Incrementality as Functor

Modeling Incremental Processes with Monoidal Categories

Dan Shiebler Alexis Toumi

University of Oxford

Category Theory Octoberfest, October 2019

Background: Categorical Grammars

Parsing Sentences with Formal Grammars

- Q: What is a grammatical sentence?
  A: Specify a grammar: i.e. a subset $L \subseteq \Sigma^\star$, where Σ is a finite set of characters (an alphabet) or words (a vocabulary).
- We have different classes of grammars, the basic trade-off being *complexity* vs *expressivity*.

Example

Chomsky hierarchy:

1. recursively enumerable (Turing machines)
2. context-sensitive (linear-bounded automaton)
3. context-free (push-down automaton)
4. regular (finite-state automaton)

Pregroups/Protogroups as Algebraic Structures

- **Monoid** Closure, Associativity, Identity
- **Group** Closure, Associativity, Identity, Invertibility
- **Pregroups and Protogroup** Sort of "in-between"
  - Apply a partial ordering
  - Replace invertibility with a left/right adjoint

Pregroups/Protogroups as Algebraic Structures

Protogroups $(P, \cdot, 1, \leq, -^l, -^r)$

$$p^l \cdot p \leq 1$$
$$p \cdot p^r \leq 1$$

Pregroups $(P, \cdot, 1, \leq, -^l, -^r)$

$$p^l \cdot p \leq 1 \leq p \cdot p^l$$
$$p \cdot p^r \leq 1 \leq p^r \cdot p$$

Pregroups/Protogroups for Language

Parts of speech (types) are elements in the pregroup/protogroup:

n :	noun
s :	declarative statement (sentence)
j :	infinitive of the verb
σ :	glueing type

Words in a vocabulary map can be assigned to parts of speech:

John	likes	Mary
n	$(n^r s n^l)$	n

Pregroups/Protogroups for Language

We call a string of words **grammatical** if the corresponding string of types is $\leq$ the sentence type (s)

John	likes	Mary
n	$(n^r s n^l)$	n

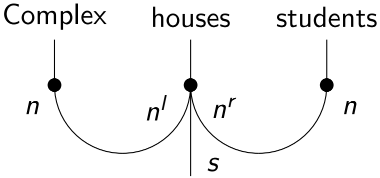
$$nn^r sn^l n \leq nn^r s \leq s$$

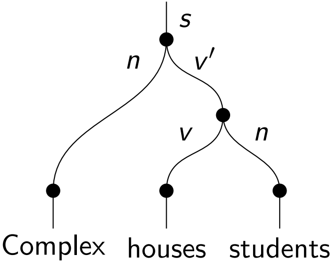
Pregroups/Protogroups as Monoidal Categories

- Types are objects
- Strings of types are tensor products of objects
- Arrows $s \to t$ are proofs that $s \leq t$ in the free pregroup.

$$p^l \otimes p \to 1$$
$$p \otimes p^r \to 1$$
$$n \otimes n^r \otimes s \otimes n^l \otimes n \to s$$

Syntax Trees and Pregroup Reductions are String Diagrams





Monoidal Grammars

Monoidal Signatures

Definition

A **Monoidal Signature** is a tuple $\Sigma = (\Sigma_0, \Sigma_1, \mathrm{dom}, \mathrm{cod})$ where Σ_0 and Σ_1 are sets of generating *objects* and *arrows* respectively, and $\mathrm{dom}, \mathrm{cod} : \Sigma_1 \to \Sigma_0^{\star}$ are pairs of functions called *domain* and *codomain*.

Definition

Free monoidal categories are the objects in the image of the free functor from **MonSig** to **MonCat**

Monoidal Presentations

Definition

A *presentation* for a monoidal category is given by a monoidal signature Σ and a set of relations $R \subseteq \coprod_{u,t \in \Sigma_0^\star} \mathbf{C}_{\mathbf{\Sigma}}(\mathbf{u}, \mathbf{t}) \times \mathbf{C}_{\mathbf{\Sigma}}(\mathbf{u}, \mathbf{t})$ between parallel arrows of the associated free monoidal category.

