Part II: Methods of AI

Chapter 3 – Knowledge Representation and Reasoning

3.1 Summary of Logic and Reasoning
3.2 Reasoning: Deduction Systems
3.3 Rulebased Reasoning

3.4 Knowledge Representation: General Issues

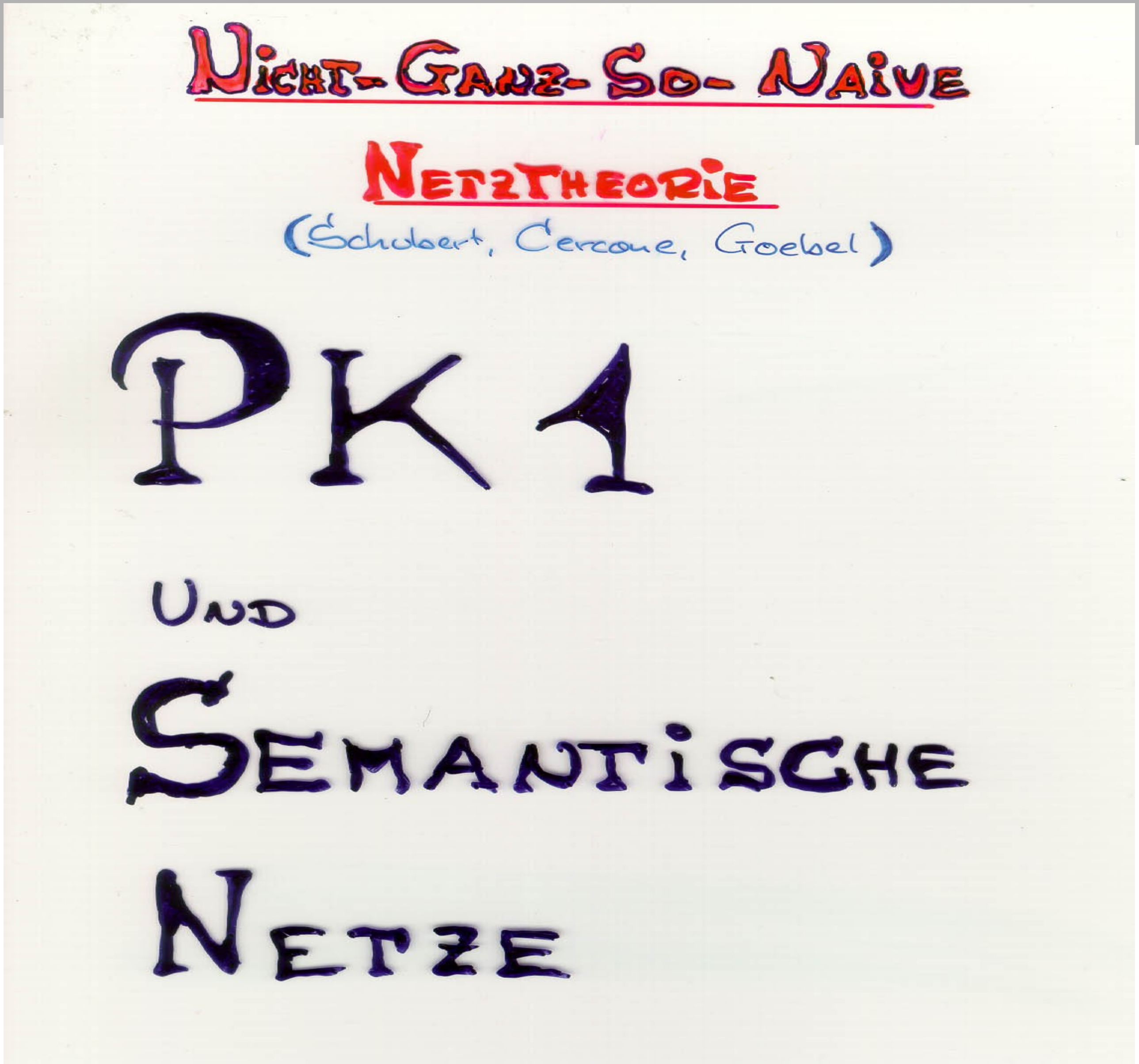
3.5 Knowledge Representation: Semantic Nets (continued)

3.6 Knowledge Representation: Description Logics

3.7 Knowledge Representation: Analogue Representation

3.5 Semantic Nets

Semantic Nets and Predicate Calculus



Nicht-Ganz-So-Naive

Netztheorie

(Schubert, Cercone, Goebel)

PK1

und

Semantische

Netze

Predicate Calculus and Semantic Networks

Semantics of SNs

(i) **Rufus ⟶ is-A ⟶ Dog ⟶ is-A ⟶ Mammal**

Rufus ∈ {Dogs| ...}

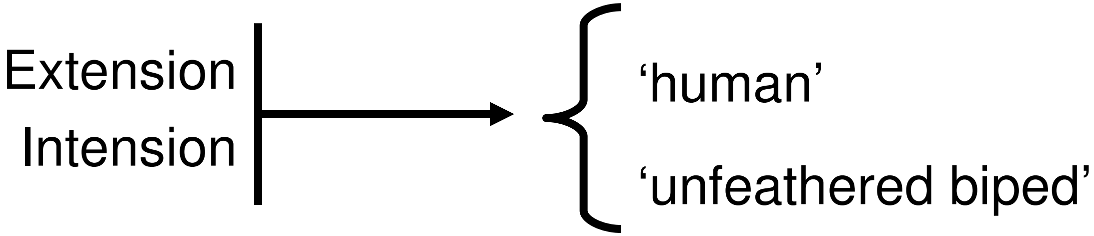
Dog ∈ {Mammal| ...} ?

Rufus = Instance ∈ Instance = Dog ?

(ii)

- Tarski Semantics ? or
- Inferential procedures as semantics?
  (Winograd, Wittgenstein)

(iii)



(iv) SNs: First Order Logic or Higher Order Logic?

Semantics of Semantic Nets

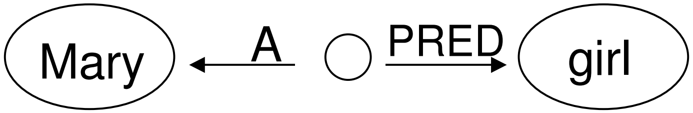
By Translation of Semantic Nets into Predicate Caculus:

IDEA:

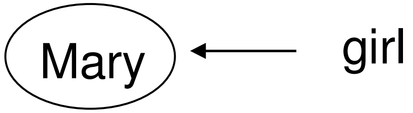
- Getting a uniform notation for the networks
- Translating this into first order PC

N - place Relations (1)

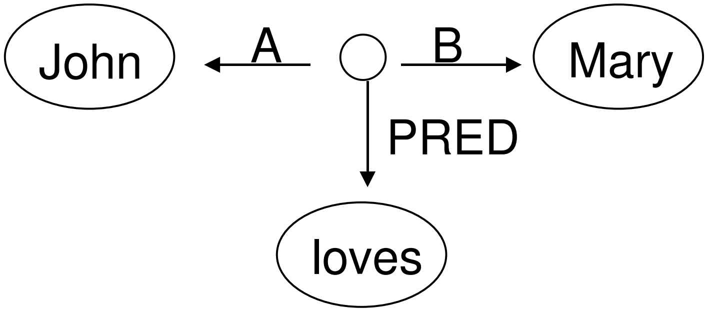
Net **Shortform**



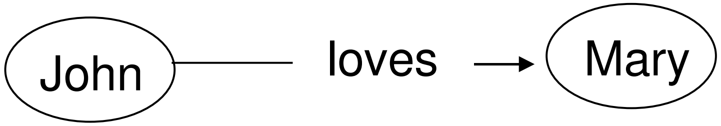
(a) “ Mary is a girl “
[Mary girl]



(b)



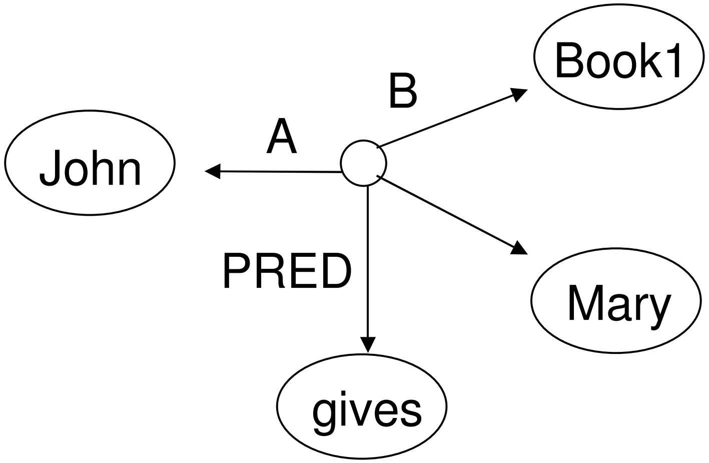
(c) “ John loves Mary “
[John loves Mary]



