Proofs as pictures

Michael Moortgat

UCLA, 22 May 2013

Abstract

Proof nets for typelogical grammars identify sequent derivations that are based on the same axiom linking between input and output atomic formulas. The technique of focusing similarly aims to avoid the spurious non-determinism of backward-chaining search in sequent calculus. But focusing identifies less derivations than proof nets. In this talk (based on joint work with Richard Moot) I introduce the distinction between a proof net and its composition graph. Interpretations are read off from the composition graph associated with a net; these interpretations then are shown to coincide with those of focused sequent proofs. I illustrate with the construal of scopal ambiguities, showing that these phenomena do not warrant the introduction of higher order syntactic type-assignments, but can be dealt with in the simplest type logic available, the purely applicative **AB**.

References

Andreoli 2001, Fodussing and proof construction. Annals of Pure and Applied Logic, 107:131-163.
Bastenhof 2012. Polarized Montagovian semantics for the Lambek-Grishin calculus. LNCS 7395, 1-16.
Moortgat and Moot 2012. Proof nets and the categorial flow of information. LIRa Yearbook 2011.

1. Background: MG and TLG

Similarities

- Fundamental composition operation: Merge, $\otimes$
- Selection wrt Merge: `=A B` compare $A \multimap B$, $A^\perp \oplus B$
- Control: `+f, -f` compare $\langle f\rangle, [f]$ with $\langle f\rangle[f]A \vdash A$

  e.g. typing wh question word: wh/(s/$\Diamond\Box$d) (Vermaat, Kurtonina/Moortgat)

Differences

- MG typing is 'first order' (no nested selection) $\leadsto$ application (Elim) $M^{A\multimap B}\ N^A$
- TLG relies on higher-order types for (overt/covert) displacement

  $\leadsto$ abstraction (Intro) $\lambda x^A.M^B$
- MG is single-conclusion, intuitionistic
- TLG (Utrecht style) exploits classical symmetries: Merge $\otimes$, Co-Merge $\oplus$ etc

2. Expressivity, complexity

Symmetric TLG

- Expressivity beyond MCS (Melissen); well-behaved LTAG fragments (Moot)
- Complexity: NP-complete (Bransen); with fixed lexicon: ?

Compare standard Lambek Calculus: CF; NP-complete; with fixed lexicon: polynomial (Pentus)

Balancing expressivity/complexity

- Syntax: ACG lesson — MCS hierarchy obtained from 1st order abstract syntax

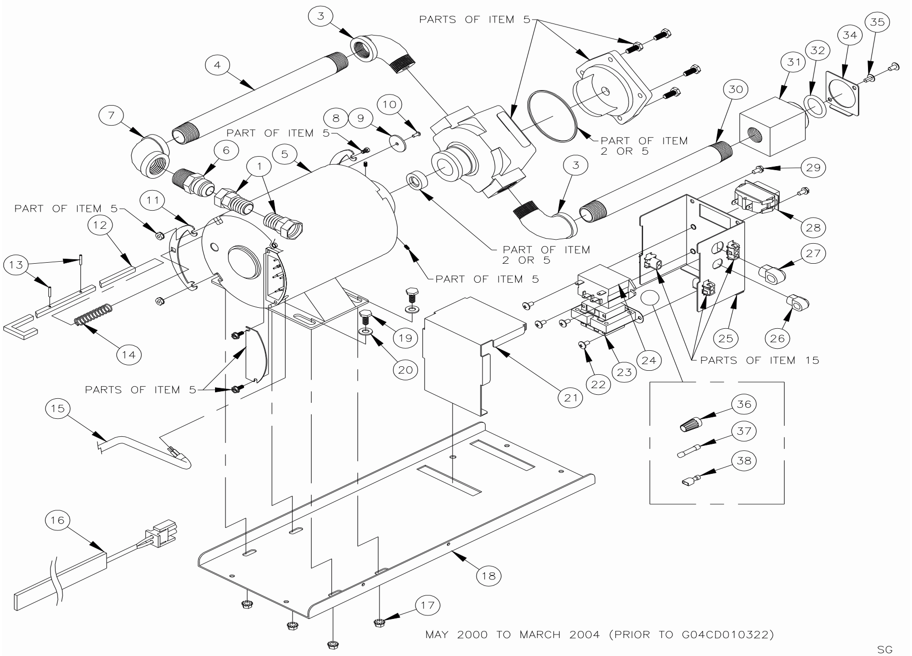
  but: no lexical anchoring
- Semantics: continuation semantics compatible with 1st order source types

  ¡¡ we need a new view on function application/abstraction !!

In the talk, I explore this balance in the pictorial language of TLG proof nets.

3. Click-n-drag nets

- Moot/Puite nets: an attractive diagrammatic calculus for TLG
  - Standard systems $/, \otimes, \backslash$; extensions $\Diamond, \Box, {}^{\mathbf{0}}\cdot, \cdot^{\mathbf{0}}$
  - Bilinear generalization: dual operations $\oslash, \oplus, \oslash, \cdot^{\mathbf{1}}, {}^{\mathbf{1}}\cdot$
  - Entropy laws: invariance under one-way deformations
  - Free of spurious ambiguity (contrast: sequent calculus)
- Continuation semantics with focusing (Bernardi/MM 2007, Bastenhof 2012): Moot-Puite nets identify too many proofs
  - nets are determined by a bijection between premise/conclusion atomic frms
  - in addition, we need a bijection between focusing/defocusing moves
- We provide proof net support for continuation semantics
  - interpretations are read off from the steps that assemble a net from its composition graph
  - composition graph: 'exploded parts' view of a net



MAY 2000 TO MARCH 2004 (PRIOR TO G04CD010322)

SG

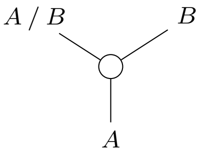
4. Recap: graphical calculus

- Basic building blocks: links (=hyperedges)
  - type: tensor, cotensor
  - premises $P_1, \ldots, P_n$, conclusions $C_1, \ldots, C_m$, $0 \leq n, m$
  - Main formula: empty or one of the P_i, C_j
- Proof structure. Set of links over finite set of frm's s.t. every frm is at most once premise and at most once conclusion of a link.
  - hypotheses: $\neg$ conclusion of any link
  - conclusions: $\neg$ premise of any link
  - axioms: $\neg$ main formula of any link
- Abstract proof structure: PS with formulas at internal nodes erased.
- Rewriting: logical and structural conversions $\rightsquigarrow$ next slides
- Proof net: APS converting to a tensor tree (possibly unrooted)

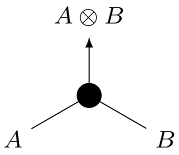
5. Logical links: binary connectives

$$A \to C/B \quad \textit{iff} \quad A \otimes B \to C \quad \textit{iff} \quad B \to A\backslash C$$

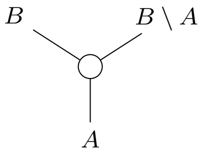
$(/L)$



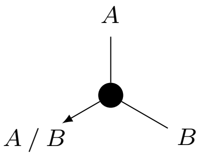
$(\otimes L)$



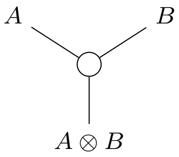
$(\backslash L)$



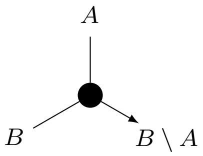
$(/R)$



$(\otimes R)$

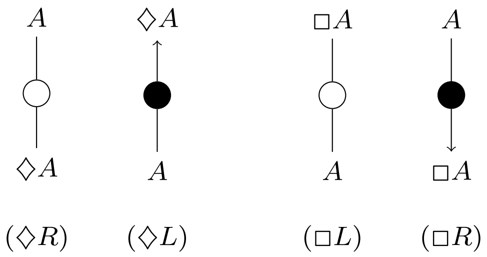


$(\backslash R)$



6. Unary connectives: residuated

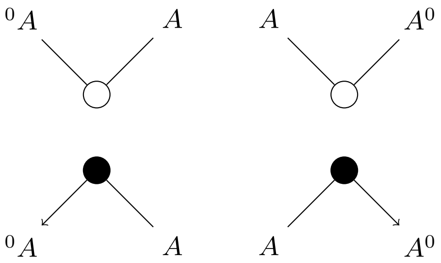
$$\Diamond A \to B \quad \textit{iff} \quad A \to \Box B$$



