A Dictionary of SNePS Case Frames

Stuart C. Shapiro, William J. Rapaport,
Sung-Hye Cho, Joongmin Choi, Elissa Feit, Susan Haller,
Jason Kankiewicz, and Deepak Kumar

Department of Computer Science
State University of New York at Buffalo
Buffalo, New York 14260

DRAFT

May 31, 1996

Contents

1 Introduction **3**

2 "Standard" Case Frames **4**
- 2.1 LEX . . . 5
- 2.2 OBJECT/PROPERNAME . . . 8
- 2.3 MEMBER/CLASS . . . 10
- 2.4 SUBCLASS/SUPERCLASS . . . 13
- 2.5 OBJECT/PROPERTY . . . 15
- 2.6 REL/OBJECT1/OBJECTN . . . 17
- 2.7 AGENT/ACT . . . 20

3 Case Frames for Rules **22**
- 3.1 MIN/MAX/ARG . . . 23
- 3.2 THRESH/ARG . . . 26
- 3.3 THRESH/THRESHMAX/ARG . . . 29
- 3.4 THRESH/&ANT/CQ (Numerical Entailment) . . . 32
- 3.5 EMIN/EMAX/ETOT/PEVB/&ANT/CQ (Numerical Quantification) . . . 35

4 Case Frames for Plans and Acts **38**
- 4.1 Achieve Act . . . 39
- 4.2 Acts . . . 41
- 4.3 Conditional Act . . . 43
- 4.4 Do-All Act . . . 46
- 4.5 Do-One Act . . . 49
- 4.6 Effects . . . 52
- 4.7 Iteration Act . . . 55
- 4.8 Plan-Act Proposition . . . 58
- 4.9 Plan-Goal Proposition . . . 61
- 4.10 Preconditions . . . 64
- 4.11 Sequence Acts . . . 66

5 Case Frames for Sentences **69**
- 5.1 LEXI-CAT . . . 70
- 5.2 EXPRESSED-EXPRESSION . . . 72
- 5.3 PRED-WORD . . . 74
- 5.4 BEG-END . . . 76
- 5.5 BSTR-CAT-STRC . . . 78

5.6 CONSTIT-CONSTITOF 81

6 Case Frames from Albert H. Yuhan 84

6.1 Group 85
6.2 Part-Whole 89
6.3 Kinship 93
6.4 Purpose Relation Between Events 96
6.5 Events 100
6.6 Spatial Event Affair - Motion 104
6.7 Spatial Event Affair - Station 108
6.8 Spatial Event Affair - Orientation 112
6.9 Spatial Proximity 116
6.10 Spatial Proximity with a Directional Reference 123
6.11 Resolved Spatial Reference Frame 132
6.12 Event Time Relation Between Conflated Spatial Event Affairs 142

Bibliography 147

Chapter 1

Introduction

This document is intended to help new users of the SNePS Semantic Network Processing System by presenting some of the case-frames that have been found useful in the past.

Chapter 2

“Standard” Case Frames

2.1 LEX

Syntax

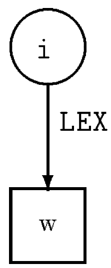

Figure 2.1:

If *w* is a lexeme and `i` is an identifier not previously used, then Figure 2.1 is a network, `w` is a sensory node, and `i` is a structured individual node.

Semantics

⟦i⟧ is the Meinongian entity expressed by uttering *w*.

Sample Context

SNePSUL Interaction

Lucy pets a dog can be represented by the following commands.

```
* (DESCRIBE (ASSERT PROPERNAME (BUILD LEX ‘‘Lucy’’)
                    OBJECT #O))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT AGENT *O
                    ACT (BUILD ACTION (BUILD LEX "pet")
                               OBJECT #O2)))

(M5! (ACT (M4 (ACTION (M3 (LEX pet)))
              (OBJECT B2)))
     (AGENT B1))

* (DESCRIBE (ASSERT MEMBER *O2 CLASS (BUILD LEX "dog")))

(M7! (CLASS (M6 (LEX dog)))
     (MEMBER B2))
```

Network

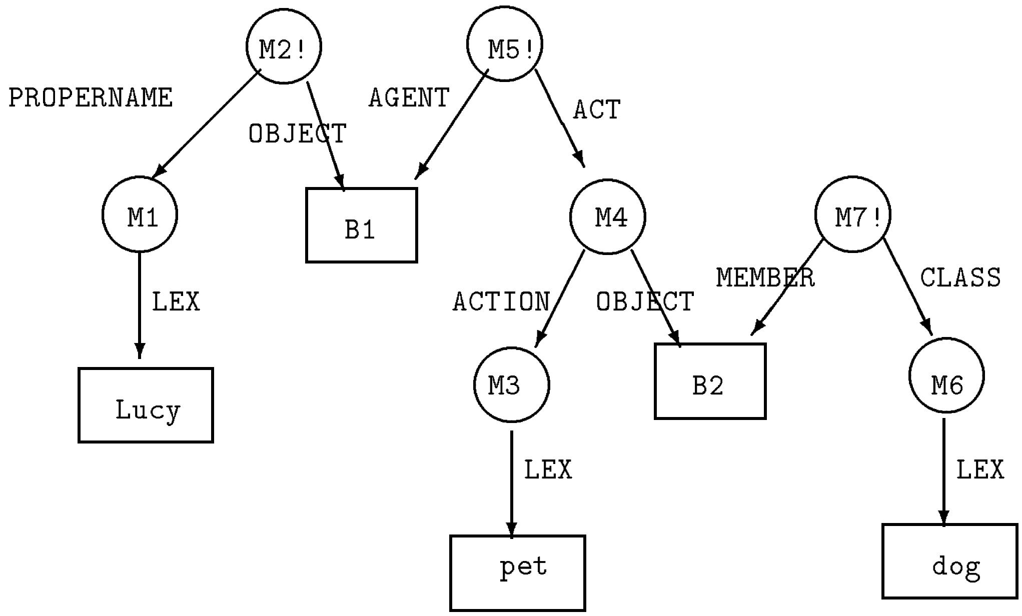

Figure 2.2:

Pictorially, this is shown in Figure 2.2. In this figure, the node `M1` represents the entity expressed as "Lucy," node `M3` represents the entity expressed as "pet," and node `M6` represents the entity expressed as "dog."

Alternatives

EXPRESSED/EXPRESSION is an alternative that allows for homonyms.

Issues

Allowing for sensory nodes other than lexemes is a subject of current research.

Related Entries

EXPRESSED-EXPRESSION.

References

See [26] for the description of the syntax and semantics for this representation. Examples are found in [12, 13, 14, 16, 20, 22].

Author

Stuart C. Shapiro, June 21, 1993

2.2 OBJECT/PROPERNAME

Syntax

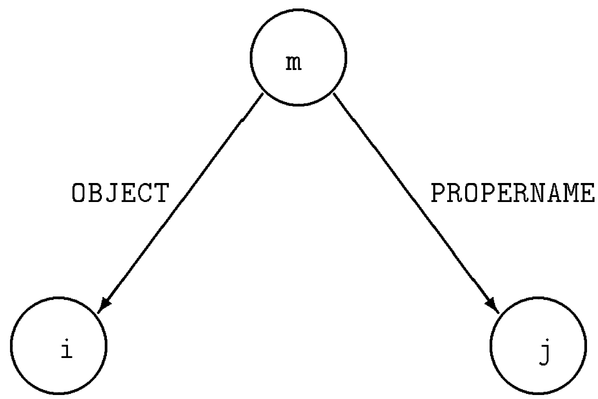

Figure 2.3:

If `i` and `j` are individual nodes and 'm' is an identifier not previously used, then Figure 2.3 is a network and `m` is a structured proposition node.

Semantics

⟦`m`⟧ is the proposition that ⟦`i`⟧ has the proper name ⟦`j`⟧.

Sample Context

Lucy is a girl can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT OBJECT #O
                    PROPERNAME (BUILD LEX "Lucy")))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT MEMBER *O
                    CLASS (BUILD LEX "girl")))

(M4! (CLASS (M3 (LEX girl)))
     (MEMBER B1))
```

Network

Pictorially, this is shown in Figure 2.4. In this figure, the node `M2!` represents the proposition that `M1` is the proper name of `B1`.

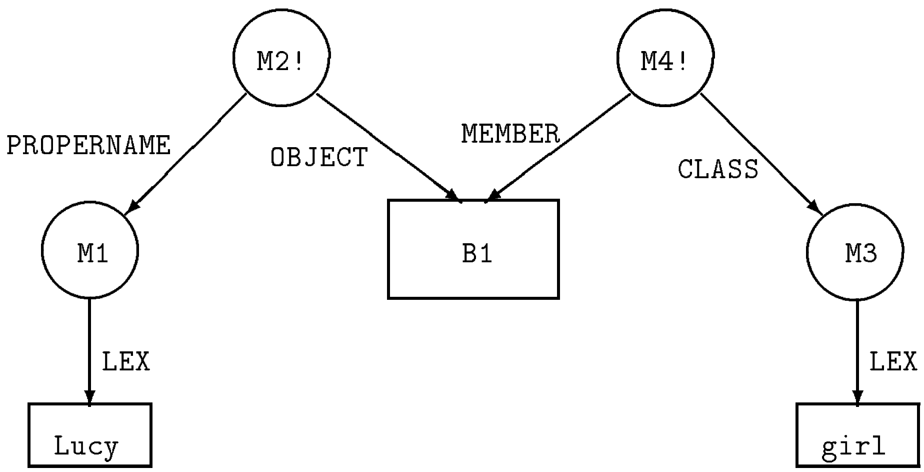

Figure 2.4: Lucy is a girl

Alternatives

None.

Issues

The importance of having a node for a proper name separate from the node for an individual is discussed in [16] and [15].

Related Entries

None.

References

See [26] for the description of the syntax and semantics for this representation. Examples are found in [22, 16, 18].

Author

Jason Kankiewicz, October 1993

2.3 MEMBER/CLASS

Syntax

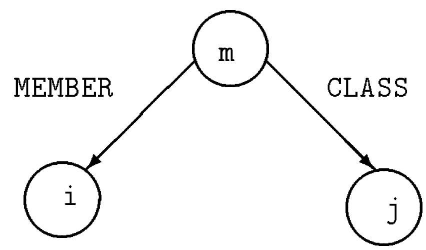

Figure 2.5:

If i and j are individual nodes and m is an identifier not previously used, then Figure 2.5 is a network and m is a structured proposition node.

Semantics

⟦m⟧ is the proposition that ⟦i⟧ is a (member of class) ⟦j⟧.

Sample Context

SNePSUL Interaction

Lucy pets a dog can be represented by the following commands.

```
* (DESCRIBE (ASSERT PROPERNAME (BUILD LEX ``Lucy'')
                    OBJECT #O))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT AGENT *O
                    ACT (BUILD ACTION (BUILD LEX "pet")
                               OBJECT #O2)))

(M5! (ACT (M4 (ACTION (M3 (LEX pet)))
              (OBJECT B2)))
     (AGENT B1))

* (DESCRIBE (ASSERT MEMBER *O2 CLASS (BUILD LEX "dog")))

(M7! (CLASS (M6 (LEX dog)))
     (MEMBER B2))
```

Network

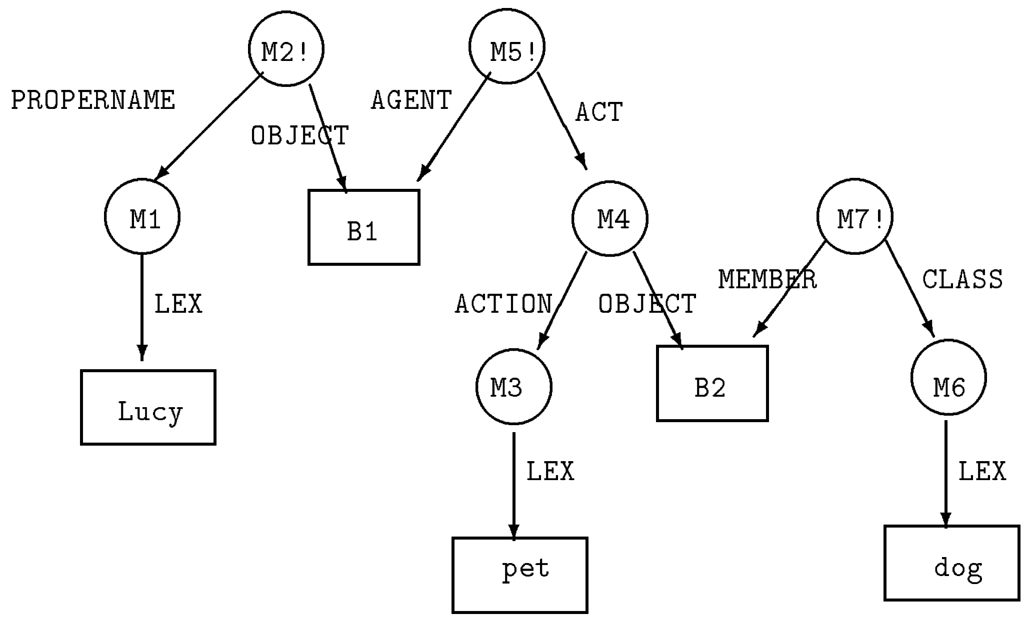

Figure 2.6:

Pictorially, this is shown in Figure 2.6. In this figure, the node `M7!` represents the proposition that `B2` is a dog.

Alternatives

MEMBER/CLASS case frame is used for basic level categories such as chairs, dogs, and cars. For non-basic categories such as furniture, vehicle, and mammal, use ARG1/REL/ARG2 case frame.

Issues

See [12, 14] for details about basic and non-basic categories.

Related Entries

None.

References

See [26] for the description of the syntax and semantics for this representation. Examples are found in [12, 13, 14, 16, 20, 22].

Author

Joongmin Choi, August 12, 1991

2.4 SUBCLASS/SUPERCLASS

Syntax

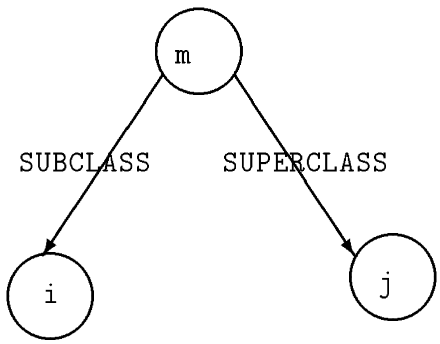

Figure 2.7:

If `i` and `j` are individual nodes and '`m`' is not previously used, then Figure 2.7 is a network and `m` is a structured proposition node.

Semantics

⟦`m`⟧ is the proposition that the class of ⟦`i`⟧s is a subclass of the class of ⟦`j`⟧s.

Sample Context

SNePSUL Interaction

```
*(DESCRIBE (ASSERT SUBCLASS (BUILD LEX "dog")
                   SUPERCLASS (BUILD LEX "animal")))

(M3! (SUBCLASS (M1 (LEX dog))) (SUPERCLASS (M2 (LEX animal))))
```

Network

In Figure 2.8, ⟦`M3!`⟧ is the proposition that dogs are animals.

Alternatives

None.

Issues

The `SUBCLASS/SUPERCLASS` case frame is used for generalization hierarchies.

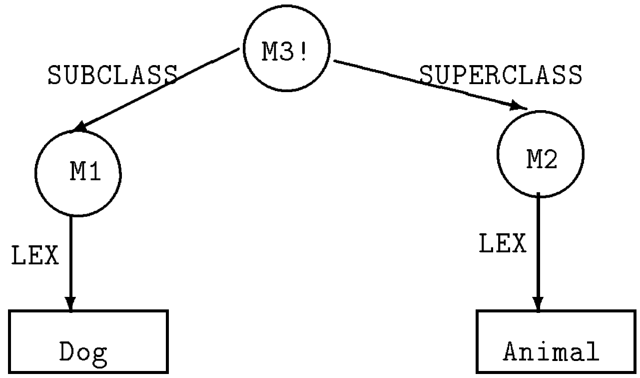

Figure 2.8: Dogs are animals

Related Entries

See also `MEMBER/CLASS` case frame.

References

See [20, 26] for a description of the syntax and the semantics for the case frame. See also [12, 13, 14] for its usage in generalization hierarchies.

Author

Sung-Hye Cho, August 6, 1991

2.5 OBJECT/PROPERTY

Syntax

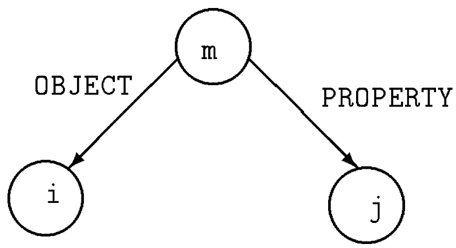

Figure 2.9:

If `i` and `j` are individual nodes and `m` is an identifier not previously used, then Figure 2.9 is a network and *m* is a structured proposition node.

Semantics

⟦m⟧ is the proposition that ⟦i⟧ has the property ⟦j⟧.

Sample Context

SNePSUL Interaction

Lucy is blond can be represented by the following commands.

```
* (DESCRIBE (ASSERT PROPERNAME (BUILD LEX ''Lucy'')
                    OBJECT #O))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT OBJECT *O PROPERTY (BUILD LEX ''blond'')))

(M4! (OBJECT B1)
     (PROPERTY (M3 (LEX blond))))
```

Network

Pictorially, this is shown in Figure 2.10.

Alternatives

None.

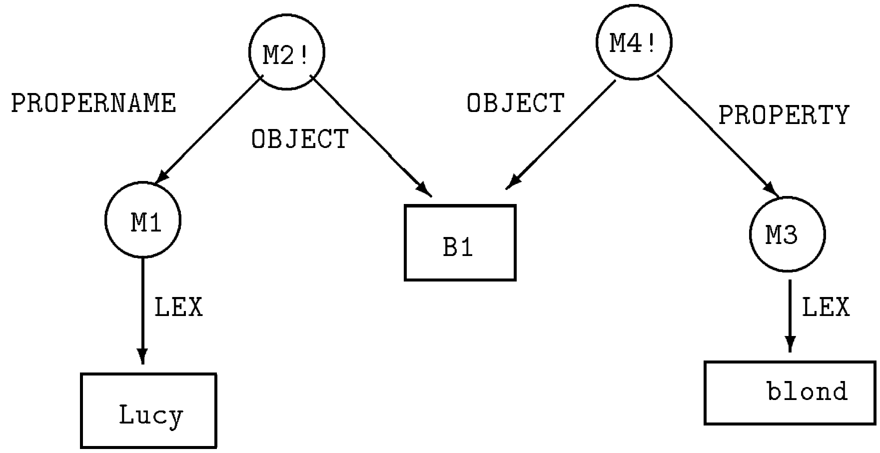

Figure 2.10:

Issues

OBJECT/PROPERTY case frame is used when time bounds are ignored. If time bounds are significant, use OBJECT/PROPERTY/EVENT case frame that is mentioned in Stative propositions section.

Related Entries

Stative propositions.

References

See [26] for the description of the syntax and semantics for this representation. Examples are found in [12, 16, 30].

Author

Joongmin Choi, August 12, 1991

2.6 REL/OBJECT1/OBJECTN

Syntax

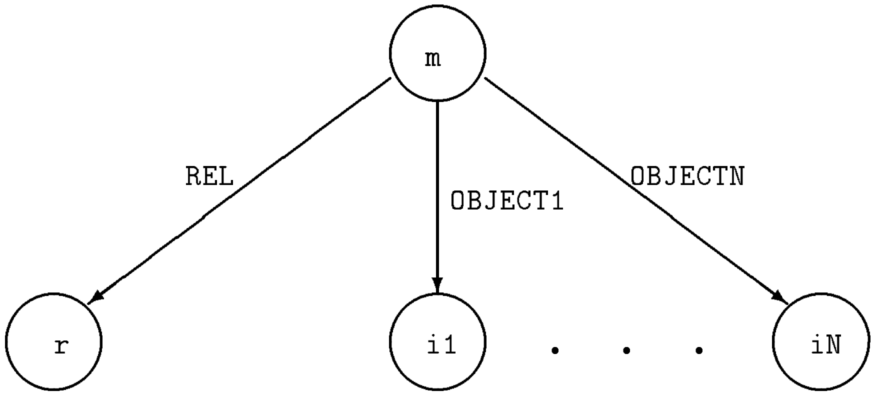

Figure 2.11:

If i1, ..., iN and r are individual nodes and 'm' is an identifier not previously used then Figure 2.11 is a network and m is structured proposition node.

Semantics

⟦m⟧ is the proposition that ⟦i1⟧, ..., ⟦iN⟧ stand in the relation ⟦r⟧.

Sample Context

B is between A and C can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT OBJECT1 #O1
            PROPERNAME (BUILD LEX "A")))

(M2! (OBJECT1 B1)
     (PROPERNAME (M1 (LEX A))))

* (DESCRIBE (ASSERT OBJECT2 #O2
                    PROPERNAME (BUILD LEX "B")))

(M4! (OBJECT2 B2)
     (PROPERNAME (M3 (LEX B))))

* (DESCRIBE (ASSERT OBJECT3 #O3
                    PROPERNAME (BUILD LEX "C")))
```

```
(M6! (OBJECT3 B3)
     (PROPERNAME (M5 (LEX C))))

* (DESCRIBE (ASSERT OBJECT1 *O2
                    REL (BUILD LEX "between")
                    OBJECT2 *O1
                    OBJECT3 *O3))

(M8! (OBJECT1 B2)
     (OBJECT2 B1)
     (OBJECT3 B3)
     (REL (M7 (LEX between))))
```

Network

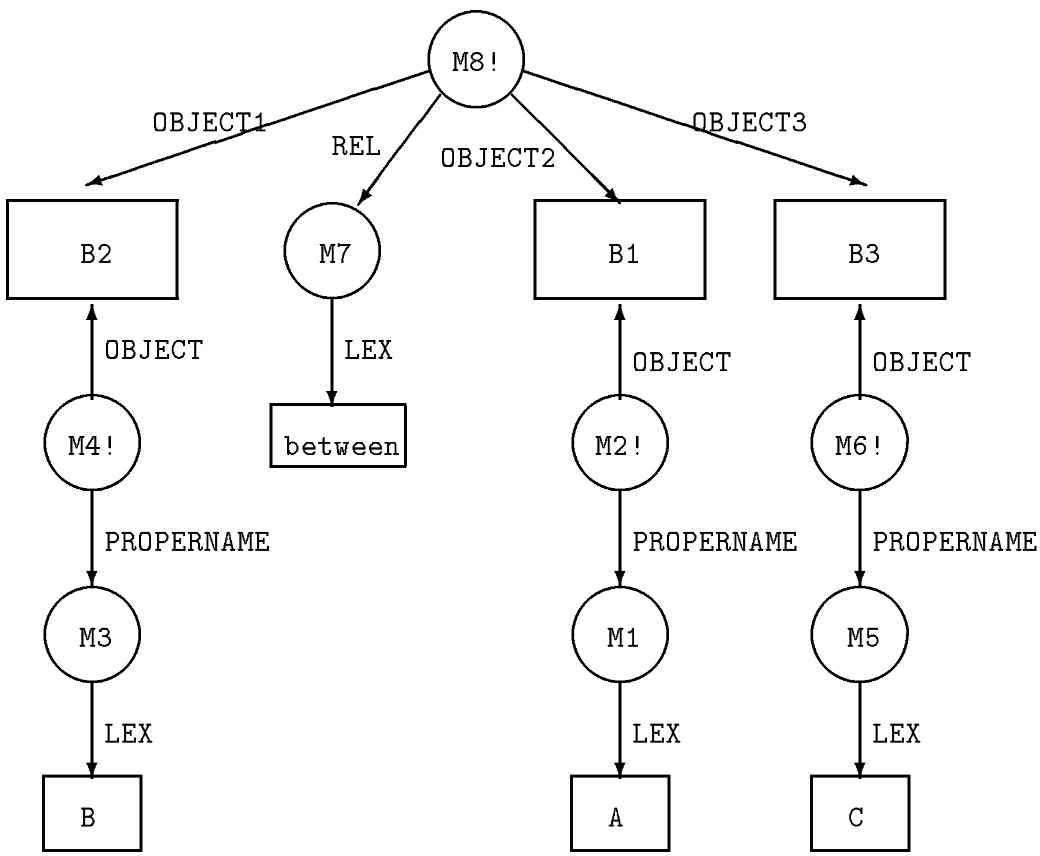

Figure 2.12: B is between A and C

Pictorially, this is shown in Figure 2.12.

Alternatives

This is essentially a default case frame that can be used to represent any relational proposition. The OBJECT/PROPERTY case frame is really the special case of REL/OBJECT1/OBJECTN where N = 1. (Some users prefer to call this case frame REL/ARG1/ARGN.)

Issues

None.

Related Entries

OBJECT/PROPERTY.

References

See [20] for the description of the syntax and semantics for this representation. Examples can be found in [22, 16, 12, 13, 14].

Author

Jason Kankiewicz, October 1993

2.7 AGENT/ACT

Syntax

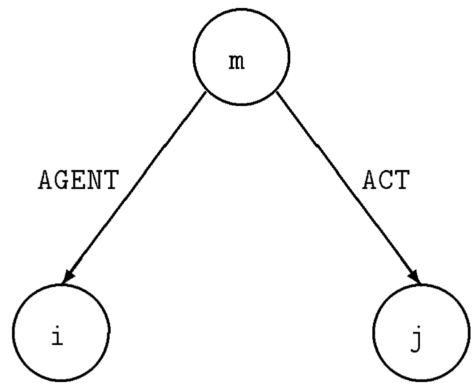

Figure 2.13:

If `i` and `j` are individual nodes and 'm' is an identifier not previously used, then Figure 2.13 is a network and `m` is a structured proposition node.

Semantics

⟦`m`⟧ is the proposition that agent ⟦`i`⟧ performs act ⟦`j`⟧.

Sample Context

Lucy sleeps can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT OBJECT #O
                    PROPERNAME (BUILD LEX "Lucy")))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT AGENT *O
                    ACT (BUILD LEX "sleeps")))

(M4! (ACT (M3 (LEX sleeps)))
     (AGENT B1))
```

Network

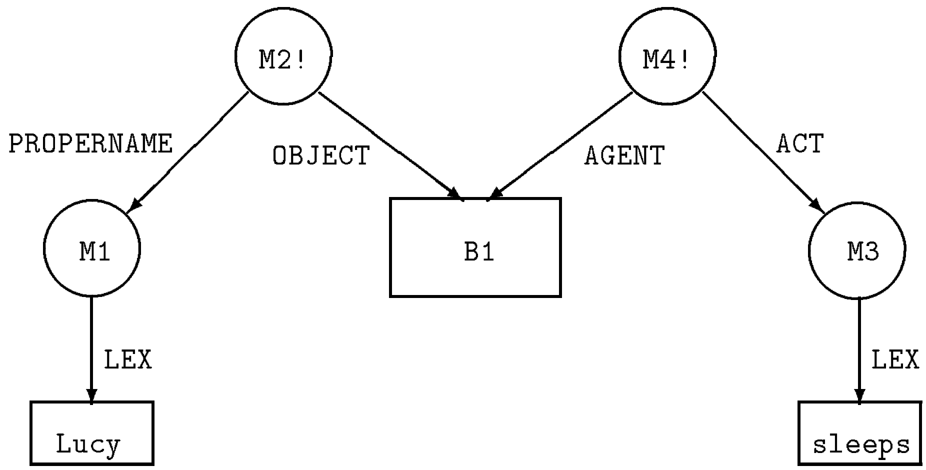

Figure 2.14: Lucy sleeps

Pictorially, this is shown in Figure 2.14. In this figure, the node `M4!` represents the proposition that `B1` sleeps.

Alternatives

None.

Issues

None.

Related Entries

Achieve Act, Acts, Conditional Act, Do-all Act, Do-one Act, Effects, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [26] for the description of the syntax and semantics for this representation. Examples are found in [17, 25].

Author

Jason Kankiewicz, October 1993

Chapter 3

Case Frames for Rules

3.1 MIN/MAX/ARG

Syntax

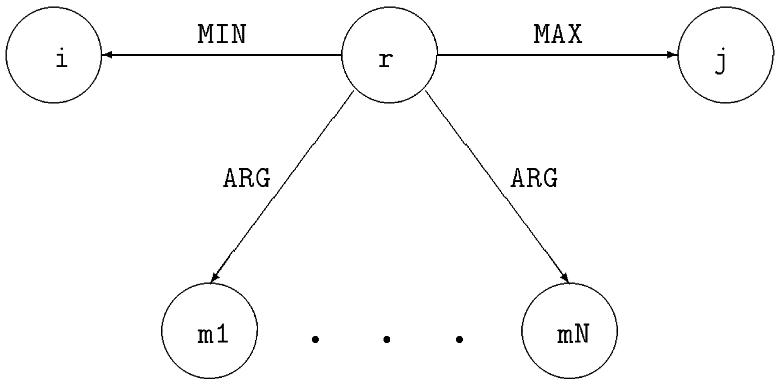

Figure 3.1:

If `m1`, ..., `mN` are structured proposition nodes, i and j are natural numbers, and 'r' is an identifier not previously used, then Figure 3.1 is a network and `r` is a rule node.

Semantics

⟦`r`⟧ is the proposition that there is a relevant connection between propositions ⟦`m1`⟧, ..., ⟦`mN`⟧ such that at least i and at most j of them are simultaneously true.

Sample Context

Lucy is not male can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT OBJECT #O
                    PROPERNAME (BUILD LEX "Lucy")))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT MIN 0
                    MAX 0
                    ARG (BUILD OBJECT *O
                               PROPERTY (BUILD LEX "male"))))

(M5! (MIN 0)
     (MAX 0)
     (ARG (M4 (OBJECT B1)
              (PROPERTY (M3 (LEX male))))))
```

Network

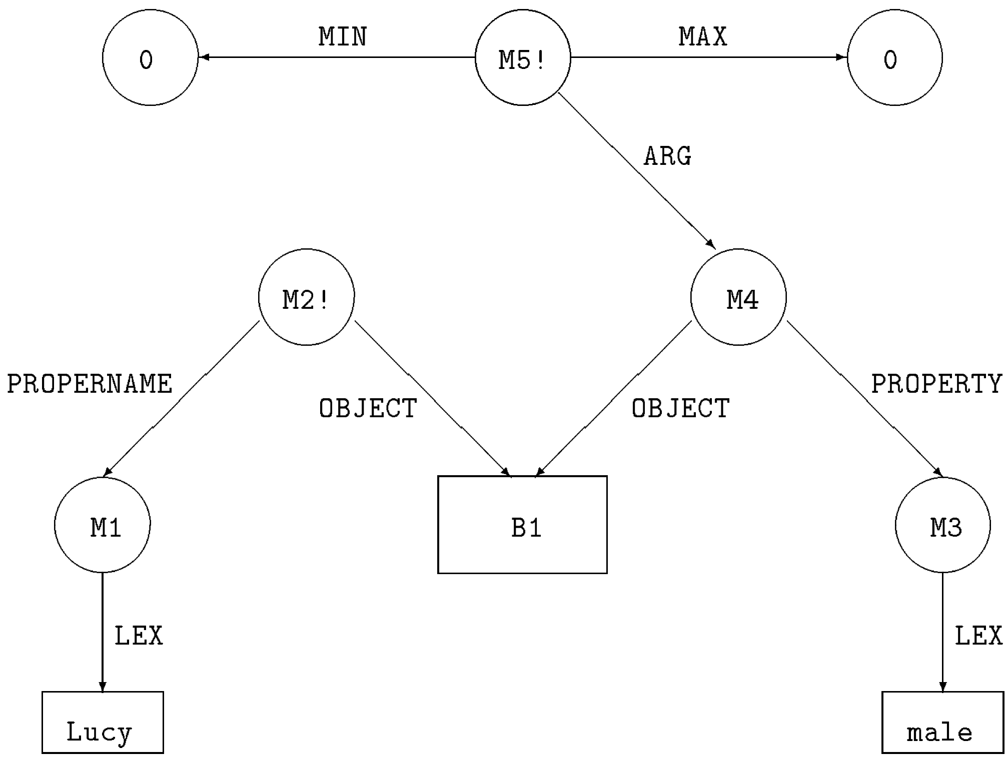

Figure 3.2: Lucy is not male

Pictorially, this is shown in Figure 3.2. In this figure, the node **M5!** represents the rule that proposition **M4** is not true.

