# Category Theory and Functional Programming

Day 2

7 October 2009

# Categories, functors, natural transformations

1. Graphs vs. categories
2. Exercise P-1.1.20.2 (Petur)
3. Transition systems revisited
4. Functors
5. Exercise P-2.1.10.3
6. Exercise ML-1.3.4
7. Natural transformations
8. Exercise P-2.3.11.2 (Mikkel)

# Graphs

- Set of points $V$
- Set of edges $E$
- For each edge $e \in E$, a source $src(e) \in V$ and a target $tgt(e) \in V$
- (Write $e : x \to y$ if $src(e) = x$ and $tgt(e) = y$)

(These are directed multigraphs; to say $E \subseteq V \times V$ is fine as long as there's at most one edge between any two points.)

- *That's all folks:* $V, E, src : E \to V, tgt : E \to V$

# Reflexive graphs

- Set of points $V$
- Set of edges $E$
- For each edge $e \in E$, a source $src(e) \in V$ and a target $tgt(e) \in V$
- (Write $e : x \to y$ if $src(e) = x$ and $tgt(e) = y$)
- For each point $x \in V$, a degenerate edge $\deg(v) \in E$

- *That's all folks:*
  $V, E, src : E \to V, tgt : E \to V, \deg : V \to E$

# Categories

- Set of points $V$
- Set of edges $E$
- For each edge $e \in E$, a source $src(e) \in V$ and a target $tgt(e) \in V$
- (Write $e : x \to y$ if $src(e) = x$ and $tgt(e) = y$)
- For each point $x \in V$, a degenerate edge $\deg(v) \in E$
- For each $e_1 : x \to y$ and $e_2 : y \to z$, a composite $e_2 \circ e_1 : x \to z$,
- with associativity: $e_3 \circ (e_2 \circ e_1) = (e_3 \circ e_2) \circ e_1$ whenever these are defined,
- and identities: for all edges $e : x \to y$, $e \circ \deg(x) = e$ and $\deg(y) \circ e = e$.
- *That's all folks:* $V, E, src : E \to V, tgt : E \to V, \deg : V \to E, \circ : E \times_V E \to E$

# Categories

- Set of objects $\mathcal{C}_0$
- Set of arrows $\mathcal{C}_1$
- For each arrow $f \in \mathcal{C}_1$, a domain $dom(f) \in \mathcal{C}_0$ and a co-domain $cod(f) \in \mathcal{C}_0$
- (Write $f : A \to B$ if $dom(f) = A$ and $cod(f) = B$)
- For each object $A \in \mathcal{C}_0$, an identity arrow $\mathrm{id}_A \in \mathcal{C}_0$
- For each $f_1 : A \to B$ and $f_2 : B \to C$, a composite $f_2 \circ f_1 : A \to C$,
- with associativity: $f_3 \circ (f_2 \circ f_1) = (f_3 \circ f_2) \circ f_1$ whenever these are defined,
- and identities: for all arrows $f : A \to B$, $f \circ \mathrm{id}_A = f$ and $\mathrm{id}_B \circ f = f$.
- *That's all folks:* $\mathcal{C}_0, \mathcal{C}_1, dom, cod : \mathcal{C}_1 \to \mathcal{C}_0, \mathrm{id} : \mathcal{C}_0 \to \mathcal{C}_1, \circ : \mathcal{C}_1 \times_{\mathcal{C}_0} \mathcal{C}_1 \to \mathcal{C}_1$

# Exercise P-1.1.20.2 (Petur)

A group $(G, *, e, {}^{-1})$ is a set $G$ equipped with a binary operation $*$, a distinguished element $e$, and a unary operation ${}^{-1}$ such that

(a) $(x * y) * z = x * (y * z)$ for all $x, y, z \in G$,

(b) $e * x = x = x * e$ for all $x \in G$, and

(c) $x * x^{-1} = e = x^{-1} * x$ for all $x \in G$.

Show how an arbitrary group can be considered as a category.

# Exercise P-1.1.20.2 (Petur)

A monoid $(G, *, e)$ is a set $G$ equipped with a binary operation $*$, a distinguished element $e$ such that

(a) $(x * y) * z = x * (y * z)$ for all $x, y, z \in G$,

(b) $e * x = x = x * e$ for all $x \in G$, and

Show how an arbitrary monoid can be considered as a category.

