Dynamic Semantics
Barker/ESSLLI 2010
Tuesday: Anaphora

Paradigm dynamic semantics:
Gronendijk and Stokhof 1991 Dynamic Predicate Logic

1. Starting point: more thinking about what it means to be dynamic versus static
2. Possible claims and operational tests about dynamicicity
3. Presentation of G&S's system, with examples
4. Brief comparison with Kamp's Discourse Representation Theory.
5. Comparison with Dekker's PLA.

Jim Pryor (via email): What I'm hankering after as a definition of the static/dynamic contrast is a definition that's agnostic as to what particular sort of abstract object a context or proposition is thought to be.

Candidate definition: a system is dynamic if it is possible to find three expressions A, B and C and and a context c such that c + A and c + B have identital truth conditions, but (c + A) + C has different truth conditions than (c + B) + C. If so, then something about adding A to the context influences the evaluation of C in a way that is different than the way that adding B to the context influences the evaluation of C, and whatever is different about A and B is independent of their truth conditions.

Candidate definition: a system is dynamic (in the intended sense) iff the truth conditions of a sentence (normally) entail facts about the state of the discourse.

==
Preface

Central question: is it part of the meaning of an expression to explicitly manipulate discourse representation (dynamic)? Or is meaning entirely about the world under discussion (static)?

Basic issue: everyone agrees that the value of an expression (denotation, content) depends in part on the context (prime example: indexicals). Does the evaluation of an expression affect subsequent context directly (dynamic) or only indirectly, through the mediation of pragmatic (perhaps Gricean) reasoning (static)?

Possible defitions and tests for dynamicity.

1. Sentences denote context update functions, truth conditions secondary (Heim)
2. Sentences have side-effects that influence subsequent evaluation (G&S)

3. Operational test: A&B $\neq$ B&A (Chierchia 1995)
4. Expressions are dynamic or not only relative to other stages in the interpretation of a sentence (Shan)

Groenendijk and Stokhof cite David Harel as someone who developed a dynamic logic for programming languages. The citation should actually be (G&S omit the second and third authors in their bibliography):

David Harel, Dexter Kozen, and Jerzy Tiuryn. 1984. Dynamic Logic.
Handbook of Philosophical Logic Volume II --- Extensions of Classical Logic.

The name *Dynamic Logic* emphasizes the principal feature distinguishing it from classical predicate logic. In the latter, truth is *static*: the truth value of a formula φ is determined by a valuation of its free variables over some structure. The valuation and the truth value of φ it induces are regarded as immutable; there is no formalism relating them to any other valuations or truth values. In Dynamic Logic, there are explicit syntactic constructs called *programs* whose main role is to change the values of variables, thereby changing the truth values of formulas. For example, the program $x := x + 1$ over the natural numbers changes the truth value of the formula "x is even".

==

Review

1. In a dynamic system, the value produced by evaluating a sentence is an updated context.

Static: a sentence denotes a function from contexts to propositions

Peregrin, relying on Kaplan (e.g., "Demonstratives"): the contribution of an utterance depends on the context of utterance.

$[\![\text{I am hungry}]\!] = \lambda c \lambda w.\mathrm{speaker}(c)\,\mathrm{is\,hungry\,in}\,w = \lambda c \lambda w.\mathrm{CB}\,\mathrm{is\,hungry\,in}\,w$

For instance, the interpretation of indexicals such as *I* depends on the context of utterance. The meaning of *I am hungry*, then, is a $\mathrm{character}$: a function from contexts to propositions. Propositions, let's say, are sets of worlds (actually, functions from worlds to truth values).

Static   c + s = p   context + sentence = proposition
context + character = content   (Kaplan)

On this view, a context and a sentence determine a proposition, the $\mathrm{content}$ of the sentence in that context.

Dynamic: a sentence denotes a function from contexts to updated contexts.

Assume further that contexts are sets of worlds.

$[\![\text{I am hungry}]\!] = \lambda c \lambda w.\mathrm{speaker}(c)\,\mathrm{is\,hungry\,in}\,w\,\&\,w \in c = \lambda c \lambda w.\mathrm{CB\,is\,hungry\,in}\,w\,\&\,w \in c$

The only new thing is restricting the value to worlds in the context.