7. Unary connectives: Galois connected

$$A \to {}^{\mathbf{0}}B \quad \textit{iff} \quad B \to A^{\mathbf{0}}$$

$(^{\mathbf{0}} \cdot L)$ $(\cdot^{\mathbf{0}} L)$



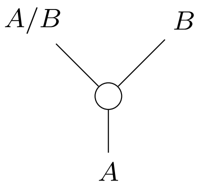
$(^{\mathbf{0}} \cdot R)$ $(\cdot^{\mathbf{0}} R)$

8. Duality: reflection

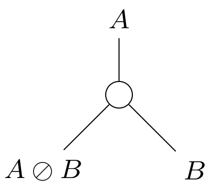
Lambek-Grishin calculus extends the vocabulary with duals of the operations considered before ($\diamond, \Box$ are self-dual):

$$B \obslash C \rightarrow A \quad \textit{iff} \quad C \rightarrow B \oplus A \quad \textit{iff} \quad C \oslash A \rightarrow B$$
$$A^{\mathbf{1}} \rightarrow B \quad \textit{iff} \quad {}^{\mathbf{1}}B \rightarrow A$$

Reflection Links for the dual residuated/Galois connected operations are obtained by reflections through the horizontal axis. Compare:



right selection



right difference

9. Proof nets

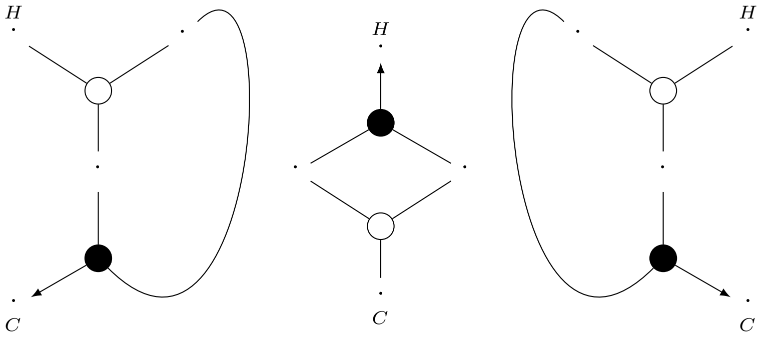
Tree A tree is an acyclic, connected (possibly unrooted) abstract proof structure which does not contain any cotensor links.

Net A proof structure P is a proof net iff its underlying abstract proof structure A converts to a tree by means of

- logical contractions
- structural rewrites

10. Logical contractions: binary connectives

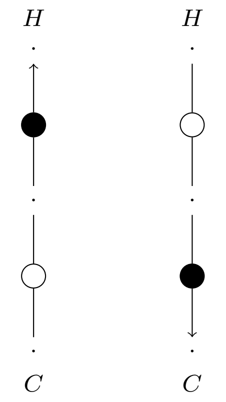
The following configurations contract to $\begin{matrix} H \\ \cdot \\ C \end{matrix}$



$(/R)$ $(\otimes L)$ $(\backslash R)$

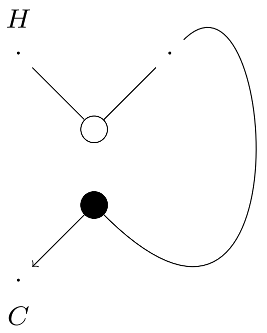
11. Logical contractions: unary connectives

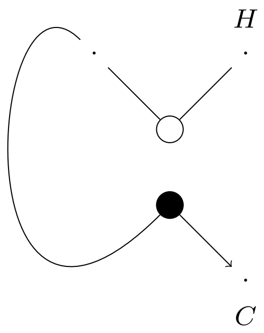
Similarly, the following unary configurations contract to $\begin{array}{c} H \\ \cdot \\ C \end{array}$



$(\Diamond L)$ $(\Box R)$

12. Contractions: Galois connected operations

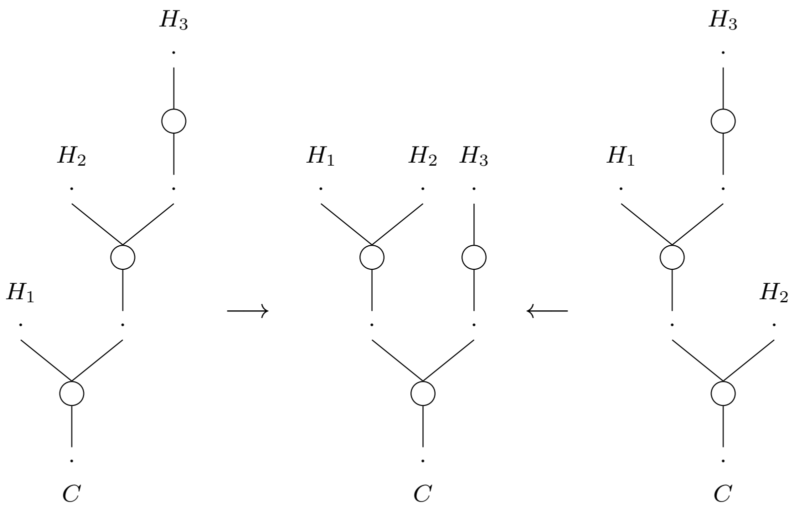




$${}^{\mathbf{0}}A \to {}^{\mathbf{0}}A$$
$$A \to {}^{\mathbf{0}}(A^{\mathbf{0}})$$

$$A^{\mathbf{0}} \to A^{\mathbf{0}}$$
$$A \to ({}^{\mathbf{0}}A)^{\mathbf{0}}$$

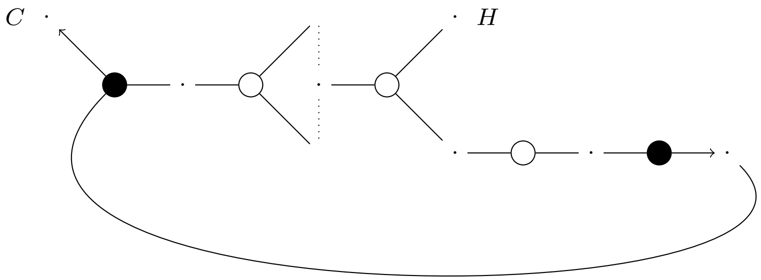
13. Structural rules: right branch extraction



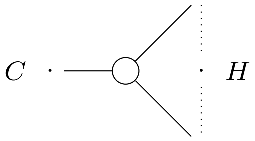
$A \otimes (B \otimes \Diamond C)$ $\qquad$ $(A \otimes B) \otimes \Diamond C$ $\qquad$ $(A \otimes \Diamond C) \otimes B$

