Underspecification of quantifier scope

Markus Egg and Michael Kohlhase
Universität des Saarlandes, Computerlinguistik/Informatik

1 Introduction

In this paper, we present an approach to a subject matter that has been discussed extensively in the literature, viz., the question of how to represent semantic structures with scope ambiguities in an underspecified way. In the following, we will focus our attention on examples like (1), which have two readings, as either quantifier may have scope over the other one:

(1) Every researcher visited a company

In addition, combining the semantics of quantifiers with the semantics of the expression that subcategorizes for them is a non-trivial problem in itself: If we assume e.g. the standard quantifier type $(e \to t) \to t$ (e stands for the individual type and t, for the truth value type) and the intuitive type $e \to e \to t$ for transitive verbs (disregarding matters of eventuality arguments), there is no direct way of combining the semantics of two NPs (or DPs) with the semantics of their subcategorizing verb by functional application.[1]

Many solutions have been proposed for an appropriate treatment of examples like (1). More traditional ones, e.g., UDRT (Reyle 1992), Cooper (1983), Pereira (1990) or Logical Form in Generative Grammar, invoke additional machinery to control the relation between the underspecified representation of sentences like (1) and its possible disambiguations. USDL (Pinkal 1996) can do without such additions to the representation formalism. However, USDL must take recourse to a metalanguage level to construct underspecified representations of structurally ambiguous sentences. In addition, the strict control over variable binding processes, which is guaranteed by other formalisms (e.g. type theory), is lost in the USDL metalanguage.[2]

We tackle these problems by reformulating the USDL analysis of examples like (1) in λ-DRT (Kohlhase, Kuschert, and Pinkal 1996), a higher-order extension of DRT. Instead of λ-DRT itself, we use the dynamic λ-calculus $\mathcal{DLC}$ (Kohlhase and Kuschert 1997), which provides a uniform algebraic basis for λ-DRT and other compositional discourse logics (Zeevat 1989; Groenendijk and Stokhof 1990; Eijck and Kamp 1995; Muskens 1994). In particular, $\mathcal{DLC}$ provides the type inference mechanism that is crucial for variable control in our derivations.

A dynamic base formalism, we contend, allows us to express underspecified representations of scopally ambiguous sentences directly without recourse to a metalanguage and to use $\mathcal{DLC}$'s strict control over variable binding. As a useful side effect, this approach makes available the results of USDL analysis of scopal quantifier ambiguities to the ongoing research on ellipsis reconstruction (Gardent and Kohlhase 1996).

Traditional formalisms that treat scope ambiguities exhibit a twopartite semantic construction: First, scopally underspecified sentences are given a 'skeleton' representation that comprises the semantics of the sentence (with variables substituting for the scopally ambiguous quantifiers), second, the semantic representations of these quantifiers are 'stored'; all that is said is that these quantifiers take scope over the skeleton sentence.

Consider e.g. the Generative Grammar account of (1): Its surface structure is not interpreted directly. Rather, the input for interpretation are expressions of Logical Form, where quantifiers have been moved out of their base positions within the skeleton sentence. They have been adjoined to IP (inflection phrase, roughly, the sentence without complementizers). Moved quantifiers leave a coindexed trace. Example (1) has two possible LF representations:

(2) $[_{IP}$ [every researcher]$_i$ $[_{IP}$ [a company]$_j$ $[_{IP}$ t$_i$ visits t$_j$]]]

(3) $[_{IP}$ [a company]$_j$ $[_{IP}$ [every researcher]$_i$ $[_{IP}$ t$_i$ visits t$_j$]]]

Traces are interpreted as variables of type e that carry the index of the trace. If we assume that combining the semantics of such a raised quantifier with the semantics of the IP it is adjoined to involves λ-abstraction (on the IP semantics) over a variable of type e with the index of the quantifier we get the two desired semantic representations of (1). E.g., (2) would be interpreted as (4):

[1] Lifting the type of the transitive verb can solve this problem, but at the cost of losing the intuitivity of the $e \to e \to t$ type.

