VINCENT WANG-MAŚCIANICA

STRING DIAGRAMS FOR TEXT

Contents

0.1 A generative grammar for text circuits 4
0.1.1 A circuit-growing grammar 4
0.1.2 Text structure 5
0.1.3 Simple sentences 5
0.1.4 Noun distribution and linking 7
0.1.5 Modifiers 9
0.1.6 Rewriting to circuit-form 10
0.1.7 Putting it all together 12
0.1.8 Extensions I: relative and reflexive pronouns 15
0.1.9 Extensions II: grammar equations 15
0.1.10 Extensions III: higher-order modifiers 15
0.1.11 Equivalence to internal wirings 16
0.1.12 Text circuit theorem 16
0.1.13 Related work 16
0.2 Text circuits: details, demos, developments 17

(Acknowledgements will go in a margin note here.)

0.1 A generative grammar for text circuits

0.1.1 A circuit-growing grammar

There are many different ways to write an n-categorical signature that generates circuits. Mostly as an illustration of expressive capacity, I will provide a signature where the terms "surface" and "deep" structure are taken literally as metaphors; the generative grammar will grow a line of words in (roughly) syntactic order, and like mushrooms on soil, the circuits will appear as the mycelium underneath words.

Simplifications: Propositions only, no determiners, only one tense, no morphological agreement between nouns and their verbs and referring pronouns, and we assume that adverbs, adverbial adjunctions, and adjectives stack indefinitely and without further order requirements; e.g. `Yesterday yesterday yesterday Alice happily secretly finds red big toy shiny car that he gives to Bob.` we consider grammatical enough. For now, we consider only the case where adjectives and adverbs appear before their respective noun or verb. Note that all of these limitations can principle be overcome by the techniques we developed in Section **??** for restricted tree-adjoining and links.

Definition 0.1.1 (Lexicon)**.** We define a lexicon $\mathcal{L}$ to be a tuple $(\mathbf{N}, \mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_S, \mathbf{A}_N, \mathbf{A}_V, \mathbf{A}_S, \mathbf{C})$

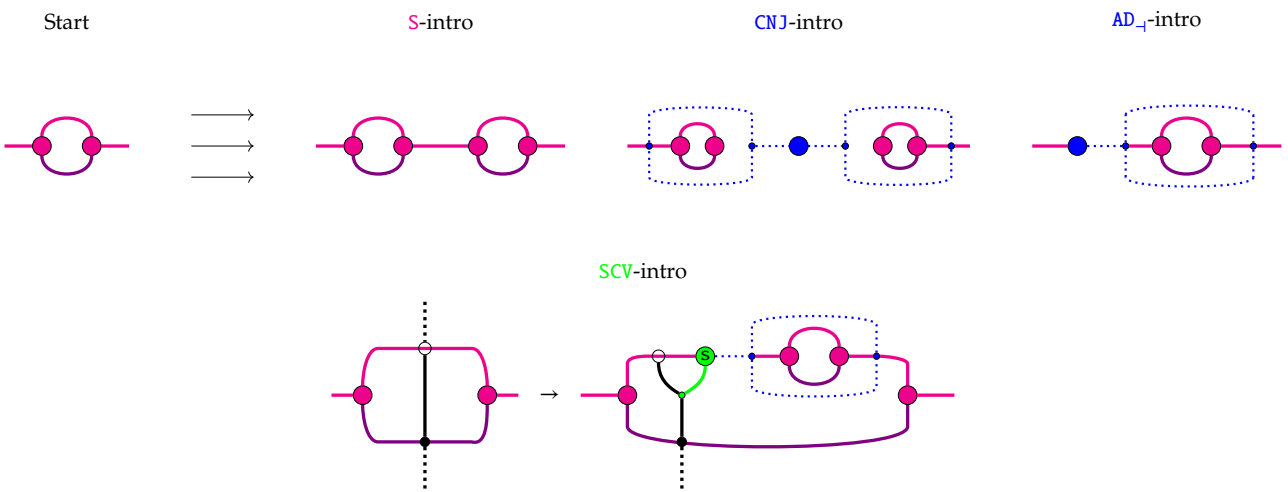
Where:

- $\mathbf{N}$ is a set of *proper nouns*
- $\mathbf{V}_1$ is a set of *intransitive verbs*
- $\mathbf{V}_2$ is a set of *transitive verbs*
- $\mathbf{V}_S$ is a set of *sentential-complement verbs*
- $\mathbf{A}_N$ is a set of *adjectives*
- $\mathbf{A}_V$ is a set of *adverbs*
- $\mathbf{A}_S$ is a set of *adverbial adpositions*
- $\mathbf{C}$ is a set of *conjunctions*

0.1.2 *Text structure*

We work in a dimension where wires behave symmetric monoidally by homotopy, and further assume compact closure rewrite rules for all wire-types. Our strategy is to generate "bubbles" for sentences, within which we can grow circuit structure piecemeal. We will only express the rewrite rules; the generators of lower dimension are implicit. We aim to recover the linear ordering of words in text (essential to any syntax) by traversing the top surface of a chain of bubbles representing sentence structure in text – this order will be invariant despite compact closure. First we introduce rules to capture the broad structure of text as a structured collection of *simple sentences*, where a simple sentence is one that only contains a single verb.

Starting from a single simple sentence bubble, we may always append another to obtain a list. However, more than list structure, we may have two simple sentences joined by a conjunction, e.g. `Alice dances while Bob drinks`. In the same vein, the unary case of a conjunction of simple sentences is an adverbial adposition, e.g. `Yesterday Alice dance(d)`. Finally, we may have verbs that take a sentential complement rather than a noun phrase, e.g. `Alice sees Bob dance`; these verbs require nouns, which we depict as wires spanning bubbles.



