A Strategy for Building a Framework for Computational Semantics (The Way Forward)

The FraCaS Consortium:

Robin Cooper, Dick Crouch, Jan van Eijck,
Chris Fox, Josef van Genabith, Jan Jaspars, Hans Kamp,
David Milward, Manfred Pinkal, Massimo Poesio, Steve Pulman

FraCaS

A Framework for Computational Semantics
LRE 62-051

Deliverable D14

January 1996

A Framework for Computational Semantics

CWI Amsterdam

University of Edinburgh
Centre for Cognitive Science and Human Communication Research Centre

Universität des Saarlandes
Department of Computational Linguistics

Universität Stuttgart
Institut für Maschinelle Sprachverarbeitung

SRI Cambridge

For copies of reports, updates on project activities and other FRACAS-related information, contact:

The FRACAS Project Administrator
University of Edinburgh
Centre for Cognitive Science
2 Buccleuch Place
Edinburgh EH8 9LW, Scotland, UK
fracas@cogsci.ed.ac.uk

Copies of reports and other material can also be accessed via anonymous ftp from `ftp.cogsci.ed.ac.uk`, directory `pub/FRACAS`.

©1996, The Individual Authors

No part of this document may be reproduced or transmitted in any form, or by any means, electronic or mechanical, including photocopy, recording, or any information storage and retrieval system, without permission from the copyright owner.

Contents

A Strategy for Building a Framework for Computational Semantics (The Way Forward)

The FraCaS Consortium:

Robin Cooper, Dick Crouch, Jan van Eijck,
Chris Fox, Josef van Genabith, Jan Jaspars, Hans Kamp,
David Milward, Manfred Pinkal, Massimo Poesio, Steve Pulman

Abstract

In this document we outline what is meant by a framework for computational semantics and describe three possible approaches: a logical approach, a conceptual approach and a toolbox approach. Various examples of the approaches are given. For the logical approach we describe a way towards providing a unification of various dynamic semantics. For the conceptual approach we summarize the various kinds of conceptual tools described in D15. For the toolbox approach we provide an informal specification of the implementation which was built in the last part of the FraCaS project.

This deliverable was originally entitled "The Way Forward" in the Technical Annex. The intention was to describe our strategy for building a framework after having reviewed previous relevant work in the preceding deliverable. It was brought to our attention that the original title suggests that the deliverable should contain material relating to the future development of research in computational semantics after the end of the project (material that in fact is included in the final deliverable D16). In order to remove this confusion we have changed the title slightly.

Chapter 1

The Notion of a Framework

What exactly is a 'Framework for Computational Semantics' and why should anyone want such a thing?

Let us address the last question first. Semantics, at the present stage of development, does not offer a set of off-the-shelf packages to the application developer. In fact, even to the insider, there seems to be relatively little consensus over the basic components of a semantic description, as opposed to, say, syntax, where virtually every rule-based computational system uses some form of unification grammar; or morphology, where the overwhelmingly dominant paradigm is that of two-level finite state morphology. Instead, with very few exceptions, computational semantics consists largely of isolated analyses of interesting phenomena, or discussions of theoretical properties of formalisms, each one often a *tour de force* in its own right, but difficult to relate to each other, and often not even (as we might say) closed under conjunction, since they make incompatible assumptions.

Advances in technology, particularly natural language processing technology, only come when people are able to build on the results already achieved in the field. Otherwise one is condemned to build and rebuild endless toy systems, each one perhaps more sophisticated theoretically than its predecessor, but not having any great increase in functionality. One of the aims of FraCaS is to encourage the field of computational semantics to achieve the status of an engineering discipline, and this will only come when there is a stable body of knowledge and techniques that can be communicated within a shared body of assumptions. It is this 'stable body of knowledge and techniques that can be communicated within a shared body of assumptions' that is what we mean by a framework for computational semantics.

In earlier work we described three different ways in which this notion might be interpreted at a more detailed level: the 'logical', the 'conceptual', and the 'toolbox' interpretations. These are by no means mutually exclusive, of course, and it would be desirable to have all three of them. In this section we elaborate on what these different conceptions of a framework are, and on some of the difficulties that stand in the way of attaining them.

1.1 A logical framework

A common feeling among those encountering new semantic theories is that much of what is being offered are old insights in new notations. Sometimes this is true, sometimes not. It would be nice to have a way of deciding when this was the case.

A traditional way of comparing different theories about the same domain is to provide a neutral, canonical representation language, with an independently stated syntax and semantics, and then try to provide descriptions of the different theories within this canonical notation. This enables the comparison of apparently different constructs to be made by relating them to a common representation about which there is prior shared agreement.

This exercise has been thought valuable on its own account even for the assessment of individual theories: in the earlier part of this century, the 'logicist' approach was adopted towards areas of science and mathematics, the canonical notation being first order logic. The aim here was to show how properties of theories could be related, or reduced to, or separated from properties that were purely conceptual or logical.

In computer science, the canonical notation has typically been set-theory or something including that (higher order logic, model theory, domain theory, category theory), and the aims have been partly logicist, to try to pin down the non-logical components of theoretical constructs, and partly comparative, for example in providing denotational semantics for programming languages so that proofs of equivalence, etc. can be formulated. The exercise of providing denotational semantics for programming languages was motivated by the fact that expressions having identical syntax might nevertheless correspond to different sequences of operations inside a computer. Without some way of describing the intended interpretation in neutral terms there is no way of deciding which implementation is correct. Thus providing a denotational semantics is useful both in practical and theoretical terms, guiding implementation and defining correct interpretation.

