Understanding Natural Language Instructions: Impact on Representation Formalisms

Barbara Di Eugenio
Computational Linguistics
Department of Philosophy
Carnegie Mellon University
Pittsburgh, PA, 15213 USA
dieugeni@lcl.cmu.edu

1 Introduction

In this paper, I discuss a formalism that I devised and implemented to deal with complex instructions, in particular those containing Purpose Clauses. I argue that such formalism supports inferences that are *an important pragmatic aspect of natural language*, and that at the same time are related to *surface reasoning, based on the syntactic structure of Natural Language*; and moreover, that by using the kind of approach I propose, namely, first define linguistic terms and then use them in the part of the KB concerning the semantics of the domain, we can start bridging the gap between the two representation languages that the organizers of the symposium mention, the first used to capture the semantics of a sentence, the second used to capture general knowledge about the domain. Details on all the topics discussed here can be found in [Di Eugenio, 1993; Di Eugenio, 1994].

2 Motivations for the representation language

The characteristics of the formalism I propose derive from an analysis of an extensive corpus of *Purpose Clauses*, infinitival *to* constructions as in *Do* α *to do* β; as some of these characteristics stem from the inferences necessary to interpret Purpose Clauses, it is with such inferences that I will start.

Interpreting *Do* α *to do* β, where β describes the goal to be achieved, in computational terms amounts to:

(1a) use β as an index into the KB;

(1b) find a collection of methods $\mathcal{M}_l$ that achieve β;

(1c) try to **match** α to an action $\gamma_{l,j}$ that appears as a component in $\mathcal{M}_l$.

These are typical *plan recognition* inferences, eg see [Wilensky, 1983; Pollack, 1986; Charniak, 1988; Litman and Allen, 1990].

In all the work on plan recognition I know of, with the exception of [Charniak, 1988], *match* in step (1c) is taken to mean that α is *instance-of* $\gamma_{l,j}$. However, given the variability of NL action descriptions, we can't assume that the input description exactly matches the knowledge that an agent has about actions and their mutual relations: my research focuses on computing a more flexible match between α and $\gamma_{l,j}$.

The two kinds of discrepancy between input and stored action descriptions I have examined so far concern *structural consistency*, and *expectations* that may need to be satisfied for a certain relation $\mathcal{R}$ to hold between α and $\gamma_{l,j}$.

As regards structural consistency, consider an example such as

(2) *[Turn the screw]*$_\alpha$ *[to loosen]*$_\beta$.

Presumably, the agent has some knowledge about the fact that γ = *Turn screw counterclockwise* results in δ = *Loosen screw*.[1] There must be some way to understand that α = *Turn screw* is an underspecified version of γ, and that in fact γ is the action to be performed. The same kind of reasoning should detect when α is more specific, or inconsistent with respect to γ, such as in the case of *Turn the screw clockwise*.

Now I will turn to computing expectations, that I will use as the main running example in the paper.

[1] This is true of the more common kind of screw.

The expectations I have dealt with so far regard the location of objects that β manipulates, and arise when α changes the *perceptual space* $\mathcal{S}$ the agent has access to. In particular, when α results in the agent going to a place $\mathcal{S}'$ with the purpose of doing β, one can infer $\mathcal{S}'$ to be the site of β: thus, if there are objects that β manipulates, expect them to be at $\mathcal{S}'$. Consider

(3a) *[Go into the kitchen]$_{\alpha_1}$ [to get me the coffee urn]$_{\beta_1}$.*

(3b) *[Go into the kitchen]$_{\alpha_2}$ [to wash the coffee urn]$_{\beta_2}$.*

In both cases, the agent goes into the kitchen, which is then expected to be the location of β_1/β_2. In (3a), but not in (3b), a further expectation that the referent of the coffee urn is in the kitchen is developed. The difference between the two may be explained by appealing to the planning notion of *qualifiers* [Schoppers, 1988; Litman and Allen, 1990], conditions that must hold for an action to be relevant, and are not meant to be achieved. If β has among its qualifiers that an argument be at $\mathcal{S}'$ for β to even be relevant, then a locational expectation develops as in (3a). If not, a weaker expectation arises, as in (3b).