0.1.3 *Simple sentences*

Simple sentences are sentences that only contain a single intransitive or transitive verb. Simple sentences will contain at least one noun, and may optionally contain adjectives, adverbs, and adpositions. The rules for

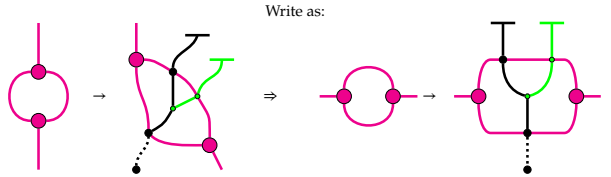
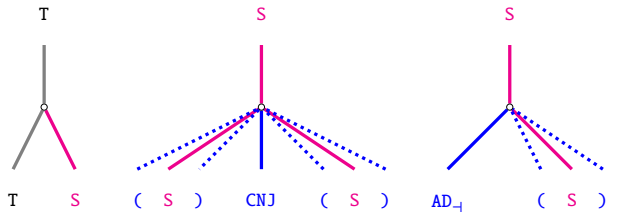


Figure 1: **How to read the diagrams in this section:** we will be making heavy use of pink and purple bubbles as frames to construct circuits. We will depict the bubbles horizontally, as we are permitted to by compact closure, or by reading diagrams with slightly skewed axes.

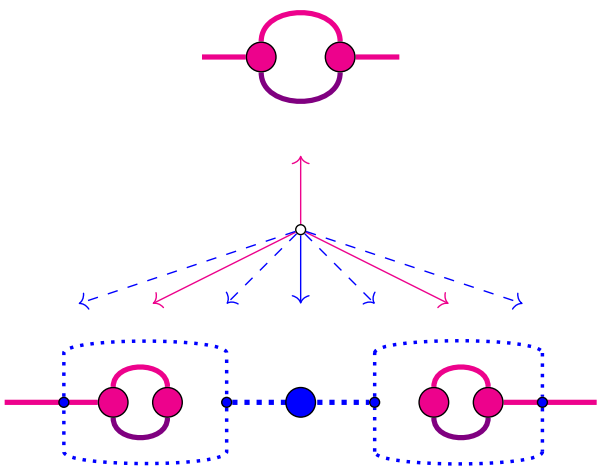
Figure 2: The blue-dotted guards are for circuit-translation later, to indicate the contents of boxes.

Definition 0.1.2 (Simple sentence structure)**.** In prose, a text is a list of sentences. A sentence can be a pair of sentences, each guarded by scopes with a conjunction in between. A sentence can carry an adverbial adposition at the start. As a CFG, these considerations are respectively depicted as:

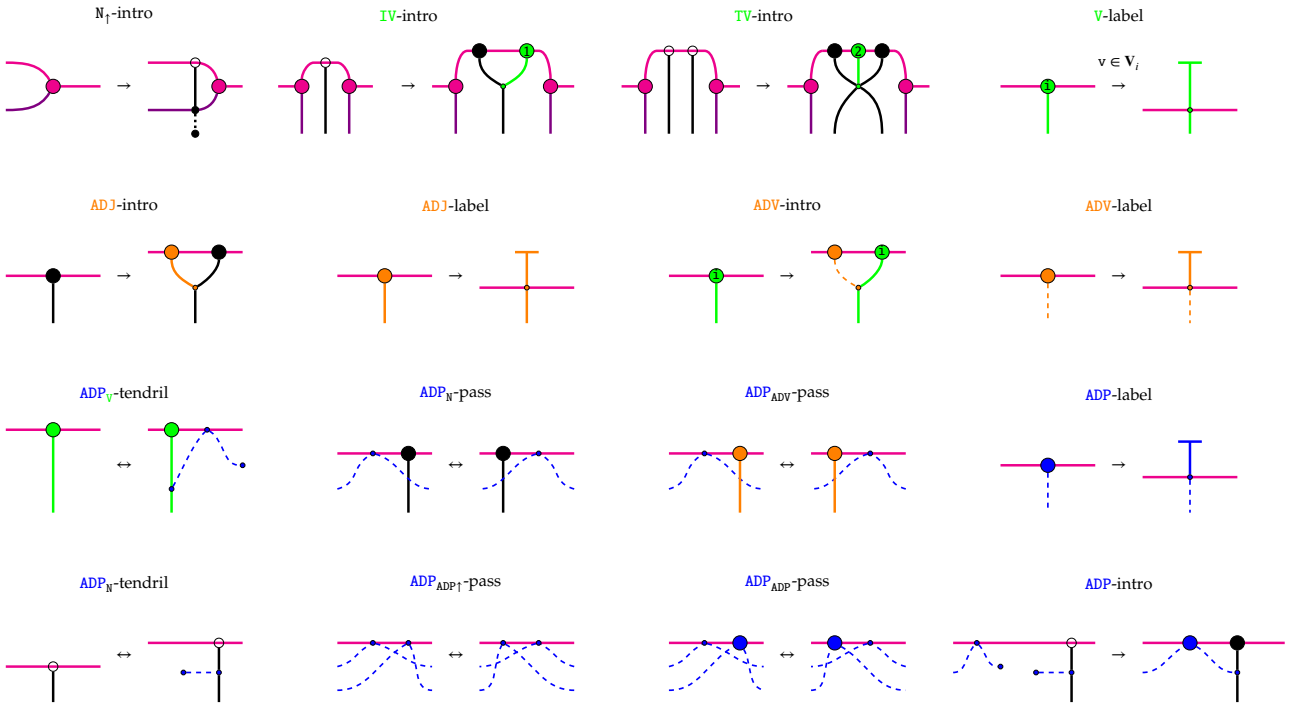


Proposition 0.1.3. The sentence introduction rules are complete with respect to simple sentence structure.

Proof. Excluding verbs with sentential complements to be dealt with later, the sentence-introduction rewrites graphically correspond to the CFG rules. □



generating simple sentences are as follows:



The N↑-intro rule introduces new unsaturated nouns from the end of a simple sentence. The IV-intro rule applies when there is precisely one unsaturated noun in the sentence, and the TV-intro rule applies when there are precisely two. Both verb-introduction rules saturate their respective nouns, which we depict with a black bulb. Adjectives may be introduced immediately preceding saturated nouns, and adverbs may be introduced immediately preceding any kind of verb. The position of adpositions in English is not context-free, and requires some aid from the deep structure beneath the surface. The ADP$_V$-tendril rule allows an unsaturated adposition to appear immediately after a verb; a bulb may travel by homotopy to the right, seeking an unsaturated noun. Conversely, the bidirectional ADP$_N$-tendril rule sends a mycelic tendril to the left, seeking a verb. The two pass-rules allow unsaturated adpositions to swap past saturated nouns and adjectives; note that by construction, neither verbs nor adverbs will appear in a simple sentence to the right of a verb, so unsaturated adpositions will move right until encountering an unsaturated noun, and in case it doesn't, the tendril- and pass- rules are bidirectional and hence reversible.