Within linguistic theory, a somewhat analogous approach has also been taken toward theories of syntactic representation. 'PATR', the original general purpose unification grammar formalism, which has an independently defined syntax and semantics, has been used to formalise aspects of different theories of syntax allowing the investigators to pinpoint exactly where the requirements of these theories exceed the expressive power of simple unification. In the case of Generalised Phrase Structure Grammar, for example, this exercise uncovered some non-monotonic aspects of the formalism which had not previously been recognised as such.

However, if we try to apply this concept of framework to semantic theories, various practical and methodological difficulties arise. Firstly, how should we choose the neutral canonical representation? The usual range of possibilities (e.g. set theory, first order logic) is not completely appropriate, because the inadequacy of these as foundations for natural language semantics is often a starting point for the theories under discussion.

It might be possible to use some other computationally implemented system: Higher Order Logic, or a version of Constructive Type Theory which could offer a more extended meta-

theoretical notation. While there have been efforts to use these as representation languages for semantic theories they do not really have the status of independently accepted canonical notations that set theory, FOL, or indeed PATR have, and which would be necessary for the exercise to have widespread acceptability within the community.

Secondly, at what level should we do the comparison? At the level of the theoretical constructs characteristic of the theory, or at the level of the semantic description of individual languages? Since in many cases the properties of particular approaches to semantic theory have only been illustrated with respect to the analyses of phenomena in particular languages, requiring the former would not be possible. But if the analysis of a particular linguistic phenomenon is involved, then the theories may be incommensurable because they interpret the data in different ways, even if the underlying theoretical mechanisms might ultimately prove to be identical.

An alternative tack, noting that some semantic theories also try to provide foundational systems, would be to choose one semantic theory and use it as a metatheory to encode the others. For example, Cooper has used situation theory to encode a version of DRT, and of generalised quantifier theory. For specific aspects of other theories, the FraCaS project has carried out similar exercises: van Genabith and Crouch have shown that translation between LFG f-structures and QLFs is quite straightforward, and Jaspars has shown how to encode various types of dynamic logic within one framework.

The results of such exercises have to be treated with a certain amount of caution. Most notations that are rich enough to do anything are also rich enough to do everything: they are Turing equivalent. Thus it is not surprising that one theory can be translated into another. The real issue concerns how difficult this is: whether the concepts and representations of one theory correspond rather closely to those of another, or whether an elaborate and intuitively artificial coding scheme has to be set up to carry the programme through.

One way to measure this is to see whether the coding requires an extra level of indirection, or extra, translation-specific machinery. Let us take QLF and DRT as an example. DRT provides two pieces of machinery for constructing DRSs: the DRS construction rules which run off the syntax; and the various 'reference resolution' rules which are assumed to be able to locate suitable antecedents for pronouns etc. and add suitable equations to the DRS to represent this contextual aspect of interpretation. QLF also provides for direct QLF construction from the syntax, and for resolution rules: these, however, instantiate meta-variables rather than add equations (although this is logically equivalent).

Given the flexibility provided by reference resolution rules (currently an under-constrained part of everyone's formalism), the QLF language used in the CLE seems powerful enough to directly encode DRT descriptions of a wide range of phenomena. Consider the following example:

(1.1) Every farmer who owns a Mercedes thrives

```
DRT:
  [[x,y:farmer(x),mercedes(y),owns(x,y)] => [thrives(x)]]
```

```
(R)QLF:
   [+f]:thrives(term(+f,<type=q,lex=every>,Restriction,forall)).
   where Restriction =
    \x.and(farmer(x),
           [+m]:own(x,term(+m,<type=q>,lex=a>,mercedes,exists,\x.x=x))),
```

These two expressions have logically equivalent first order translations. However, given the semantics of QLF as defined in D8, it is not possible to directly encode the 'donkey' equivalents in a similar way. (The CLE actually analyses these as E-type pronouns). Consider:

(1.2) Every farmer who owns a Mercedes drives it

```
DRT:
  [[x,y:farmer(x),mercedes(y),owns(x,y)] => [u:drives(x,u),u=y]]

(R)QLF:
    [+f]:drives(term(+f,<type=q,lex=every>,Restriction,forall),
                term(+i,<type=pro,lex=it>,\x.x=+m,exists)).
      where Restriction =
        \x.and(farmer(x),
               [+m]:own(x,term(+m,<type=q>,lex=a>,mercedes,exists,\x.x=x)))
```

Assuming that the reference resolution process interpreted the pronoun as associated with the index of 'mercedes', (by giving it the property `\x.x=+m`) the result will not be validly interpretable (for the analogue of the scope reasons traditionally pointed out).

What we could do, however, is to write a resolution rule specific to the indefinite article, which had the effect of giving it wide scope when in certain contexts, and also interpreting it as a universal in those contexts. (A rather similar rule is ised to resolve 'any'). Then the resolved QLFs for the two examples would be:

```
(R)QLF: [+m,+f]:thrives(term(+f,<type=q,lex=every>,Restriction,forall))
     where Restriction =
      \x.and(farmer(x),
             own(x,term(+m,<type=q>,lex=a>,mercedes,forall,\x.x=x)))

(R)QLF: [+m,+f]:drives(term(+f,<type=q,lex=every>,Restriction,forall),
                       term(+i,<type=pro,lex=it>,\x.x=+m,exists)).
      where Restriction =
       \x.and(farmer(x),
              own(x,term(+m,<type=q>,lex=a>,mercedes,forall,\x.x=x)))
```

The first order translation of this RQLF is equivalent to the first order translation of the DRT representation.