[2] However, it is possible to reconstruct this kind of control in the syntax-semantics interface (Pinkal 1996, 16).

(4) $\lambda P \forall x(\mathbf{researcher'}(x) \longrightarrow P(x))(\lambda x_i(\lambda Q \exists y(\mathbf{company'}(y) \wedge Q(y))(\lambda x_j(\mathbf{visit'}(x_i, x_j))))) =$
$\forall x(\mathbf{researcher'}(x) \longrightarrow \exists y(\mathbf{company'}(y) \wedge \mathbf{visit'}(x, y)))$

USDL, on the other hand, is a semantic metalanguage. Its expressions denote object language expressions or operations. E.g., the constant **company'** denotes an expression in a suitable object language, say, 'λx **company'**(x)' in a language of predicate logic. '@' stands for functional application, and 'L_x' models object-level λ-abstraction over a variable x. In contrast to λx the metalevel L_x may capture free occurrences of x. For scopally ambiguous sentences S, USDL yields a set of constraints on the meaning of S, which describes the set of unambiguous readings of S. These constraints involve context variables C_n which are mere place-holders. They are instantiated by suitable material during the calculation of the unambiguous readings from such a constraint set. For (1), USDL deducts the set (5) of constraints over its meaning X_0:

(5) $\{X_0 = C_5(\mathbf{visit'}@y@x),\ X_0 = C_1(\mathbf{every'}@\mathbf{researcher'}@L_x\ C_2(\mathbf{visit'}@y@x)),$
$X_0 = C_3(\mathbf{a'}@\mathbf{company'}@L_y\ C_4(\mathbf{visit'}@y@x))\}$

(5) means in prose that the sentence meaning X_0 is equal to some material (C_5) applied to the skeleton sentence meaning **visit'**$@y@x$. In addition, for each of the two quantifiers, X_0 can also be described as involving the quantifier meaning applied to a λ-abstraction of a constituent that contains the skeleton sentence meaning. This entails that each quantifier has scope over the verb that syntactically subcategorizes for it. The syntax-semantic interface alone ensures that the subscripts of the metalanguage lambda operators L_n match the variables x_n in the skeleton sentence meaning. There is no $\alpha\beta\eta$-equivalence in USDL.

Applying a linear second-order unification algorithm (e.g. Niehren, Ruhrberg, and Pinkal 1997) to the constraint set (5) yields the substitution

$$\{C_1 = C_4 = \lambda X.X, C_2 = \lambda X.\mathbf{a'}@\mathbf{company'}@L_y\ (X), C_3 = \lambda X.\mathbf{every'}@\mathbf{researcher'}@L_x\ (X)\}$$

Applying this to the original set (5) yields the sentence semantics

$$\{X_0 = \mathbf{every'}@\mathbf{researcher'}@L_x\ (\mathbf{a'}@\mathbf{company'}@L_y\ (\mathbf{visit'}@y@x))\}$$

which corresponds to (4) via the USDL translation to predicate logic.

2 The dynamic approach

A simple attempt at rewriting the USDL analysis to act directly on standard predicate logic must fail, because it is not possible to substitute solutions for contexts in the same way: The free variables of the skeleton sentence must not be captured in the process of variable instantiation. E.g., short of renaming the free x_j in the equation (6), the solution $C_2 = \lambda p \exists x_j(\mathbf{company'}(x_j) \wedge p)$ (with p of type t) could not be inserted into (6). In the USDL metalevel, this 'binding' was unproblematic due to the lack of binding constraints imposed by L_x in (5). The USDL analysis of examples like (1) relies crucially on this feature of the metalanguage.

(6) $X_0 = C_1(\forall x_i(\mathbf{researcher'}(x_i) \longrightarrow C_2(\mathbf{visit'}(x_i, x_j))))$

What we need, is a dynamic formalism that allows the capturing of free variables. Such a formalism could overcome the problem of how to replace C_2 in (6), indeed, it would allow us to take over the USDL analyses of scope underspecification directly.