Dynamic $c_1 + s = c_2$ context + sentence = updated context
sentences denote context update functions

Crucial idea: the meaning of the sentence directly controls the updated context.

Note: the updated context must also be updated by non-linguistic sources of information. For instance, something about the original context enables us to discern that the speaker of the utterance is me (CB). If the next utterance in the discourse is spoken by someone else, something must happen to the output of my utterance to reflect the change in speaker. So even in a dynamic system, the context change potential of a sentence can never be the only thing that adjusts contexts.

==
Groenendijk and Stokhof (1991): Dynamic Predicate Logic

Points about DPL:
* Compositionality issue wrt Kamp's DRT, others
* Computational connection: dynamic binding
* Mechanics: it's all about assignment functions
* Filters, tests, etc.
* Loss of information (choice of indefinite index)
* Dekker's PLA reformulation

(page numbers from the pre-print version)
p. 4: The general starting point of the kind of semantics DPL is an instance of, is that the meaning of a sentence does not lie in its truth conditions, but rather in the way it changes (the representation of) the information of the interpreter.

p. 5: [T]here is a strong correspondence between the dynamic view on meaning, and a bsic idea underlying the denotational approach to the semantics of programming languages, viz., that the meaning of a program can be captured in terms of a relationa between machine states... A machine state may be identified with an assignment of objects to variables. The interpretation of a program can then be regarded as a set of ordered pairs of assignments, as the set of all its possible 'input-output' pairs. A pair <g,h> is in the interpretation of a program π, if when π is executed in state g, a possible resulting state is h.

Notation: '$\wedge$' is dynamic conjunction (order matters); '&' is static conjunction.

Two variables, x and y. Two men, John and Bill. Four distinct assignment functions:

g1	g2	g3	g4
x-->j	x-->j	x-->b	x-->b
y-->j	y-->b	y-->j	y-->b

Standard view: *He*$_x$ *is John* is true relative to g1 and g2, but not wrt g3 or g4.

Notation: "h[x]g": assignment functions h and g differ at most in the value they assign to x.

(A) g1[x]g1 g1[x]g3 g2[x]g2 g2[x]g4 g3[x]g3 g3[x]g1 g4[x]g4 g4[x]g2

In general, g[x]g for all g (reflexive)
if h[x]g, g[x]h (symmetric)
if h[x]g and g[x]f, then h[x]f (transitive)

Thus each variable induces an equivalence relation on the set of assignments. The cells of the partition induced by x are {g1,g3} and {g2, g4}.

F is the interpretation function assigning individual names to the individual constants, and sets of n-tuples of individuals to the n-place predicates.
For instance, F(man) = {j,b}. If P is the predicate man, we have

$$
\begin{aligned}
&[\![\exists x\, Px]\!] = \{ <g,h> \mid h[x]g \,\&\, h(x) \in F(P) \} \\
&= [\![\exists x\, \mathrm{man}\, x]\!] = \{ <g,h> \mid h[x]g \,\&\, h(x) \in F(\mathrm{man}) \} \\
&= [\![\exists x\, \mathrm{man}\, x]\!] = \{ <g,h> \mid h[x]g \,\&\, h(x) \in \{j,b\} \}
\end{aligned}
$$

h has to be an assignment function that maps x onto j or b. All the assignment functions do that. The only requirement on g is that it is in the same equivalence class as h wrt x. So the value of this formula is the set of pairs of assignment functions given in A:

$$
\begin{aligned}
&[\![\exists x\, Px]\!] \\
&= \{ <g_1,g_1>, <g_1,g_3>, <g_2,g_2>, <g_2,g_4>, <g_3,g_3>, <g_3,g_1>, <g_4,g_4>, <g_4,g_2> \}
\end{aligned}
$$

You can think of it this way: no matter what the input assignment assigns x to, the output assignment makes sure that x is pointing to a man.

