Discrete Mathematics with Applications I

COMPSCI&SFWRENG 2DM3

McMaster University, Fall 2019

Wolfram Kahl

2019-11-27

Bag Product and Bag Reconstitution

Recall: A **bag** is "like a set, but each element can occur any (finite) number of times".

$$\wr\, x : \mathbb{Z} \mid -2 \le x \le 2 \bullet x \cdot x \,\int \quad = \quad \wr 4, 1, 0, 1, 4 \int \quad = \quad \wr 0, 1, 1, 4, 4 \int \quad \ne \quad \wr 0, 1, 4 \int$$

$\_\#\_ : t \to Bag\ t \to \mathbb{N}$ counts the number of occurrences: $1 \# \wr 0, 0, 0, 1, 1, 4 \int = 2$

$\_\sqsubseteq\_ : t \to Bag\ t \to \mathbb{B}$ is membership, with $x \sqsubseteq B \equiv x \# B \ne 0$: $1 \sqsubseteq \wr 0, 0, 0, 1, 1, 4 \int = true$

Calculate: $\wr x \mid x \sqsubseteq \wr 0, 0, 0, 1, 1, 4 \int \int = \textbf{?}$

Define $bagProd : Bag\ \mathbb{N} \to \mathbb{N}$ such that: $bagProd\ \wr e_1, e_2, \ldots, e_n \int = e_1 \cdot e_2 \cdot \ldots \cdot e_n$

e.g., $bagProd\ \wr 2, 2, 3, 3, 5 \int = 180$

- Easy with exponentiation $\_{*}{*}\_$: $bagProd\ B = \prod \textbf{?}$
- Without exponentiation: **?**

Related question: For sets, we have (11.5): $S = \{x \mid x \in S \bullet x\}$

What is the corresponding theorem for bags?

Bag reconstitution: $B = \wr\ \textbf{?} \mid \textbf{?} \bullet \textbf{?}\ \int$

Plan for Today

- **Graph Concepts via Relations: Closures, Reachability**
- **Induction, Induction Principles**

Recall: Symmetric Closure

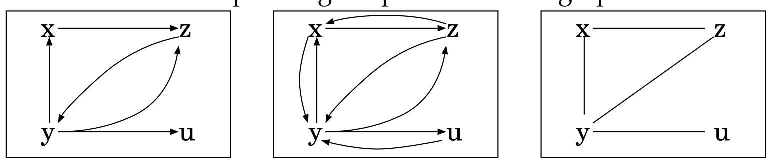
Relation $Q : B \leftrightarrow B$ is the **symmetric closure** of $R : B \leftrightarrow B$ iff Q is the smallest symmetric relation containing R,

or, equivalently, iff

- $R \subseteq Q$
- $Q = Q^{\smile}$
- $(\forall P : B \leftrightarrow B \mid R \subseteq P = P^{\smile} \bullet Q \subseteq P)$

Theorem: The symmetric closure of $R : B \leftrightarrow B$ is $R \cup R^{\smile}$.

Fact: If R represents a simple directed graph, then the symmetric closure of R is the associated relation of the corresponding simple undirected graph.



Recall: Reflexive Closure

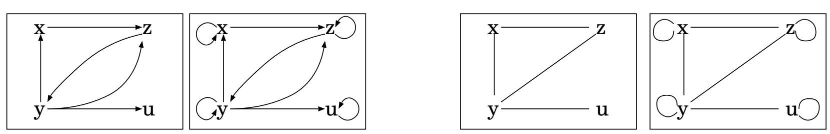
Relation $Q : B \leftrightarrow B$ is the **reflexive closure** of $R : B \leftrightarrow B$ iff Q is the smallest reflexive relation containing R,

or, equivalently, iff

- $R \subseteq Q$
- $\mathrm{Id} \subseteq Q$
- $(\forall P : B \leftrightarrow B \mid R \subseteq P \land \mathrm{Id} \subseteq P \bullet Q \subseteq P)$

Theorem: The reflexive closure of $R : B \leftrightarrow B$ is $R \cup \mathrm{Id}$.

Fact: If R represents a graph, then the reflexive closure of R "ensures that each node has a loop edge".