I hope that the previous examples give the reader the gist of my approach to NL instructions, which is to take into account both their surface form and the domain knowledge needed to interpret them: in many systems, the former is in a sense subordinate to the latter. In Ex. (3a), it is not just the domain knowledge, but the purpose relation between α and γ that gives rise to the expectation — in this, my system performs *surface reasoning, based on the syntactic structure of the NL.*

To support this approach, it is necessary to represent action descriptions from both linguistic and planning points of view.

Linguistic characteristics include:

1. Like individuals, sets of individuals, etc., actions should be part of the ontology.
2. The formalism must be able to represent not only the usual participants such as agent or patient, but also means, manner, extent, etc which are pervasive in NL instructions.
3. The action described in the input may not exactly match the actions the system knows about, and it is impossible to define all action descriptions a priori. Therefore, the formalism must be able to support inferences to understand the relation between the input action descriptions and the stored ones.

As regards **planning** knowledge about actions:

1. Classic planning notions, such as action decomposition into substeps, qualifiers, effects, must be provided.
2. The formalism must be able to represent various relations among actions, such as at least temporal relations, *generation* and *enablement*. Intuitively, *generation* holds between α and β if β is done **by** executing α; *enablement* holds between α and β if α brings about conditions necessary to execute β. See [Goldman, 1970; Pollack, 1986; Balkanski, 1993]. In [Di Eugenio, 1993] I show that purpose clauses do express generation and enablement.
3. The formalism must be either usable by "lower level" processes or be easily interfaced to them. I believe the latter is true both of my KBs, and of the *plan graph*, the output of my interpretation module, that is used as input by other *AnimNL* modules — *AnimNL* is the system my work is embedded in, see below.

To address these issues, I have devised and implemented a formalism composed of two KBs. The first, the *action taxonomy*, stores linguistic knowledge about actions, the second, the *plan library*, contain *action recipes* [Pollack, 1986; Balkanski, 1993], ie common sense planning knowledge about actions. Both are implemented by means of the hybrid system CLASSIC [Brachman *et al.*, 1991]:[2] the terms defined in the action taxonomy are used in the recipes in the plan library.

The framework for the work presented here is the *AnimNL* (*Animation from Natural Language*) project [Webber *et al.*, 1992; Webber *et al.*, 1993]. *AnimNL* has as its goal the automatic creation of animated task simulations from NL instructions. Its agents are animated human figures, engaged in *assembly procedures* and *maintenance procedures.* The underlying animation system, *Jack*^{TM}, provides human figures capable of realistic motion through model-based inverse kinematics [Badler *et al.*, 1993]. Both agents and spatial environments can be modified, so as to vary agents' physical capabilities and the situation in which tasks are carried out.

3 The representation formalism

3.1 The action taxonomy

A hybrid formalism by itself is not sufficient to provide a sound lexical decomposition for verbs, as it lacks a well-defined set of semantic primitives.[3] I found a source for such semantic primitives in

[2] Thus, the distinction between *action taxonomy* and *plan library* is conceptual rather than real.

[3] In the sense of the linguistic notion of semantics.

Conceptual Structures — CSs for short [Jackendoff, 1990]. CSs are well suited to represent action descriptions, as they capture generalizations about actions and their relationships to one another; moreover, CS representations are suitable to express the logical form of an instruction, as they reveal where information may be missing from an utterance and has to be provided by inference.

The advantages of integrating the two formalisms are: using linguistically sound primitives in the T-Box transforms it into a real lexicon; a KL-ONE style representation endows CSs with a hierarchical organization and with the possibility of extending the lexicon.

A CS entity may be of ontological type *Thing, Place, Path, Event, State, Manner* or *Property*. CSs may also contain complex features generated by conceptual functions over other CSs. For instance, the function TO: Place → Path describes a Path that ends in the specified Place:

(4) $[_{\text{Path}}\ \text{TO}([_{\text{Place}}\ \text{IN}([_{\text{Thing}}\ \text{KITCHEN}]_k)]_l)]_m$

Among functions that yield *Events*, we have GO: Thing × Path → Event:

(5)

$$[_{\text{Event}}\ \text{GO}\quad([\text{YOU}]_i,\ [_{\text{Path}}\ \text{TO}([\text{IN}([\text{KITCHEN}]_k)]_l)]_m)]$$

CSs are readily integrated into CLASSIC, as shown in Fig. 1, that presents the portion of T-Box relative to the CS primitive GO — the different kinds of GO's shown vary on the *semantic field*, such as *spatial* or *control*, and on the kind of path. Jackendoff introduces *semantic fields* to capture the similarities between sentences like *Jack went into the kitchen* and *The gift went to Bill*. The semantic field *control* was introduced in [Di Eugenio and White, 1992] to represent the notion of *having control over* some object.