This approach must employ a concept of dynamic binding that differs from the one employed e.g. in Dynamic Predicate Logic (DPL, Groenendijk and Stokhof 1991) in two respects:

- it assigns a certain amount of dynamicity to universal quantifiers
- its notion of dynamicity is fine-grained, to distinguish dynamicity of existential and universal quantifiers

The first request follows from the fact that resolutions of constraints (predicate logic reformulations of USDL constraints like in (5)) may involve substitutions of universal quantifiers for context variables C_n, too. E.g., deriving the reading of (1) in which the universal quantifier has narrow scope would involve substituting $\lambda p \forall x_i(\mathbf{researcher'}(x_i) \longrightarrow p)$ for C_3 in (7) ((7) is one of the constraints on the meaning of (1)), which would have to bind the free x_i in **visit'**(x_i, x_j):

(7) $X_0 = C_3(\exists x_j(\mathbf{company'}(x_j) \wedge C_4(\mathbf{visit'}(x_i, x_j))))$

The second request follows from the observation that universal quantifiers differ from existential ones with respect to their binding behaviour. Pace some marginal examples (which might be attributed to properties of very restricted discourse domains, see Groenendijk and Stokhof 1990), universal quantifiers typically cannot exert binding force beyond their scope, otherwise examples like (8) would have to be grammatical:

(8) * Every researcher was thirsty. He entered the 'Fox and Hounds'.

Hence, if one wants to assign universal quantifiers a certain amount of dynamic potential, it must be restricted to the scope of the quantifier.

The approach we use to recast the USDL account of scope ambiguity is λ-DRT (Kohlhase, Kuschert, and Pinkal 1996), a higher-order extension of DRT (Kamp 1981). As in USDL, we render the meaning of a sentence by a constraint set over this meaning, which can describe more than one unambiguous reading, e.g., (5). Deriving unambiguous readings from such constraint sets is easy in $\mathcal{DLC}$; in particular, the problematic case of quantifier insertion (as in (6) or (7)) poses no problem for $\mathcal{DLC}$.

Underspecification enters by giving the meaning of constituents (especially, of sentences) in terms of equations in λ-DRT. This formalism combines the Montagovian type theoretic framework (Montague 1974) with dynamic approaches, such as DRT: it uses β-reduction to compute a first-order DRT representation of sentences from higher-order lexical items. Take for instance the sentence `A`^{i} `researcher sleeps.` (i denoting an index for anaphoric binding).

$$(\lambda Q.\ \boxed{\begin{array}{c} U_i \\ res(U_i) \end{array}} \otimes Q(U_i)\ )\ (\lambda X.\ \boxed{sleep(X)}\ ) \longrightarrow^*_\beta \boxed{\begin{array}{c} U_i \\ res(U_i) \end{array}} \otimes \boxed{sleep(U_i)} \longrightarrow_\delta \boxed{\begin{array}{c} U_i \\ res(U_i) \\ sleep(U_i) \end{array}}$$

'$\otimes$' is the λ-DRT conjunction operator that intuitively merges two DRSes by uniting the sets of discourse referents and that of conditions; it is formalized by δ-reduction in λ-DRT. We employ a linearised variant of this representation, where discourse referents are introduced by a dynamic binding operator δ:

$$(\lambda Q.\delta U_i.\mathsf{res}(U_i) \otimes Q(U_i))(\lambda X.\mathsf{sleep}(X)) \longrightarrow^*_\beta \delta U_i.\mathsf{res}(U_i) \otimes \mathsf{sleep}(U_i) \longrightarrow_\delta \delta U_i.\mathsf{res}(U_i) \wedge \mathsf{sleep}(U_i)$$