If only John is happy, then we only have pairs in which the second element maps x onto John:

$[\![\exists x\, \text{happy}\, x]\!]$
$= \{ < g_1, g_1 > , < g_2, g_2 > , < g_3, g_1 > , < g_4, g_2 > \}$

TESTS (PURE expressions) p. 7:
Unlike existentially quantified formulas, atomic formulas do not have dynamic effects of their own. Rather they function as a kind of 'test' on incoming assignments.

DPL semantics (p. 14):

Definition 2 (Semantics)

1. $[\![Rt_1 \dots t_n]\!] = \{\langle g, h\rangle \mid h = g \;\&\; \langle [\![t_1]\!]_h \dots [\![t_n]\!]_h\rangle \in F(R)\}$
2. $[\![t_1 = t_2]\!] = \{\langle g, h\rangle \mid h = g \;\&\; [\![t_1]\!]_h = [\![t_2]\!]_h\}$
3. $[\![\neg\phi]\!] = \{\langle g, h\rangle \mid h = g \;\&\; \neg\exists k\colon \langle h, k\rangle \in [\![\phi]\!]\}$
4. $[\![\phi \wedge \psi]\!] = \{\langle g, h\rangle \mid \exists k\colon \langle g, k\rangle \in [\![\phi]\!] \;\&\; \langle k, h\rangle \in [\![\psi]\!]\}$
5. $[\![\phi \vee \psi]\!] = \{\langle g, h\rangle \mid h = g \;\&\; \exists k\colon \langle h, k\rangle \in [\![\phi]\!] \vee \langle h, k\rangle \in [\![\psi]\!]\}$
6. $[\![\phi \to \psi]\!] = \{\langle g, h\rangle \mid h = g \;\&\; \forall k\colon \langle h, k\rangle \in [\![\phi]\!] \Rightarrow \exists j\colon \langle k, j\rangle \in [\![\psi]\!]\}$
7. $[\![\exists x\phi]\!] = \{\langle g, h\rangle \mid \exists k\colon k[x]g \;\&\; \langle k, h\rangle \in [\![\phi]\!]\}$
8. $[\![\forall x\phi]\!] = \{\langle g, h\rangle \mid h = g \;\&\; \forall k\colon k[x]h \Rightarrow \exists j\colon \langle k, j\rangle \in [\![\phi]\!]\}$

John is a man:

$$\begin{aligned}[\![\text{man}\, j]\!] &= \{ < g, h > | h = g \,\&\, [\![j]\!]_h \in F(\text{man})\} \\ &= \{ < g, h > | h = g \,\&\, j \in \{j, b\}\} \\ &= \{ < g1, g1 > , < g2, g2 > , < g3, g3 > , < g4, g4 > \}\end{aligned}$$

Since John is a man, and since the fact that John is a man does not depend on the assignment of variables to entities, all input assignment functions pass the test.

He$_x$ is a man:

$$\begin{aligned}[\![\text{man}\, x]\!] &= \{ < g, h > | h = g \,\&\, [\![x]\!]_h \in F(\text{man})\} \\ &= \{ < g, h > | h = g \,\&\, h(x) \in \{j, b\}\} \\ &= \{ < g1, g1 > , < g2, g2 > , < g3, g3 > , < g4, g4 > \}\end{aligned}$$

Only those assignment functions that map x onto a man survive.

He$_x$ is happy:

$$
\begin{aligned}
[\![\text{happy } x]\!] &= \{ <g,h> | h = g \,\&\, [\![x]\!]_h \in F(\text{happy})\} \\
&= \{ <g,h> | h = g \,\&\, h(x) \in \{j\}\} \\
&= \{ <g1,g1>, <g2,g2> \}
\end{aligned}
$$

Only those assignment functions that map x onto a happy object survive (in this case, only assignment functions that map x onto John).

A man$_x$ exists. He$_x$ is happy.