0.1.4 Noun distribution and linking

We require a definition of coreference structure in order verify that our rewrite rules are complete with respect to them. One reasonable constraint on coreference in text is that coreference always goes backwards. One way to express this constraint mathematically is the following:

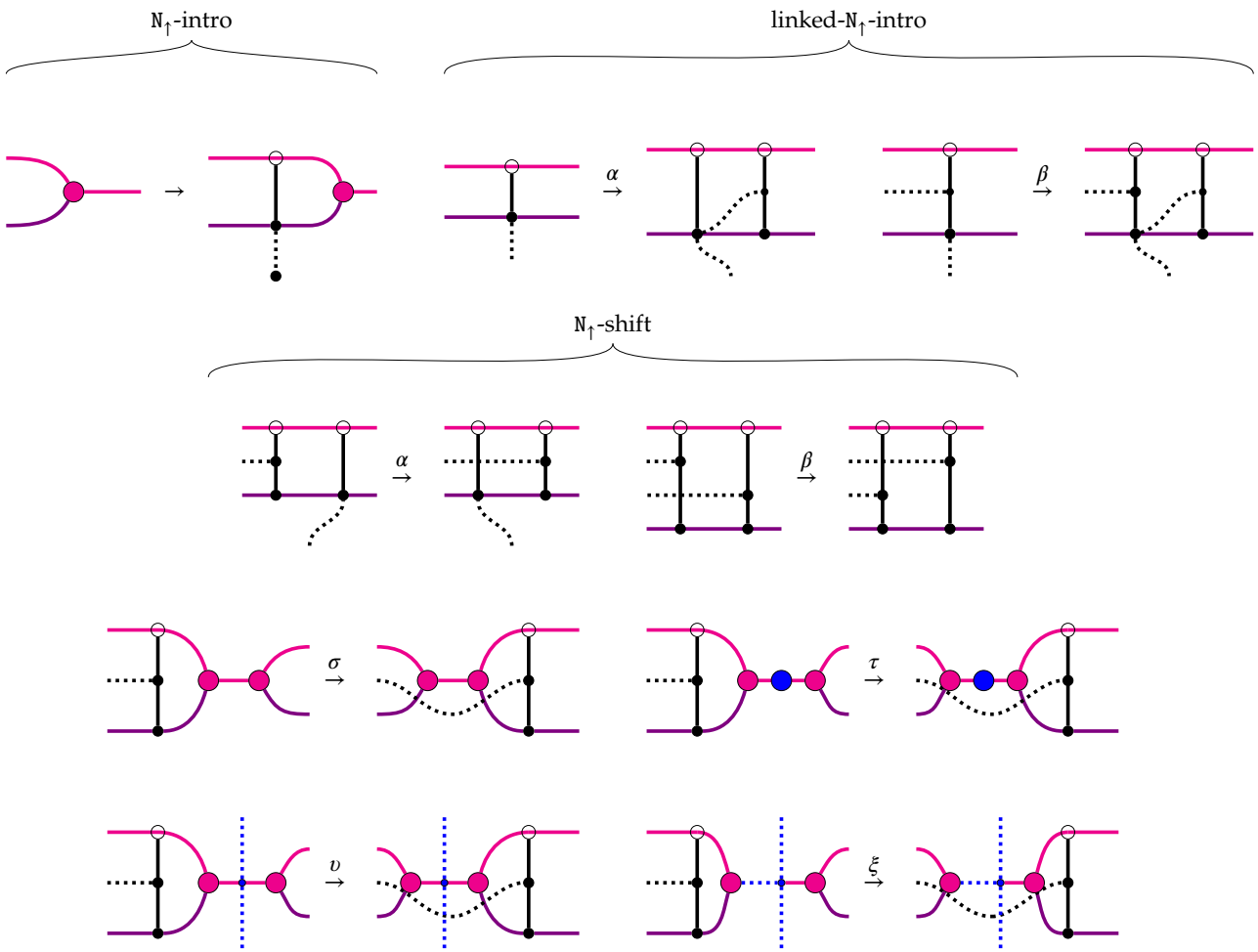
Definition 0.1.4 (Coreference structure)**.** In a text with $K \in \mathbb{N}$ distinct nouns and their coreferents, the linear order of noun+referents in a text corresponds to a list of positive integers that:

1. Starts with 1
2. Contains every $k \in [1 \cdots K]$
3. For all $k \in [1 \cdots K]$, the head of the list up to the first occurence of k contains all $j < k$.

Proposition 0.1.5. The noun-introduction and manipulation rules are expressively complete with respect to coreference structure.

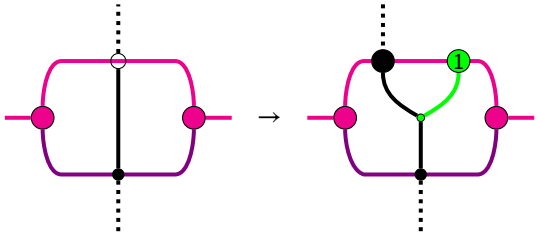
Proof. □

Now we deal with nouns. The major complication here is the accommodation of coreference. We want to keep track of which (pro)nouns share a reference so that we can ultimately eliminate the distinction between e.g. `Bob likes himself` and `Bob likes Bob`. So we will generate *unsaturated* nouns and their coreference structure first. We will further ask for a distinction between *unsaturated* and *saturated* nouns – only the latter of which are ready to be labelled – which will resolve a minor complication regarding the context-sensitivity of adpositions. We define three classes of rules for nouns. The first is the introduction of novel unsaturated nouns, which can occur in any sentence-bubble. The second class handles the introduction of a new, coreferentially-linked noun to the right of an extant noun. The third concerns swapping the positions of unsaturated and linked nouns within and between bubbles.

