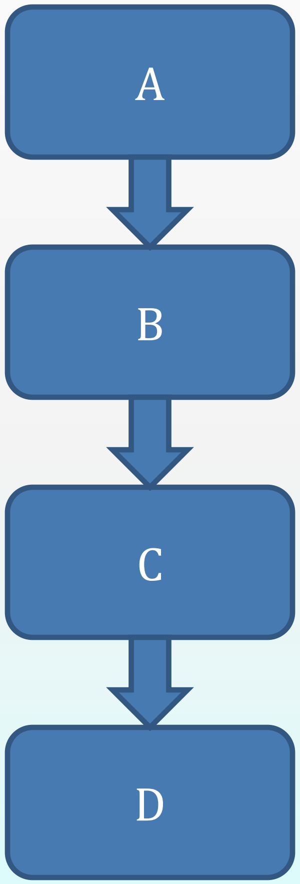
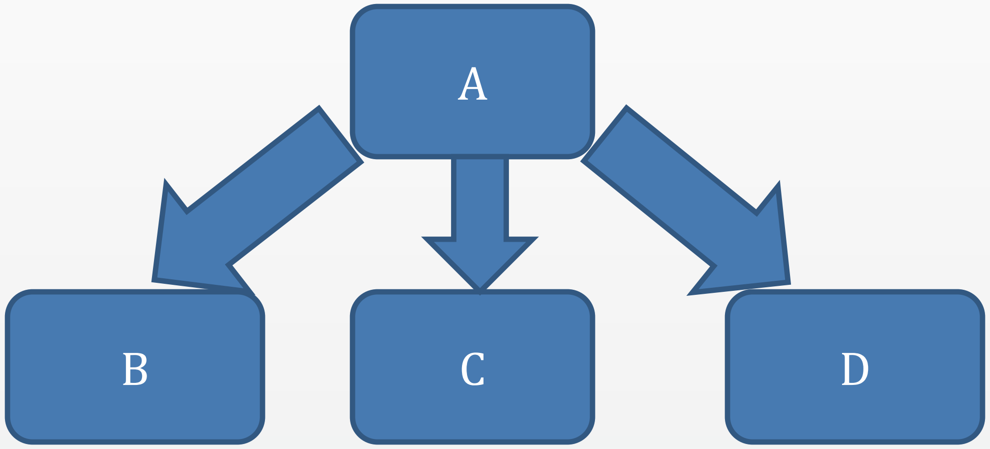
Automatic Parallelism Discovery

Hongyu Gao

Introduction





Sequential vs Parallel execution

Introduction

- Why do we need parallel execution?
  - Ever increasing computation scale
  - Limited computational power of a single core

Introduction

- A dilemma:
  - Emerging need for parallel computing
  - Difficulty of parallel programming
- A solution:
  - Automatic parallel execution of sequential program

Related work

- Swift:
  - "A system for the rapid and reliable specification, execution, and management of large-scale science and engineering workflows."
- Seems like all we need?

Related work

- Drawbacks:
  - Language limitation:
    - Single assignment
  - Scalability issue
- Proposed solution:
  - Dependency graph generation

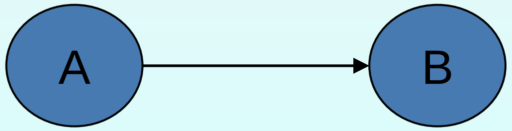
    + execution engine

Dependency graph generation

- A directed acyclic graph
- A node:

  The smallest block of code that is scheduled for parallel execution
- An edge:

