Computing Word Meanings by Interpolation

Makoto Kanazawa
University of Tokyo

Abstract
I outline a natural algorithm for solving a central problem in the task of learning word-to-meaning mappings, as formulated by Siskind (1996, 2000) and extended to the typed lambda calculus setting by Kanazawa (2001). The algorithm is based on a new syntactical method for proving the Interpolation Theorem for the implicational fragment of intuitionistic propositional logic.

A central problem in the task of learning word-to-meaning mappings, as formulated by Siskind (1996, 2000), can be illustrated by the following example. The learner knows from the outset that the meaning of each word is represented by some first-order term with zero or more free variables, and the sentence meaning is composed from the meanings of the component words by performing a sequence of substitutions in a suitable order. Suppose that the learner has already inferred from evidence presented so far that the meaning of *lifted*, whatever it is, is built up from three symbols CAUSE, GO, and UP, together with some number of variables. Suppose that the learner is now given the information that the meaning of *John lifted Mary* is represented by the first-order term

$$\mathrm{CAUSE}(\mathbf{John}, \mathrm{GO}(\mathbf{Mary}, \mathrm{UP})).$$

The available evidence suffices to uniquely pin down the meaning of *lifted* to

$$\mathrm{CAUSE}(x, \mathrm{GO}(y, \mathrm{UP})),$$

which can be computed by a simple algorithm, even if the meanings of *John* and *Mary* may still be indeterminate.

Following Kanazawa (2001), I generalize this problem to the typed lambda calculus setting as follows. The meaning of each word is represented by a closed λI-term (with one or more constant symbols). The meaning of a sentence is obtained by plugging the meanings of the words in a suitable *meaning recipe*, represented by a *linear* (or *BCI*) λ-term (containing one free variable for each of the words) of type t and then computing the β-normal form of the resulting λ-term. The central problem now becomes:

Mapping Problem. Given a closed λI-term N of type t in β-normal form containing constant symbols $c_1^{A_1}, \dots, c_m^{A_m}, d_1^{B_1}, \dots, d_n^{B_n}$ ($m \geq 1, n \geq 0$), find a closed λI-term M (of type E) satisfying the following conditions:

- The constant symbols appearing in M are $c_1^{A_1}, \dots, c_m^{A_m}$;

- There is a λI-term $P[z^E]$ of type t with z^E as its only free variable such that $P[M] \twoheadrightarrow_\beta N$.

In the above formulation, N is the meaning of a sentence, $\{c_1^{A_1}, \dots, c_m^{A_m}\}$ is the set of constant symbols that are in one of the words in the sentence, and $P[z^E]$ is supposed to be the result of combining the meaning recipe for the sentence (a λI-term) and the meanings of the remaining words (linear λ-terms). ($P[z^E]$ is not just any λI-term—for example, it must be a linear λ-term in case $n = 0$, but this point will be ignored.)

An instance of the Mapping Problem may be given by the λI-term

$$\boldsymbol{\forall}(\lambda x^e.\boldsymbol{\to}(\mathbf{thing}\ x)(\mathbf{give}\ \mathbf{Bill}\ x\ \mathbf{John})) \tag{1}$$

with its set of constants divided into $\{\boldsymbol{\forall}, \boldsymbol{\to}, \mathbf{thing}\}$ and $\{\mathbf{give}, \mathbf{Bill}, \mathbf{John}\}$. (Here, $\boldsymbol{\forall}$, $\boldsymbol{\to}$, $\mathbf{thing}$, $\mathbf{give}$, $\mathbf{Bill}$, and $\mathbf{John}$ are constants of type $(e \to t) \to t$, $t \to t \to t$, $e \to e \to e \to t$, e, and e, respectively.) The above λ-term is supposed to represent the meaning of *John gave Bill everything*, and the problem is to build a correct meaning for *everything* out of $\boldsymbol{\forall}, \boldsymbol{\to}, \mathbf{thing}$.