Closures

Let Ω be a property on relations, i.e.:

$$\Omega \quad : \quad (B \leftrightarrow C) \to \mathbb{B}$$

Relation $Q : B \leftrightarrow C$ is the **Ω-closure** of $R : B \leftrightarrow C$ iff

- Q is the smallest relation
- that contains R
- and has property Ω

or, equivalently, iff

- $R \subseteq Q$
- $\Omega\, Q$
- $(\forall P : B \leftrightarrow C \mid R \subseteq P \land \Omega\, P \bullet Q \subseteq P)$

(For some properties, closures are not defined, or not always defined.)

Transitive Closure

Relation $Q : B \leftrightarrow B$ is the **transitive closure** of $R : B \leftrightarrow B$
iff Q is the smallest transitive relation containing R,

or, equivalently, iff

- $R \subseteq Q$
- $Q \mathbin{;} Q \subseteq Q$
- $(\forall P : B \leftrightarrow B \mid R \subseteq P \land P \mathbin{;} P \subseteq P \bullet Q \subseteq P)$

Definition: The transitive closure of $R : B \leftrightarrow B$ is written R^+.

Theorem: $R^+ = (\cap\ P \mid R \subseteq P \land P \mathbin{;} P \subseteq P \bullet P)$.

Theorem: $R^+ = (\cup\ i : \mathbb{N} \mid i > 0 \bullet R^i)$

Powers of a homogeneous relation $R : B \leftrightarrow B$:

- $R^0 = \mathrm{Id}$
- $R^1 = R$
- $R^{n+1} = R^n \mathbin{;} R$

Reflexive Transitive Closure

$Q : B \leftrightarrow B$ is the **reflexive transitive closure** of $R : B \leftrightarrow B$
iff Q is the smallest reflexive transitive relation containing R,

or, equivalently, iff

- $R \subseteq Q$
- $\mathrm{Id} \subseteq Q \land Q \mathbin{;} Q \subseteq Q$
- $(\forall P : B \leftrightarrow B \mid R \subseteq P \land \mathrm{Id} \subseteq P \land P \mathbin{;} P \subseteq P \bullet Q \subseteq P)$

Definition: The reflexive transitive closure of R is written R^*.

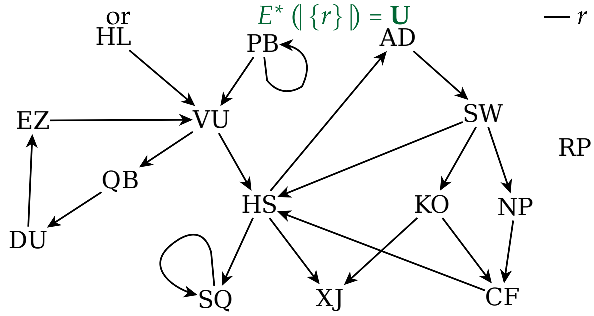
Theorem: $R^* = (\cap\ P \mid R \subseteq P \land \mathrm{Id} \subseteq P \land P \mathbin{;} P \subseteq P \bullet P)$.

Theorem: $R^* = (\cup\ i : \mathbb{N} \bullet R^i)$

- Transitive closure R^+ is reachability via at least one R-step
- Reflexive transitive closure R^* is reachability via any number of R-steps
- Variants of the **Warshall algorithm** calculate these closures in cubic time.

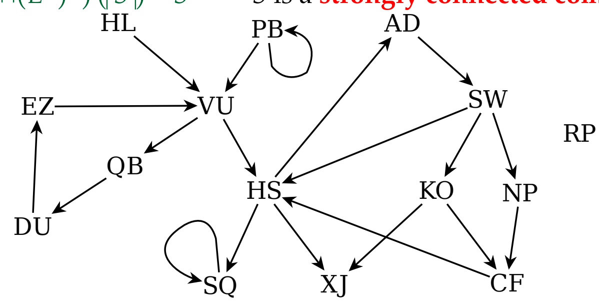
Reachability in graph $G = (V, E)$ — 1 (ctd.)

