ACCOMMODATING CONTEXT CHANGE

Bonnie Lynn Webber and Breck Baldwin
Department of Computer and Information Science
University of Pennsylvania
Philadelphia, PA 19104-6389
Internet: {bonnie@central,breck@linc}.cis.upenn.edu*

ABSTRACT

Two independent mechanisms of context change have been discussed separately in the literature – context change by *entity introduction* and context change by *event simulation*. Here we discuss their integration. The effectiveness of the integration depends in part on a representation of events that captures people's uncertainty about their outcome – in particular, people's incomplete expectations about the changes effected by events. We propose such a representation and a process of *accommodation* that makes use of it, and discuss our initial implementation of these ideas.

Introduction

Consider the following example:

Example 1

John made a handbag from an inner-tube.
a. He sold *it* for twenty dollars.
b. *He sold *them* for fifty dollars.
c. He had taken *it* from his brother's car.
d. Neither of *them* was particularly useful.

Here two entities are introduced via indefinite noun phrases (NPs) in the first sentence. The alternative follow-ons (a-d) show that subsequent reference to those entities is constrained. In particular, (b) highlights the difference in their existential status, even though there is no *syntactic* difference in how they are introduced. Now consider

Example 2

Mix the flour, butter and water.
a. Knead *the dough* until smooth and shiny.
b. Spread *the paste* over the blueberries.
c. Stir *the batter* until all lumps are gone.

In each of the alternative follow-on (a–c), a different definite NP refers to the result of the mixing, even though the terms "dough", "paste" and "batter" are not interchangeable. (They denote substances with different consistencies, from a pliant solid – dough – to a liquid – batter.)

In both these examples, events[1] are mentioned that change the *world* being described. These examples will be used to show why the two mechanisms of *context* change discussed separately in the literature (context change by *entity introduction* and context change by *event simulation*) must be integrated (Section 2). For such integration to be effective, we argue that it must be based on a representation of events that captures people's uncertainty about their outcome – in particular, people's incomplete expectations about the changes effected by events. An understanding system can then use these expectations to *accommodate* [15] the particular changes that are mentioned in subsequent discourse (Section 3). In Section 4, we discuss our initial implementation of these ideas.

This work is being carried out as part of a project (**AnimNL**) aimed at creating animated task simulations from Natural Language instructions [2; 4; 5; 6; 7; 14; 20]. Instructions are a form of text rich in the specification of events intended to alter the world in some way. Because of this, the issues discussed in this paper are particularly important to both understanding and generating instructions.

*This work was partially supported by ARO grant DAAL 03-89-C-0031, DARPA grant N00014-90-J-1863, and NSF grant IRI 90-16592 to the University of Pennsylvania. The paper draws upon material first presented at the workshop on *Defeasible Reasoning in Semantics and Pragmatics* held at the European Summer School on Logic, Language and Information, Saarbrücken, Germany, August 1991.

[1] *Event* is used informally to mean any kind of action or process.

Mechanisms of Context Change

Computational Linguistics research has recognized two independent mechanisms of context change. The first to have been recognized might be called context change by *entity introduction*. It was first implemented in Woods' question-answering system LUNAR [21; 22]. For each non-anaphoric referential noun phrase (NP) in a question, including a questioned NP itself, LUNAR would create a new constant symbol to represent the new entity, putting an appropriate description on its property list. For example, if asked the question "Which breccias contain molybdenum?", LUNAR would create one new constant to represent molybdenum and another to represent the set of breccias which contain molybdenum. Each new constant would be added to the front of LUNAR's *history list*, thereby making it available as a potential referent for subsequent pronominal and definite NP anaphors (e.g. "Do they also contain titanium?"). Webber [19] further developed this procedure for introducing and characterizing discourse entities available for anaphoric reference

A similar mechanism of context change is embedded in formal dynamic theories of discourse, including Kamp's Discourse Representation Theory [11] and Heim's File Change Semantics [10]. We briefly describe Heim's approach, to show this similarity.

Heim's files constitute an intermediate level of representation between the sentences of a text and the model which gives them their truth values. A sentence can be viewed as denoting a function from an input file to an output file. Each *indefinite* NP in a sentence requires a new *file card* in the output file which does not appear in the input file, on which is inscribed the properties of the new entity. Each *definite* NP must either map to an existing file card or have a semantic association with an existing card, allowing it to be *accommodated* into the discourse. In the latter case, a new file card is inserted in the input file which the definite NP is now taken as mapping to. Context change therefore consists of new annotations to existing cards and new cards added for indefinite NPs and accommodated definite NPs. The files do not change in any other way that reflects events described in the text.