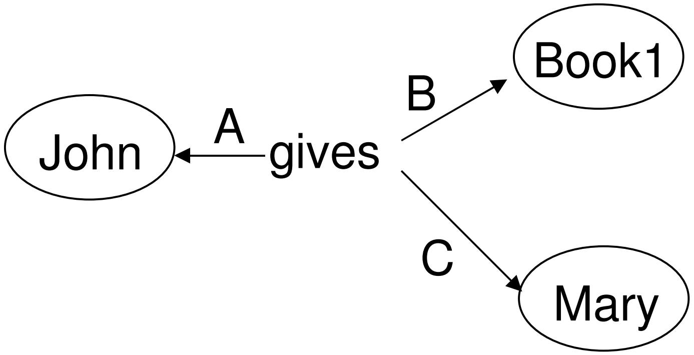
(d)

N - place Relations (2)

Net	**Shortform**



(e) “ John gives the book to Mary “
[John gives Book1 Mary]



(f)

N – place Relations (4)

Remark:

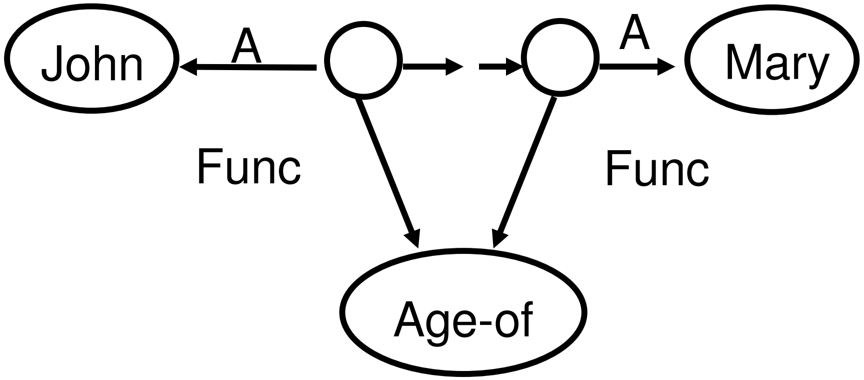
Infix versus Prefix

➡ [John Happy] because [John know [Mary loves John]]:

➡ Because [Happy [John], know [John, loves [Mary, John]]

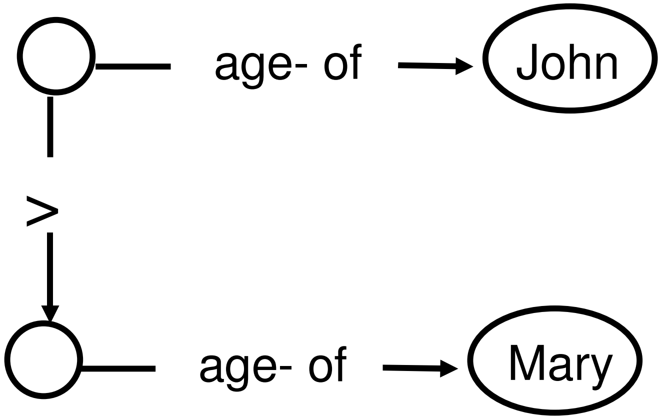
Functions (1)

Functional



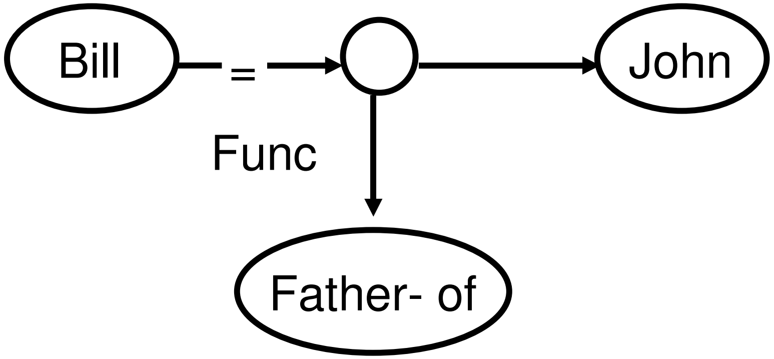
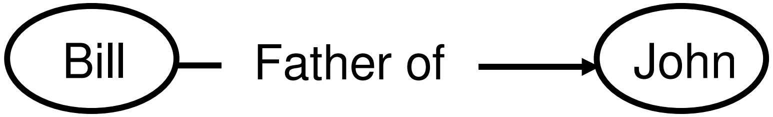
(a) “ John is older than Mary “
[(age- of John) > (age- of Mary)]

Relational

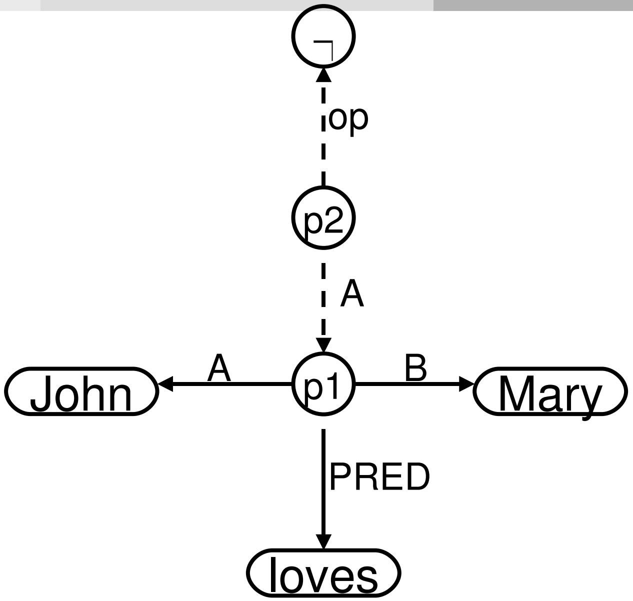


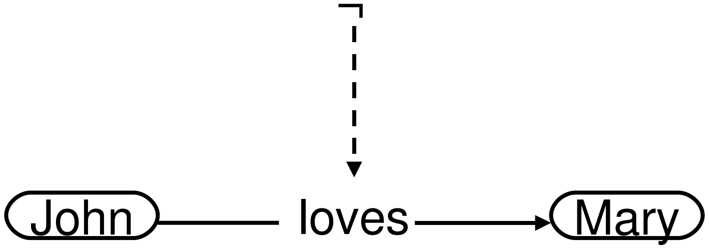
(c) “ John is older than Mary “
[Age1 age- of John] [Age1 > Age2],
[Age2 age- of Mary]

Functions (2)

Functional	Relational
 (b) “ Bill is the father of John “ [Bill = (father- of John)]	 (d) “ Bill is the father of John “ [Bill father- of John]

Logical Operators: NOT



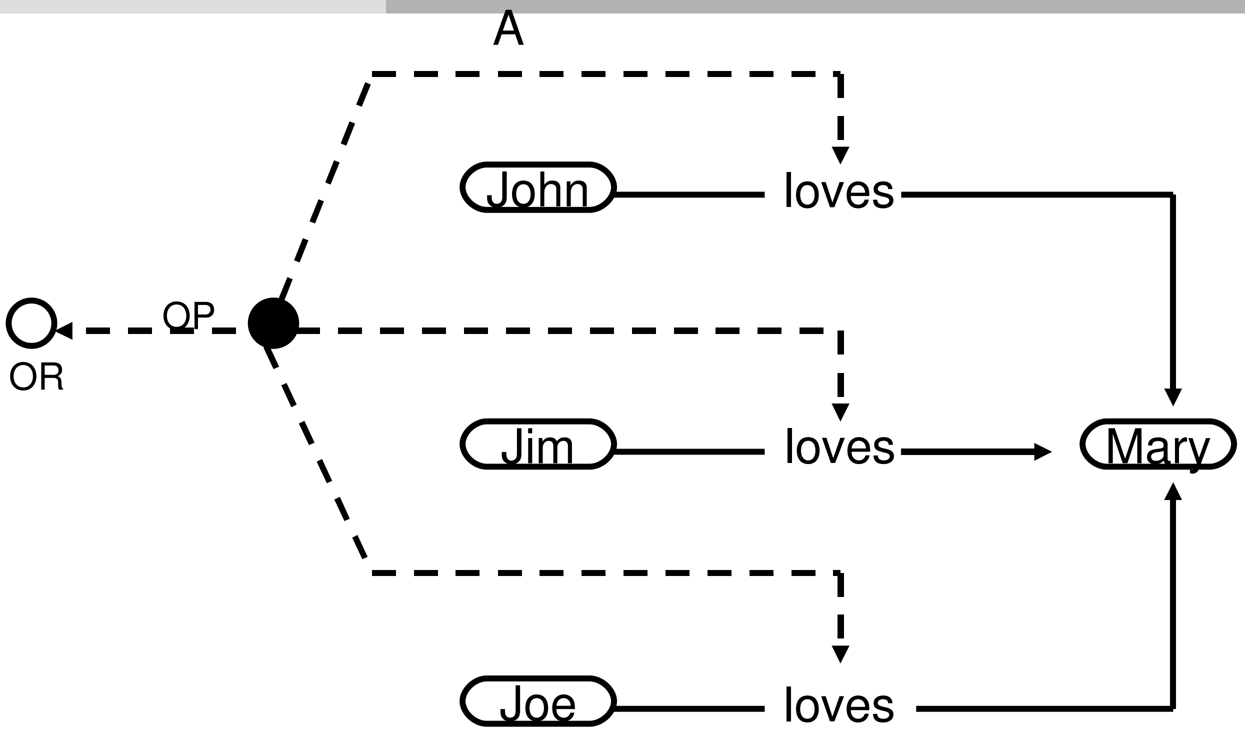


(a) “ John does not love Mary “
¬ [John loves Mary], or perhaps
[[John loves Mary] not]