- No edge ends at node s
  $s \notin \mathit{Ran}\ E$ or $s \in \sim(\mathit{Ran}\ E)$ — s is called a **source** of G
- No edge starts at node s
  $s \notin \mathit{Dom}\ E$ or $s \in \sim(\mathit{Dom}\ E)$ — s is called a **sink** of G
- Node n_2 is reachable from node n_1 via a three-edge path
  $n_1 \llbracket E^3 \rrbracket n_2$ or $n_1 \llbracket E \mathbin{;} E \mathbin{;} E \rrbracket n_2$
- Every node is reachable from node r
  $\{r\} \times \mathbf{U} \subseteq E^*$ or $E^* \llparenthesis \{r\} \rrparenthesis = \mathbf{U}$ — r is called a **root** of G



Reachability in graph $G = (V, E)$ — 2

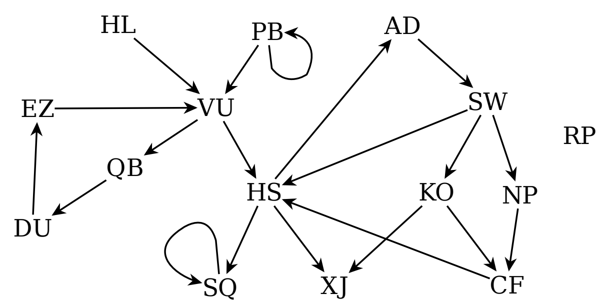
- From every node, each node is reachable
  $V \times V \subseteq E^*$ or $\sim \mathrm{Id} \subseteq E^+$ — G is **strongly connected**
- From every node, each node is reachable by traversing edges in either direction
  $V \times V \subseteq (E \cup E^{\smile})^*$ or $\sim \mathrm{Id} \subseteq (E \cup E^{\smile})^+$ — G is **connected**
- Nodes n_1 and n_2 reachable from each other both ways
  $n_1 \, (E^* \cap (E^*)^{\smile}) \, n_2$ — n_1 and n_2 are **strongly connected**
- S is an equivalence class of strong connectedness between nodes
  $S \times S \subseteq E^* \wedge (E^* \cap (E^*)^{\smile})(\!| S |\!) = S$ — S is a **strongly connected component (SCC)** of G



Reachability in graph $G = (V, E)$ — 3

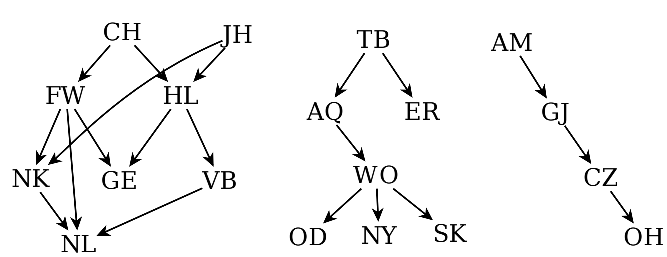
- A node n is said to "lie on a cycle" if there is a non-empty path from n to n

  $\mathit{cycleNodes} := \mathit{Dom}(E^+ \cap \mathrm{Id})$
- No node lies on a cycle
  $E^+ \cap \mathrm{Id} = \{\}$ — G is called **acyclic** or **cycle-free** or a **DAG**



Reachability in graph $G = (V, E)$ — 4 — DAGs

- No node lies on a cycle
  $E^+ \cap \mathrm{Id} = \{\}$ — G is a **directed acyclic graph**, or **DAG**
- Each node has at most one predecessor
  $E \,;\, E^{\smile} \subseteq \mathrm{Id}$ or E is injective
  — if G is also acyclic, then G is called a **(directed) forest**
- Every node is reachable from node r
  $\{r\} \times V \subseteq E^*$ — if G is also a forest, then G is called a **(directed) tree**, and r is its **root**



Natural Numbers — Induction Principle

- The set of all **natural numbers** is written $\mathbb{N}$.
- Zero "0" is a natural number.
- If n is a natural number, then its successor "suc n" is a natural number, too.

Induction principle for the natural numbers:

- if $P(0)$ — If P holds for 0
- and if $P(m)$ implies $P(\mathsf{suc}\ m)$, — and whenever P holds for m, it also holds for suc m,
- then for all $m : \mathbb{N}$ we have $P(m)$. — then P holds for all natural numbers.

Natural Numbers — Induction Principle

Recall: Induction principle for the natural numbers:

- if $P(0)$ — If P holds for 0
- and if $P(m)$ implies $P(\mathsf{suc}\ m)$, — and whenever P holds for m, it also holds for suc m,
- then for all $m : \mathbb{N}$ we have $P(m)$. — then P holds for all natural numbers.