This might or might not be a defensible analysis but it can certainly be done with relatively little effort. If most DRT phenomena do not require even this small effort, then one would

have to conclude that there is quite a large overlap between the two approaches, at least as far as the ultimate representations are concerned. But if every new phenomenon requires one or more new resolution rules, then it is clear that QLF is really being used as a programming language within which to recode DRT, and the fact that this extra machinery is necessary shows that they do not overlap much.

1.2 A conceptual framework

Any discipline has a range of shared concepts, understanding of which is presupposed by any research or development activity within it. The early years of an area of investigation are usually characterised by intense debate about the nature of the most basic concepts, even about the boundaries of the discipline. In (computational) semantics this is the stage of development: there is debate about the nature of propositions, events, situations, presupposition, focus, quantification, reference etc. There is uncertainty about how many of the phenomena falling under the man in the street's concept of meaning are semantic, in the strict sense, and how much are to do with discourse (if that is a separate area) or non-linguistic knowledge. There are a variety of different interpretations of the most basic notions.

What we have tried to do in the FraCaS project is to map out those areas where there is reasonable agreement. We have done this on several different levels:

- a glossary (the 'Bluffer's Guide') of the most common terms of art used in semantics, along with (we hope) a consensus description of their meaning and some pointers into the relevant literature.

- a survey (D5,D7) of what we believe to be the most central semantic phenomena, exemplified for English, along with a test suite (in D16) illustrating characteristic patterns of inference associated with these semantic phenomena. While none of this is definitive, we hope that this provides a first approximation to the current consensus notion of the investigative domain of computational semantics. (Of course, as the discipline develops, the boundaries of the domain may change. During the course of the project, we have become more conscious of the difficulty of maintaining a principled separation between syntax-driven semantics, lexical semantics, and discourse structure, and most of us would now feel that little purpose was served in maintaining a very strict separation.)

- a description of a common set of semantic/logical concepts which are required, in some form, in any theory which is capable of describing the central semantic phenomena: these concepts include abstraction, quantification, proposition, predication, connectives, and variables. Our original intention was to provide an abstract formulation of these notions in such a way that the different ways in which they are implemented within different semantic theories could be meaningfully compared. Cooper carried out part of this programme for the example of 'predication' but it soon became clear that to

complete this line of investigation would consume more resources than were available on the project.

One further area which would be worth pursuing, but which we have not done for lack of time, would be to enrich the description of phenomena with an indication of assumptions shared by the most successful analyses of them. Obviously, analyses are usually couched in one particular theory, but for many phenomena it should be possible to abstract the common elements of an analysis in a relatively theory-neutral way. In the domain of syntax, one can point to analyses which can be described in a relatively informal way: for example, Chomsky's 'On Wh-movement' analyses of a range of phenomena (questions, topicalisation, relatives, comparatives, 'tough'-movement) in terms of a single movement operation. This analysis can be implemented within a variety of different formalisms, many of which share very few if any assumptions of the framework Chomsky was working in. It would be extremely useful to have a catalogue of analyses in semantics at a similar level of informal detail, such that the potential implementor could know what his system would have to do to get the facts right without having either to re-invent the wheel, or spend weeks reading earlier papers. To some extent, D9 can be seen as partly fulfilling this role. However, all the analyses described there are couched within the framework of a particular theory. It is thus not easily possible to disentangle the general from the theory-specific.

For some phenomena and some analyses, disentangling general techniques from theory specific techniques is relatively easy. For example, for generalized quantifiers there is a relatively well accepted general account. For other phenomena, disentanglement is non-trivial but looks possible. An example here is the treatment of definite descriptions in situation semantics using the notion of a resource situation. Finally there are phenomena where it is not at all clear what any common solution should look like. An important example of this is the semantics of abstract objects, and abstract object anaphora. We expand on this in more detail in D15.

1.3 A toolbox

By a 'toolbox' we mean a set of techniques, algorithms or representations which can be taken off the shelf and incorporated into an implementation. At the present stage of development these are for the most part better thought of as specifications than as implementations. An example might be the Hobbs-Shieber quantifier scoping algorithm. Another example might be the 'semantic operators' proposed by Johnson and Kay. Further examples are provided by the 'aspectual coercion' calculus described elsewhere in the FraCaS project, and the library of programs for semantic composition described in D16. The latter is an implementation primarily intended for pedagogical rather than for system development purposes.

Of course, to be maximally useful, a tool must make as few requirements as possible on the rest of the infrastructure that is presupposed. For this reason, it is extremely difficult to point to a long list of candidate tools in semantics, for much of what has been achieved

so far has made specific assumptions about, for example, the syntax which supports it, the representations that are produced, and so on. One way to try to overcome this dependency is to formulate parameterised versions of them, where the values for the parameters represent what is required of any theory or implementation which will use the tool. In principle, this is an adequate response: in practice, it is often necessary to generalise to the worst case in order to achieve true independence from particular theoretical assumptions. This can make the incorporation of the tool into an implementation rather clumsy. A sure sign that this is happening is where the parameterisation is more complex than an ad-hoc special purpose component would be.

There is an overlap between the notion of a toolbox, and some aspects of a conceptual framework. For example, if one had a repertoire of basic semantic concepts like 'abstraction', or 'variables', along with a range of different choices as to how they could be concretely implemented, that would be just as much a tool as an off-the-shelf aspectual coercion module, albeit at a deeper level and with wider consequences for the remainder of the implementation.

Also, the notion of a toolbox presupposes that some aspects of a logical framework are in place, for there must be a sufficient similarity between the functioning of the tool, and the presuppositions of the remainder of the implementation, for the tool to be useful. A flint axe is not much use in a modern engineering works; and a computer controlled lathe is redundant in a place without a power supply.

Thus the various ways in which the notion of a framework can be interpreted are by no means independent of each other.