$\exists x \,\text{man}\, x \qquad \text{happy}\, x$

Interpret pronoun as a variable. (Excercise: what happens when we choose a different variable from the one bound by the existential?) Interpret sequence of sentences as (dynamic) conjunction:

$[\exists x \,\text{man}\, x] \wedge \text{happy}\, x$

Note that the existential quantifier only has scope over the first conjunct. Nevertheless, it will 'bind' the pronoun in the following sentence.

$$
\begin{aligned}
[\![[\exists x \,\text{man}\, x] \wedge \text{happy}\, x]\!] &= \{ <g,h> | \exists k{:} <g,k> \in [\![\exists x \,\text{man}\, x]\!] \,\&\, <k,h> \in [\![\text{happy } x]\!]\} \\
&= \{ <g,h> | \exists k{:} <g,k> \in \{11, 13, 22, 24, 33, 31, 44, 42\} \\
&\qquad\qquad \& <k,h> \in \{11, 22\}\} \\
&= \{11, 22, 31, 42\}
\end{aligned}
$$

This function takes any input and reassigns x to a happy man.

Excercise: prove that you get the same result starting with $\exists x [\text{man}\, x \wedge \text{happy}\, x]$.

He$_x$ is happy. A man$_x$ exists.

$$
\begin{aligned}
[\![\text{happy}\, x \wedge \exists x \,\text{man}\, x]\!] &= \{ <g,h> | \exists k{:} <g,k> \in [\![\text{happy } x]\!] \,\&\, <k,h> \in [\![\exists x \,\text{man}\, x]\!]\} \\
&= \{11, 13, 22, 24\}
\end{aligned}
$$

This function accepts any input that maps x onto a happy object, and reassigns x to point to a man (any man, happy or not).

Truth: a formula ϕ is true wrt an assignment function g iff $\exists h{:} <g,h> \in [\![\phi]\!]$.

So *A man exists and he is happy* is true with respect to any assignment, as long as there is some man who is happy. But *He is happy and a man exists* is true only with respect to an assignment that assigns x to a happy creature, as long as there is at least one man (who may or may not be happy).

Excercise: Prove that *He*$_x$ *is happy and a man*$_x$ *exists* does not entail that *He*$_x$ *is happy*.

Characteristic theorem of DPL:

$$\exists x \phi \wedge \psi \simeq \exists x [\phi \wedge \psi]$$

This fact "illustrates the dynamics of the existential quantifier: its binding power extends indefinitely to the right. This is what makes DPL a suitable instrument for the representation of antecedent-anaphor relations across sentence bundaries."

From Benjamine Pierce, 1995. Preprint of article: Foundational Calculi for Programming Languages. In the CRC Handbook of Computer Science and Engineering.

Syntax:

$$\begin{array}{rcll} P, Q, R & ::= & \mathbf{0} & \text{inert process} \\ & & x(y).\,P & \text{input prefix} \\ & & \bar{x}y.\,P & \text{output prefix} \\ & & P \mid Q & \text{parallel composition} \\ & & (\nu x)\,P & \text{restriction} \\ & & !P & \text{replication} \end{array}$$

Renaming of bound variables:

$$\begin{array}{rcll} x(y).\,P & = & x(z).([z/y]P) & \text{if } z \notin FV(P) \\ (\nu y)\,P & = & (\nu z)\,([z/y]P) & \text{if } z \notin FV(P) \end{array}$$

Structural Congruence:

$$\begin{array}{rclll} P \mid Q & \equiv & Q \mid P & & \text{commutativity of parallel composition} \\ (P \mid Q) \mid R & \equiv & P \mid (Q \mid R) & & \text{associativity of parallel composition} \\ ((\nu x)P) \mid Q & \equiv & (\nu x)\,(P \mid Q) & \text{if } x \notin FV(Q) & \text{“scope extrusion”} \\ !P & \equiv & P \mid !P & & \text{replication} \end{array}$$

Operational Semantics:

$$\begin{array}{rclll} \bar{x}y.P \mid x(z).Q & \longrightarrow & P \mid [y/z]Q & & \text{communication} \\ P \mid R & \longrightarrow & Q \mid R & \text{if } P \longrightarrow Q & \text{reduction under } \mid \\ (\nu x)P & \longrightarrow & (\nu x)Q & \text{if } P \longrightarrow Q & \text{reduction under } \nu \\ P & \longrightarrow & Q & \text{if } P \equiv P' \longrightarrow Q' \equiv Q & \text{structural congruence} \end{array}$$