14. Action-at-a-distance: generalized contractions

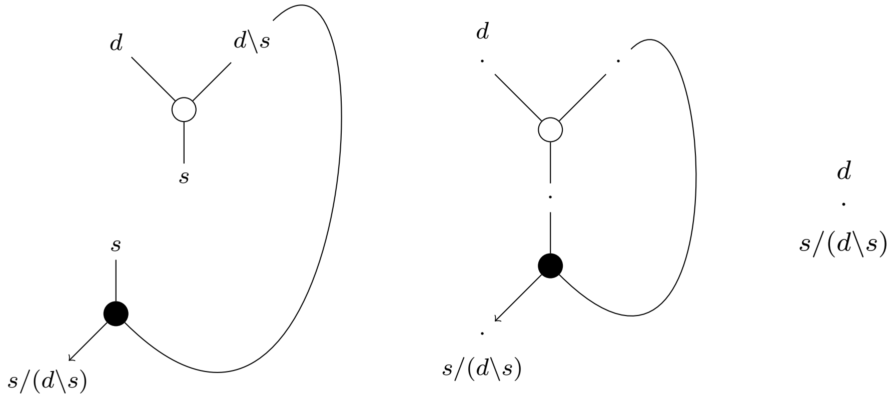
◇ **controlled right branch extraction** the dots abbreviate a tensor tree



contracts to

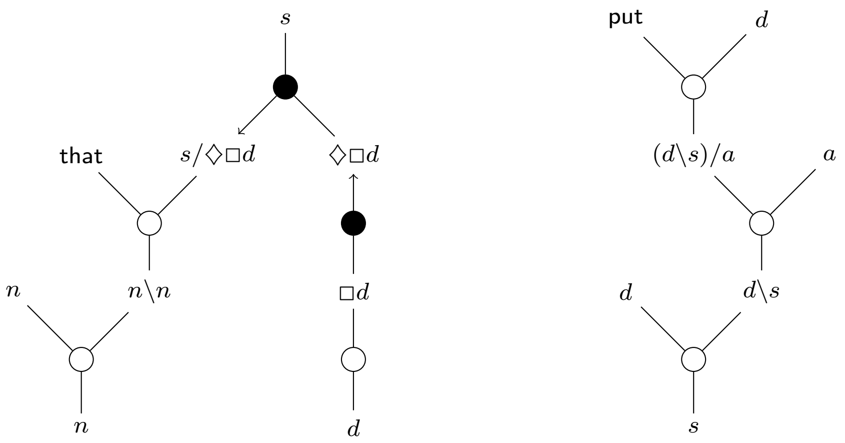


15. Illustration: logical contraction



Left: output unfolding for the formula $s/(d\backslash s)$; middle: abstract proof structure for $d \Rightarrow s/(d\backslash s)$; right: contraction to a tree.

16. Illustration: structural rewriting

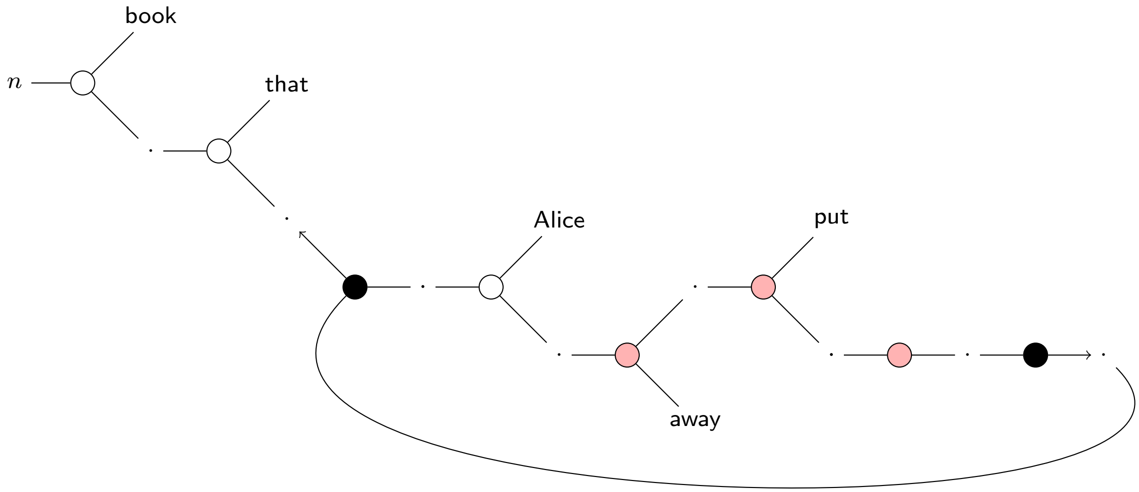


that :: $(n\backslash n)/(s/\Diamond\Box d)$ put :: $((d\backslash s)/a)/d$

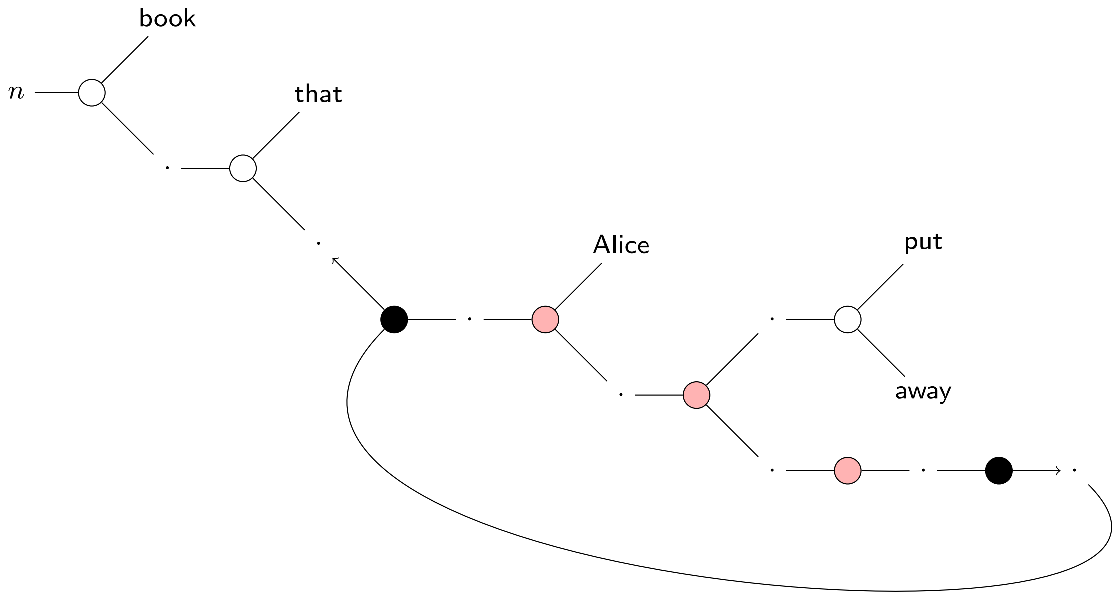
Derivation non-peripheral extraction with the right-branch extraction postulates, e.g. the n phrase 'book that Alice put ␣ away'. Red: tensor nodes involved in structural rewrites; blue: input configurations for logical contraction.

17. Input: abstract proof structure

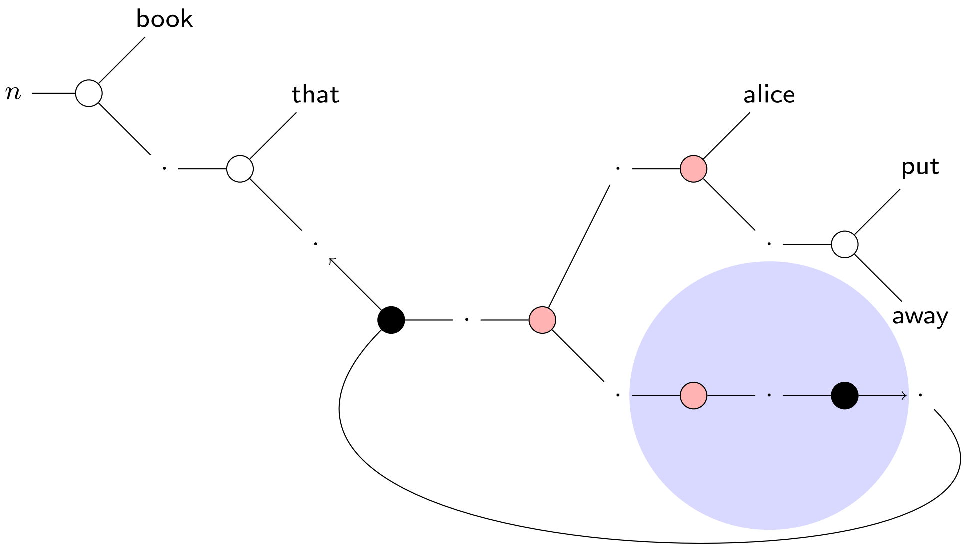
We first compute the stepwise reduction to a tensor tree, with the right-branch extraction postulates. After that, the one-step reduction, using the generalized contraction.



