Higher-order neuromorphic computations with linear streams (extended abstract)

Michael Bukatin[1]

Dataflow Matrix Machines project[2]

1. One of the ways to reconcile Scott domains and vector spaces is via the route of making + an invertible operation. Typically, this requires balancing partially defined elements with overdefined elements, so that one could get the exactly defined zero element on addition. E.g. in interval arithmetic, one introduces pseudosegments $[a, b]$ with the contradictory property $b < a$ (the first mention known to the author is [8]). For probability theory, this corresponds to admitting negative values of probabilities (the first mention known to the author is [9]). This leads to bicontinuous domains [7], and to the rich mathematical theory linking partial inconsistency and the structure of vector spaces on various domains for denotational semantics. We call this mathematical theory *partial inconsistency landscape*. Its focal points include various forms of negative measure (negative length and distance, negative probability and signed measures, negative membership and signed multisets), bilattices and lattice-ordered groups with the respective $x = (x \wedge \bot) \sqcup (x \vee \bot)$ and $x = (x \wedge 0) + (x \vee 0)$ identities, bitopology, non-monotonic and anti-monotonic inferences, computations with involutions, and more (see [5] and references therein). This rich mathematical landscape suggests that programming with vector-like elements might be sufficiently expressive to enable general-purpose programming.

2. The essence of neural model of computations is that linear and non-linear computations are interleaved. So, the natural degree of generality for neuromorphic computations is to work not with streams of numbers, but with arbitrary streams supporting the notion of linear combination of several streams (**linear streams**). There are various kinds of linear streams, including streams of numbers, sparse vectors and sparse tensors (both of finite and infinite dimension), streams of functions and distributions. We found streams of V-values (flexible tensors based on tree-shaped indices) to be of particular use. V-values can be viewed as finite linear combinations of finite strings, finite prefix trees with numeric leaves, or sparse tensors of "mixed rank" with finite number of non-zero elements. The space of V-values can be also thought of as a solution of a vector space isomorphism, $V \cong \mathbb{R} \oplus (L \to V)$, where L is a countable alphabet [4].

3. We call neural machines working with arbitrary linear streams **dataflow matrix machines (DMMs)**. One takes a finite or countable collection of neuron types, and a countable number of copies of a neuron of each type. Then a DMM is defined by a countably-sized connectivity matrix between outputs and inputs of neurons[3]. We consider DMMs given by matrices with finite num-

[1] e-mail: bukatin@cs.brandeis.edu

[2] https://anhinga.github.io

[3] A single DMM can work with multiple kinds of linear streams, or it can be based on a single kind of linear streams sufficiently expressive for a given situation, such as countably-sized connectivity matrices or V-values.

ber of non-zero elements. Thus we obtain a programming formalism where **programs can be continuously deformed** (see [4] and references therein). The collection of DMM-related programming techniques and examples known at this point is sufficiently diverse to support the claim that this formalism is sufficiently expressive for general purpose programming [3].

4. DMM neurons can handle streams of network-sized matrices, which makes various self-referential constructions possible. We explored self-modifying DMMs with **Self** neuron configured as an accumulator of the current network connectivity matrix taking updates from other neurons in the network [4]. This setup allows to compose various self-modification primitives and to work with their linear combinations (**compositional metalearning**). We experimentally explored emerging properties in the dynamic systems defined by self-modifying DMMs and the ability to modify running DMMs on the fly using their self-modification capabilities [6].

5. This is a new field with a lot of interesting open problems, theoretical and applied. Some of those open problems are collected in the recent preprints [1, 2]. Connections between DMMs and Transformers might be of particular interest.

References

[1] M. Bukatin. *Dataflow matrix machines: a collaborative research agenda.* August 2020. `www.cs.brandeis.edu/~bukatin/dmm-collaborative-research-agenda.pdf`

[2] M. Bukatin. *Using streams of probabilistic samples in neural machines.* January 2020. `https://github.com/anhinga/2020-notes/tree/master/research-notes`

[3] M. Bukatin. *Overview of programming techniques and examples in DMM literature.* January 2020.
`https://github.com/anhinga/2020-notes/tree/master/programming-overview`

[4] M. Bukatin, J. Anthony. Dataflow Matrix Machines and V-values: a Bridge between Programs and Neural Nets. In B. Gyuris, K. Mády, G. Recski, editors, *K + K = 120: Papers dedicated to László Kálmán and András Kornai on the occasion of their 60th birthdays*, Research Institute for Linguistics, Hungarian Academy of Sciences, 2017 (online volume), 2019 (paperback, pp.153-185). `arXiv:1712.07447`

[5] M. Bukatin, S. Matthews. Linear Models of Computation and Program Learning. In G. Gottlob, G. Sutcliffe, A. Voronkov, editors, GCAI 2015, *EasyChair Proceedings in Computing*, **36**, 66-78. `https://easychair.org/publications/paper/Q4lW`

[6] DMM technical report 11-2018. *Dataflow matrix machines: recent experiments and notes for next steps.* November 2018.
`https://github.com/jsa-aerial/DMM/tree/master/technical-report-2018`

[7] K. Keimel. Bicontinuous domains and some old problems in domain theory. *Electronic Notes in Theoretical Computer Science*, **257**:35-54, 2009

[8] M. Warmus. Calculus of Approximations. *Bull. Acad. Pol. Sci., Cl. III*, **4**(5):253–259, 1956.

[9] E.P. Wigner. On the quantum correction for thermodynamic equilibrium, *Physical Review* **40** (June 1932) 749-759