Figure 2: Syntax and Operational Semantics of the Pi-Calculus

!. The first says that the scope of a ν binding may be enlarged to enable reduction, as in $((\nu z)\, \bar{x}z.\, P) \mid x(y).\, Q \equiv (\nu z)\, \bar{x}z.\, P \mid x(y).\, Q \longrightarrow (\nu z)\, (P \mid [z/y]Q)$. The second formalizes the

==

Comparison of DPL with DRT

p. 34: DPL is intended to be 'empirically equivalent' to DRT... The difference between the two approaches is primarily of a methodological nature and compositionality is the watershed between the two.

Conditions: parts of a Discourse Representation Structure

Definition 26 (*DRT*-**semantics**)

1. $[\![Rt_1 \dots t_n]\!]^{Cond} = \{g \mid \langle [\![t_1]\!]_g \dots [\![t_n]\!]_g \rangle \in F(R)\}$
2. $[\![t_1 = t_2]\!]^{Cond} = \{g \mid [\![t_1]\!]_g = [\![t_2]\!]_g\}$
3. $[\![\neg\phi]\!]^{Cond} = \{g \mid \neg\exists h{:}\, \langle g, h\rangle \in [\![\phi]\!]^{DRS}\}$
4. $[\![\phi \vee \psi]\!]^{Cond} = \{g \mid \exists h{:}\, \langle g, h\rangle \in [\![\phi]\!]^{DRS} \vee \langle g, h\rangle \in [\![\psi]\!]^{DRS}\}$
5. $[\![\phi \rightarrow \psi]\!]^{Cond} = \{g \mid \forall h{:}\, \langle g, h\rangle \in [\![\phi]\!]^{DRS} \Rightarrow \exists k{:}\, \langle h, k\rangle \in [\![\psi]\!]^{DRS}\}$
6. $[\![[x_1 \dots x_k][\phi_1 \dots \phi_n]]\!]^{DRS} =$
   $\{\langle g, h\rangle \mid h[x_1 \dots x_k]g \;\&\; h \in [\![\phi_1]\!]^{Cond} \;\&\; \dots \;\&\; h \in [\![\phi_n]\!]^{Cond}\}$

Note: no conjunction, no quantifiers! But definition (6) takes a bunch of conditions,

conjoins them, prepends a bunch of variables, and (in effect) existentially quantifies over the variables.

DRT is dynamic:

p. 36: The interpretation of a DRS, being the same kind of object as the interpretation of formulas in DPL, is of a dynamic nature.

p. 39: ...one of the trademarks of theories such as those of Kamp and Heim is the non-quantificational analysis of indefinite terms, whereas it is characteristic of DPL that it does allow us to treat such terms as existentially quantified expressions.

(1) A man walks in the park. He whistles.

(1a) $\exists x[\texttt{man}(x) \wedge \texttt{walk\_in\_the\_park}(x) \wedge \texttt{whistle}(x)]$ *PL/†DRT*

(1b) $\exists x[\texttt{man}(x) \wedge \texttt{walk\_in\_the\_park}(x)] \wedge \texttt{whistle}(x)$ *DPL*

(1c) $[x][\texttt{man}(x), \texttt{walk\_in\_the\_park}(x), \texttt{whistle}(x)]$ *DRT*

(3) Every farmer who owns a donkey beats it

(3a) $\forall x \forall y[[\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)] \rightarrow \texttt{beat}(x,y)]$ *PL*

(3b) $\forall x[[\texttt{farmer}(x) \wedge \exists y[\texttt{donkey}(y) \wedge \texttt{own}(x,y)]] \rightarrow \texttt{beat}(x,y)]$ *DPL*

(3c) $[\ ][[x,y][\texttt{farmer}(x), \texttt{donkey}(y), \texttt{own}(x,y)] \rightarrow [\ ][\texttt{beat}(x,y)]]$ *DRT*

(3d) $\exists x \exists y[\texttt{farmer}(x) \wedge \texttt{donkey}(y) \wedge \texttt{own}(x,y)] \rightarrow \texttt{beat}(x,y)$ *†DRT*

Consider the first example. If our diagnosis of the problem that such a sequence poses, viz., that the real problem is to provide a compositional translation of such sequences of sentences into a logical representation language, is correct, then the *DRT*-representation has as little to offer as the *PL*-translation. The two component sentences cannot be retrieved from (1c), neither can they be isolated from (1a) as subformulas. The *DPL*-representation differs precisely at this point.