(b)

OP- Edge
PRED- Edge
FUNC- Edge
ARG- Edge

Logical Operators: OR



"John, Jim or Joe loves Mary"

[[John loves Mary] [Jim loves Mary] or [Joe loves Mary]]

AND analog

Quantors (1)

Praenex form and skolemization

:Example I *'There is a man who likes all women '*

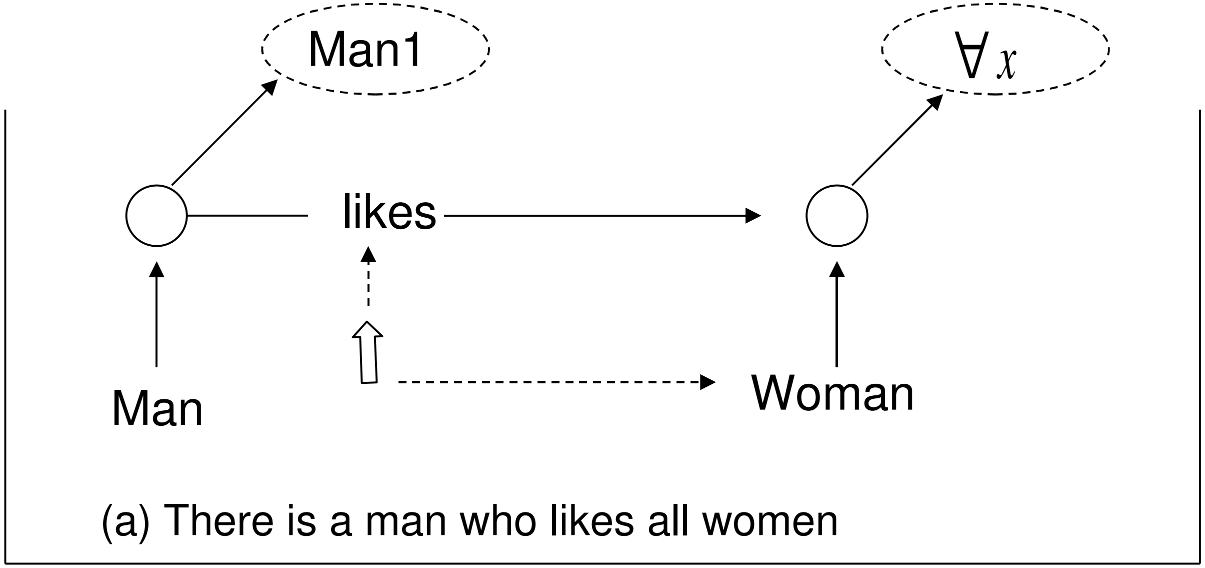
$\exists y \forall x.$ Man (y) $\wedge$ Woman (x) $\Rightarrow$ Likes (y,x)

Skolemization and infix:

[Man1 Man], [x Woman] $\Rightarrow$[Man1, likes x]

Quantors (2)

Example I: Net



(a) There is a man who likes all women

○ Individuum- Nodes
(with or without names)

◌ ∀ - Quantified VARB

Quantors (3)

:Example II *‘ Every man likes some woman '*

$$\forall x . \exists y \; \Big[\, [\, x \text{ Man} \,] \Rightarrow \Big[\, [\, y \text{ Woman} \,] \wedge [\, x \text{ likes } y \,] \Big] \Big]$$

Skolemization:

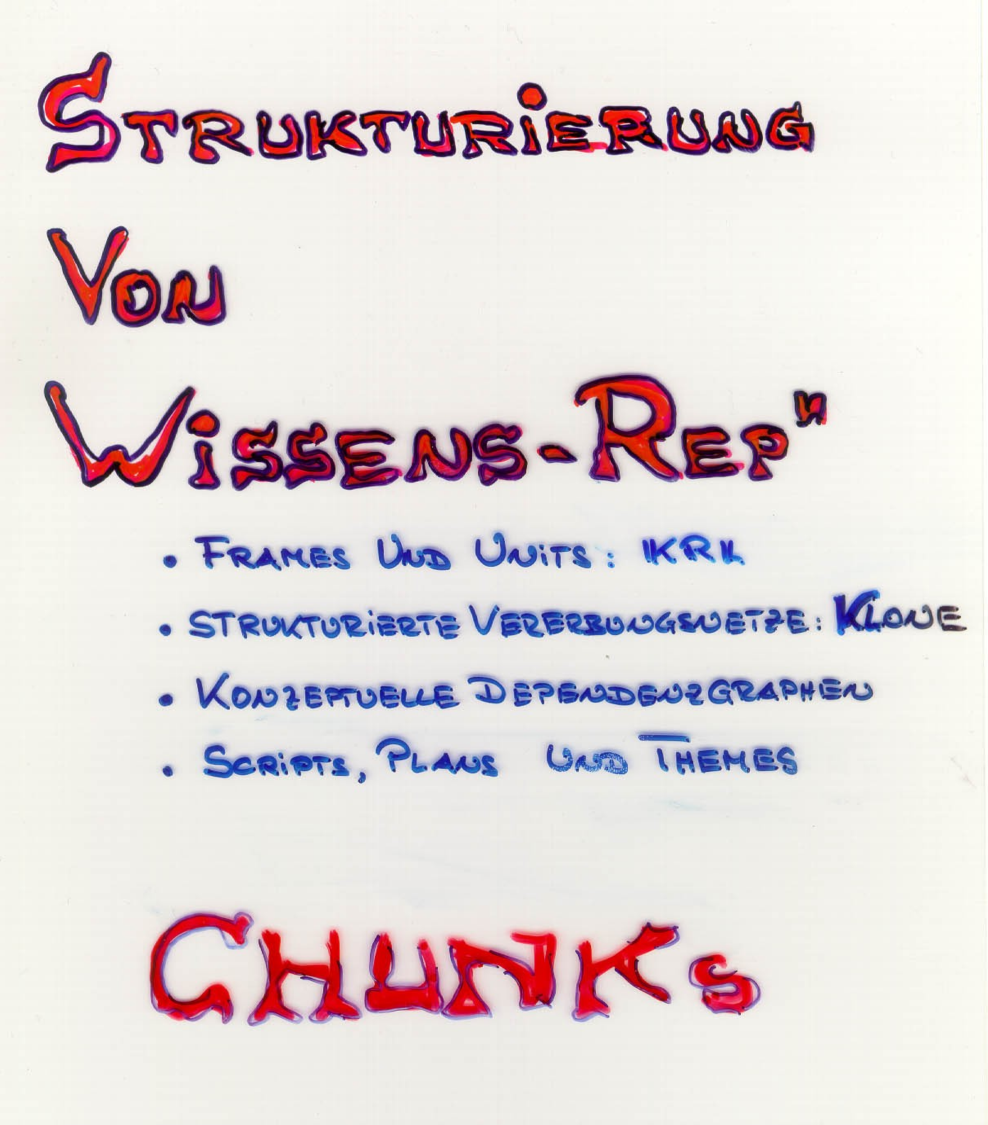
$$\Big[\, [\, x \text{ Man} \,] \Rightarrow \Big[\, [\, y\,(x) \text{ Woman} \,] \wedge [\, x \text{ likes } y\,(\,x\,) \,] \Big] \Big]$$

3.5 Semantic Nets

Structuring a Large Knowledge Base

Problem 2: Structuring a large Knowledge Base

- Type and Token: Referential Nets
- Fast Access: Theme Hierarchy
- Partitioned Networks
- Frames and Units
- Structured Inheritance Networks: KL-One
- Conceptual Dependency Graphs
- Scripts, Plans and Themes