Unlike in the first-order case of Siskind, the Mapping Problem in the general higher-order setting has many solutions of varying strengths. The following are some of the solutions, along with their types, to the above instance of the Mapping Problem:

$$\begin{aligned} \lambda w^{((e\to e\to t)\to t)\to(t\to t\to t)\to(e\to t)\to t}.w\ (\lambda u^{e\to e\to t}.\boldsymbol{\forall}(\lambda x^e.uxx))\ \boldsymbol{\to}\ \mathbf{thing} \\ : (((e \to t) \to t) \to (t \to t \to t) \to (e \to t) \to t) \to t \end{aligned} \tag{2}$$

$$\lambda u^{e\to t}.\boldsymbol{\forall}(\lambda x^e.\boldsymbol{\to}(\mathbf{thing}\ x)(ux)) : (e \to t) \to t \tag{3}$$

$$\lambda v^{e\to e\to t}y^e.\boldsymbol{\forall}(\lambda x^e.\boldsymbol{\to}(\mathbf{thing}\ x)(vxy)) : (e \to e \to t) \to e \to t \tag{4}$$

$$\lambda w^{e\to e\to e\to t}z^e y^e.\boldsymbol{\forall}(\lambda x^e.\boldsymbol{\to}(\mathbf{thing}\ x)(wzxy)) : (e \to e \to e \to t) \to e \to e \to t \tag{5}$$

The solutions (2)–(5) are linearly ordered in terms of their strength, with (2) as the strongest. The notion of 'strength' in question is given by the following pre-order on terms, which I call the *definability ordering* (Kanazawa 2001).

Definition 1. Let M^A and N^B be closed λ-terms (with constants) in β-normal form. N^B is *BCI-definable* in terms of M^A (written $N^B \leq M^A$) if and only if there is a linear λ-term $P^B[z^A]$ without constants whose only free variable is z^B such that $P^C[M^A] \twoheadrightarrow_\beta N^B$.

Note that if $N^B \leq M^A$, the set of constants that occur in M^A is the same as the set of constants that occur in N^B.

Proposition 2. *Let M^A and N^B be closed λI-terms such that $N^B \leq M^A$. If M^A solves an instance of the Mapping Problem, then N^B solves it, too.*

Continuing with the example (1), which solution does one want? One certainly does not want the strongest (2), which wildly overgenerates: for instance, it can be used to produce

$$\boldsymbol{\forall}(\lambda x^e.\boldsymbol{\to}(\mathbf{give}\ \mathbf{Bill}\ x\ \mathbf{John})(\mathbf{thing}\ x)) \tag{6}$$

as a meaning for *John gave Bill everything*.[1] Nor does one want the weakest (5), which cannot even generate the meaning for a sentence like *John saw everything*. The weakest meaning cannot be a principled solution because in general assigning the weakest meaning to one word in a sentence is incompatible with assigning the weakest meaning to another word in the same sentence. Kanazawa's (2001) algorithm finds the solution (4), which is too weak to generate the meaning for an intransitive sentence like *everything disappeared*. In this paper, I outline an algorithm that finds (3), which has the simplest type and happens to be the conventionally assumed meaning for *everything*.

A solution to the Mapping Problem which has a simplest type can be found by a syntactical proof of the *Interpolation Theorem* for intuitionistic propositional logic, which states:

Interpolation Theorem. *If a sequent $A_1, \dots, A_m, B_1, \dots, B_n \Rightarrow C$ is provable, then there are provable sequents $A_1, \dots, A_m \Rightarrow E$ and $E, B_1, \dots, B_n \Rightarrow C$ such that all propositional variables in C occur both in $A_1, \dots, A_m$ and in $B_1, \dots, B_n \Rightarrow C$.*

There are two well-known syntactical methods for proving this theorem. Maehara's method (see Troelstra and Schwichtenberg 2000) works on sequent calculus and Prawitz's (1965) method works on natural deduction. Both methods take a (cut-free or normal) proof $\mathcal{D}\colon A_1, \dots, A_m, B_1, \dots, B_n \Rightarrow C$ as input and compute two proofs $\mathcal{D}_1\colon A_1, \dots, A_m \Rightarrow E$ and $\mathcal{D}_0\colon E, B_1, \dots, B_n \Rightarrow C$ which satisfy the condition in the theorem. Crucially for our purposes, the proofs $\mathcal{D}_1$ and $\mathcal{D}_0$ found by these methods in fact satisfy much stronger conditions. Let $N^C[x_1^{A_1}, \dots, x_m^{A_m}, y_1^{B_1}, \dots, y_n^{B_n}], M^E[x_1^{A_1}, \dots, x_m^{A_m}], P^C[z^E, y_1^{B_1}, \dots, y_n^{B_n}]$ be the λ-terms corresponding to the proofs $\mathcal{D}, \mathcal{D}_1, \mathcal{D}_0$, respectively. Then we have:

(i) $P^C[M^E[x_1^{A_1}, \dots, x_m^{A_m}], y_1^{B_1}, \dots, y_n^{B_n}] \twoheadrightarrow_\beta N^C[x_1^{A_1}, \dots, x_m^{A_m}, y_1^{B_1}, \dots, y_n^{B_n}]$;

(ii) No two occurrences of the same propositional variable inside E are *linked* in $\mathcal{D}_1$ or $\mathcal{D}_0$.