Accusations of non-compositionality are a bit surprising given that the DPL paper does not provide any details of how to translate English sentences into DPL! However, G&S also wrote a paper around the same time (1987 or so) called Dynamic Montague Grammar (available from the seminar web page) that contains a fully compositional treatment of a fragment of English, so their claims of compositionality for their system are justified.

Their claims that DRT fails to be compositional, however, have never sat well with me.

Certainly DRT (as in Kamp 1981) provides a way of building DRSs that is fully compositional. The rules for interpreting those DRSs depend on recognizing which sub-DRSs are subordinate to each other. For instance, the DRS build from the antecedent of a conditional is stipulated to be accessible from the DRS built from the consequent. Variables (discourse referents) introduced within one DRS are accessible (visible) from all subordinate DRSs. Certainly the net result is as G&S portray, and certainly the representations as copied above do not show a clear mapping onto syntactic constituents. However, the difference between Kamp's system and G&S's reformulation of it are

humongous. It would be equally legitimate to say that DPL produces the same non-compositional representations (after all, the whole point is that the DPL representations are logically equivalent to the non-compositional representations). So I am far from willing to endorse G&S's claim that Kamp's system suffers from egregious non-compositionality.

==

Dekker 2004 (‘PLA’)

p. 1-2: ...in Stalnaker 1998 is it argued that...inter-sentential anaphoric relationships are not inconsistent with a classical conception of meaning, if only one pays due attention to the pragmatics of interpretation...The semantics is spelled out as a classical satisfaction relation, which is extended so as to account for two, arguably systematic, pragmatic principles.

p. 5: ...it is one thing to observe that meaning and pragmatics might, in principle, be able to account for a set of data; it is another one to actually provide that account.

p. 5: It thus appears that the temporal order of utterances is relevant to interpretation, and also that this appears to be quite a systematic fact.

PLA: p. 8 ff.:

* the length n(phi) of a formula phi corresponds to the number of existential that are not in the scope of a negation.

* a pronoun p_i is indexically analyzed, in that it refers back to the i-th indefinite noun phrase which it finds if it looks back in the discourse from the location where the pronoun occurs.

* Interpretation is defined relative to a variable assignment and a sequence of individuals.

$$[\![x]\!]_{g,\vec{e}} = g(x)$$

$$[\![p_i]\!]_{g,\vec{e}} = \vec{e}_i$$

(7) $\vec{e} \models_{M,g} Rt_1 \dots t_m$ iff $\langle [t_1]_{g,\vec{e}}, \dots, [t_m]_{g,\vec{e}} \rangle \in E(R)$
$\vec{e} \models_{M,g} \exists x\phi$ iff $\vec{e}{-}1 \models_{M,g[x/\vec{e}_1]} \phi$
$\vec{e} \models_{M,g} \neg\phi$ iff $\neg\exists\vec{c} \in D^{n(\phi)}$: $\vec{c}\vec{e} \models_{M,g} \phi$
$\vec{e} \models_{M,g} \phi \wedge \psi$ iff $\vec{e}{-}n(\psi) \models_{M,g} \phi$ and $\vec{e} \models_{M,g} \psi$
where $\vec{e}{-}m$ is the sequence $\vec{e}_{m+1}, \vec{e}_{m+2}, \dots$

(8) ϕ is true wrt M, g and $\vec{e}$ iff $\exists\vec{c} \in D^{n(\phi)}$: $\vec{c}\vec{e} \models_{M,g} \phi$

The negation of a formula ϕ tells us that ϕ is simply false. It states that there is no way to fill ϕ's open places with a sequence $\vec{c}$ of $n(\phi)$ individuals. A negation thus closes the 'existential holes' of the formula ϕ in its scope, so that, e.g., $\neg\exists xFx$, as usual, means that no x is F. As a consequence, existential quantifiers (corresponding to indefinites) in ϕ cannot serve as antecedents for subsequent pronouns.