As **inference rule**:

Informally:

$$\frac{P(0) \qquad \begin{array}{c} \ulcorner P(m) \urcorner \\ \vdots \\ P(\mathsf{suc}\ m) \end{array}}{P(m)}$$

Formally:

$$\frac{P[m := 0] \qquad \begin{array}{c} \ulcorner P \urcorner \\ \vdots \\ P[m := \mathsf{suc}\ m] \end{array}}{P}$$

As **axiom / theorem**: $P[m := 0] \quad \Rightarrow \quad (\forall\, m : \mathbb{N} \mid P \bullet P[m := \mathsf{suc}\ m]) \quad \Rightarrow \quad (\forall\, m : \mathbb{N} \bullet P)$

```
Axiom “Induction over ℕ”:
   P[n ≔ 0]
   ⇒ (∀ n : ℕ ❙ P • P[n ≔ suc n])
   ⇒ (∀ n : ℕ • P)
```

Proving "Right-identity of +" Using the Induction Principle

```
Axiom “Induction over ℕ”:
   P[n ≔ 0]
   ⇒ (∀ n : ℕ ❙ P • P[n ≔ suc n])
   ⇒ (∀ n : ℕ • P)

Theorem “Right-identity of +”: ∀ m : ℕ • m + 0 = m
Proof:
  Using “Induction over ℕ”:
    Subproof for `(m + 0 = m)[m ≔ 0]`:
      By substitution and “Definition of +”
    Subproof for `∀ m : ℕ ❙ m + 0 = m • (m + 0 = m)[m ≔ suc m]`:
      For any `m : ℕ` satisfying `m + 0 = m`:
          (m + 0 = m)[m ≔ suc m]
        =⟨ Substitution, “Definition of +” ⟩
          suc (m + 0) = suc m
        =⟨ Assumption `m + 0 = m`, “Reflexivity of =” ⟩
          true
```

Proving "Right-identity of +" `Using` the Induction Principle (v2)

```
Axiom "Induction over ℕ":
   P[n ≔ 0]
   ⇒ (∀ n : ℕ ❙ P • P[n ≔ suc n])
   ⇒ (∀ n : ℕ • P)

Theorem "Right-identity of +": ∀ m : ℕ • m + 0 = m
Proof:
  Using "Induction over ℕ":
    Subproof for `0 + 0 = 0`:
      By "Definition of +"
    Subproof for `∀ m : ℕ ❙ m + 0 = m • suc m + 0 = suc m`:
      For any `m : ℕ` satisfying `m + 0 = m`:
          suc m + 0
        =⟨ "Definition of +" ⟩
          suc (m + 0)
        =⟨ Assumption `m + 0 = m` ⟩
          suc m
```

Proving "Right-identity of +" `Using` the Induction Principle (v3)

```
Theorem "Right-identity of +": ∀ m : ℕ • m + 0 = m
Proof:
  Using "Induction over ℕ":
    Subproof:
        0 + 0
      =⟨ "Definition of +" ⟩
        0
    Subproof:
      For any `m : ℕ` satisfying "IndHyp" `m + 0 = m`:
          suc m + 0
        =⟨ "Definition of +" ⟩
          suc (m + 0)
        =⟨ Assumption "IndHyp" ⟩
          suc m
```

```
Axiom "Induction over ℕ":
   P[n ≔ 0]
   ⇒ (∀ n : ℕ ❙ P • P[n ≔ suc n])
   ⇒ (∀ n : ℕ • P)
```

- Using induction principles directly is not much more verbose than "By induction on …"
- "By induction on …" only supports **very few** built-in induction principles
- Induction principles can be derived as theorems, or provided as axioms, and then can be used directly!

Sequences — Induction Principle

Induction principle for sequences:

- if $P(\epsilon)$ — If P holds for ϵ
- and if $P(xs)$ implies $P(x \triangleleft xs)$ for all $x : A$, — and whenever P holds for xs, it also holds for any $x \triangleleft xs$,
- then for all $xs : \mathsf{Seq}\ A$ we have $P(xs)$. — then P holds for all sequences over A.

