Monoidal substitution diagrams

Samuel B. Steakley
j.w.w. Florian Schwartz

May 1, 2026

The objective of this work-in-progress is to give a construction of free symmetric monoidal closed (SMClosed) categories in the form of a graphical calculus based on **monoidal substitution diagrams**. Our work differs from previous constructions in a certain technical sense (strictness), but it is also quite different in look and feel. As we will show, there is a certain way to read monoidal substitution diagrams that lends a unique visual intuition to an otherwise thorny structure.

Instead of the traditional mathematical formulae for objects of an SMClosed category (e.g. $A \otimes B$, $C \multimap D$, $(E \otimes F) \multimap G$), our calculus uses labeled directed acyclic graphs called **monoidal dependency graphs**. There are procedures that translate in both directions between monoidal dependency graphs and the traditional formulae. For example, the formula below left is translated into the monoidal dependency graph below right:

$$\big((A \multimap B) \otimes (B \multimap C)\big) \multimap (A \multimap C) \qquad \rightsquigarrow \qquad \begin{array}{ccccc} & & A & & \\ & \searrow & & & \\ A & \longrightarrow & B & \longrightarrow & C \\ & \nearrow & & & \\ B & \longrightarrow & C & & \end{array} \tag{1}$$

For now, we limit ourselves to observing that lollipop symbols, which stand for internal hom (or, linear implication) within the formula, are somehow changed into the edges of the graph, whereas tensor symbols are somehow encoded implicitly.

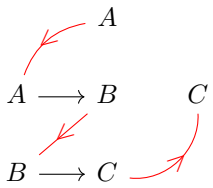
In our calculus, morphisms are represented by labeled directed acyclic graphs called monoidal substitution diagrams. A monoidal substitution diagram is produced by augmenting a monoidal dependency graph with extra edges that link together pairs of nodes that share the same label. For example, using the monoidal dependency graph on the right-hand side of (1) as a base, we can produce a monoidal substitution diagram that represents the internal composition morphism

$$\mathrm{comp}_{A,B,C} : (A \multimap B) \otimes (B \multimap C) \to A \multimap C$$

that is present in any SMClosed category:

$$\mathrm{comp}_{A,B,C} \qquad \rightsquigarrow \qquad \begin{array}{ccccc} & & A & & \\ & \swarrow & & \searrow & \\ A & \longrightarrow & B & \longrightarrow & C \\ & \searrow & & \nearrow & \\ B & \longrightarrow & C & \circlearrowright & \end{array} \tag{2}$$

How does one see that this diagram represents the internal composition? If we remove the last edge from every chain of edges in the underlying monoidal dependency graph, the result is called the **reduced graph** of the diagram:



(3)

Now we read the directed path beginning at the top A as though it is telling us a certain way in which inputs of type A, $A \multimap B$, and $B \multimap C$ can be used to produce an output of type C: we plug the A input into the map of type $A \multimap B$, then we plug the resulting output into the map of type $B \multimap C$, and then we return the resulting C as the final output. Note how this reading is evocative of the way one might typically think of composition of functions, or programs, as being implemented.

In our presentation, we will introduce monoidal substitution diagrams and the fundamental theorems, as well as pending conjectures, that demonstrate how they may serve as the combinatorial basis of a graphical calculus for SMClosed categories. We will show what distinguishes this calculus from previous ones. We will highlight the contributions of the present work, and we will suggest directions for future work.

Related Work

Key axioms for both monoidal dependency graphs and monoidal substitution diagrams were independently published by Heijltjes, Hughes & Straßburger [1], who called the corresponding structures "arenas" and "arena nets," respectively. The gap between the two works is due to the fact that the previous work was primarily concerned with intuitionistic logic, which made it unnecessary to pursue the full treatment of the unit that the present work requires.

References

[1] Willem B. Heijltjes, Dominic J. D. Hughes, and Lutz Straßburger. Intuitionistic proofs without syntax. In *2019 34th Annual ACM/IEEE Symposium on Logic in Computer Science (LICS)*, pages 1–13, Vancouver, BC, Canada, June 2019. IEEE.