Alternatives

None.

Issues

Rule **r** is called *AND/OR* and is a unified generalization of negation when $i = j = 0$, binary conjunction when $i = 2$ & $j = 2$, binary exclusive disjunction when $i = 0$ & $j = 1$, etc. This case frame is a part of SNIP and is not user-defined.

Related Entries

None.

References

See [21] for a description of the syntax and semantics of this representation. Examples are found in [20, 26, 10, 17, 23].

Author

Jason Kankiewicz, October 1993

3.2 THRESH/ARG

Syntax

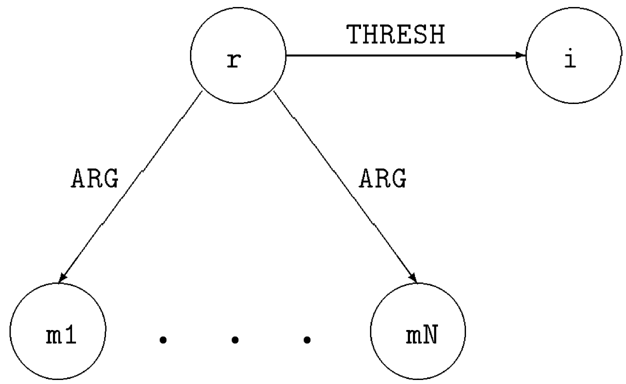

Figure 3.3:

If `m1`, ..., `mN` are structured proposition nodes, i is a natural number, and 'r' is an identifier not previously used, then Figure 3.3 is a network and `r` is a rule node.

Semantics

⟦`r`⟧ is the proposition that there is a relevant connection between propositions ⟦`m1`⟧, ..., ⟦`mN`⟧ such that either fewer than i of them are true or they are all true.

Sample Context

Lucy is both human and female or neither can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT OBJECT #O
                    PROPERNAME (BUILD LEX "Lucy")))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT THRESH 1
                    ARG (BUILD MEMBER *O
                               CLASS (BUILD LEX "human"))
                    ARG (BUILD OBJECT *O
                               PROPERTY (BUILD LEX "female"))))

(M7! (THRESH 1)
     (ARG (M4 (CLASS (M3 (LEX human)))
```

```
          (MEMBER B1))
      (M6 (OBJECT B1)
          (PROPERTY (M5 (LEX female))))))
```

Network

Pictorially, this is shown in Figure 3.4. In this figure, the node `M7!` represents the rule that either propositions `M4` and `M6` are both true or they are both false.

Alternatives

THRESH/ARG is actually a special case of THRESH/THRESHMAX/ARG where the THRESHMAX, j, is omitted and j = N - 1 by default. A *THRESH* can be represented using nested MIN/MAX/ARG but this should not be done.

Issues

Rule r is called *THRESH*; it is the material biconditional when $i = 1$ and there are two ARGs; it is a generalization of the material biconditional when $i \neq 1$. This case frame is a part of SNIP and is not user-defined.

Related Entries

THRESH/THRESHMAX/ARG.

References

See [20] for a description of the syntax and semantics of this representation. Examples are found in [21, 26, 23].

Author

Jason Kankiewicz, November 1993

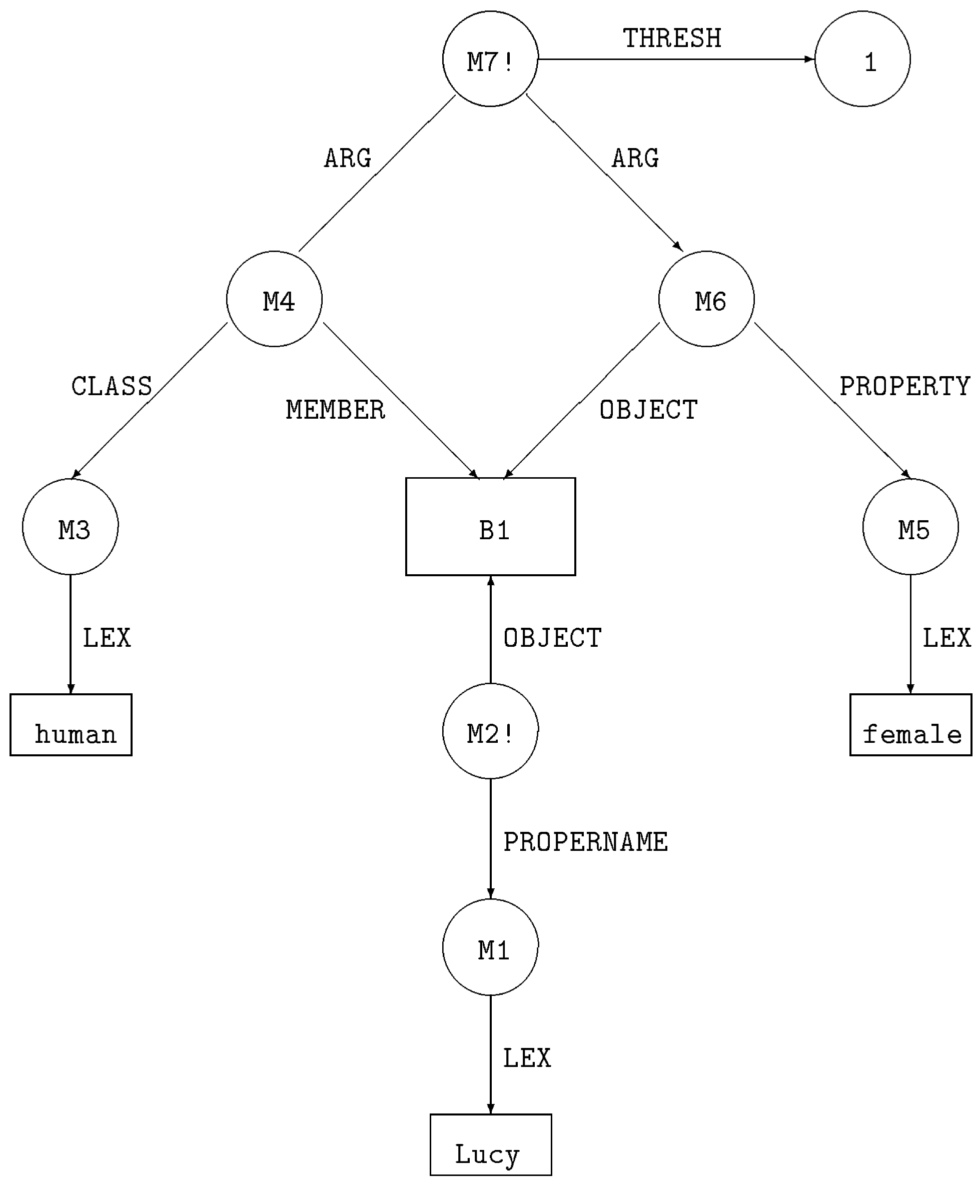

Figure 3.4: Lucy is both human and female or neither

3.3 THRESH/THRESHMAX/ARG

Syntax

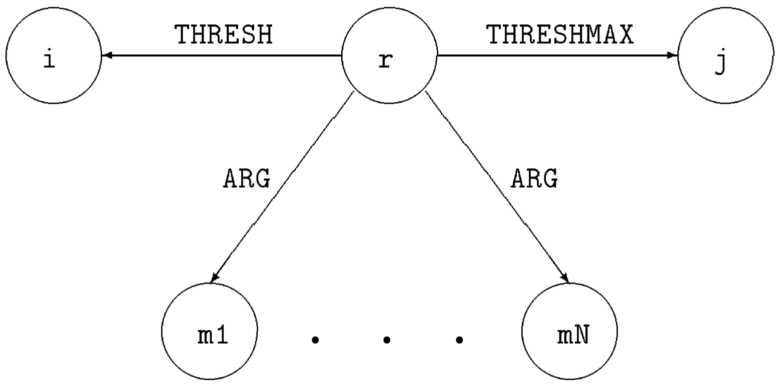

Figure 3.5:

If `m1`, ..., `mN` are structured proposition nodes, i and j are natural numbers, and 'r' is an identifier not previously used, then Figure 3.5 is a network and **r** is a rule node.

Semantics

⟦r⟧ is the proposition that there is a relevant connection between propositions ⟦m1⟧, ..., ⟦mN⟧ such that either fewer than i of them are true or more than j of them are true.

Sample Context

If Lucy is tall and slim, tall and pretty or slim and pretty then she is tall, slim and pretty can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT OBJECT #O
                    PROPERNAME (BUILD LEX "Lucy")))

(M2! (OBJECT B1)
     (PROPERNAME (M1 (LEX Lucy))))

* (DESCRIBE (ASSERT THRESH 2
                    THRESHMAX 2
                    ARG (BUILD OBJECT *O
                               PROPERTY (BUILD LEX "tall"))
                    ARG (BUILD OBJECT *O
                               PROPERTY (BUILD LEX "slim"))
                    ARG (BUILD OBJECT *O
```

```
                              PROPERTY (BUILD LEX "pretty"))))

(M9! (THRESH 2)
     (THRESHMAX 2)
     (ARG (M4 (OBJECT B1)
              (PROPERTY (M3 (LEX tall))))
          (M6 (OBJECT B1)
              (PROPERTY (M5 (LEX slim))))
          (M8 (OBJECT B1)
              (PROPERTY (M7 (LEX pretty))))))
```

Network

Pictorially, this is shown in Figure 3.6. In this figure, the node **M9!** represents the rule that no two of the propositions **M4**, **M6** and **M8** can be true without all of them being true.

Alternatives

THRESH i, j and *AND/OR* i, j are inverses of each other.

Issues

Rule r is called *THRESH* and it may be used in forward or backward inference to conclude that one or more of its arguments is to be asserted, or that the negation of one or more of its arguments is to be asserted. This case frame is a part of SNIP and is not user-defined.

Related Entries

THRESH/ARG.

References

See [20] for a description of the syntax and semantics of this representation. Examples are found in [21, 26, 23].

Author

Jason Kankiewicz, November 1993

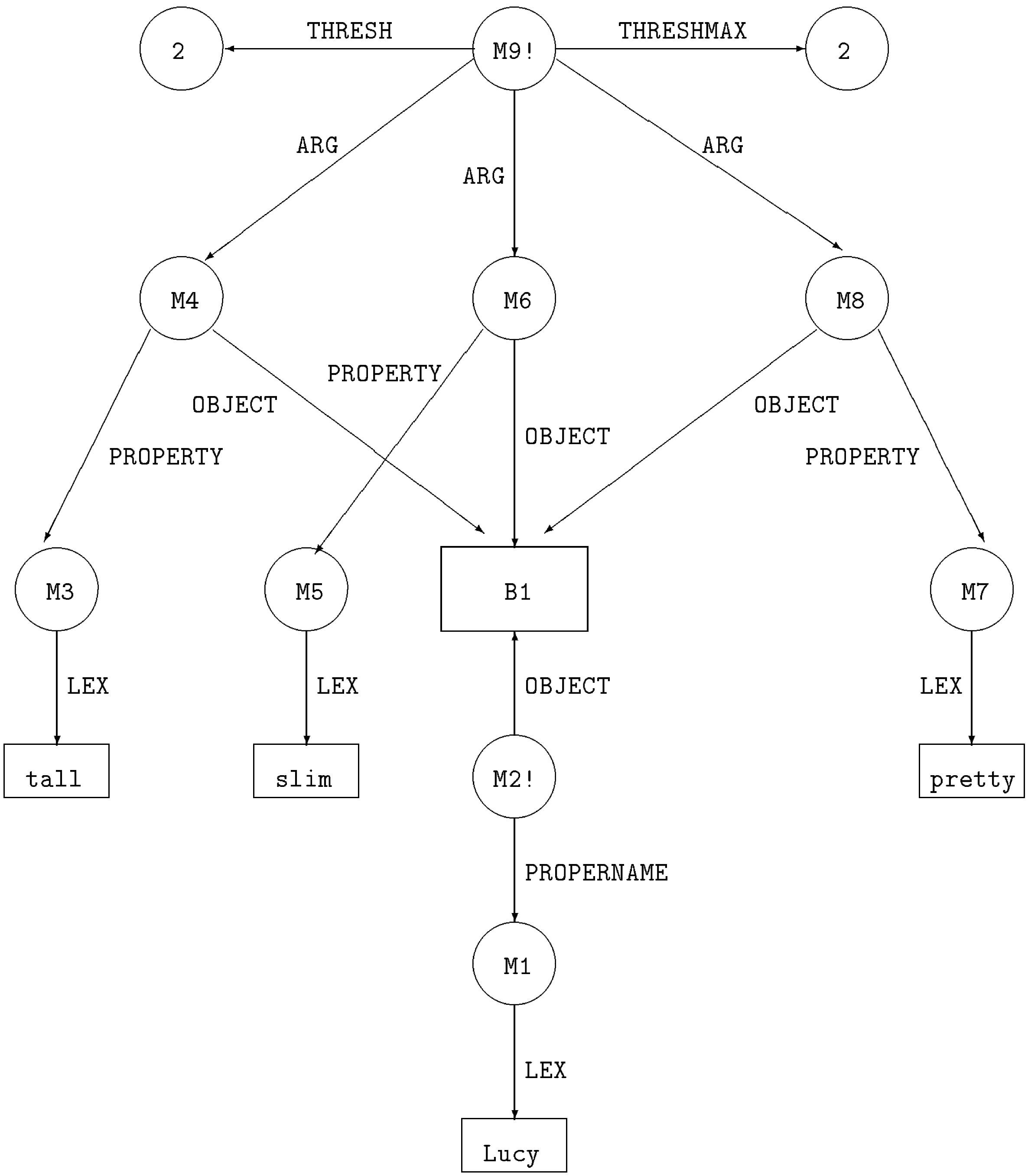
2
THRESH
M9!
THRESHMAX
2
ARG
ARG
ARG
M4
M6
M8
PROPERTY
OBJECT
OBJECT
OBJECT
PROPERTY
PROPERTY
M3
M5
B1
M7
LEX
LEX
OBJECT
LEX
tall
slim
M2!
pretty
PROPERNAME
M1
LEX
Lucy

Figure 3.6:

3.4 THRESH/&ANT/CQ (Numerical Entailment)

Syntax

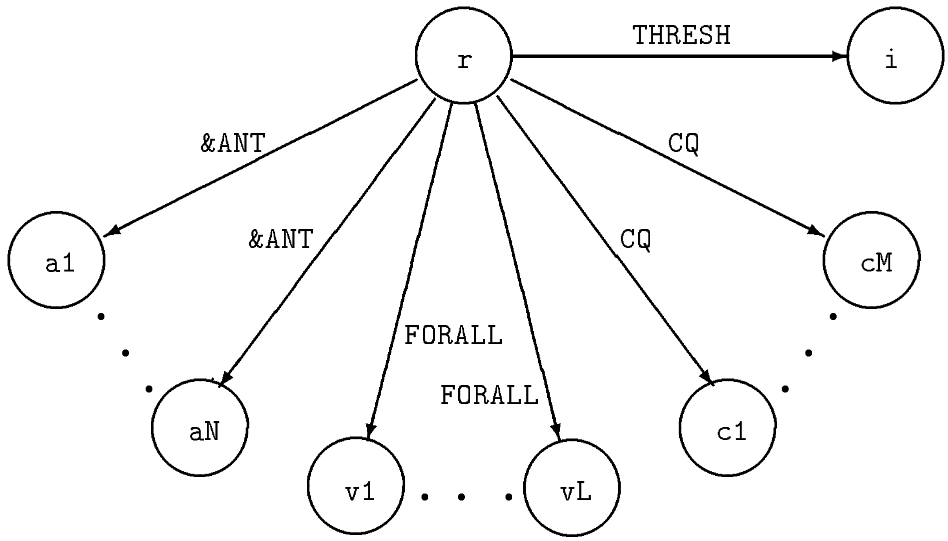

Figure 3.7:

If `a1`, ..., `aN` and `c1`, ..., `cM` are structured proposition nodes, i is a natural number and 'r' is an identifier not previously used then Figure 3.7 is a network and `r` is a rule node.

Semantics

⟦`r`⟧ is the proposition that if i or more of ⟦`a1`⟧, ..., ⟦`aN`⟧ are true then all of ⟦`c1`⟧, ..., ⟦`cM`⟧ are also true.

Sample Context

If food is sweet and sour, sweet and hot, sour and hot, or sweet, sour and hot then it is spicy can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT MEMBER #0
                    CLASS (BUILD LEX "food")))

(M2! (CLASS (M1 (LEX food))))
     (MEMBER B2)


* (DESCRIBE (ASSERT THRESH 2
```

```
                    &ANT (BUILD OBJECT *O
                                PROPERTY (BUILD LEX "sweet"))
                    &ANT (BUILD OBJECT *O
                                PROPERTY (BUILD LEX "sour"))
                    &ANT (BUILD OBJECT *O
                                PROPERTY (BUILD LEX "hot"))
                    CQ (BUILD OBJECT *O
                              PROPERTY (BUILD LEX "spicy"))))

(M11! (THRESH 2)
      (&ANT (M4 (OBJECT B2)
                (PROPERTY (M3 (LEX sweet))))
            (M6 (OBJECT B2)
                (PROPERTY (M5 (LEX sour))))
            (M8 (OBJECT B2)
                (PROPERTY (M7 (LEX hot)))))
      (CQ (M10 (OBJECT B2)
               (PROPERTY (M9 (LEX spicy))))))
```

Network

Pictorially, this is shown in Figure 3.8. In this figure, the node `M11!` represents the rule that proposition `M10` is true if at least two of the propositions `M4`, `M6` and `M8` are true.

Alternatives

None.

Issues

This case frame is a part of SNIP and is not user-defined.

Related Entries

None.

References

See [21] for a description of the syntax and semantics of this representation. Examples are found in [26, 23].

Author

Jason Kankiewicz, November 1993

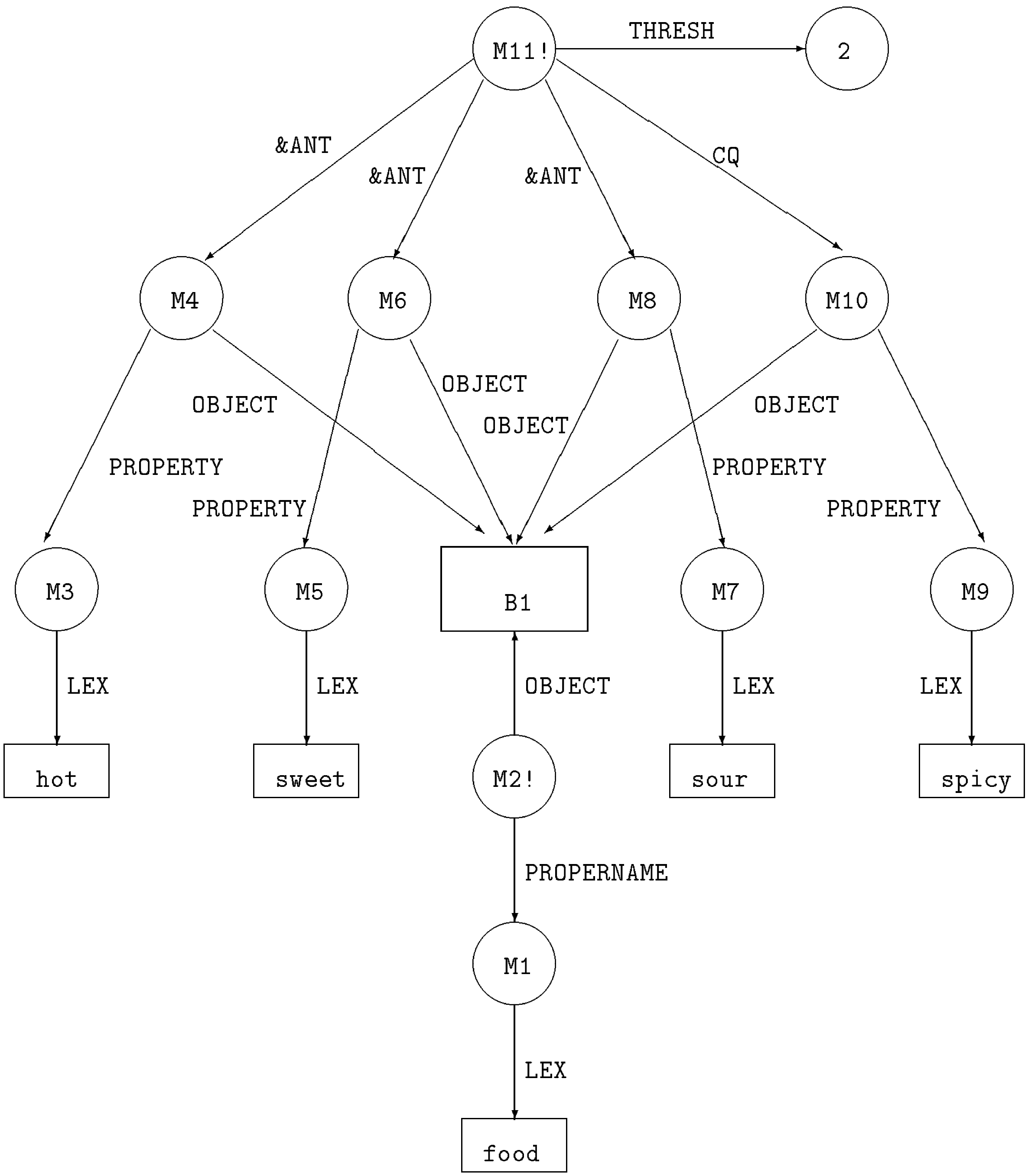
M11!
THRESH
2
&ANT
&ANT
&ANT
CQ
M4
M6
M8
M10
OBJECT
OBJECT
OBJECT
OBJECT
PROPERTY
PROPERTY
PROPERTY
PROPERTY
M3
M5
B1
M7
M9
LEX
LEX
OBJECT
LEX
LEX
hot
sweet
M2!
sour
spicy
PROPERNAME
M1
LEX
food

Figure 3.8:

3.5 EMIN/EMAX/ETOT/PEVB/&ANT/CQ (Numerical Quantification)

Syntax

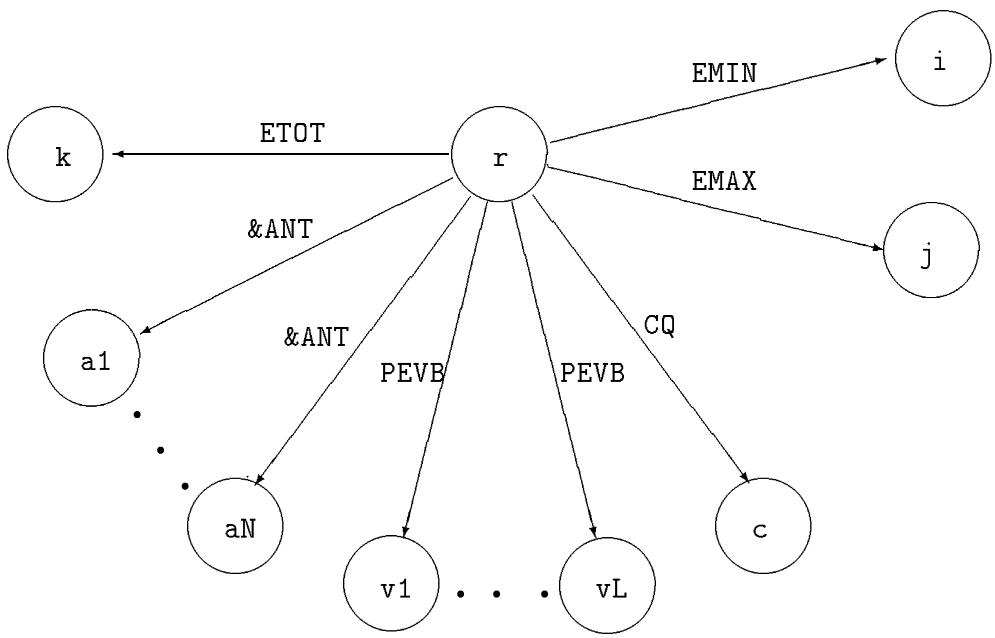

Figure 3.9:

If `a1`, ..., `aN` and `c1`, ..., `cM` are structured proposition nodes, `v1`, ..., `vL` are variable nodes dominated by one or more of `a1`, ..., `aN` and `c`, i, j and k are natural numbers such that $1 \leq i \leq j \leq k$ and 'r' is an identifier not previously used then Figure 3.9 is a network and `r` is a rule node.

Semantics

⟦`r`⟧ is the proposition that of the k sequences of individuals which, when substituted for the sequence `v1`, ..., `vL` of variable nodes, make all of ⟦`a1`⟧, ..., ⟦`aN`⟧ true, between i and j of them also satisfy ⟦`c`⟧.

Sample Context

Of 26 letters, at least 1 and no more than 6 are vowels can be represented by the following commands:

SNePSUL Interaction

```
* (DESCRIBE (ASSERT EMIN 1
```

```
                 EMAX 6
                 ETOT 26
                 PEVB $CHAR
                 &ANT (BUILD MEMBER *CHAR
                             CLASS (BUILD LEX "letters"))
                 CQ (BUILD OBJECT *CHAR
                           PROPERTY (BUILD LEX "vowel"))))

(M3! (PEVB V1)
     (EMIN 1)
     (EMAX 6)
     (ETOT 26)
     (&ANT (P1 (CLASS (M1 (LEX letters)))
               (MEMBER V1)))
     (CQ (P2 (OBJECT V1)
             (PROPERTY (M2 (LEX vowel))))))
```

Network

Pictorially, this is shown in Figure 3.10. In this figure, the node `M3!` represents the rule that if the proposition node assigned to `V1` implies the truth of proposition `P1` then proposition `P2` is true.

Alternatives

None.

Issues

This case frame is a part of SNIP and is not user-defined.

Related Entries

None.

References

See [19] for a description of the syntax and semantics of this representation. Examples are found in [26, 23].

Author

Jason Kankiewicz, November 1993

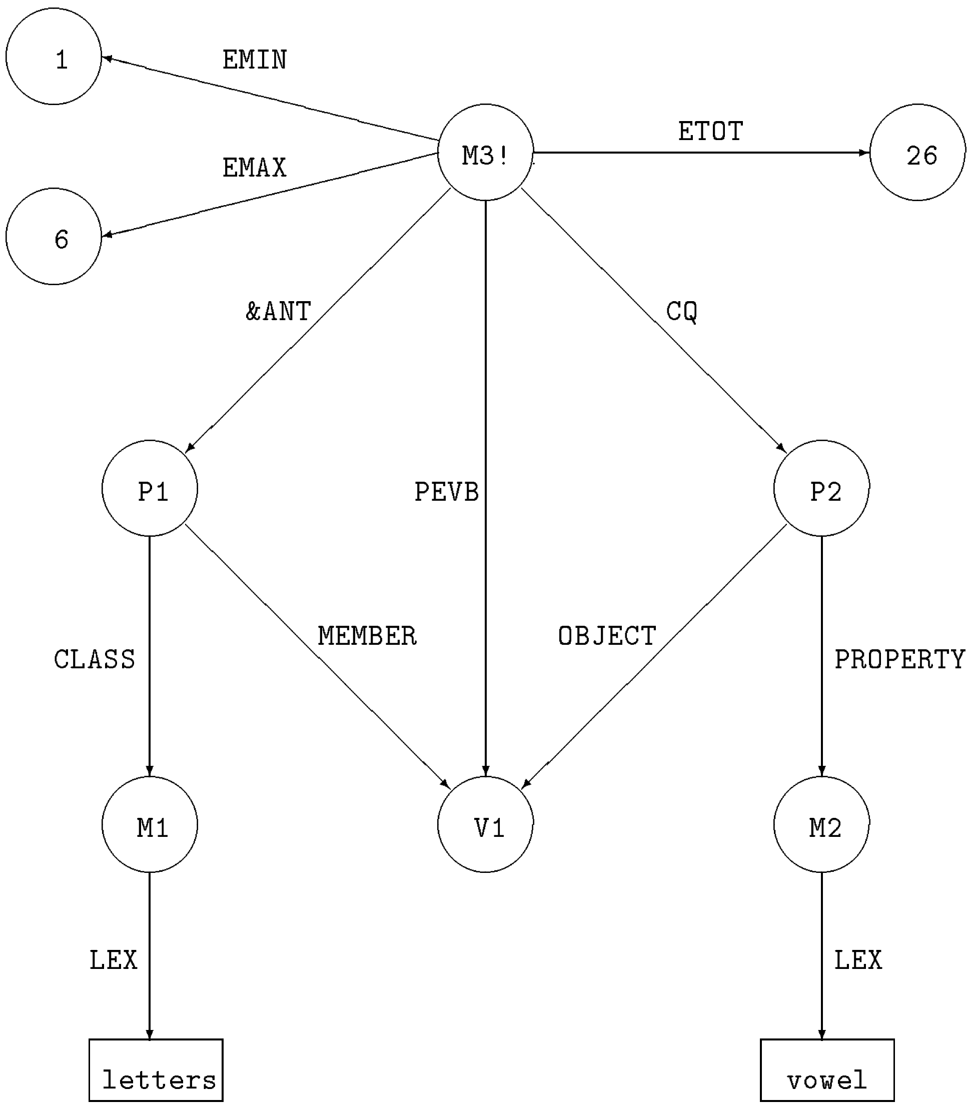

Figure 3.10: Of 26 letters, at least 1 and no more than 6 are vowels

Chapter 4

Case Frames for Plans and Acts

4.1 Achieve Act

This is a *control action* for the acting executive. See entries Sequence, Do-One , Do-All, Conditional, and Iteration for descriptions of other control actions.

Syntax

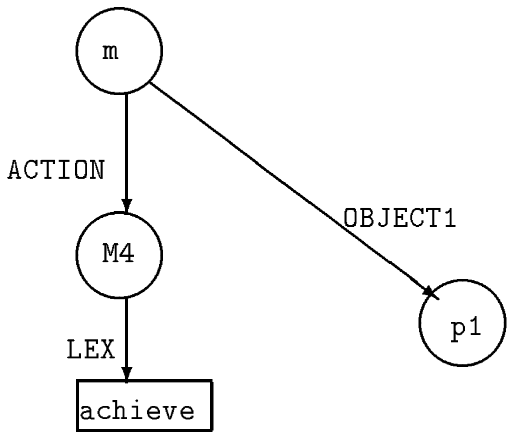

Figure 4.1: Achieve Act

`m` is a structured individual node with an `ACTION` arc to an individual node which is expressed as '`achieve`' and an `OBJECT1` arc to a proposition node `p1`.

Semantics

The semantics of control actions is defined operationally based on the design of the acting executive (see entry Acts). ⟦`m`⟧ is the act of achieving that ⟦`p1`⟧ is believed. If ⟦`p1`⟧ is believed then nothing happens, otherwise plans to achieve ⟦`p1`⟧ are deduced and a `do-one` of those is placed at the front of the act queue.

Sample Context

The act of achieving "`A is clear`" is shown here.

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX ``achieve'')
           OBJECT1 (BUILD PROPERTY (BUILD LEX ``clear'')
                          OBJECT (BUILD LEX ``A'')))
...
(M72 (ACTION (M4 (LEX achieve)))
     (OBJECT1 (M71 (PROPERTY (M26 (LEX clear)))
                   (OBJECT (M42 (LEX A))))))
```

Network

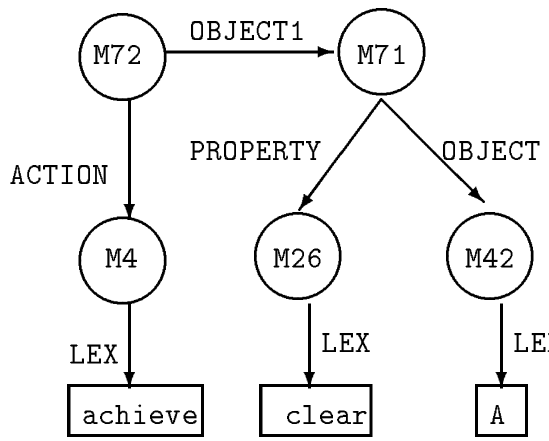

Figure 4.2: "`Achieve A is clear.`"

In Figure 4.2, ⟦`M72`⟧ is the act of achieving (⟦`M4`⟧) that A (⟦`M42`⟧) is clear (⟦`M26`⟧).