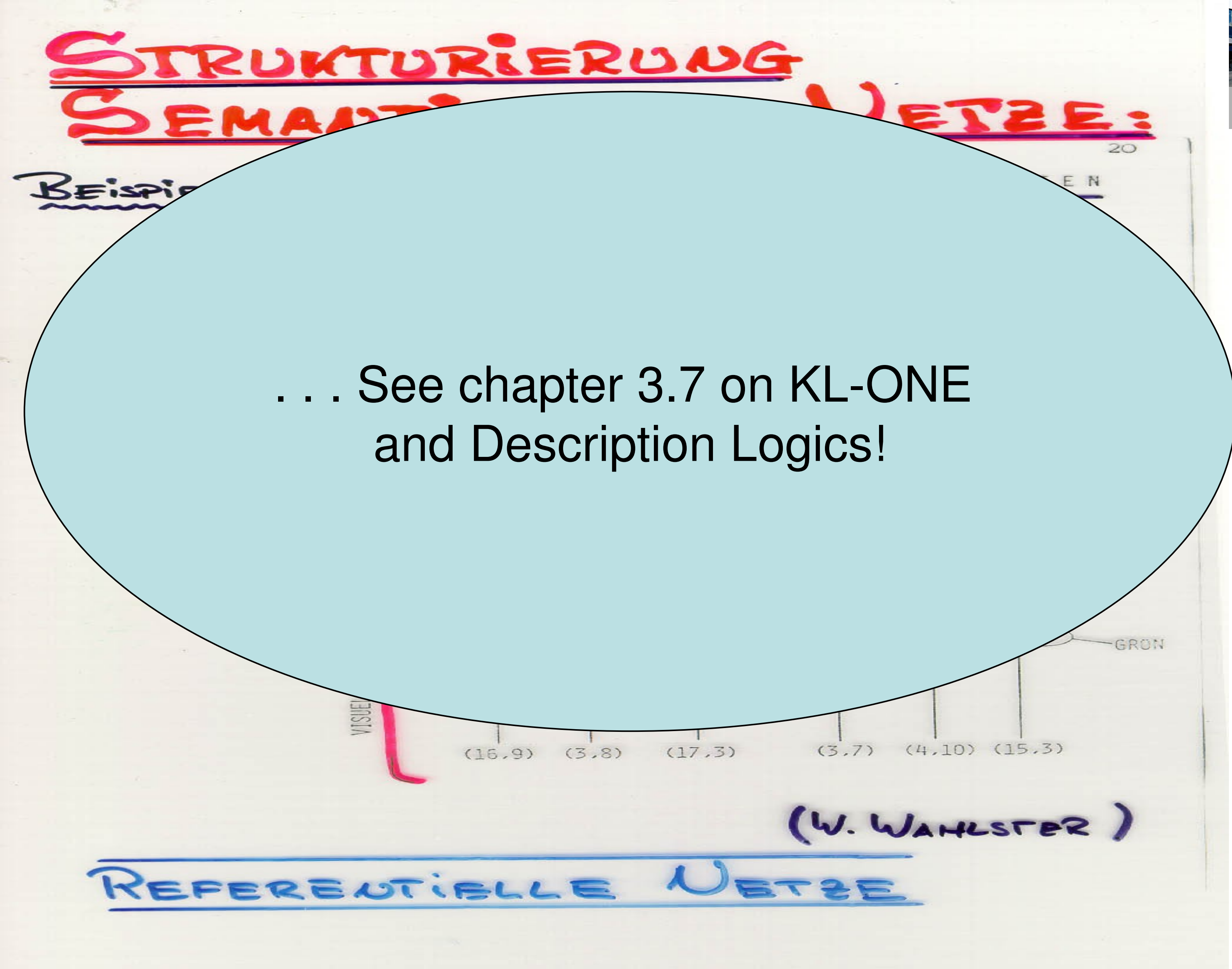


STRUKTURIERUNG VON WISSENS-REP"

- FRAMES UND UNITS: KRL
- STRUKTURIERTE VERERBUNGSNETZE: KLONE
- KONZEPTUELLE DEPENDENZGRAPHEN
- SCRIPTS, PLANS UND THEMES

CHUNKS

Strukturierung Semant Netze:


Beispie
20
EN
GRÜN
VISUEL
(16,9)
(3,8)
(17,3)
(3,7)
(4,10)
(15,3)

. . . See chapter 3.7 on KL-ONE and Description Logics!

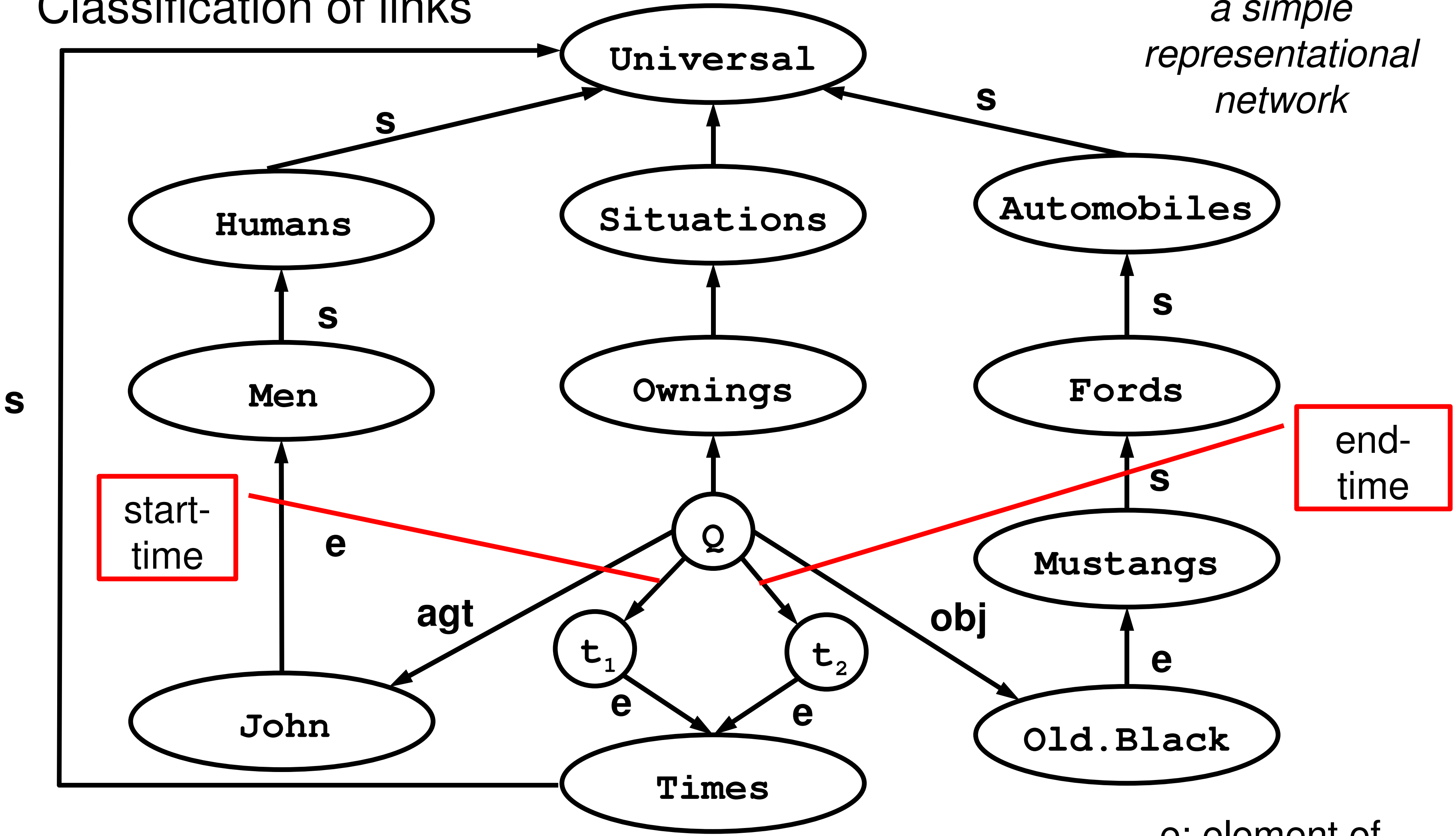
(W. Wahlster)