Formal theories of discourse have been broadened to allow for types of "embedded contexts" associated with modals [17] and with propositional attitudes [1]. Although they have also begun to deal with problems of tense and the temporal relationship of events described in a text [12; 16], there is still no connection between the events described in a text and the individuals introduced therein.

Context change by *event simulation* is a feature of Dale's recent Natural Language generation system EPICURE [3], which generates recipe texts from an underlying plan representation. In EPICURE, the individuals available for reference change in step with the events described in the text.[2] In a sense, EPICURE is *simulating* the effects of the events that the text describes.

In implementing this, Dale represents actions with STRIPS-like operators which can change the world from one state to another. Each object and state in EPICURE has a unique index, with the set of objects available in a given state constituting its *working set*. With respect to objects[3], an action can have two types of effects: it can change a property of an object (e.g., from being an individual carrot to being a mass of grated carrot), or it can add an object to or remove it from the world, as represented in the current working set (e.g., flour disappears as an independent entity when combined with water, and dough appears). The preconditions and postconditions of each action indicate the objects required in the working set for its performance and the changes it makes to objects in the working set as a result. For example, ADD (in the sense of "add X to Y") has as preconditions that X and Y be in the current working set and as post-conditions, that X and Y are absent from the resulting working set and a new object Z is present whose constituents are X and Y.

The form of recipe that EPICURE generates is the common one in which a list of ingredients is followed by instructions as to what to do with them. Thus all entities are introduced to the reader in this initial list (e.g., "four ounces of butter beans", "a large onion", "some sea salt", etc.) before any mention of the events that will (deterministically) change their properties or their existential status. As a result, in the *text* of the recipe, EPICURE only embodies context change by event simulation: no new entities are introduced in the text that are not already known from the list of ingredients.

[2] In earlier work, Grosz [8] noticed that in task-oriented dialogues, the performance of actions could alter what objects the speakers would take to be in *focus* and hence take as the intended referents of definite pronouns and NPs. However, actual changes in the properties and existential status of objects due to actions were not part of Grosz' study.

[3] Dale construes and also implements the notion of object very broadly, so that the term applies equally well to a two-pound package of parsnips and a tablespoon of salt

Our work on integrating these two mechanisms of context change involves dropping Dale's assumption that states are complete specifications of an underlying model. (To emphasize that descriptions are partial, we will use the term *situation* rather than *state*.) As in EPICURE, actions are represented here by operators – functions from one situation to another. The meaning of a clause is given in terms of these operators.[4] Also as in EPICURE, the term *working set* is used for the set of entities in the discourse context. For clarity, we refer to the working set associated with the situation prior to the described event as the WS_i, and the working set associated with the situation after it as the WS_o. An indefinite NP in the clause may *introduce* an entity into the WS_i. Alternatively, it may denote an entity in the WS_o that corresponds to a result of the event being described. Whether an entity introduced into WS_i *persists* into WS_o will depend on the particular event. This is characterized as in EPICURE by preconditions on WS_i and postconditions on WS_o, plus a default assumption, that if an action is not known to affect an object and the text does not indicate that the object has been affected, then one assumes it has not been.

For example, consider an operator corresponding to MAKE X FROM Y (in the sense used in Example 1). Its precondition is that X is in WS_i. Its postconditions are that X is not in WS_o, Y is in WS_o, and mainConstituentOf(Y,X). In response to the sentence "John made a handbag from an inner-tube" (or alternatively, "John made an inner-tube into a handbag"), a new entity (x_1) corresponding to inner-tube would be introduced into the current WS_i. The situation resulting from the MAKE action contains a new entity (x_2) corresponding to its product, which is what "a handbag" is taken to denote. The postconditions on MAKE specify that x_1 does not persist into WS_o as a separate object.[5]

Now consider the alternative follow-ons to Example 1. The sentence

He sold it for $20.

describes a subsequent event. Its WS_i is the WS_o of the previous utterance, augmented by an entity introduced by the NP $20. Entities introduced into WS_i that persist through to WS_o continue to be available for reference in clauses describing subsequent events, as illustrated by the subsequent reference to John ("he") above.