  A node depends on the completion of another node before it can be executed


$$\begin{tikzcd} A \arrow[r] & B \end{tikzcd}$$

An example

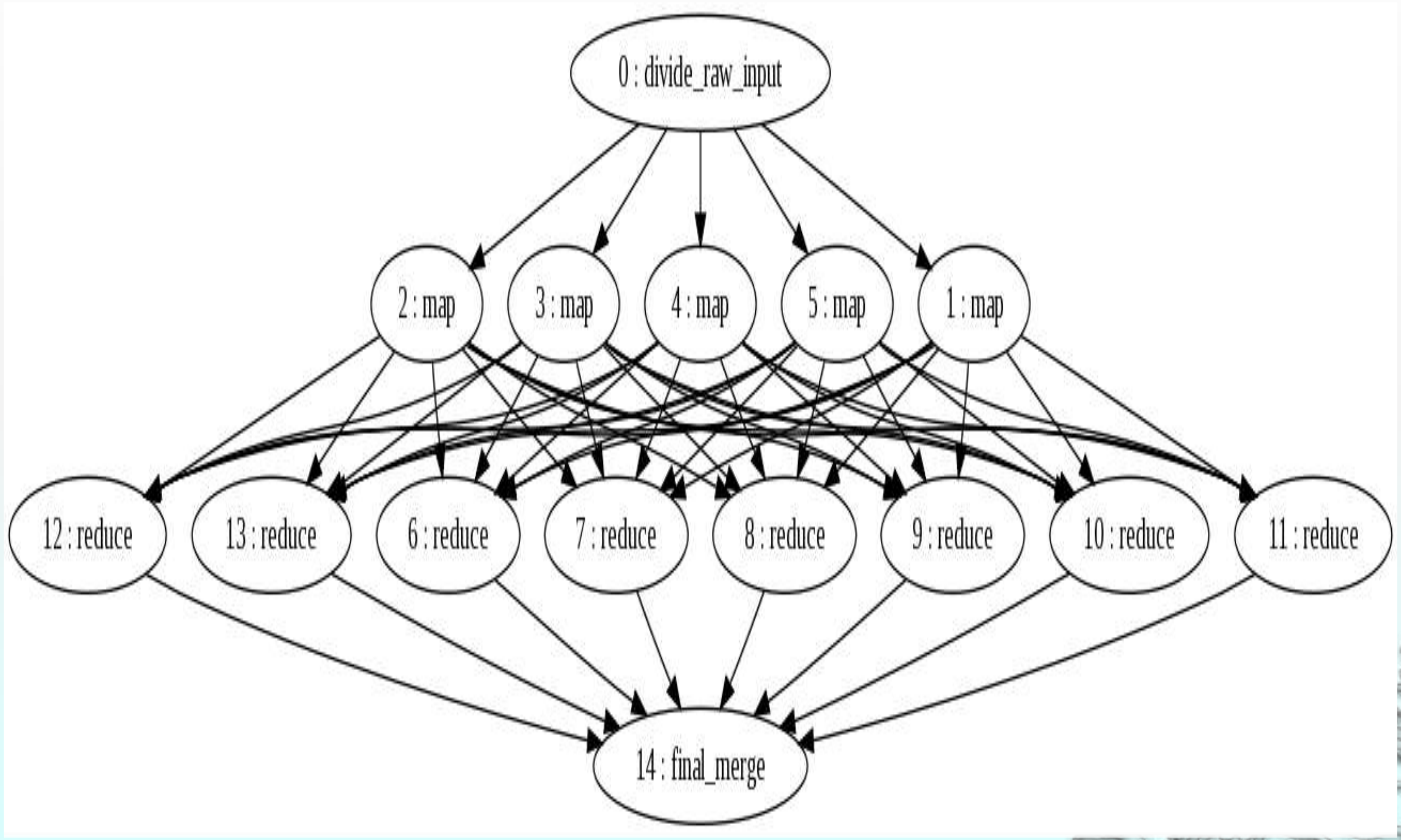
```
divide_raw_input(in_file, in_file_1, …, in_file_MAPSIZE)

for (i = 0; i<MAPSIZE ; i++):
        Map(in_file_i, intermed_file_i_1, …,
        intermed_file_i_REDUCESIZE)

for (i = 0; i<REDUCESIZE ; i++):
        Reduce(intermed_file_1_i, …, intermed_file_MAPSIZE_i,
        out_file_i)

Combine_output(out_file_1, …, out_file_REDUCESIZE, out_file)
```

An example



0 : divide_raw_input
2 : map
3 : map
4 : map
5 : map
1 : map
12 : reduce
13 : reduce
6 : reduce
7 : reduce
8 : reduce
9 : reduce
10 : reduce
11 : reduce
14 : final_merge

Task execution

- A node (task) can be executed if:
  - It has no in-edge
  - All nodes that it depends on have been completed

Task execution

- A set of nodes ready to be executed
- A dependency factor for each node
- Update the dependency factor upon the completion of every node
- Update the “ready set”
- O(E) time complexity

Further optimization

- Pipeline the graph building and the task execution
  - A window of size n on the dependency graph will be enforced while the execution is working
  - Address the scalability issue

Questions?

Thank you!