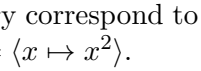
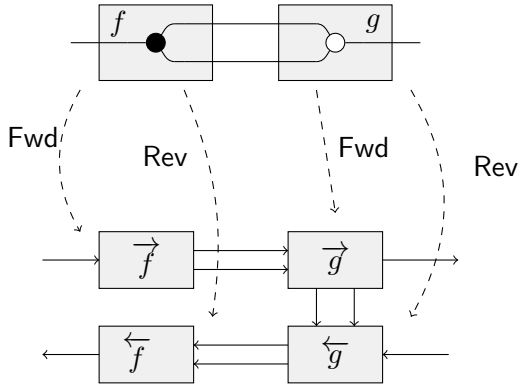
Data-Parallel Differentiation by Optic Composition

1 Reverse Derivatives by Optic Composition

We give data-parallel datastructures and algorithms for computing symbolic reverse derivatives [3] by optic composition [4]. More precisely, given a symmetric monoidal category presented by generators and operations, as well as a choice of derivative for each operation, we give an algorithm to transform a morphism into its reverse derivative [3]. This algorithm is *data-parallel* [10]: it runs in time logarithmic in the size of the morphism on a PRAM machine, and linear time on a sequential machine.

Consider the following example to illustrate the approach. Choose a 'base' category of polynomial circuits [9], which include basic arithmetic operations copying  , addition  , and multiplication  . Morphisms in this category correspond to tuples of polynomials. For example, the composition  . represents the 1-tuple $p = \langle x \mapsto x^2 \rangle$.

Abstractly, one can picture the process of mapping such a morphism to a composition of optics as below.

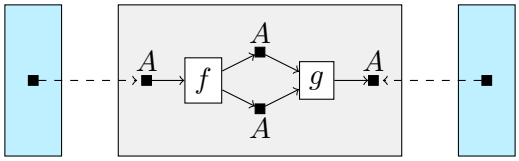


Here, each operation h is sent to its 'forward' $\overrightarrow{h}$ and 'backward' $\overleftarrow{h}$ components, which are connected by a 'residual' object M_h. For $f =$  the residual $M_f = I$ is the unit object depicted as empty space between $\overrightarrow{f}$ and $\overleftarrow{f}$, whereas the residual of $g =$  is $M_g = A \times A$.

Notice that wires are oriented: while the forward components are connected left-to-right, the *backwards* components are connected right-to-left. Essentially, this says that the gradients of $\overrightarrow{g}$ are used to compute the gradients of $\overrightarrow{f}$. However, to make this more precise it is necessary to describe the datastructure we use to represent morphisms: open hypergraphs.

2 Open Hypergraphs

An open hypergraph is a cospan of hypergraphs [2] whose 'feet' are discrete. (They are *structured cospans* [10, 1].) The morphism can be depicted as the following cospan of hypergraphs:

