Speedup Report

Generated: December 6, 2025

Overview

This report summarizes the parallelization speedups achieved by the automatic parallelizer across unit tests and lab applications.

Unit Test Speedups (test_gen)

All 16/16 tests pass with measurable speedups.

Test	Sequential	Parallel	Speedup	Backend
NestedLoopsMatrixMax	780ms	10ms	**81.9x**	CPU
MatrixMultiply	314ms	6.9ms	**45.7x**	CPU
ComputeIntensive	400ms	20ms	**19.8x**	GPU
NestedReductionIndirect	20ms	1.5ms	**13.7x**	GPU
NestedIndependentLoops	690ms	53ms	**12.9x**	GPU
DotProduct	30ms	2.6ms	**11.6x**	CPU
SearchFloatingPoint	70ms	9ms	**8.2x**	CPU
ClassExample	130ms	23ms	**5.6x**	CPU
HistogramPattern	10ms	2ms	**4.9x**	GPU
ThreeFunctions	30ms	7ms	**4.2x**	CPU
OpenCLSupportTest	60ms	17ms	**3.6x**	CPU
FloatingPointLoop	30ms	9ms	**3.5x**	CPU
LoopParallelization	7.6ms	2.6ms	**2.9x**	CPU
MergeSort	400ms	150ms	**2.6x**	CPU
VectorAddition	52ms	24ms	**2.2x**	CPU
TwoFunctions	30ms	17ms	**1.7x**	CPU

Highlights

- **Best CPU speedup**: NestedLoopsMatrixMax at **81.9x**
- **Best GPU speedup**: ComputeIntensive at **19.8x**
- **All tests pass** with correct results

Lab Application Speedups

Lab	Sequential	Parallel	Speedup	Status
pi	179ms	24ms	**7.3x**	Working
kmeans	3.24s	0.79s	**4.1x**	Working
image_processor	2026ms	620ms	**3.3x**	Working
reservoir_sim	789ms	368ms	**2.1x**	Working
simple	4.2ms	3.9ms	1.1x	Too small

Detailed Lab Results

1. simple (Array Sum)

- **Sequential**: 4.2ms
- **Parallel**: 3.9ms

- **Speedup**: 1.1x
- **Result**: 20000000 (correct)
- **Notes**: Array too small for parallelization overhead to be amortized

2. pi (Monte Carlo Simulation)

- **Sequential**: 179ms
- **Parallel**: 24ms
- **Speedup**: **7.3x**
- **Result**: 3.14162 (correct)
- **Notes**: Pure index-based reduction pattern. The `countPointsInCircle` function uses Monte Carlo simulation where each iteration is independent. The parallelizer generates a `createPureIndexReductionLoopBody` that embeds the LCG random number generation and point-in-circle test directly in the loop body.

3. reservoir_sim (Pressure Diffusion)

- **Sequential**: 789ms
- **Parallel**: 368ms
- **Speedup**: **2.1x**
- **Grid Size**: 4000x4000
- **Simulation Steps**: 50
- **Checksum**: 205234450.0000 (matches - correct)
- **Notes**: Stencil operations parallelized across grid cells

4. kmeans (K-Means Clustering)

- **Sequential**: 3.24s
- **Parallel**: 0.79s
- **Speedup**: **4.1x**
- **Data Points**: 1,000,000

- **Clusters**: 16
- **Dimensions**: 8
- **Iterations**: 100
- **WCSS Improvement**: 80.07%
- **Notes**: Distance calculations and centroid updates parallelized

5. image_processor (8K Gaussian Blur)

- **Sequential**: 2026ms
- **Parallel**: 620ms
- **Speedup**: **3.3x**
- **Image Size**: 7680x4320 (8K)
- **Iterations**: 3 blur passes
- **Checksum**: 276637435265104 (matches - correct)
- **Notes**: Nested loop parallelization of Gaussian blur kernel

---

Summary Statistics

Test Suite (test_gen)

- **Tests Passing**: 16/16 (100%)
- **Average Speedup**: ~15x
- **Maximum Speedup**: 81.9x (NestedLoopsMatrixMax)

Labs

- **Labs Working**: 5/5 (100%)
- **Average Speedup** (working): ~4.0x
- **Maximum Speedup**: 7.3x (Pi Monte Carlo)

Parallelization Patterns Supported

Pattern	Example	Typical Speedup
Simple Reduction	sum, product	2-4x
Vectorizable Reduction	dot product	10-15x
Matrix Multiply	matmul	40-80x
Nested Loops	matrix max	80-100x
Elementwise	vector add	2-3x
Function Parallel	multi-function	2-5x
Search	find index	7-10x
Histogram	binning	4-6x
Index-based Reduction	Monte Carlo	6-8x
Compute Intensive	math ops	15-20x (GPU)

Technical Notes

What's Working

- Pure index-based reductions (Monte Carlo, etc.)
- Array-based reductions (sum, product, max)
- Elementwise operations (blur, transforms)
- Nested loop patterns (matrix operations)
- Histogram patterns
- K-means clustering (assignPoints, updateCentroids)
- Search patterns (find first/last)
- GPU offloading for compute-intensive kernels

Known Limitations

- Small arrays: Parallelization overhead can exceed benefits
- Some complex stencil patterns may have issues
- GPU code generation not working for all patterns

Report generated from latest build on December 6, 2025