The alternative follow-on

He had taken it from his brother's car.

describes the situation prior to the previous event. Its WS_i is the WS_i of the previous event, augmented by entities corresponding to "his brother" and "his brother's car. The only way to refer anaphorically to entities from different working sets is with a follow-on that refers *atemporally* across situations (e.g. "Neither of them was particularly useful).

To date, we have not found any *individual* event descriptions whose semantics requires specifying more than the situations prior to and following the event. This is not to say that events cannot be described in terms of a sequence of situations (e.g. "John began to mix the flour, butter and water. He mixed them for 5 minutes. He finished mixing them."). The point is that the semantics of a *single* event description appears to require no more than specifying properties of WS_i and WS_o.

Before discussing Example 2 in detail in the next section, we would like to draw the reader's attention to two variations of that example:

Example 3

a. Mix the flour and butter into *a dough*.
b. Mix the nuts and butter into *the dough*.

What is of interest is the different roles that the prepositional phrase plays in these two cases and how they are disambiguated. In 3a, "into a dough" specifies the *goal* of the mixing. An operator representing this sense of MIX X INTO Y would, like the operator for MAKE Y FROM X above, have as its precondition that X is in WS_i. Its post-conditions are that Y is in WS_o and that constituentsOf(Y,X). In response to 3a, the definite NP "the flour and butter" would have to be resolved against entities already in WS_i, while "a dough" would be taken to denote the new entity entered into WS_o, corresponding to the product of the mixing.

In 3b however, "into the dough" specifies the *destination* of the ingredients, with mixing having this additional sense of translational motion. An operator representing this sense of MIX X INTO Y would have as its precondition that *both* X and Y are in WS_i. Its post-conditions are that Y is in WS_o and that X is added to the set of constituents of Y. In

[4] We are ignoring a clause's aspectual character here – that it may not imply the completion of the denoted action. What is offered here are necessary but not sufficient features of a solution.

[5] Non-destructive constructive actions such as "build", "assemble", etc. (e.g. "build a house of Lego blocks") do not have this property: constituent entities retain their individual existence.

response to 3b, not only would the definite NP "the nuts and butter" have to be resolved against entities already in WS_i, but "the dough" would have to be so resolved as well.

With a definite NP in a MIX INTO prepositional phrase, disambiguating between these two senses is simple: it can only be the latter sense, because of the precondition that its referent already be in WS_i. With an indefinite NP however, it can only be a matter of preference for the first sense.

Expectation and Accommodation

For the integration proposed above to effectively handle Example 4 below (Example 2 from the Introduction) and Example 5, one needs both a more accurate representation of people's beliefs about events and a way of dealing with those beliefs.

Example 4

Mix the flour, butter and water.
a. Knead *the dough* until smooth and shiny.
b. Spread *the paste* over the blueberries.
c. Stir *the batter* until all lumps are gone.

Example 5

John carved his father a chair for his birthday.
a. *The wood* came from Madagascar.
b. *The marble* came from Vermont.

If the definite NPs in examples 4 and 5 are taken as definite by virtue of their association with the previously mentioned event (just as definites have long been noted as being felicitous by virtue of their association with previously mentioned objects), then Example 4 shows people associating a variety of different *results* with the same action and Example 5, a variety of different *inputs*. To deal with this, we argue for

1. characterizing an agent's knowledge of an action in terms of *partial constraints* on its WS_i and *partial expectations* about its WS_o;
2. accommodating [15] definite NPs in subsequent utterances as instantiating either a partial constraint in WS_i or a partial expectation in WS_o.

There appear to be three ways in which an agent's knowledge of an action's constraints and expectations may be partial, each of which manifests itself somewhat differently in discourse: the knowledge may be *abstract*, it may be *disjunctive*, or it may involve *options* that may or may not be realized.

Abstract Knowledge. An agent may believe that an action has a predictable result, without being able to give its particulars. For example, an agent may know that when she adds white paint to any other color paint, she gets paint of a lighter color. Its *particular color* will depend on the color of the original paint and the amount of white she adds. In such cases, one might want to characterize the agent's partial beliefs as abstract descriptions. The agent may then bring those beliefs to bear in generating or understanding text describing events. That is, in both narrative and instructions, the speaker is taken to know more about what has happened (or should happen) than the listener. The listener may thus not be able immediately to form *specific* expectations about the results of described events. But she can accommodate [15] a definite NP that can be taken to denote an instantiation of those expectations.