Definition

MonPres is the category of monoidal presentations and monoidal presentation homomorphisms (monoidal signature homomorphisms that commute nicely with the relations in R)

Monoidal Grammar

Definition

A *monoidal grammar* is a tuple $G = (V, \Sigma, R, s)$ where V is a finite vocabulary and (Σ, R) is a finite presentation with $V \subseteq \Sigma_0$ and $s \in \Sigma_0^\star$.

Monoidal grammars form a subcategory of $(V \cup \{s\})^*/\mathbf{MonPres}$

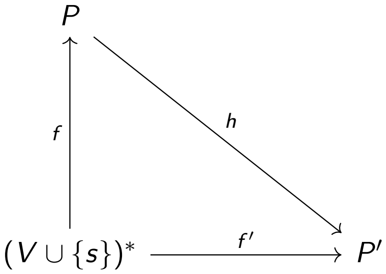
$$(V \cup \{s\})^*$$

$$\downarrow f$$

$$(\Sigma_0, \quad \Sigma_1, \quad dom, \quad cod, R)$$

Monoidal Grammar

- Objects are pairs (f, P) where f picks out the word objects and sentence token in the presentation P
- Morphisms are presentation homomorphisms (functors in the generated categories) $h : P \to P'$ such that:



Example: Pregroup Grammars

- $V = \{w_1, w_2, w_3, ...\}$
- $\Sigma_0 = V \cup \{s, n, j, ...\} \cup \{s^r, n^r, j^r, ...\} \cup \{s^l, n^l, j^l, ...\}$
- $\Sigma_1 =$

$$\{w_1 \to n, ...\} \cup \{cup_n : n^l \otimes n \to 1, ...\} \cup \{cap_n : 1 \to n \otimes n^l, ...\} \cup ...$$

- $R =$ Snake equations

Parse States and Parsings

Definition

A **parse state** for the monoidal grammar (V, Σ, R, s) is an arrow in the generated category of (V, Σ, R, s) of the form $w_1 \otimes w_2 \otimes ... \otimes w_n \to o$

Definition

A **parsing** is a parse state $w_1 \otimes w_2 \otimes ... \otimes w_n \to s$

Definition

The **language** of a monoidal grammar is the set of all $w_1 \otimes w_2 \otimes ... \otimes w_n$ that have at least one parsing.

Incremental Monoidal Grammar

Speech

Monoidal grammars operate on a fixed string of words. In speech, words are introduced one at a time. How can we reconcile this?

Parse States are Understanding

- A parse state $w_1 \otimes w_2 \otimes ... \otimes w_n \to o$ represents the syntactic understanding of $w_1 \otimes w_2 \otimes ... \otimes w_n$
- A new word w should evolve this understanding

New Word = New Parse States

Given (f, C) generated by the monoidal grammar $G = (V, \Sigma, R, s)$, a new word $w \in V$ defines an endofunctor over C:

$$W_w : (f, C) \to (f, C)$$
$$W_w(o) = o \otimes w$$
$$W_w(a) = a \otimes id_w$$

Hence, we get an action of the free monoid $V^\star$ on the category of endofunctors.

New Word = New Parse States

$W_w(a) = a \otimes id_w$ is not enough. Ideally we can capture all of the ways understanding can evolve in the face of a new word.

New Word = New Parse States

W_w^* maps the parse state a to all of parse states that factor into $a \otimes id_w$

$$W_w(a) = a \otimes id_w$$
$$W_w^*(a) = \{a' \circ W_w(a) \mid a' \in Ar(C), dom(a') = (cod(a) \otimes w)\}$$

W_w^* captures how the parsing system evolves when a new word is introduced.

Monoidal Grammars as Automata Coalgebraically

Transition Function

Over the vocabulary V, set of states X, and start state "", a deterministic automaton is:

$$\Delta : X \times V \rightarrow X$$
$$accept : X \rightarrow \mathbb{B}$$

A nondeterministic automaton is:

$$\Delta : X \times V \rightarrow \mathcal{P}(X)$$
$$accept : X \rightarrow \mathbb{B}$$

W^* is a Transition Function

Remember W^*, which maps the word w and the parse state a to all of parse states that factor into $a \otimes id_w$?

$$W^* : Ar(C) \times V \to \mathcal{P}(Ar(C))$$

W^* looks like a nondeterministic automata transition function! Can we formalize this?