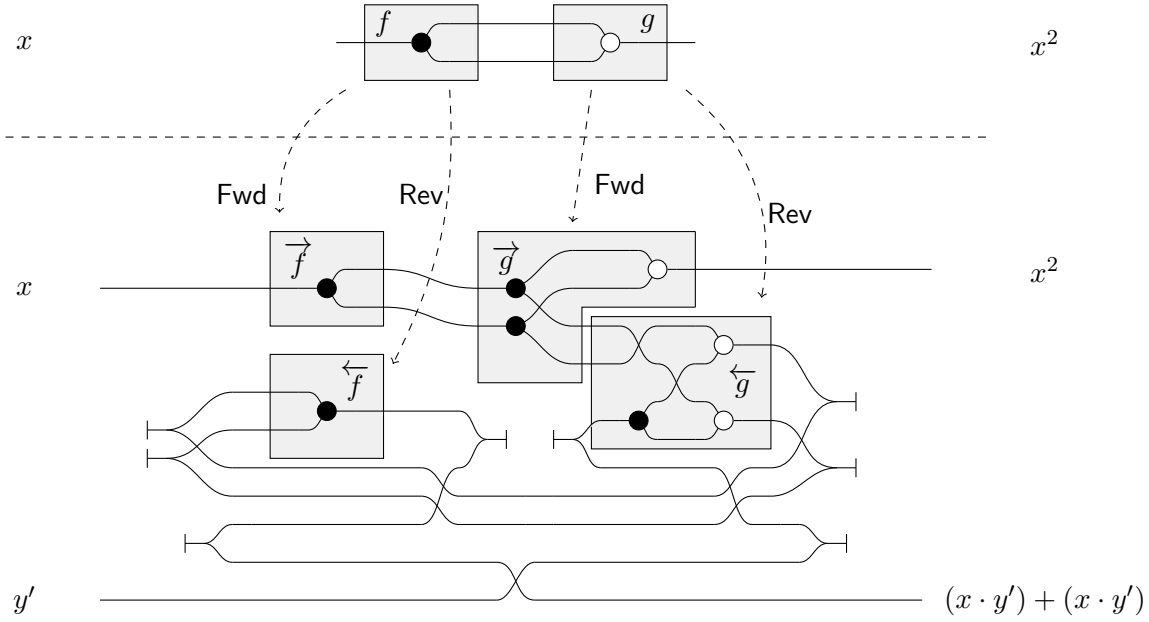


In this depiction, the blue boxes are discrete hypergraphs containing only nodes (■), while the central grey box is a hypergraph with both nodes and hyperedges (□). Both are labeled: nodes with generating objects A, and hyperedges with operations f, g. In addition, note that hyperedges have an *ordered list* of input and output nodes.

Without additional restrictions on the hypergraph structure (specifically the condition of monogamous acyclicity described in [2]), open hypergraphs in fact correspond to morphisms in the category presented by the given generators and operations extended with Special Frobenius structure, whose operations are depicted below:

(1)

The compositions $\vdash \fatsemi$  and  $\fatsemi \dashv$ form 'cup' and 'cap' morphisms in the category, which effectively allows for 'bending' wires around. Consequently, morphisms can be composed as 'optics' without the need for explicitly directed wires. Returning to our earlier example, the process of mapping to a composition of optics is shown below:



Notice that the tangle of wires at the bottom of the diagram captures the 'reverse' gradient flow: outputs of $\overleftarrow{g}$ flow into inputs of $\overleftarrow{f}$. Note also that the apparent complexity is only an artifact of the *connectivity* of the diagram: each wire in the diagram corresponds to a single hypernode and the particular decomposition into Frobenius operations depicted here adds no additional overhead in the representation.

3 Summary

Our approach means that a symmetric monoidal category can be built in a modular fashion by choosing or extending a set of primitives and their optics; one then obtains data-parallel machinery for optic composition for free. Reverse derivative category structure in particular is guaranteed to be preserved by [11, Theorem 3.1]. The algorithms we discuss are described in detail in [10], and we provide two implementations: Yarrow [8] and Open Hypergraphs [7]. Moreover, we have implemented a deep learning framework using the latter called CATGRAD [6] which is commercially supported by Hellas.AI.

We also aim to explore three further applications in future. First, we will exploit the data-parallel nature of our algorithms for gradient-based learning over large programs learned by evolutionary methods. Second, since optics model a wide variety of phenomena including Bayesian learning [5], we hope to apply our algorithm beyond the domain of automatic differentiation. Thirdly, since the setting of symmetric monoidal categories is so general, we hope to apply our framework to more examples of categories such as circuits–and in particular, those categories which have a notion of iteration via the *trace*.

References

[1] John C. Baez and Kenny Courser. Structured cospans. 2019. doi: 10.48550/ARXIV.1911.04630. URL https://arxiv.org/abs/1911.04630.

[2] Filippo Bonchi, Fabio Gadducci, Aleks Kissinger, Paweł Sobociński, and Fabio Zanasi. Rewriting modulo symmetric monoidal structure. In *Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science*. ACM, jul 2016. doi: 10.1145/2933575.2935316. URL https://doi.org/10.1145%2F2933575.2935316.

[3] Robin Cockett, Geoffrey Cruttwell, Jonathan Gallagher, Jean-Simon Pacaud Lemay, Benjamin MacAdam, Gordon Plotkin, and Dorette Pronk. Reverse derivative categories, 2019.

[4] G. S. H. Cruttwell, Bruno Gavranović, Neil Ghani, Paul Wilson, and Fabio Zanasi. Categorical foundations of gradient-based learning, 2021. URL https://arxiv.org/abs/2103.01931.

[5] Toby St. Clere Smithe. Bayesian updates compose optically, 2020. URL https://arxiv.org/abs/2006.01631.

[6] Paul Wilson. Catgrad, . URL https://github.com/statusfailed/catgrad.

[7] Paul Wilson. open-hypergraphs, . URL https://github.com/statusfailed/open-hypergraphs.

[8] Paul Wilson. yarrow-diagrams, . URL https://github.com/yarrow-id/diagrams.

[9] Paul Wilson and Fabio Zanasi. Categories of differentiable polynomial circuits for machine learning, 2022. URL https://arxiv.org/abs/2203.06430.

[10] Paul Wilson and Fabio Zanasi. Data-parallel algorithms for string diagrams, 2023.

[11] Paul Wilson and Fabio Zanasi. An axiomatic approach to differentiation of polynomial circuits. *Journal of Logical and Algebraic Methods in Programming*, 135:100892, 2023. ISSN 2352-2208. doi: https://doi.org/10.1016/j.jlamp.2023.100892. URL https://www.sciencedirect.com/science/article/pii/S2352220823000469.