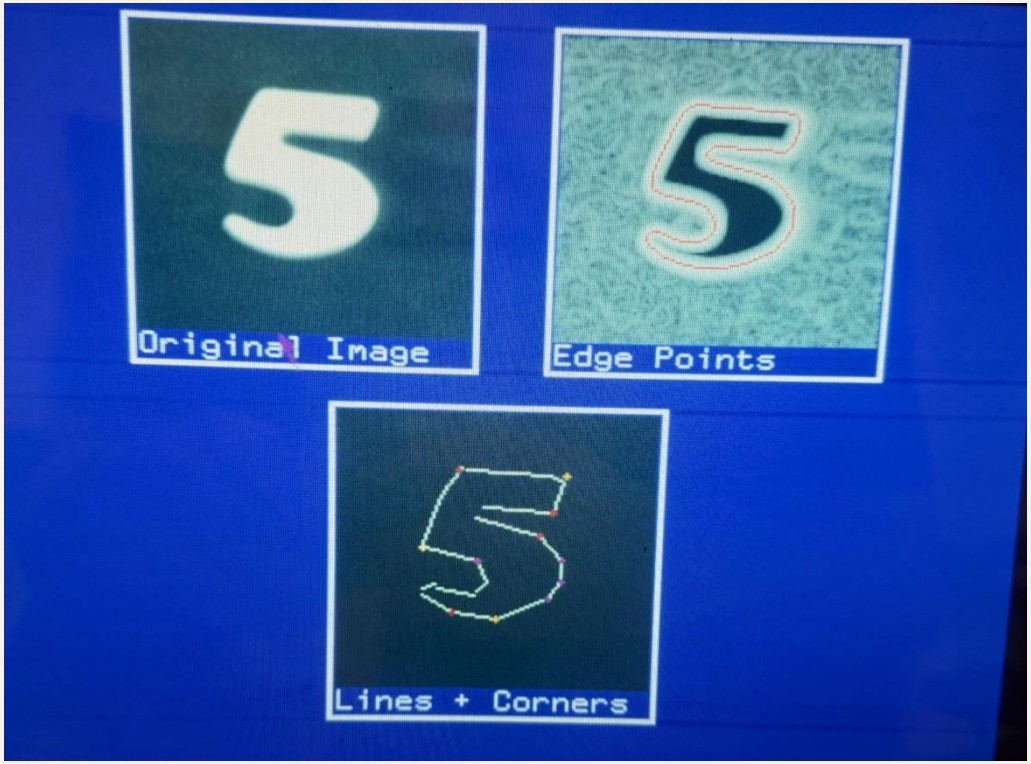
Now: The single biggest performance consideration when programming GPUs. How you access memory still dominates speed, exactly as CM engineers found in 1987.

Segmented Operations

Then: Apply different operations to different sections of data simultaneously — like having 65,536 processors sort 100 different lists all at the same time.

Now: The backbone of how GPUs process neural network layers and the "attention" mechanism at the heart of AI language models like GPT.

Computer Vision on the CM-2



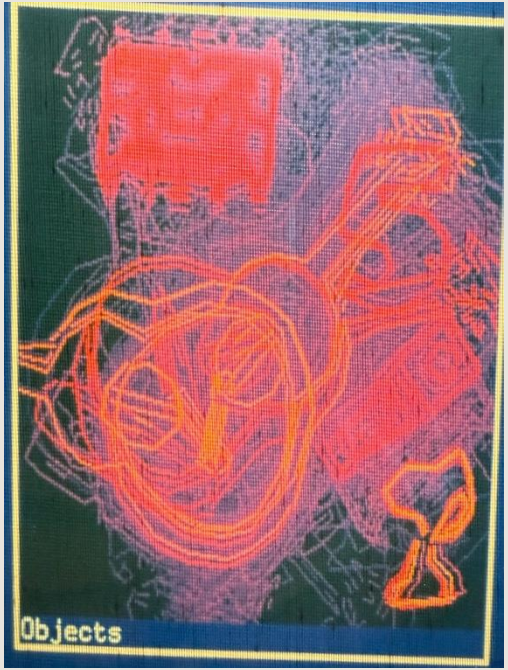
TRAINING

Processing all pixels in parallel, the Connection machine uses convolution to find edges and turns edges into line segments. These lines form the features used to identify the object in a scene.