Coalgebra

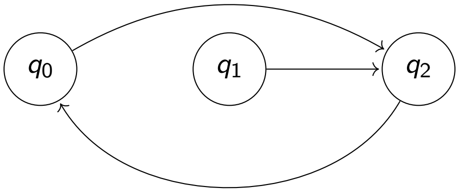
- A coalgebra of a functor F is a pair (f, X) where $f : X \to FX$.
- Coalgebras over **Set** endofunctors can model an array of dynamical systems

Coalgebra: Example

Say we define:

$$F : \mathbf{Set} \to \mathbf{Set}$$
$$FX = X$$

Then the pair $(f, \{q_0, q_1, q_2\})$ where f is defined below is a coalgebra of F:



Coalgebra: Automata

- Deterministic automata are coalgebras of $FX = \mathbb{B} \times X^V$
- Non-deterministic automata are coalgebras of $FX = \mathbb{B} \times \mathcal{P}(X)^V$

Incremental Functor

W^* is uniquely defined by a monoidal grammar, so we can now rephrase our informal statement:

We can define a functor, $I_{\mathcal{P}}$, from the category of monoidal grammars to coalgebras of $\mathbb{B} \times \mathcal{P}(Ar(C))^V$

Incremental Functor

The functor $I_{\mathcal{P}}$:

- Objects: Monoidal grammars are mapped to automata where the transition function is defined by W^*
- Morphisms: Functors between monoidal grammars are mapped to coalgebra homomorphisms

Bisimulation

A bisimulation between automata is a relation that describes how each automata can simulate the other. Bisimulations correspond to coalgebra homomorphisms, so we can state the following:

If two monoidal grammar categories have functors between them, then the corresponding automata are bisimulatable

Future Work

Probabilistic Incremental Functor

- A "weighted monoidal grammar" is a monoidal grammar equipped with a functor to $\mathbb{R}$
- There is a functor between the category of weighted monoidal grammars and coalgebras of the **Set** endofunctor $\mathbb{R} \times \mathcal{V}(X)^V$, where $\mathcal{V}$ is the **valuation monad**

Incremental Semantics

- Functor from syntax to semantics (e.g. vector spaces, booleans)
- Apply semantics to parse states to study the evolution of semantics over time

Steve Awodey.
Category Theory.
Ebsco Publishing, May 2006.

Antonin Delpeuch and Jamie Vicary.
Normalization for planar string diagrams and a quadratic equivalence algorithm.
arXiv:1804.07832 [cs], April 2018.

André Joyal and Ross Street.
Planar diagrams and tensor algebra.
Unpublished manuscript, available from Ross Street's website, 1988.

André Joyal and Ross Street.
The geometry of tensor calculus, I.
Advances in Mathematics, 88(1):55–112, July 1991.

A Markov.
On certain insoluble problems concerning matrices.
In *Doklady Akad. Nauk SSSR*, volume 57, pages 539–542, 1947.

R Oehrle.
A parsing algorithm for pregroup grammars.
Proceedings of Categorial Grammars 2004, pages 59–75, January 2004.

Matthew Purver, Ronnie Cann, and Ruth Kempson.
Grammars as Parsers: Meeting the Dialogue Challenge.
Research on Language and Computation, 4(2):289–326, October 2006.

Emil L. Post.
Recursive Unsolvability of a problem of Thue.
Journal of Symbolic Logic, 12(1):1–11, March 1947.

James F. Power.
Thue's 1914 paper: A translation.
arXiv:1308.5858 [cs], August 2013.

Anne Preller.
Linear Processing with Pregroups.
Studia Logica: An International Journal for Symbolic Logic, 87(2/3):171–197, 2007.

Alexandra Silva, Filippo Bonchi, Marcello Bonsangue, and Jan Rutten.
Generalizing determinization from automata to coalgebras.
Logical Methods in Computer Science, 9(1):9, March 2013.

P. Selinger.
A Survey of Graphical Languages for Monoidal Categories.
New Structures for Physics, pages 289–355, 2010.

Axel Thue.
Probleme Über Veränderungen von Zeichenreihen Nach Gegebenen Regeln.
na, 1914.