If we evaluate a conjunction $\phi \wedge \psi$ relative to a sequence $\vec{e}$, we evaluate the first conjunct ϕ relative to $\vec{e}{-}n(\psi)$, which is $\vec{e}$ with the contribution of ψ stripped of. Intuitively, this says that ϕ is evaluated *before* ψ has contributed its discourse referents. Probably it is easier to read it in a constructive way. If $\vec{e}$ satisfies ϕ, and $\vec{c}\vec{e}$ satisfies ψ, where $\vec{c}$ fits the indefinites contributed by ψ, so that the length of $\vec{c}$ is $n(\phi)$, then $\vec{c}\vec{e}$ satisfies $\phi \wedge \psi$ as well. The reader may observe that this is a truly dynamic notion of conjunction. The dynamics resides in the fact that the interpretation of the first conjunct ϕ is updated. Sequences satisfying ϕ are updated with the information that $n(\psi)$ more terms have occurred when we evaluate ϕ's conjunction with ψ.

Let us briefly see how *PLA* handles key-note examples of systems of dynamic interpretation:

(9) A diver found a pearl. $[\exists x(Dx \wedge \exists y(Py \wedge Fxy))]$

The length of formula [9] is 2, and straightforward calculations show that:

(10) $cd\vec{e} \models_{M,g} [9]$ iff $c \in E(D)$, $d \in E(P)$ and $\langle c, d \rangle \in E(F)$

that is, if, and only if, c is a diver who found pearl d. Next consider:

(11) She sold it to a tourist. $[\exists z(Tz \wedge S\mathbf{p}_1\mathbf{p}_2 z)]$

The length of this formula is 1, but it imposes restrictions, not only on the first element which it mentions, but also on two preceding subjects. The formula is satisfied by a sequence $bcd\vec{e}$:

(12) $bcd\vec{e} \models_{M,g} [11]$ iff $b \in E(T)$ and $\langle c, d, b \rangle \in E(S)$

that is, if and only if b is a tourist which c sold d to. In the conjunction of [9] and [11], the two pronouns are resolved:

(13) A diver found a pearl. She sold it to a tourist. $[9 \wedge 11]$

The length of the conjunction is 3 and since it is resolved, it only places constraints on these first three elements of a satisfying sequence:

(14) $bcd\vec{e} \models_{M,g} [13]$ iff $cd\vec{e} \models_{M,g} [9]$ and $bcd\vec{e} \models_{M,g} [11]$ iff
c, a diver, sold d, a pearl c found, to tourist b

The pitch:

The *PLA*-notion of conjunction is meant to account for the fact that when we extend our evaluation of an utterance of ϕ with that of a subsequent utterance of ψ, we take into account that $n(\psi)$ more terms with referential intentions have been used after the conjunction of ϕ with ψ. The *PLA*-satisfaction of a conjunction $\phi \wedge \psi$ thus is stated in terms of the satisfaction of both ϕ and ψ, while it bears witness of the fact that satisfaction of ϕ is evaluated $n(\phi)$ terms before that of ψ is assessed. This notion of conjunction clearly shows the indexical nature of *PLA*'s system of interpretation. The satisfaction of $\phi \wedge \psi$ is stated in terms of the satisfaction of ψ, and the satisfaction of ϕ *before all of the indefinites of* ψ *have been used*, and this is why pronouns in ψ may pick up intended referents mentioned before. Thus conceived, all that is dynamic about the *PLA*-notion of conjunction—and, hence, about the *PLA*-notion of interpretation—is that it accounts for the intuition that in a succession of two assertions, one assertion literally precedes the other.[27]

(Conclusions include discussion of speech acts other than assertion.)