Alternatives

None.

Issues

See entry Acts.

Related Entries

Acts, Conditional Act, Do-all Act, Do-one Act, Effects, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, August 6, 1991

4.2 Acts

In SNePS we consider acts as mental objects. Acts are represented as structured individual nodes with the following syntax:

Syntax

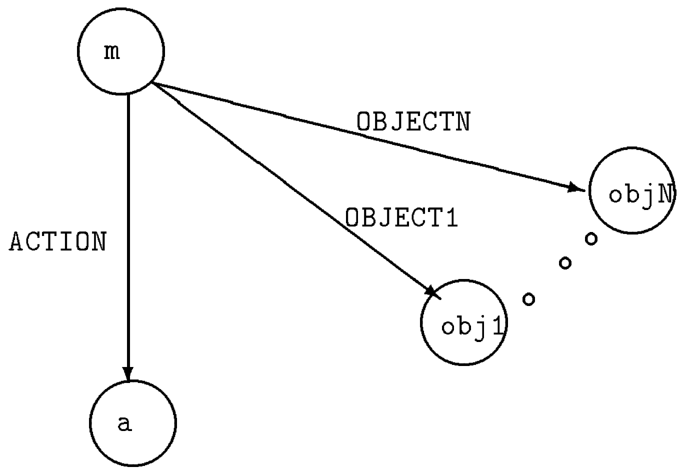

Figure 4.3:

`m` is a structured individual node with an `ACTION` arc to the individual node, `a`, and `OBJECT1`, . . . , `OBJECTN` arcs to individual nodes `obj1`, . . . , `objN`, respectively.

Semantics

⟦`m`⟧ is the act whose action is ⟦`a`⟧ and ⟦`obj1`⟧, . . . , ⟦`objN`⟧ are the objects of ⟦`a`⟧.

Sample Context

In a blocksworld, the act of picking up an object can be represented as:

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX "pickup")
           OBJECT1 (BUILD LEX "A"))
...
(M3! (ACTION (M1 (LEX pickup))
     (OBJECT1 (M2 (LEX A))))
```

Network

⟦`M3`⟧ is the act of picking (⟦`M1`⟧) up A (⟦`M2`⟧).

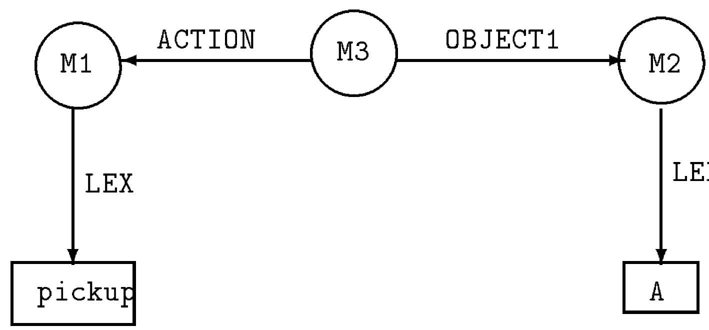

Figure 4.4: "`Pick up A.`"

Alternatives

None.

Issues

There is a system called the SNePS Acting System that has an acting executive. The acting executive enables the modeled agent to perform actions in the external world. One issue currently being researched is to amalgamate the notions of acting and inference in a single model. This has several advantages including some notions of reactivity and sensory acts. Representations and discussions relating to this can be found in [4].

Related Entries

Achieve Act, Conditional Act, Do-all Act, Do-one Act, Effects, Iteration Act, Plan-Act Proposition, Plan-Goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, July 30, 1991

4.3 Conditional Act

This is a *control action* for the acting executive. See entries Sequence , Do-All, Do-One, and Iteration for descriptions of other control actions.

Syntax

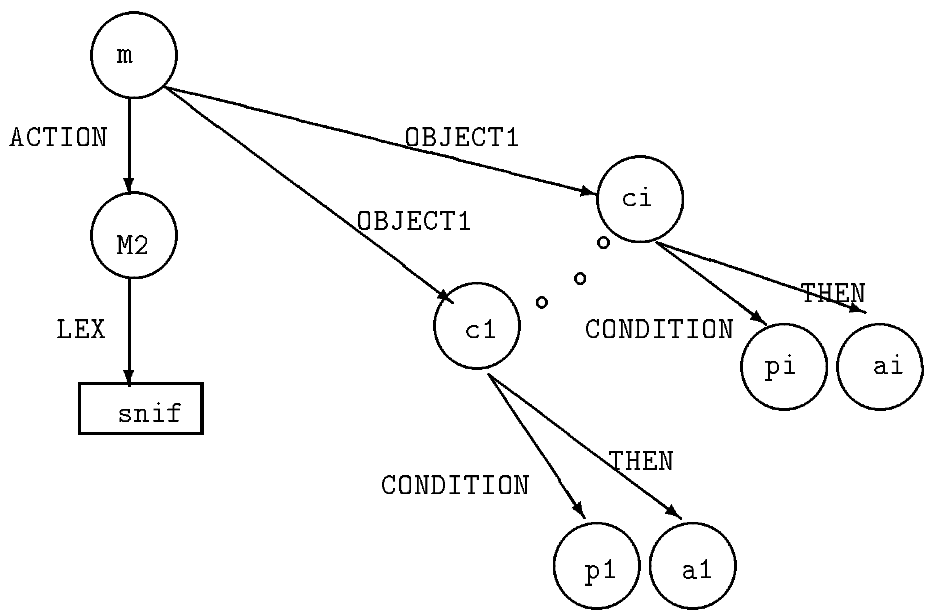

Figure 4.5: Conditional Act

`m` is a structured individual node with an `ACTION` arc to an individual node which is expressed as '`snif`' and `OBJECT1` arcs to structured individual nodes, `c1`, ..., `ci`. Each of the `c1`, ..., `ci` has a `CONDITION` arc to proposition nodes `p1`, ..., `pi`, and a `THEN` arc to structured act nodes `a1`, ..., `ai`, respectively.

Semantics

The semantics of control actions is defined operationally based on the design of the acting executive (see entry Acts). ⟦`m`⟧ is the act of first testing all of the propositions ⟦`p1`⟧, ⟦`pi`⟧. If none are true, nothing happens. Otherwise a `do-one` act whose objects are all the acts ⟦`a1`⟧, ..., ⟦`ai`⟧ whose corresponding ⟦`p1`⟧, ..., ⟦`pi`⟧'s are true is put on the front of the act queue.

Sample Context

The act "`Pick up A if A is red.`" is shown here.

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX ''snif'')
           OBJECT1 (BUILD CONDITION (BUILD PROPERTY (BUILD LEX ''red'')
                                           OBJECT (BUILD LEX ''A''))
                          THEN (BUILD ACTION (BUILD LEX ''pickup'')
                                       OBJECT (BUILD LEX ''A''))))
...
(M253 (ACTION (M2 (LEX snif)))
      (OBJECT1 (M252 (CONDITION (M251 (PROPERTY (M211 (LEX red)))
                                       (OBJECT (M42 (LEX A)))))
                     (THEN (M165 (ACTION (M13 (LEX pickup)))
                                  (OBJECT (M42)))))))
```

Network

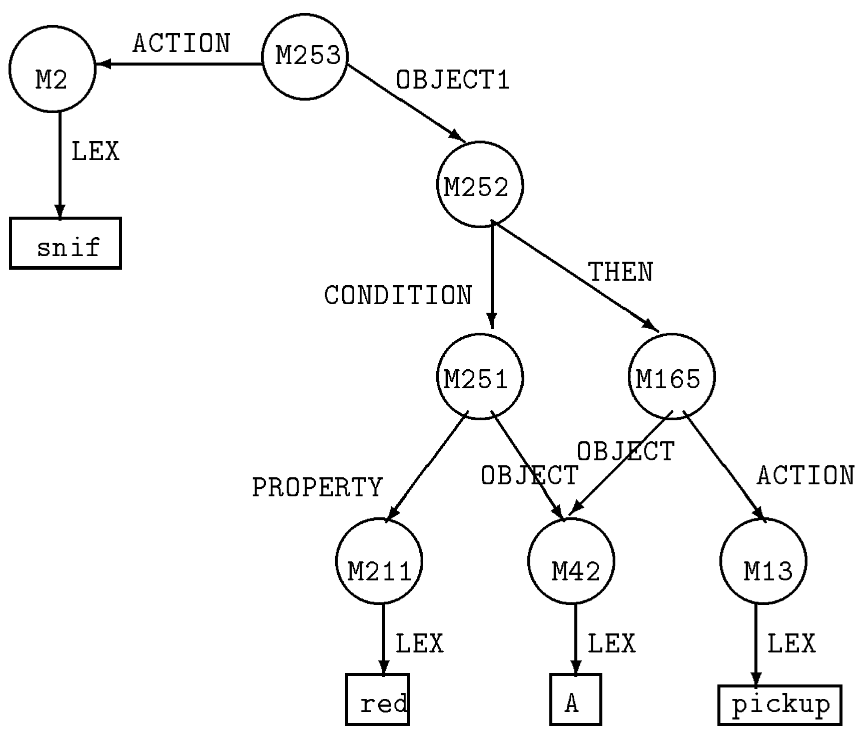

Figure 4.6: "`Pick up A if A is red.`"

In Figure 4.6, ⟦`M253`⟧ is the conditional act picking up `A` (⟦`M165`⟧) if `A` is red (⟦`M251`⟧).

Alternatives

None.

Issues

See entry Acts.

Related Entries

Achieve Act, Acts, Do-all Act, Do-one Act, Effects, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, August 6, 1991

4.4 Do-All Act

This is a *control action* for the acting executive. See entries for Sequence, Do-One, Conditional, and Iteration Acts for descriptions of other control actions.

Syntax

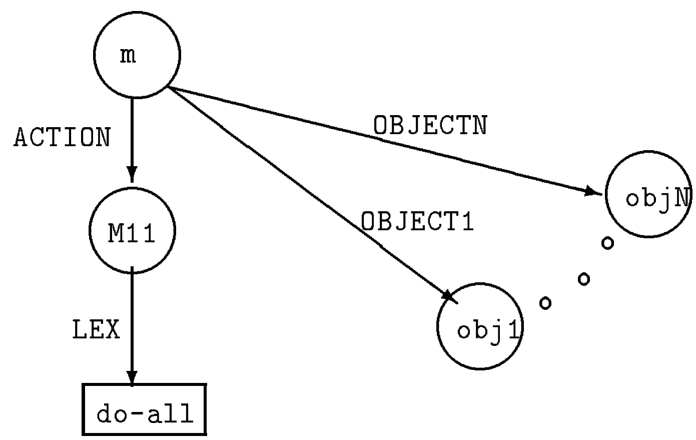

Figure 4.7:

`m` is a structured individual node with an `ACTION` arc to an individual node which is expressed as '`do-all`' and `OBJECT1` arcs to structured individual act nodes, `act1`, ..., `actN`.

Semantics

The semantics of control actions is defined operationally based on the design of the acting executive (see entry Acts). All of the `acti`'s are performed in some unspecified order. ⟦`m`⟧ is the act of non-deterministically doing all of the acts ⟦`act1`⟧, ..., ⟦`actN`⟧.

Sample Context

The act of saying Hello, Jambo, and Aloha is shown below.

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX "do-all")
           OBJECT1 ((BUILD ACTION (BUILD LEX "say") = SAY
                           OBJECT (BUILD LEX "Hello"))
                    (BUILD ACTION *SAY
                           OBJECT (BUILD LEX "Jambo"))
                    (BUILD ACTION *SAY
                           OBJECT (BUILD LEX "Aloha"))))
...
```

```
(M207 (ACTION (M11 (LEX do-all)))
      (OBJECT1 (M202 (ACTION (M199 (LEX say)))
                     (OBJECT (M201 (LEX Hello))))
               (M204 (ACTION (M199))
                     (OBJECT (M203 (LEX Jambo))))
               (M206 (ACTION (M199))
                     (OBJECT (M205 (LEX Aloha))))))
```

Network

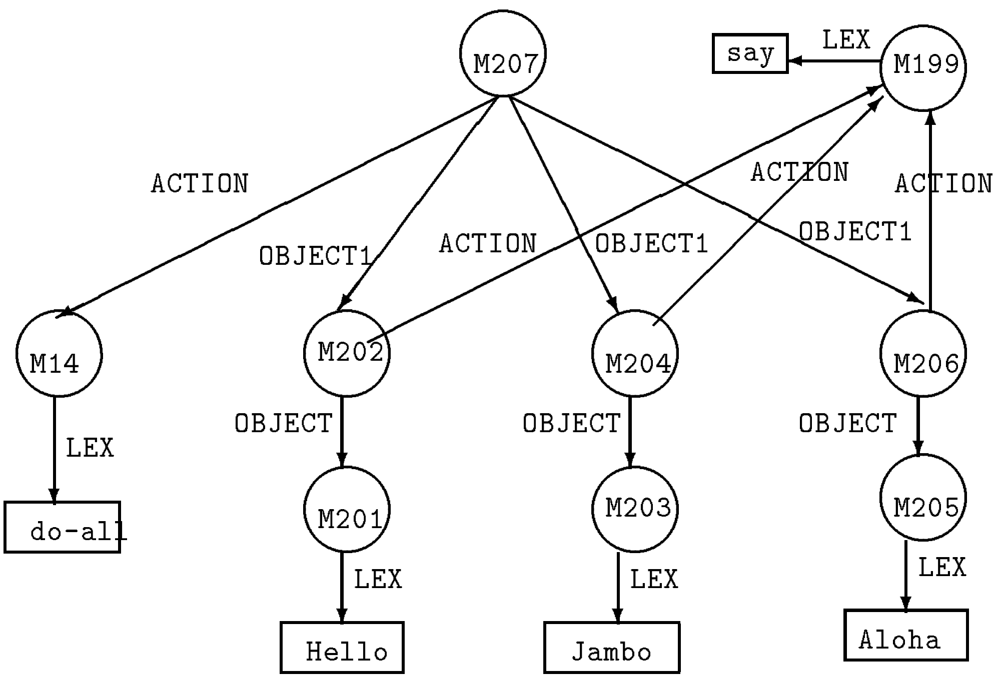

Figure 4.8: "`Say Hello, Jambo, and Aloha.`"

⟦`M205`⟧ is the act of saying `Hello` (⟦`M201`⟧), `Jambo` (⟦`M202`⟧), and `Aloha` (⟦`M203`⟧) in Figure 4.8.

Alternatives

None.

Issues

See entry Acts.

Related Entries

Achieve Act, Acts, Conditional Act, Do-one Act, Effects, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, August 6, 1991

4.5 Do-One Act

This is a *control action* for the acting executive. See entries Sequence , Do-All, Conditional, and Iteration for descriptions of other control actions.

Syntax

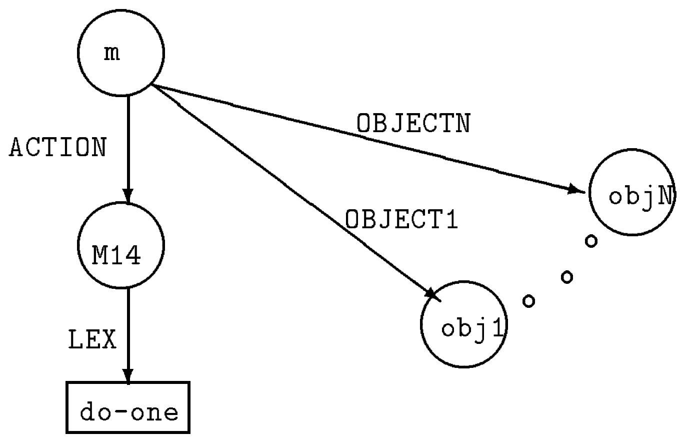

Figure 4.9:

`m` is a structured individual node with an `ACTION` arc to an individual node which is expressed as '`do-one`' and `OBJECT1` arcs to structured individual act nodes, `act1`, . . . , `actN`.

Semantics

The semantics of control actions is defined operationally based on the design of the acting executive (see entry Acts). One of the `acti`'s is performed. ⟦`m`⟧ is the act of non-deterministically doing one of the acts ⟦`act1`⟧, . . . , ⟦`actN`⟧.

Sample Context

The act of saying Hello or Jambo or Aloha is shown below.

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX "do-one")
           OBJECT1 ((BUILD ACTION (BUILD LEX "say") = SAY
                           OBJECT (BUILD LEX "Hello"))
                    (BUILD ACTION *SAY
                           OBJECT (BUILD LEX "Jambo"))
                    (BUILD ACTION *SAY
                           OBJECT (BUILD LEX "Aloha"))))
...
```

```
(M208 (ACTION (M14 (LEX do-one)))
      (OBJECT1 (M202 (ACTION (M199 (LEX say)))
                     (OBJECT (M201 (LEX Hello))))
               (M204 (ACTION (M199))
                     (OBJECT (M203 (LEX Jambo))))
               (M206 (ACTION (M199))
                     (OBJECT (M205 (LEX Aloha))))))
```

Network

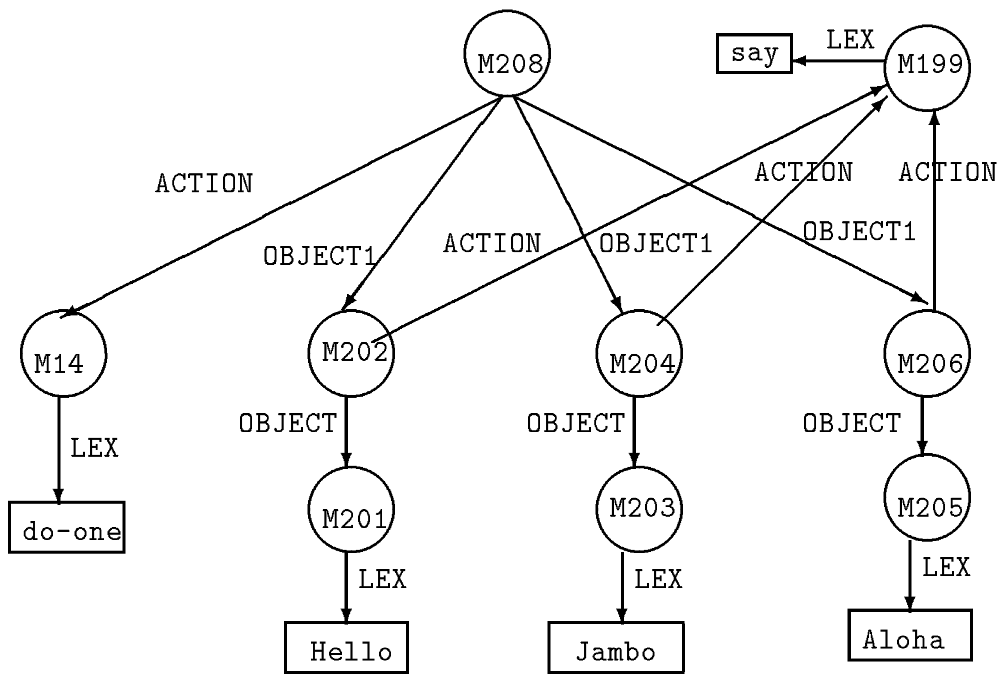

Figure 4.10: "Say Hello or Jambo or Aloha"

⟦M204⟧ is the act of saying Hello (⟦M201⟧) or Jambo (⟦M202⟧) or Aloha (⟦M203⟧) in Figure 4.10.

Alternatives

None.

Issues

See entry Acts.

Related Entries

Achieve Act, Acts, Conditional Act, Do-all Act, Effects, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, August 6, 1991

4.6 Effects

This representation is used to specify effects of actions.

Syntax

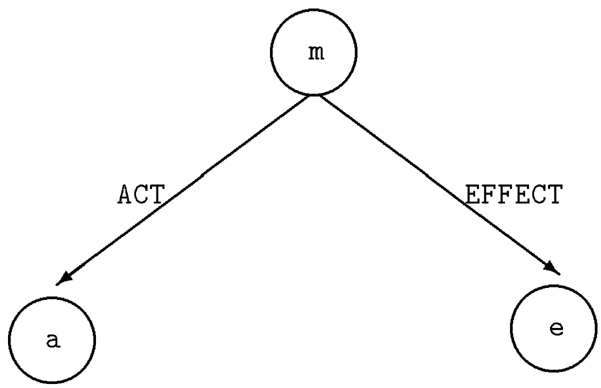

Figure 4.11: The syntax of an effect proposition.

`m` is a structured proposition node with an `ACT` arc to a structured act node, `a`, and an `EFFECT` arc to a structured proposition node `e`.

Semantics

⟦m⟧ is the proposition that the effect of act ⟦a⟧ is ⟦e⟧.

Sample Context

A representation of, "`After picking up A, it is no longer held.`" is shown below.

SNePSUL Interaction

```
* (ASSERT ACT (BUILD ACTION (BUILD LEX ''pickup'')
                     OBJECT (BUILD LEX ''A''))
          EFFECT (BUILD OBJECT  (BUILD LEX ''A'')
                        PROPERTY (BUILD LEX ''held'')))
(M169! (ACT (M165 (ACTION (M13 (LEX pickup)))
                  (OBJECT (M42 (LEX A)))))
       (EFFECT (M160 (OBJECT (M42))
                     (PROPERTY (M33 (LEX held))))))
```

Network

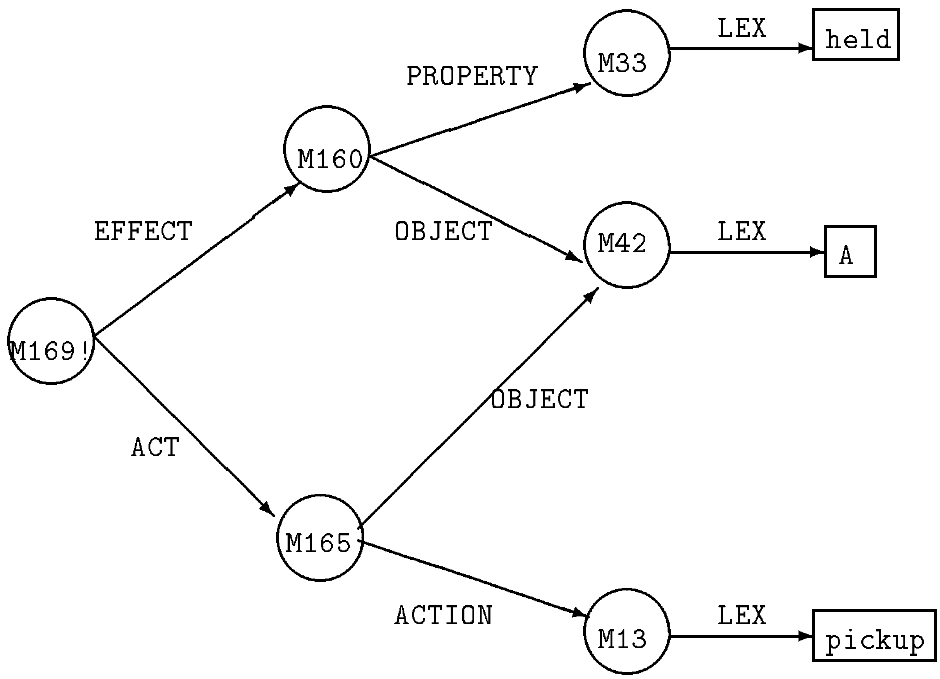

Figure 4.12: "`After picking up A, it is no longer held.`"

Alternatives

None.

Issues

The effects and preconditions of an act are represented in the same way as other beliefs about other mental objects; we do not need a special data structure (or an operator formalism) for acts in which pre and post conditions are special fields. Such a representation also enables us to assert context-dependent effects of actions (See references above). Also, see entry Acts.

Related Entries

Achieve Act, Acts, Conditional Act, Do-all Act, Do-one Act, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27].

Author

Deepak Kumar, August 2, 1991

4.7 Iteration Act

This is a *control action* for the acting executive. See entries Sequence, Do-One, Do-All, and Conditional Acts for descriptions of other control actions.

Syntax

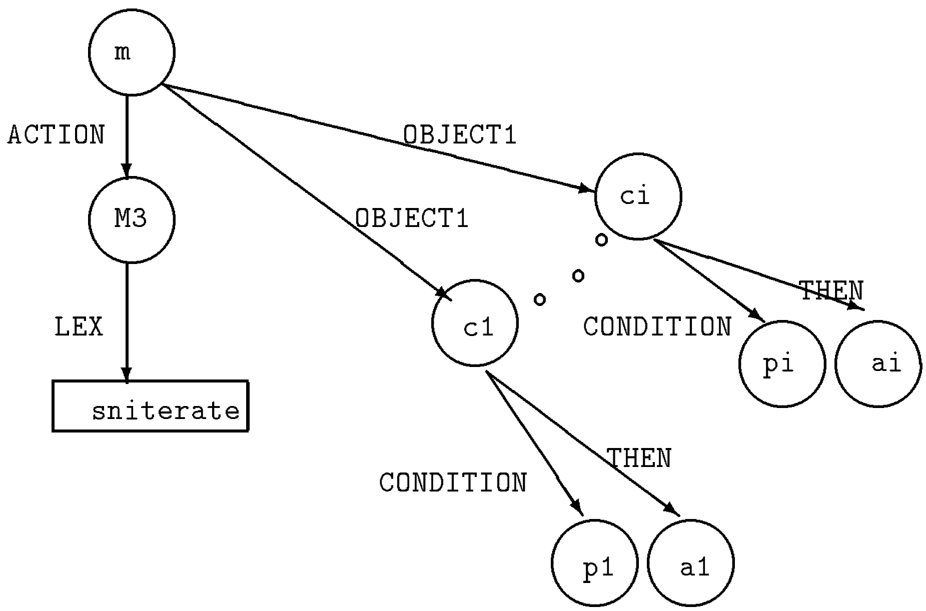

Figure 4.13: Iteration Act

`m` is a structured individual node with an `ACTION` arc to an individual node which is expressed as '`sniterate`' and `OBJECT1` arcs to structured individual nodes, `c1`, ..., `ci`. Each of the `c1`, ..., `ci` has a `CONDITION` arc to proposition nodes `p1`, ..., `pi`, and a `THEN` arc to structured act nodes `a1`, ..., `ai`, respectively.

Semantics

The semantics of control actions is defined operationally based on the design of the acting executive (see entry Acts). ⟦`m`⟧ is the act of first testing all of the propositions ⟦`p1`⟧, ⟦`pi`⟧. If none are true, nothing happens. Otherwise a `do-one` act whose objects are all the acts ⟦`a1`⟧, ..., ⟦`ai`⟧ whose corresponding ⟦`p1`⟧, ..., ⟦`pi`⟧'s are true is put on the front of the act queue after the `do-all` act is reinserted.

Sample Context

The act "`While A is red pick it up.`" is shown here.

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX ``sniterate'')
           OBJECT1 (BUILD CONDITION (BUILD PROPERTY (BUILD LEX ``red'')
                                           OBJECT (BUILD LEX ``A''))
                          THEN (BUILD ACTION (BUILD LEX ``pickup'')
                                      OBJECT (BUILD LEX ``A''))))
...
(M254 (ACTION (M2 (LEX sniterate)))
      (OBJECT1 (M252 (CONDITION (M251 (PROPERTY (M211 (LEX red)))
                                      (OBJECT (M42 (LEX A)))))
                     (THEN (M165 (ACTION (M13 (LEX pickup)))
                                 (OBJECT (M42)))))))
```

Network

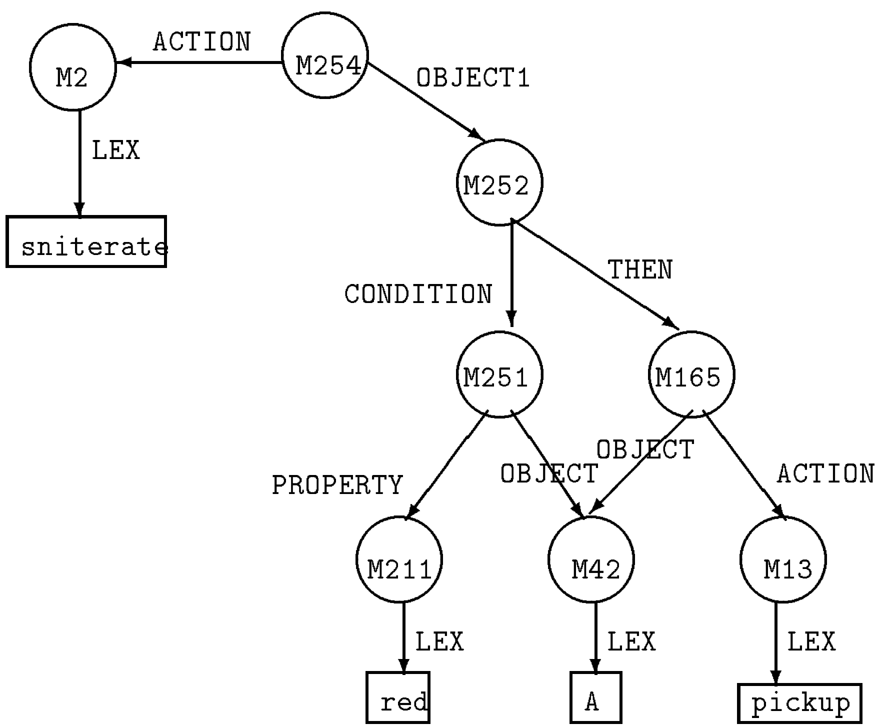

Figure 4.14: "While A is red pick it up."

Alternatives

None.

Issues

See entry Acts.

Related Entries

Achieve Act, Acts, Do-all Act, Do-one Act, Effects, Conditional Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, August 6, 1991

4.8 Plan-Act Proposition

A plan is a structure of acts. The structuring syntax for plans is described in terms of control actions (See entries Sequence, Do-one, Do-all, Conditional, Iteration, and Achieve Acts).

Syntax

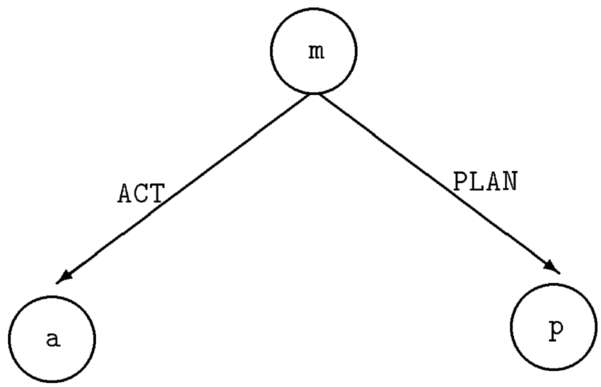

Figure 4.15: A plan-act proposition.

`m` is a structured proposition node with an `ACT` arc to a structured act node, `a`, and a `PLAN` arc to a structured act node, `p`.

Semantics

⟦m⟧ is a proposition that ⟦p⟧ is a plan for carrying out act ⟦a⟧. The act ⟦a⟧ is termed a *complex* act and in order to perform it the plan ⟦p⟧ has to be carried out.

Sample Context

The complex act `pile` and a plan for it is shown here.

SNePSUL Interaction

```
* (ASSERT ACT (BUILD ACTION (BUILD LEX ``pile'')
                   OBJECT1 (BUILD LEX ``A'') = A
                   OBJECT2 (BUILD LEX ``B'') = B
                   OBJECT3 (BUILD LEX ``C'') = C)
          PLAN (BUILD ACTION (BUILD LEX ``snsequence'') = SEQ
                    OBJECT1
                      (BUILD ACTION *SEQ
                             OBJECT1 (BUILD ACTION (BUILD LEX ``put'') = PUT
                                            OBJECT1 *C
```