Consider now continuing the sentence above by `He`$_i$ `snores`, which is represented by $\mathsf{snore}(U_i)$. From the representation of the two sentences, the representation of the discourse may then be constructed by applying them to $\lambda P.\lambda Q.P \otimes Q$ and reducing, arriving at $\delta U_i.\mathsf{res}(U_i) \wedge \mathsf{sleep}(U_i) \wedge \mathsf{snore}(U_i)$. In this process the free variable U_i in $\mathsf{snore}(U_i)$, representing the pronoun `he`$_i$, has in the course of β-reduction been captured by the δ-abstraction δU_i in the representation of the first sentence, which is part of the representation of `a`^{i} `res`. Indeed, the capturing of free variables, *the* thing impossible in pure λ-calculus, is the driving force of the establishment of anaphoric binding in λ-DRT. We will use this very possibility of capturing variables in a controlled way (this control over variables is the main improvement over USDL), to formalize quantifier underspecification.

Next, we will sketch $\mathcal{DLC}$ (the concrete implementation of λ-DRT in this paper), and then give an example of using higher-order unification in $\mathcal{DLC}$ to represent quantifier underspecification.

3 Dynamic λ-Calculus

The central idea behind $\mathcal{DLC}$ is to extend the type system to incorporate information about variables. In particular, types have to represent information about the free variables of a formula (those that can be captured) and about those that have dynamic binding power (those that can capture free variables of other formulae). This distinction is expressed by annotating the variables with '$-$' for the former, and '$+$', for the latter in a so-called *mode* that is part of the type. If, for instance, $\Gamma = X^-, U^+$ is a mode and α is the base type t (for truth values), then the type $\Gamma\#t$ describes the set of propositions that contain a free variable X and dynamically bind a variable (introducing a discourse referent) U. Naturally, since $\mathcal{DLC}$ is a λ-calculus, the set of types is also closed under function types (i.e. $\alpha \to \beta$ is a type whenever α and β are).

$\mathcal{DLC}$-formulae are λ-calculus formulae together with dynamic abstractions of the form $\delta U.\mathbf{A}$. $\mathcal{DLC}$-well-typedness is verified (from type assumptions in a signature Σ and a variable context $\mathcal{A}$) using a variant of the classical λ-calculus inference system: $\mathbf{A}$ is a well-formed $\mathcal{DLC}$ formula of type α, iff the judgement $\mathcal{A} \vdash \mathbf{A}\colon\alpha$ is provable in the λ-calculus inference system with an added rule for δ-abstractions and where the λ-abstraction rule has been modified to manipulate the mode information in the types as indicated below.

$$\frac{\mathcal{A}, [U\colon U^-\#\overline{\overline{\beta}}] \vdash \mathbf{A}\colon\alpha}{\mathcal{A}, [U\colon U^-\#\overline{\overline{\beta}}] \vdash \delta U_\beta.\mathbf{A}\colon U^+\#\alpha} \qquad \frac{\mathcal{A}, [X\colon \Gamma^-\#\overline{\overline{\beta}}] \vdash \mathbf{A}\colon\alpha \quad X \notin \mathbf{Dom}(\Gamma)}{\mathcal{A} \vdash (\lambda X_\beta.\mathbf{A})\colon(\Gamma^-\#\overline{\overline{\beta}}) \to \alpha}$$

where $\overline{\overline{\alpha}} = \alpha$, and $\overline{\overline{\alpha \to \beta}} = \Gamma\#\alpha \to \Delta\#\beta$ for new modes Γ and Δ. These rules employ the convention that at the merging of modes, positively signed variables overwrite negatively signed of the same type and name. Note that this overwriting is important for the last rule for *dynamic abstraction*, where U^- may have been present in Γ, indicating the presence of a free variable U in $\mathbf{A}$. It is this overwriting effect that models the variable capturing at function application on the syntactic side.