Chapter 2

How to Build a Framework

2.1 Introduction to the logical approach

In this section we describe an example of the logical approach applied to dynamic semantics. Here we provide model-theoretic specifications of various dynamic logics in terms of the possible world semantics of intuitionistic logic. Additionally, we define a (dynamic) modal language (canonical notation) to reason about these models [Van Benthem, 1991].

This enterprise makes it possible to compare different dynamic semantic theories of natural language in a precise formal manner. Moreover, such a machinery paves the way for importing knowledge about dynamic logics from outside the field of formal/computational linguistics and makes it possible to extract technical/logical questions from ongoing research in dynamic semantics in an unambiguous way.

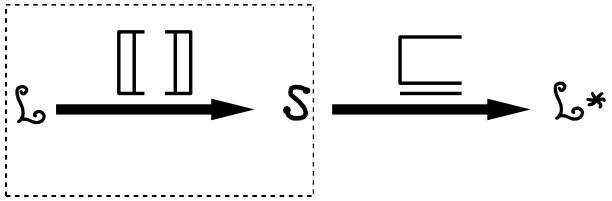
Maybe it seems that this kind of logical research is rather retrospective/corrective in nature when it comes to its linguistic merits. However, there is also a strong innovative aspect related to these logical investigations for linguistic theory, namely, when we want to combine other semantic theories with dynamic semantics. The logical formalism which we will describe in this section extrapolates the 'statics' and the 'dynamics' of a dynamic logic. In fact, it explains how the latter evolves from the former. This suggests that implementation of 'dynamics' into a given (static) semantic theory can be achieved by following the process of 'dynamification' as will be explained in this section below.

2.1.1 Unified dynamic logic

Change is an important concept in such diverse research areas as computer science, cognitive science and linguistics. This has led to a great proliferation of so-called *logics of change* (or *dynamic logics*) in these areas. In linguistics, for example, the dynamics of variable assignments has proven to be very effective for the treatment of anaphoric dependencies in discourse. The dynamic movement in general is concerned with a much-wider dynamification of semantic parameters, some of which are directly relevant for the semantics of discourse.

Since in different systems different aspects of the interpretation are dynamified, comparing and/or unifying the various logics is a non-trivial yet interesting task. [Van Benthem *et al.*, 1995] note that developing a single logic in which all relevant aspects of semantics are combined will surely lead to 'a disaster' and that it would be more useful to have 'one common general purpose logic', in which the various approaches can work 'in tandem'.

In [Jaspars and Krahmer, 1995] we intend to come to a *bird's-eye view* on the gamut of dynamic theories. To arrive at this perspective we use a version of *Dynamic Modal Logic* (DML) as introduced in [Van Benthem, 1991] and [De Rijke, 1992]. This logic has been developed to model general reasoning about extending and reducing information states. An *information-model* M for a certain language $\mathcal{L}$ looks as follows: $M = \langle S, \sqsubseteq, [\![.]\!] \rangle$, where S is a non-empty set of information-states, $\sqsubseteq$ a pre-order over S and $[\![.]\!]$ an interpretation-function. The extension of $\mathcal{L}$ ('the static language') with modal operators will be designated as $\mathcal{L}^*$ ('the dynamic language'). The following picture visualizes the perspective:



The dashed region may be thought of as (part of) some well-known logic. The non-dashed part allows us to *dynamify* the logic. For this purpose the semanticist has to implement his specific philosophy of the flow of information into the definition of $\sqsubseteq$. To reason about it, modal operators are added to $\mathcal{L}$ resulting in the extended $\mathcal{L}^*$.

In [Jaspars and Krahmer, 1995] it is shown that many well-known dynamic theories can be fully characterized by specifying the five parameters mentioned above: $\mathcal{L}$, S, $[\![.]\!]$, $\sqsubseteq$ and $\mathcal{L}^*$. In general, the static parts of these systems are richer than in DML (where the static part is usually restricted to propositional logic), while the dynamic parts consist of a limited set of *at most* four modal operators to be defined below (this may be contrasted with the relational wealth of DML).

More concretely, suppose that we have specified a language $\mathcal{L}$ and its extension $\mathcal{L}^*$, and that we have an information-model $M = \langle S, \sqsubseteq, [\![.]\!] \rangle$, then we can define various interpretations. First, there is the static interpretation represented as $[\![.]\!]_M$. When $\varphi \in \mathcal{L}$, $[\![\varphi]\!]_M$ is just $[\![\varphi]\!]$. In order to extend $[\![.]\!]_M$ to the full $\mathcal{L}^*$, we define the meaning of a proposition with respect to an information-state s. $\sqsubseteq$ is used to define what an *expansion* (represented as $+$) or a *reduction* ($-$) of a state s is, these functional interpretations being defined as follows:

$$[\![\varphi]\!]^+_{M,s} = \{t \in S \mid s \sqsubseteq t \;\&\; t \in [\![\varphi]\!]_M\}$$

$$[\![\varphi]\!]^-_{M,s} = \{t \in S \mid t \sqsubseteq s \;\&\; t \notin [\![\varphi]\!]_M\}$$

Minimal expansions (or *updates*, represented as $+\mu$) and minimal reductions (*downdates*,