```
                                                OBJECT2 *TABLE)
                                OBJECT2 (BUILD ACTION *PUT
                                               OBJECT1 *B
                                               OBJECT2 *C))
                       OBJECT2 (BUILD ACTION *PUT
                                      OBJECT1 *A
                                      OBJECT2 *B)))
(M134!
  (ACT (M127 (ACTION (M46 (LEX pile)))
             (OBJECT1 (M42 (LEX A)))
             (OBJECT2 (M52 (LEX B)))
             (OBJECT3 (M54 (LEX C)))))
  (PLAN (M133 (ACTION (M1 (LEX snsequence)))
              (OBJECT1 (M131 (ACTION (M1))
                       (OBJECT1 (M129 (ACTION (M10 (LEX put)))
                                      (OBJECT1 (M54))
                                      (OBJECT2 B1)))
                       (OBJECT2 (M130 (ACTION (M10))
                                      (OBJECT1 (M52))
                                      (OBJECT2 (M54))))))
              (OBJECT2 (M132 (ACTION (M10))
                             (OBJECT1 (M42))
                             (OBJECT2 (M52)))))))
```

Network

⟦M134⟧ is the proposition that ⟦M133⟧ is a plan for piling A om B on C (⟦M127⟧).

Alternatives

None.

Issues

See entry Acts.

Related Entries

Achieve Act, Acts, Do-all Act, Do-one Act, Effects, Conditional Act, Iteration Act, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27].

Author

Deepak Kumar, August 2, 1991

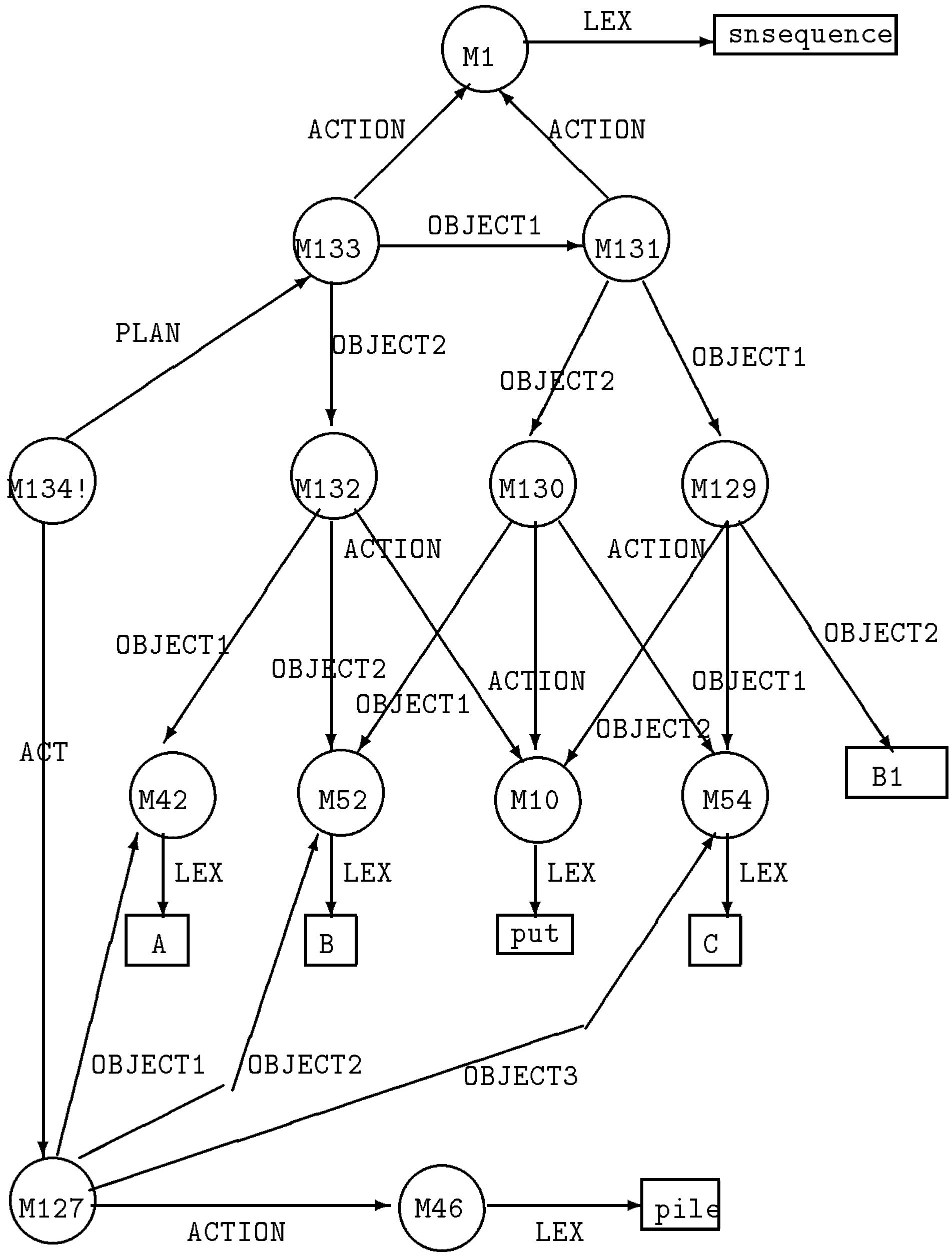

Figure 4.16: "A plan to pile block A on B on C is to first put C on the table and then put B on C and then put A on B."

4.9 Plan-Goal Proposition

That some plan is a plan for achieving some goal is called a plan-goal proposition and is represented as:

Syntax

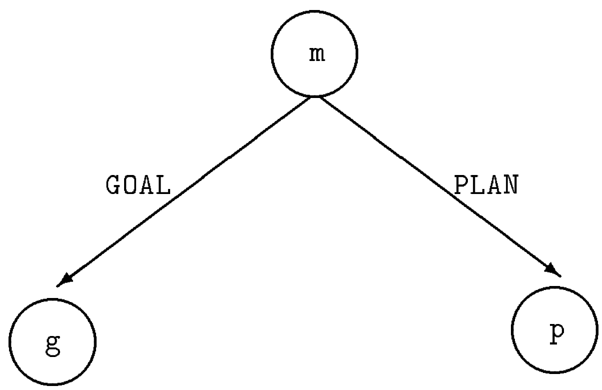

Figure 4.17: A Plan-Goal Proposition.

`m` is a structured proposition node with a `GOAL` arc to a structured proposition node `g`, and a `PLAN` arc to a structured act node `p`.

Semantics

⟦m⟧ is a proposition that ⟦p⟧ is a plan for achieving goal ⟦g⟧. The structuring syntax for plans is described in terms of control actions (See Sequence, Do-one, Do-all, Conditional, and Iterative Acts).

Sample Context

We will consider the sentence, "`A plan for achieving that A is held is to pick it up.`"

SNePSUL Interaction

```
* (ASSERT GOAL (BUILD OBJECT (BUILD LEX ``A'') = A
                      PROPERTY (BUILD LEX ``held''))
          PLAN (BUILD ACTION (BUILD LEX ``pickup'')
                      OBJECT *A))
(M166! (GOAL (M160 (OBJECT (M42 (LEX A)))
                   (PROPERTY (M33 (LEX held)))))
       (PLAN (M165 (ACTION (M13 (LEX pickup)))
                   (OBJECT (M42))))))
```

Network

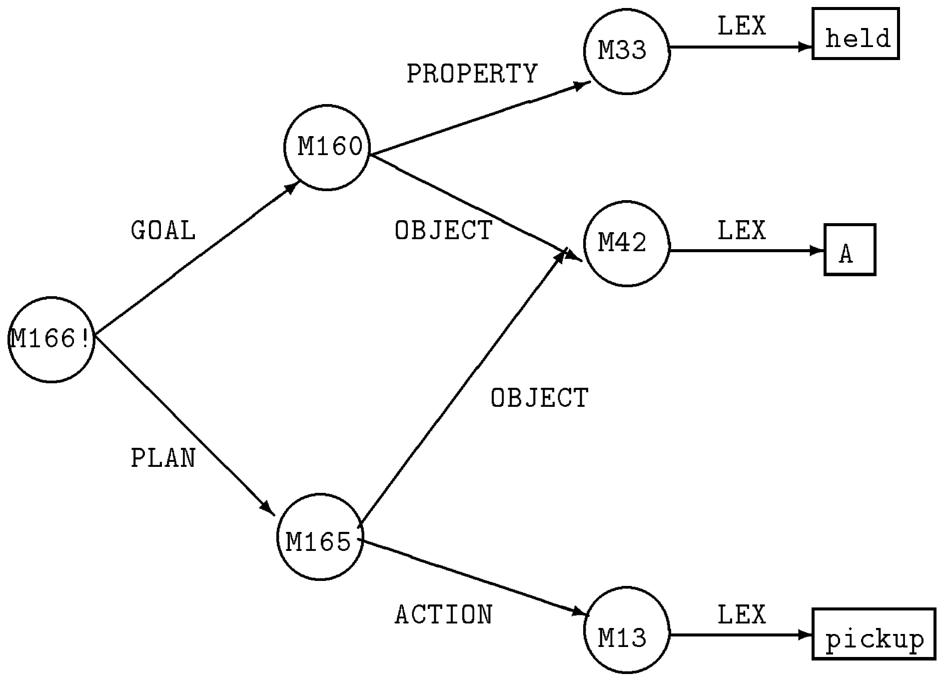

Figure 4.18: A plan for achieving that a block is held.

⟦M166⟧ is aproposition that ⟦M165⟧ is a plan for achieving goal ⟦M160⟧.

Alternatives

None.

Issues

See entries Plan-Act Proposition and Acts.

Related Entries

Achieve Act, Acts, Do-all Act, Do-one Act, Effects, Conditional Act, Iteration Act, Plan-act Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27].

Author

Deepak Kumar, August 7, 1991

4.10 Preconditions

This representation is used to specify preconditions of actions.

Syntax

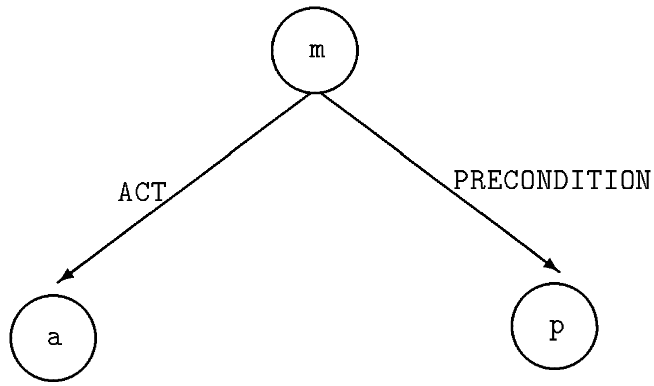

Figure 4.19: The syntax of a precondition proposition.

`m` is a structured proposition node with an `ACT` arc to a structured act node, `a`, and an `PRECONDITION` arc to a structured proposition node `p`.

Semantics

⟦m⟧ is the proposition that the precondition of act ⟦a⟧ is ⟦p⟧.

Sample Context

A representation of, "`Before picking up A make sure it is clear.`" is shown below.

SNePSUL Interaction

```
* (ASSERT ACT (BUILD ACTION (BUILD LEX ``pickup'')
                     OBJECT (BUILD LEX ``A''))
          PRECONDITION (BUILD OBJECT  (BUILD LEX ``A'')
                              PROPERTY (BUILD LEX ``clear'')))
(M167! (ACT (M165 (ACTION (M13 (LEX pickup)))
                  (OBJECT (M42 (LEX A)))))
       (PRECONDITION (M60 (OBJECT (M42))
                          (PROPERTY (M33 (LEX clear))))))
```

Network

In Figure 4.20, ⟦M167⟧ is the proposition that ⟦M60⟧ is a precondition of the act ⟦M165⟧.

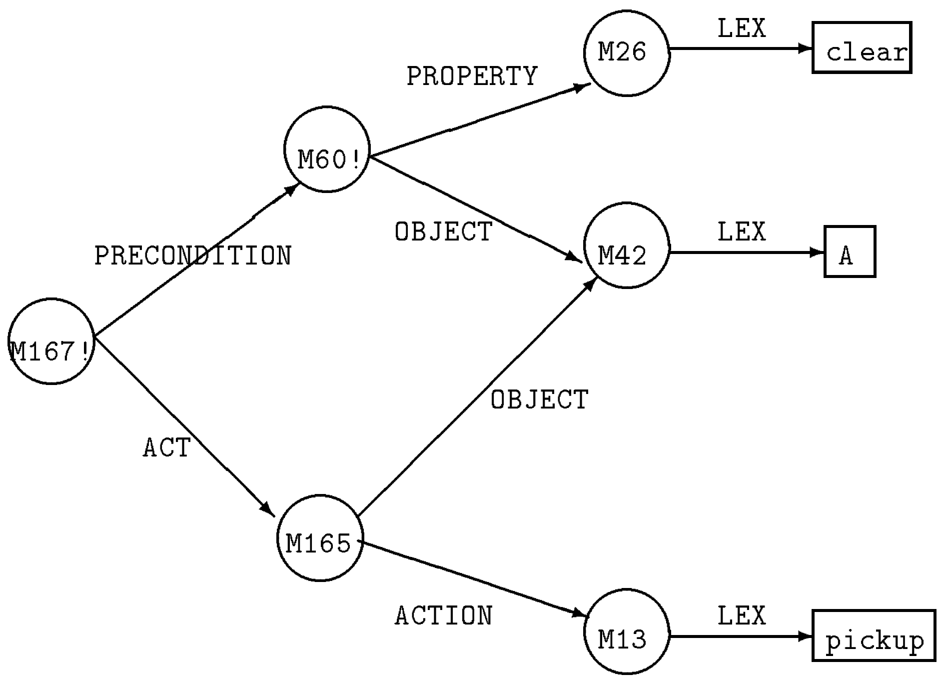

Figure 4.20: "`Before picking up A make sure it is clear.`"

Alternatives

None.

Issues

See entry Effects and Acts.

Related Entries

Achieve Act, Acts, Conditional Act, Do-all Act, Do-one Act, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions, Sequence Acts.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27].

Author

Deepak Kumar, August 2, 1991

4.11 Sequence Acts

This is a *control action* for the acting executive. See entries for Do-One, Do-All, Conditional, and Iteration Acts for descriptions of other control actions.

Syntax

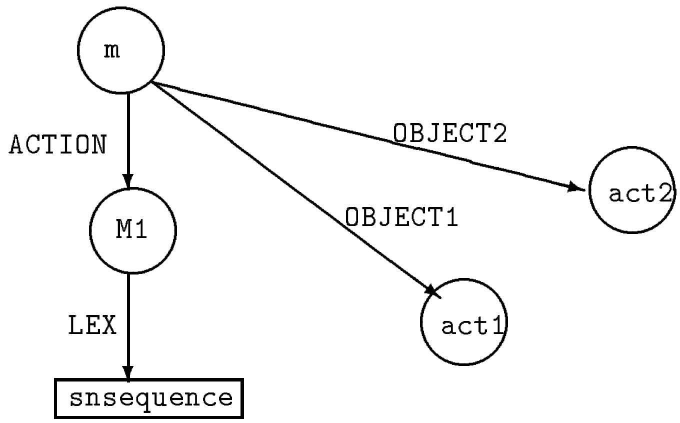

Figure 4.21:

`m` is a structured individual node with an `ACTION` arc to an individual node which is express as '`snsequence`' and `OBJECT1` and `OBJECT2` arcs to structured individual act nodes, `act1` and `act2` respectively.

Semantics

The semantics of control actions is defined operationally based on the design of the acting executive (See entry Acts). `act2` is inserted in front of the act queue, and then `act1` is inserted in front of it. ⟦`m`⟧is the act of first performing ⟦`act1`⟧and the performing ⟦`act2`⟧.

Sample Context

Representation for the act of, "**`First pick up A and then put it on B`**" is shown in figure 4.22.

SNePSUL Interaction

```
* (PERFORM ACTION (BUILD LEX "snsequence")
           OBJECT1 (BUILD ACTION (BUILD LEX "pickup")
                          OBJECT (BUILD LEX "A"))
           OBJECT2 (BUILD ACTION (BUILD LEX "put")
                          OBJECT1 (BUILD LEX "A")
                          OBJECT2 (BUILD LEX "B")))
```

```
...
(M255 (ACTION (M1 (LEX snsequence)))
      (OBJECT1 (165 (ACTION (M13 (LEX pickup)))
                    (OBJECT (M42 (LEX A)))))
      (OBJECT2 (M132 (ACTION (M10 (LEX put)))
                     (OBJECT1 (M42))
                     (OBJECT2 (M52 (LEX B))))))
```

Network

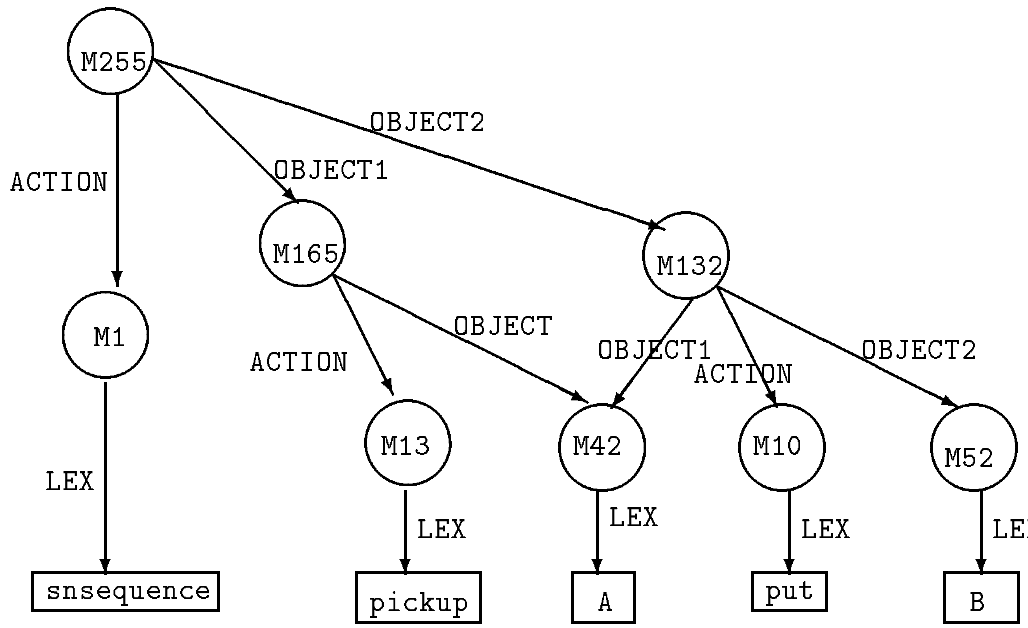

Figure 4.22: "**Pick up A and then Put it on B**."

⟦M165⟧is the act of picking (⟦M13⟧) up A (⟦M42⟧), ⟦M132⟧ is the act of putting (⟦M10⟧) A on B (⟦M52⟧), ⟦M255⟧ is the act of first doing ⟦M165⟧ and then doing ⟦M132⟧.

Alternatives

None.

Issues

It might appear that only two acts can be sequenced this way, however, since the act at the end of OBJECT2 can be another sequencing act, we have a general structure for sequencing any number of acts. Also, see entry Acts.

Related Entries

Achieve Act, Acts, Conditional Act, Do-all Act, Do-one Act, Effects, Iteration Act, Plan-act Proposition, Plan-goal Proposition, Preconditions.

References

See [6, 5, 25, 8] for a description of the underlying motivations for this representation. Examples in the domain of blocksworlds are in [27]. The general architecture of an intelligent agent using the acting executive and this representation of acts is described in [7].

Author

Deepak Kumar, July 31, 1991

Chapter 5

Case Frames for Sentences

5.1 LEXI-CAT

Syntax

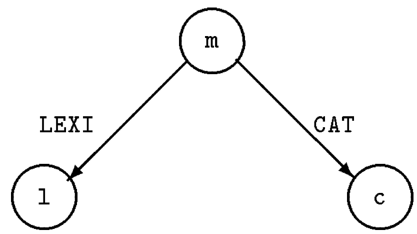

Figure 5.1: The LEXI-CAT case frame.

If `l` and `c` are individual nodes and '`m`' is an identifier not previously used, then Figure 5.1 is a network and `m` is a structured proposition node.

Semantics

⟦`m`⟧ is the proposition that lexeme "⟦`l`⟧" is a member of category ⟦`c`⟧.

Sample Context

The proposition *"Snow" is a noun* can be specified by the following SNePSUL commands:

SNePSUL Interaction

```
* (assert expressed (find cat- (build lexi "snow"
                                      cat  #x))
          expression "noun")

(M2!)
```

Network

Shown in Figure 5.2, `M1` is the proposition that the lexeme "`SNOW`" is in the category `B1`, which is expressed by the word "`NOUN`".

Alternatives

None.

Issues

The `LEXI-CAT` case frame expresses a member-class relation, specifically that between a word or lexeme and the category of which it is a member.

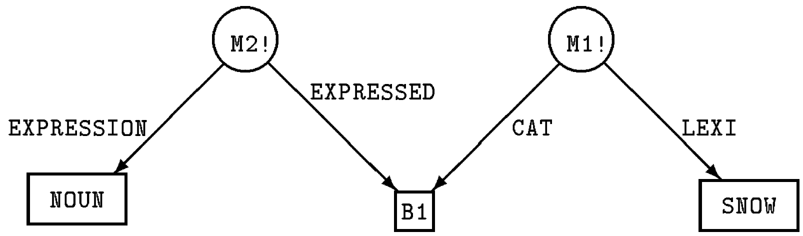

Figure 5.2: Representation of *"Snow" is a noun.*

Related Entries

See entry `MEMBER-CLASS` for a related representation.

References

See [11].

Author

Elissa L. Feit, August 9, 1991

5.2 EXPRESSED-EXPRESSION

Syntax

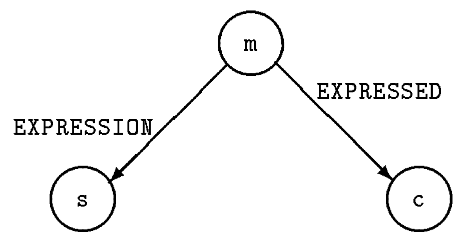

Figure 5.3: The EXPRESSED-EXPRESSION case frame.

If `s` and `c` are individual nodes and '`m`' is an identifier not previously used, then Figure 5.3 is a network and `m` is a structured proposition node.

Semantics

⟦`m`⟧ is the proposition that structure or parsed-string ⟦`s`⟧ expresses concept ⟦`c`⟧.

Sample Context

The proposition *"Snow" is a noun* can be specified by the following SNePSUL commands:

SNePSUL Interaction

```
* (assert expressed (find cat- (build lexi "snow"
                                      cat  #x))
          expression "noun")

(M2!)
```

Network

Shown in Figure 5.4, `M2` is the proposition that `B1` represents the concept expressed by the word "`NOUN`". `M1` is the proposition that the lexeme "`SNOW`" is in the category called "`NOUN`".

Alternatives

None.

Issues

None.

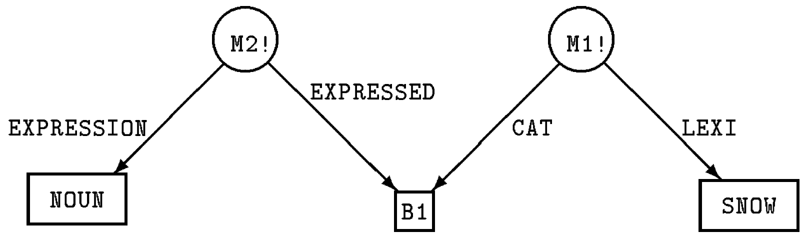

Figure 5.4: Representation of *"Snow" is a noun.*

Related Entries

None.

References

See [11].

Author

Elissa L. Feit, August 9, 1991

5.3 PRED-WORD

Syntax

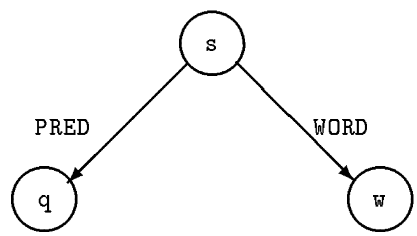

Figure 5.5: The PRED-WORD case frame.

If `q` and `w` are individual nodes and '`s`' is an identifier not previously used, then Figure 5.5 is a network and `s` is a structured individual node.

Semantics

⟦`s`⟧ is the string consisting of the word or symbol ⟦`w`⟧ concatenated to the left of initial string ⟦`q`⟧. ⟦`q`⟧ may be the null string represented by some base node.

Sample Context

The concept denoted by the string *"Snow is white"* can be specified within a SNePS network created by the following SNePSUL commands:

SNePSUL Interaction

```
* ... (build beg (build pred #nil
                       word "snow")
             end (build pred (build pred (find pred *nil)
                                   word "is")
                       word "white")  )
      ...
```

Network

Shown in Figure 5.6, ⟦`M5`⟧ is the proposition that the concept ⟦`B2`⟧ is expressed by the bounded string ⟦`M4`⟧. ⟦`B1`⟧ is (presumably) the empty string. ⟦`M2`⟧ is the string "`snow`", ⟦`M3`⟧ is the string "`snow is`", and ⟦`M4`⟧ is the string "`snow is white`".

Alternatives

None.

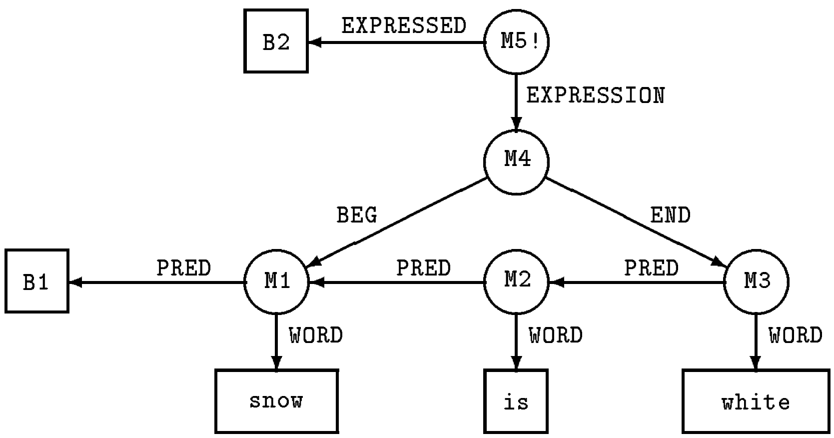

Figure 5.6: A representation of the bounded string "*snow is white*"

Issues

The `pred-word` case frame is used to build ordered networks, as opposed to most SNePS networks which are unordered.

Related Entries

None.

References

See [11].

Author

Elissa L. Feit, September 25, 1991

5.4 BEG-END

Syntax

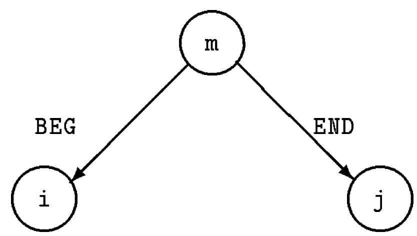

Figure 5.7: The BEG-END case frame.

If `i` and `j` are individual nodes and '`m`' is an identifier not previously used, then Figure 5.7 is a network and `m` is a structured individual node.

Semantics

Bounded string ⟦`m`⟧ is the surface string beginning with the last word of initial string ⟦`i`⟧ and ending with the last word of initial string ⟦`j`⟧, where ⟦`i`⟧ precedes ⟦`j`⟧ or ⟦`i`⟧ is the same initial string as ⟦`j`⟧.

Sample Context

The concept denoted by the string *"Snow is white"* can be specified within a SNePS network created by the following SNePSUL commands:

SNePSUL Interaction

```
* (assert expression (build beg (build pred #nil
                                       word "snow")
                            end (build pred (build pred (find pred *nil)
                                                    word "is")
                                       word "white")  )
          expressed #b)

(M5!)
```

Network

Shown in Figure 5.8, ⟦`M5`⟧ is the proposition that the concept ⟦`B2`⟧ is expressed by the bounded string ⟦`M4`⟧. ⟦`B1`⟧ is (presumably) the empty string. ⟦`M2`⟧ is the string "`snow`", ⟦`M3`⟧ is the string "`snow is`", and ⟦`M4`⟧ is the string "`snow is white`".

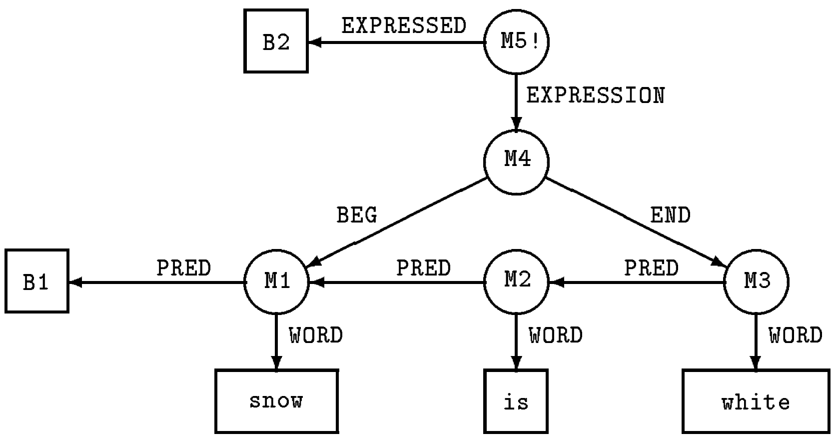

Figure 5.8: A representation of the bounded string "*snow is white*"

Alternatives

None.

Issues

The `BEG-END` case frame is used to build ordered networks, as opposed to most SNePS networks which are unordered.

Related Entries

None.

References

See [11].

Author

Elissa L. Feit, September 25, 1991

5.5 BSTR-CAT-STRC

Syntax

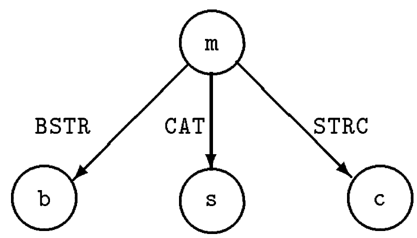

Figure 5.9: The BSTR-CAT-STRC case frame.

If `b`, `c` and `s` are individual nodes and '`m`' is an identifier not previously used, then Figure 5.9 is a network and `m` is a structured individual node.

Semantics

[[`m`]] is the concept that the bounded string [[`b`]] is in category [[`c`]] and that [[`s`]] is the structure or parse of [[`b`]].

Sample Context

`M13` is the proposition that the bounded string `M7`, "a student", is an indefinite noun phrase which constitutes the bounded strings `M8`, "a", an indefinite determiner, and `M10`, "student", a noun.

SNePSUL Interaction

```
* (assert expression "indef-det"
          expressed (find cat- (assert lexi "a" cat #x)))

(M2!)

*  (assert expression "noun"
          expressed (find cat- (assert lexi "student" cat #x)))

(M4!)

* (assert beg (assert pred #nil  word "a")
          end (assert pred (find word "a")
                      word "student"))

(M7!)
```

```
* (assert cat (find cat- (find lexi "a"))
          bstr (assert beg (find word "a")
                       end (find word "a"))
          strc #x)

(M9!)

* (assert cat (find cat- (find lexi "student"))
          bstr (assert beg (find word "student")
                       end (find word "student"))
          strc #x)

(M11!)

* (assert constit (find strc- m11)
          constitof (find strc-
                     (assert strc #x
                             cat (find expressed-
                                  (assert expressed #y
                                          expression "indef-NP"))
                             bstr m7)))


(M14!)