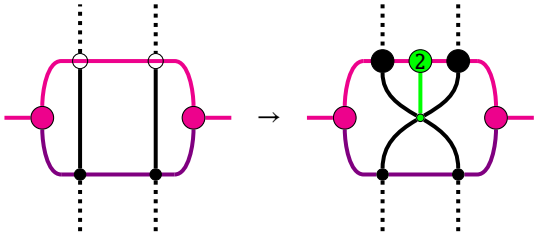

$\mathtt{N}_\uparrow$-intro
linked-$\mathtt{N}_\uparrow$-intro
α
β
$\mathtt{N}_\uparrow$-shift
α
β
σ
τ
υ
ξ

Nouns require verbs in order to be saturated. From left-to-right; if there is precisely one unlabelled noun, we may introduce an unlabelled intransitive verb and saturate the noun so that it is now ready to grow a label; or if there are two unlabelled nouns, we may introduce an unlabelled transitive verb on the surface and saturate the two nouns that will be subject and object; and verbs may be labelled.

IV-intro

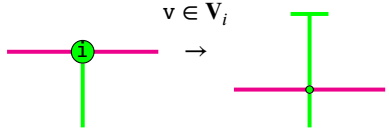


TV-intro



V-label

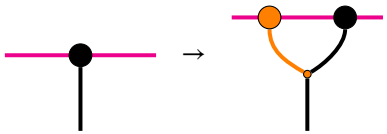
$v \in \mathbf{V}_i$



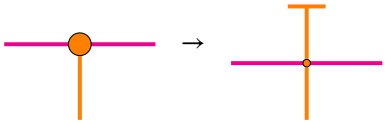
0.1.5 Modifiers

Modifiers are optional parts of sentences that modify (and hence depend on there being) nouns and verbs. We consider adjectives, adverbs, and adpositions. From left to right; we allow adjectives to sprout immediately before a saturated noun, and we allow adverbs to sprout immediately before any verb.

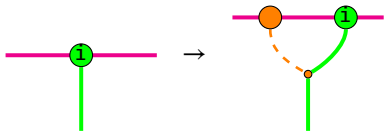
ADJ-intro



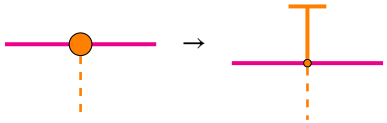
ADJ-label



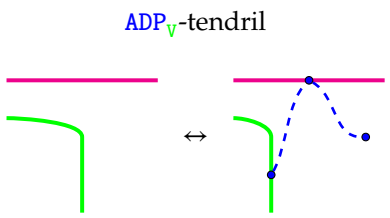
ADV-intro

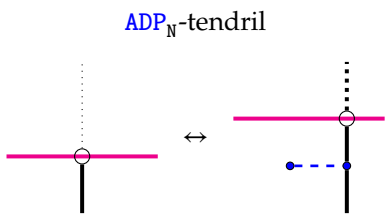


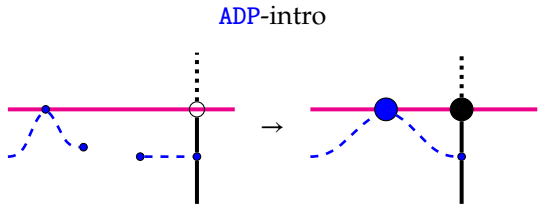
ADV-label

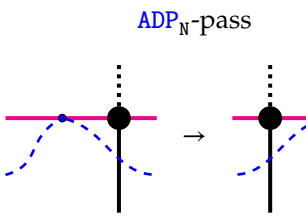


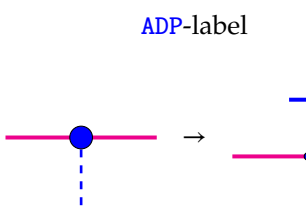
Adpositions modify verbs by tying in an additional noun argument; e.g. while `runs` is intransitive, `runs towards` behaves as a transitive verb. Some more advanced technology is required to place adpositions and their thematic nouns in the correct linear order on the surface. In the left column; an adposition tendril can sprout from a verb via an unsaturated adposition, seeking an unsaturated noun to the right; an unsaturated noun can sprout an tendril seeking a verb to connect to on the left. Both of these rewrites are bidirectional, as tendrils might attempt connection but fail, and so be retracted. In the centre, when an unsaturated adposition and its tendril find an unsaturated noun, they may connect, saturating the adposition so that it is ready to label. In the right column; an unsaturated adposition may move past a saturated noun in the same sentence, which allows multiple adpositions for the same verb; finally, a saturated adposition can be labelled.





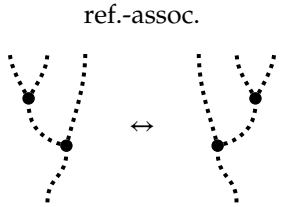


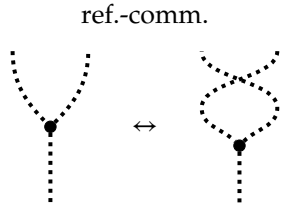


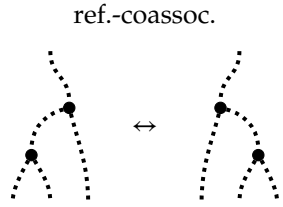


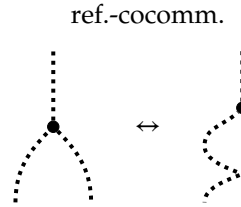
0.1.6 *Rewriting to circuit-form*

Resolving references

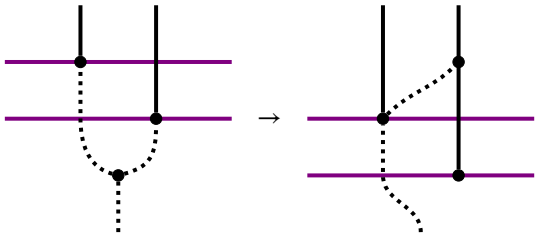




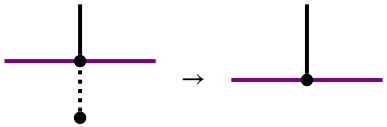




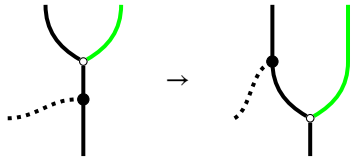
ref-link



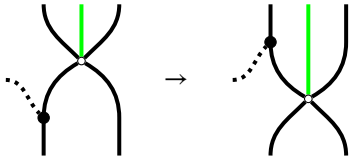
ref-kill



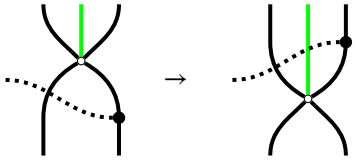
link-up(IV)



