CS 105: Department Introductory Course on Discrete Structures

Instructor : S. Akshay
Guest Lecture by : R. Govind

Sep 05, 2023
Lecture 13 – Basic Mathematical Structures
Chains and Antichains

Recap: Partial order relations

Last class we saw

- Partial orders: definition and examples
- Posets, chains and anti-chains
- Graphical representation as Directed Acyclic Graphs
- Topological sorting (application to task scheduling)

Recall: Partial Orders

- A poset is a set S with a partial order $\preceq \ \subseteq S \times S$.
- A totally ordered set is a poset in which every pair of elements is comparable, i.e., $\forall a, b \in S$, either $a \preceq b$ or $b \preceq a$.

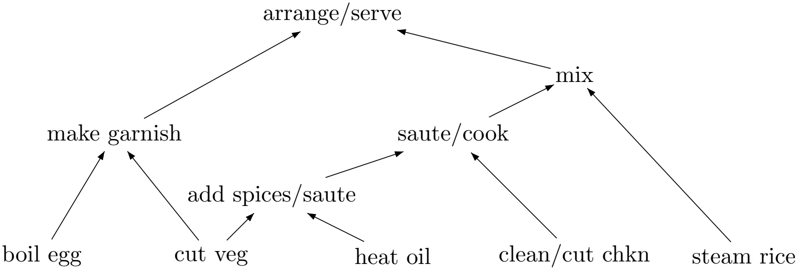
Definitions: Let $(S, \preceq)$ be a poset.

- A subset $B \subseteq S$ is called a chain if every pair of elements in B is related by $\preceq$.
- A subset $A \subseteq S$ is called an anti-chain if no two distinct elements of A are related by $\preceq$.

Examples and applications

A task scheduling example

Let us represent a recipe for making Chicken Biriyani as a poset!

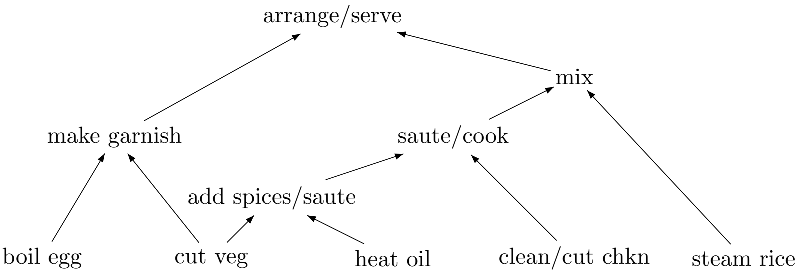


- Clearly, this shows the dependencies.

Examples and applications

A task scheduling example

Let us represent a recipe for making Chicken Biriyani as a poset!

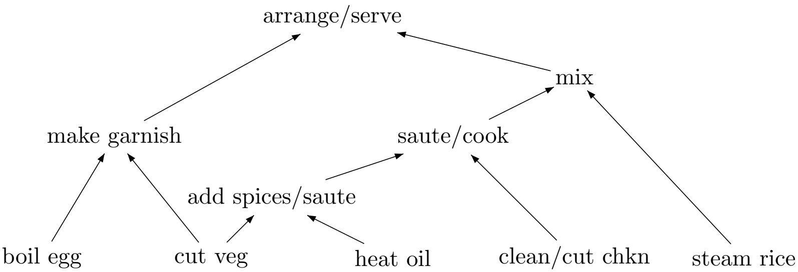


- Clearly, this shows the dependencies.
- But when you cook you need a total order, right?

Examples and applications

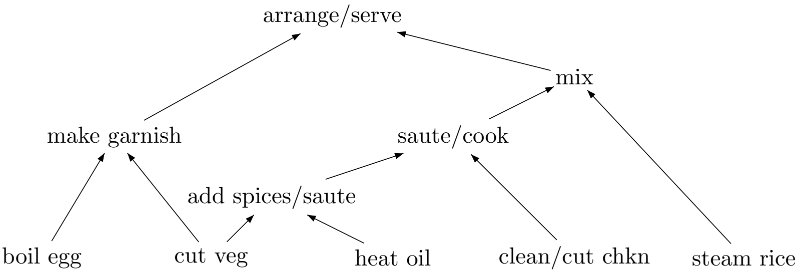
A task scheduling example

Let us represent a recipe for making Chicken Biriyani as a poset!



- Clearly, this shows the dependencies.
- But when you cook you need a total order, right?
- Further, this total order must be consistent with the po.
- This is called a linearization or a topological sorting.

Tasks scheduling as a poset



Theorem

- Every finite poset has a topological sort, i.e., a totally ordered set that is consistent with the poset (H.W).