3.2 The plan library

A network representation of *recipe* concepts is shown in Fig. 2. It illustrates the *move-recipe*, which represents a method for (physically) moving an object from one location to another; the same recipe is shown, in perhaps a more readable format, and with more details, in Fig. 3. In Fig. 2, **SAME-AS** constraints, necessary to express coreference, e.g. between the arguments of an annotation and the substeps of the recipe it belongs to, are not shown.

Recipes have a *header*, *body*, *qualifiers* and *effects*. The terminology, especially *header* and *body*, is reminiscent of STRIPS, but the relations between these components are expressed in terms of *enablement* and *generation*, e.g. the body *generates* its header. The representation does not employ preconditions, because it is very difficult to draw the line between what is a precondition and what is part of the body of the action — see [Di Eugenio, 1993; Geib, 1992]. Action recipes express what is traditionally expressed by means of preconditions by means of actions, which are substeps in the body that generates a certain header. Other components of a recipe are the *annotations* on the body, that specify the relations between the substeps, e.g. enablement and temporal relations, the latter derived from [Allen, 1984]; *qualifiers*; and *effects*, what must be true after an action has been executed.

3.3 The role of the hybrid system

I adopted CLASSIC as the underlying backbone of the whole system because I found the *classification* and *realization* inferences that hybrid systems provide both powerful enough and restricted enough to compute the relation between the action described in the input and the stored ones.

Hybrid systems have often been used in the literature to maintain an organized KB of action concepts and plans, to facilitate adding new concepts and to infer of which plan type a certain sequence of events in the world is an instance. For example, in CLASP [Devanbu and Litman, 1991], which is the closest to my work among the systems that exploit subsumption to deal with plans, networks or rules, plans are built out of actions, defined as classic STRIPS planning operators. I go one step further by including linguistic knowledge, totally absent from CLASP, and by exploiting classification also in cases in which the input action description is an instance of a *virtual* concept.

The hybrid system inference engine is used as follows.

First of all, the individual action descriptions contained in the logical form[4] are asserted into the A-Box, whose recognizer computes the set of types of which such descriptions are instances.

Then, the goal β is used as an index into the *plan library*, namely, the CLASSIC query language is used to find those headers of which β is an instance: the corresponding recipes $\mathcal{R}_l$'s are then retrieved.[5]

At this point, for each retrieved recipe $\mathcal{R}_l$, α is checked against each substep $\gamma_{l,j}$ for consistency, namely to understand whether α is less or more specific, or inconsistent with $\gamma_{l,j}$.

In the case of (2), the recipe relating γ = *Turn screw counterclockwise* and δ = *Loosen screw* is re-

[4] The logical form is generated by a Combinatory Categorial Parser [White, 1992].

[5] The match between β and headers is constrained to be *instance-of*; however, see [Di Eugenio, 1993] on possible relaxations of this restriction, in which β is allowed to be less specific than the header of a recipe.