# Transition systems revisited

- A transition system is a tuple $(S, i, L, Tr)$ with $Tr \subseteq S \times L \times S$. Goal: Externalize this
- A transition system is a graph $(S, Tr)$ with an initial state $i : * \to S$ and a labeling $\lambda : Tr \to L$
- $*$ : the one-element set; $i : * \to S$ picks out one element of $S$
- The category of pointed sets: *comma category* $* \downarrow$ **Set**
- objects: $\begin{tikzcd} * \arrow[d] \\ A \end{tikzcd}$ arrows: $\begin{tikzcd} & * \arrow[dl] \arrow[dr] & \\ A \arrow[rr] & & B \end{tikzcd}$

$\Rightarrow$ objects: sets with a basepoint
arrows: functions which preserve the basepoint

# Transition systems revisited

- Transition system without labels = pointed graph

$\Rightarrow$ want comma category $* \downarrow$ **Graph**

- Turn one-element set $*$ into graph: add degenerate edge

$\Rightarrow$ the "terminal" reflexive graph:

$$* = \quad x \circlearrowright \mathrm{deg}(x)$$

- The comma category of pointed reflexive graphs $* \downarrow$ **RGraph**:
- objects: reflexive graphs with initial state
  arrows: graph homomorphisms which preserve the initial state

$=$ unlabeled transition systems (and functional simulations)

# Transition systems revisited

- A transition system is a pointed reflexive graph $* \xrightarrow{i} (S, Tr)$ together with a labeling $\ell : Tr \to L$.

Need more externalization

- Idea: a set is a graph with one point:

$$\{a, b, c\} = \begin{tikzcd} x \arrow[loop left, "a"] \arrow[loop above right, "b"] \arrow[loop below, "c"] \end{tikzcd}$$

$\Rightarrow$ A transition system is a diagram $* \xrightarrow{i} (S, Tr) \xrightarrow{\ell} (*, L)$ in the category of reflexive graphs.

- Forget about internal structure: $* \xrightarrow{i} T \xrightarrow{\ell} G_L$ (externalization!)

# Transition systems revisited

- A morphism of transition systems $T = (S, i, L, Tr)$, $T' = (S', i', L', Tr')$ is a pair $f = (\sigma, \lambda) : T \to T'$ of functions $\sigma : S \to S'$, $\lambda : L \to L'_\perp$ for which $\sigma(i) = i'$ and

$$(s_1, a, s_2) \in Tr \quad \text{implies} \quad (\sigma(s_1), \lambda(a), \sigma(s_2)) \in Tr'_\perp$$

- Now looks like

$$\begin{tikzcd} & * \arrow[dl, "i"'] \arrow[dr, "i'"] & \\ T \arrow[rr, "\sigma"] \arrow[d, "\ell"'] & & T' \arrow[d, "\ell'"] \\ G_L \arrow[rr] & & G'_L \end{tikzcd}$$

$\Rightarrow$ a diagram in the category of reflexive graphs

- ("Pointed arrow category")

# Functors

- A functor from a category $\mathcal{C}$ to a category $\mathcal{D}$ consists of a function $F$ on objects and a function $F$ on arrows

$$
\begin{tikzcd}
\mathcal{C} & & \mathcal{D} \\
A \arrow[rr, mapsto, "F"] \arrow[d] \arrow[drr, phantom, "f \xmapsto{F} F(f)"] & & F(A) \arrow[d] \\
B \arrow[rr, mapsto, "F"'] & & F(B)
\end{tikzcd}
$$

- for which $F(\mathrm{id}_A) = \mathrm{id}_{F(A)}$
- and $F(g \circ f) = F(g) \circ F(f)$.

- Structure-preserving function between categories.

- $F$ is full $\Leftrightarrow$ surjective on arrows
- $F$ is faithful $\Leftrightarrow$ injective on arrows

# Exercise P-2.1.10.3

Let $M$, $N$ be two monoids (groups; preorders) considered as one-object categories. What are the functors from $M$ to $N$?

# Exercise ML-1.3.4

Prove that there is no functor from groups to Abelian groups which maps each group to its center.