* (assert constit (find strc- m9)
          constitof (find strc- (find bstr m7)))

(M15!)
```

Network

Alternatives

None.

Issues

None.

Related Entries

None.

References

See [11].

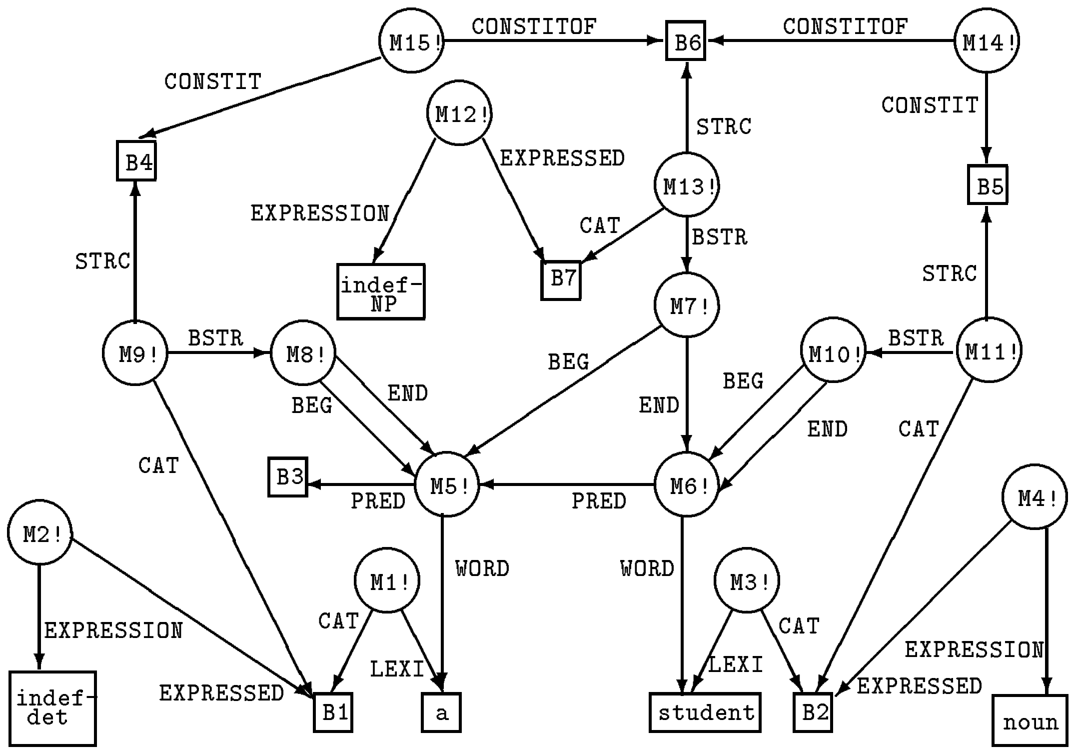

Figure 5.10: A representation of the structure of the indefinite NP "a student".

Author

Elissa L. Feit, September 25, 1991

5.6 CONSTIT-CONSTITOF

Syntax

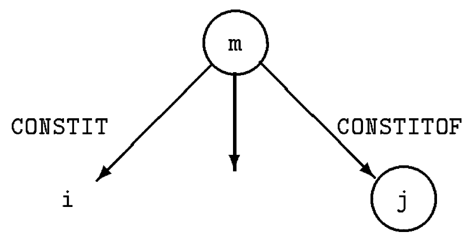

Figure 5.11: The CONSTIT-CONSTITOF case frame.

If `i` and `j` are individual nodes and '`m`' is an identifier not previously used, then Figure 5.11 is a network and `m` is a structured individual node.

Semantics

⟦`m`⟧ is the concept that the structure ⟦`i`⟧ is a constituent of structure ⟦`j`⟧.

Sample Context

`M13` is the proposition that the bounded string `M7`, "a student", is an indefinite noun phrase which constitutes the bounded strings `M8`, "a", an indefinite determiner, and `M10`, "student", a noun.

SNePSUL Interaction

```
* (assert expression "indef-det"
          expressed (find cat- (assert lexi "a" cat #x)))

(M2!)

*  (assert expression "noun"
          expressed (find cat- (assert lexi "student" cat #x)))

(M4!)

* (assert beg (assert pred #nil  word "a")
          end (assert pred (find word "a")
                      word "student"))

(M7!)
```

```
* (assert cat (find cat- (find lexi "a"))
          bstr (assert beg (find word "a")
                       end (find word "a"))
          strc #x)

(M9!)

* (assert cat (find cat- (find lexi "student"))
          bstr (assert beg (find word "student")
                       end (find word "student"))
          strc #x)

(M11!)

* (assert constit (find strc- m11)
          constitof (find strc-
                      (assert strc #x
                              cat (find expressed-
                                   (assert expressed #y
                                           expression "indef-NP"))
                              bstr m7)))


(M14!)

* (assert constit (find strc- m9)
          constitof (find strc- (find bstr m7)))

(M15!)
```

Network

Alternatives

None.

Issues

None.

Related Entries

None.

References

See [11].

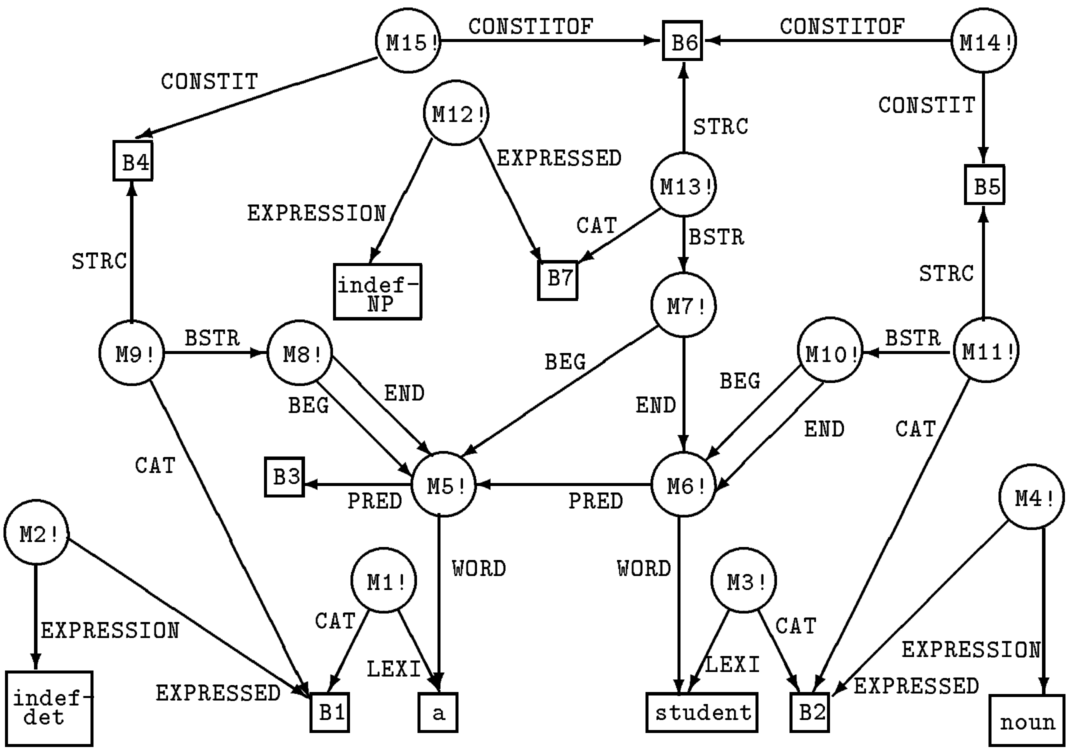

Figure 5.12: A representation of the structure of the indefinite NP "a student".

Author

Elissa L. Feit, September 25, 1991

Chapter 6

Case Frames from Albert H. Yuhan

6.1 Group

Syntax

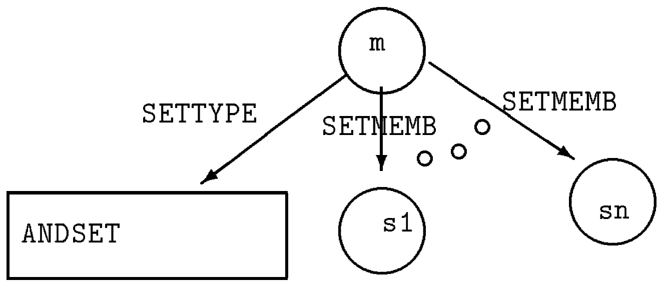

Figure 6.1:

`m` is a structured individual node with `SETMEMB` arcs to the individual nodes `s1`, ..., `sn`, and a `SETTYPE` arc to the constant base node `ANDSET`.

Semantics

⟦`m`⟧ is the concept of a group whose members are ⟦`s1`⟧, ... ,⟦`sn`⟧.

Sample Context

Tom and Mary are playing tennis in the park.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (BUILD AGENT
                       (BUILD SETMEMB #B1
                              SETMEMB #B2
                              SETTYPE ANDSET)
                     ACT
                       (BUILD ACTION  (BUILD LEX "play")
                        OBJECT1 (BUILD LEX "tennis")))
          EPLACE #B3
          ETIME  #B4)
...
(M6!
  (EAFFAIR
    (M5 (ACT (M4 (ACTION (M2 (LEX play)))
                 (OBJECT1 (M3 (LEX tennis)))))
        (AGENT (M1 (SETMEMB B1 B2)
                   (SETTYPE ANDSET)))))
  (EPLACE B3)
```

```
  (ETIME B4))


* (ASSERT ARG1 *B3
          ARG2 #B5
          SR   (BUILD LEX "in"))
...
(M8! (ARG1 B3)
     (ARG2 B5)
     (SR (M7 (LEX in))))

* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "park"))
...
(M10! (CLASS (M9 (LEX park)))
      (MEMBER B5))


* (ASSERT OBJECT     *B1
          PROPERNAME (BUILD LEX "Tom"))
...
(M12! (OBJECT B1)
      (PROPERNAME (M11 (LEX Tom))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "Mary"))
...
(M14! (OBJECT B2)
      (PROPERNAME (M13 (LEX Mary))))
```

Network

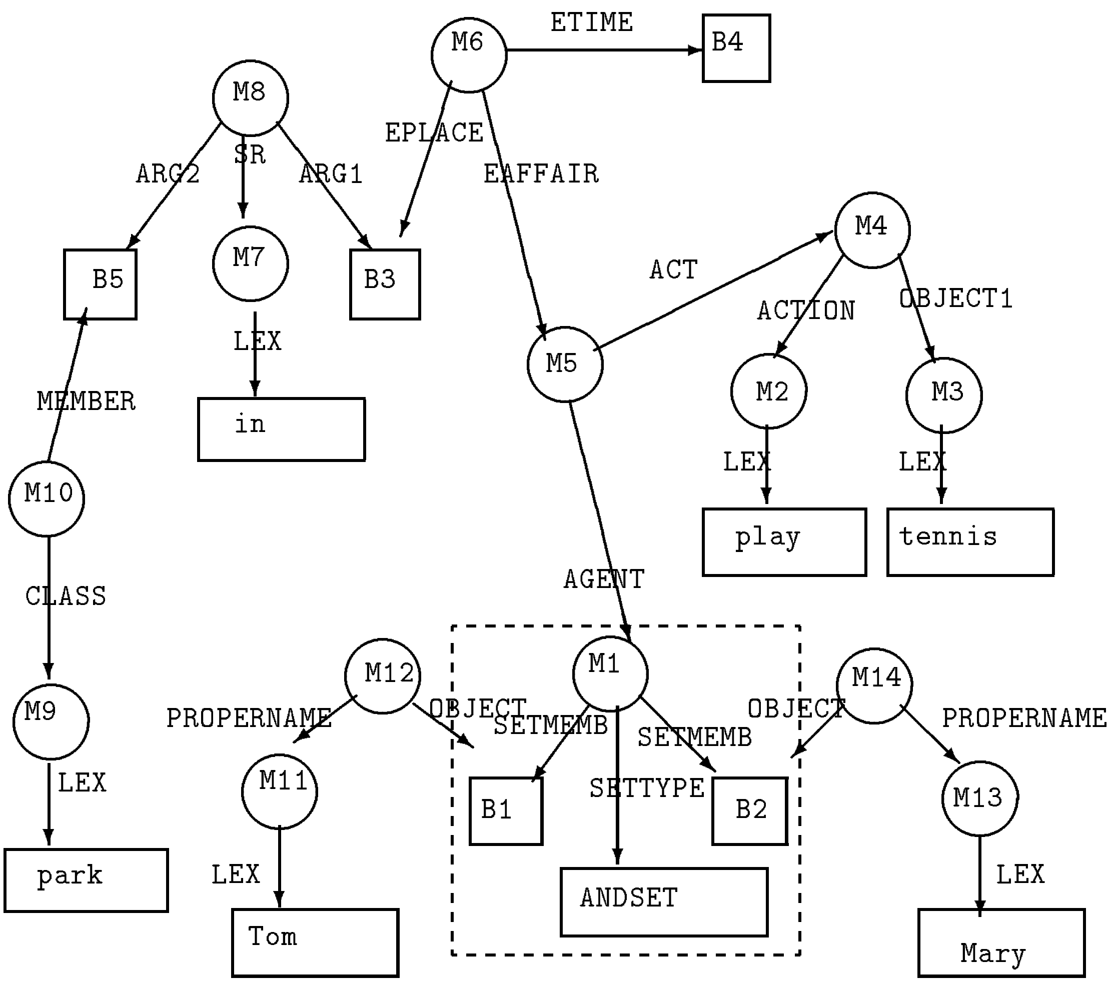

Figure 6.2: "`Tom and Mary are playing tennis in the park.`"

In figure 6.2, ⟦`M6`⟧ is the proposition that the event affair (⟦`M5`⟧) of the group Tom and Mary (⟦`M1`⟧) playing tennis (⟦`M4`⟧) in the park (⟦`B3`⟧) takes place during a time ⟦`B4`⟧.

Alternatives

None.

Issues

This case frame is motivated by the fact that sentences like the above expanded to a logical full form are not equivalent, that is, "Tom is playing tennis and Mary is playing tennis" $\neq$

"Tom and Mary are playing tennis". Hence, the group, Tom and Mary, are represented as a collective, intensional entity. An *or* conjunction over individuals also constitutes a group in which each individual complements the others.

Related Entries

None.

References

See [31] for further discussion of this representation.

Author

Susan Haller, August 1991

6.2 Part-Whole

Syntax

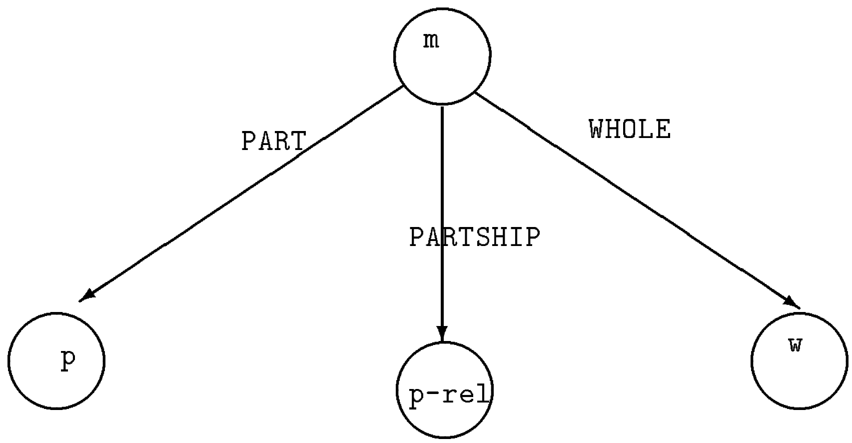

Figure 6.3:

`m` is a structured individual node with a `PART` arc to the individual node `p`, a `PARTSHIP` arc to the inividual node `p-rel`, and a `WHOLE` arc to the individual node `w`.

Semantics

⟦`m`⟧ is the proposition that ⟦`p`⟧ is the ⟦`p-rel`⟧ part of ⟦`w`⟧.

Sample Context

Mary is at the front of the hall.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
           (BUILD  SFIG     #B1
                   LOCATIVE #B2
                   SPRED    BE-LOC)
         ETIME #B3)
...
(M2! (EAFFAIR (M1 (LOCATIVE B2)
                  (SFIG B1)
                  (SPRED BE-LOC)))
     (ETIME B3))
```

```
* (ASSERT OBJECT *B1
          PROPERNAME (BUILD LEX "Mary"))
...
(M4! (OBJECT B1)
     (PROPERNAME (M3 (LEX Mary))))


* (ASSERT ARG1 *B2
          ARG2 #B4
          SR   (BUILD LEX "ideal-point"))
...
(M6! (ARG1 B2)
     (ARG2 B4)
     (SR (M5 (LEX ideal-point))))


* (ASSERT PART      *B4
          PARTSHIP  (BUILD LEX "front")
          WHOLE     #B5)
...
(M8! (PART B4)
     (PARTSHIP (M7 (LEX front)))
     (WHOLE B5))

* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "hall"))
...
(M10! (CLASS (M9 (LEX hall)))
      (MEMBER B5))
```

Network

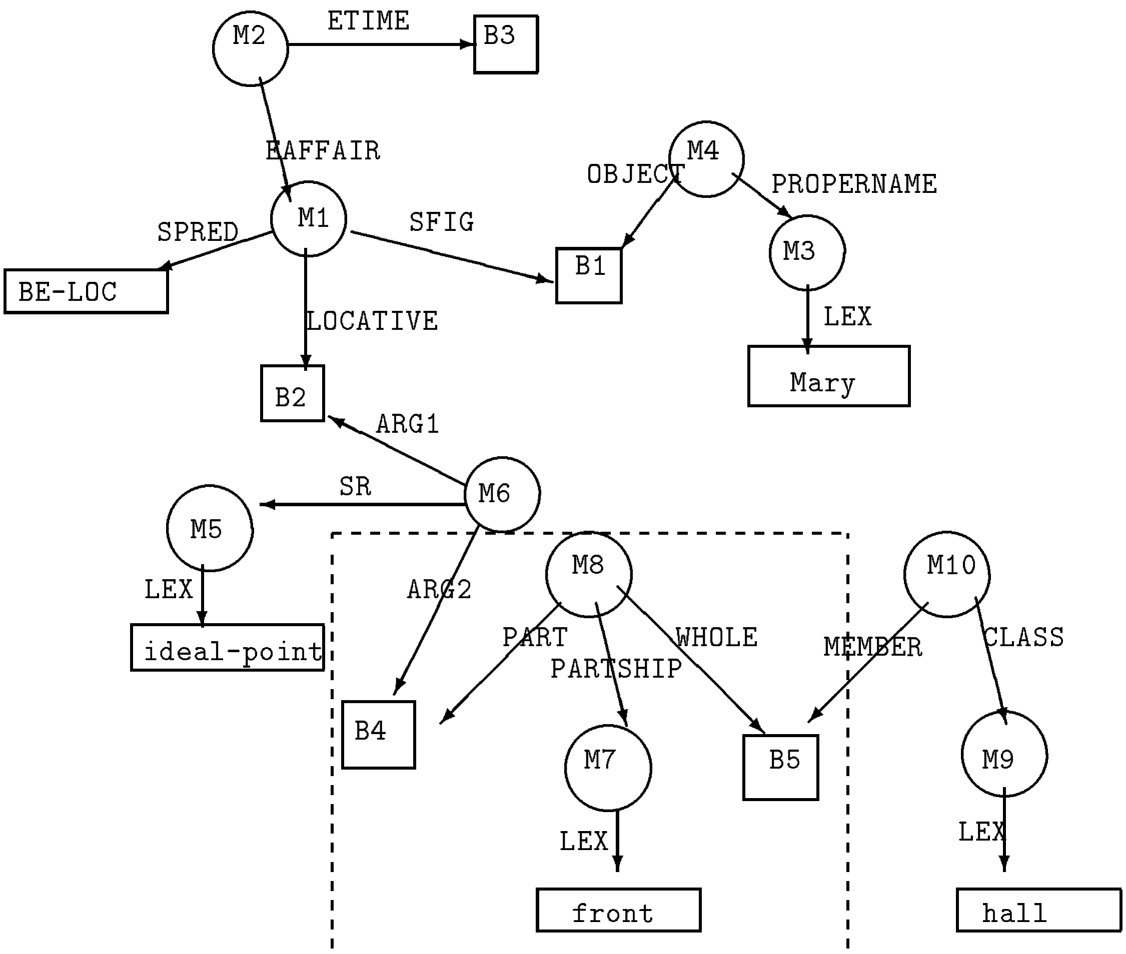

Figure 6.4: "`Mary is at the front of the hall.`"

In figure 6.4, ⟦`M2`⟧ is the proposition that the event affair (⟦`M1`⟧) of Mary (⟦`B1`⟧) being located at a place (⟦`B2`⟧) which is the front of the hall (⟦`B4`⟧) takes place during a time ⟦`B3`⟧.

Alternatives

An alternative approach is the REL-POSSESSOR case frame. This treatment views part-whole relationships as possessives. Another alternative is the part hierarchy used to express graghical deep knowledge [3]. This is a richer representation for parts that allows the system to reason about three different types of part-whole relationships.

Issues

One use of possessives in natural language is to identify objects that are difficult to reference by relating them to a certain whole of which they are a part. For example, *Tom's nose*, could be represented by this case frame.

Related Entries

Rel-Possessor, Part Hierarchy

References

See [31] for further discussion of this representation.

Author

Susan Haller, August 1991

6.3 Kinship

Syntax

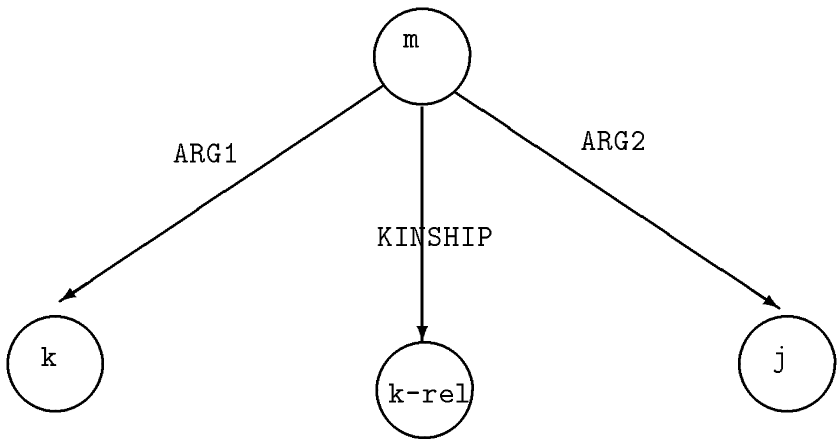

Figure 6.5:

`m` is a structured individual node with a **`ARG1`** arc to the individual node `k`, a **`KINSHIP`** arc to the inividual node `r-rel`, and a **`ARG2`** arc to the individual node `j`.

Semantics

⟦`m`⟧ is the proposition that ⟦`k`⟧ is ⟦`j`⟧'s ⟦`r-rel`⟧.

Sample Context

Bill is Tom's uncle.

SNePSUL Interaction

```
* (ASSERT ARG1 #B1
          ARG2 #B2
          KINSHIP (BUILD LEX "uncle"))
...
(M2! (ARG1 B1)
     (ARG2 B2)
     (KINSHIP (M1 (LEX uncle))))

* (ASSERT OBJECT     *B1
          PROPERNAME (BUILD LEX "Bill"))
...
(M4! (OBJECT B1)
```

```
     (PROPERNAME (M3 (LEX Bill))))

* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "Tom"))
...
(M6! (OBJECT B2)
     (PROPERNAME (M5 (LEX Tom))))
```

Network

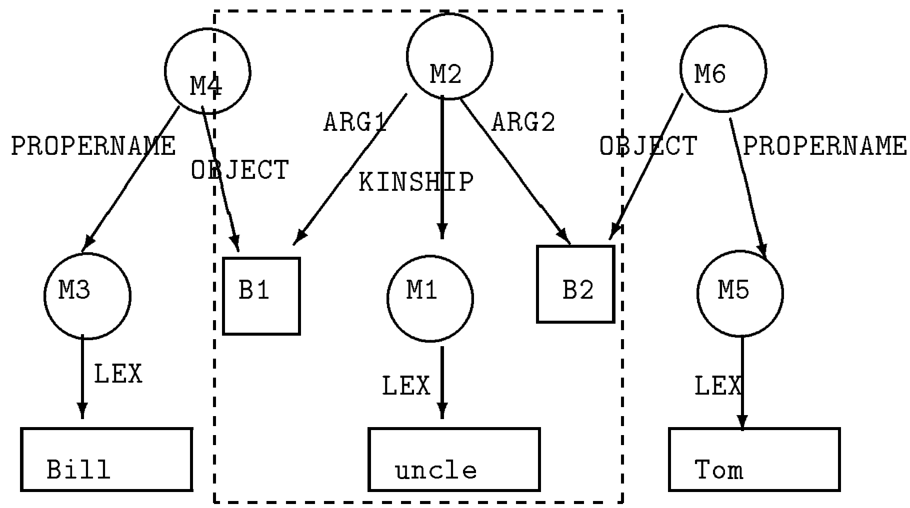

Figure 6.6: "`Bill is Tom's uncle.`"

In figure 6.6, ⟦`M2`⟧ is the proposition that Bill (⟦`B1`⟧) is the uncle of Tom (⟦`B2`⟧).

Alternatives

An alternative approach is the REL-POSSESSOR case frame. This treatment views family relationships as possessives.

Issues

Related Entries

Rel-Possessor

References

See [31] for further discussion of this representation.

Author

Susan Haller, August 1991

6.4 Purpose Relation Between Events

Syntax

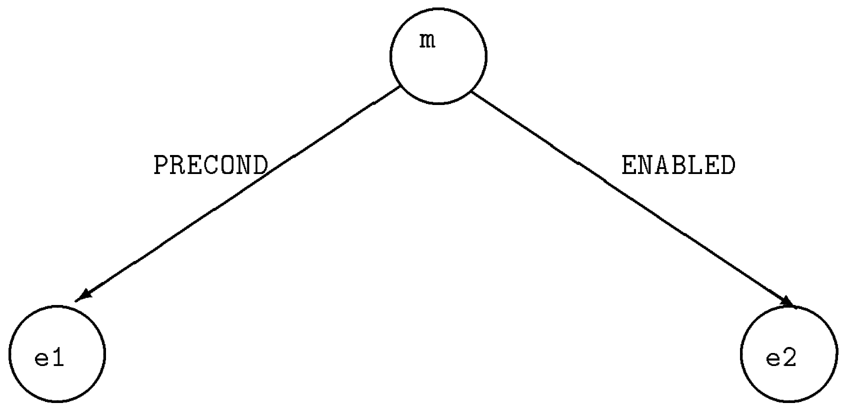

Figure 6.7:

m is a structured individual node with a PRECOND arc to the individual node e1 and an ENABLED arc to the inividual node e2.

Semantics

⟦m⟧ is the proposition that ⟦e1⟧ took place in order for ⟦e2⟧ to take place.

Sample Context

John went to the store in order buy milk.

SNePSUL Interaction

```
* (ASSERT PRECOND
            (ASSERT EAFFAIR
                      (BUILD SPRED MOVE
                             SFIG  #B1
                             GOAL  #B2)
                    EPLACE #B3
                    ETIME  #B4)
          ENABLED
            (ASSERT EAFFAIR
                      (BUILD ACT
                               (BUILD ACTION
                                        (BUILD LEX "buy")
                                      OBJECT1
                                        (BUILD LEX "milk"))
```

```
                              AGENT *B1)
                     EPLACE *B2
                     ETIME  #B5))
...
(M8!
  (ENABLED
    (M7!
      (EAFFAIR
        (M6 (ACT (M5 (ACTION (M3 (LEX buy)))
                     (OBJECT1 (M4 (LEX milk)))))
            (AGENT B1)))
      (EPLACE B2)
      (ETIME B5)))
  (PRECOND
    (M2! (EAFFAIR (M1 (GOAL B2)
                      (SFIG B1)
                      (SPRED MOVE)))
         (EPLACE B3)
         (ETIME B4))))

* (ASSERT BEFORE   *B4
          AFTER    *B5
          DURATION #B6)
...
(M9! (AFTER B5)
     (BEFORE B4)
     (DURATION B6))


* (ASSERT ARG1 *B2
          ARG2 #B7
          SR   (BUILD LEX "ideal point"))
...
(M11! (ARG1 B2)
      (ARG2 B7)
      (SR (M10 (LEX ideal point))))


* (ASSERT OBJECT     *B1
          PROPERNAME (BUILD LEX "John"))
...
(M13! (OBJECT B1)
      (PROPERNAME (M12 (LEX John))))


* (ASSERT MEMBER *B7
          CLASS  (BUILD LEX "store"))
...
```

```
(M15! (CLASS (M14 (LEX store)))
      (MEMBER B7))
```

Network

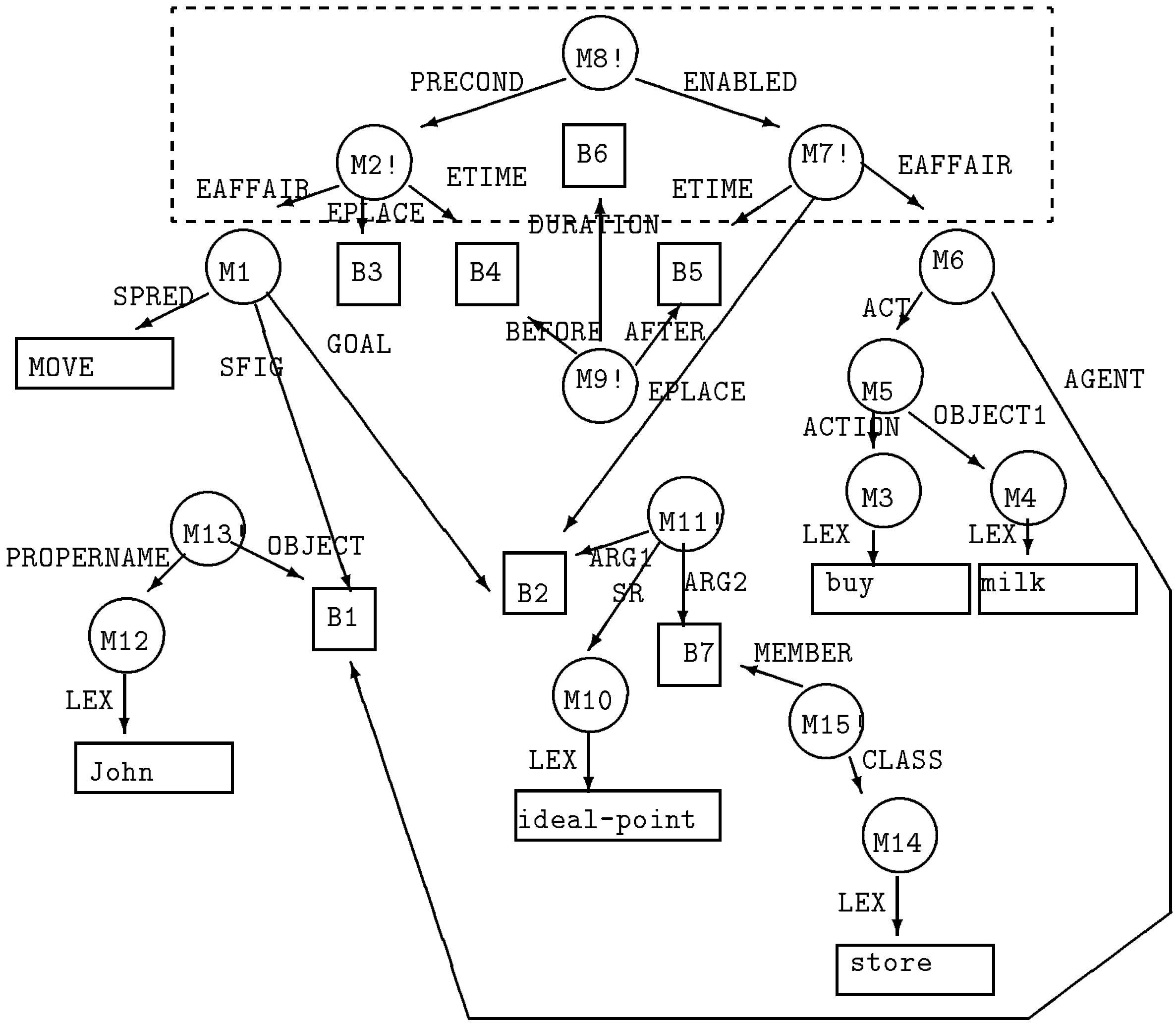

Figure 6.8: "`John went to the store in order to buy milk.`"

In figure 6.8, [[M8]] is the proposition that the event of John going to the store ([[M2]]) was done in order for the event of John buying milk ([[M7]]) to occur.

Alternatives

Issues

One theory of discourse coherence holds that there are predicate relations between sentence clauses that are recognized by the reader/listener and used by him to analyze the current sentence constituent. Some examples of the predicates are solutionhood, causality, justification, and, of course, enablement. The proposition expressed as ⟦M8⟧ in our example should not be confused with any of the plan representations. Rather, ⟦M8⟧ expresses a system belief that the occurrence of ⟦M2⟧ will enable the event ⟦M7⟧. It may be the case that the former event is not sufficient or even necessary for the latter event to occur. For instance, with respect to our example, it may be the case that John goes to the store but doesn't have enough money (not sufficient), or it could be that John could have had milk delivered (not necessary).

The system designed by Yuhan uses this representation to store information about contextual goals that is recognizes from syntactic clues during analysis. With respect to his work, this information may be needed later in the discourse analysis to resolve a spatial reference frame.

Related Entries

References

See [31] for further discussion of this representation. [9] provides a complete analysis of discourse coherence in terms of propositional relations between subsequent clauses.

Author

Susan Haller, August 1991

6.5 Events

Syntax

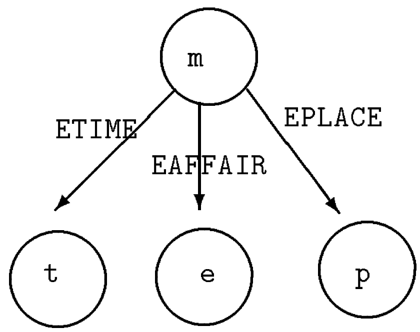

Figure 6.9:

m is a structured proposition node with an ETIME arc to a base node t, an EAFFAIR arc to a structured node e, and an EPLACE arc to a base node p.

Semantics

⟦m⟧ is the proposition that event ⟦e⟧ takes place at time ⟦t⟧ at place ⟦p⟧.

Sample Context

John went to the store in Buffalo.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
    (BUILD SPRED MOVE
   SFIG  #B1
   GOAL  #B2)
  EPLACE #B3
  ETIME  #B4)
...
(M2! (EAFFAIR (M1 (GOAL B2)
                  (SFIG B1)
                  (SPRED MOVE)))
     (EPLACE B3)
     (ETIME B4))


* (ASSERT BEFORE   *B4
  AFTER    #B5
```

```
  DURATION #B6)
...
(M3! (AFTER B5)
     (BEFORE B4)
     (DURATION B6))


* (ASSERT OBJECT *B1
  PROPERNAME (BUILD LEX "John"))
...
(M5! (OBJECT B1)
     (PROPERNAME (M4 (LEX John))))


* (ASSERT SR   (BUILD LEX "ideal point")
  ARG1 *B2
  ARG2 #B7)
...
(M7! (ARG1 B2)
     (ARG2 B7)
     (SR (M6 (LEX ideal point))))


* (ASSERT MEMBER *B7
  CLASS  (BUILD LEX "store"))
...
(M9! (CLASS (M8 (LEX store)))
     (MEMBER B7))


* (ASSERT ARG1 *B3
  SR   (BUILD LEX "in")
  ARG2 #B8)

(M11! (ARG1 B3)
      (ARG2 B8)
      (SR (M10 (LEX in))))

* (ASSERT OBJECT     *B8
  PROPERNAME (BUILD LEX "Buffalo"))
...
(M13! (OBJECT B8)
      (PROPERNAME (M12 (LEX Buffalo))))
```