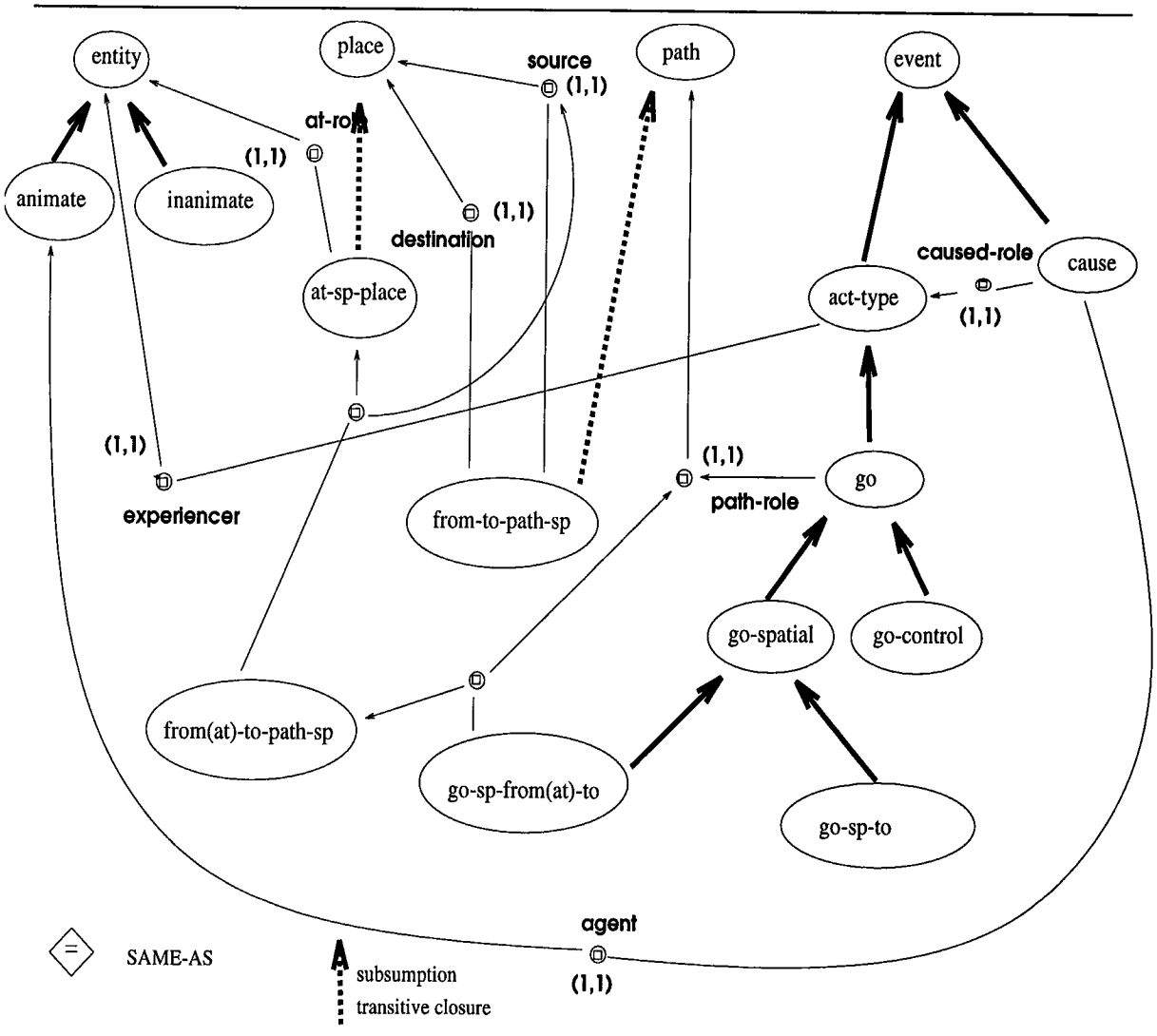

Figure 1: The *go* Hierarchy

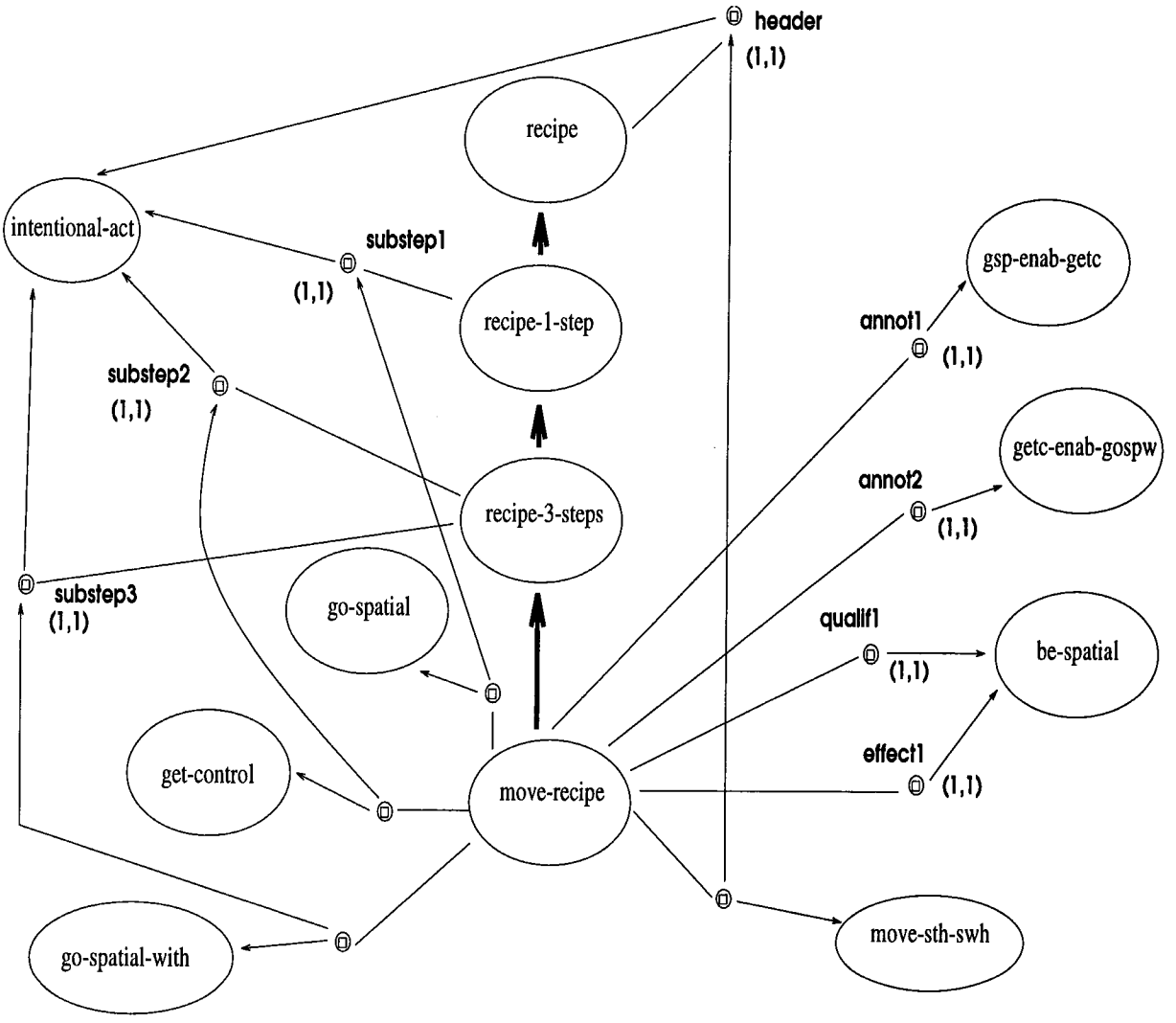

Figure 2: The *Recipe* Hierarchy

Header

$[\text{CAUSE}([\text{AGENT}]_i, [\text{GO}_{\text{Sp}}(j, k)])]$

$$\left[\begin{array}{l}\text{FROM}(m)\\ \text{TO}(l)\end{array}\right]_k$$

Body

- $[\text{GO}_{\text{Sp}}([i, [\text{TO}(m)])]_{\gamma_1}$
- $[\text{CAUSE}(i, [\text{GO}_{\text{Ctrl}}(j, [\text{TO}([\text{AT}(i)])])])]_{\gamma_2}$
- $\left[\begin{array}{l}\text{GO}_{\text{Sp}}(i, k)\\ [\text{WITH}(j)]\end{array}\right]_{\gamma_3}$

Annotations

- γ_1 *enables* γ_2
- γ_2 *enables* γ_3

Qualifiers

- $[\text{BE}_{\text{Sp}}(j, m)]$

Effects

- $[\text{BE}_{\text{Sp}}(j, l)]$

Figure 3: A *Move Something Somewhere* Recipe

trieved; then α = *Turn the screw* is found to subsume γ, thus the action to be performed is taken to be γ. If α were *Turn the screw carefully*, the algorithm would infer that α and γ are mutually consistent, and that the action to be performed is *Turn the screw counterclockwise carefully*; if α were *Turn the screw clockwise*, α and γ would be found to be inconsistent. In this latter case, the instruction *Turn the screw clockwise to loosen* could actually be felicitous, if the screw were of the appropriate, less common type; it is left for future work how to deal with default assumptions.

