Category Theory and Functional Programming

Day 2

7 October 2009

Categories, functors, natural transformations



Graphs vs. categories Exercise P-1.1.20.2 Transition systems revisited Functors Exercise P-2.1.10.3 Exercise ML-1.3.4 Natural transformations Exercise P-2.3.11.2



(Mikkel)

- Graphs
 - Set of points V
 - Set of edges E
 - For each edge e ∈ E, a source src(e) ∈ V and a target tgt(e) ∈ V
 - (Write $e: x \rightarrow y$ 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 → V, tgt : E → V

Reflexive graphs

- Set of points V
- Set of edges E
- For each edge e ∈ E, a source src(e) ∈ V and a target tgt(e) ∈ V
- (Write $e: x \rightarrow y$ 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: $V, E, src : E \rightarrow V, tgt : E \rightarrow V, deg : V \rightarrow E$

Categories

- Set of points V
- Set of edges E
- For each edge e ∈ E, a source src(e) ∈ V and a target tgt(e) ∈ V
- (Write $e: x \rightarrow y$ 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: e₃ ∘ (e₂ ∘ e₁) = (e₃ ∘ e₂) ∘ e₁ whenever these are defined,
- and identities: for all edges e : x → y, e ∘ deg(x) = e and deg(y) ∘ 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 C_0
- Set of arrows C_1
- For each arrow f ∈ C₁, a domain dom(f) ∈ C₀ and a co-domain cod(f) ∈ C₀
- (Write $f : A \rightarrow B$ if dom(f) = A and cod(f) = B)
- For each object $A \in C_0$, an identity arrow $id_A \in C_0$
- 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: f₃ ∘ (f₂ ∘ f₁) = (f₃ ∘ f₂) ∘ f₁ whenever these are defined,
- and identities: for all arrows $f : A \rightarrow B$, $f \circ id_A = f$ and $id_B \circ f = f$.
- That's all folks: $C_0, C_1, dom, cod : C_1 \rightarrow C_0, id : C_0 \rightarrow C_1, \circ : C_1 \times_{C_0} C_1 \rightarrow C_1$

Exercise P-1.1.20.2



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



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 : * → S picks out one element of S
- The category of pointed sets: comma category * 1 Set



⇒ 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:

$$* = x \operatorname{deg}(x)$$

- The comma category of pointed reflexive graphs * \ 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:

$$\{a,b,c\} = a \begin{pmatrix} c \\ x \\ c \end{pmatrix}^{b}$$

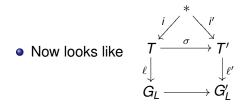
- ⇒ A transition system is a diagram $* \xrightarrow{i} (S, Tr) \xrightarrow{\ell} (*, L)$ in the category of reflexive graphs.
 - Forget about internal structure: * → T → G_L (externalization!)

Graphs vs. categories Exercise Transition systems Functors Exercises Natural transformations Exercise

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

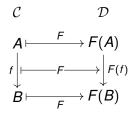
 $(s_1, a, s_2) \in Tr$ implies $(\sigma(s_1), \lambda(a), \sigma(s_2)) \in Tr'_{\perp}$



- \Rightarrow a diagram in the category of reflexive graphs
 - ("Pointed arrow category")



• A functor from a category C to a category D consists of a function F on objects and a function F on arrows



- for which $F(id_A) = id_{F(A)}$
- 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 ⇔ 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 ∈ 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 $\eta : F \rightarrow G$ between functors $F, G : C \rightarrow D$ is a function from *C*-objects to *D*-arrows, $A \mapsto \eta_A : F(A) \rightarrow G(A)$ such that the diagrams

commute for all arrows $f : A \rightarrow B$ in C.

Exercise P-2.3.11.2



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 $\tau : S \to T$ if and only if $S(C) \leq T(C)$ for all $C \in \mathcal{C}$.

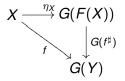
Adjoint functors



Definition Example (Pierce 2.4.1-2) Example: free groups Co-units Examples

Special types of adjoints

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



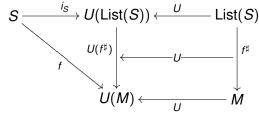
commutes. This is called the universal property.