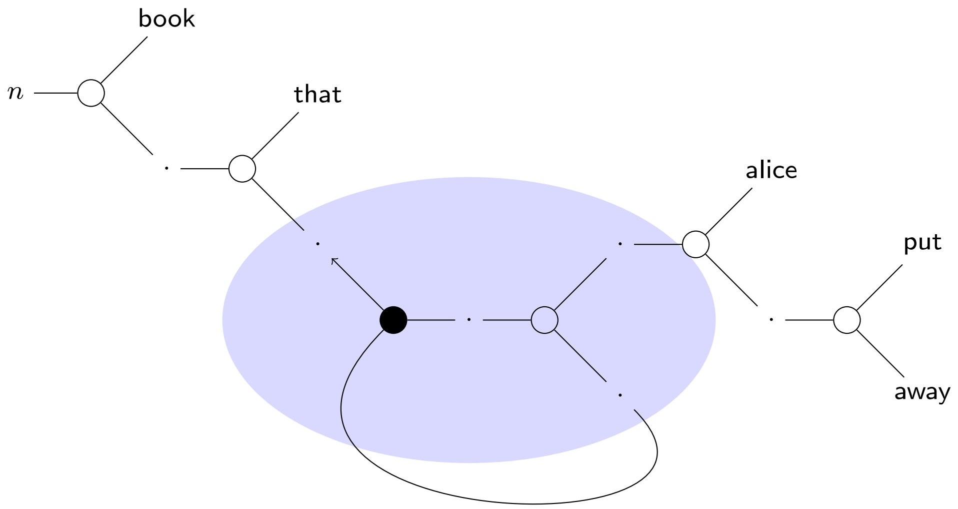
after structural rewriting: mixed commutativity



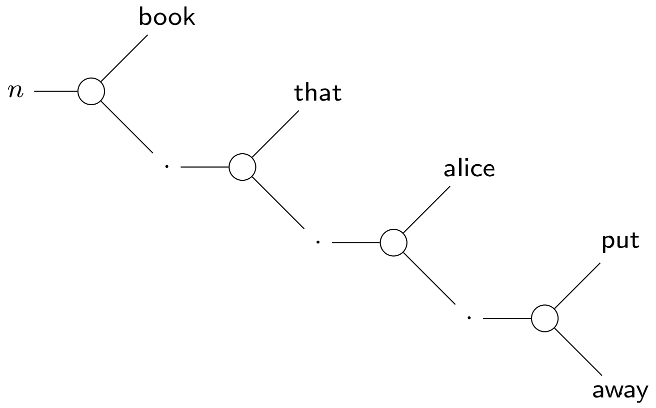
after structural rewriting: mixed associativity



after logical contraction: $\diamondsuit$

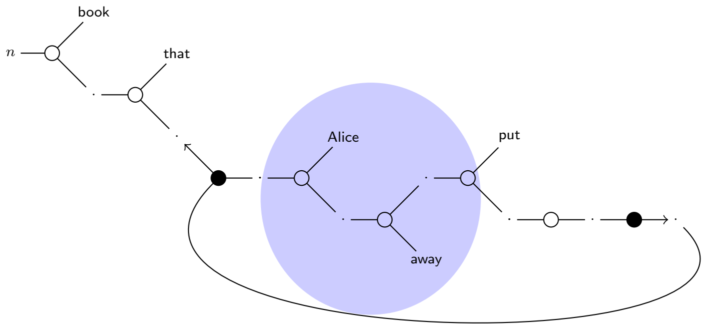


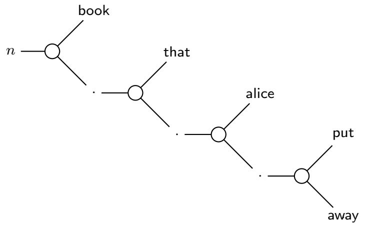
after logical contraction: $/$



$\rightsquigarrow$ the proof structure rewrites to a tensor tree.

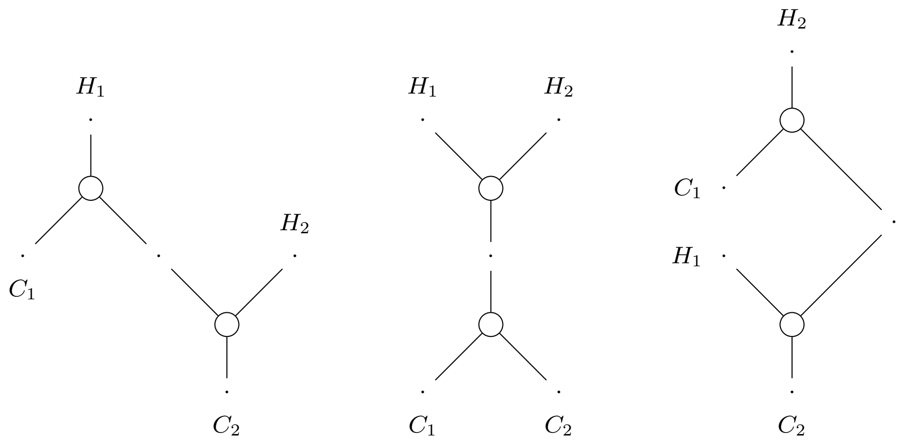
One-step generalized contraction





18. Structural rewriting: distributivity

Grishin's left distributivity laws (right distributivity: reflection along vertical axis)



$$C \oslash A \to D/B \quad \dashv \quad A \otimes B \to C \oplus D \quad \vdash \quad C \oslash B \to A \backslash D$$

$\rightsquigarrow$ long-distance contraction

19. Illustration: expressivity beyond CF

Time permitting the encoding of LTAG from Moot 2007 . . .

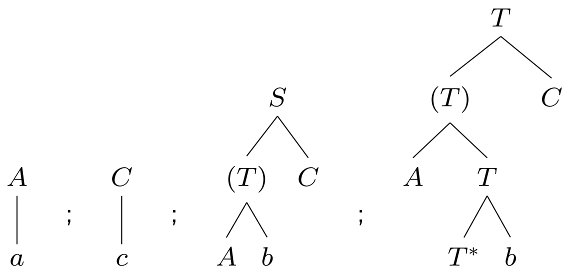
Ingredients:

- lowering $(A \oslash B) \obslash A \vdash B$
- action at a distance: long-distance contraction

20. Expressivity beyond CF: encoding LTAG

$$\{a^n b^n c^n \mid n > 0\}$$

LTAG Auxiliary tree on the right; adjunction node (T).