The types of λ-abstractions must explicitly transport mode information from arguments of applications. Thus, the rule discharges the type assumption about the variable X from the variable context, since it is no longer free in the abstraction. Read backwards, this move augments the variable context with type information for X. The set of possible types of X_β in $\lambda X_\beta.\mathbf{A}$ is $\Gamma^-\#\overline{\overline{\beta}}$. The Γ^- part stands for the (as yet unknown) set of free variables of the argument. If the argument is itself functional, $\overline{\overline{\beta}} \neq \beta$, which reflects the fact that we do not know yet how the argument affects the free variables of its own argument within $\mathbf{A}$. (For details, see Kohlhase and Kuschert 1997.) The rule also enforces a kind of occurs-check for X in Γ^-. Finally, we assume the type identity $\Gamma\#(\Delta\#\alpha) = \Delta, \Gamma\#\alpha$ for all Γ, Δ, α.

Given this, we attain λ-DRT by fixing the set of base types to $\mathcal{BT} = \{e, t\}$ (individuals and truth values) and defining a set of connectives as conjunction, disjunction, negation and implication. In standard DRT, the binding properties of discourse referents in the context must be defined in an accessibility relation. With the extended type system, we can now express this relation directly in the syntax. E.g., we define the λ-DRT's negation operator (called dynamic negation, $\underline{\neg}$) to be of type $\underline{\neg}: (\Gamma\#t) \to \Gamma^-\#t$ where Γ^- is the set of negatively signed variables in the mode Γ. Thus, only those variables which are not bound dynamically in the argument of the negation – i.e., the free variables – are available outside the negated expression while the scope of the dynamic binders is closed off. In the same way we define a collection of dynamic implication operators of type $\underline{\Rightarrow}_{\Gamma\Delta}: (\Gamma\#t) \to (\Delta\#t) \to ((\Gamma^-, (\Delta^- \setminus \Gamma^+))\#t)$. Note that (Δ^-/Γ^+) captures the accessibility relation between the two arguments: only those free variables of the second argument are available to the outside which are not bound by the discourse referents of the first argument.

To get the feeling for the system, consider a variation of (1), `Every researcher sleeps.` Its representation is constructed by applying the representation of $\mathsf{sleeps}(U)$ to $\lambda Q.(\delta U.\mathsf{res}(U)\underline{\Rightarrow}Q)$, the representation of `every researcher`. We first derive that the quantifier *every researcher* represented by $\lambda Q_t.(\delta U_e.\mathsf{res}U) \Rightarrow Q$.

$$\dfrac{\dfrac{}{\mathcal{Q},\mathcal{U} \vdash \underline{\Rightarrow}: U^+\#t \to \Gamma^-\#t \to \Gamma^-_{-U}\#t} \quad \dfrac{\dfrac{\dfrac{}{\mathcal{Q},\mathcal{U}\vdash U: U^-\#e} \quad \dfrac{}{\mathcal{Q},\mathcal{U}\vdash \mathsf{res}: U^-\#e\to U^-\#t}}{\mathcal{Q},\mathcal{U}\vdash \mathsf{res}U: U^-\#t}}{\mathcal{Q},\mathcal{U}\vdash \delta U_e.\mathsf{res}U: U^+\#t} \quad \dfrac{}{\mathcal{Q},\mathcal{U}\vdash Q:\Gamma^-\#t}}{\dfrac{\mathcal{Q},\mathcal{U}\vdash (\delta U.\mathsf{res}(U))\underline{\Rightarrow}Q: \Gamma^-_{-U}\#t}{\mathcal{U}\vdash \lambda Q_t.(\delta U.\mathsf{res}(U))\underline{\Rightarrow}Q: \Gamma^-\#t\to\Gamma^-_{-U}\#t}}$$

where $\mathcal{Q} = [Q: \Gamma^-\#t]$ and $\mathcal{U} = [U : U^-\#e]$ and Δ_{-U} stands for Δ without U. Then for `sleeps` we have $\mathcal{U} \vdash \mathsf{sleeps}U: U^-\#t$. Thus the result type of `every researcher sleeps` is $\emptyset\#t$. This is as one would expect since $(\delta U.\mathsf{res}(U))\underline{\Rightarrow}\mathsf{sleep}(U))$ has no dynamic binding power to the outside of $\delta U.\mathsf{res}(U)\underline{\Rightarrow}\mathsf{sleep}(U)$. Unlike in Nerbonne (1992), the record of binders and free variables characterizes the representations themselves, not their form.