Referentielle Netze

Partitioned Networks: An Example

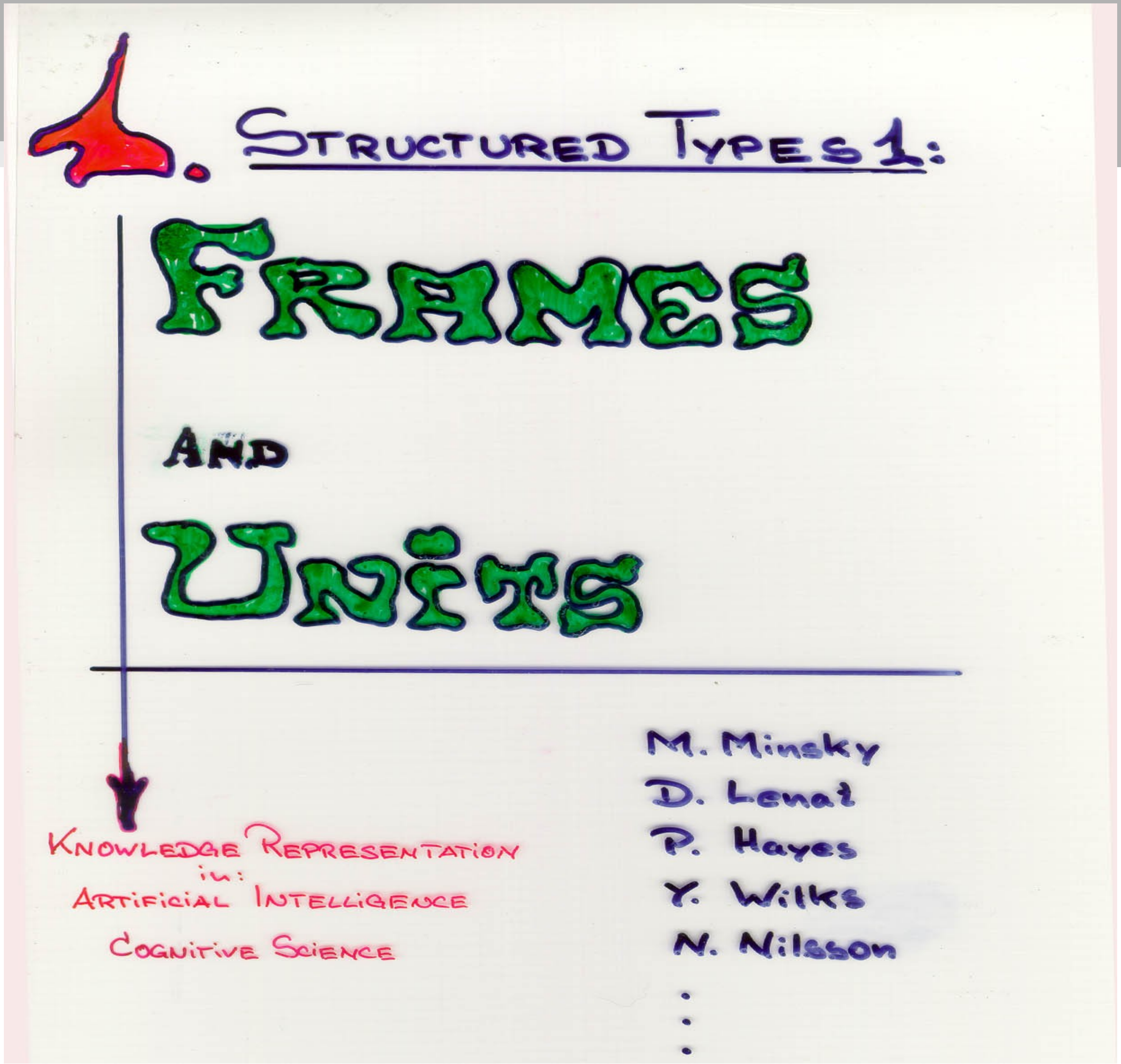
Classification of links

a simple representational network



e: element of

s: subset of



1. Structured Types 1:

Frames and Units

Knowledge Representation in:

Artificial Intelligence

Cognitive Science

M. Minsky

D. Lenat

P. Hayes

Y. Wilks

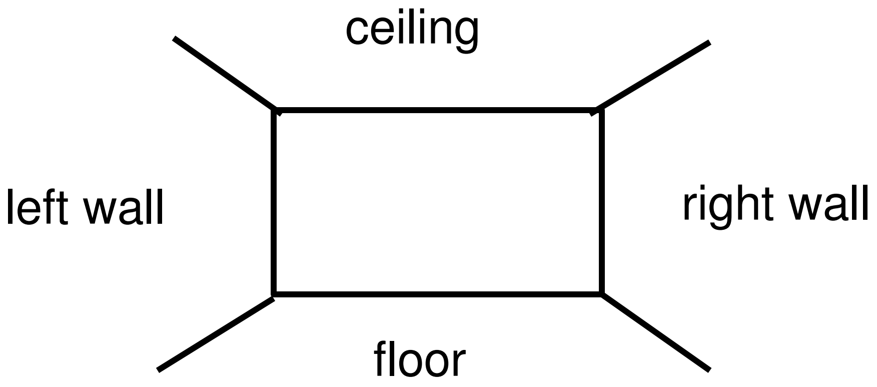
N. Nilsson

⋮

M. Minsky: „A Framework for Representing Knowledge“

How do we see a room?_

The Room Frame:



- Which edges are essential ?
- Which frame in the first place ? Hen-and-Egg Problem!
- When to calculate the details ?
- Which details ?
- When focusing and on what ?   etc.

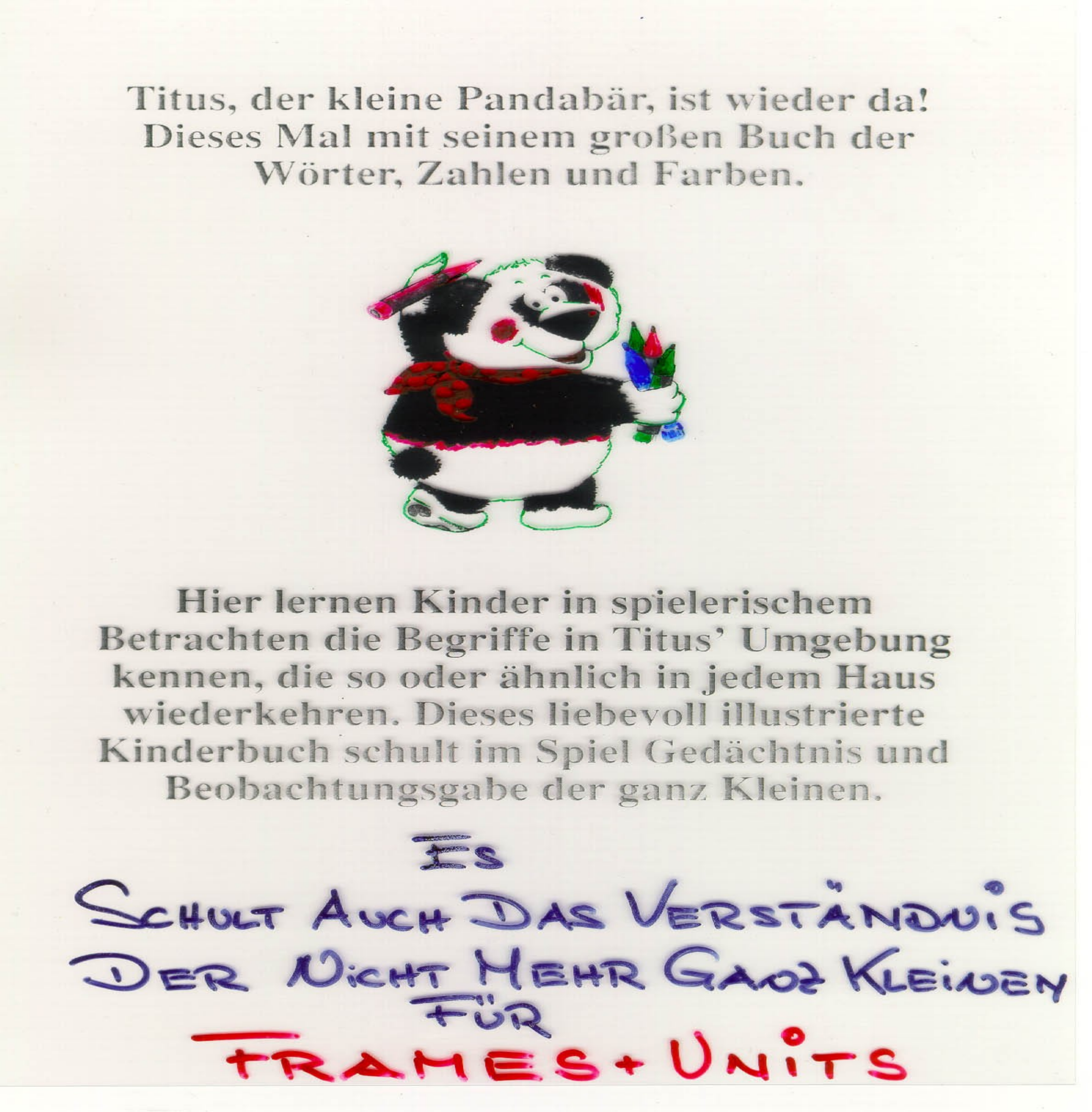
M. Minsky: “A Framework for Representing Knowledge“

- Local/ Global Knowledge Units
- Kuhn‘s Paradigm, Bartlett‘s Scheme
- Frames as Structured Representation of Knowledge:
  - Processing of a Room Scene
  - Terminals, Slots
  - The Baroque Room
  - Picasso, Bacon. . .

FRAMES

An Example is
TITUS, the little Panda

….and the way
children learn!



Titus, der kleine Pandabär, ist wieder da!
Dieses Mal mit seinem großen Buch der
Wörter, Zahlen und Farben.

