CS 105: Department Introductory Course on Discrete Structures

Instructor : S. Akshay

Sep 02, 2024

Lecture 15 – Basic structures: chains and anti-chains

Logistics

Optional help session today: at 7pm

- Problemsheet 5

Logistics

Optional help session today: at 7pm

- Problemsheet 5

Answersheets of Quiz 1

- Answerkey and grading scheme on piazza by afternoon.

Logistics

Optional help session today: at 7pm

- Problemsheet 5

Answersheets of Quiz 1

- Answerkey and grading scheme on piazza by afternoon.
- Venue: KR125, Time: 7.45pm onwards.

Logistics

Optional help session today: at 7pm

- Problemsheet 5

Answersheets of Quiz 1

- Answerkey and grading scheme on piazza by afternoon.
- Venue: KR125, Time: 7.45pm onwards.
- In batches of 25, 10min each batch.

Logistics

Optional help session today: at 7pm

- Problemsheet 5

Answersheets of Quiz 1

- Answerkey and grading scheme on piazza by afternoon.
- Venue: KR125, Time: 7.45pm onwards.
- In batches of 25, 10min each batch.
- Exact schedule by Roll No. will be on piazza by afternoon.
- Please see that and come only on your slot.

Summary

Week 01 and 02: Proofs and Reasoning

- Propositions, Predicates, Quantifiers
- Theorems and Types of Proofs
- Induction and variants

Summary

Week 01 and 02: Proofs and Reasoning

- Propositions, Predicates, Quantifiers
- Theorems and Types of Proofs
- Induction and variants

Week 03 and 04: Basic Structures – Sets and Functions

- Finite and infinite sets.
- Using functions to compare sets: focus on bijections.
- Countable, countably infinite and uncountable sets.
- Cantor's diagonalization argument

Summary

Week 01 and 02: Proofs and Reasoning

- Propositions, Predicates, Quantifiers
- Theorems and Types of Proofs
- Induction and variants

Week 03 and 04: Basic Structures – Sets and Functions

- Finite and infinite sets.
- Using functions to compare sets: focus on bijections.
- Countable, countably infinite and uncountable sets.
- Cantor's diagonalization argument

Week 05 and 06: Basic Structures – Relations

- Equivalence relations and partitions of a set
- Partially ordered sets (posets), chains and anti-chains
- Applications: to task scheduling

Chains and Anti-chains

Definition

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 $\preceq$.
- That is, $\forall a, b \in B$, we have $a \preceq b$ or $b \preceq a$ (or both).

Chains and Anti-chains

Definition

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 $\preceq$.
- That is, $\forall a, b \in B$, we have $a \preceq b$ or $b \preceq a$ (or both).
- Thus, $\preceq$ is a total order on B.

Chains and Anti-chains

Definition

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 $\preceq$.
- That is, $\forall a, b \in B$, we have $a \preceq b$ or $b \preceq a$ (or both).
- Thus, $\preceq$ is a total order on B.

Definition

Let $(S, \preceq)$ be a poset. A subset $A \subseteq S$ is called

- an anti-chain if no two distinct elements of A are related to each other under $\preceq$.

Chains and Anti-chains

Definition

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 $\preceq$.
- That is, $\forall a, b \in B$, we have $a \preceq b$ or $b \preceq a$ (or both).
- Thus, $\preceq$ is a total order on B.

Definition

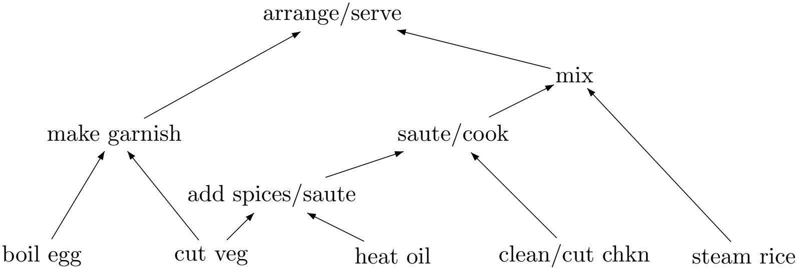
Let $(S, \preceq)$ be a poset. A subset $A \subseteq S$ is called

- an anti-chain if no two distinct elements of A are related to each other under $\preceq$.
- That is, $\forall a, b \in A, a \neq b$, we have neither $a \preceq b$ nor $b \preceq a$.