4 Example

We will now apply our analysis to sentence (1). Just like USDL, it generates the constraints

$$X_0 = C_1(\mathsf{er}(C_3(\underline{\mathsf{vis}UV}))) = C_2(\mathsf{ac}(C_4(\mathsf{vis}UV)))$$

where the new variables C_i have types $\overline{\overline{t \to t}} = (\Gamma_i\#t) \to (\Xi_i\#t)$. ('er' and 'ac' are abbreviations for the semantics of *every researcher* and *a company*, respectively.) We assume the following lexicon:[3]

word	formula	type
visited	vis	$\Gamma^-\#e \to \Delta^-\#e \to \Gamma^- \cup \Delta^-\#t$
sleeps	sleeps	$\Gamma^-\#e \to \Gamma^-\#t$
researcher	res	$\Gamma^-\#e \to \Gamma^-\#t$
company	comp	$\Gamma^-\#e \to \Gamma^-\#t$
every	$\lambda R_{e\to t}.\lambda Q_t.(\delta U_e.RU)\underline{\Rightarrow}Q$	$(\Gamma\#e \to t) \to (\Delta, U^-\#t) \to (\Gamma^- \cup (\Delta^- \setminus \Gamma^+))\#t)$
a	$\lambda R_{e\to t}.\lambda Q_t.\delta V_e.RV \otimes Q$	$(\Gamma\#e \to t) \to (\Delta, V^-\#t) \to (\Gamma \cup \Delta, V^+)\#t)$

[3] The types specify verbs and nouns as n-ary relations that coordinate the free variables of their arguments accordingly. (Since we omit modification here, we assume simplified types of nouns and verbs. We enforce non-degenerate binding for quantifiers that dynamically bind some variable U, by adding the mode U^- to the type specification of the mode Δ in the above representations of quantifiers. This expresses that they require a free variable U in their second argument.)

The added expressivity of $\mathcal{DLC}$ allows us to express the behaviour of binders and variables in the types of $\mathcal{DLC}$ expressions. As functional application and instantiation of mode variables must be well-typed, the linear higher-order unification and the simplification of type constraints induced by type inference mutually constrain each other, giving us a powerful mechanism for propagating information about variables and quantifiers binding them.

In addition to the constraints taken over from USDL, we can use $\mathcal{DLC}$ types to encode the general observation that, at text level, there may be no free variables except those introduced by deictic pronouns. (Single sentences qualify as a text, too, if they occur in isolation.) This entails that if the type of the sentence (1) is $\Theta\#t$ then $\Theta^- = \emptyset$, and, in particular, the variables U and V are no longer free at the sentence level.

Let us now calculate the type constaints Γ_i and Ξ_i imposed by the equational constraints: $\mathsf{vis}UV$ is of type $U^-, V^-\#t$, hence, $\Gamma_3 = \Gamma_4 = U^-, V^-$. As we have seen at the end of section 3, the domain type of er is Δ, U^- and by similar reasoning that of ac is Δ', V^-, so $\Xi_3 = \Delta, U^-$ and $\Xi_4 = \Delta', V^-$. Analogously we obtain values for the type of C_1 and C_2, which entails the following type judgements.

$$\boxed{\begin{array}{ll} C_1\colon(\Delta^-\#t) \to (\Theta\#t) & C_2\colon(\Delta', V^+\#t) \to (\Theta\#t) \\ C_3\colon(U^-, V^-\#t) \to (\Delta, U^-\#t) & C_4\colon(U^-, V^-\#t) \to (\Delta', V^-\#t) \end{array}}$$