$-\mu$) are given by $\mathsf{min}_M([\![\varphi]\!]^+_{M,s})$ and $\mathsf{max}_M([\![\varphi]\!]^-_{M,s})$.[1] These context-sensitive interpretations allow us to define modal operators to facilitate explicit dynamic reasoning.[2] Let act be some action (like $+$, $-$, $+\mu$ and $-\mu$). For every $[\![.]\!]^{\mathsf{act}}$ there are two corresponding modalities —the box-modalities $[.]^{\mathsf{act}}$, and the diamonds $\langle . \rangle^{\mathsf{act}}$— with the following interpretations:

$$\begin{aligned} [\![[\varphi]^{\mathsf{act}}\,\psi]\!]_M &= \{s \mid [\![\varphi]\!]^{\mathsf{act}}_{M,s} \subseteq [\![\psi]\!]_M)\} \\ [\![\langle\varphi\rangle^{\mathsf{act}}\,\psi]\!]_M &= \{s \mid [\![\varphi]\!]^{\mathsf{act}}_{M,s} \cap [\![\psi]\!]_M \neq \emptyset\} \end{aligned}$$

The $\mathcal{L}^*$ parameter of a dynamic logic specifies which modals are used. For example, $[\varphi]^+\psi$ is the statement that each expansion of the current state with φ yields a ψ-state. $\langle\varphi\rangle^{-\mu}\psi$ expresses that at least one minimal reduction (downdate) with φ brings us to a ψ-state.

How can we do dynamic semantics in this set-up? In [Jaspars and Krahmer, 1995] this question is answered by explicitly looking at various logics of change and showing how they can be reformulated in terms of information-models. The most simple examples are the constructive logics, which is no surprise in light of the origin of the information-models as a possible-worlds semantics for intuitionistic logic. An easy extension with linguistic relevance is so-called *Data-Semantics* of [Veltman, 1985] which differs from the constructive logics in that it also employs $\langle . \rangle^+$ operators. Other instances are the *Update Semantics* of [Veltman, 1991] and various theories of *belief revision* as defined in [Gärdenfors, 1988]. These two formalisms employ minimal (μ) operators over a single information model. Finally, [Jaspars and Krahmer, 1995] discuss various logics devised to deal with the semantics of discourse, in particular *Discourse Representation Theory*, *File Change Semantics* and *Dynamic Predicate Logic*. For the sake of illustration, let us briefly discuss two examples: the theory of belief-revision of [Alchourrón *et al.*, 1985] and Kamp's discourse representation theory [Kamp and Reyle, 1993].

Example. AGM-theory of belief revision can be given a model theory as follows: we take $\mathcal{L}$ to be classical propositional logic and the dynamic modal extension supplements only the up- and downdate operators. This language is interpreted over a single $\mathcal{L}$-information model:

$$\begin{aligned} S &= \text{the collection of all deductively closed sets (theories).} \\ T_1 \sqsubseteq T_2 &= T_1 \subseteq T_2 \\ [\![\phi]\!] &= \{T \in S \mid \phi \in T\} \end{aligned}$$

This model suffices for a DML-imitation of the so-called *full meet*-contraction and revision. More cautious forms of revisions can be modeled by replacing the information order by a subrelation of the inclusion relation. Individual alternatives (such as the partial meet and maxichoice revision) corresponds to additional specific constraints on this refined information order (for details see [Jaspars and Krahmer, 1995]).

[1] With $\mathsf{min}_M(T) = \{x \in T \mid \forall y \in T(y \sqsubseteq x \Rightarrow x \sqsubseteq y)\}$ and $\mathsf{max}_M(T) = \{x \in T \mid \forall y \in T(x \sqsubseteq y \Rightarrow y \sqsubseteq x)\}$.

[2] Most often the dynamic meaning of a proposition is denoted by a relation, which can be obtained by abstraction over the context: $[\![\phi]\!]^{\mathsf{act}}_M = \{\langle s,t\rangle \in S \times S \mid t \in [\![\phi]\!]^{\mathsf{act}}_{M,s}\}$ where act is an action type.

Example. The DML-specification of first order DRT is as follows. The static language is a normal first–order language without quantifiers (but with identity), and the dynamic extension adds the operators $[.]^{+\mu}$ and $\langle . \rangle^{+\mu}$. Its information models are of the form:

$$\begin{aligned} S &= \text{the collection of all partial variable assignments } \mathcal{A} \\ &\quad \text{over a given first order model.} \\ a \sqsubseteq a' &\Leftrightarrow \text{if } a(x) \text{ is defined, then } a'(x) = a(x). \\ [\![\phi]\!] &= \{a \in \mathcal{A} \mid M \models \phi[a]\} \end{aligned}$$

Discourse Representation Structures (DRSs) reappear as dynamic modal operators, and conditions as regular propositions of the full DRT-language. For example, a DML-formula like $[\phi]^{+\mu} \langle \psi \rangle^{+\mu} \top$ over the information models above is the same as $\phi \Rightarrow \psi$ in DRT.

2.1.2 Concluding remarks

Let us finally make some general remarks about the present, unified perspective.

- First of all, the dynamic 'common general purpose logic' we have presented and examined above points in the direction of a 'core dynamic logic'. We can start characterizing *common actions* found in all 'logics of change' as well as *distinguishing axioms* which separate between them. This also makes it possible to reduce proliferation in the field, which is very useful in light of the ever increasing divergence of dynamic logics.

- It allows for transfer of logical knowledge between systems. For instance, the constructive logics have been around for a long time and have been studied extensively. We may hope that this contributes to the meta-theoretical study of newer dynamic logics.

- Finally, it not only provides us with a method to characterize *existing* dynamic systems in terms of possible world semantics, it also enables us to create *new* dynamic systems. For that we only need to define a 'dynamic' component on top of an existing static logic. One of the advantages of this 'generating capacity' is that existing semantic theories can quite easily be extended to enhance the analysis of natural language phenomena which involve sophisticated modes of information flow. For instance, [Van Eijck, 1995] shows that it is useful to study presupposition from the general DML perspective. In [Jaspars and Kameyama, 1995] defeasible logic has been dynamified in the DML fashion for preferential reasoning over possible anaphoric resolutions.