Examples and applications

A task scheduling example

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

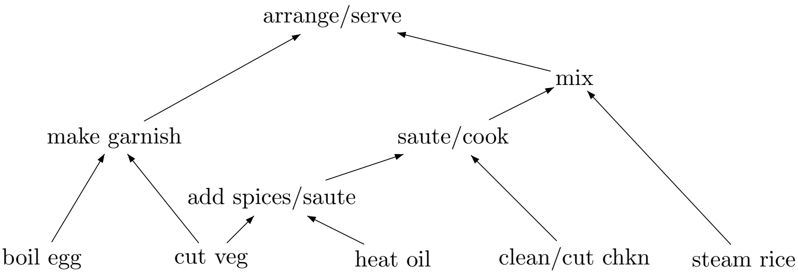


- Dependencies give rise to a poset

Examples and applications

A task scheduling example

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



- Dependencies give rise to a poset
- But for cooking you need a total order, that is consistent with the po.
- This is called a linearization or a topological sorting.

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.

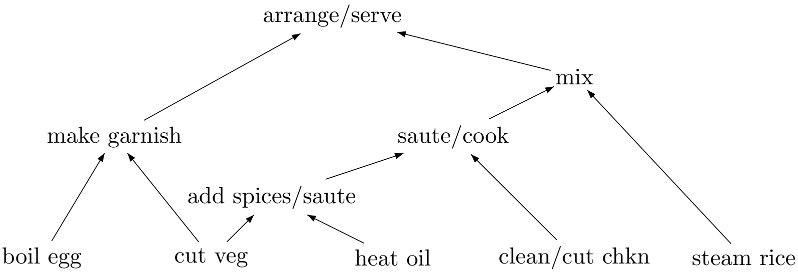
Proof Sketch

- Use the following lemma:
  - Every finite non-empty poset has at least one minimal element (x is minimal if $\nexists y, y \preceq x$).
- Construct the linearization, one minimal element at a time, to complete the proof.

Parallel Task Scheduling and chains

Coming back to our example,

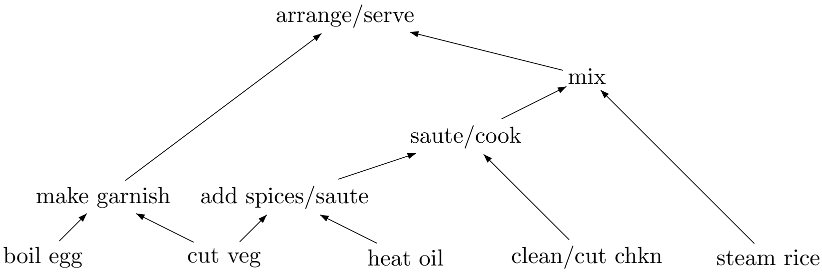
- What if there are many cooks, i.e., parallel processors?
- How do we schedule the tasks to minimize time used?



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?

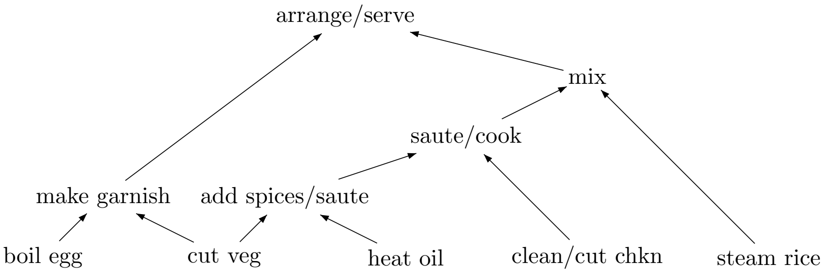


- Assume that every task takes 1 time unit.