In Example 4, for example, one might characterize the agent's expectation about the object resulting from a blending or mixing action abstractly as a *mixture*. Given an instruction to mix or blend something, the agent can then accommodate a subsequent definite reference to a particular kind of mixture – a *batter*, a *paste* or a *dough* – as instantiating this expectation.

An agent's knowledge of the input constraints on an action may be similarly abstract, characterizing, for example, the input to "carve" as a unit of solid material. Having been told about a particular carving action, a listener can understand reference to a unit of particular material (stone, wood, ice, etc.) as instantiating this input object.

Disjunctive Knowledge. An experienced agent has, for example, alternative expectations about the result of beating oil into egg yolks: the resulting object will be either an emulsion (i.e., mayonnaise) or a curdled mass of egg yolk globules floating in oil. Most often, one of the disjuncts will correspond to the intended result of the action, although "intended" does not necessarily imply "likely". (The result may in fact be quite *unpredictable*.) In a text, the disjunctive knowledge that an agent has, or is meant to have, about actions is manifest in the descriptions given of all (or several) alternatives. Often, the unintended alternatives are presented in a conditional mood.

Options. A third type of partial knowledge that an agent may have about an action is that it may or may not produce a particular, usually secondary, result, depending on circumstances. As with disjunctive expectations, these results are unpredictable. A com-

mon way to specify options such as these in recipes is with the "if any" construction, as in

Example 6

Saute garlic until lightly browned. Remove the burnt bits, if any, before continuing.

Our work to date has focussed on modelling an agent's abstract knowledge of actions and how it can be used in updating context and accommodating subsequent referring expressions, as in Examples 4 and 5.[6] These abstract constraints and expectations can be applied immediately as a clause describing their associated action is processed. Context changes will then reflect explicit lexical material, when present, as in

Mix the flour, butter and water into a paste.

or simply the agent's (abstract) expectations, when explicit lexical material is not present, as in

Mix the flour, butter and water.

In the latter case, a subsequent definite NP denoting a particular kind of mixture (the solution, the paste, etc) can be taken as referring to an entity that is in the current working set, merely refining its description, as in Example 4 above.

Initial Implementation

Entity Introduction and Elimination

The Natural Language and reasoning components of the AnimNL project are being implemented in Prolog. In our initial implementation of context change, entities can be *entered* into the context by either entity introduction or event simulation, but they are never actually *removed*. Instead, actions are treated as changing the properties of entities, which may make them inaccessible to subsequent actions. For example, mixing flour, butter and water (Examples 3a and 4) is understood as changing the properties of the three ingredients, so that they are no longer subject to independent manipulation. (Here we are following Hayes' treatment of "liquid pieces" [9] which holds, for example, that the piece of water that was in a container still "exists" even after being poured into a lake: It is just no longer independently accessible.) This approach seems to simplify reference resolution decisions, but we are not rigidly committed to it.

The mechanism for changing properties and introducing entities uses STRIPS-like operators such as

```
mix(E,X,Y)
  precond: [manipulable(X)]
  delete:  [manipulable(X)]
  postcond: [mixture(Y) & manipulable(Y)
      & constituentsOf(Y,X)]
```

which would be instantiated in the case of mixing flour, butter and water to

```
mix(e1,{f,w,b},m) & flour(f) & water(w)
      & butter(b) & definite({f,w,b})
  precond: [manipulable({f,w,b})]
  delete: [manipulable({f,w,b})]
  postcond: [mixture(m) & manipulable(m)
      & constituentsOf(m,{f,w,b})]
```

The predicate in the header `definite({f,w,b})` is an instruction to the back chainer that unique antecedents need to be found for each member of the set. (In recipes, the antecedents may be provided through either the previous discourse or the ingredients list.) If `definite` is absent, as in the case of interpreting "mix some flour, water and butter", the back chainer introduces new entities into the working set. It also inserts into the working set a new entity corresponding to the postcondition `mixture(m)`, whether this entity has a lexical realization (as in Example 3a) or not (as in Example 4).