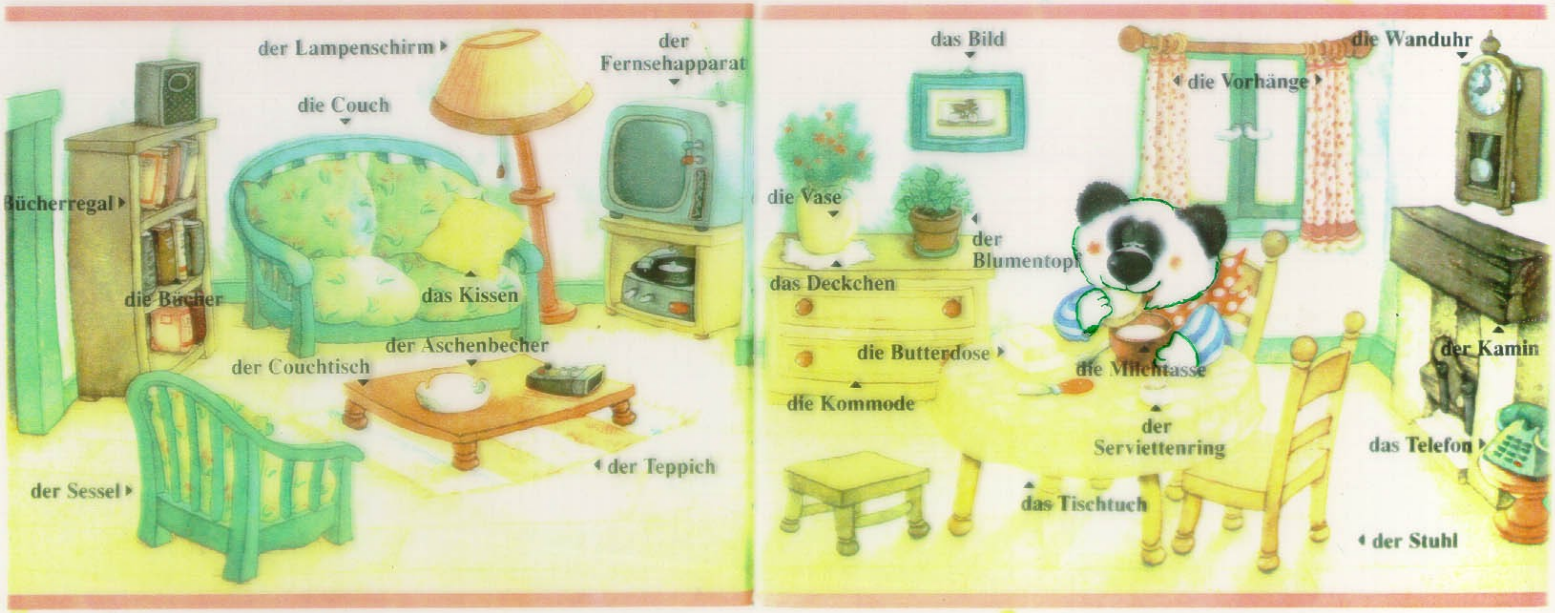
Hier lernen Kinder in spielerischem Betrachten die Begriffe in Titus' Umgebung kennen, die so oder ähnlich in jedem Haus wiederkehren. Dieses liebevoll illustrierte Kinderbuch schult im Spiel Gedächtnis und Beobachtungsgabe der ganz Kleinen.

Es
Schult Auch Das Verständnis
Der Nicht Mehr Ganz Kleinen
Für
Frames + Units

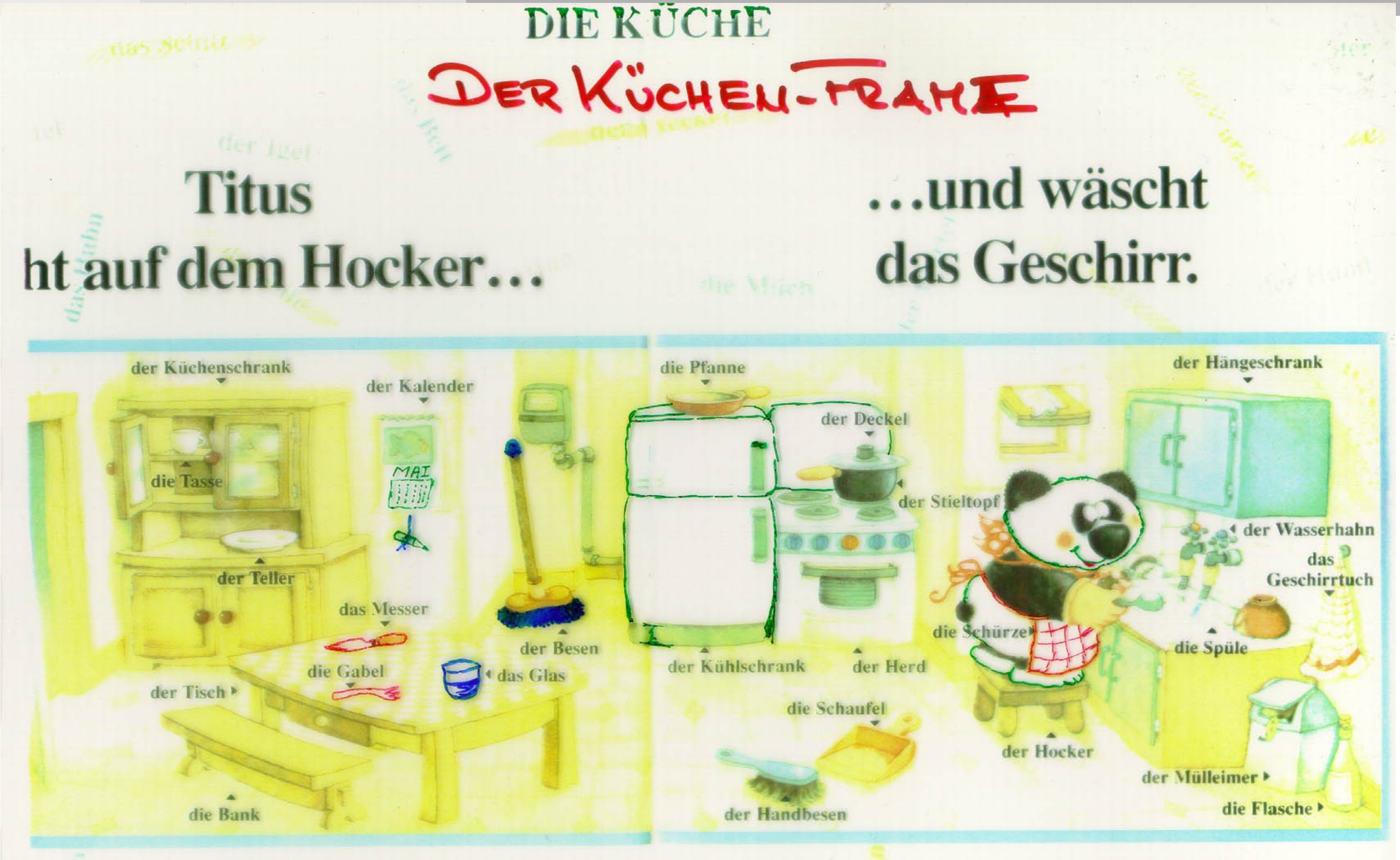
TITUS: The Dining Room Frame

DAS ESSZIMMER - FRAME

Titus frühstückt… …in seinem Eßzimmer.



TITUS: The Kitchen Frame



DIE KÜCHE

DER KÜCHEN-FRAME

Titus
ht auf dem Hocker…

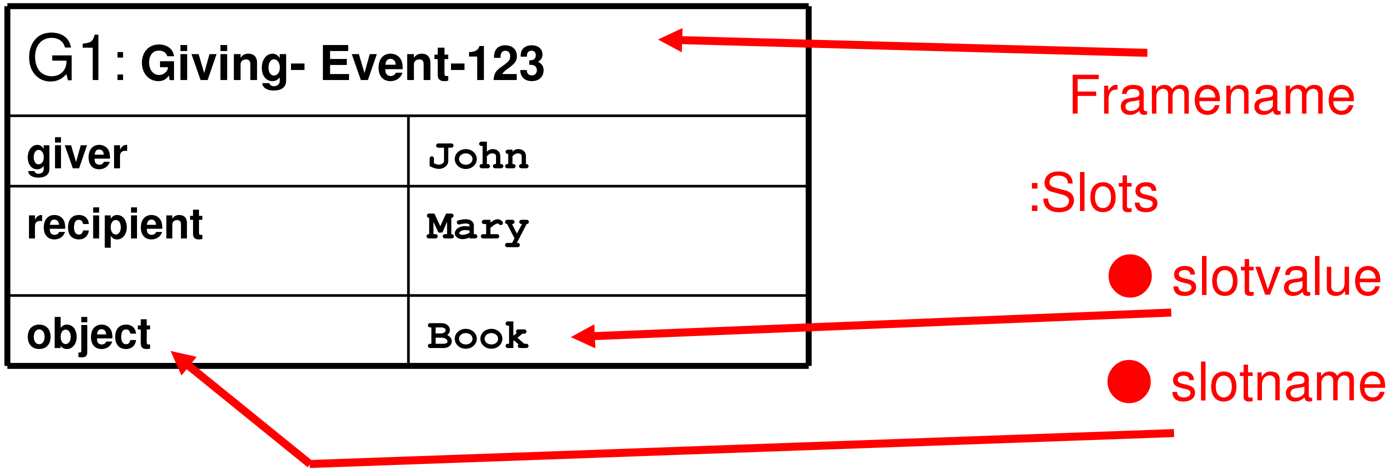
…und wäscht
das Geschirr.

UNIVERSITÄT DES SAARLANDES

Frames: The Data Structure

:Example

John gave Mary the Book



G1: **Giving- Event-123**	
giver	`John`
recipient	`Mary`
object	`Book`

Framename

:Slots

- slotvalue
- slotname

Plus:
- Default-Value
- Frame als SV
- Typeninformation
- Procedural Attachments, Demons

A “Record” in Comp. Science, but. . .