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?

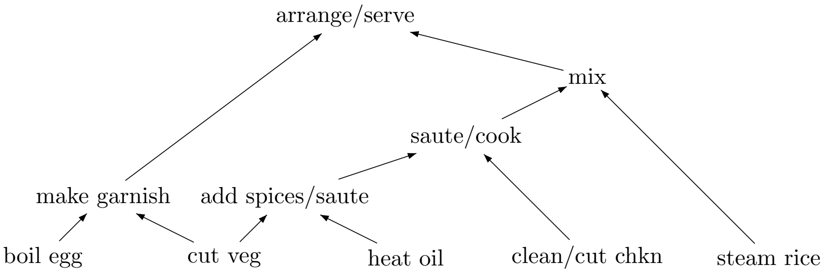


- Assume that every task takes 1 time unit.
- Clearly, we still need at least 5 time units.

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 size of the largest/longest chain (size 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 size 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 size of the longest chain.

We will in fact prove:

Theorem

For a finite poset $(S, \preceq)$ with size 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 size of the longest chain.

We will in fact prove:

Theorem

For a finite poset $(S, \preceq)$ with size 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

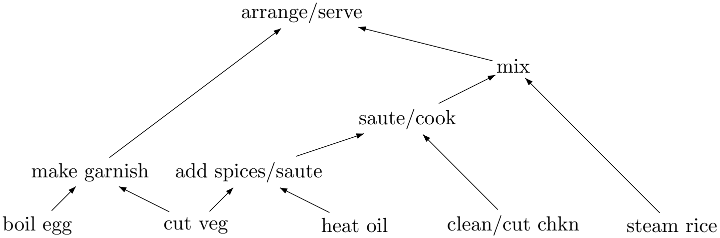
For a finite poset $(S, \preceq)$ with size of longest chain $= t$, we can partition S into t subsets $S_1, \ldots, S_t$ such that $\forall i \in \{1, \ldots, t\}$, $\forall a \in S_i$, if $b \preceq a, b \neq a$ then $b \in S_1 \cup \ldots \cup S_{i-1}$.

Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with size 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 size of the largest chain ending at a.

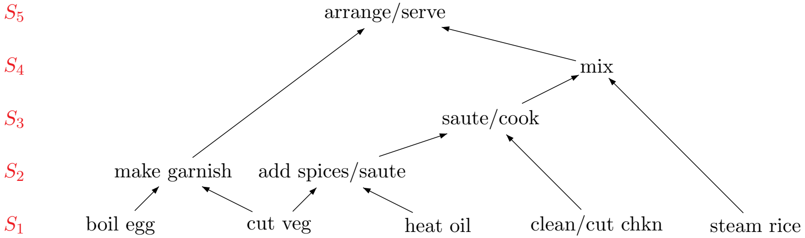


Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with size 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 size of the largest chain ending at a. Now proof by contradiction:



Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with size 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 size of the largest chain ending at a. Now proof by contradiction:

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

Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with size 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 size of the largest chain ending at a. Now proof by contradiction:

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

Parallel task scheduling and chains

Theorem

For a finite poset $(S, \preceq)$ with size 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 size of the largest chain ending at a. Now proof by contradiction:

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

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 largest chain in a poset $(S, \leq)$ is of size 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 largest chain in a poset $(S, \leq)$ is of size t, then S can be partitioned into t anti-chains.

Exercise: 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 largest chain in a poset $(S, \leq)$ is of size t, then S can be partitioned into t anti-chains.

Exercise: 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 size greater than t
- or an antichain with at least $\frac{n}{t}$ elements.

Exercise: Prove it!