Now we start linear unification, calculating and simplifying the type constraints after each step. First, we project the variable C_1, i.e., we guess that $C_1 = \lambda Z_t.Z$. This yields the wide scope reading of *every researcher* for (1). Linear imitation for C_1 is also possible and yields the narrow scope reading, which we will not pursue here. Since $\lambda Z_t.Z$ has type $(\Psi\#t) \to (\Psi\#t)$, we have

$$\boxed{\Delta^- = \Theta \qquad C_1\colon(\Theta\#t) \to (\Theta\#t)}$$

Thus even at this early stage of the computation, we can see that

- Θ is empty, because by $\Delta^- = \Theta$ it contains no positive variables and by the status of (1) as a text it contains no negative variables. In other words, X_0 has neither free variables nor binding potential. This was to be expected, as the reading of (1) we are looking at is a text in which the universal quantifier *every researcher* has scope over everything else.
- the value of C_3 must involve some kind of binder for V: the target type of C_3 does not mention V^-, while the argument type does.

In the next unification step, we must imitate C_2, since the projection would lead to immediate failure, i.e. we bind C_2 to $\lambda Z_t.\mathsf{er}(C_5 Z)$, where C_5 is a new variable of characteristic type (i.e., the type without mode) $t \to t$. Type inference reveals that the type of C_5 must be

$$\boxed{C_5\colon(\Delta', V^+\#t) \to (\Delta, U^-\#t)}$$

After decomposition the problem has the form $C_3(\mathsf{vis}UV) = C_5(\mathsf{ac}(C_4(\mathsf{vis}UV)))$. We continue the unification by projecting $(C_5 = \lambda Z_t.Z)$ and get the type constraint $\Delta', V^+\#t = (\Delta, U^-\#t)$ (argument and target types of $\lambda Z.Z$ must be identical) or equivalently

$$\boxed{\Delta' = \Phi, U^- \qquad \Delta = \Phi, V^+}$$

for a new mode Φ. Now we imitate C_3 by binding it to $\lambda Z_t.\mathsf{ac}(C_6 Z)$. C_6 is a new variable of type

$$\boxed{C_6\colon\Delta', U^- V^-\#t \to (\Phi, U^-, V^-\#t)}$$

After decomposition the problem has the form $C_6(\mathsf{vis}UV) = C_4(\mathsf{vis}UV)$. If we guess that $C_4 = C_6 = \lambda Z_t.Z$, then we are left with a trivial unification problem and we have completed the linear unification. Type inference gives us the constraints

$$\boxed{\Phi = \emptyset \qquad \Delta = V^+ \qquad \Delta' = U^-}$$

Collecting all constraints makes all the dynamic potentials explicit:

variable	value	type
C_1	$\lambda Z_t.Z$	$t \to t$
C_2	er	$(U^-, V^+\#t) \to t$
C_3	ac	$(U^-, V^-\#t) \to (U^-, V^+\#t)$
C_4	$\lambda Z_t.Z$	$(U^-, V^-\#t) \to (U^-, V^-\#t)$
X_0	(er(ac(vis U V)))	t

5 Conclusion and further work

To sum up, in this paper we have presented an approach to the underspecified representation of scopally ambiguous sentences that accounts for them directly in terms of a dynamic lambda calculus, without having to rely on additional machinery. Deriving fully specified readings from underspecified representations involves only simple substitutions of typed context variables. The binding potential of expressions is highlighted throughout, in the derivation and in the resolution of the constraints on sentence meanings.

So far, we used the strict record on binders and variables merely to make the treatment of binders and variables explicit. But we expect the proposed approach to prove useful in restricting potential ambiguities within texts. Consider e.g. the standard example of forcing wide scope of the existential quantifier in an example like (9) by picking up the referent of the quantifier anaphorically in the next sentence. This can easily be described in our formalism:

(9) Every researcher visited a company. It belonged to a tycoon.