2.2 Introduction to conceptual tools

In D15 we will present a selection of conceptual tools that we have found useful in moving towards a semantic framework. The selection is not meant to be exhaustive in any sense but we feel that it represents a number of the kind of conceptual clarifications which are necessary when one begins to think in framework terms rather than in terms of individual approaches. Conceptual tools fall into several groups:

1. those which represent concerns shared by all the approaches for which conceptual work is needed to unify them. This is represented in our conceptual tools by the section *event structure*.

2. those which are general logical tools which can have equal application to different semantics approaches. This is represented by *Paradoxes*.

3. those which represent a concept which currently is only employed in one approach but which could usefully be introduced into other approaches. This is represented by *resource situations* and *dynamics and situations*.

4. those which show how to map between different approaches. This is represented by two tools: *Glue languages, QLF, UDRT and LFG* and *Austinian propositions and property theory*.

5. those which represent mainly computational concerns which need to be represented in such a way that they have application to all the approaches. Our example of this is *underspecification*

We now give a brief summary of each of the conceptual tools that are discussed in D15.

Event structure

A central concern for all the semantic approaches concerns the structure of events and what exactly the relationship is between events and situations on the one hand and facts, propositions and event types on the other. There is no single agreed on ontology even within a single approach. For example, do we need to distinguish facts and propositions or are facts just true propositions? If facts are not propositions can they be considered as an event type or are they something different? Are there clear linguistic ways of distinguishing reference to these entities? These are central topics of current theoretic al research and we cannot, in a project of this nature, hope to solve the problems and present a coherent view. However, we have attempted in this section (which in fact contains three subsections by different authors) to take recognition of the fact that reasoning about events is of central importance for practical applications and in order to provide useful results the problems must be approached in a framework oriented way.

Paradoxes

One of the problems that the working linguist or system builder confronts in trying to put together a formal system is that paradoxes may creep in and make the system inconsistent. Within logic there are now a small number of general and powerful techniques which can be employed to avoid this pitfall. This section makes these techniques available in a simple way so that the non-logician who nevertheless wants to build a formal system can take advantage of them.

Resource situations

The notion of resource situation to represent a certain kind of contextual dependence for definite descriptions and restrictions on the range of quantifiers has been around for many years in situation semantics. The idea of a resource situation as opposed to a context set is computationally attractive because machines have to be able to deduce such restrictions from information about situations. However, the idea has been difficult to import into other approaches because of the considerable weight of situation theoretic baggage that comes along with it. This is an attempt to present the idea with at least less of the baggage than usual.

Dynamics and situations

This section shows one way in which concepts from dynamic semantics and situation semantics can be put together in the analysis of difficult cases of anaphora.

Glue languages, QLF, UDRT and LFG

This presents a number of translation functions mapping between the f-structures of Lexical Functional Grammar, QLF and UDRS. The translation functions provide a useful tool for semantic comparison. The section shows how f-structure can be given direct or indirect semantic interpretations, either in the style of QLF or of UDRS. It also shows how scope underspecification in QLF can be related to glue language techniques based on linear logic, opening the path to an alternative and hopefully more perspicuous semantics for underspecified representations like QLF.

Austinian propositions and property theory

Both property theory and situation theory employ an axiomatic approach to structured semantic universes in order to deal with intensionality and there are clear similarities in the way they go about this. One clear difference in the two theories is the existence of Austinian propositions in situation theory. It seems appropriate therefore to show a way of introducing Austinian propositions (and thereby also situations, of course) into property theory.

Underspecification

The representation of underspecified meaning is a computational concern which also turns out to have a good deal of theoretical interest attached to it. Within the life of the project we have spent a good deal of time discussing the nature of underspecification and trying to abstract away from individual proposals that have been made within distinct approaches. This section represents some of our attempts to take a more framework oriented view towards underspecification and to isolate the techniques from their particular approaches so that they can be imported into other approaches.

2.3 Informal specification of the toolbox implementation

The framework tool illustrates the kinds of compositional techniques which have been discussed in the FraCaS reports. While this does not constitute a full semantics workbench, we do feel it could provide the kernel of a workbench and also provides a valuable teaching aid which makes some of our discussion concrete in computational terms. It also provides a graphical user interface which aids the computational exploration of the different approaches and techniques.

At the heart of the framework tool is a parameterised interface which allows the user to specify various grammars and to specify one of a range of semantic formalisms such as first order logic, intensional logic, lambda calculus with generalized quantifiers, discourse representation structures, λ-DRT and extended Kamp notation (EKN). The system provides support for converting internal prolog representations to "pretty" linear representations (e.g. using $\lambda x.\, run(x)$ rather than $X\hat{}run(X)$) and to graphical notations (e.g. DRSs and EKN). The representations can be zoomed and various layout features can be altered interactively.

The tool includes support for the following operations: functional application, generalized function application, functional composition, simultaneous abstraction, β-reduction (also for simultaneous abstraction), reduction of Austinian propositions and distribution of restrictions in EKN, quantifier storage and semantics driven composition.

To make it as clear as possible where theories share concepts and operations, the main component of the tool, CLEAR (Computational Linguistics Educational and Research Tool) provides small fragments for the different semantic formalisms which share as much code as possible. A larger fragment illustrating the application of computational semantics techniques based on simultaneous abstraction was also implemented. CLEAR can be hooked to CLIG, a stand alone interactive grapher which provides graphical representations of semantic representations such as DRSs and EKNs. Finally we provide a specification of a reasoning module for aspectual class which could be plugged into the tool. This points the way to an experimental environment into which various generic reasoning modules could be plugged to allow the investigation of the compatibility of these modules with the various semantic representations available within the framework tool.