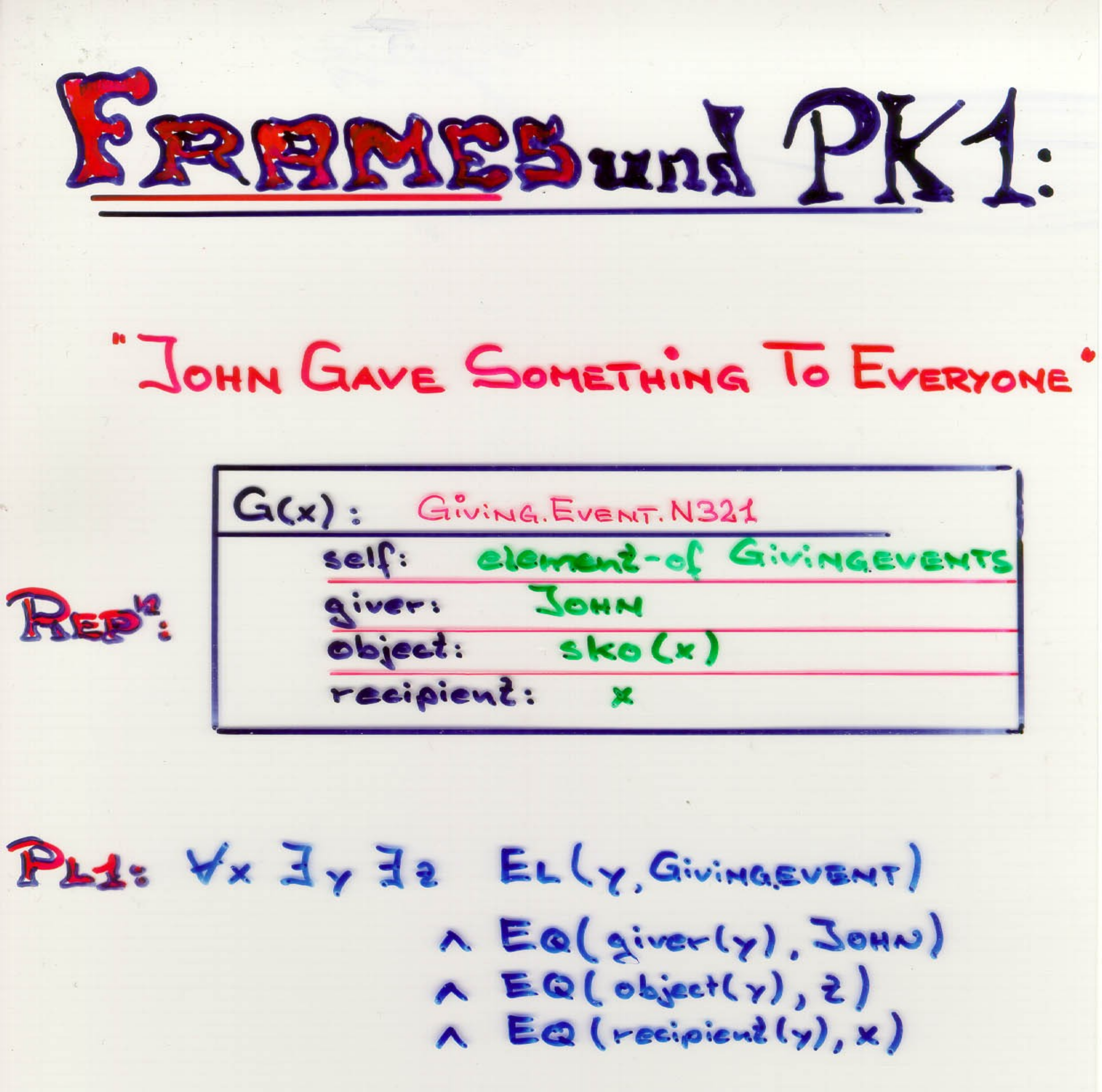
Cross References

" John gave the Book to Mary “ ●

" Bill gave the pen to the person to whom John gave the book “ ●

```
G1:
     EL ( G1, Giving.event )
     giver:         John
     Object:        Book
     recipient:    Mary
```

```
G2:
      EL ( G2, Giving. Event )
      giver:   Bill
      object:  Pen
      recipient: recipient( G1 )
```



FRAMES und PK1:

"John Gave Something To Everyone"

Rep^n:

G(x):	Giving.Event.N321
self:	element-of GivingEvents
giver:	John
object:	sko(x)
recipient:	x

PL1: $\forall x \; \exists y \; \exists z \;\; EL(y, GivingEvent)$
$\wedge \; EQ(giver(y), John)$
$\wedge \; EQ(object(y), z)$
$\wedge \; EQ(recipient(y), x)$

The BIG DEBATE
AI and LOGIC

And its smaller cousin:
Frames and Logic
(Pat Hayes)

Translation of Frames into Predicate Logic

Step1:

- John gave Mary the Book
- John is a Programmer
- Mary is a Lawyer
- John´s Adresse is 37 Maple Street

The Example

An unstructured PC1- Representation:

Step2:

- Give (John, Mary, Book)
- Occupation (John, Programmer)
- Occupation (Mary, Lawyer)
- Adress (John, 37- Maple- Str.)

Object Centered Representation

Step3:

John

Give (John, Mary, Book)

Occupation (John, Programmer)

Adresse (John, 37- Maple- Str.

Mary

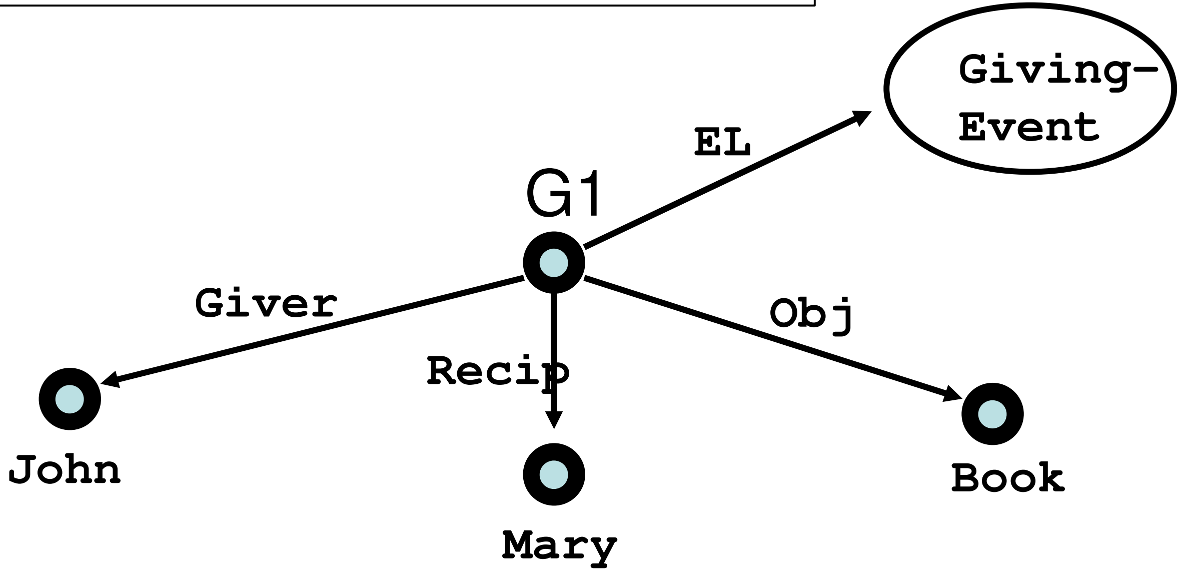
Give (John, Mary, Book)

Occupation (Mary, Lawyer)

Coding into Binary Relations

Step4:

```
EL(G1, Giving.Event)
Giver(G1, John)
Recipient(G1, Mary )
Obj(G1, Book)
```



Using Functions

Step5:

```
EL(G1, Giving - Events)
giver(G1) = John
 recip(G1)= Mary
obj(G1) = Book
```

= : equality infix

Modularized Representation

G1

EL (G1, GIVING - EVENTS)
giver (G1) = JOHN
recip (G1) = MARY
obj (G1) = BOOK

FROM PREDICATE CALCULUS
TO UNITS

Step6:

OC1

EL (OC1, OCCUPATION- EVENTS)
worker (OC1) = JOHN
profession (OC1) = PROGRAMMER

OC2

EL (OC2, OCCUPATION- EVENTS)
worker (OC2) = MARY
profession (OC2) = LAWYER]

ADR

EL (ADR1, ADRESS- EVENTS)
person (ADR1) = JOHN
location (ADR1) = 37- MAPLE-ST

Units or Frames :

G1	
giver:	**John**
recipient:	**Mary**
object:	**Book**

SLOTS:

- SLOT NAME
- SLOT VALUE

Step7:

Final Form: The Frame Representation

G1
self: (element-of GIVING- EVENTS)
giver: JOHN
recip: MARY
obj: BOOK

OC1
self: (element- of OCCUPATION- EVENTS)
worker : JOHN
profession : PROGRAMMER

OC2
self: (element-of OCCUPATION- EVENTS)
worker : MARY
profession : LAWYER

ADR1
self: (element–of ADRESS- EVENTS)
person : JOHN
location : 37- MAPLE-ST

JOHN
self: (element-of PERSONS)

MARY
self: (element-of PERSONS)

BOOK
self: (element-of PHYS-OBJS)

PROGRAMMER
self: (element-of JOBS)

LAWYER
self: (element-of JOBS)