- A group $G$ is Abelian if its operation $*$ is commutative; $x * y = y * x$ for all $x \in G$.
- The center $Z(G)$ of a group $G$ is the set of all elements which commute with all others;

$$Z(G) = \{x \in G \mid \forall y \in G : x * y = y * x\}$$

# Natural transformations

A natural transformation $\eta : F \stackrel{\cdot}{\to} G$ between functors $F, G : \mathcal{C} \to \mathcal{D}$ is a function from $\mathcal{C}$-objects to $\mathcal{D}$-arrows, $A \quad \mapsto \quad \eta_A : F(A) \to G(A)$ such that the diagrams

$$\begin{array}{ccc} F(A) & \xrightarrow{\eta_A} & G(A) \\ {\scriptstyle F(f)}\downarrow & & \downarrow{\scriptstyle G(f)} \\ F(B) & \xrightarrow[\eta_B]{} & G(B) \end{array}$$

commute for all arrows $f : A \to B$ in $\mathcal{C}$.

# Exercise P-2.3.11.2 (Mikkel)

Let $\mathcal{P}$ be a preorder (regarded as a category) and $\mathcal{C}$ a category. Let $S, T : \mathcal{C} \to \mathcal{P}$ be functors. Show that there is a unique natural transformation $\tau : S \stackrel{.}{\to} T$ if and only if $S(C) \leq T(C)$ for all $C \in \mathcal{C}$.

# Adjoint functors

# Adjoint functors

Definition: Functors $F : \mathcal{C} \leftrightarrows \mathcal{D} : G$ are adjoint if there is a natural transformation $\eta : I_{\mathcal{C}} \xrightarrow{\cdot} G \circ F$ such that for each arrow $f : X \to G(Y) \in \mathcal{C}$, there is a *unique* arrow $f^\sharp : F(X) \to Y \in \mathcal{D}$ for which the diagram

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[dr, "f"'] & G(F(X)) \arrow[d, "G(f^\sharp)"] \\ & G(Y) \end{tikzcd}$$

commutes. This is called the *universal property*.

- $F =$ left adjoint, $G =$ right adjoint
- $\eta =$ unit – transformation from identity functor to $G \circ F$

# Example (Pierce 2.4.1-2)

The functor List : **Set** $\to$ **Mon** is left adjoint to the forgetful functor $U$ : **Mon** $\to$ **Set**, with unit $i : I_{\mathbf{Set}} \stackrel{\cdot}{\to} U \circ \mathrm{List}$ given by $i_S(s) = [s]$:

$$
\begin{tikzcd}
S \arrow[r, "i_S"] \arrow[dr, "f"'] & U(\mathrm{List}(S)) \arrow[d, "U(f^\sharp)"'] \arrow[dr, phantom, "\xleftarrow{U}"] & \mathrm{List}(S) \arrow[l, "U"'] \arrow[d, "f^\sharp"] \\
 & U(M) & M \arrow[l, "U"]
\end{tikzcd}
$$

- Example 2.4.2: $length = 1^\sharp$
- Left adjoints to forgetful functors are called free functors
- So List($S$) is the free monoid on $S$

# Example: free groups

For a set $S$, define the free group $F(S)$ on $S$ as follows:

- Let $W$ be the set of finite words $w = w_1 w_2 \dots w_n$, with each $w_i \in S$, or $w_i = s^{-1}$ for some $s \in S$. That's just *syntax*.
- A word $w$ can be reduced if it contains a subword $ss^{-1}$ or $s^{-1}s$. Then $w$ is equivalent to $w$-with-the-subword-removed.
- This defines an equivalence relation on $W$. The free group $F(S)$ is the set of equivalence classes of $W$.
- Example:
  $F(\{a, b\}) = \{\varepsilon, a, b, ab, ab^{-1}, a^{-1}b, a^{-1}b^{-1}, ba, ba^{-1}, \dots\}$

This is functorial, and $F$ is left adjoint to the forgetful functor $U$ : **Group** $\to$ **Set**.

