2.3 Lecture Summary

2 Functional Parallelism

2.3 Memoization

Lecture Summary: In this lecture, we learned the basic idea of "memoization", which is to remember results of function calls $f(x)$ as follows:

1. Create a data structure that stores the set $\{(x_1, y_1 = f(x_1)), (x_2, y_2 = f(x_2)), \ldots\}$ for each call $f(x_i)$ that returns y_i.
2. Perform look ups in that data structure when processing calls of the form $f(x')$ when x' equals one of the x_i inputs for which $f(x_i)$ has already been computed.

Memoization can be especially helpful for algorithms based on dynamic programming. In the lecture, we used Pascal's triangle as an illustrative example to motivate memoization.

The memoization pattern lends itself easily to parallelization using futures by modifying the memoized data structure to store $\{(x_1, y_1 = future(f(x_1))), (x_2, y_2 = future(f(x_2))), \ldots\}$. The lookup operation can then be replaced by a *get()* operation on the future value, if a future has already been created for the result of a given input.

Optional Reading:

1. Wikipedia article on Memoization.

Mark as completed