$$\begin{aligned} P[xs := \epsilon] \quad &\Rightarrow \quad (\forall\, xs : \mathsf{Seq}\ A \mid P \bullet (\forall\, x : A \bullet P[xs := x \triangleleft xs]) \\ &\Rightarrow \quad (\forall\, xs : \mathsf{Seq}\ A \bullet P) \end{aligned}$$

```
Axiom "Induction over sequences":
    P[xs ≔ ϵ]
    ⇒ (∀ xs : Seq A ❙ P • (∀ x : A • P[xs ≔ x ◃ xs]))
    ⇒ (∀ xs : Seq A • P)
```

$$P[m := 0] \quad \Rightarrow \quad (\forall\, m : \mathbb{N} \mid P \bullet P[m := \mathsf{suc}\ m]) \quad \Rightarrow \quad (\forall\, m : \mathbb{N} \bullet P)$$

```
Axiom "Induction over ℕ":
   P[n ≔ 0]
   ⇒ (∀ n : ℕ ❙ P • P[n ≔ suc n])
   ⇒ (∀ n : ℕ • P)
```

Recall: Tail is different — LADM Proof

Theorem (13.7) "Tail is different": (∀ xs : Seq A • (∀ x : A • x ◁ xs ≠ xs))
Proof:
  By induction on `xs : Seq A`:
    Base case:
      For any x : A:
```
          x ◁ ϵ ≠ ϵ
        ≡⟨ "Cons is not empty" ⟩
          true
```
    Induction step:
      For any z : A:
        For any x : A:
```
            x ◁ (z ◁ xs) ≠ z ◁ xs
          ≡⟨ "Definition of ≠", "Injectivity of ◁" ⟩
            ¬ (x = z ∧ z ◁ xs = xs)
          ⇐⟨ "Consequence", "De Morgan", "Weakening", "Definition of ≠" ⟩
            z ◁ xs ≠ xs
          ≡⟨ Induction hypothesis ⟩
            true
```

Proving "Tail is different" Using the Ind. Principle

```
Axiom "Induction over sequences":
    P[xs ≔ ϵ]
    ⇒ (∀ xs : Seq A ❙ P • (∀ x : A • P[xs ≔ x ◁ xs]))
    ⇒ (∀ xs : Seq A • P)

Theorem (13.7) "Tail is different": ∀ xs : Seq A • ∀ x : A • x ◁ xs ≠ xs
Proof:
  Using "Induction over sequences":
    Subproof for `∀ x : A • x ◁ ϵ ≠ ϵ`:
      For any `x : A`:
          x ◁ ϵ ≠ ϵ
        ≡⟨ "Cons is not empty" ⟩
          true
    Subproof for `∀ xs : Seq A ❙              (∀ x : A • x ◁ xs ≠ xs)
                                   • (∀ z : A • (∀ x : A • x ◁ z ◁ xs ≠ z ◁ xs))`:
      For any `xs : Seq A` satisfying "Ind. Hyp." `(∀ x : A • x ◁ xs ≠ xs)`:
        For any `z : A`, `x : A`:
            x ◁ z ◁ xs ≠ z ◁ xs
          ≡⟨ "Definition of ≠", "Injectivity of ◁" ⟩
            ¬ (x = z ∧ z ◁ xs = xs)
          ⇐⟨ "Consequence", "De Morgan", "Weakening", "Definition of ≠" ⟩
            z ◁ xs ≠ xs
          ≡⟨ Assumption "Ind. Hyp." ⟩
            true
```

Idea Behind Induction — How Does It Work? — Informally

Proving $(\forall\, x : t \;\bullet\; P)$ by induction, **for an appropriate type** t:

- You are familiar with proving a base case and an induction step
- The base case establishes $P[x := S]$ where S is "the simplest t"
- The induction step works for $x : t$ for which we already know $P[x := x]$ and from that establishes $P[x := C\, x]$ for elements $C\, x : t$ that "are slightly more complicated than x".
- Since the construction principle ("C") used in the induction step is sufficiently powerful to construct all $x : t$, this justifies $(\forall\, x : t \;\bullet\; P)$.

Looking at this from the other side:

- Each element $x : t$ is either a "simplest element" ("S"), or constructed via a construction principle ("C") from "slightly simpler elements" y, that is, $x = C\, y$.
- In the first case, the base case gives you the proof for $P[x := S]$.
- In the second case, you obtain $P[x := Cy]$ via the induction step from a proof for $P[x := y]$, if you can find that.
- You can find that proof if repeated decomposition into S or C always terminates.