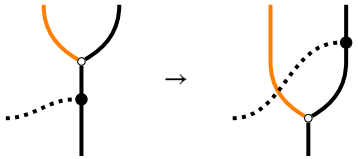
link-up(TV:s)



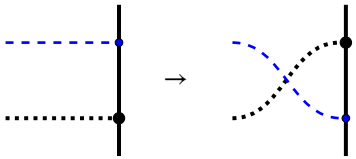
link-up(TV:o)



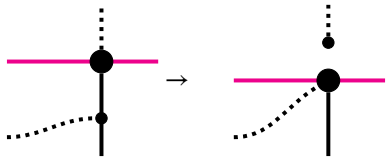
link-up(ADJ)



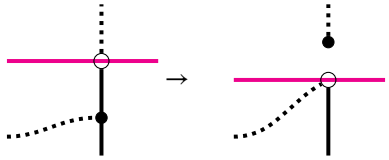
link-up(ADP)



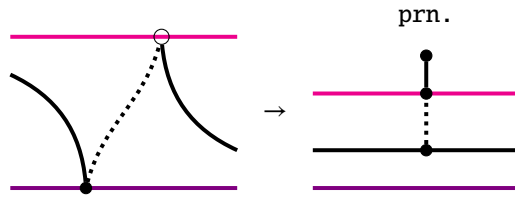
link-finish(N:sat)



link-finish(N)



prn-intro



Connecting circuits

0.1.7 *Putting it all together*

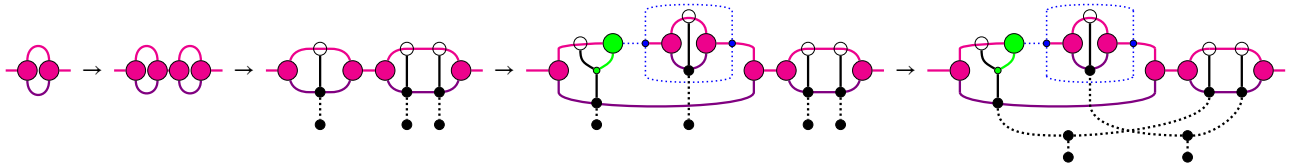


Figure 3: Starting from the initial sentence bubble, we generate a new sentence, introduce some nouns and an SCV, and then we connect our references at the bottom.

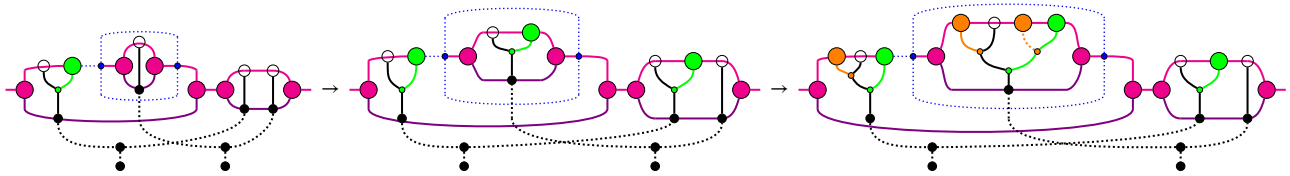


Figure 4: Then we introduce intransitive verbs to saturate nouns, and we may also sprout some modifying adjectives and adverbs.

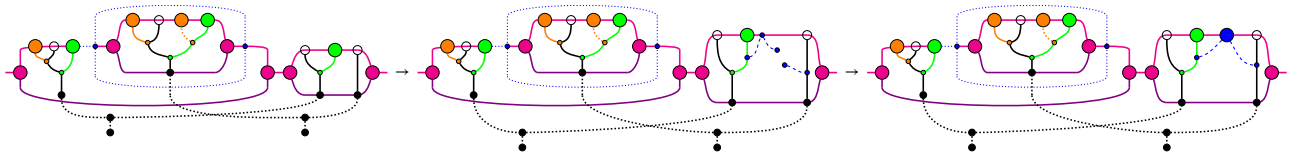


Figure 5: For the remaining unsaturated noun, we use the adposition introduction rules to sprout tendrils off of the other verb in the bubble, and connect.

Figure 6: All of the non-noun words may now be labelled.

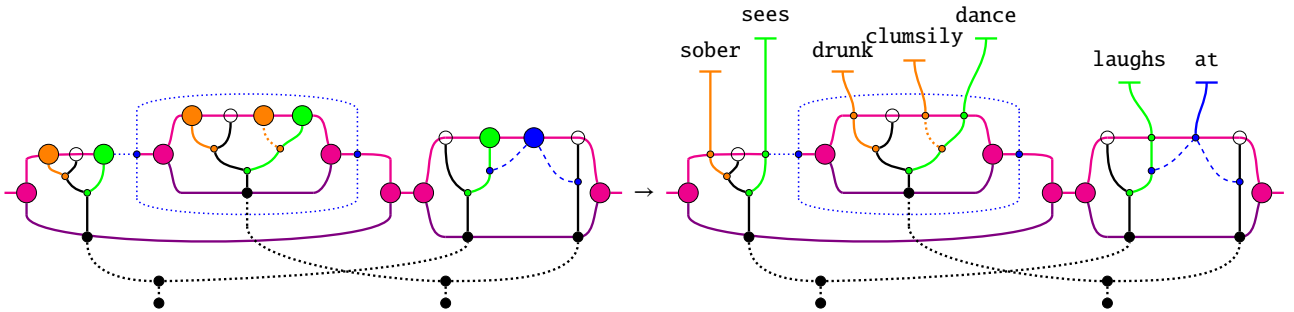


Figure 7: To begin assigning nouns, observe that by compact closure of the bubble boundaries, we can deform the diagram to obtain suitable forms for our local rewrite rules for link-generation at the bottom.