Topological sorting

Definition

A topological sort or a linearization of a poset $(S, \preceq)$ is a poset $(S, \preceq_t)$ with a total order $\preceq_t$ such that $x \preceq y$ implies $x \preceq_t y$.

Topological sorting

Definition

A topological sort or a linearization of a poset $(S, \preceq)$ is a poset $(S, \preceq_t)$ with a total order $\preceq_t$ such that $x \preceq y$ implies $x \preceq_t y$.

Theorem

Every finite poset has a topological sort.

Topological sorting

Definition

A topological sort or a linearization of a poset $(S, \preceq)$ is a poset $(S, \preceq_t)$ with a total order $\preceq_t$ such that $x \preceq y$ implies $x \preceq_t y$.

Theorem

Every finite poset has a topological sort.

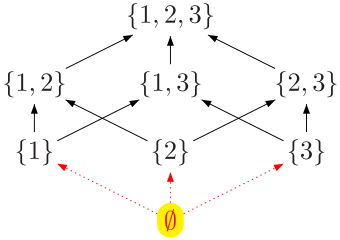
Proof: (H.W)

- Recall the lemma:
  - Every finite non-empty poset has at least one minimal element (x is minimal if $\nexists y, y \preceq x$).
- Then, construct a (new) chain to complete the proof.

Topological sorting: example

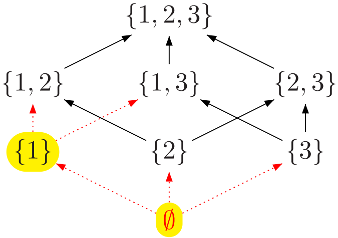
$$
\begin{tikzcd}
 & & \{1,2,3\} & & \\
\{1,2\} \arrow[urr] & & \{1,3\} \arrow[u] & & \{2,3\} \arrow[ull] \\
\{1\} \arrow[u] \arrow[urr] & & \{2\} \arrow[ull] \arrow[urr] & & \{3\} \arrow[ull] \arrow[u] \\
 & & \emptyset \arrow[ull] \arrow[u] \arrow[urr] & &
\end{tikzcd}
$$

