19 Fall Parallel Functional Programming Project Report

Parallel Sudoku Solver in Haskell

Jiaheng He(jh4003)

Project Background

Sudoku is a logic based, number placement puzzle. The objective of this game is to fill a 9*9 grid with digits that satisfy some rules, like no duplicate elements in a row/column/box.

5	3			7				
6			1	9	5			
	9	8					6	
8				6				3
4			8		3			1
7				2				6
	6					2	8	
			4	1	9			5
				8			7	9

5	3	4	6	7	8	9	1	2
6	7	2	1	9	5	3	4	8
1	9	8	3	4	2	5	6	7
8	5	9	7	6	1	4	2	3
4	2	6	8	5	3	7	9	1
7	1	3	9	2	4	8	5	6
9	6	1	5	3	7	2	8	4
2	8	7	4	1	9	6	3	5
3	4	5	2	8	6	1	7	9

It's easy to understand as it looks. Sudoku can be modeled as a constraint satisfaction problem, and also can be solved with naive backtracking, even possible with heuristic search if you like. There are many ways to solve this problem, but their efficiency may vary a lot.

Project Goal

I chose to take this course because I didn't have any functional programming experience. And I did some multi-thread programming and performance related projects. As far as I know, some object oriented programming language like Golang, C++, Python has some functional programming paradigm embedded in. And sometimes it's hard for me to follow a fixed programming paradigm. Because some code looks like functional and some looks like object oriented, and they are in the same project. And some people say that it's easy to use FP to solve parallel computing problems.

So I'd like to take this chance to learn:

1. How to write parallel and concurrency code(example question: I've read some posts about add -threaded when compile may make the program faster, but there is any code start a new thread)
2. Pros and Cons of FP(efficiency, maybe easy to code for parallel and concurrency?)
3. When to use FP? What kind of projects is more suitable for FP?

Instead of choosing a hard problem to solve, I'd like to take this simple problem and try to find out the answers of these three problems. So that I can focus more on the language rather than the problem.

Project Delivery

1. Code of solving a Sudoku in parallel, and comparing with a sequential naive solver. And see the bottleneck of when providing more resources(# of CPU) to the solver.
2. Code of solving multiple Sudokus with the help of parallel and concurrency. And compare these two solutions.
3. Analyze pros and cons of parallel/concurrency programming in Haskell and compare it with C++.