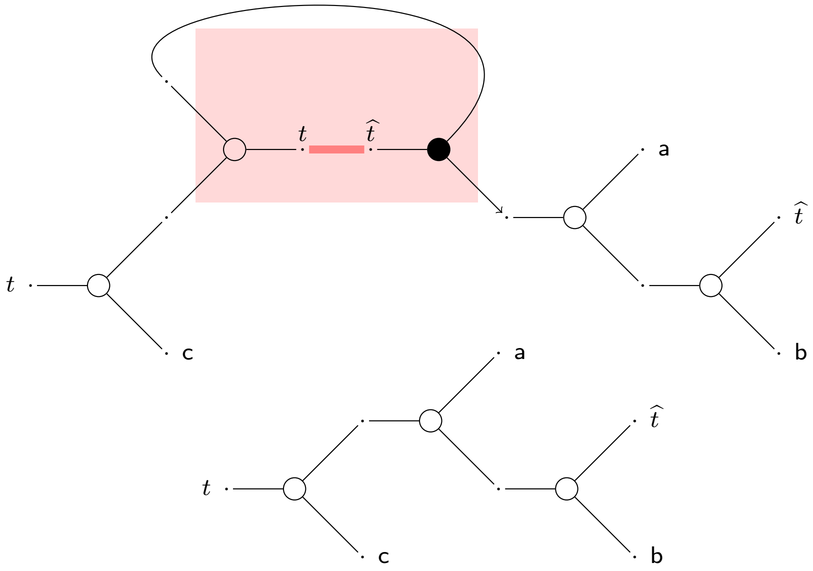


LG Type assignments with $\widehat{T}$ such that $\widehat{T} \vdash T$ but not v.v. $a :: A$, $c :: C$ and

$$b :: A\backslash((T \oslash (S/C)) \obslash \widehat{T}) \quad ; \quad b :: \widehat{T}\backslash(A\backslash((T \oslash (T/C)) \obslash \widehat{T}))$$

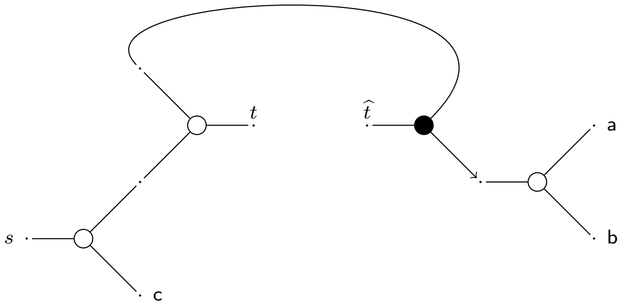
21. Deriving $aabbcc$: the auxiliary formula

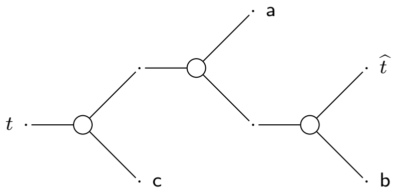
Step 1 For $n > 1$, we use $n-1$ times the auxiliary formula b :: $\widehat{t}\backslash(a\backslash((t\oslash(t/c))\obslash\widehat{t}))$. The nth use (no further adjunction) is internally connected, and contracts.



22. Deriving $aabbcc$: adjunction

Step 2 To obtain $aabbcc$, take the the initial graph for b :: $a\backslash((t \oslash (s/c)) \obslash \widehat{t})$

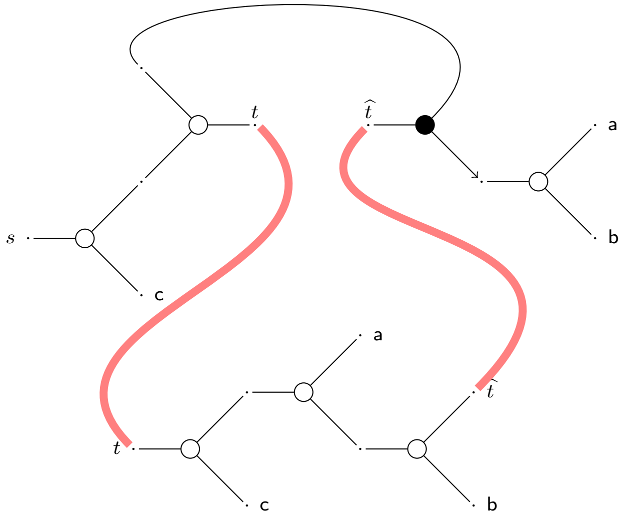




and adjoin into it the contracted auxiliary graph of the previous slide.

23. Deriving $aabbcc$: adjunction

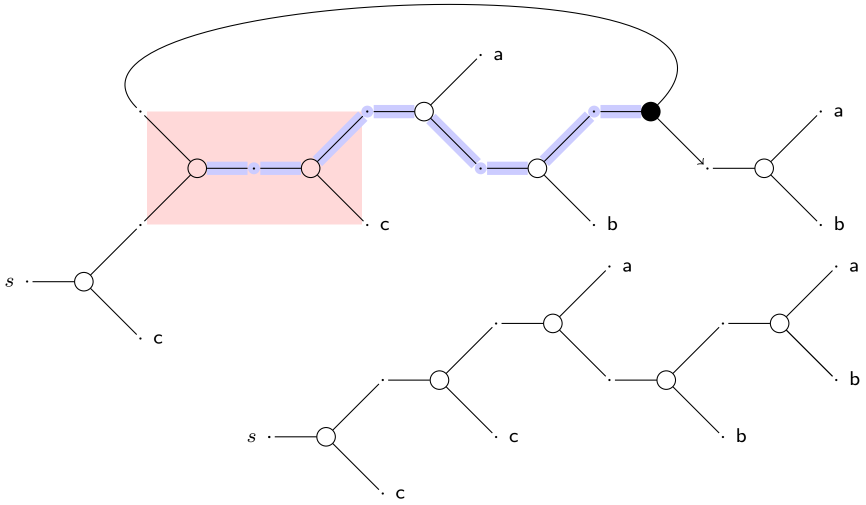
Step 2 To obtain $aabbcc$, take the the initial graph for b :: $a\backslash((t \oslash (s/c)) \obslash \widehat{t})$



and adjoin into it the contracted auxiliary graph of the previous slide.

24. Deriving $aabbcc$: distribution

Step 3 In the rectangle is the input configuration for distribution. The leftmost tensor can reach the matching cotensor link across the highlighted path: the graph is contractible to a tree.



25. Interpretation

$$\mathbf{Nets}^{\mathcal{A}}_{/,\otimes,\backslash,\oslash,\oplus,\obslash,\ldots} \xrightarrow{\lceil\cdot\rceil} \mathsf{LP}^{\mathcal{A}\cup\{\bot\}}_{\otimes,\cdot^{\bot}} \xrightarrow{\cdot^{\ell}} \mathsf{IL}^{\{e,t\}}_{\times,\to}$$

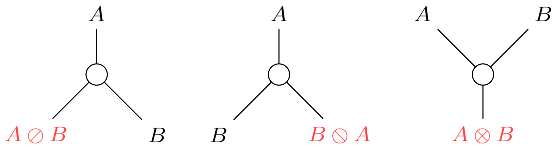
- Separating derivational and lexical semantics
  - $\lceil\cdot\rceil$ maps proof nets of syntactic source to target linear lambda terms
  - $\cdot^{\ell}$ maps lexical constants to possibly non-linear terms
- Target for compositional interpretation: continuation semantics
  - overall result of computation: distinguished atom $\bot$
  - linear products and restricted linear implication $A \multimap \bot$ (abbrev $A^{\bot}$)
  - context/state added as explicit parameter of meaning computation:

$$\lceil A\backslash B\rceil = (\lceil A\rceil \otimes \lceil B\rceil^{\bot})^{\bot}$$

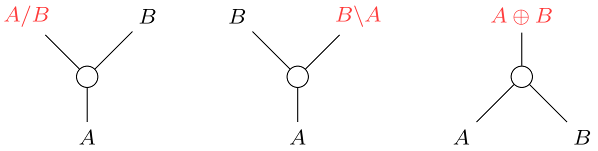
- Polarities: lean $\lceil\cdot\rceil$ translation avoiding 'bureaucratic redexes'

26. Polarities

- Partitioning of complex formulas
  - Positive: conclusion main formula of tensor links



  - Negative: hypothesis main formula of tensor links



- Atoms: arbitrary polarity bias (we'll assume: s negative, rest positive)

27. Interpretation: types

Atoms positive bias $\lceil p\rceil = p$; negative bias $\lceil p\rceil = p^{\perp}$.

Positive Conclusions of tensor links.