Let's now turn to the case of computing expectations, which arise from consistency checks that are more complex than those discussed so far.

In the case of (3a), after *move-recipe* has been retrieved through β_1, α_1 = *go(you,to(in(kitchen$_2$)))*[6] and the three substeps $\gamma_1, \gamma_2, \gamma_3$ are checked for consistency. α_1 is found to be an instance of γ_1, with i bound to *you* and m to *in(kitchen$_2$)*.[7]

Then, the algorithm checks whether there is any other information on i or m in `Recipe` that makes i's or m's bindings untenable, such as m = *outdoors*. The only other constraint on these parameters comes from the qualifier, instantiated to *BE(coffee-urn$_3$,m)*. As the binding *in(kitchen$_2$)* for m is consistent with the type restriction *place* placed on m by *BE*,[8] we can conclude *BE(coffee-urn$_3$, in(kitchen$_2$))*, and produce the *plan graph* in Fig. 4.

As regards (3b), the recipe for *wash* doesn't include such a qualifier, thus the expectation about the coffee urn being in the kitchen doesn't arise; however, the weaker expectation *BE(washing-site, in(kitchen$_2$))* is derived as a side effect of the consistency checks.

4 Conclusions

I have presented a formalism in which both linguistic and planning knowledge about actions are represented; the two KBs are unified by using CLASSIC as their common backbone.

On the one hand, I believe that this is the right direction to go, as it is necessary, first, to represent both linguistic and planning knowledge about actions, second, to provide flexibility of match between the logical form and the stored knowledge. Moreover, notice that complex objects — in this case, of type *action* — find a very natural treatment through the concept forming language of the hybrid system.

On the other hand, I am fully aware that I have totally avoided important problems such as quantifiers. Among the questions I would like to explore at the symposium is how far this formalism can go, more specifically as far as representing quantified expressions and negation: negation is very frequent in instructions, to be found, either implicitly or explicitly, in both warnings and (side) effects to be avoided.

[6] This notation is just for readability. α_1 is an individual in CLASSIC.

[7] "*i* bound to *you*" is shorthand for "the role *experiencer* (corresponding to *i*) on the CLASSIC concept γ_1 is filled by the individual *you*".

[8] *BE* is a concept of type *State* in CLASSIC.

References

[Allen, 1984] James Allen. Towards a General Theory of Action and Time. *Artificial Intelligence*, 23:123–154, 1984.

[Badler *et al.*, 1993] Norman I. Badler, Cary B. Phillips, and Bonnie Webber. *Simulating Humans: Computer Graphics Animation and Control.* Oxford University Press, 1993.

[Balkanski, 1993] Cecile Balkanski. *Actions, Beliefs and Intentions in Multi-Action Utterances.* PhD thesis, Harvard University, 1993. Technical Report TR-16-93.

[Brachman *et al.*, 1991] Ronald Brachman, Deborah McGuinness, Peter Patel-Schneider, Lori Alperin Resnick, and Alexander Borgida. Living with CLASSIC: When and How to Use a KL-ONE-like Language. In John F. Sowa, editor, *Principles of Semantic Networks — Explorations in the Representation of Knowledge*, pages 401–456. Morgan Kaufmann, 1991.

[Charniak, 1988] Eugene Charniak. Motivation Analysis, Abductive Unification, and Nonmonotonic Equality. *Artificial Intelligence*, 34:275–295, 1988.

[Devanbu and Litman, 1991] Premkumar Devanbu and Diane Litman. Plan-Based Terminological Reasoning. In *KR91, Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning*, pages 128–138, 1991.

[Di Eugenio and White, 1992] Barbara Di Eugenio and Michael White. On the Interpretation of Natural Language Instructions. In *COLING92, Proceedings of the Fourteenth International Conference on Computational Linguistics*, pages 1147–1151, 1992.