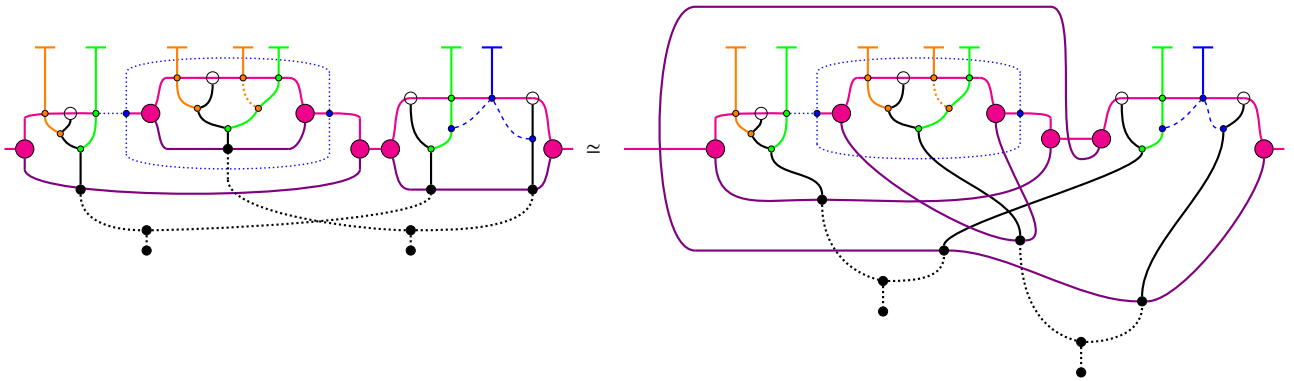


Figure 8: Now we can introduce our noun labels and linearise our link structure.

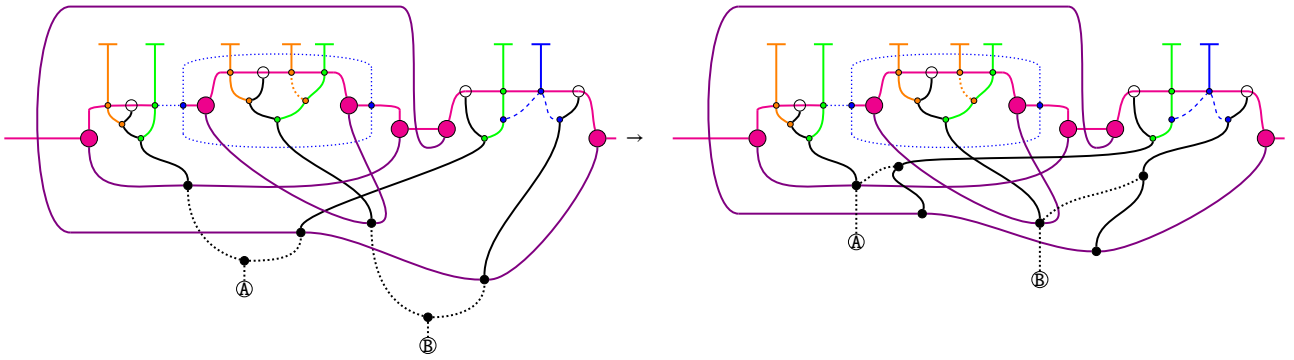


Figure 9: Once the link structure is linearised, we can undo the deformation, and propagate links to the surface.

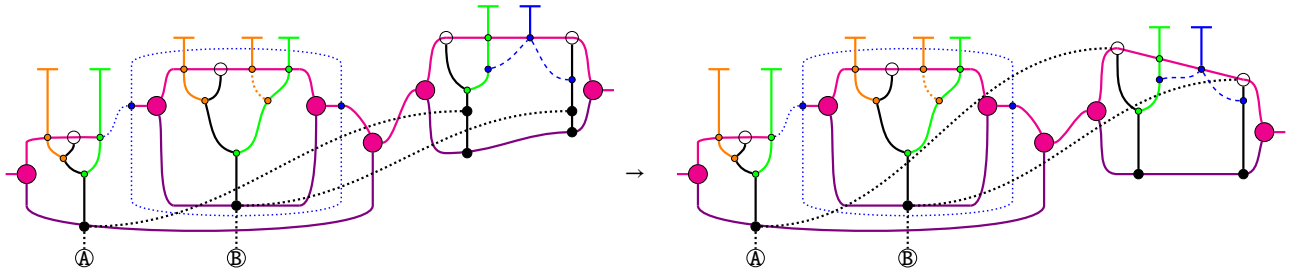
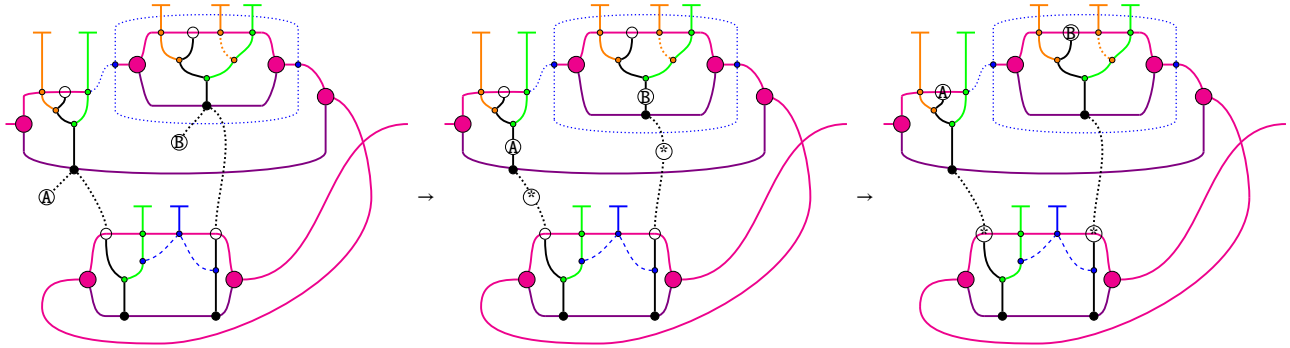


Figure 10: The bubbles may be rearranged to respect circuit-form, such that the propagation of noun labels to the surface traces out the wires of the end circuit.



0.1.8 Extensions I: relative and reflexive pronouns

Subject relative pronouns

Example 0.1.6.

Object relative pronouns

Example 0.1.7.

Reflexive pronouns

Example 0.1.8.

0.1.9 Extensions II: grammar equations

Attributive vs. predicative modifiers

Example 0.1.9.