- Universal property:

$$\begin{tikzcd} S \arrow[r, "i"] \arrow[dr, "f"'] & U(F(S)) \arrow[d, "U(f^\sharp)"] \\ & U(G) \end{tikzcd}$$

# Alternative characterizations of adjoints

Functors $F : \mathcal{C} \leftrightarrows \mathcal{D} : G$ are adjoint if

there is a unit $\eta : I_{\mathcal{C}} \xrightarrow{\cdot} G \circ F$ such that for each arrow $f : X \to G(Y) \in \mathcal{C}$, there is a *unique* arrow $f^\sharp : F(X) \to Y \in \mathcal{D}$ for which the diagram

$$\begin{tikzcd} X \arrow[r, "\eta_X"] \arrow[dr, "f"'] & G(F(X)) \arrow[d, "G(f^\sharp)"] \\ & G(Y) \end{tikzcd}$$

commutes.

there is a co-unit $\varepsilon : F \circ G \xrightarrow{\cdot} I_{\mathcal{D}}$ such that for each arrow $g : F(X) \to Y \in \mathcal{D}$, there is a *unique* arrow $g^* : X \to G(Y) \in \mathcal{C}$ for which the diagram

$$\begin{tikzcd} F(G(Y)) \arrow[r, "\varepsilon_Y"] & Y \\ F(X) \arrow[u, "F(g^*)"] \arrow[ur, "g"'] & \end{tikzcd}$$

commutes.

- For $F$ : **Set** $\leftrightarrows$ **Mon** : $G$ and $F$ : **Set** $\leftrightarrows$ **Group** : $G$, $\varepsilon_Y([s_1, s_2, \ldots, s_n]) = s_1 * s_2 * \cdots * s_n$.

# Example RB-6.3.1: floor and ceiling

- $\mathbb{Z}$ and $\mathbb{R}$ are partial orders $\Rightarrow$ categories
- The forgetful functor $U : \mathbb{Z} \to \mathbb{R}$ has
- left adjoint $\mathrm{Ceil} : \mathbb{R} \to \mathbb{Z}$ ("smallest integer not smaller than") and
- right adjoint $\mathrm{Floor} : \mathbb{R} \to \mathbb{Z}$ ("greatest integer not greater than")
- Adjunction $\mathrm{Ceil} : \mathbb{R} \leftrightarrows \mathbb{Z} : U$ has unit $\eta_X = (X \leq \mathrm{Ceil}(X))$ and co-unit $\varepsilon_Y = (\mathit{Ceil}(Y) = Y)$ (an *iso*!)
- Adjunction $U : \mathbb{Z} \leftrightarrows \mathbb{R} : \mathrm{Floor}$ has unit $\eta_X = (X = \mathrm{Floor}(X))$ (an *iso*!) and co-unit $\varepsilon_Y = (\mathrm{Floor}(Y) \leq Y)$

# Example RB-6.3.4(1): free category on a graph

The free category $F(G)$ on a graph $G = (V, E)$ has

- as objects all points in $V$
- as arrows all paths in $G$: all sequences $(e_1, e_2, \ldots, e_n)$ of edges in $E$ with $tgt(e_i) = src(e_{i+1})$
- and composition of arrows is concatenation of paths

$\Rightarrow$ left adjoint to the forgetful functor: $F : \mathbf{Graph} \leftrightarrows \mathbf{Cat} : U$

– Like the adjunction **Set** $\leftrightarrows$ **Mon**, but *many-object*!

# Special types of adjoints

An adjunction $F : \mathcal{C} \leftrightarrows \mathcal{D} : G$ is

- a reflection if $G$ is fully faithful
  $\Leftrightarrow$ all arrows $\varepsilon_Y : F(G(Y)) \to Y$ are isos
- a co-reflection if $F$ is fully faithful
  $\Leftrightarrow$ all arrows $\eta_X : X \to G(F(X))$ are isos
- an adjoint equivalence if it is both a reflection and a co-reflection

# Transition systems, synchronization trees, languages

# Synchronization trees

- Recall: A transition system is a tuple $(S, i, L, Tr)$ with $Tr \subseteq S \times L \times S$. (Back to the old notation!)
- A synchronization tree is a transition system in which there is precisely one path from $i$ to any state $s \in S$:
  - every state is reachable
  - acyclic
  - no joins
- Recall: A morphism of transition systems is a pair $(\sigma, \lambda) : (S, i, L, Tr) \to (S', i', L', Tr')$ of functions $\sigma : S \to S'$, $\lambda : L \to L'_\perp$ for which $\sigma(i) = i'$ and

$$(s_1, a, s_2) \in Tr \quad \text{implies} \quad (\sigma(s_1), \lambda(a), \sigma(s_2)) \in Tr'_\perp$$