A	B	$\lceil A\otimes B\rceil$	$\lceil A\oslash B\rceil$	$\lceil B\obslash A\rceil$
$-$	$-$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil^{\perp}$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil$	$\lceil B\rceil\otimes\lceil A\rceil^{\perp}$
$-$	$+$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil^{\perp}$	$\lceil B\rceil^{\perp}\otimes\lceil A\rceil^{\perp}$
$+$	$-$	$\lceil A\rceil\otimes\lceil B\rceil^{\perp}$	$\lceil A\rceil\otimes\lceil B\rceil$	$\lceil B\rceil\otimes\lceil A\rceil$
$+$	$+$	$\lceil A\rceil\otimes\lceil B\rceil$	$\lceil A\rceil\otimes\lceil B\rceil^{\perp}$	$\lceil B\rceil^{\perp}\otimes\lceil A\rceil$

Negative Premises of tensor links.

A	B	$\lceil A/B\rceil$	$\lceil B\backslash A\rceil$	$\lceil A\oplus B\rceil$
$-$	$-$	$\lceil A\rceil\otimes\lceil B\rceil^{\perp}$	$\lceil B\rceil^{\perp}\otimes\lceil A\rceil$	$\lceil A\rceil\otimes\lceil B\rceil$
$-$	$+$	$\lceil A\rceil\otimes\lceil B\rceil$	$\lceil B\rceil\otimes\lceil A\rceil$	$\lceil A\rceil\otimes\lceil B\rceil^{\perp}$
$+$	$-$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil^{\perp}$	$\lceil B\rceil^{\perp}\otimes\lceil A\rceil^{\perp}$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil$
$+$	$+$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil$	$\lceil B\rceil\otimes\lceil A\rceil^{\perp}$	$\lceil A\rceil^{\perp}\otimes\lceil B\rceil^{\perp}$

28. Illustration

$\lceil \cdot \rceil$ interpretation of some source types. Atom bias is explicitly indicated. Composition with $\cdot^\ell$, for the typing of lexical constants, assuming

$$d^\ell = e \text{ (entities)},\ n^\ell = e \to t \text{ (sets)}, \text{ and } s^\ell = \perp^\ell = t \text{ (truth values)}$$

(The outermost $\cdot^\perp$ is for use of these types as hypotheses, cf below)

		$\lceil \cdot \rceil^\perp$	$(\lceil \cdot \rceil^\perp)^\ell$
a	$(d^+ \backslash s^-)/d^+$	$((d \otimes s^\perp) \otimes d)^\perp$	$((e \times (tt)) \times e) \to t$
b	d^+/n^+	$(d^\perp \otimes n)^\perp$	$((et) \times (et)) \to t$
c	$(d^+ \backslash s^-)/s^-$	$((d \otimes s^\perp) \otimes s^{\perp\perp})^\perp$	$((e \times (tt)) \times ((tt)t)) \to t$

Translations of the lexical constants:

a	finds	$\lambda\langle\langle x, c\rangle, y\rangle.(c\ (\textsc{find}^{eet}\ y\ x))$
b	some	$\lambda\langle x, y\rangle.(\exists\ \lambda z.(\wedge\ (y\ z)\ (x\ z)))$
c	thinks	$\lambda\langle\langle x, c\rangle, q\rangle.(c\ (\textsc{thinks}^{((tt)t)et}\ q\ x))$

29. Values, contexts, commands

We distinguish three types of subnets:

- Value
  - active hypotheses of tensor links
  - tensor subnets with conclusion (positive) main formula
- Context
  - active conclusions of tensor links;
  - tensor subnets with hypothesis (negative) main formula
- Command
  - the result of cutting a value subnet against a context subnet
  - extensions of the above with cotensor links

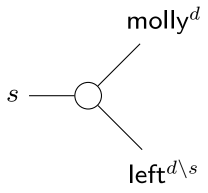
30. Proof net, composition graph

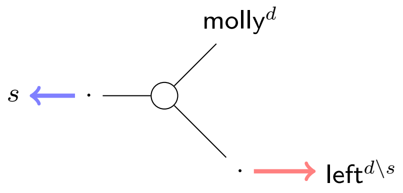
Interpretation of proofs is computed from an 'exploded view' of their proof net.

Composition graph Given a proof net P, the associated composition graph $cg(P)$ is obtained as follows.

- all vertices of P with formula label A are expanded into polarized axiom links
- axiom links connecting subnets of the same type (value or context) are collapsed.

Example Proof net and its exploded view.





31. Interpretation: proof nets

We simultaneously introduce a term language for the components of the composition graph of a net, and their interpretation as linear lambda terms:

- Atoms: hypotheses $x, y \ldots$, conclusions $\alpha, \beta \ldots$
- Axiom links: commands, focusing
- Logical links: tensor/cotensor

Hypotheses, conclusions Interpretation: add an outermost negation for the translation of positive (negative) conclusion (hypothesis) vertices.

pol(A)	$\lceil x^A \rceil$	$\lceil \alpha^A \rceil$
$+$	$\widetilde{x} : \lceil A \rceil$	$\widetilde{\alpha} : \lceil A \rceil^{\perp}$
$-$	$\widetilde{x} : \lceil A \rceil^{\perp}$	$\widetilde{\alpha} : \lceil A \rceil$

32. Axiom links

Commands (red): function application; focusing (blue): function abstraction.

Positive

$$\begin{tikzcd}
v \arrow[d] & \widetilde{\mu} x.c \\
\cdot & \cdot \arrow[u] \\
\alpha & x
\end{tikzcd}$$

$$\lceil \langle v \upharpoonright \alpha \rangle \rceil = (\widetilde{\alpha}\ \lceil v \rceil) \qquad \lceil \widetilde{\mu} x.c \rceil = \lambda \widetilde{x}.\lceil c \rceil$$

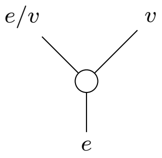
Negative

$$\begin{tikzcd}
x & \alpha \\
\cdot & \cdot \arrow[d] \\
e \arrow[u] & \mu\alpha.c
\end{tikzcd}$$

$$\lceil \langle x \upharpoonleft e \rangle \rceil = (\widetilde{x}\ \lceil e \rceil) \qquad \lceil \mu\alpha.c \rceil = \lambda \widetilde{\alpha}.\lceil c \rceil$$

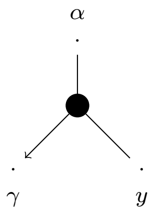
33. Logical links

Tensor (focused) All tensor links are interpreted by linear pairs. For example (A/B):



$$\lceil e/v \rceil = \langle \lceil e \rceil, \lceil v \rceil \rangle$$

Cotensor (neutral) All cotensor links are interpreted by the case construct:



$$\left\lceil \frac{\alpha\ y}{\gamma}.c \right\rceil = \textbf{case } \widetilde{\gamma} \textbf{ of } \langle \widetilde{\alpha}, \widetilde{y} \rangle \textbf{ in } \lceil c \rceil$$

Reduction **case** $\langle t, u \rangle$ **of** $\langle x, y \rangle$ **in** $v \implies v[t/x, u/y]$

34. Assembling the net from its composition graph

Components Maximal tensor subnets of $cg(P)$ with a single main formula. Mark these tensor links as visited.

Assembling the net While $cg(P)$ contains unvisited links do the following:

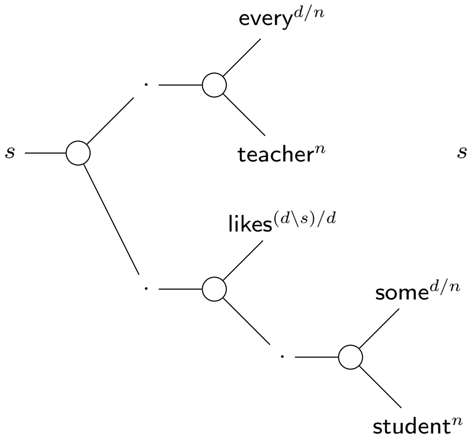
1. follow an unvisited command link attached to a previously calculated maximal subnet, forming a correct command subnet;
2. for each cotensor link with both active formulas attached to the current command subnet, pass to its main formula, forming a new command; repeat this step until no such cotensor links remain attached;
3. follow a focusing link to a new vertex, forming a larger value or context subnet.

