CS101-Quiz9-Review

Suting Chen

CS101-Quiz9-Review

Key Points

1. Topological Sort
2. Greedy Algorithms

Topological Sort

1. An algorithm for ordering the vertices of a directed acyclic graph (DAG) in a linear ordering.

2. Time complexity: $O(V + E)$

3. Space complexity: $O(V)$

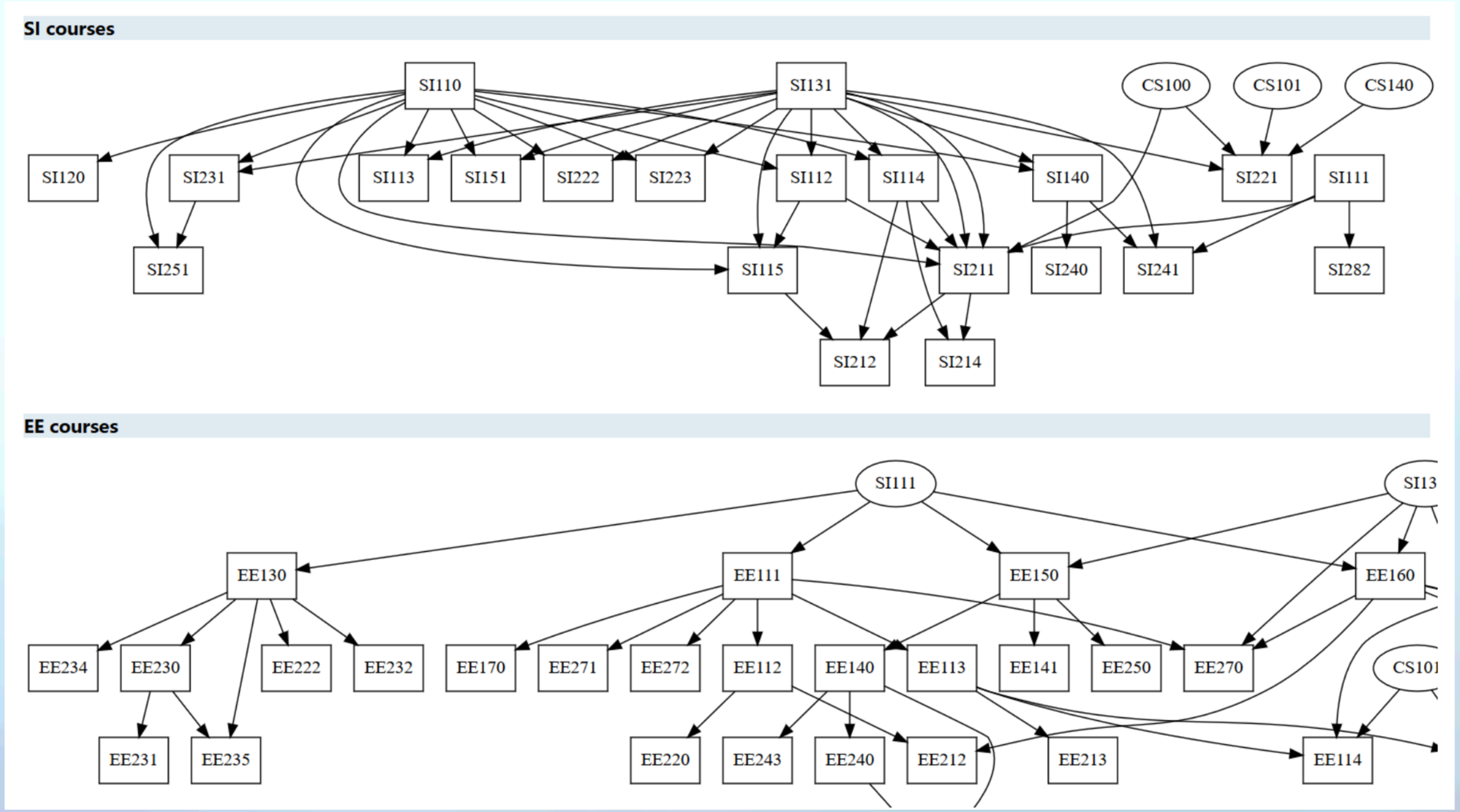
Topological Sort

Basic Theorems and Lemmas

1. A graph is a DAG if and only if it has a topological sorting
2. A DAG always has at least one vertex with in-degree zero.
3. Any sub-graph of a DAG is a DAG.