The condition (i) is pointed out by Čubrić (1994) for Prawitz's method, and the condition (ii) is stated by Carbone (1997), who works on sequent calculus. The notion of *links* referred to in (ii) is easiest to explain with reference to cut-free sequent derivations: two occurrences of a variable in the endsequent are linked if they originated opposite to each other in an initial sequent. The condition (ii) implies[2] that each occurrence of a propositional variable inside E has a counterpart linked to it outside E, and the usual condition for an interpolant follows.

Let us deviate from standard usage and call a term $M^E[x_1^{A_1}, \dots, x_m^{A_m}]$ an *interpolant* to the term $N^C[x_1^{A_1}, \dots, x_m^{A_m}, y_1^{B_1}, \dots, y_n^{B_n}]$ if there is a term $P^C[z^E, y_1^{B_1}, \dots, y_n^{B_n}]$ such that M, N, P satisfy the above conditions (i) and (ii).

[1] One can indeed write a grammar (with syntax and semantics) where *everything* has the meaning (2) and *John gave Bill everything* is ambiguous between (1) and (6). One can do this with a Lambek categorial grammar if *everything* is allowed to be syntactically ambiguous. With a *lambda grammar* (Muskens 2001, to appear), one can even make *everything* syntactically unambiguous.

[2] The condition (ii) is stated for the relevance logic R, whose implicational fragment corresponds to the λI-calculus. It needs to be strengthened for intuitionistic logic.

Replacing the constants $\boldsymbol{\forall}, \boldsymbol{\to}, \mathbf{thing}, \mathbf{give}, \mathbf{Bill}, \mathbf{John}$ in (1)–(5) by free variables $x_1^{(e\to t)\to t}, x_2^{t\to t\to t}, x_3^{e\to t}, y_1^{e\to e\to e\to t}, y_2^e, y_3^e$, respectively, we see that (3) is an interpolant to (1) while (2), (4), (5) are not.

There are two complications, however. First, the Interpolation Theorem in fact does not hold in the form stated above for the implicational fragment of intuitionistic logic, which corresponds to the simply typed λ-calculus. Even if $A_1, \dots, A_m, B_1, \dots, B_n \Rightarrow C$ is a sequent in the implicational fragment, the interpolation formula E may have to contain $\wedge$. Moreover, Maehara's and Prawitz's methods actually insert $\bot$ and $\wedge$ in places where they are not necessary; as a result, they produce interpolation formulas more complex than $(e \to t) \to t$ in the case at hand.

A way of circumventing this difficulty is to use a sequence of formulas $E_1, \dots, E_m$ in place of a single formula E in the statement of the theorem (see Wroński 1984 and Pentus 1997). Both standard methods can be easily modified to accommodate this change. The modified methods produce (3) when given (1) as input.

A second complication is that interpolants in the sense of (i), (ii) are not unique in general. Maehara's method (in the modified form) finds possibly different interpolants for different sequent derivations corresponding to the same λ-term; however, not all interpolants are found in this way. Prawitz's method finds one particular interpolant, but there is no good way of characterizing this interpolant except for the fact that it is the one found by Prawitz's method. In particular, a *strongest* interpolant (in the sense that its type implies the types of all others) is sometimes missed by both these methods.

In a paper in preparation, I describe an algorithm for computing a strongest interpolant. The algorithm is similar to the (modified) Prawitz method in that it works by induction on natural deduction proofs, but it is different in that in my method, assumptions never switch classes in the partition of the assumptions into two classes during the course of the induction, like they do in the Prawitz method.

For example, given a natural deduction proof

$$\dfrac{x_1 : (p_4 \to p_7 \to p_9) \to p_{10} \qquad \dfrac{\dfrac{\dfrac{y_1 : p_8 \to p_9 \qquad \dfrac{\dfrac{x_2 : (p_1 \to p_6) \to p_7 \to p_8 \qquad \dfrac{\dfrac{y_2 : p_5 \to p_6 \qquad \dfrac{\dfrac{x_3 : p_3 \to p_4 \to p_5 \qquad \dfrac{y_3 : p_2 \to p_3 \qquad \dfrac{x_4 : p_1 \to p_2 \quad w : p_1}{p_2}{\to}E}{p_3}{\to}E}{p_4 \to p_5}{\to}E \qquad u : p_4}{p_5}{\to}E}{p_6}{\to}E}{p_1 \to p_6}{\to}I, w}{p_7 \to p_8}{\to}E \qquad v : p_7}{p_8}{\to}E}{p_9}{\to}E}{p_7 \to p_9}{\to}I, v}{p_4 \to p_7 \to p_9}{\to}I, u}{p_{10}}{\to}E$$