2.3.1 CLEAR: Computational Linguistics Education and Research

There are currently a wide variety of semantic formalisms used in teaching and research, for example, various versions of 'Montague-Grammar', DRT, and Situation Semantics. As different as they look on first sight, they share many common assumptions, and provide similar treatments of many phenomena. CLEAR allows exploration and comparison of different semantic formalisms, and their interaction with syntax. This enables users to get an idea of the range of possibilities of semantic construction, and also to see where there is real convergence between theories.

The system provides a teaching tool, and a library of algorithms together with test suites. The teaching tool provides an interactive and extendible environment to support students

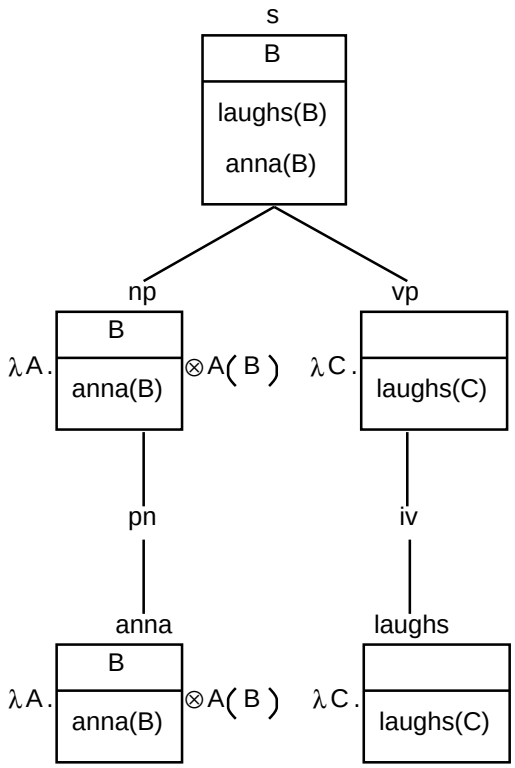


Figure 2.1: Representation of `Anna laughs` in λ-DRT

in learning and comparing different formalisms. For example users can generate a syntax tree which is either unannotated, annotated with syntactic rule names, or with semantic operations (such as apply the first daughter to the second). The user can explore various syntax to semantics mapping, and can interactively control the step-by-step construction of a semantic representation. For example, the user can choose when to apply lambda reduction, quantifier storing or discharging, to apply a meaning postulate, or to merge two DRSs. An example of a completed construction is given in Figure 2.1.

As a teaching tool, CLEAR has two main uses. Firstly it can be used for individual study, not just for exploring semantic construction, but also as the basis for student project work involving the implementation of new modules for specific linguistic phenomena. Secondly, CLEAR provides the possibility for the teacher to provide interactive demonstrations and to produce example slides and handouts.

Because such a tutorial system has to be based largely on standard routines and algorithms that are fundamental for the area of computational semantics, it makes sense to build it on top of a library that can also be used for other applications. Therefore a second aspect of CLEAR is the provision of a set of well documented programs which could form the nucleus

of a larger library of reusable code for this field. These programs include algorithms which interface with the grapher, allowing users to input Prolog syntax for e.g. a DRS structure, and to output a graphical object which can be displayed or printed.

2.3.2 A grammar illustrating computational semantics techniques based on simultaneous abstraction

The second aspect of our effort on developing a library of techniques used in computational semantics was the development of a grammar illustrating how different techniques can be put together and providing a partial implementation of the fragments discussed in D9.

The grammar currently includes

- a treatment of quantified sentences based on generalised quantifiers theory and a treatment of scope ambiguity based on nested Cooper storage;
- a compositional formulation of the treatment of indefinites, anaphoric expressions and accessibility constraints proposed in DRT;
- a treatment of tense and events derived from Situation Theory (i.e., in which events are seen as instances of the type specified by a predicate, instead of being arguments of a relation)
- a treatment of context-dependent and presuppositional aspects of the interpretation derived from Kaplan's theory of context.

The grammar is built around the notion of SIMULTANEOUS ABSTRACTION described in the 'logical tools' section. A 'simultaneous' abstraction is a generalized form of abstraction that denotes a function from *assignments* to values, rather than a function from single objects to values. Simultaneous abstraction has been used in the grammar both to provide a compositional formulation of the process of DRS construction and to formalize nested storage avoiding free variables.

The grammar was implemented on top of EKNTOOL, a tool that supports the development of grammars using simultaneous abstraction originally developed as part of the DYANA project.

2.3.3 CLIG: Computational Linguistics Interactive Grapher

A major part of our work on the tool was the development of a general graphical browser or *grapher* for the graphical notations used in computational semantics. As a matter of fact, our goal was more ambitious: we wanted a grapher which could display most graphical notations familiar in computational linguistics. These notations include usual formats such as trees and *Attribute-Value-Matrices* , together with semantically motivated representations such as DRS boxes and EKN [Barwise and Cooper, 1993]. As you can see e.g. in Figure 2.1, the layout algorithm is furthermore able to display nested graphical structures containing several different notations.

Two other design points which were especially important were the *extendibility* of the grapher for future applications and the possibility of *interaction* between the user, the grapher and the underlying application.

The grapher is written in Tcl/Tk, a programming system for developing graphical user interfaces (GUI) applications (see [Ousterhout, 1994]). It is portable and free for non-commercial purposes. We chose Tcl/Tk as the developing environment for our grapher because it allows the rapid development of user interfaces without having to deal with low level details as in C or C++.

Other features useful for our application included

Scalable Graphs: Graphical output can be resized almost automatically with Tk.