Topological sorting: example

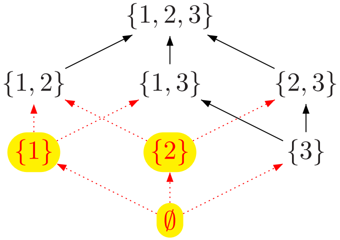


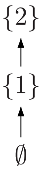
$\emptyset$

Topological sorting: example

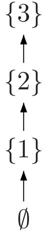


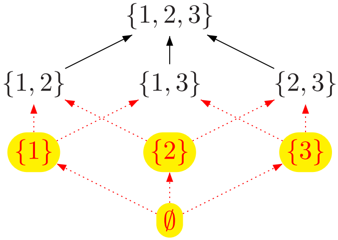
Topological sorting: example



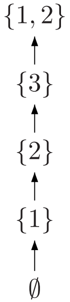


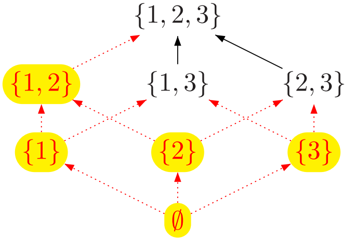
Topological sorting: example



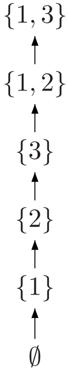


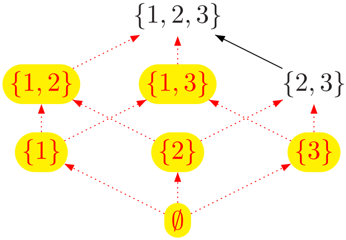
Topological sorting: example



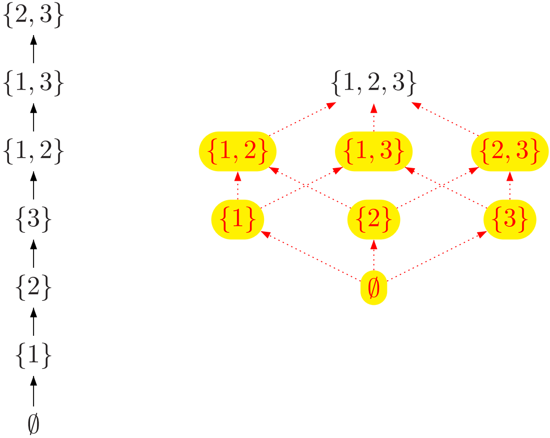


Topological sorting: example





Topological sorting: example



Topological sorting: example

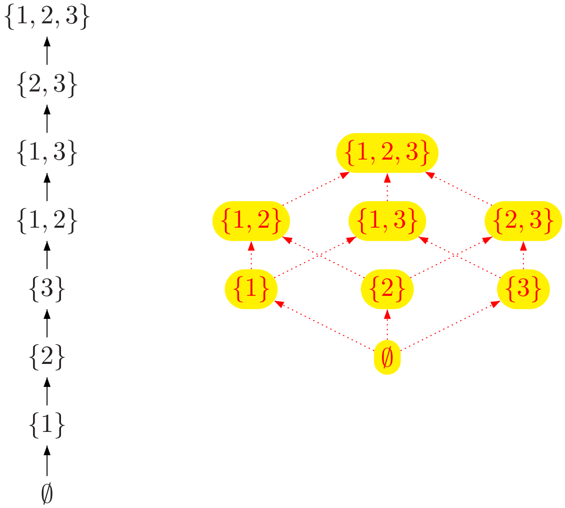
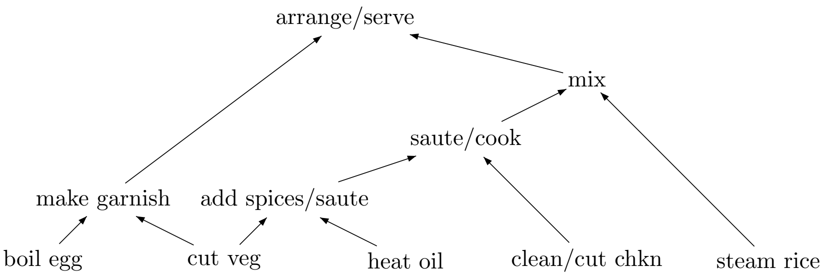


Figure: A poset and its Topological sort.

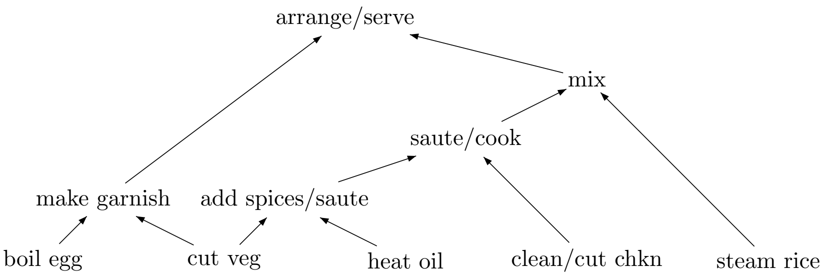
Tasks scheduling as a poset

Coming back to our example,



Tasks scheduling as a poset

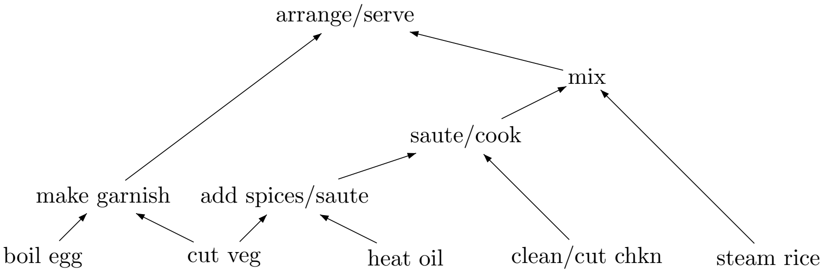
Coming back to our example,



- Assume that every task takes 1 time unit.

Tasks scheduling as a poset

Coming back to our example,

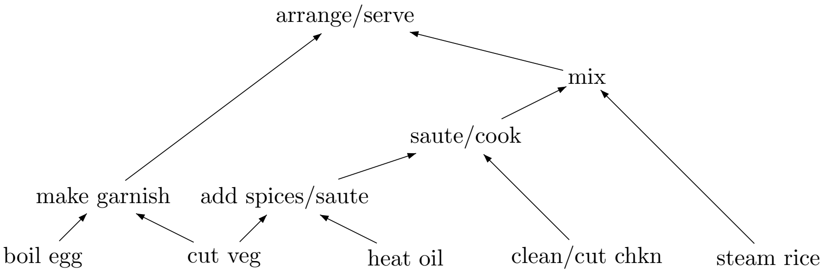


- Assume that every task takes 1 time unit.
- How much time is required?