Network

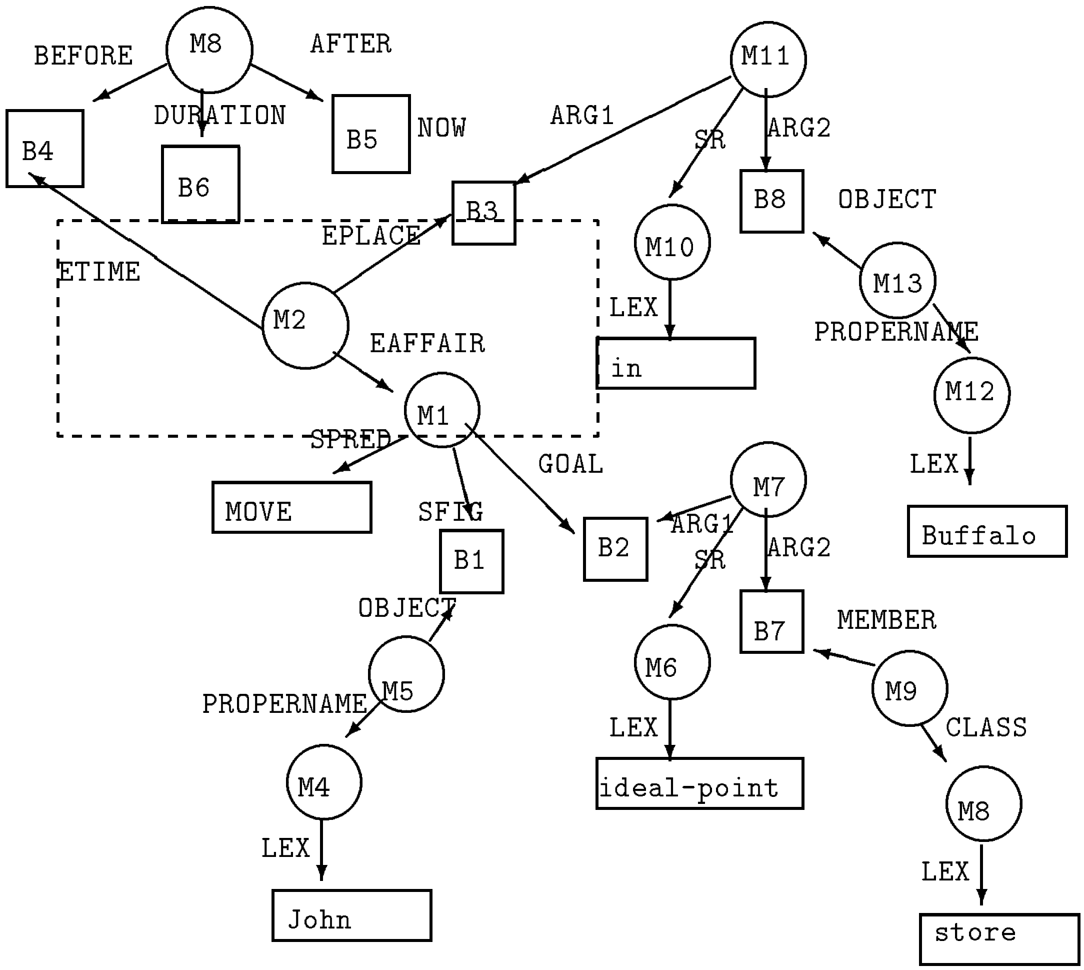

Figure 6.10: "`John went to the store in Buffalo.`"

In figure 6.10, ⟦M2⟧ is the proposition that the event of John going to the store (⟦M1⟧) to place during a certain time (⟦B4⟧) at a place in Buffalo (⟦B3⟧).

Alternatives

See Almeida's representation of the same.

Issues

An event is composed of an event affair (the action and arguments to that action), a time, and a place. The event proposition is an intensional entity, however, the base nodes corresponding to time and place connect the network to an external "when" and "where" that the event is known to have taken place. But, these base nodes can be associated with hypothetical events in which case, these nodes represent a hypothetical time and place. (Why not just say they are the system's concept of the time and place?)

References

See [31] for further discussion of this representation.

Author

Susan Haller, August 1991

6.6 Spatial Event Affair - Motion

Syntax

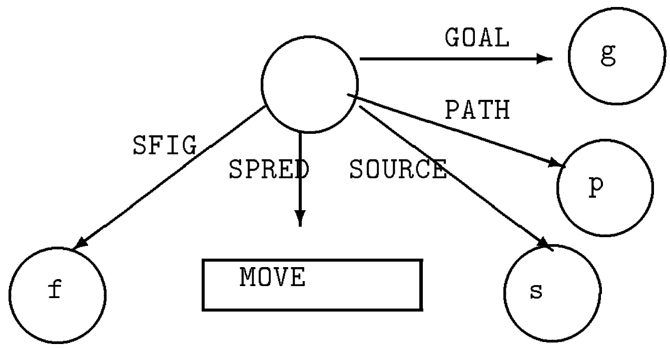

Figure 6.11:

`m` is a structured individual node with an `SFIG` arc to the individual node `f` and an `SPRED` arc to the internal constant base node `MOVE`, and any or all of the arcs `SOURCE, PATH, GOAL` to the individual nodes `s, p, g` respectively.

Semantics

⟦`m`⟧ is the proposition that an event involving the motion (⟦`MOVE`⟧) of ⟦`f`⟧ takes place from ⟦`s`⟧ to ⟦`g`⟧ along ⟦`p`⟧.

Sample Context

John went to the store in Buffalo.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
    (BUILD SPRED MOVE
   SFIG  #B1
   GOAL  #B2)
  EPLACE #B3
  ETIME  #B4)
...
(M2! (EAFFAIR (M1 (GOAL B2)
                  (SFIG B1)
                  (SPRED MOVE)))
     (EPLACE B3)
     (ETIME B4))
```

```
* (ASSERT BEFORE   *B4
  AFTER    #B5
  DURATION #B6)
...
(M3! (AFTER B5)
     (BEFORE B4)
     (DURATION B6))


* (ASSERT OBJECT *B1
  PROPERNAME (BUILD LEX "John"))
...
(M5! (OBJECT B1)
     (PROPERNAME (M4 (LEX John))))


* (ASSERT SR   (BUILD LEX "ideal point")
  ARG1 *B2
  ARG2 #B7)
...
(M7! (ARG1 B2)
     (ARG2 B7)
     (SR (M6 (LEX ideal point))))


* (ASSERT MEMBER *B7
  CLASS  (BUILD LEX "store"))
...
(M9! (CLASS (M8 (LEX store)))
     (MEMBER B7))


* (ASSERT ARG1 *B3
  SR   (BUILD LEX "in")
  ARG2 #B8)

(M11! (ARG1 B3)
      (ARG2 B8)
      (SR (M10 (LEX in))))

* (ASSERT OBJECT     *B8
  PROPERNAME (BUILD LEX "Buffalo"))
...
(M13! (OBJECT B8)
      (PROPERNAME (M12 (LEX Buffalo))))
```

Network

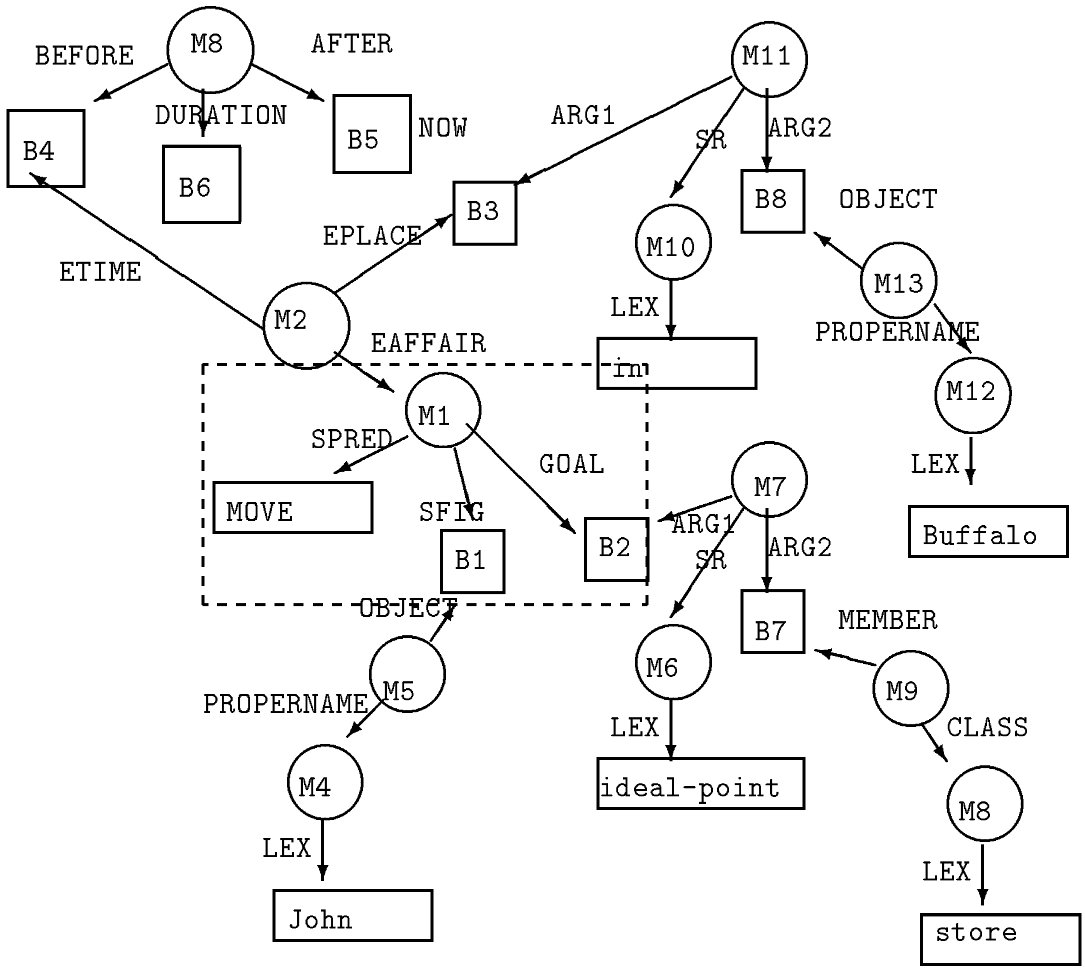

Figure 6.12: "`John went to the store in Buffalo.`"

In figure 6.12, ⟦`M1`⟧ is the proposition that that John (⟦`B1`⟧) moves towards a place, ⟦`B2`⟧, which is at the store (⟦`M7`⟧).

Alternatives

None.

Issues

Non-spatial event affairs are event affairs that do not have a place in their valence structure, that is, they do not require a place as an argument to them. As the event affair to an event, they can have a place (example: "Tom played poker in the hotel room"), however, the event place is not an essential argument to the event itself. For example, "Tom played poker" makes sense. One subset of event affairs is itself spatial. Spatial event affairs are events that involve a figure object's motion, station, or orientation as represented by a spatial deep case predicate `MOVE`, `BE-LOC` or `BE-ORIENTED` respectively. As presented here, motion event affairs can have have a event time and place separate from, and in addition to, places representing the source, path, and goal point of the motion.

John is walking *from the drinking fountain to the baseball diamond in the park.*

In this case, the source, path and goal locations are arguments to the spatial event affair, `MOVE`, and they are distinct from the place (`EPLACE`) for the entire event, the park. Spatial event affairs that involve the figure object being situated (`BE-LOC`) involve a place as one of their constituent arguments.

John stayed *at home.*
John was upside down *in bed.*

As a result, at the event level of structure, the `EPLACE` arc is not used when the event affair specified under `EAFFAIR` is a spatial event affair predicated by a spatial deep case `BE-LOC`. A spatial orientation event affair involves the figure object being oriented towards a goal. This is reflected by the goal argument to the spatial event, `BE-ORIENTED`. The spatial event affair of the figure being oriented has a distinct and separate place of occurrence, hence at the event-level of structure, there is an `EPLACE` arc. To summarize, between one and three places (`SOURCE`, `PATH`, `GOAL`) can be specified for spatial event affairs predicated by `MOVE`. The spatial event affairs predicated by a `BE-LOC` will always be associated with a place argument specified by the `LOCATIVE`. In this case, there is no `EPLACE` arc used at the event level. The spatial predicate `BE-ORIENTED` will have a `GOAL` arc indicating the direction of orientation.

Related Entries

Spatial Event Affair - Station, Spatial Event Affair - Orientation, Events

References

See [31] for further discussion of this representation. See [29] for a discussion of spatial deep cases, an analysis on which this representation is based.

Author

Susan Haller, August 1991

6.7 Spatial Event Affair - Station

Syntax

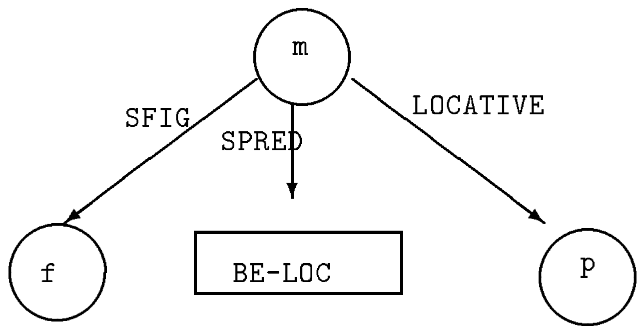

Figure 6.13:

`m` is a structured individual node with an `SFIG` arc to the individual node `f` and an `SPRED` arc to the internal constant base node `BE-LOC`, and the arc `LOCATIVE` to the individual nodes `p`.

Semantics

⟦`m`⟧ is the proposition that an event involving the station (⟦`BE-LOC`⟧) of ⟦`f`⟧ takes place at ⟦`p`⟧.

Sample Context

Mary is at the front of the hall.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (BUILD  SFIG     #B1
                    LOCATIVE #B2
                    SPRED    BE-LOC)
          ETIME #B3)
...
(M2! (EAFFAIR (M1 (LOCATIVE B2)
                  (SFIG B1)
                  (SPRED BE-LOC)))
     (ETIME B3))


* (ASSERT OBJECT *B1
          PROPERNAME (BUILD LEX "Mary"))
```

```
...
(M4! (OBJECT B1)
     (PROPERNAME (M3 (LEX Mary))))


* (ASSERT ARG1 *B2
          ARG2 #B4
          SR   (BUILD LEX "ideal-point"))
...
(M6! (ARG1 B2)
     (ARG2 B4)
     (SR (M5 (LEX ideal-point))))


* (ASSERT PART      *B4
          PARTSHIP  (BUILD LEX "front")
          WHOLE     #B5)
...
(M8! (PART B4)
     (PARTSHIP (M7 (LEX front)))
     (WHOLE B5))

* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "hall"))
...
(M10! (CLASS (M9 (LEX hall)))
      (MEMBER B5))
```

Network

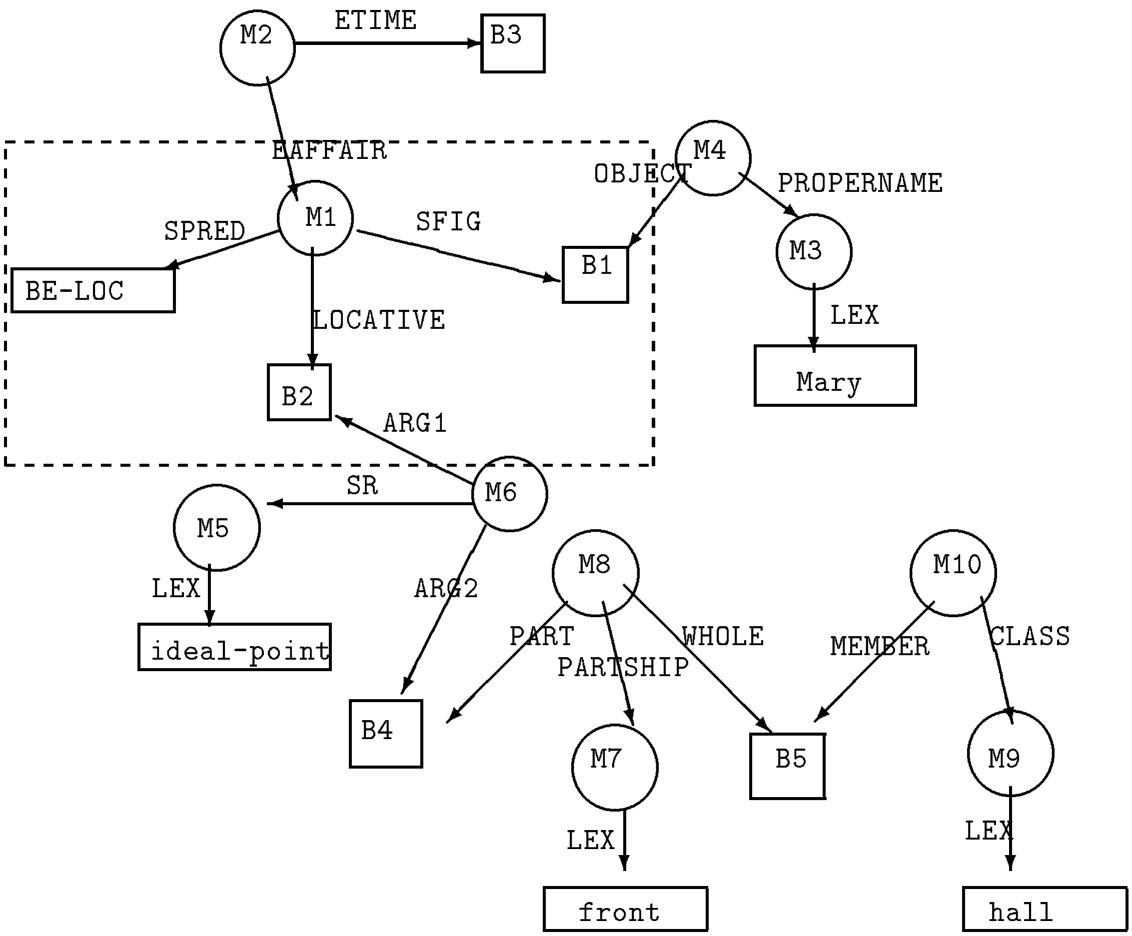

Figure 6.14: "**`Mary is at the front of the hall.`**"

In figure 6.14, ⟦`M1`⟧ is the proposition that that Mary (⟦`B1`⟧) is located at a place ⟦`B2`⟧, which is at the the front of the hall (⟦`M6`⟧, ⟦`M8`⟧).

Alternatives

None.

Issues

Non-spatial event affairs are event affairs that do not have a place in their valence structure, that is, they do not require a place as an argument to them. As the event affair to an event, they can have a place (example: "Tom played poker in the hotel room"), however, the event place is not an essential argument to the event itself. For example, "Tom played poker" makes sense. One subset of event affairs is itself spatial. Spatial event affairs are events that involve a figure object's motion, station, or orientation as represented by a spatial deep case predicate `MOVE`, `BE-LOC` or `BE-ORIENTED` respectively. Motion event affairs can have have a event time and place separate from, and in addition to, places representing the source, path, and goal point of the motion.

> John is walking *from the drinking fountain to the baseball diamond in the park.*

In this case, the source, path and goal locations are arguments to the spatial event affair, `MOVE`, and they are distinct from the place (`EPLACE`) for the entire event, the park. As presented here, spatial event affairs that involve the figure object being situated (`BE-LOC`) involve a place as one of their constituent arguments.

> John stayed *at home.*
> John was upside down *in bed.*

As a result, at the event level of structure, the `EPLACE` arc is not used when the event affair specified under `EAFFAIR` is a spatial event affair predicated by a spatial deep case `BE-LOC`. A spatial orientation event affair involves the figure object being oriented towards a goal. This is reflected by the goal argument to the spatial event, `BE-ORIENTED`. The spatial event affair of the figure being oriented has a distinct and separate place of occurrence, hence at the event-level of structure, there is an `EPLACE` arc. To summarize, between one and three places (`SOURCE`, `PATH`, `GOAL`) can be specified for spatial event affairs predicated by `MOVE`. The spatial event affairs predicated by a `BE-LOC` will always be associated with a place argument specified by the `LOCATIVE`. In this case, there is no `EPLACE` arc used at the event level. The spatial predicate `BE-ORIENTED` will have a `GOAL` arc indicating the direction of orientation.

Related Entries

Spatial Event Affair - Motion, Spatial Event Affair - Orientation, Events

References

See [31] for further discussion of this representation. See [29] for a discussion of spatial deep cases, an analysis on which this representation is based.

Author

Susan Haller, August 1991

6.8 Spatial Event Affair - Orientation

Syntax

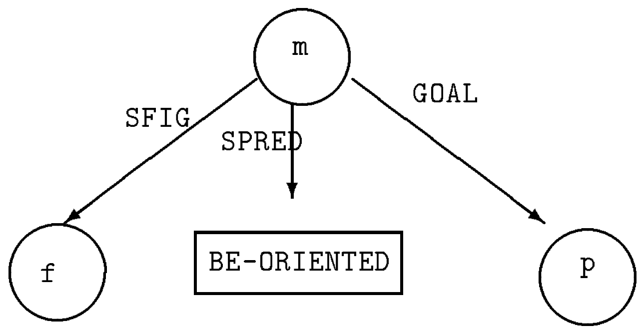

Figure 6.15:

`m` is a structured individual node with an `SFIG` arc to the individual node `f` and an `SPRED` arc to the internal constant base node `BE-ORIENTED`, and a `GOAL` arc to the individual node `p`.

Semantics

⟦m⟧ is the proposition that an event involving the orientation (⟦BE-ORIENTED⟧) of ⟦f⟧ towards a place ⟦p⟧ takes place.

Sample Context

Mary is facing the back of the theater hall from the front of the theater hall.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (ASSERT SPRED BE-ORIENTED
                    GOAL  #B1
                    SFIG  #B2)
          EPLACE #B3
          ETIME  #B4)
...
(M2! (EAFFAIR (M1! (GOAL B1)
                   (SFIG B2)
                   (SPRED BE-ORIENTED)))
     (EPLACE B3)
     (ETIME B4))
```

```
* (ASSERT  ARG1 *B1
           SR   (BUILD LEX "ideal point")
           ARG2 #B5)
...
(M4! (ARG1 B1)
     (ARG2 B5)
     (SR (M3 (LEX ideal point))))


* (ASSERT PART     *B5
          PARTSHIP (BUILD LEX "back")
          WHOLE    #B6)
...
(M6! (PART B5)
     (PARTSHIP (M5 (LEX back)))
     (WHOLE B6))


* (ASSERT MEMBER *B6
          CLASS  (BUILD LEX "hall"))
...
(M8! (CLASS (M7 (LEX hall)))
     (MEMBER B6))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "Mary"))
...
(M10! (OBJECT B2)
      (PROPERNAME (M9 (LEX Mary))))


* (ASSERT  ARG1 *B3
           ARG2 #B7
           SR   (BUILD LEX "ideal point"))
...
(M11! (ARG1 B3)
      (ARG2 B7)
      (SR (M3 (LEX ideal point))))


* (ASSERT PART     *B7
          PARTSHIP (BUILD LEX "front")
          WHOLE    *B6)
...
(M13! (PART B7)
      (PARTSHIP (M12 (LEX front)))
      (WHOLE B6))
```

Network

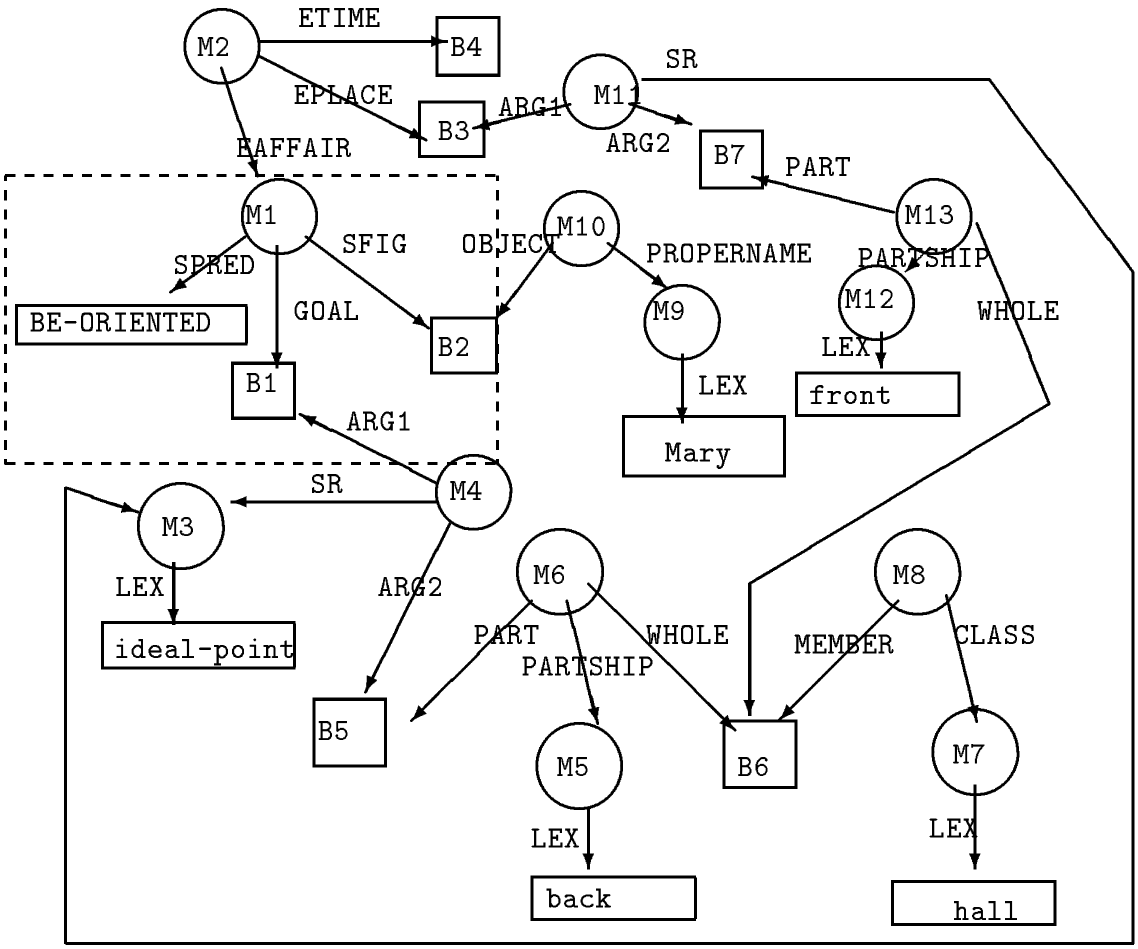

Figure 6.16: "`Mary is facing the back of the theater hall from the front of the theater hall.`"

In figure 6.16,

Alternatives

None.

Issues

Non-spatial event affairs are event affairs that do not have a place in their valence structure, that is, they do not require a place as an argument to them. As the event affair to an event, they can have a place (example: "Tom played poker in the hotel room "), however, the event place is not an essential argument to the event itself. For example, "Tom played poker" makes sense. One subset of event affairs is itself spatial. Spatial event affairs are events that involve a figure object's motion, station, or orientation as represented by a spatial deep case predicate `MOVE`, `BE-LOC` or `BE-ORIENTED` respectively. As presented here, motion event affairs can have have a event time and place separate from, and in addition to, places representing the source, path, and goal point of the motion.

> John is walking *from the drinking fountain to the baseball diamond in the park.*

In this case, the source, path and goal locations are arguments to the spatial event affair, `MOVE`, and they are distinct from the place (`EPLACE`) for the entire event, the park. Spatial event affairs that involve the figure object being situated (`BE-LOC`) involve a place as one of their constituent arguments.

> John stayed *at home.*
> John was upside down *in bed.*

As a result, at the event level of structure, the `EPLACE` arc is not used when the event affair specified under `EAFFAIR` is a spatial event affair predicated by a spatial deep case `BE-LOC`. A spatial orientation event affair involves the figure object being oriented towards a goal. This is reflected by the goal argument to the spatial event, `BE-ORIENTED`. The spatial event affair of the figure being oriented has a distinct and separate place of occurrence, hence at the event-level of structure, there is an `EPLACE` arc. To summarize, between one and three places (`SOURCE`, `PATH`, `GOAL`) can be specified for spatial event affairs predicated by `MOVE`. The spatial event affairs predicated by a `BE-LOC` will always be associated with a place argument specified by the `LOCATIVE`. In this case, there is no `EPLACE` arc used at the event level. The spatial predicate `BE-ORIENTED` will have a `GOAL` arc indicating the direction of orientation.

Related Entries

Spatial Event Affair - Station, Spatial Event Affair - Motion, Events

References

See [31] for further discussion of this representation. See [29] for a discussion of spatial deep cases, an analysis on which this representation is based.

Author

Susan Haller, August 1991

6.9 Spatial Proximity

Syntax

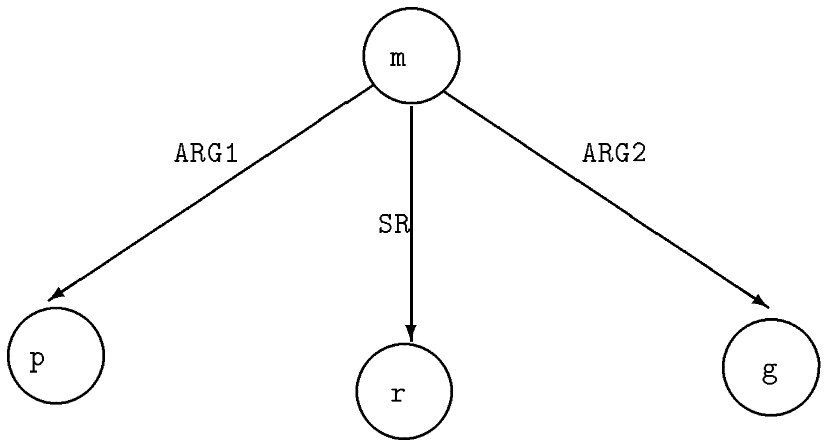

Figure 6.17:

m is a structured individual node with an ARG1 arc to the individual node p, an SR arc to the individual node r, and an ARG2 arc to the individual node g.

Semantics

⟦m⟧ is the proposition that place ⟦p⟧ is delimited by a spatial proximity relationship ⟦r⟧ with respect to a ground object ⟦g⟧.

Sample Context

John and Mary are playing tennis in the park.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (BUILD AGENT
                     (BUILD SETMEMB #B1
                            SETMEMB #B2
                            SETTYPE ANDSET)
                   ACT
                     (BUILD ACTION  (BUILD LEX "play")
                      OBJECT1 (BUILD LEX "tennis")))
          EPLACE #B3
          ETIME  #B4)
...
```

```
(M6!
  (EAFFAIR
    (M5 (ACT (M4 (ACTION (M2 (LEX play)))
                 (OBJECT1 (M3 (LEX tennis)))))
        (AGENT (M1 (SETMEMB B1 B2)
                   (SETTYPE ANDSET)))))
  (EPLACE B3)
  (ETIME B4))


* (ASSERT ARG1 *B3
          ARG2 #B5
          SR   (BUILD LEX "in"))
...
(M8! (ARG1 B3)
     (ARG2 B5)
     (SR (M7 (LEX in))))

* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "park"))
...
(M10! (CLASS (M9 (LEX park)))
      (MEMBER B5))


* (ASSERT OBJECT     *B1
          PROPERNAME (BUILD LEX "Tom"))
...
(M12! (OBJECT B1)
      (PROPERNAME (M11 (LEX Tom))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "Mary"))
...
(M14! (OBJECT B2)
      (PROPERNAME (M13 (LEX Mary))))
```