Postscript Output: Graphic canvases can be translated into Postscript which can be included into papers or slides.

2.3.4 An Aspectual Composition Machine

'Determination of aspectual status' of a sentence or phrase means working out whether it describes an event or a state, and what internal structure the event or state might have. While inherent lexical semantic properties place constraints on the possibilities for aspectual status, they do not uniquely determine it, and it is not an objective semantic property of sentences. The aspectual status of a sentence presents a particular way of looking at an episode or situation, or a component of an episode or situation, such that contextual assumptions about its temporal, causal, or relevance relations to other events are required for felicitous interpretation. These assumptions may be implicated thereby (they may be presuppositions).

However, although aspectual status is not an objective semantic property of sentences (in the way, for example, that 'kissing' entails 'touching' is an objective semantic property of 'kiss'), we do need to determine aspectual status in order to know about the truth conditions of a sentence. A sentence which describes a state typically is intended to be evaluated at the current temporal reference point. One describing an event will typically advance the current temporal reference point.

(2.1) Joe sneezed. It was dusty.

(2.2) Joe sneezed. He took out his handkerchief.

Aspectual status may also determine whether we have singular or plural reference:

(2.3) Joe was sneezing as I took the picture.

(2.4) Joe was sneezing all morning.

Although particular verbs may be more naturally interpreted in one way than another ('sneeze' describes events, 'like' describes states), phrases containing them may have different status when the verbs are combined with arguments or modifiers. While there have been some

notable attempts to describe how this happens ([Verkuyl, 1989], [Verkuyl, 1972], [Moens and Steedman, 1988]) it is still the case that:

> '... the central problem for the theory of aspect is to determine how the aspectual characteristics of complex phrases are determined by those of their parts' [Kamp and Reyle, 1993]:570

The aim of this component is to provide a mechanism for doing this which is compatible with a variety of different approaches to semantic theory. The approach taken, largely following Moens and Steedman is to think of the framework of aspectual categories as being like a typing system. We can model it by taking three basic types: `state, process, point`, and allowing for an operator to form complex types `<point,state>` and `<process,state>`.

As well as this basic ontological repertoire there are a variety of 'coercion' operators which allow one type of event or state to be viewed as another. For example:

```
iterate: point -> process
```

You can iterate a point event to become a process consisting of more than one occurrence of the event.

The system works this out in detail for an example fragment. It makes as few assumptions as possible about other aspects of semantics and can thus serve as a general purpose module that can be reused in a variety of implementations.

Bibliography

[Alchourrón *et al.*, 1985] Alchourrón, C.E.; Gärdenfors, P.; and Makinson, D. 1985. On the logic of theory change. *Journal of Symbolic Logic* 50:510–530.

[Barwise and Cooper, 1993] Barwise, J. and Cooper, R. 1993. Extended Kamp Notation. In Aczel, P.; Israel, D.; Katagiri, Y.; and Peters, S., editors 1993, *Situation Theory and its Application Vol. 3.* CSLI, Stanford. chapter 2, 29–54.

[De Rijke, 1992] De Rijke, M. 1992. A system of dynamic modal logic. Technical Report 92-170, CSLI, Stanford, CA. to appear in the Journal of Philosophical Logic.

[Gärdenfors, 1988] Gärdenfors, P. 1988. *Knowledge in Flux: Modelling the Dynamics of Epistemic States.* MIT Press, Cambridge Mass.

[Jaspars and Kameyama, 1995] Jaspars, J. and Kameyama, M. 1995. Linguistic preferences in dynamic logic. Unpublished Manuscript.

[Jaspars and Krahmer, 1995] Jaspars, J. and Krahmer, E. 1995. Unified dynamics. Unpublished Manuscript (forthcoming as a CWI research report). A draft version is available from `http://www.cwi.nl/~jaspars`.

[Kamp and Reyle, 1993] Kamp, H. and Reyle, U. 1993. *From Discourse to Logic.* Kluwer, Dordrecht.

[Moens and Steedman, 1988] Moens, M. and Steedman, M. 1988. Temporal ontology and temporal reference. *Computational Linguistics* 14(2):15–28.

[Ousterhout, 1994] Ousterhout, J. 1994. *Tcl and the Tk Toolkit.* Professional Computing. Addison-Wesley, Reading, Massachusetts.

[Van Benthem *et al.*, 1995] Van Benthem, J.; Muskens, R.; and Visser, A. 1995. "Dynamics". In Benthem, J. van and Meulen, A. ter, editors 1995, *Handbook of Logic and Language.* Elsevier Science. To appear.

[Van Benthem, 1991] Van Benthem, J. 1991. *Language in Action: Categories, Lambdas and Dynamic Logic.* Studies in Logic 130. Elsevier, Amsterdam.

[Van Eijck, 1995] Van Eijck, J. 1995. Presuppositions and information updating. In Kanazawa, M.; Piñon, C.; and Swart, H. de, editors 1995, *Quantifiers, Deduction, and Context*. CSLI. To appear.

[Veltman, 1985] Veltman, F. 1985. *Logics for Conditionals*. Ph.D. Dissertation, University of Amsterdam, Amsterdam.

[Veltman, 1991] Veltman, F. 1991. Defaults in update semantics. Technical report, Department of Philosophy, University of Amsterdam. To appear in the Journal of Philosophical Logic.

[Verkuyl, 1972] Verkuyl, H. 1972. *On the Compositional Nature of the Aspects*. Reidel, Dordrecht.

[Verkuyl, 1989] Verkuyl, H. 1989. Aspectual classes and aspectual composition. *Linguistics and Philosophy* 12(1):39–94.