Abstract Knowledge of Actions

The `mix` operator shown above introduces a new entity in the WS_o `mixture(m)` which is the the result of successful mixing. The definite NP in Example 4a "the dough" both takes `m` as an antecedent and provides more information about `m`'s make-up – that it is dough. The definite reference resolution algorithm applies the knowledge that the existence of a mixture in the discourse is consistent with that mixture being dough, and the discourse is updated with `dough(m)`. The application of unsound inference, in this case that the mixture is dough (or in 4b, paste, or in 4c, batter) is supported in a backchaining environment via the following axioms:

```
[mixture(X)] ==> [dough(X)]
[mixture(X)] ==> [paste(X)]
[mixture(X)] ==> [batter(X)]
```

[6] Tenenberg has used an abstraction hierarchy of action descriptions to simplify the task of planning [18], and Kautz, to simplify plan inference [13]. This same knowledge can be applied to language processing.

This axiomatization is problematic in not preventing the back chainer from proving that the mixture which was subsequently referred to as dough, is also a batter. That is, there is no mechanism which treats the axioms as being mutually exclusive. This is handled by a consistency checker which takes every new assertation to the discourse model, and determines that it is consistent with all 1-place relations that hold of the entity.

Disjunctive Knowledge about Actions

The various forms of partial specification of actions can be represented as explicit disjunction in an action knowledge base.[7] For example, **mix** has several operator realizations that reflect the action's completion and its success. The first category of (un)successfully (in)completed actions is represented by an event modifier which determines which action description is pulled from the action KB. In the case of mixing, successfully completed actions are represented more fully as:

```
mix(E,X,M) & complete(E) & successful(E)
   precond: [manipulable(X)]
   delete: [manipulable(X)]
   postcond: [mixture(M) & manipulable(M)
       & constituentsOf(M,X)]
```

This is the same basic representation as before, except with the 'to be mixed' entities unspecified, and the event modifiers added.

Agents differ in their expectations about incomplete mixing action. The following entry has the same preconditions and delete list as above, but the post-condition differs in that there is no mixture introduced to the discourse.

```
mix(E,X) & incomplete(E)
   precond: [manipulable(X)]
   delete: [manipulable(X)]
   postcond: □
```

A different agent could have a different characterization of incomplete mixings – for example, a postcondition introducing an entity describable as **mess(m)**, or **incomplete_mixture(m)**. The point is that degree of completion does effect the introduction of new entities into the discourse model. One can envision other event modifiers that change the impact of an action on the WS_o, either with properties of entities changing or individuals being introduced or not.

[7] An abstraction hierarchy has not yet been constructed.

The next class of disjunctive action descriptions are those that introduce contingencies that are not naturally handled by event modifiers as above. Consider the following representations of two different outcomes of sauteing garlic:

```
saute(E,Y,X) & complete(E)
   precond: [sauteable(Y)]
   delete: □
   postcond: [sauteed(Y) & burnt_bits(X)]

saute(E,Y) & complete(E)
   precond: [sauteable(Y)]
   delete: □
   postcond: [sauteed(Y)]
```

The only difference in the entries is that one introduces burnt bits and the other does not. Ideally, one would like to combine these representations under a single, more abstract entry, such as proposed in [18]. Even with appropriate abstract operators though, the fact that we are modelling discourse introduces a further complication. That is, instructions may address several contingencies in the discourse, so the issue is not that one must be chosen for the discourse, but any number may be mentioned, for example

Example 7

> Dribble 1/2 c. oil into the egg yolks, beating steadily. If you do this carefully, the result will be mayonnaise. If it curdles, start again.

This is a substantial challenge to representing the meaning of instructions in the discourse model because (as above) the various outcomes of an action may be mutually exclusive. Here, successful completion of the action introduces 'mayonnaise(m)' into the discourse model, while unsuccessful completion introduces 'curdled_mess(m)'.

One possible solution is to partition the discourse model into different contexts, corresponding to different outcomes. This too has been left for future exploration.

Conclusion

We hope to have shown that is is both necessary and possible to integrate the two types of context change mechanisms previously discussed in the literature. The proposed integration requires sensitivity to both syntactic/semantic features of Natural Language text (such as definiteness, tense, mood,etc) and to the same beliefs about actions that an agent uses in planning and plan inference. As such, one has some hope that as we become more able to endow Natural Language systems with abilities to plan and recognize the plans of others, we will also be able to endow them with greater language processing capabilities as well.

