

# ComS 402c

Team topic: Generator of Parallelism Pattern for code analysis

# Team Members

Nika Babtsov,

Tyler Cahill,

John Lawless,

Kieran McCormick

Jake Strauch,

Le Chen

# Target of this project

## Project Title \*

Generator of Parallelism Pattern for code analysis

**Describe WHAT is the project about? (1) Who are the users? (2) What are the main features? (3) What are the important use-cases? \***

1. Anyone who wants to work on code analysis with parallelism patterns.
2. It would be able to generate a dataset with labels for the application of parallelism application.
3. It would be helpful for the problem of inadequacy dataset in this area. With the development of Machine Learning, more data mean more possible to find the rules and patterns.

# What? Why?

What is parallel processing?

two or more calculations/operations happen simultaneously

practical motivation: enhanced performance

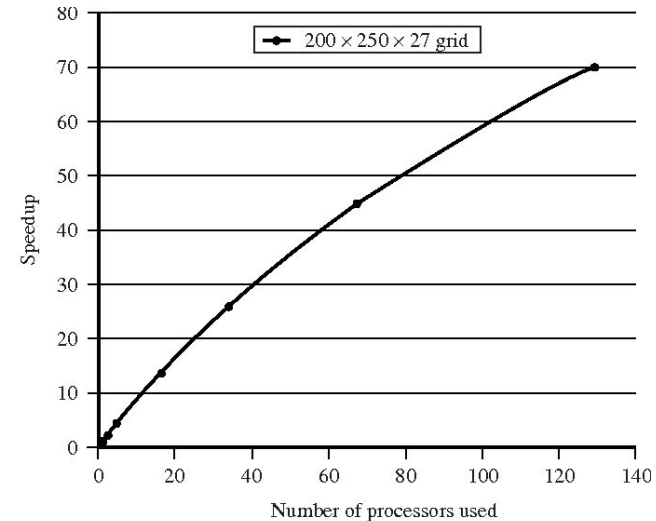


Figure 1.1

Performance of the MM5 weather code.

# How?

## Parallel Architectures:

SISD

SIMD

MISD

MIMD

		Instruction Streams	
		one	many
Data Streams	one	<b>SISD</b> traditional von Neumann single CPU computer	<b>MISD</b> May be pipelined Computers
	many	<b>SIMD</b> Vector processors fine grained data Parallel computers	<b>MIMD</b> Multi computers Multiprocessors

# How?

The goal of a programmer in a modern computing environment is not just to take advantage of processors with two or four cores. Instead, it must be to write **scalable** applications that can take advantage of any amount of parallel hardware

when you run sequential program:

- Instructions executed on 1 core
- Other cores are idle

# How?

From Sequential Code to Parallel Code: we take advantage of OpenMP

- Defacto standard API for writing shared memory parallel applications in C, C++, and Fortran
- easy to apply
- incremental apply

# Plan for this semester

13 weeks:

1-2: get familiar with parallel programming; openmp; parallelization patterns

3-4: check benchmarks; build and run both parallel & sequential version

5-6: check seed of different patterns; build programs for basic parallelization pattern programs (mid-term)

7-10: start to code generator

11-12: evaluation

# 9/7 meeting

1. background Q&A
2. resources needed for this project
  - a. repo: gitlab(Nika)
  - b. VM: checked
  - c. server:
3. some tools:
  - a. get the vm.
  - b. CMake, gcc-6,
  - c. llvm8.0.1(<https://llvm.org/docs/GettingStarted.html>) download (<https://releases.llvm.org/download.html#8.0.1>)
    - i. [LLVM source code \(.sig\)](#)
    - ii. [Clang source code \(.sig\)](#)
    - iii. [compiler-rt source code \(.sig\)](#)
  - d. discopop: <https://github.com/discopop-project/discopop>

# Q&A

1. The book
  - a. chapter 1~2
  - b. 3.1-3.3, 3.5.2, 3.8.3
  - c. 4.1, 5.1,
  - d. chapter 7
  - e. 8.1

# TODOs

Le: Slides, Poll

COMS TEAM: VM, repo, try to install LLVM and DiscoPoP

checked: gcc6; cmake;

llvm: todo

# Plan

1. applications and benchmarks => dataset that comes from well-recognized resources
  - a. run benchmarks: Polybench; BOTS; NAS Bench;
  - b. setup a dataset folder
  - c. run applications: LULESH(cmake)
2. dataset generation:
  - a. AST to code: <https://github.com/inducer/cgen>
  - b. metaprogramming/template programming
  - c. <https://lidavidm.github.io/sympy/tutorial/index.html>
    - i. get a equation-> generate code from the equation
    - ii.

## 2.1 dataset covers different patterns