[Di Eugenio, 1993] Barbara Di Eugenio. *Understanding Natural Language Instructions: a Computational Approach to Purpose Clauses.* PhD thesis, University of Pennsylvania, December

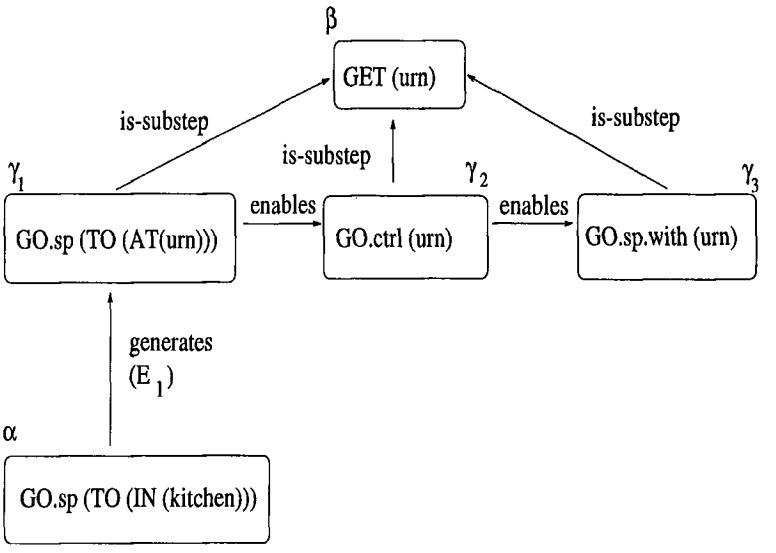

Figure 4: The *plan graph* for *Go into the kitchen to get me the coffee urn*

1993. Technical Report MS-CIS-93-91 (Also Institute for Research in Cognitive Science report IRCS-93-52).

[Di Eugenio, 1994] Barbara Di Eugenio. Action Representation for interpreting Purpose Clauses in Natural Language Instructions. In J. Doyle, E. Sandewall, and P. Torasso, editors, *Principles of Knowledge Representation and Reasoning: Proceedings of the Fourth International Conference (KR94)*. Morgan Kaufmann, San Mateo, CA, 1994.

[Geib, 1992] Christopher Geib. Intentions in Means/End Planning. Technical Report MS-CIS-92-73, University of Pennsylvania, 1992.

[Goldman, 1970] Alvin Goldman. *A Theory of Human Action*. Princeton University Press, 1970.

[Jackendoff, 1990] Ray Jackendoff. *Semantic Structures*. Current Studies in Linguistics Series. The MIT Press, 1990.

[Litman and Allen, 1990] Diane Litman and James Allen. Discourse Processing and Commonsense Plans. In P. Cohen, J. Morgan, and M. Pollack, editors, *Intentions in Communication*. MIT Press, 1990.

[Pollack, 1986] Martha Pollack. *Inferring Domain Plans in Question-Answering*. PhD thesis, University of Pennsylvania, 1986.

[Schoppers, 1988] Marcel Schoppers. *Representation and Automatic Synthesis of Reaction Plans*. PhD thesis, University of Illinois at Urbana-Champaign, 1988.

[Webber *et al.*, 1992] Bonnie Webber, Norman Badler, F. Breckenridge Baldwin, Welton Becket, Barbara Di Eugenio, Christopher Geib, Moon Jung, Libby Levison, Michael Moore, and Michael White. Doing What You're Told: Following Task Instructions In Changing, but Hospitable Environments. Technical Report MS-CIS-92-74, University of Pennsylvania, 1992. To appear in *Language and Vision across the Pacific*, Y. Wilks and N. Okada editors.

[Webber *et al.*, 1993] Bonnie Webber, Norman Badler, Barbara Di Eugenio, Christopher Geib, Libby Levison, and Michael Moore. Instructions, Intentions and Expectations. Technical Report MS-CIS-93-61, University of Pennsylvania, 1993. To appear in *Artificial Intelligence Journal*, Special Issue on Computational Theories of Interaction and Agency.

[White, 1992] Michael White. Conceptual Structures and CCG: Linking Theory and Incorporated Argument Adjuncts. In *COLING92, Proceedings of the Fourteenth International Conference on Computational Linguistics*, pages 246–252, 1992.

[Wilensky, 1983] Robert Wilensky. *Planning and Understanding. A Computational Approach to Human Reasoning.* Addison-Wesley Publishing Company, 1983.