References

[1] Asher, N. A Typology for Attitude Verbs and their Anaphoric Properties. *Linguistics and Philosophy* 10(2), May 1987, pp. 125-198.

[2] Norman Badler, Bonnie Webber, Jeff Esakov and Jugal Kalita. Animation from Instructions. *Making Them Move: Mechanics, Control and Animation of Articulated Figures.* Morgan-Kaufmann, 1990.

[3] Dale, R. *Generating Referring Expressions: Constructing Descriptions in a Domain of Objects and Processes.* PhD Thesis, University of Edinburgh, 1989. (Cambridge MA: MIT Press, forthcoming).

[4] Di Eugenio, B. Action Representation for Natural Language Instructions. *Proc. 1991 Annual Meeting of the Assoc. for Computational Linguistics*, Berkeley CA, June 1991, pp. 333-334.

[5] Di Eugenio, B. Understanding Natural Language Instructions: The Case of Purpose Clauses. *Proc. 1992 Annual Meeting of the Assoc. for Computational Linguistics*, Newark DL, July 1992.

[6] Di Eugenio, B. and Webber, B. Plan Recognition in Understanding Instructions. *Proc. First Int'l Conf. on AI Planning Systems*, College Park MD, June 1992.

[7] Di Eugenio, B. and White, M. On the Interpretation of Natural Language Instructions. *Proc. 1992 Int. Conf. on Computational Linguistics* (COLING-92), Nantes, France, July 1992.

[8] Grosz, B. *The Representation and Use of Focus in Dialogue Understanding.* Technical Note 151, Artificial Intelligence Center, SRI International, 1977.

[9] Hayes, Patrick. Naive Physics I: Ontology for Liquids. Reprinted in J. Hobbs and R. Moore (eds.), *Formal Theories of the Commonsense World.* Norwood NJ: ABLEX Publishing, 1985.

[10] Heim, I. *The Semantics of Definite and Indefinite Noun Phrases.* PhD dissertation, University of Massachusetts, Amherst MA, 1982.

[11] Kamp, H. A Theory of Truth and Semantic Representation. In J. Groenendijk, T. Janssen and M. Stokhof (eds.), *Truth, Interpretation and Information*, Dordrecht: Foris, 1981, pp. 1-41.

[12] Kamp, H. and Rohrer, C. *manuscript of book on temporal reference.* To appear.

[13] Kautz, H. A Circumscriptive Theory of Plan Recognition. In J. Morgan, P. Cohen and M. Pollack (eds.), *Intentions in Communication.* Cambrdige MA: MIT Press, 1990.

[14] Levison, L. Action Composition for the Animation of Natural Language Instructions. Dept of Computer & Information Science, Univ. of Pennsylvania, Technical Report MS-CIS-91-28, September 1991.

[15] Lewis, D. Scorekeeping in a Language Game. *J. Philosophical Logic* 8, 1979, pp. 339-359.

[16] *Linguistics and Philosophy* 9(1), February 1986. Special issue on Tense and Aspect in Discourse.

[17] Roberts, C. Modal Subordination and Pronominal Anaphora in Discourse. *Linguistics and Philosophy* 12(6), December 1989, pp. 683-721.

[18] Tenenberg, J. Inheritance in Automated Planning. *Proc. Principles of Knowledge Representation and Reasoning* (KR'89), Morgan Kaufmann, 1989, pp. 475-485.

[19] Webber, B. *A Formal Approach to Discourse Anaphora.* Technical Report 3761, Bolt Beranek and Newman, Cambridge MA, 1978. (Published by Garland Press, New York, 1979.)

[20] Webber, B., Badler, N., Di Eugenio, B., Levison, L. and White, M. Instructing Animated Agents. *Proc. First US–Japan Workshop on Integrated Systems in Multi-Media Environments*, Las Cruces NM, December 1991.

[21] Woods, W., Kaplan, R. and Nash-Webber, B. The Lunar Sciences Natural Language Information System: Final Report. Technical Report 2378, Bolt Beranek and Newman, Cambridge MA, 1972.

[22] Woods, W. Semantics and Quantification in Natural Language Question Answering. *Advances in Computers*, Volume 17, Academic Press, 1978.