my algorithm produces

$$\dfrac{\dfrac{z_1 : ((p_2 \to p_6) \to p_8) \to (p_3 \to p_5) \to p_9 \quad \dfrac{\dfrac{\dfrac{\dfrac{x_2 : (p_1 \to p_6) \to p_7 \to p_8 \quad \dfrac{\dfrac{z_2 : p_2 \to p_6 \quad \dfrac{x_4 : p_1 \to p_2 \quad w : p_1}{p_2}\,{\to}E}{p_6}\,{\to}E}{p_1 \to p_6}\,{\to}I, w}{p_7 \to p_8}\,{\to}E \quad v : p_7}{p_8}\,{\to}E}{(p_2 \to p_6) \to p_8}\,{\to}I, z_2}{(p_3 \to p_5) \to p_9}\,{\to}E \quad \dfrac{\dfrac{\dfrac{x_3 : p_3 \to p_4 \to p_5 \quad z_3 : p_3}{p_4 \to p_5}\,{\to}E \quad u : p_4}{p_5}\,{\to}E}{p_3 \to p_5}\,{\to}I, z_3}{p_9}\,{\to}E$$

$$\dfrac{\dfrac{x_1 : (p_4 \to p_7 \to p_9) \to p_{10} \quad \dfrac{\dfrac{p_9}{p_7 \to p_9}\,{\to}I, v}{p_4 \to p_7 \to p_9}\,{\to}I, u}{p_{10}}\,{\to}E}{(((p_2 \to p_6) \to p_8) \to (p_3 \to p_5) \to p_9) \to p_{10}}\,{\to}I, z_1$$

while the Prawitz method produces

$$\dfrac{\dfrac{\dfrac{x_2 : (p_1 \to p_6) \to p_7 \to p_8 \quad \dfrac{\dfrac{z_2 : ((p_2 \to p_3) \to p_5) \to p_6 \quad \dfrac{\dfrac{\dfrac{x_3 : p_3 \to p_4 \to p_5 \quad \dfrac{z_3 : p_2 \to p_3 \quad \dfrac{x_4 : p_1 \to p_2 \quad w : p_1}{p_2}\,{\to}E}{p_3}\,{\to}E}{p_4 \to p_5}\,{\to}E \quad u : p_4}{p_5}\,{\to}E}{(p_2 \to p_3) \to p_5}\,{\to}I, z_3}{p_6}\,{\to}E}{p_1 \to p_6}\,{\to}I, w}{p_7 \to p_8}\,{\to}E \quad v : p_7}{p_8}\,{\to}E}{(((p_2 \to p_3) \to p_5) \to p_6) \to p_8}\,{\to}I, z_2$$

$$\dfrac{z_1 : ((((p_2 \to p_3) \to p_5) \to p_6) \to p_8) \to p_9 \quad (((p_2 \to p_3) \to p_5) \to p_6) \to p_8}{p_9}\,{\to}E$$

$$\dfrac{\dfrac{x_1 : (p_4 \to p_7 \to p_9) \to p_{10} \quad \dfrac{\dfrac{p_9}{p_7 \to p_9}\,{\to}I, v}{p_4 \to p_7 \to p_9}\,{\to}I, u}{p_{10}}\,{\to}E}{(((((p_2 \to p_3) \to p_5) \to p_6) \to p_8) \to p_9) \to p_{10}}\,{\to}I, z_1$$

which is strictly weaker. It remains to be seen whether there are instances of the Mapping Problem involving linguistically natural word meanings for which the choice among different interpolants is significant.