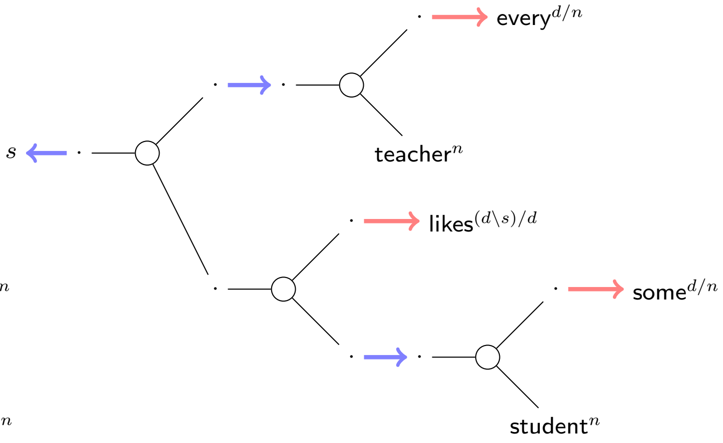
Non-determinism The choice of the subnet to start from is free, but one is committed to that choice, and traversal has to continue from that subnet. A wrong choice may lead to failure to assemble the complete net.

35. Illustrations: scope construal

- Local ambiguity within one sentence
- Local vs non-local scope from embedded clauses
- 'seek' versus 'find'
- scope-taking in non-sentential domains

Scope ambiguity. Proof net, exploded view, components

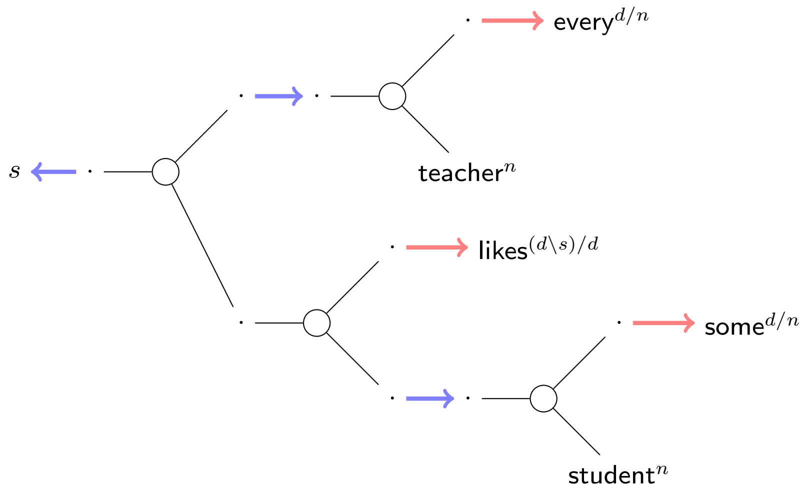




Three components: maximal tensor subnets with a single main formula.

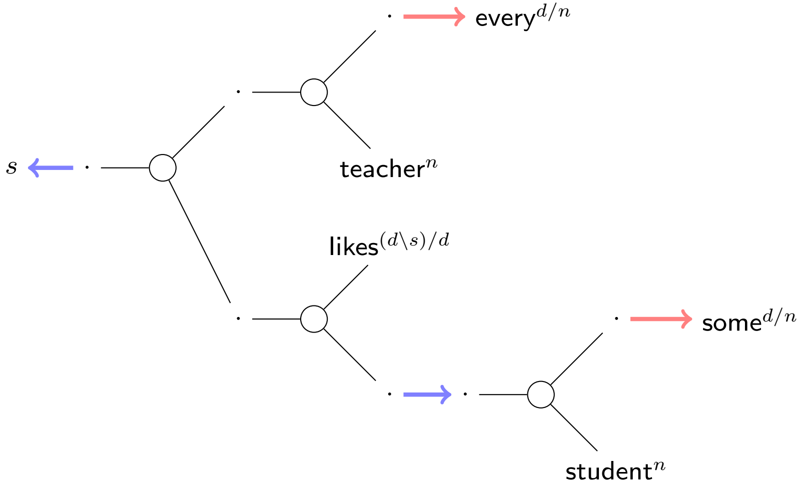
For the direction of the focusing (blue) arrows: recall that s has negative bias, the other atoms are positively biased.

Deriving the inverted scope reading $\exists/\forall$

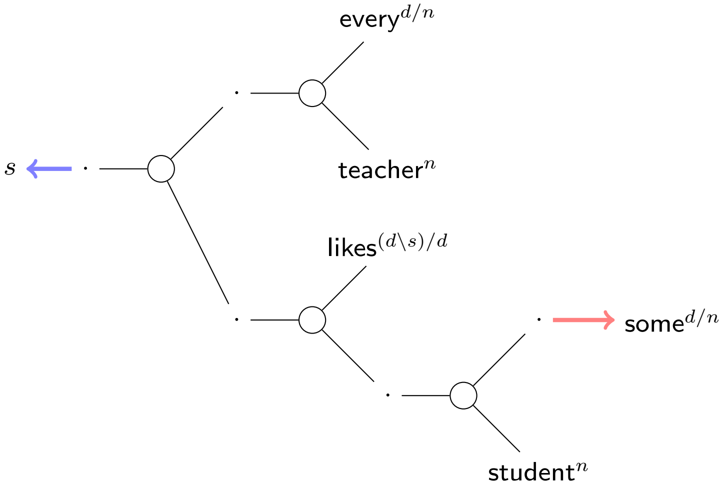


3×3 possible command/focusing matchings; only two lead to a successful assembling of the net. We compute the inverted scope reading.

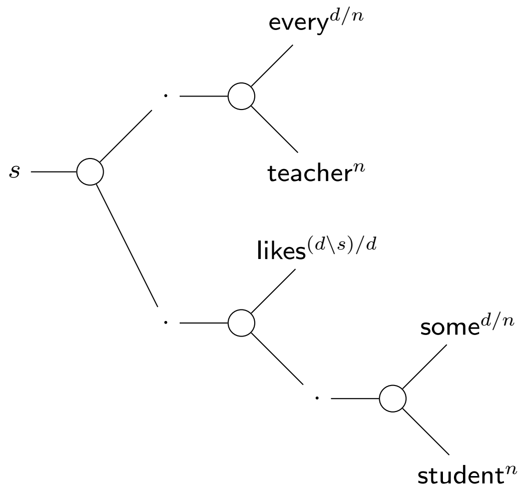
Deriving the inverted scope reading $\exists/\forall$: step 1



Deriving the inverted scope reading $\exists/\forall$: step 2



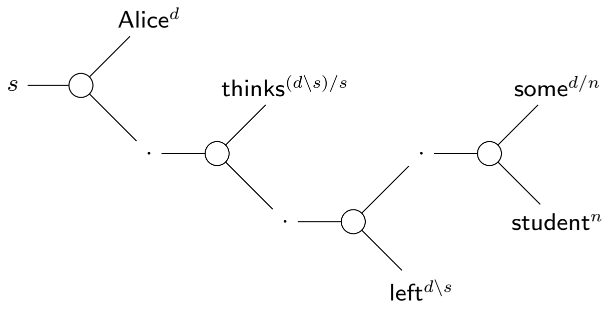
Deriving the inverted scope reading ∃/∀: step 3