- **T**: category of transition systems
- **S**: fully faithful subcategory of synchronization trees

# Synchronization trees

- $i : \mathbf{S} \to \mathbf{T}$ is fully faithful
- Right adjoint: unfolding:
- Given transition system $T = (S, i, L, Tr)$, define synchronization tree $ts(T) = (S', i', L, Tr')$ (same labels) by
  - $S' =$ set of all paths in $T$
  - $i' = ()$ (empty path)
  - $Tr' =$ one-step continuations of paths:

$$Tr' = \{((s_1, \dots, s_k), a, (s_1, \dots, s_k, s_{k+1}) \mid (s_k, a, s_{k+1}) \in Tr\}$$

- Co-unit morphisms $\varepsilon_T : i(ts(T)) \to T$ given as $\varepsilon_T = (\varphi, \mathrm{id}_L)$, with

$$\varphi(()) = i \qquad \varphi(s_1, \dots, s_n) = s_n$$

- Universal property:

$$\begin{tikzcd} i(ts(T)) \arrow[r, "\varepsilon_T"] & T \\ i(Y) \arrow[u, "i(f^*)"] \arrow[ur, "f"'] & \end{tikzcd}$$

# Synchronization trees

$\Rightarrow$ co-reflection $i : \mathbf{S} \leftrightarrows \mathbf{T} : ts$

$\Rightarrow$ all unit morphisms are isos.

- That is, for all synchronization trees $Y$, the morphism $\eta_Y : Y \to ts(i(Y))$ is an iso.
- Any synchronization tree is isomorphic to its unfolding.

# Languages

- A language over a labeling $L$ is a pair $(H, L)$ with $H \subseteq L^*$ prefix-closed: $\forall s \in L^* \, \forall a \in L : sa \in H \Rightarrow s \in H$
- Morphisms of languages $(H, L) \to (H', L')$: partial functions $\lambda : L \to L'_\perp$ for which $\lambda^*(w) \in H'$ for all $w \in H$

$\Rightarrow$ category of languages **L**

- The language of a transition system $T = (S, i, L, Tr)$: usual stuff: $tl(T) = (H, L)$ with

$$H = \{a_1 a_2 \dots a_n \mid \exists \text{ path } i \xrightarrow{a_1} s_1 \xrightarrow{a_2} \cdots \xrightarrow{a_n} s_n \text{ in } T\}$$

- Extend to functor $tl : \mathbf{T} \to \mathbf{L}$ by $sl(\sigma, \lambda) = \lambda$
- Composition gives functor $sl = tl \circ i : \mathbf{S} \to \mathbf{T} \to \mathbf{L}$

# Languages

- Languages as synchronization trees: Given language $(H, L)$, define $ls(H, L) = (H, \varepsilon, L, Tr)$ with $Tr = \{(h, a, ha) \mid ha \in H\}$
- Extend to functor $ls : \mathbf{L} \to \mathbf{S}$ by $ls(\lambda) = (\lambda^*{\upharpoonright}_H, \lambda)$
  (restriction of $\lambda^*$ to $H$)
- $sl : \mathbf{S} \leftrightarrows \mathbf{L} : ls$ is an adjunction:
- Co-unit morphisms $\varepsilon_{(H,L)} : sl(ls(H, L)) \to (H, L)$ are identities
- Universal property:

$$\begin{tikzcd} sl(ls(H,L)) \arrow[r, "\mathrm{id}"] & (H,L) \\ sl(Y) \arrow[u, "sl(\lambda^\sharp)"] \arrow[ur, "\lambda"'] & \end{tikzcd}$$

$\Rightarrow$ $sl : \mathbf{S} \leftrightarrows \mathbf{L} : ls$ is a reflection

# Conclusion

- Co-reflection $i : \mathbf{S} \leftrightarrows \mathbf{T} : ts$
- Reflection $sl : \mathbf{S} \leftrightarrows \mathbf{L} : ls$
- But the composed functors $i \circ ls : \mathbf{L} \to \mathbf{T}$, $\mathbf{L} \leftarrow \mathbf{T} : sl \circ ts$ are not even adjoint!