Network

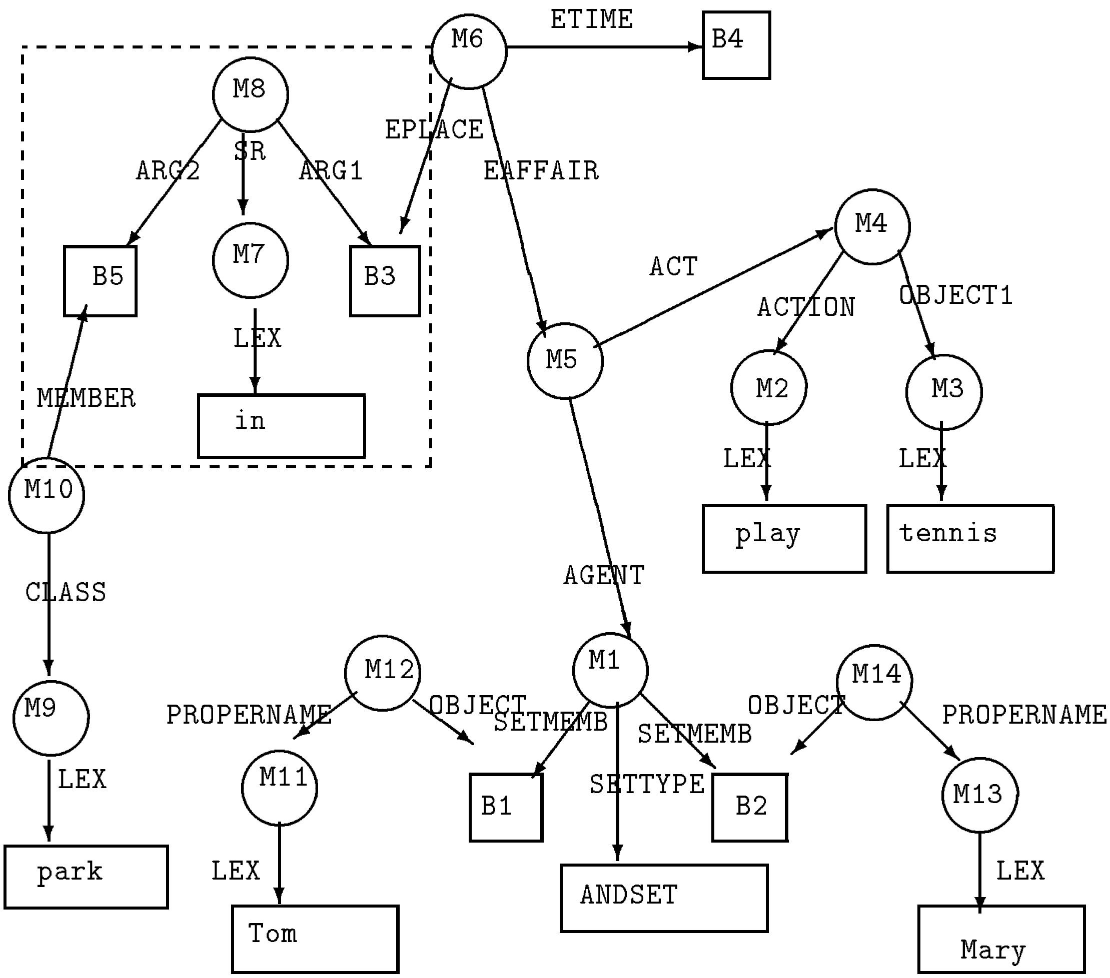

Figure 6.18: "`John and Mary are playing tennis in the park.`"

In figure 6.18, [[`M8`]] is the proposition that the place ([[`B3`]]) is delimited by the park ([[`B5`]]) through the spatial relationship [[`M7`]].

Sample Context

John went to the store in Buffalo.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
    (BUILD SPRED MOVE
```

```
   SFIG  #B1
   GOAL  #B2)
  EPLACE #B3
  ETIME  #B4)
...
(M2! (EAFFAIR (M1 (GOAL B2)
                  (SFIG B1)
                  (SPRED MOVE)))
     (EPLACE B3)
     (ETIME B4))


* (ASSERT BEFORE   *B4
  AFTER    #B5
  DURATION #B6)
...
(M3! (AFTER B5)
     (BEFORE B4)
     (DURATION B6))


* (ASSERT OBJECT *B1
  PROPERNAME (BUILD LEX "John"))
...
(M5! (OBJECT B1)
     (PROPERNAME (M4 (LEX John))))


* (ASSERT SR   (BUILD LEX "ideal point")
  ARG1 *B2
  ARG2 #B7)
...
(M7! (ARG1 B2)
     (ARG2 B7)
     (SR (M6 (LEX ideal point))))


* (ASSERT MEMBER *B7
  CLASS  (BUILD LEX "store"))
...
(M9! (CLASS (M8 (LEX store)))
     (MEMBER B7))


* (ASSERT ARG1 *B3
  SR   (BUILD LEX "in")
  ARG2 #B8)
```

```
(M11! (ARG1 B3)
      (ARG2 B8)
      (SR (M10 (LEX in))))

* (ASSERT OBJECT     *B8
  PROPERNAME (BUILD LEX "Buffalo"))
...
(M13! (OBJECT B8)
      (PROPERNAME (M12 (LEX Buffalo))))
```

Network

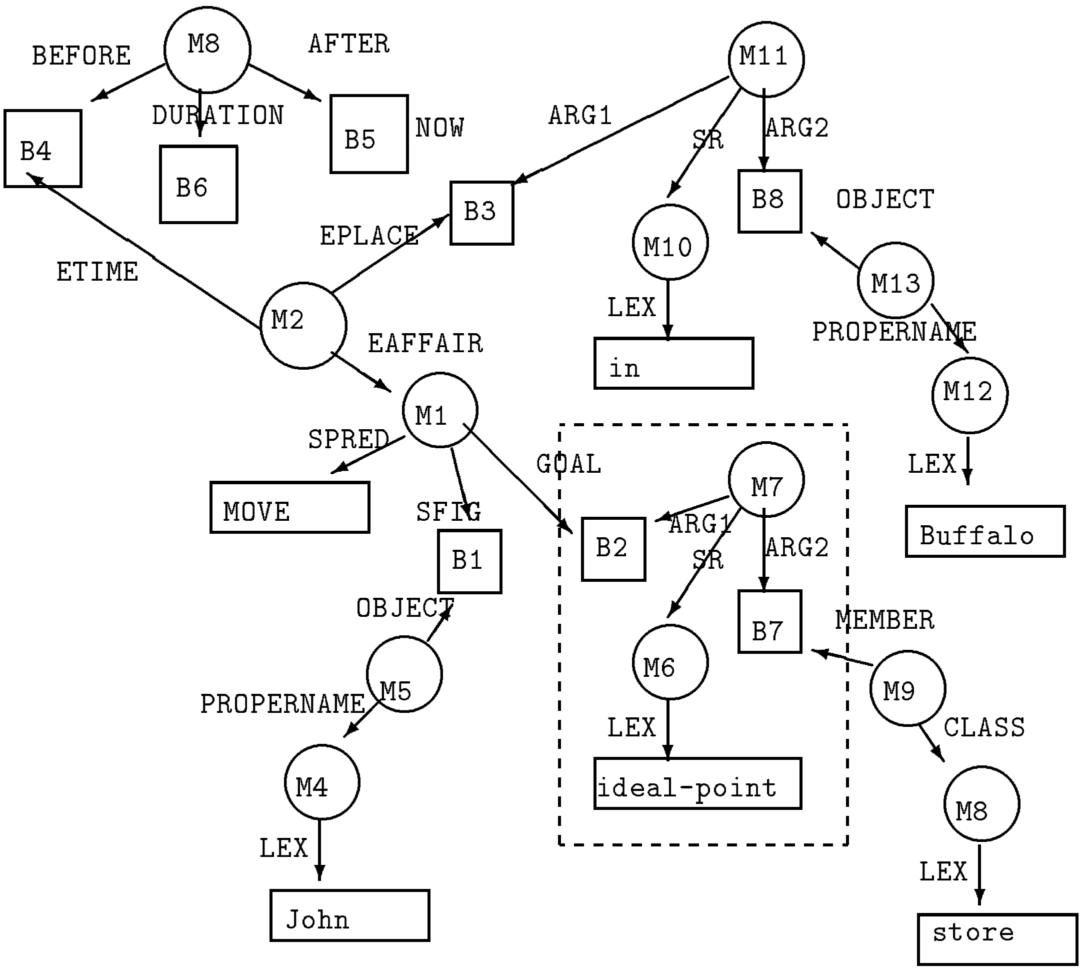

Figure 6.19: "`John went to the store in Buffalo.` "

In figure 6.19, [[`M7`]] is the proposition that the place ([[`B2`]]) is delimited by the store ([[`B7`]]) through the spatial relationship [[`M6`]].

Alternatives

None.

Issues

Spatial proximity is a tool used in human cognition to localize a space in the vicinity of the space occupied by a ground object. A space can be delimited in terms of its spatial proximity to an associated ground object through a directional, or inclusional relationship to that object. As in the first example, directional and inclusional relationships to a ground object are explicitly stated through prepositional choice. The preposition "in" indicates that the park boundary and all space included in that boundary defines the space of interest. Sometimes however, a space is not defined with respect to the ground, but rather is defined as being coincident with it. This is the case in the second example. In this situation, the ground delimits the space by serving as an object that the space can be coincident with. Hence the space is defined being the ground object idealized as a point.

Related Entries

References

See [31] for further discussion of this representation. [28] is an analysis of space in terms of place predicates. This work is the motivational basis for this representation of space.

Author

Susan Haller, August 1991

6.10 Spatial Proximity with a Directional Reference

Syntax

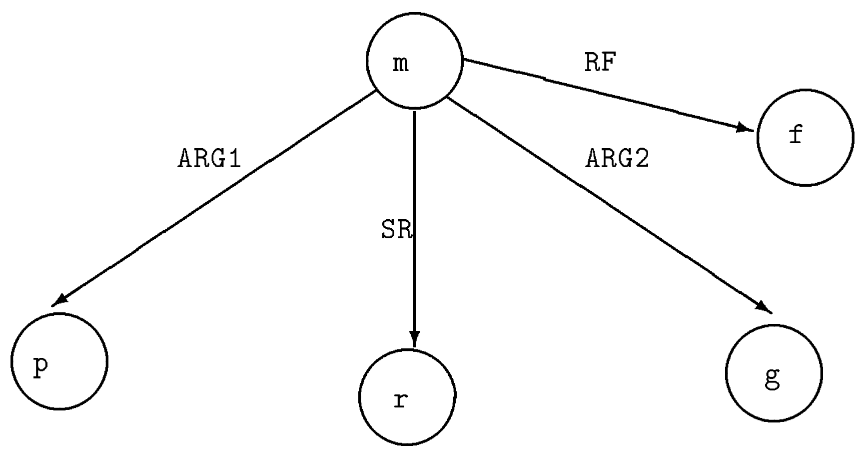

Figure 6.20:

m is a structured individual node with an ARG1 arc to the individual node p, an SR arc to the individual node r, an ARG2 arc to the individual node g, and an RF arc to the individual node f.

Semantics

⟦m⟧ is the proposition that place ⟦p⟧ is delimited by a spatial proximity relationship ⟦r⟧ with respect to a ground object ⟦g⟧ where the directional reference is understood with respect to the frame of reference ⟦f⟧.

Sample Context

Intrinsic reference frame resolution: John and Mary are playing catch in front of the house.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
           (BUILD AGENT
                  (BUILD SETMEMB #B1
                         SETMEMB #B2
                         SETTYPE ANDSET)
                 ACT
                  (BUILD ACTION  (BUILD LEX "play")
                   OBJECT1 (BUILD LEX "catch")))
```

```
          EPLACE #B3
          ETIME  #B4)
...
(M6!
  (EAFFAIR
    (M5 (ACT (M4 (ACTION (M2 (LEX play)))
                 (OBJECT1 (M3 (LEX catch)))))
        (AGENT (M1 (SETMEMB B1 B2)
                   (SETTYPE ANDSET)))))
  (EPLACE B3)
  (ETIME B4))


* (ASSERT ARG1 *B3
          ARG2 #B5
          RF   *B3
          SR   (BUILD LEX "in front of"))
...
(M8! (ARG1 B3)
     (ARG2 B5)
     (RF   B3)
     (SR (M7 (LEX in front of))))

* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "house"))
...
(M10! (CLASS (M9 (LEX house)))
      (MEMBER B5))


* (ASSERT OBJECT     *B1
          PROPERNAME (BUILD LEX "Tom"))
...
(M12! (OBJECT B1)
      (PROPERNAME (M11 (LEX Tom))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "Mary"))
...
(M14! (OBJECT B2)
      (PROPERNAME (M13 (LEX Mary))))
```

Network

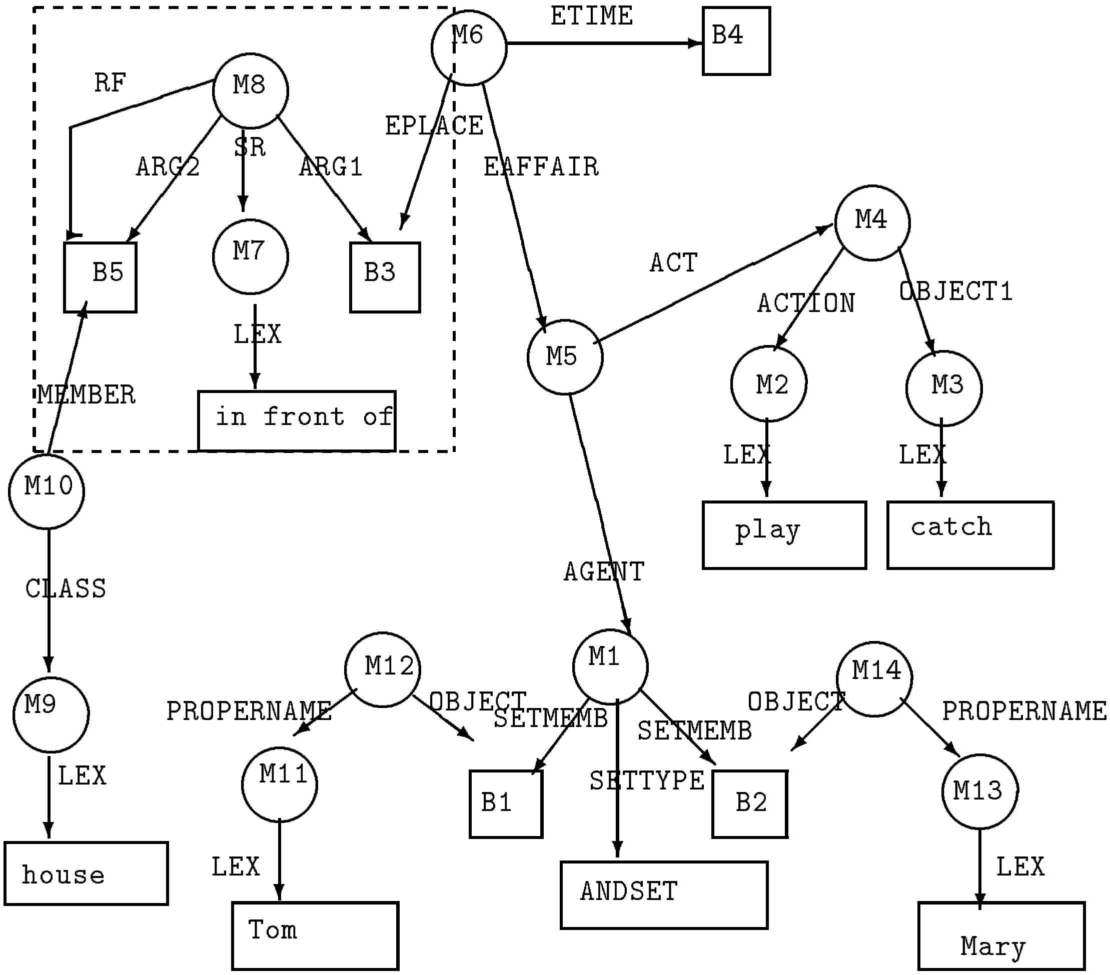

Figure 6.21: "John and Mary are playing catch in front of the house."

In figure 6.21, ⟦M8⟧ is the proposition that the place (⟦B3⟧) is delimited by the house (⟦B5⟧) through the spatial relationship ⟦M7⟧ where the directional reference is understood with respect to ⟦B5⟧.

Issues

Intrinsic reference frame resolution is the first of three types of reference frame resolution. A reference frame is that object or entity that directional references are made with respect to. The reference frame is not always the same as the explicitly stated reference point in a spatial expression which is called the ground object. In many situations the reference frame for the spatial expression is anchored in some other object than the ground.

Some objects have inherent directionality, that is, they are conceptualized as having axes that identify an up/down, front/back, and left/right for the object. Some objects have inherent directionality that is incomplete. Trees, for instance, have a top and bottom (up/down) but no front/back or left/right.
Intrinsic resolution is when the reference frame is anchored in the directionality of the ground object which must have inherent directionality. In terms of the example above, a location is being specified. The directionality expressed by "in front of" is used with respect to a reference frame, which is the ground object, the house. The house is a suitable reference frame because it has an inherent front and back.

Alternatives

None.

Sample Context

Inherited Reference Frame Resolution: John is in front of Mary with respect to the theater hall.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (ASSERT SPRED    BE-LOC
                    LOCATIVE #B1
                    SFIG     #B2)
          ETIME #B3)
...
(M2! (EAFFAIR (M1! (LOCATIVE B1)
                   (SFIG B2)
                   (SPRED BE-LOC)))
     (ETIME B3))


* (ASSERT ARG1 *B1
          SR   (BUILD LEX "in front of")
          ARG2 #B4
          RF   #B5)
...
(M4! (ARG1 B1)
     (ARG2 B4)
     (RF B5)
     (SR (M3 (LEX in front of))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "John"))
...
(M6! (OBJECT B2)
     (PROPERNAME (M5 (LEX John))))
```

```
* (ASSERT OBJECT     *B4
          PROPERNAME (BUILD LEX "Mary"))
...
(M8! (OBJECT B4)
     (PROPERNAME (M7 (LEX Mary))))


* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "hall"))
...
(M10! (CLASS (M9 (LEX hall)))
      (MEMBER B5))
```

Network

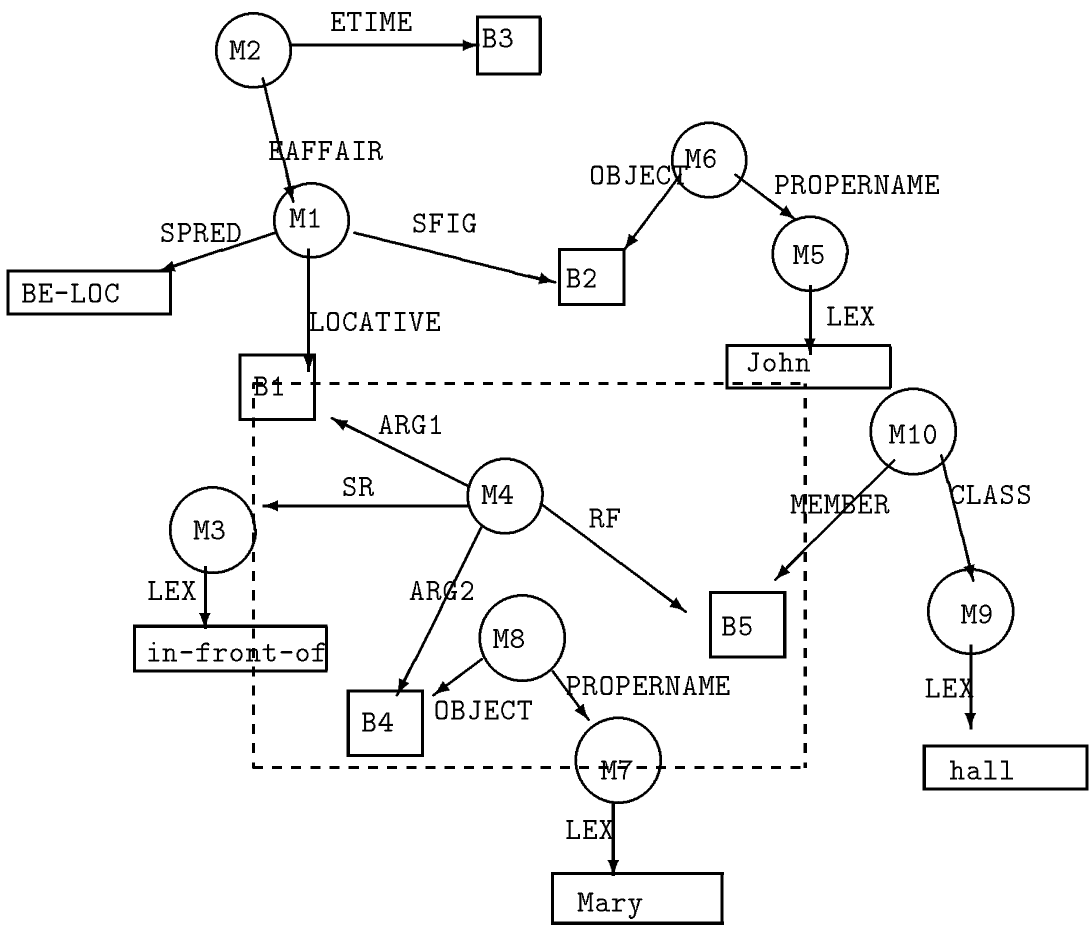

Figure 6.22: "**`John is in front of Mary with respect to the theater hall.`** "

In figure 6.22, ⟦`M4`⟧ is the proposition that the place (⟦`B1`⟧) delimited by Mary (⟦`B4`⟧) through the spatial relationship ⟦`M3`⟧ is understood with repsect to the hall (⟦`B5`⟧) as a reference frame.

Issues

Inherited reference frame resolution occurs when the reference frame is anchored in an object that the ground object is a part of, or housed in. This object, called the host, must have inherent directionality. In the context of Mary seated in a theater, the place identified in the above example is *not* in front of Mary's front. A quick test bares this out. If Mary was turned around talking to someone behind her, "in front of Mary" still succeeds in identifying the same place that it would if she where sitting facing forward. Hence the directionality, "in front", must be with respect to a frame of reference other than the ground object, Mary.

The reference frame is inherited from a host object, in this case the theater.

Alternatives

None.

Sample Context

Perspective Reference Frame Resolution: John is in front of the tree.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (ASSERT SPRED    BE-LOC
                    LOCATIVE #B1
                    SFIG     #B2)
          ETIME #B3)
...
(M2! (EAFFAIR (M1! (LOCATIVE B1)
                   (SFIG B2)
                   (SPRED BE-LOC)))
     (ETIME B3))


* (ASSERT ARG1 *B1
          ARG2 #B4
          SR   (BUILD LEX "in front of")
          RF   PE)
...
(M4! (ARG1 B1)
     (ARG2 B4)
     (RF PE)
     (SR (M3 (LEX in front of))))


* (ASSERT OBJECT *B2
          PROPERNAME (BUILD LEX "John"))
...
(M6! (OBJECT B2)
     (PROPERNAME (M5 (LEX John))))


* (ASSERT MEMBER *B4
          CLASS  (BUILD LEX "tree"))
...
(M8! (CLASS (M7 (LEX tree)))
     (MEMBER B4))
```

Network

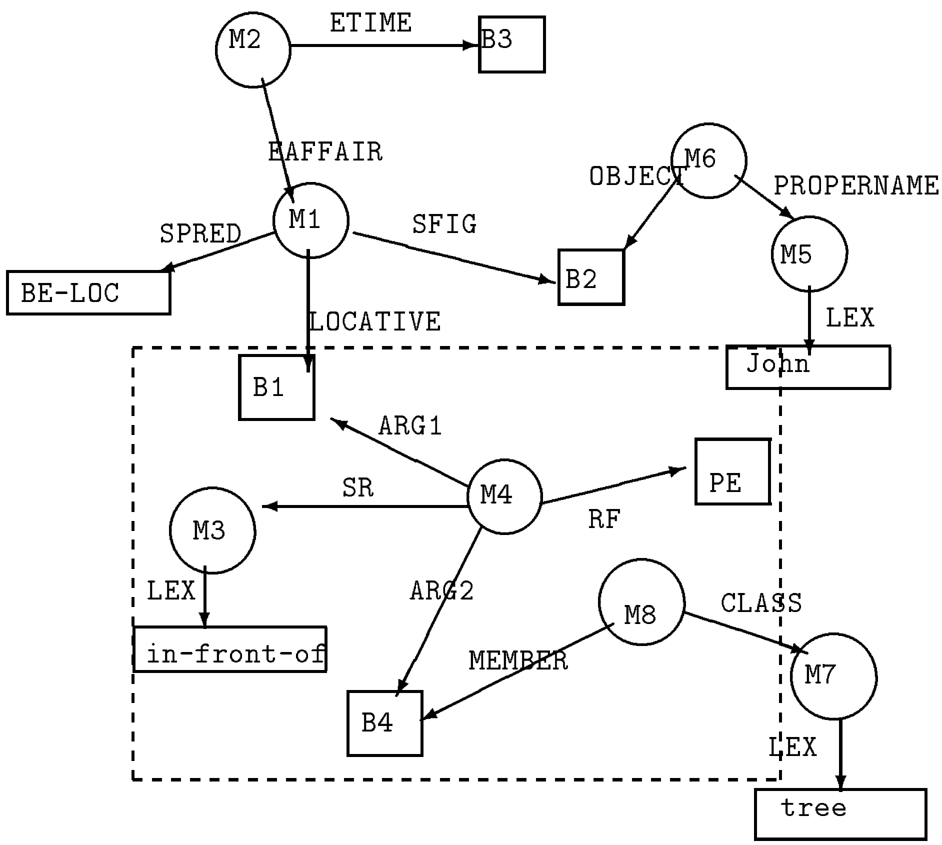

Figure 6.23: "`John is in front of the tree.` "

In figure 6.23, [[`M4`]] is the proposition that the place ([[`B1`]]) delimited by the tree ([[`B4`]]) through the spatial relationship [[`M3`]] is understood with repsect to the perspective ego ([[`PE`]]) as a reference frame.

Issues

Sometimes a directional reference is made using a ground object with no appropriate directionality, and with no potential host objects from which a reference frame can be inherited. In this case, the reference frame is not anchored in an object that is part of the spatial situation, but rather, lies with an imaginary onlooker's point of view, an onlooker that the speaker and hearer can both identify with, and who is called the perspective ego (PE). This kind of resolution is called perspective reference frame resolution.
In the above example, trees have no inherent front/back, yet a place is identified with "in front of the tree". The PE is considered to be in the canonical encounter position, that is, the PE's front faces the other object which is assumed to be facing the PE. Of course, the

object facing the PE does not have a front (this is the case with trees), so whatever is facing the PE is distinguished as the front of the object. If the PE viewed the tree from the other side, the same place would be identified by "in back of the tree". Hence, we can see that the frame of reference does not lie with any object in the spatial situation; the frame of reference is the PE in an encounter situation with the ground object (in this case, the tree). We note here that the PE can also take on a projecting perspective. This is when the PE coincides with the object, facing the same was as the object is considered to be facing. In English this is seldom used, and it occurs when the relevant other object has a directionality of its own (a car for example). Hence projective perspective reference frame resolution is just a special case of inherited reference frame resolution discussed in the previous example.

Alternatives

None.

Related Entries

Spatial Proximity

References

See [31] for further discussion of this representation. [1, 2] are studies of object-inherent and perspectively imposed directionality on which this representation is based.

Author

Susan Haller, August 1991

6.11 Resolved Spatial Reference Frame

Syntax

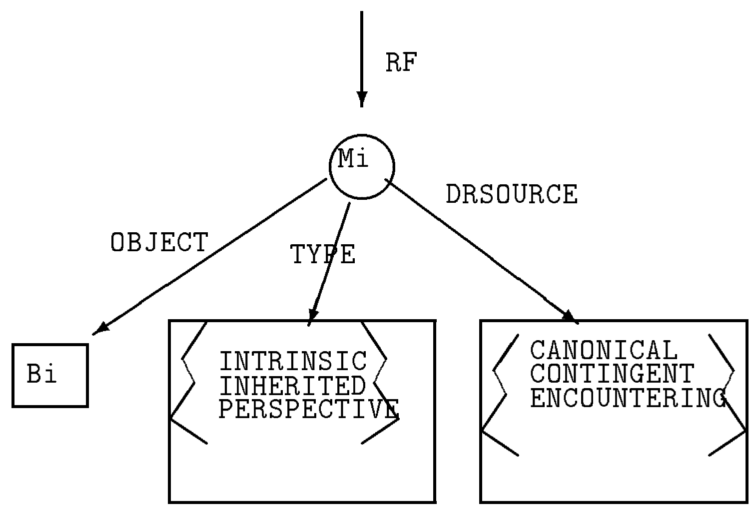

Figure 6.24:

m is a structured node with an OBJECT arc to the individual node f, a TYPE arc to one of the constant base nodes INTRINSIC, INHERITED, or PERSPECTIVE, and a DRSOURCE arc to one of the constant base nodes CANONICAL, CONTINGENT, or ENCOUNTERING.