Omitting deictic use of pronouns, we request (complete) texts to contain no free variables, that is to say, all free variables introduced in the text must be bound off within it as well. Hence, if the type of the second sentence in (9) is $\Delta^+, U^-\#t$ (the free variable U is introduced by the pronoun *it*), there has to be a preceding sentence in the text whose mode comprises a binder for U, i.e., is of type $\Gamma, U^+\#t$, for any Γ. In (9), this restricts the ambiguity of the first sentence in the desired way: It must provide a binder for the variable as introduced by the pronoun, which excludes the reading in which the indefinite quantifier has narrow scope w.r.t. the universal quantifier. Scoping a universal quantifier over an existential would have 'shadowed off' the binding potential of the existential quantifier.

The proposed approach will also be of use in the domain of nested quantifiers. As noted by e.g. Nerbonne (1992), deriving fully specified readings from underspecified representations for them needs strict control over variables and their binders to rule out overgeneration. As we have shown, our approach provides this. Further work will be dedicated to exploring this potential of the proposed approach.

References

Cooper, R. (1983). *Quantification and syntactic theory.* Dordrecht: Reidel.

Eijck, J. v. and H. Kamp (1995). Representing discourse in context. In J. van Benthem and A. ter Meulen (eds), *Handbook of Logic and Language.* Elsevier Science B.V.

Gardent, C. and M. Kohlhase (1996). Higher–Order Coloured Unification and Natural Language Semantics. In *Proc. ACL'96.* Santa Cruz.

Groenendijk, J. and M. Stokhof (1990). Dynamic Montague Grammar. In L. Kálmán and L. Pólos (eds), *Papers from the Second Symposium on Logic and Language*, pp. 3 – 48. Budapest.

Groenendijk, J. and M. Stokhof (1991). Dynamic predicate logic. *Linguistics & Philosophy 14*, 39–100.

Kamp, H. (1981). A theory of truth and semantic representation. In J. Groenendijk, T. Janssen, and M. Stokhof (eds), *Formal Methods in the Study of Language*, pp. 277 – 322. Amsterdam.

Kohlhase, M. and S. Kuschert (1997). *Dynamic λ-Calculus.* CLAUS-Report 92, Uni. d. Saarlandes.

Kohlhase, M., S. Kuschert, and M. Pinkal (1996). A type-theoretic semantics for λ-DRT. In P. Dekker and M. Stokhof (eds), *Proc. 10th Amsterdam Colloquium*, pp. 479–498. ILLC, Amsterdam.

Montague, R. (1974). The proper treatment of quantification in ordinary English. In R. Montague (ed.), *Formal Philosophy. Selected Papers.* New Haven: Yale University Press.

Muskens, R. (1994). A compositional discourse representation theory. In P. Dekker and M. Stokhof (eds), *Proceedings of the 9th Amsterdam Colloquium*, pp. 467 – 486. ILLC, Amsterdam.

Nerbonne, J. (1992). A feature-based syntax/semantics interface. DFKI Report RR-92-42. Saarbrücken.

Niehren, J., P. Ruhrberg, and M. Pinkal (1997). On equality up-to constraints over finite trees, context unification and one-step rewriting. In *Proceedings of CADE'97.*

Pereira, F. (1990). Categorial semantics and scoping. *Computational Linguistics 16*, 1–10.

Pinkal, M. (1996). Radical underspecification. In P. Dekker and M. Stokhof (eds), *Proceedings of the 10th Amsterdam Colloquium*, pp. 587–606. ILLC, Amsterdam.

Reyle, U. (1992). Dealing with ambiguities by underspecification: a first order calculus for unscoped representations. In *Proceedings of the Eighth Amsterdam Colloquium.*

Zeevat, H. (1989). A compositional approach to DRT. *Linguistics & Philosophy 12*, 95–131.