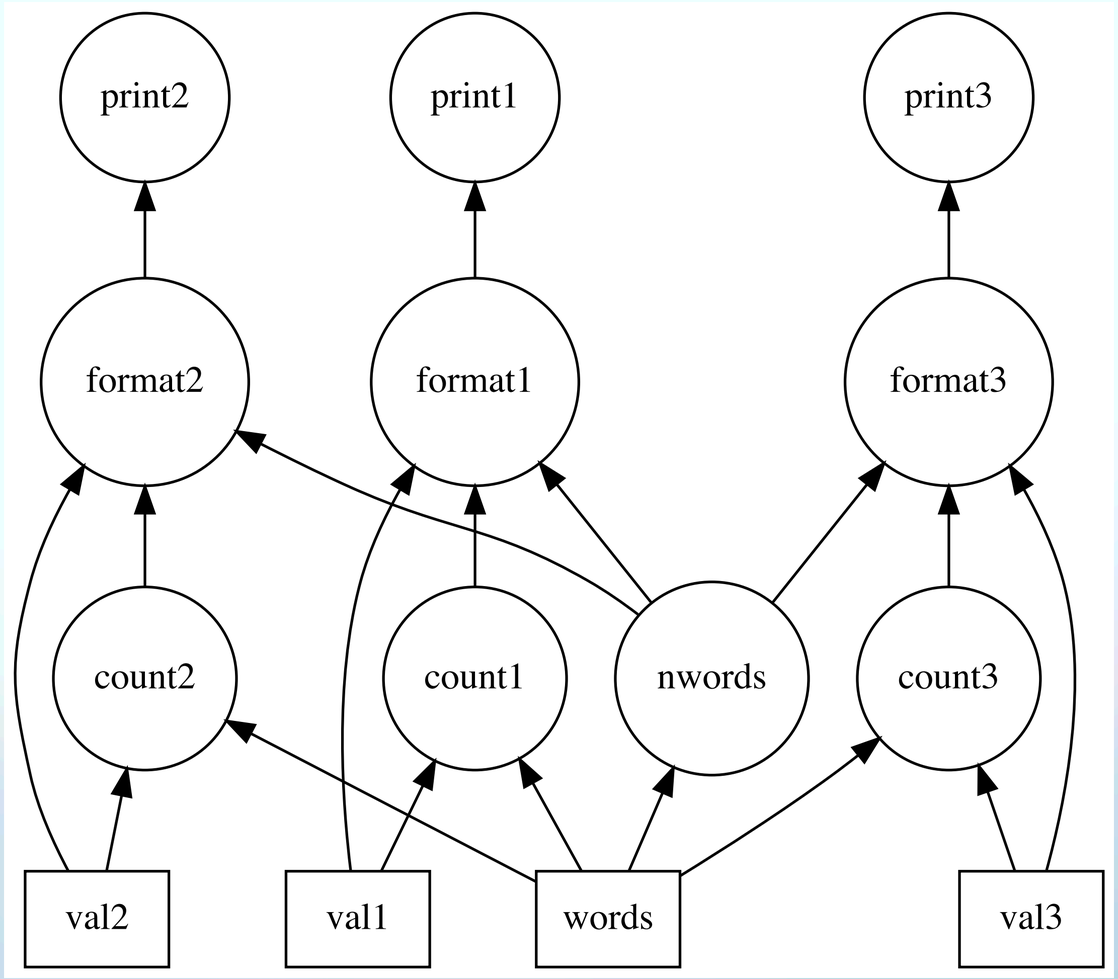
Topological Sort

Curriculum scheduling



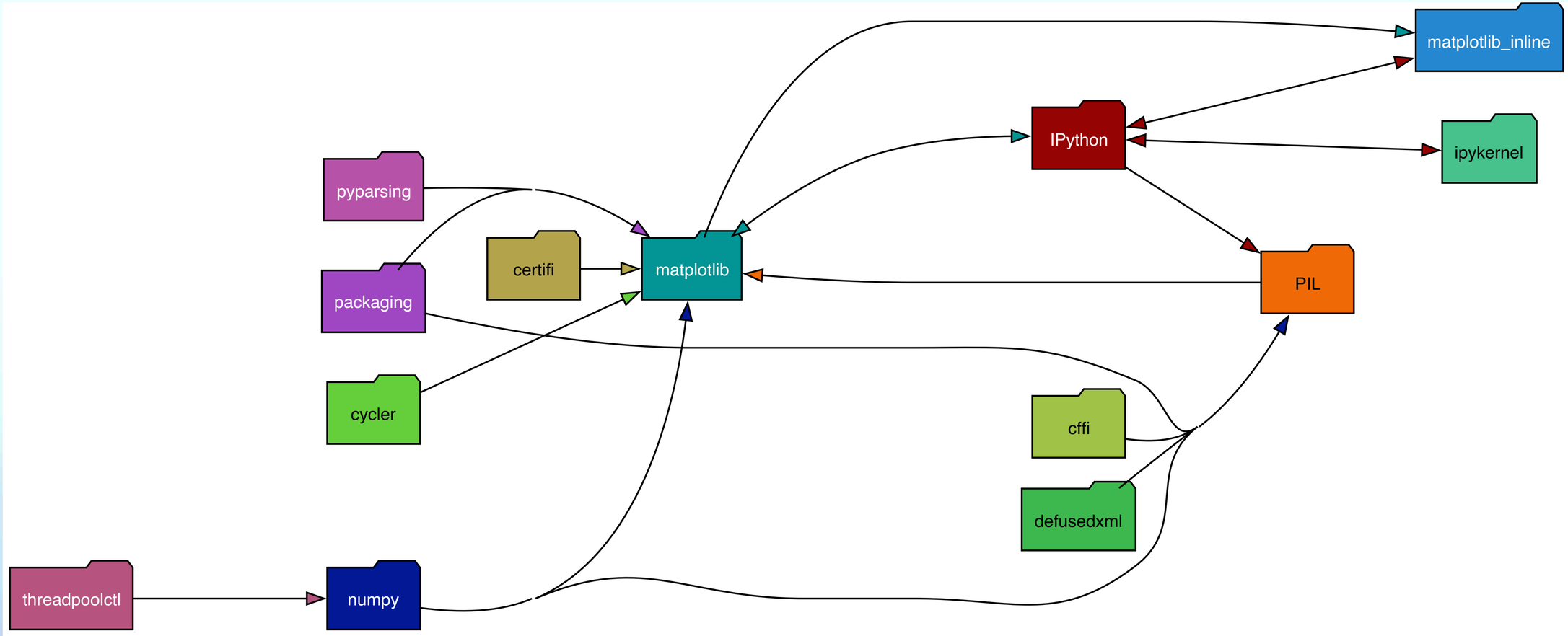
Topological Sort

Task graph — Python Dask



Topological Sort

Task graph — Dependency



CS101-Quiz9-Review

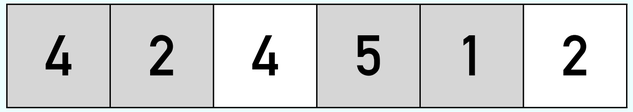
Key Points

1. Topological Sort
2. **Greedy Algorithms**

Greedy Algorithms

Array Section

```
1 def ArraySection(A):
2   s = A[1]
3   g = 1
4   for i = [2, ..., n]:
5     s = s + A[i]
6     if s > M:
7       s = A[i]
8       g = g + 1
9   return g
```



Greedy Algorithms

Array Section — exchange argument

1. When all optimal solutions have the **same size and differ only in their cost**.
2. Justify with how you will replace some parts of the optimal solution.

Greedy Algorithms

Array Section — exchange argument

1. Define subproblem and compare
2. Exchange
3. Iterate

Greedy Algorithms

Array Section — exchange argument

1. Sub-problem $g(i)$ means minimizing the number of sections for $\langle a_1, \cdots, a_i \rangle$, and then maximizing the start indices of all the sections.
2. Exchange
3. Iterate

Greedy Algorithms

Array Section — exchange argument

1. Sub-problem $g(i)$ means minimizing the number of sections for $\langle a_1, \cdots, a_i \rangle$, and then maximizing the start indices of all the sections.
2. Exchange
3. Iterate

Greedy Algorithms

Array Section — exchange argument

1. Sub-problem $g(i)$ means minimizing the number of sections for $\langle a_1, \cdots, a_i \rangle$, and then maximizing the start indices of all the sections.

2. Assume the start index of sections is $\langle s_1, \cdots, s_k \rangle$. Assume we can make s_t smaller.

   - Prove the new solution is not better than our solution (see (e.) in the answer book).

3. Iterate

Greedy Algorithms

Array Section — exchange argument

1. Sub-problem $g(i)$ means minimizing the number of sections for $\langle a_1, \cdots, a_i \rangle$, and then maximizing the start indices of all the sections.

2. Assume the start index of sections is $\langle s_1, \cdots, s_k \rangle$. Assume we can make s_t smaller.

   - Prove the new solution is not better than our solution (see (e.) in the answer book).

3. 显然成立

Greedy Algorithms

Array Section — exchange argument — Take-home message

1. The **exchange method** assumes that the "size" of the solution is the same for both greedy and optimal solutions.

2. Exchange is then used to show that it is optimal.

3. Mathematical induction is often used to show how this consistency holds as the problem size increases.