Copulas

Example 0.1.10.

Possessive pronouns

Example 0.1.11.

0.1.10 Extensions III: higher-order modifiers

Intensifiers

Example 0.1.12.

Comparatives

Example 0.1.13.

0.1.11 Equivalence to internal wirings

0.1.12 Text circuit theorem

0.1.13 Related work

0.2 *Text circuits: details, demos, developments*

This section covers some practical developments, conventions, references for technical details of text circuits. The most striking demonstration to date is that circuits are defined over a large enough fragment of language to *leverage* several bAbi tasks CITE , which are a family of 20 general-reasoning text tasks – the italicised choice of wording will be elaborated shortly. Each family of tasks consists of tuples of text in simple sentences concluded by a question, along with an answer. It was initially believed that world models were required for the solution of these tasks, but they have been solved using transformer architectures. While there is no improvement in capabilities by solving bAbi using text circuits, the bAbi tasks have been used as a dataset to learn word gates from data, in a conceptually compliant and compositional manner. Surprisingly, despite the low-data, low-compute regime, the tasks for which the current theory has the expressive capacity to cover are solved better by text circuits than by LSTMs; a proof-of-concept that with the aid of appropriate mathematics, not only might fundamental linguistic considerations help rather than hinder NLP, but also that explainability and capability are not mutually exclusive. Experimental details are elaborated in a forthcoming report CITE . While there are expressivity constraints contingent on theoretical development, this price buys a good amount of flexibility within the theoretically established domain: text circuits leave room for both learning-from-data and "hand-coded" logical constraints expressed process-theoretically, and naturally accommodate previously computed vector embeddings of words.

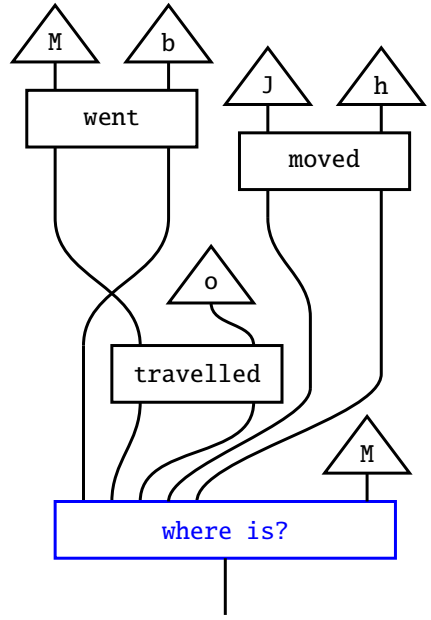
In practice, the process of obtaining transparently computable text goes through two phases. First, one has to obtain text circuits from text, which is conceptually simple: typelogical parsers for sentences can be modified to produce circuit-components rather than trees, and a separate pronomial resolution module dictates symmetric monoidal compositionality; details are in the same forthcoming report CITE . Second, one implements the text circuits on a computer. On quantum computers, boxes are modelled as quantum combs CITE . On classical computers, boxes are sandwiches of generic vector-manipulation neural nets, and boxes with 'dot dot dot' typing are interpreted as families of processes, which can be factored for instance as a pair of content-carrying gates along with a monoid+comonoid convolution to accommodate multiplicity of wires. The theoretical-to-practical upshot of text circuits when compared to DisCoCat is that the full gamut of compositional techniques, variations, and implementation substrates of symmetric monoidal categories may be used for modelling, compared to the restrictions inherent in hypergraph and strongly compact closed categories.

In terms of underpinning mathematical theory, the 'dot dot dot' notation within boxes is graphically formal (**?**), and interpretations of such boxes were earlier formalised in (**???**). The two forms of interacting composition, one symmetric monoidal and the other by nesting is elsewhere called *produoidal*, and the reader is referred to CITE for formal treatment and a coherence theorem. Boxes with holes may be interpreted in several different ways. Firstly, boxes may be considered syntactic sugar for higher-order processes in monoidal closed categories, and boxes are diagrammatically preferable to combs in this regard, since the latter admits

An example from Task 1, "single supporting fact", is:

```
Mary went to the bathroom.
John moved to the hallway.
Mary travelled to the office.
(Query:) Where is Mary?
(Answer:) office.
```

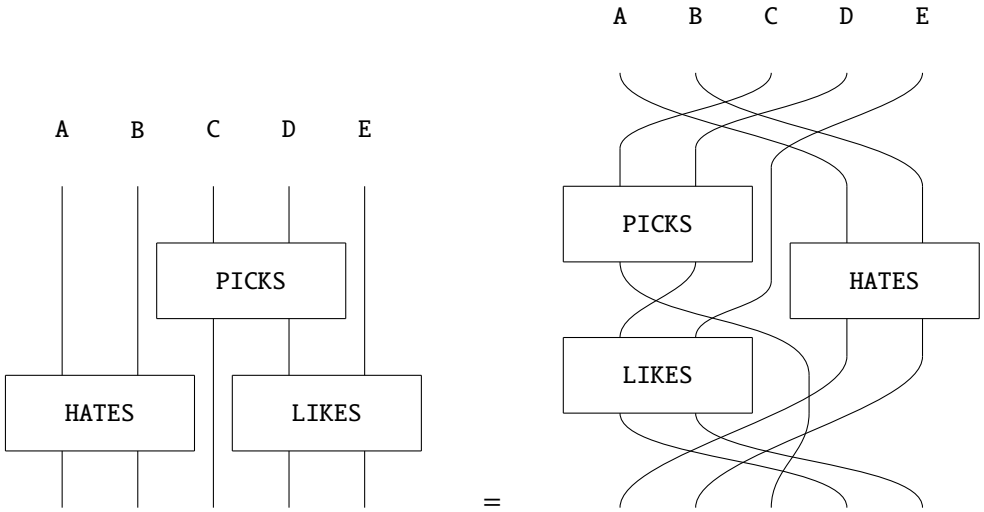
Translating the setup of each task into a circuit of neural nets-to-be-learnt, and queries into appropriately typed measurements-to-be-learnt, each bAbi task becomes a training condition: the depicted composite process ought to be equal to the `office` state.