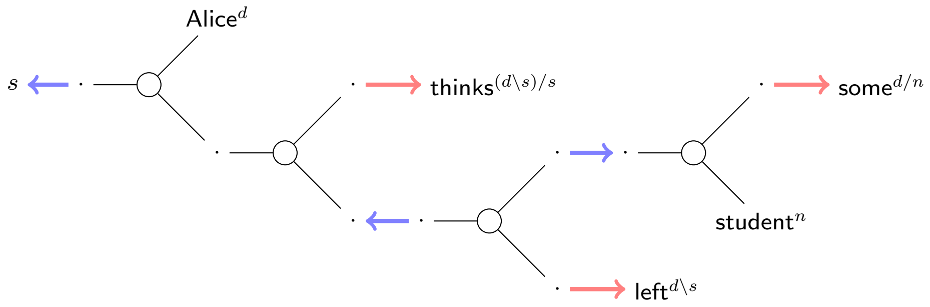


$$\mathsf{Net} = \quad \mu\alpha.\langle\ \mathsf{some} \upharpoonleft (\widetilde{\mu} y.\langle\ \mathsf{every} \upharpoonleft (\widetilde{\mu} z.\langle\ \mathsf{likes} \upharpoonleft ((z \backslash \alpha) / y)\ \rangle / \mathsf{teacher})\ \rangle / \mathsf{student})\ \rangle$$

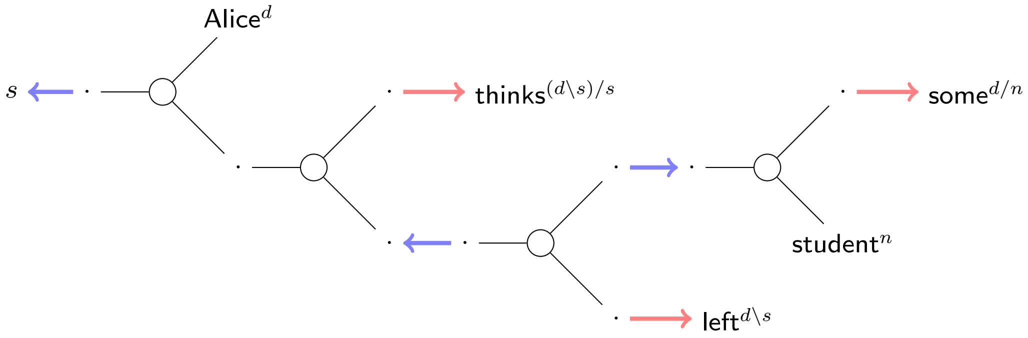
$$\lceil \mathsf{Net} \rceil = \quad \lambda\widetilde{\alpha}.(\mathsf{some}^{\ell}\ \langle \lambda\widetilde{y}.(\mathsf{every}^{\ell}\ \langle \lambda\widetilde{z}.(\mathsf{likes}^{\ell}\ \langle\langle \widetilde{z}, \widetilde{\alpha}\rangle, \widetilde{y}\rangle), \mathsf{teacher}^{\ell}\rangle), \mathsf{student}^{\ell}\rangle)$$

Embedded clause. Proof net, exploded view



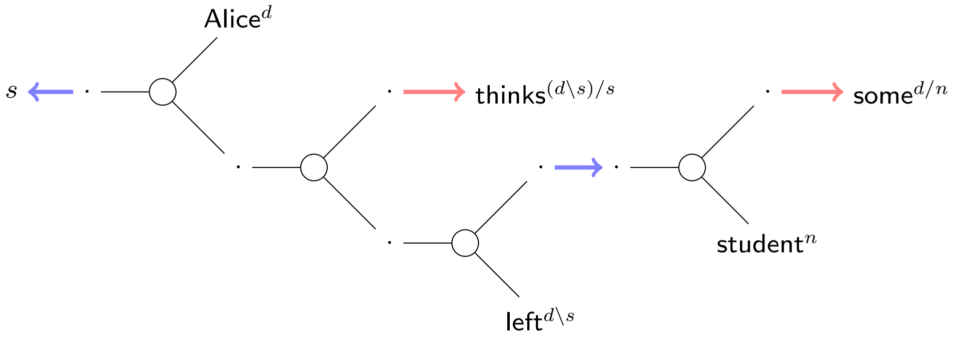


Deriving the wide scope ∃ reading

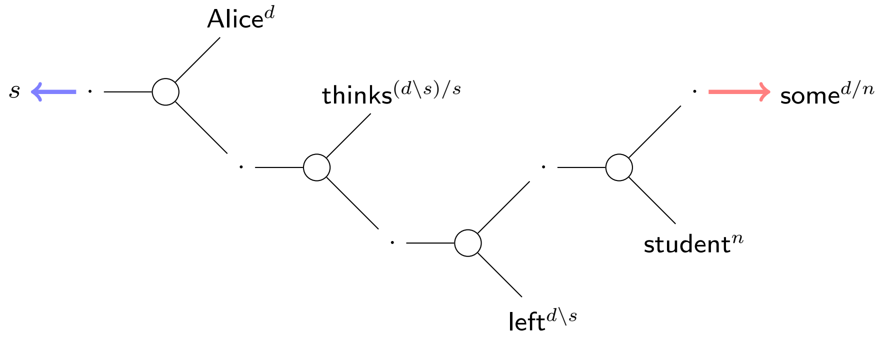


Again, 3×3 possible command/focusing matchings, two of which are successful. We compute the non-local ∃ reading.

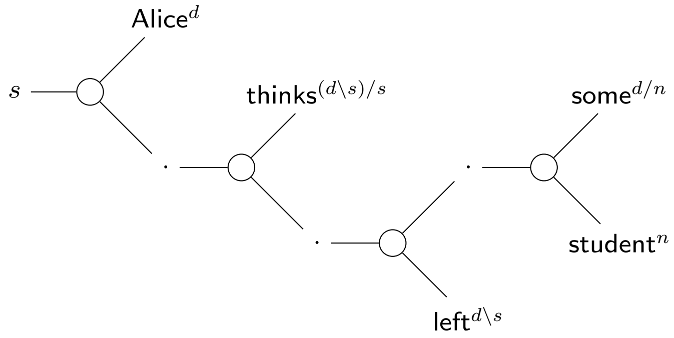
Deriving the wide scope ∃ reading: step 1



Deriving the wide scope $\exists$ reading: step 2



Deriving the wide scope ∃ reading: step 3



$$\text{Net} = \quad \mu\alpha.\langle\ \text{some} \upharpoonleft (\widetilde{\mu} y.\langle\ \text{thinks} \upharpoonleft ((\text{alice} \backslash \alpha) \,/\, \mu\gamma.\langle\ \text{left} \upharpoonleft (y \backslash \gamma)\ \rangle)\ \rangle \,/\, \text{student})\ \rangle$$

$$\lceil \text{Net} \rceil = \quad \lambda\widetilde{\alpha}.(\text{some}^{\ell}\ \langle \lambda\widetilde{y}.(\text{thinks}^{\ell}\ \langle\langle \text{alice}^{\ell}, \widetilde{\alpha}\rangle, \lambda\widetilde{\gamma}.(\text{left}^{\ell}\ \langle\widetilde{y}, \widetilde{\gamma}\rangle)\rangle), \text{student}^{\ell}\rangle)$$

36. Discussion

- scope construal (covert displacement) obtainable in pure application **AB**
- what about overt displacement?
- shifts via (dual) Galois connected operations (MM 2010)
  - find: $(d\backslash s)/d$ versus seek: $(d\backslash s)/^{\mathbf{0}}d^{\mathbf{0}}$
  - scope in non-sentential contexts, e.g. picture of: $n/^{\mathbf{0}}d^{\mathbf{0}}$
- enforcing locality constraints?
- link with Morrill's **D** $\uparrow, \downarrow$ via ACG-style string interpretation (MM 2009)

$$\lceil d/n \rceil^{\sigma} = \sigma \multimap (\sigma \multimap \sigma) \multimap \sigma$$

- . . .