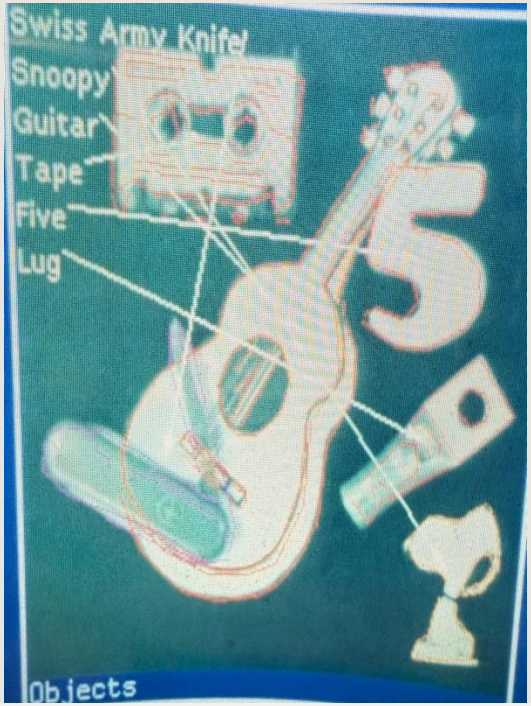
Computer Vision on the CM-2

NEW SCENE

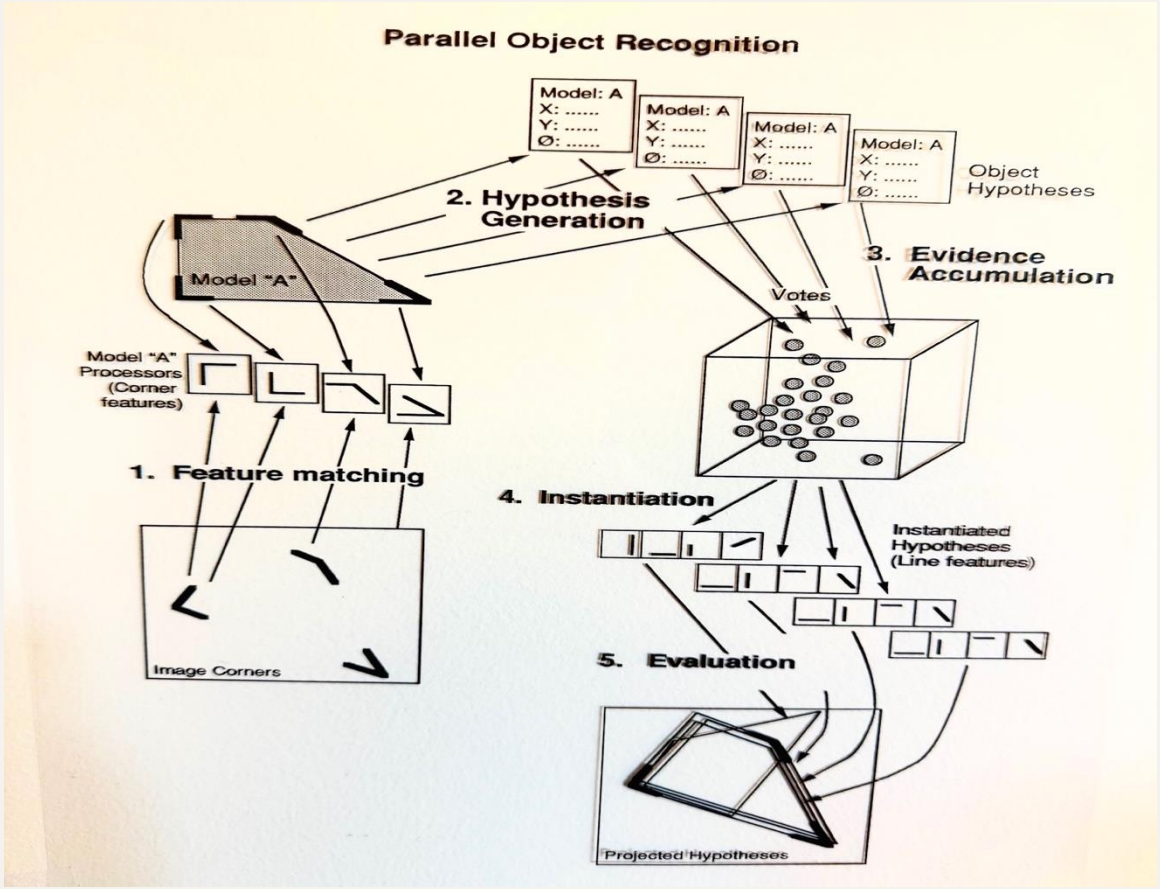
FEATURE MATCHING



OBJECTS FOUND



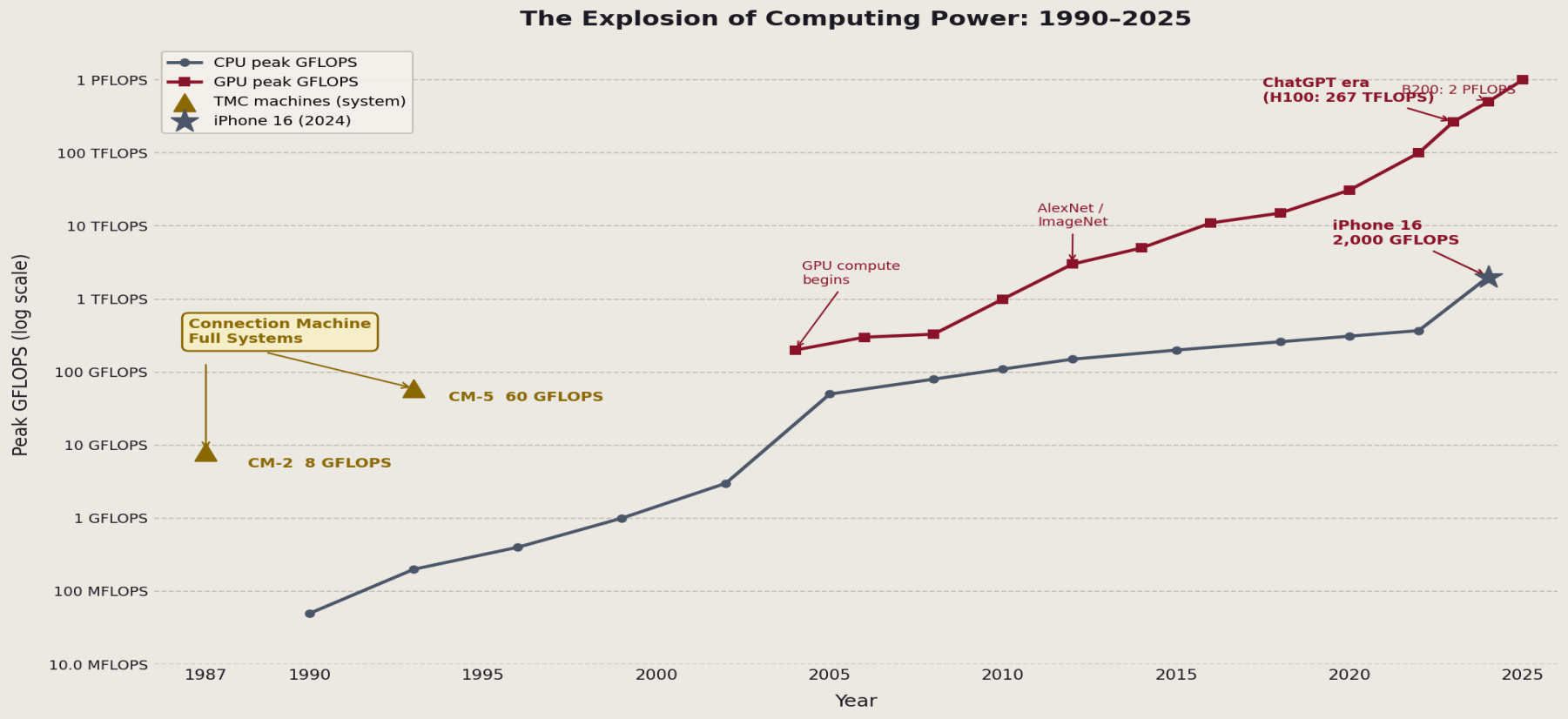
Thinking in Parallel



"Evidence is collected and hypotheses generated all in parallel."

Sets of lines found in the image generate millions of hypotheses which together accumulate evidence for the presence of different objects in the scene

The Explosion of Computing Power



CPU performance roughly doubled every 2 years (Moore's Law). GPUs, designed for parallel graphics, turned out to be ideal for AI — and accelerated far faster.

HyperScale Data Centers

→ The internet drove computing to a scale no single company had imagined — billions of users, petabytes of data.

→ The response: entire data centers, measured in Megawatts of power consumed, built to run a single application — google search, video, social media.