Parallel Task Scheduling and chains

Coming back to our example,

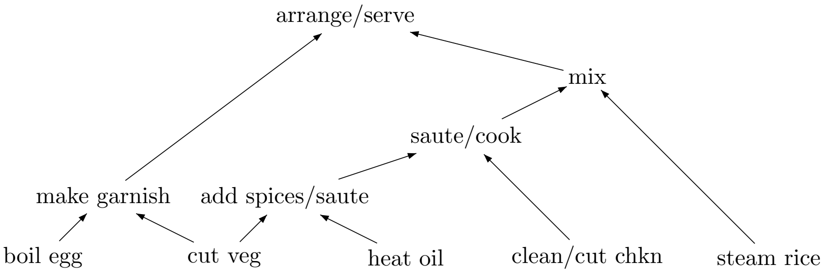
- What if there are many cooks, i.e., parallel processors?
- How do we schedule the tasks to minimize time used?



Parallel Task Scheduling and chains

Coming back to our example,

- What if there are many cooks, i.e., parallel processors?
- How do we schedule the tasks to minimize time used?

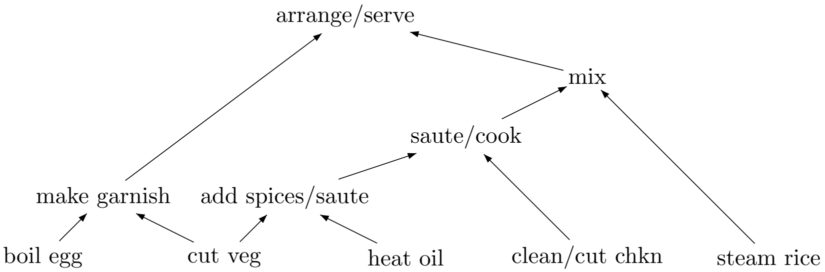


- Assume that every task takes 1 time unit.

Parallel Task Scheduling and chains

Coming back to our example,

- What if there are many cooks, i.e., parallel processors?
- How do we schedule the tasks to minimize time used?

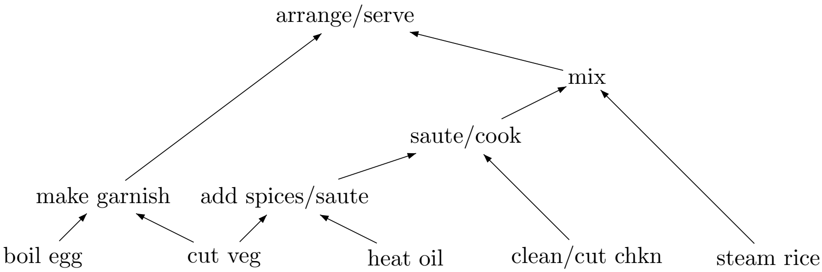


- Assume that every task takes 1 time unit.
- Clearly, we still need at least 5 time units.

Parallel Task Scheduling and chains

Coming back to our example,

- What if there are many cooks, i.e., parallel processors?
- How do we schedule the tasks to minimize time used?



- Assume that every task takes 1 time unit.
- Clearly, we still need at least 5 time units.
- That is, the length of the longest chain (length of chain = no. of elements in it).

Parallel Task Scheduling

For any poset, there is a legal parallel schedule that runs in t steps, where t is the length of the longest chain.

Parallel Task Scheduling

For any poset, there is a legal parallel schedule that runs in t steps, where t is the length of the longest chain.

We will in fact prove:

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Parallel Task Scheduling

For any poset, there is a legal parallel schedule that runs in t steps, where t is the length of the longest chain.

We will in fact prove:

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Assuming this theorem,

- Observe that we can schedule all of S_i at time i (since we know that all previous tasks were done earlier!).
- Thus, each S_i is an anti-chain.
- This solves the parallel task scheduling problem.

Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

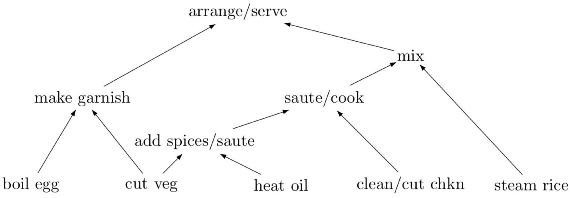
Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$



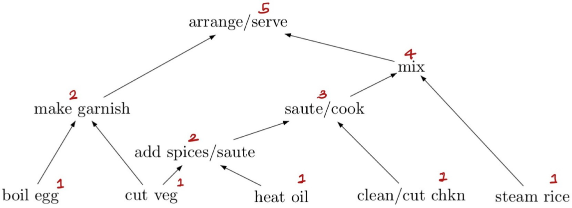
Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$



Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Parallel task scheduling and chains

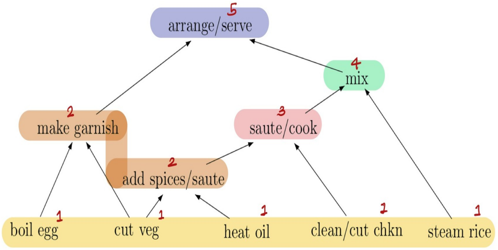
Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$



Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Lemma

For $a \in S_i$, if $b \preceq a$ and $b \neq a$, then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Parallel task scheduling and chains

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Lemma

For $a \in S_i$, if $b \preceq a$ and $b \neq a$, then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Parallel task scheduling and chains

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Lemma

For $a \in S_i$, if $b \preceq a$ and $b \neq a$, then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

- Suppose $a \in S_i, b \preceq a, b \neq a$ but $b \notin S_1 \cup \ldots \cup S_{i-1}$.

Parallel task scheduling and chains

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Lemma

For $a \in S_i$, if $b \preceq a$ and $b \neq a$, then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

- Suppose $a \in S_i, b \preceq a, b \neq a$ but $b \notin S_1 \cup \ldots \cup S_{i-1}$.
- By defn of S_i, $\exists$ chain of length at least i ending at b.

Parallel task scheduling and chains

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Lemma

For $a \in S_i$, if $b \preceq a$ and $b \neq a$, then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

- Suppose $a \in S_i, b \preceq a, b \neq a$ but $b \notin S_1 \cup \ldots \cup S_{i-1}$.
- By defn of S_i, $\exists$ chain of length at least i ending at b.
- But now, $b \preceq a, b \neq a$ implies we can extend the chain to chain of length $\geq i + 1$, ending at a.

Parallel task scheduling and chains

$$\mathsf{height}(x) = \{\text{max size of a chain ending at } x\}$$

$$S_i = \{x \mid \mathsf{height}(x) = i\}$$

Lemma

For $a \in S_i$, if $b \preceq a$ and $b \neq a$, then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof:

- Suppose $a \in S_i, b \preceq a, b \neq a$ but $b \notin S_1 \cup \ldots \cup S_{i-1}$.
- By defn of S_i, $\exists$ chain of length at least i ending at b.
- But now, $b \preceq a, b \neq a$ implies we can extend the chain to chain of length $\geq i + 1$, ending at a.
- But then a cannot be in S_i. Contradiction. $\square$

Parallel task scheduling and chains

Theorem

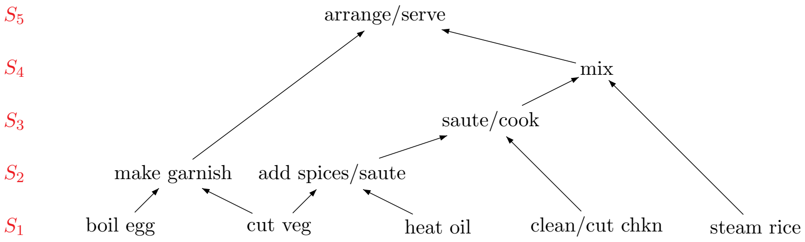
For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with length of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Proof: Put each $a \in S$ in S_i such that i is the length of the longest chain ending at a.



Consequences for chains and anti-chains

Since each S_i was an anti-chain, a celebrated result follows...

Corollary (Mirsky's theorem, 1971)

If the longest chain in a poset $(S, \preceq)$ is of length t, then S can be partitioned into t anti-chains.

Consequences for chains and anti-chains

Since each S_i was an anti-chain, a celebrated result follows...

Corollary (Mirsky's theorem, 1971)

If the longest chain in a poset $(S, \preceq)$ is of length t, then S can be partitioned into t anti-chains.

(H.W) Apply this on the poset $(\mathcal{P}(S), \subseteq)$, where $S = \{1, 2, 3\}$. What does it imply for this poset?

Consequences for chains and anti-chains

Since each S_i was an anti-chain, a celebrated result follows...

Corollary (Mirsky's theorem, 1971)

If the longest chain in a poset $(S, \preceq)$ is of length t, then S can be partitioned into t anti-chains.

(H.W) Apply this on the poset $(\mathcal{P}(S), \subseteq)$, where $S = \{1, 2, 3\}$. What does it imply for this poset?

Another corollary (Dilworth's Lemma)

For all $t > 0$, any poset with n elements must have

- either a chain of length greater than t
- or an antichain with at least $\frac{n}{t}$ elements.

(H.W): Prove it!