Semantics

⟦m⟧ is the proposition that the object ⟦f⟧ is being used as a reference frame as a result of intrinsic (⟦INTRINSIC⟧), inherited (⟦INHERITED⟧), or perspective (⟦PERSPECTIVE⟧) reference frame resolution. The source of (or reason for) the directionality of ⟦f⟧ is due to a canonical (⟦CANONICAL⟧), contingent (⟦CONTINGENT⟧), or encountering (⟦ENCOUNTERING⟧) property of the object.

Sample Context

Inherited Reference Frame Resolution due to a Canonical Property of the Reference Frame: John is in front of Mary with respect to the theater hall.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
          (ASSERT SPRED    BE-LOC
                  LOCATIVE #B1
                  SFIG     #B2)
        ETIME #B3)
```

```
...
(M2! (EAFFAIR (M1! (LOCATIVE B1)
                   (SFIG B2)
                   (SPRED BE-LOC)))
     (ETIME B3))


* (ASSERT   ARG1 *B1
            SR   (BUILD LEX "in front of")
            ARG2 #B4
            RF   #B5)
...
(M4! (ARG1 B1)
     (ARG2 B4)
     (RF B5)
     (SR (M3 (LEX in front of))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "John"))
...
(M6! (OBJECT B2)
     (PROPERNAME (M5 (LEX John))))


* (ASSERT OBJECT     *B4
          PROPERNAME (BUILD LEX "Mary"))
...
(M8!  (OBJECT B4)
      (PROPERNAME (M7 (LEX Mary))))


* (ASSERT MEMBER *B5
          CLASS  (BUILD LEX "hall"))
...
(M10! (CLASS (M9 (LEX hall)))
      (MEMBER B5))


* (ASSERT OBJECT  *B5
          TYPE    INHERITED
          SOURCE  CANONICAL)
...
(M11! (OBJECT B5)
      (SOURCE CANONICAL)
      (TYPE INHERITED))
```

Network

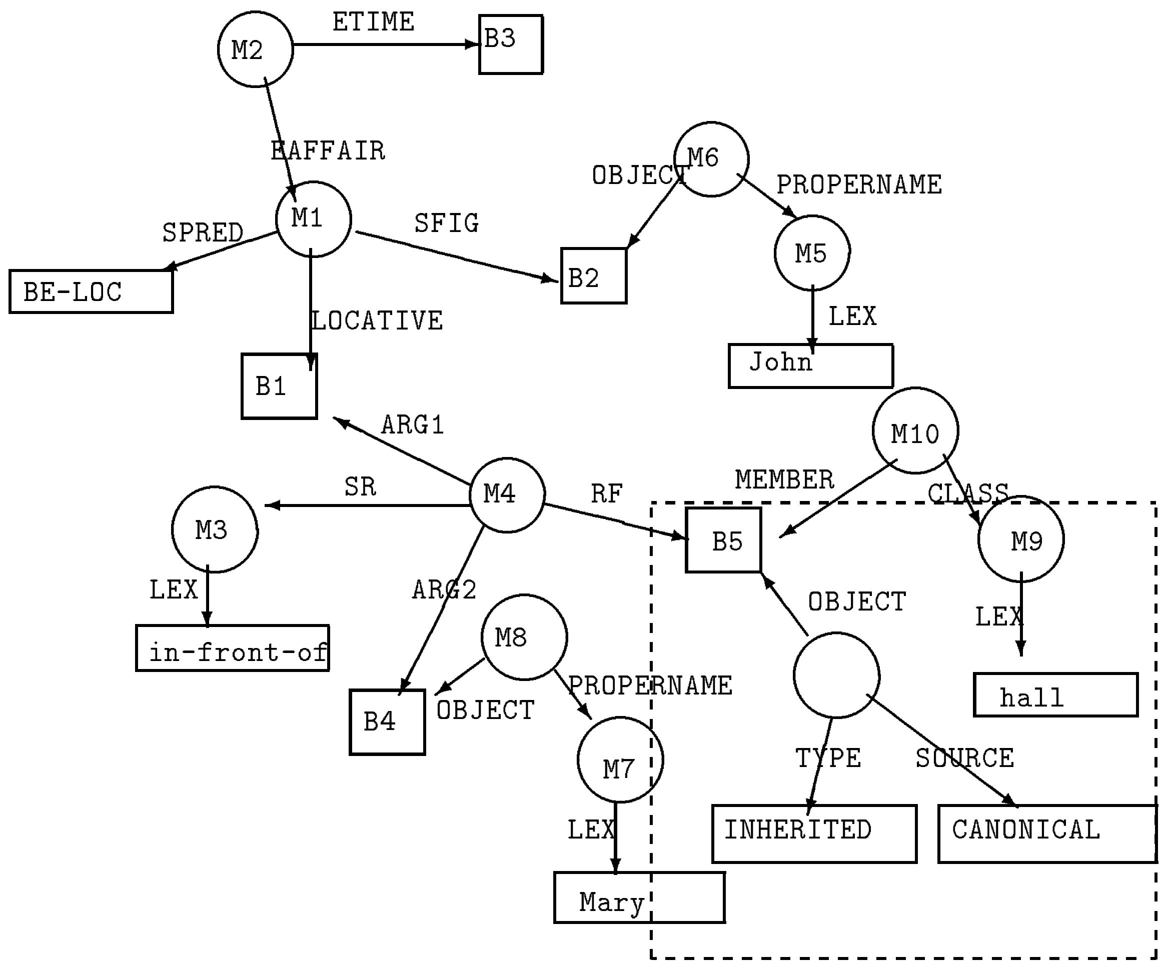

Figure 6.25: " John is in front of Mary with respect to the theater hall."

In figure 6.25, [[M11]] is the proposition that the reference frame ([[B5]]) is inherited ([[INHERITED]]) and the reference frame's directionality is canonically understood ([[CANONICAL]]).

Issues

The reference frame for the directional reference "in front" is not the ground object, Mary, but rather, the theater which acts as a host object (INHERITED). The source of theater hall's directionality is (CANONICAL). This means that the object has statically understood features that determine a front and back. Note that this representation in part, makes explicit information that is implicit in the case frame for inheritedly resolved directional reference (see entry for).

Alternatives

None.

Sample Context

Intrinsic Reference Frame Resolution due to a Contingent Property of the Reference Frame: John is in front of the rolling boulder.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (ASSERT SPRED    BE-LOC
                    LOCATIVE #B1
                    SFIG     #B2)
          ETIME #B3)
...
(M2! (EAFFAIR (M1! (LOCATIVE B1)
                   (SFIG B2)
                   (SPRED BE-LOC)))
     (ETIME B3))


* (ASSERT ARG1 *B1
          ARG2 #B4
          RF   *B4
          SR   (BUILD LEX "in front of"))
...
(M4! (ARG1 B1)
     (ARG2 B4)
     (RF B4)
     (SR (M3 (LEX in front of))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "John"))
...
(M6! (OBJECT B2)
     (PROPERNAME (M5 (LEX John))))


* (ASSERT MEMBER *B4
          CLASS  (BUILD LEX "boulder"))
...
(M8! (CLASS (M7 (LEX boulder)))
     (MEMBER B4))


* (ASSERT OBJECT   *B4
```

```
          PROPERTY (BUILD LEX "rolling"))
...
(M10! (OBJECT B4)
      (PROPERTY (M9 (LEX rolling))))


* (ASSERT OBJECT *B4
          TYPE   INTRINSIC
          SOURCE CANONICAL)
...
(M11! (OBJECT B4)
      (SOURCE CANONICAL)
      (TYPE INTRINSIC))
```

Network

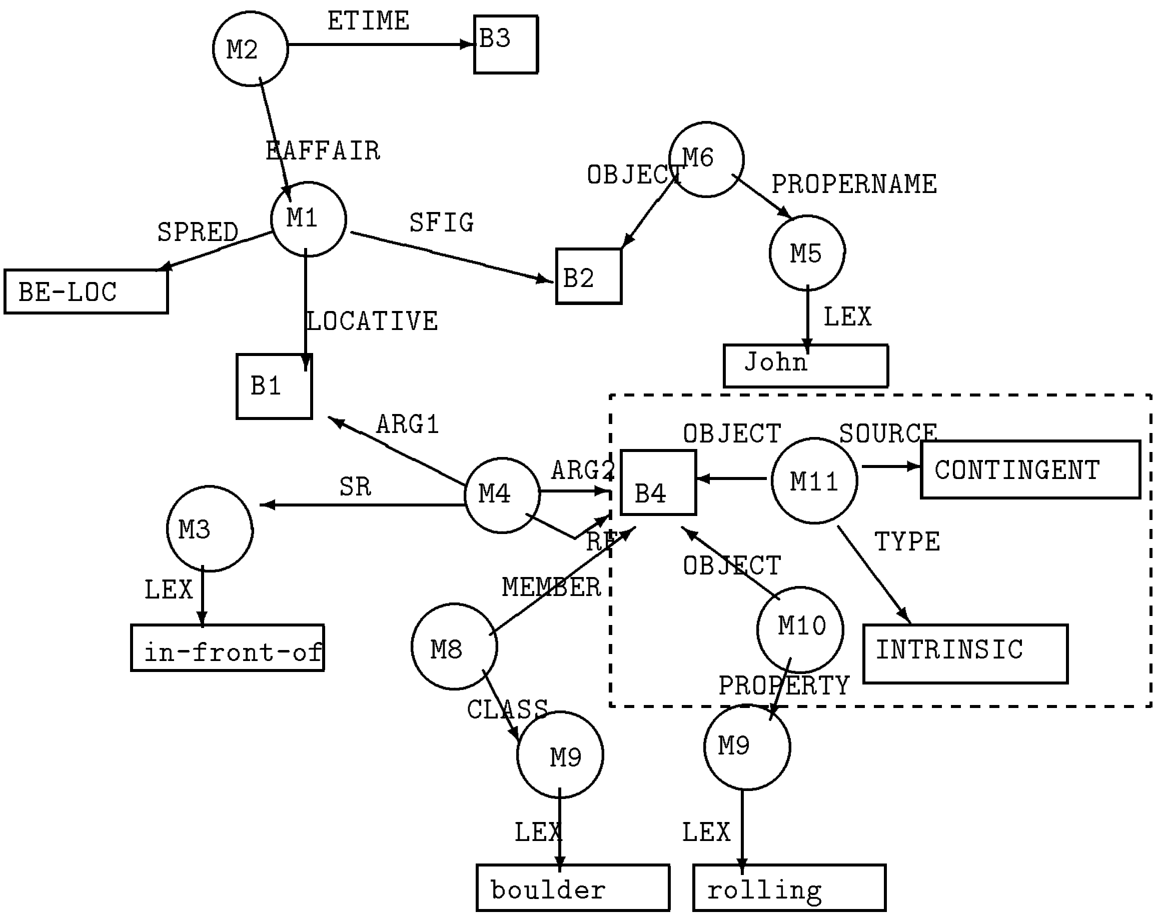

Figure 6.26: " John is in front of the rolling boulder."

In figure 6.26, ⟦M11⟧ is the proposition that the reference frame (⟦B4⟧) is intrinsic (⟦INTRINSIC⟧) and the reference frame's directionality is contigently understood (⟦CONTINGENT⟧).

Issues

The reference frame for the directional reference "in front" is the ground object, the boulder, which is an INTRINSIC resolution. (see spatial proximity with a directional reference). The source of the boulder's directionality is CONTINGENT to the its motion, that is, the boulder, statically understood, has no front. However, it does have a front (and a back) determined by the path of its movement.

Alternatives

None.

Sample Context

Perspective Reference Frame Resolution due to the Encounter Property of the Reference Frame: John is in front of the tree.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (ASSERT SPRED    BE-LOC
                    LOCATIVE #B1
                    SFIG     #B2)
          ETIME #B3)
...
(M2! (EAFFAIR (M1! (LOCATIVE B1)
                   (SFIG B2)
                   (SPRED BE-LOC)))
     (ETIME B3))


* (ASSERT ARG1  *B1
          ARG2  #B4
          SR    (BUILD LEX "in front of")
          RF    PE)
...
(M4! (ARG1 B1)
     (ARG2 B4)
     (RF PE)
     (SR (M3 (LEX in front of))))


* (ASSERT OBJECT     *B2
          PROPERNAME (BUILD LEX "John"))
...
(M6! (OBJECT B2)
     (PROPERNAME (M5 (LEX John))))


* (ASSERT MEMBER *B4
          CLASS  (BUILD LEX "tree"))
...
(M8! (CLASS (M7 (LEX tree)))
     (MEMBER B4))


* (ASSERT OBJECT  PE
```

```
          TYPE    ENOUNTER
          SOURCE  PERSPECTIVE)
...
(M9! (OBJECT PE)
     (SOURCE PERSPECTIVE)
     (TYPE ENOUNTER))
```

Network

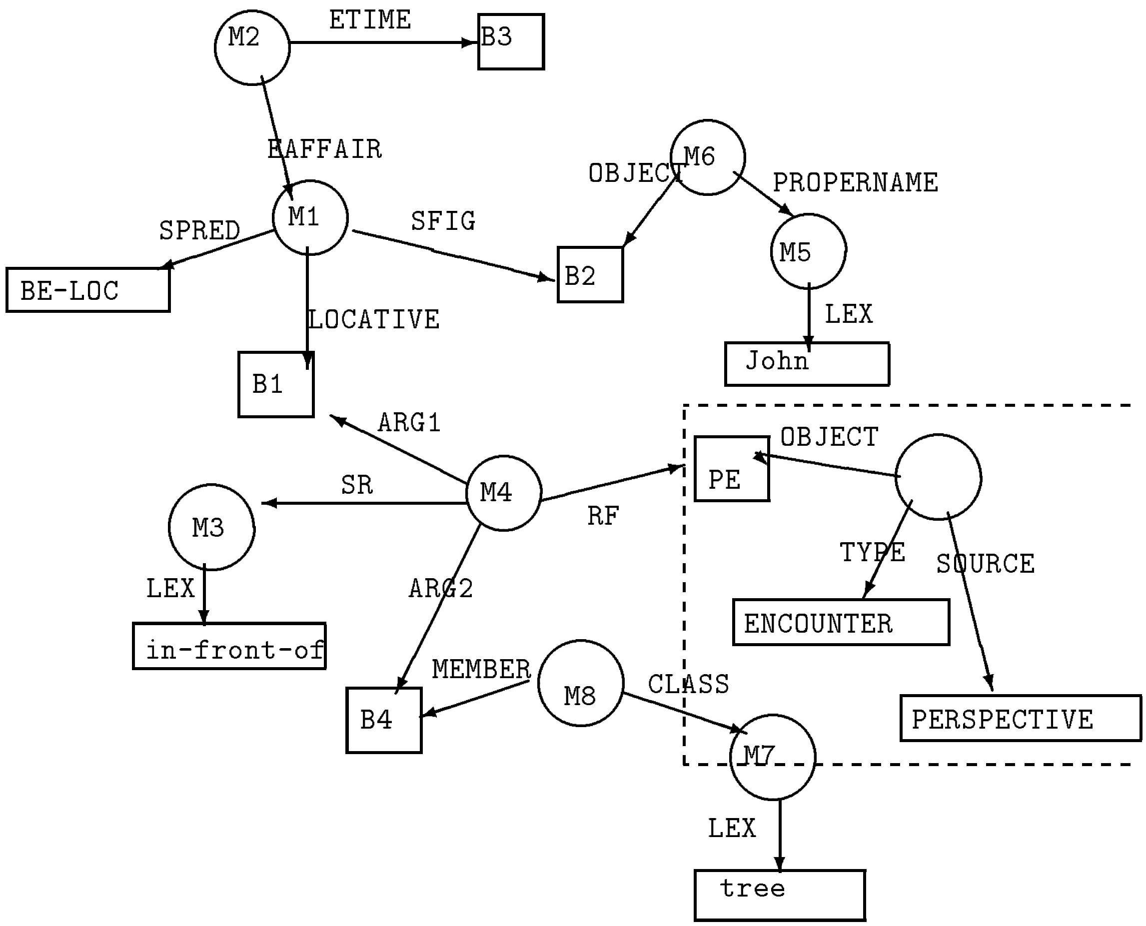

Figure 6.27: " John is in front of the tree."

In figure 6.27, ⟦M9⟧ is the proposition that the reference frame is the perspective ego (⟦PE⟧) which is an encounter (⟦ENCOUNTER⟧) and the reference frame's directionality is perspectively understood (⟦PERSPECTIVE⟧).

Issues

The reference frame for the directional reference "in front" is not the ground object, the tree, but rather, the perspective ego (PERSPECTIVE). The source of perspective ego's directionality is inherent, but ENCOUNTERING. We note that this distinction is necessary because the PE can be coincident with an object, in which case, the front of the PE does not face the other object, but rather coincides with it. This kind of directionality (called projective perspective directionality) rarely occurs, and when it does, the object the PE coincides with has inherent directionality and can act as a host. Hence, this directionality source is not needed.

Alternatives

None.

Related Entries

Spatial Proximity, Spatial Proximity with a Directional Reference

References

See [31] for further discussion of this representation. [1, 2] are studies of object-inherent and perspectively imposed directionality on which this representation is based.

Author

Susan Haller, August 1991

6.12 Event Time Relation Between Conflated Spatial Event Affairs

Syntax

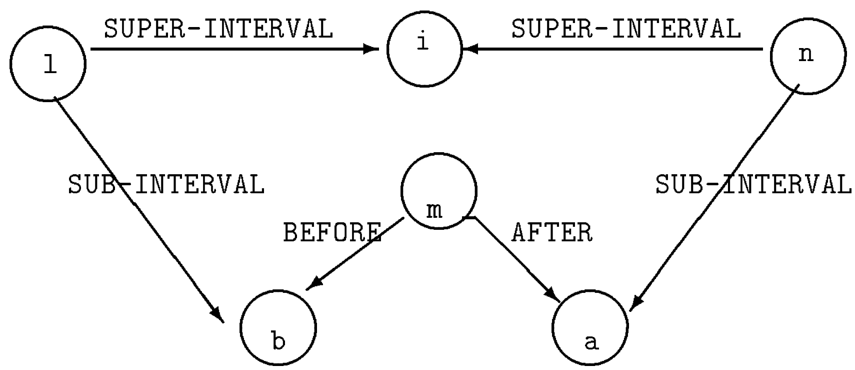

Figure 6.28:

`m` is a structured proposition node with a `BEFORE` arc to the individual node `b` and an `AFTER` arc to the individual node `a`. `l` is a structured proposition node with a `SUPER-INTERVAL` arc the individual node `i` and a `SUB-INTERVAL` arc to the individual node `b`. `n` is a structured proposition node with a `SUPER-INTERVAL` arc the individual node `i` and a `SUB-INTERVAL` arc to the individual node `a`.

Semantics

⟦m⟧ is the proposition that the time interval ⟦b⟧ took place before the time interval ⟦a⟧. ⟦l⟧ and ⟦n⟧ are the propositions that ⟦b⟧ ans ⟦a⟧ repspectively are subintervals of the time interval ⟦i⟧.

Sample Context

John went to the store in Buffalo.

SNePSUL Interaction

```
* (ASSERT EAFFAIR
            (ASSERT SPRED MOVE
                    SFIG  #B1
                    GOAL  #B2)
          ETIME #B3)
...
(M2! (EAFFAIR (M1! (GOAL B2)
                   (SFIG B1)
```

```
                   (SPRED MOVE)))
     (ETIME B3))


* (ASSERT OBJECT     *B1
          PROPERNAME (BUILD LEX "John"))
...
(M4! (OBJECT B1)
     (PROPERNAME (M3 (LEX John))))


* (ASSERT ARG1 *B2
          ARG2 #B4
          SR   (BUILD LEX "ideal point"))
...
(M6! (ARG1 B2)
     (ARG2 B4)
     (SR (M5 (LEX ideal point))))


* (ASSERT MEMBER *B4
          CLASS  (BUILD LEX "store"))
...
(M8! (CLASS (M7 (LEX store)))
     (MEMBER B4))


* (ASSERT EAFFAIR
            (ASSERT SPRED    BE-LOC
                    LOCATIVE *B2
                    SFIG     *B1)
          ETIME #B5)
...
(M10! (EAFFAIR (M9! (LOCATIVE B2)
                    (SFIG B1)
                    (SPRED BE-LOC)))
      (ETIME B5))


* (ASSERT BEFORE *B3
          AFTER  *B5
          DURATION #B6)
...
(M11! (AFTER B5)
      (BEFORE B3)
      (DURATION B6))
```

```
* (ASSERT SUPER-INTERVAL *B6
          SUB-INTERVAL   *B3)
...
(M12! (SUB-INTERVAL B3)
      (SUPER-INTERVAL B6))


* (ASSERT SUPER-INTERVAL *B6
          SUB-INTERVAL   *B5)
...
(M13! (SUB-INTERVAL B5)
      (SUPER-INTERVAL B6))
```

Network

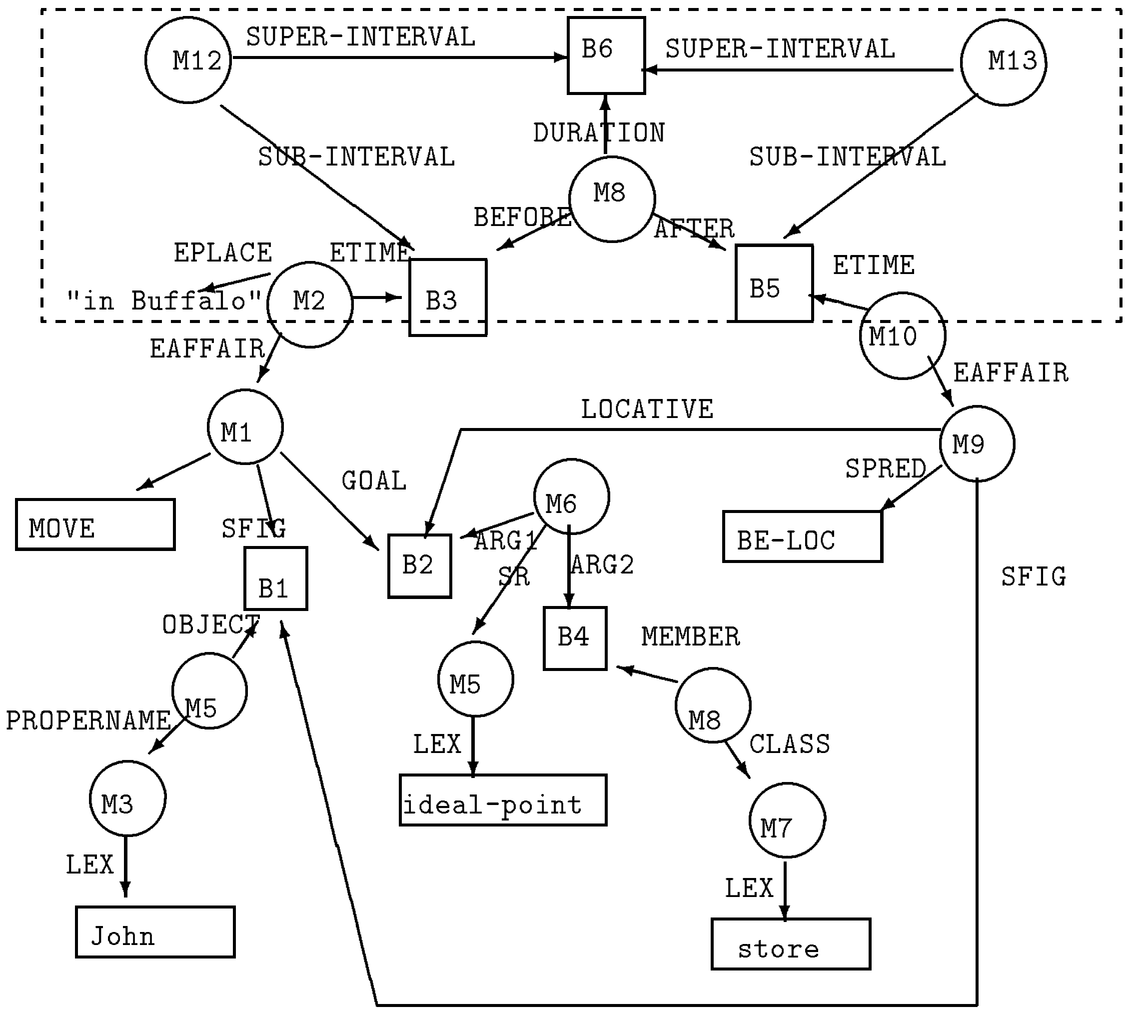

Figure 6.29: "`John went to the store in Buffalo.`"

In figure 6.29, ⟦`M11`⟧ is the proposition that that the time interval of John going to the store (⟦`B3`⟧) is before the time interval of John being at the store (⟦`B5`⟧). ⟦`M12`⟧ and ⟦`m13`⟧ are the propositions that ⟦`B3`⟧ and ⟦`B5`⟧ respectively are subintervals of time interval ⟦`B6`⟧.

Alternatives

None.

Issues

In English, many verbs that indicate motion are conflated with semantic information pertaining to manner (roll, run, stride, etc.), or pertaining to path (enter, exit, put, etc.) [29]. A spatial event affair involving a motion verb (spatial deep form `MOVE`) can be conflated with a spatial event affair involving a station act (spatial deep form `BE-LOC`) whenever a source or goal of the motion is indicated. Unlike motion-manner or motion-path conflation however, the two spatial acts cannot occur simultaneously, they must be temporally ordered. In the example above, going to the store conflates two spatial acts, a `MOVE` followed by a `BE-LOC` in the store.

Related Entries

Spatial Event Affair - Station, Spatial Event Affair - Motion, Events

References

See [31] for further discussion of this representation. See [29] for a discussion of spatial deep cases and conflation, an analysis on which this representation is based.

Author

Susan Haller, August 1991

Bibliography

[1] H. H. Clark. Space, time, semantics, and the child. In T. E. Moore, editor, *Cognitive Development and the Aquisition of Language*. Academic Press, 1973.

[2] C. J. Fillmore. Subjects, speakers, and roles. In R. J. Jarvella and W. Klein, editors, *Speech, Place, and Action: Studies in Deixis and Related Topics*. John Wiley Press, 1982.

[3] James Geller. *A Knowledge Represenation Theory for Natural Language Graphics*. PhD thesis, Department of Computer Science, State University of New York at Buffalo, 1988.

[4] D. Kumar. An integrated model of acting and inference. In D. Kumar, editor, *Current Trends in SNePS–Semantic Network Processing System: Proceedings of the First Annual SNePS Workshop*, pages 55–65, Buffalo, NY, 1990. Springer-Verlag.

[5] D. Kumar, S. S. Ali, J. Haas, and S. C. Shapiro. The SNePS acting system. In K. E. Bettinger and G. Srikantan, editors, *Proceedings of the Fifth Annual University at Buffalo Graduate Conference on Computer Science*, pages 91–100, 1990.

[6] Deepak Kumar, Syed Ali, and Stuart C. Shapiro. Discussing, using and recognizing plans in SNePS preliminary report—SNACTor: An acting system. In P V S Rao and P Sadanandan, editors, *Modern Trends in Information Technology: Proceedings of the Seventh Biennial Convention of South East Asia Regional Computer Confederation*, pages 177–182. Tata McGraw-Hill, New Delhi, India, 1988.

[7] Deepak Kumar and Stuart C. Shapiro. Architecture of an intelligent agent in SNePS. *SIGART Bulletin*, 2(4):89–92, August 1991.

[8] Deepak Kumar and Stuart C. Shapiro. Modeling a rational cognitive agent in SNePS. In P. Barahona, L. Moniz Pereira, and A. Porto, editors, *EPIA 91: 5th Portugese Conference on Artificial Intelligence, Lecture Notes in Artificial Intelligence 541*, pages 120–134. Springer-Verlag, Heidelberg, 1991.

[9] William C. Mann and Sandra A. Thompson. Relational propositions in discourse. *Discourse Processes*, 9:57–90, 1986.

[10] J. P. Martins and S. C. Shapiro. A model for belief revision. *Artificial Intelligence*, 35(1):25–79, 1988.

[11] * J. G. Neal. *A Knowledge Based Approach to Natural Language Understanding*. PhD thesis, Technical Report 85–06, Department of Computer Science, SUNY at Buffalo, 1985.

[12] S. L. Peters and S. C. Shapiro. A representation for natural category systems. In *Proceedings of the Ninth Annual Conference of the Cognitive Science Society*, pages 379–390. Lawrence Erlbaum Publishers, 1987.

[13] S. L. Peters and S. C. Shapiro. A representation for natural category systems. In *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, pages 140–146, Los Altos, CA, 1987. Morgan Kaufmann.

[14] S. L. Peters, S. C. Shapiro, and W. J. Rapaport. Flexible natural language processing and Roschian category theory. In *Proceedings of the Tenth Annual Conference of the Cognitive Science Society*, pages 125–131. Lawrence Erlbaum Publishers, 1988.

[15] * W. J. Rapaport, S. C. Shapiro, and J. M. Wiebe. Quasi-indicators, knowledge reports, and discourse. Technical Report 86–15, Department of Computer Science, SUNY at Buffalo, 1986.

[16] W. J. Rapaport. Logical foundations for belief representation. *Cognitive Science*, 10:371–422, 1986.

[17] W. J. Rapaport. A knowledge-representation challenge for SNePS. SNeRG Technical Note 20, Department of Computer Science, SUNY at Buffalo, 1988.

[18] William J. Rapaport. Predication, fiction, and artificial intelligence. *Topoi*, 10:79–111, 1991.

[19] S. C. Shapiro. Numerical quantifiers and their use in reasoning with negative information. In *Proceedings of the Sixth International Joint Conference on Artificial Intelligence*, pages 791–796, Los Altos, CA, 1979. Morgan Kaufmann.

[20] S. C. Shapiro. The SNePS semantic network processing system. In N. V. Findler, editor, *Associative Networks: The Representation and Use of Knowledge by Computers*, pages 179–203. Academic Press, New York, 1979.

[21] S. C. Shapiro. Using non-standard connectives and quantifiers for representing deduction rules in a semantic network, 1979. Invited paper presented at Current Aspects of AI Research, a seminar held at the Electrotechnical Laboratory, Tokyo.

[22] S. C. Shapiro. Generalized augmented transition network grammars for generation from semantic networks. *The American Journal of Computational Linguistics*, 8(1):12–25, 1982.

[23] S. C. Shapiro and The SNePS Implementation Group. *SNePS-2 User's Manual*. Department of Computer Science, SUNY at Buffalo, 1989. Superseded by [24].

[24] S. C. Shapiro and The SNePS Implementation Group. *SNePS 2.1 User's Manual*. Department of Computer Science, SUNY at Buffalo, 1994.

[25] S. C. Shapiro, D. Kumar, and S. Ali. A propositional network approach to plans and plan recognition. In *Proceedings of the 1988 Workshop on Plan Recognition*, page 21, Los Altos, CA, 1989. Morgan Kaufmann.

[26] S. C. Shapiro and W. J. Rapaport. SNePS considered as a fully intensional propositional semantic network. In N. Cercone and G. McCalla, editors, *The Knowledge Frontier*, pages 263–315. Springer–Verlag, New York, 1987.

[27] Stuart C. Shapiro, Beverly Woolf, Deepak Kumar, Syed S. Ali, Penelope Sibun, David Forster, Scott Anderson, James Pustejovesky, and Juergen Haas. Discussing, using, and recognizing plans–Project Report. Technical Report RADC-TR-90-404, Volume II (of 18), North-East Artificial Intelligence Consortium, Griffiss Air Force Base, NY, 1990.

[28] N. Sondheimer. Semantic analysis of reference to spatial properties. *Linguistics and Philosophy*, 2:235–280, 1978.

[29] L. Talmy. Figure and ground in complex sentences. In J. Greenberg, C. Ferguson, and E. Moravcik, editors, *Universals of Human Languages*. Stanford University Press, 1978.

[30] J. M. Wiebe and W. J. Rapaport. Representing *de re* and *de dicto* belief reports in discourse and narrative. *Proceedings of the IEEE*, 74(10):1405–1413, 1986.

[31] Albert Hanyong Yuhan. *Dynamic Computation of Spatial Reference Frames in Narrative Understanding*. PhD thesis, Department of Computer Science, State University of New York at Buffalo, 1991.