→ A hyperscale data center is, in concept, what TMC imagined: a single massively parallel machine

Today's AI made possible by 3 things

→ Hyperscale Computing Power
→ Massive neural networks with billions of parameters
→ Huge training sets by crawling the internet

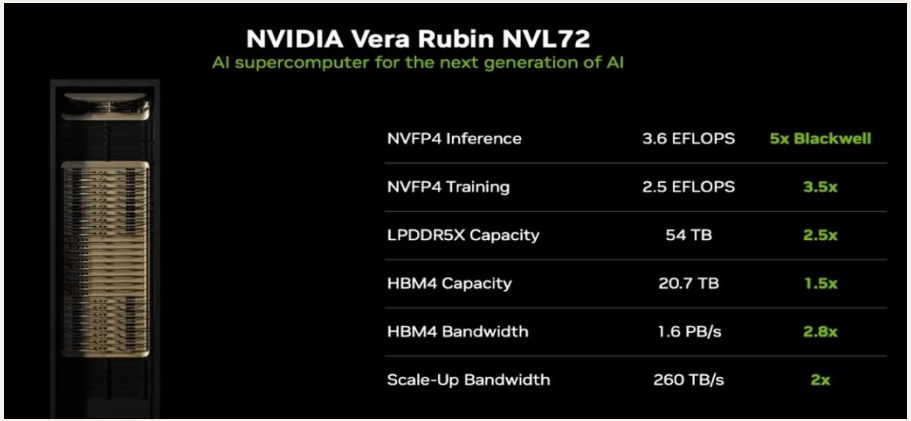
Aerial view of a hyperscale data center campus

GPUs: A Connection Machine on a Chip

"The Connection Machine's key insight — that you can hide the complexity of massive parallelism behind a data-parallel programming model — is exactly what modern GPU programming rediscovered 20 years later."

A modern Nvidia GPU contains up to 16,896 processing cores — all working on the same problem simultaneously.

Originally designed as SIMD array processors for gaming consoles. Lately used in AI for ultra-fast matrix multiplication.



Then and Now

	Connection Machine (1985)	Nvidia H100 GPU (2023)
Processors	65,536 (1-bit each)	**16,896 (full 32-bit)**
Programming	Write for arrays; runs on all	**Write for one; GPU runs on all**
Best for	Weather, genomics, finance	**AI training, graphics, science**
Power	~200 kW (fills a room)	**700 W (shoebox)**
Cost	$5 million	**~$30,000**

Today's AI Datacenter Computers are at Massive Scale

CM's 1985 vision: thousands of processors + fast connections = one powerful parallel computer. That vision now runs at planetary scale.

GPU-based Nodes

A single AI server holds 8 H100/B200 GPUs — up to 1.5 TB of memory and ~72 petaflops. GPUs connect via NVSwitch at 900 GB/s. Nvidia's Rubin nodes (2026) projected to reach ~400 petaflops per node.

8 GPUs · 1.5 TB · 72 PF

per node today → ~400 PF with Rubin (2026)

High Speed Network Interconnect

Connects thousands of GPU servers together at 400 Gb/s. GPT-4 was trained across more than 25,000 GPUs connected this way — the CM's inter-rack network, scaled to a warehouse.

25,000+

GPUs in a single AI training run

Nvidia Collective Operations

Software that lets thousands of GPUs combine results — AllReduce, Broadcast, Scatter, AllGather. These are the Connection Machine's collective operations, reimplemented at global scale.

AllReduce

= CM's parallel reduction, at cluster scale

Summary

What Thinking Machines taught us

The Bet Was Right

Massive parallelism — thousands of simple processors working together — beats a few fast ones for the most important computing tasks. Modern AI training proves this every day.

The Programming Model Endures

Hide complexity from the programmer — write for one, run on many. CM Lisp (1986) → CUDA (2007) → PyTorch (2016). The same idea, rediscovered every decade.

Algorithms Still Rule

Parallel reduction, prefix scan, gather/scatter — invented on the CM between 1985 and 1987. They are the building blocks of every GPU kernel and every AI training run today.

The Vision Scales

Today's AI clusters — tens of thousands of GPU nodes, connected at 400 Gb/s, running NCCL collectives — are the Connection Machine at data warehouse scale. TMC was just 20 years early.

Thinking Machines Corporation closed in 1994. It may have been ahead of its time, but its impact is still felt today.

Think Different, But Dress Alike

—Thanks