The decision to base a learning algorithm on interpolation has repercussions on the question of what λ-terms can be possible word meanings. The algorithm can only find λ-terms $M^E[x_1^{A_1}, \dots, x_m^{A_m}]$ that are interpolants to themselves. In addition to (2), (4), (5), this rules out otherwise innocent-looking

$$\lambda u^{e\to t}.u\ \mathbf{John}^e, \quad \lambda v^{(e\to t)\to t}u^{e\to t}.\mathbf{seem}^{t\to t}(vu), \quad \lambda u^{e\to t}x^e.\mathbf{try}^{t\to e\to t}(ux)x,$$
$$\lambda v^{(e\to t)\to t}x^e.v(\lambda y^e.\mathbf{find}^{e\to e\to t}\ y\ x),$$

in favor of

$$\mathbf{John}^e, \quad \mathbf{seem}^{t\to t}, \quad \mathbf{try}^{t\to e\to t}, \quad \mathbf{find}^{e\to e\to t},$$

while allowing both of

$$\lambda v^{(e\to t)\to t}x^e.\mathbf{try}^{t\to e\to t}(v(\lambda y^e.\mathbf{find}^{e\to e\to t}\ y\ x))x, \quad \lambda y^e x^e.\mathbf{try}^{t\to e\to t}(\mathbf{find}^{e\to e\to t}\ y\ x)x.$$

(I ignore intensionality for simplicity.) While this may not exactly be the restriction one wants, some such restriction on the λI-terms that can represent word meanings should add a welcome new perspective to type-logical semantics.

Even with the restriction to ‘self-interpolants’, my algorithm may not necessarily find the correct meaning. For instance, if the meaning of *seeks* is represented by an unanalyzed constant $\mathbf{seek}^{((e\to t)\to t)\to e\to t}$ rather than in terms of $\mathbf{try}^{t\to e\to t}$ and $\mathbf{find}^{e\to e\to t}$, from the *de re* reading of *John seeks a unicorn*

$$\exists^{(e\to t)\to t}(\lambda y^{e}.\wedge(\mathbf{unicorn}^{e\to t}\, y)(\mathbf{seek}^{((e\to t)\to t)\to e\to t}(\lambda u^{e\to t}.uy)\mathbf{John}^{e})),$$

the algorithm finds Montague’s

$$\mathbf{seek}_{*}^{e\to e\to t} = \lambda y^{e}.\mathbf{seek}^{((e\to t)\to t)\to e\to t}(\lambda u^{e\to t}.uy),$$

rather than $\mathbf{seek}^{((e\to t)\to t)\to e\to t}$, which may not be a desirable result. This and many other problems must be dealt with in order to achieve a successful generalization of Siskind’s elegant approach to type-logical semantics. I will pursue this goal further in future work.

Acknowledgment

This research has been supported by the Kayamori Foundation of Informational Science Advancement.

References

Carbone, A. 1997. Interpolants, cut elimination and flow graphs for the propositional calculus. *Annals of Pure and Applied Logic* **83**, 249–299.

Čubrić, Djordje. 1994. Interpolation property for bicartesian closed categories. *Archive for Mathematical Logic* **33**, 291–319.

Kanazawa, Makoto. 20001. Learning word-to-meaning mappings in logical semantics. In Robert van Rooy and Martin Stokhof, editors, *Proceedings of the Thirteenth Amsterdam Colloquium*, pages 126–131. ILLC/Department of Philosophy, University of Amsterdam.

Pentus, Mati. 1997. Product-free Lambek calculus and context-free grammars. *Journal of Symbolic Logic* **62**, 648–660.

Reinhard Muskens. 2001. Lambda grammars and the syntax-semantics interface. In Robert van Rooy and Martin Stokhof, editors, *Proceedings of the Thirteenth Amsterdam Colloquium*, pages 150–155. ILLC/Department of Philosophy, University of Amsterdam.

Reinhard Muskens. To appear. Lambdas, language, and logic. In Geert-Jan Kruijff and Richard Oehrle, editors, *Resource Sensitivity in Binding and Anaphora*, Studies in Linguistics and Philosophy. Dordrecht: Kluwer.

Prawitz, Dag. 1965. *Natural Deduction: A Proof-Theoretical Study*. Stockholm: Almqvist & Wiksell.

Siskind, J. Mark. 1996. A computational study of cross-situational techniques for learning word-to-meaning mappings. *Cognition* **61**(1–2), 39–91. (Reprinted in Michael Brent, ed., 1996, *Computational Approaches to Language Acquisition*, pages 39–91, Amsterdam: Elsevier.)

Siskind, J. Mark. 2000. Learning word-to-meaning mappings. In Peter Broeder and Jaap Murre, eds., *Models of Language Acquisition: Inductive and Deductive Approaches*, pages 121–153. Oxford: Oxford University Press.

Troelstra, A. S. and H. Schwichtenberg. 2000. *Basic Proof Theory*. Second Edition. Cambridge: Cambridge University Press.

Wroński, Andrej. 1984. Interpolation and amalgamation properties of *BCK*-algebras. *Mathematica Japonica* **29**, 115–121.