• F = left adjoint, G = right adjoint

• $\eta = \text{unit} - \text{transformation from identity functor to } G \circ F$

Definition Examples Co-units Examples Special types of adjoints
Example (Pierce 2.4.1-2)

The functor List : **Set** \rightarrow **Mon** is left adjoint to the forgetful functor U : **Mon** \rightarrow **Set**, with unit $i : I_{\text{Set}} \rightarrow U \circ \text{List}$ given by $i_{S}(s) = [s]$:



Example 2.4.2: *length* = 1[#]

- Left adjoints to forgetful functors are called free functors
- So List(S) is the free monoid on S

 Definition
 Examples
 Co-units
 Examples
 Special types of adjoints

 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** \rightarrow **Set**.

• Universal property:

$$S \xrightarrow{i} U(F(S))$$

$$\downarrow U(f^{\sharp})$$

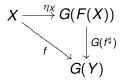
$$U(G)$$

Alternative characterizations of adjoints

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

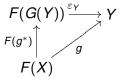
there is a unit $\eta : I_{\mathcal{C}} \to G \circ F$ such that for each arrow $f : X \to G(Y) \in \mathcal{C}$, there is a *unique* arrow $f^{\sharp} : F(X) \to Y \in \mathcal{D}$ for which

the diagram



commutes.

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



commutes.

• For F : Set \leftrightarrows Mon : G and F : Set \leftrightarrows Group : G, $\varepsilon_{\gamma}([s_1, s_2, \dots, s_n]) = s_1 * s_2 * \dots * s_n$.

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 Ceil : $\mathbb{R} \to \mathbb{Z}$ ("smallest integer not smaller than") and
- right adjoint Floor : $\mathbb{R} \to \mathbb{Z}$ ("greatest integer not greater than")
- Adjunction Ceil : $\mathbb{R} \cong \mathbb{Z} : U$ has unit $\eta_X = (X \le \text{Ceil}(X))$ and co-unit $\varepsilon_Y = (Ceil(Y) = Y)$ (an *iso*!)
- Adjunction $U : \mathbb{Z} \hookrightarrow \mathbb{R}$: Floor has unit $\eta_X = (X = \text{Floor}(X))$ (an *iso*!) and co-unit $\varepsilon_Y = (\text{Floor}(Y) \le Y)$

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 (e₁, e₂,..., 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 : Graph \leftrightarrows Cat : U
- Like the adjunction **Set** \leftrightarrows **Mon**, but *many-object*!

Co-units

Examples

Special types of adjoints

Special types of adjoints

An adjunction $F : C \leftrightarrows D : G$ is

- a reflection if G is fully faithful
 ⇔ all arrows ε_Y : F(G(Y)) → 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 Languages Conclusion

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 ∈ S:
 - every state is reachable
 - acyclic
 - no joins
- Recall: A morphism of transition systems is a pair $(\sigma, \lambda) : (S, i, L, Tr) \rightarrow (S', i', L', Tr')$ of functions $\sigma : S \rightarrow S'$, $\lambda : L \rightarrow L'_{\perp}$ for which $\sigma(i) = i'$ and

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

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

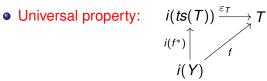
Synchronization trees

- $i: \mathbf{S} \to \mathbf{T}$ 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:

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

• Co-unit morphisms $\varepsilon_T : i(ts(T)) \to T$ given as $\varepsilon_{T} = (\varphi, \mathsf{id}_{I}), \mathsf{with}$

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



Synchronization trees

- \Rightarrow co-reflection *i* : **S** \leftrightarrows **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* ⊆ *L** prefix-closed: ∀s ∈ L* ∀a ∈ L : sa ∈ H ⇒ s ∈ H
- Morphisms of languages (H, L) → (H', L'): partial functions λ : L → L'_⊥ for which λ^{*}(w) ∈ H' for all w ∈ H
- \Rightarrow category of languages L
 - The language of a transition system T = (S, i, L, Tr): usual stuff: tl(T) = (H, L) with

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

- Extend to functor $t : \mathbf{T} \to \mathbf{L}$ by $s (\sigma, \lambda) = \lambda$
- 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 $ls(\lambda) = (\lambda_{1H}^*, \lambda)$ (restriction of λ^* to H)
- $sl : S \leftrightarrows L : ls$ is an adjunction:
- Co-unit morphisms ε_(H,L) : sl(ls(H, L)) → (H, L) are identities

• Universal property: $sl(ls(H, L)) \xrightarrow{id} (H, L)$ $sl(\lambda^{\sharp}) \uparrow \qquad \lambda$ sl(Y)

 \Rightarrow *sl* : **S** \leftrightarrows **L** : *ls* is a reflection

Conclusion

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