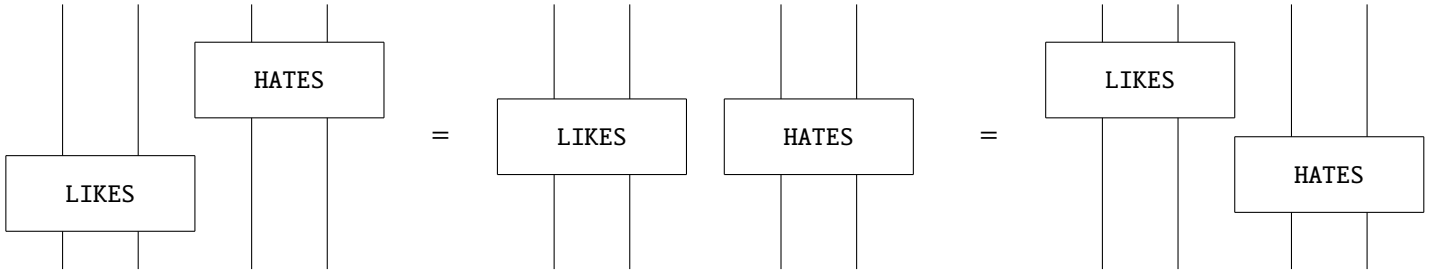
Solving bAbi in this way also means that each word gate has been learnt in a conceptually-compliant manner, insofar as the grounded meanings of words are reflected in how words interact and modify one another. One may argue that the verb `to go` and synonyms have been learnt-from-data in a way that coheres with human conceptions by appeal to the bAbi dataset.

a typing pathology where two mutually facing combs interlock. Secondly, boxes need not be decomposable as processes native to the base category, admitting for instance an interpretation as elementwise inversion in linear maps, which specialises in the case of **Rel** (viewed as **Vect** over the boolean ring) to negation-by-complement. In some sense, none of these formalities really matter, on the view that text circuits are algebraic jazz for interpreting text, where facets are open to interpretation and modification.

Convention 0.2.1. Sometimes we allow wires to twist past each other, and we consider two circuits the same if their gate-connectivity is the same:

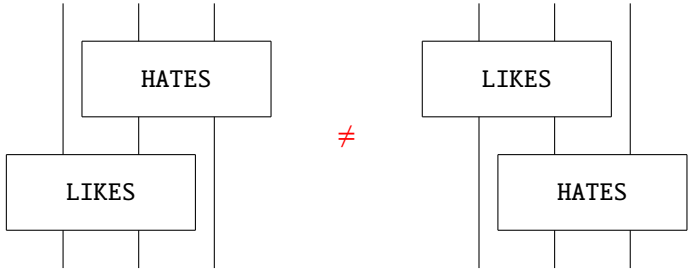


Since only gate-connectivity matters, we consider circuits the same if all that differs is the horizontal positioning of gates composed in parallel:

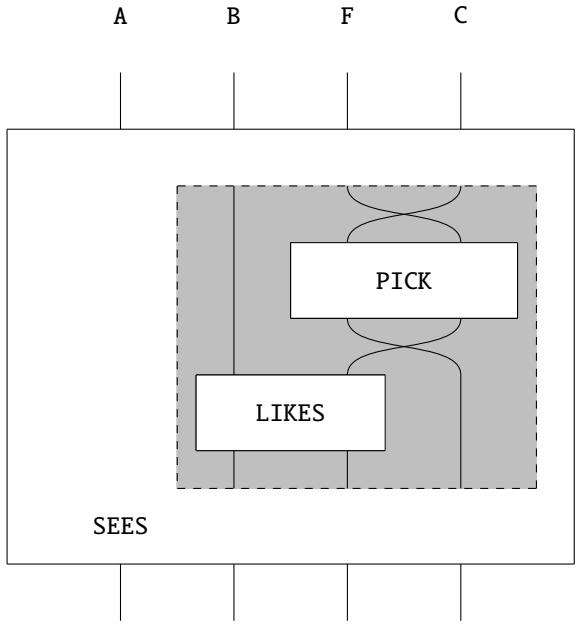


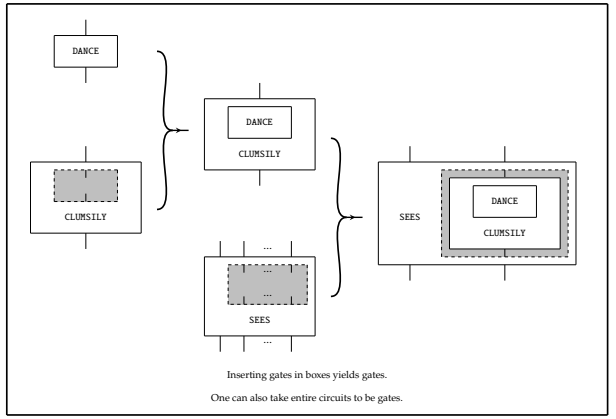
We do care about output-to-input connectivity, so in particular, we **do not** consider circuits to be equal up to

sequentially composed gates commuting past each other:



Example 0.2.2. The sentence ALICE SEES BOB LIKES FLOWERS THAT CLAIRE PICKS can intuitively be given the following text circuit:





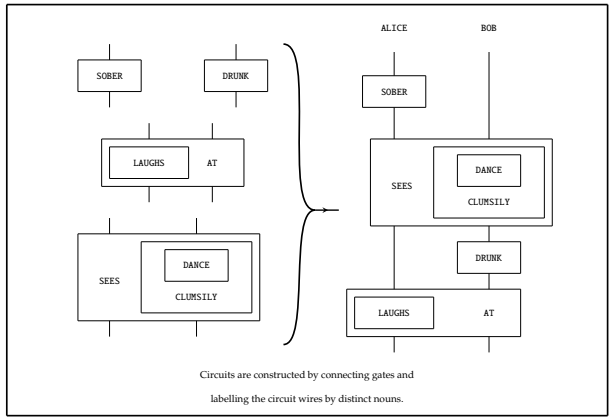


Figure 11: Sentences correspond to filled gates, boxes with fixed arity correspond to first-order modifiers such as adverbs and adpositions, and boxes with variable arity correspond to sentential-level modifiers such as conjunctions and verbs with sentential complements. Composition by connecting wires corresponds to identifying coreferences in discourse, and composition by nesting corresponds to grammatical structure within sentences.