37- MAPLE-ST
self: (element-of ADRESSES)

PERSONS
self: (subset-of ANIMALS)

Step8

Frames: More Advanced Topics

- Variables as Slot-fillers

  (e.g. as Representation of Answers)

- Matching of Frames/ Units
- Rep-Set
- Deductions by Units/ Frames
- Procedural Attachments for Units/ Frames

Variables as Slot-Fillers (1)

Example: **To Whom Did John Give The Book?**

```
    X
       self: (element-of GIVING-EVENTS)
       giver: JOHN
       recip: y
       obj: BOOK
```

Example: **Did Bill Give Mary The Pen?**

```
    X
       self: (element-of GIVING-EVENTS)
       giver: BILL
       recip: MARY
       obj: PEN
```

Variables as Slot-Fillers (2)

MATCH G1: {x ← G1, y ← MARY}

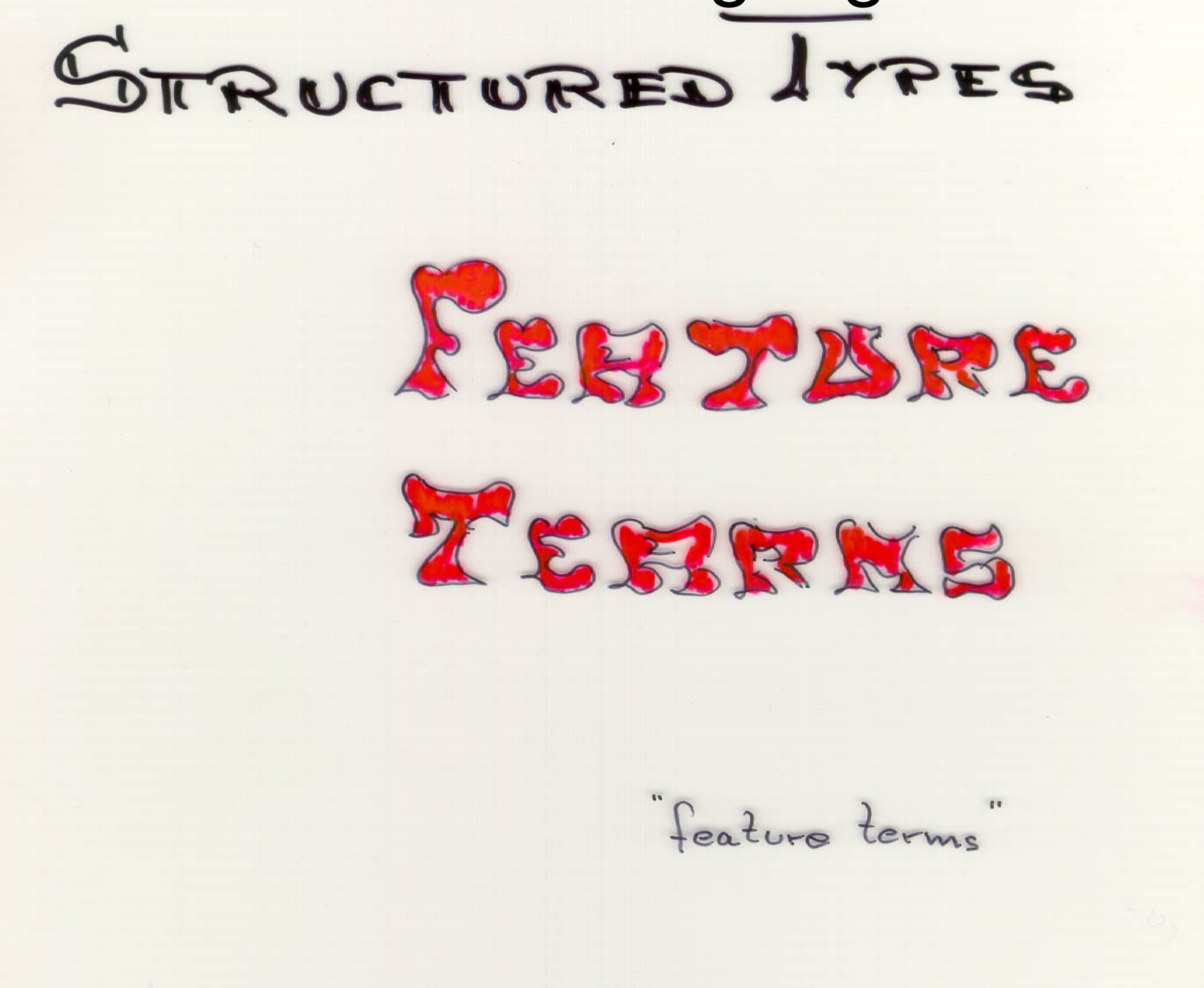
MATCH G2: {x ← G2, MARY = recip(G1)}

Evaluation Function for recip(G1) !

```
    G1
      self: (element-of GIVING-EVENTS)
      giver: JOHN
      recip: MARY
      obj: BOOK
    G2
      self: (element-of GIVING-EVENTS)
      giver: BILL
      recip: recip(G1)
      obj: PEN
```

Excursion (optional)

CoLi: Natural Language Processing



STRUCTURED TYPES

FEATURE TERMS

"feature terms"

A related Data Structure for **Computational Linguistics**

Structured Types: **Features**

Term-Represenation:

Person (John, 23, 170, spouse(Mary, 25))

Feature Structure:

$$\begin{bmatrix} \texttt{type:} & \texttt{person} \\ \texttt{name:} & \texttt{John} \\ \texttt{age:} & \texttt{23} \\ \texttt{height:} & \texttt{170} \\ \texttt{spouse:} & \begin{bmatrix} \texttt{name:} & \texttt{Mary} \\ \texttt{age:} & \texttt{25} \end{bmatrix} \end{bmatrix}$$

Facts: John is 23 Years Old and 170cm Tall.
His Wife Is Mary and she Is 25 Years Old.

Essential Properties of Feature Terms

Labeling of Substructures:

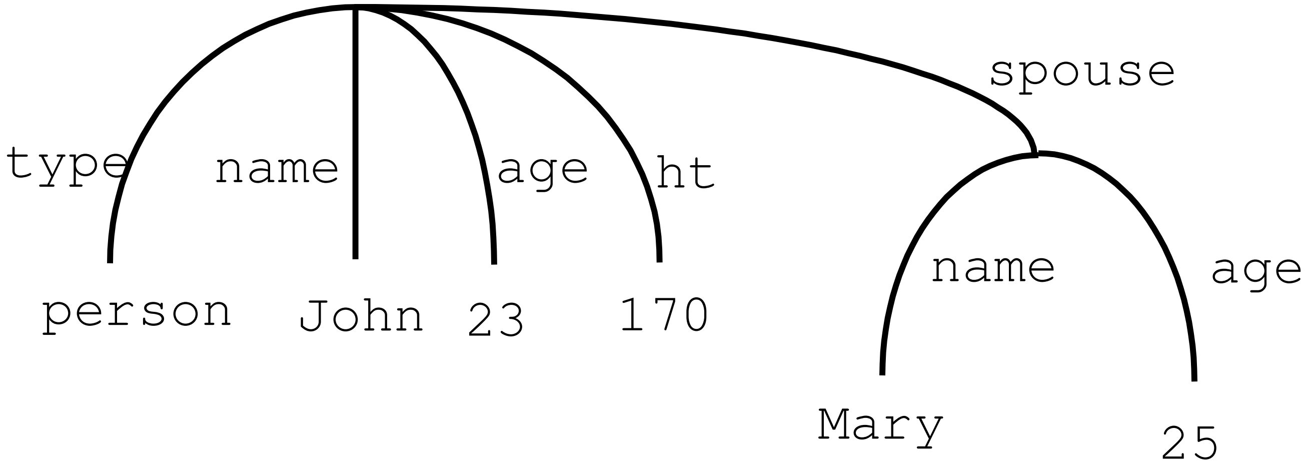
Term: Person (John, 23, 170, spouse(Mary, 25))

Feature Structure:

$$\begin{bmatrix} \texttt{type:} & \texttt{person} \\ \texttt{name:} & \texttt{John} \\ \texttt{age:} & \texttt{23} \\ \texttt{height:} & \texttt{170} \\ \texttt{spouse:} & \begin{bmatrix} \texttt{name:} & \texttt{Mary} \\ \texttt{age:} & \texttt{25} \end{bmatrix} \end{bmatrix}$$

Essential Properties of Feature Terms

Graph Representation:



No Fixed Arity

No Function Argument Distinction

Feature Terms: Variables

Variables:

Placeholder for Future Instantiation

Equality on Subterms: Coreference

:Example *" John has the same age as his wife Mary "*

$$\begin{bmatrix} \texttt{type:} & \texttt{person} \\ \texttt{name:} & \texttt{John} \\ \texttt{age:} & \boxed{1} \\ \texttt{height:} & \texttt{[]} \\ \texttt{spouse:} & \begin{bmatrix} \texttt{name:} & \texttt{mary} \\ \texttt{age:} & \boxed{1} \end{bmatrix} \end{bmatrix}$$

Predicate calculus: Person(John, y, z, spouse(w, y))

Feature Terms: Variables

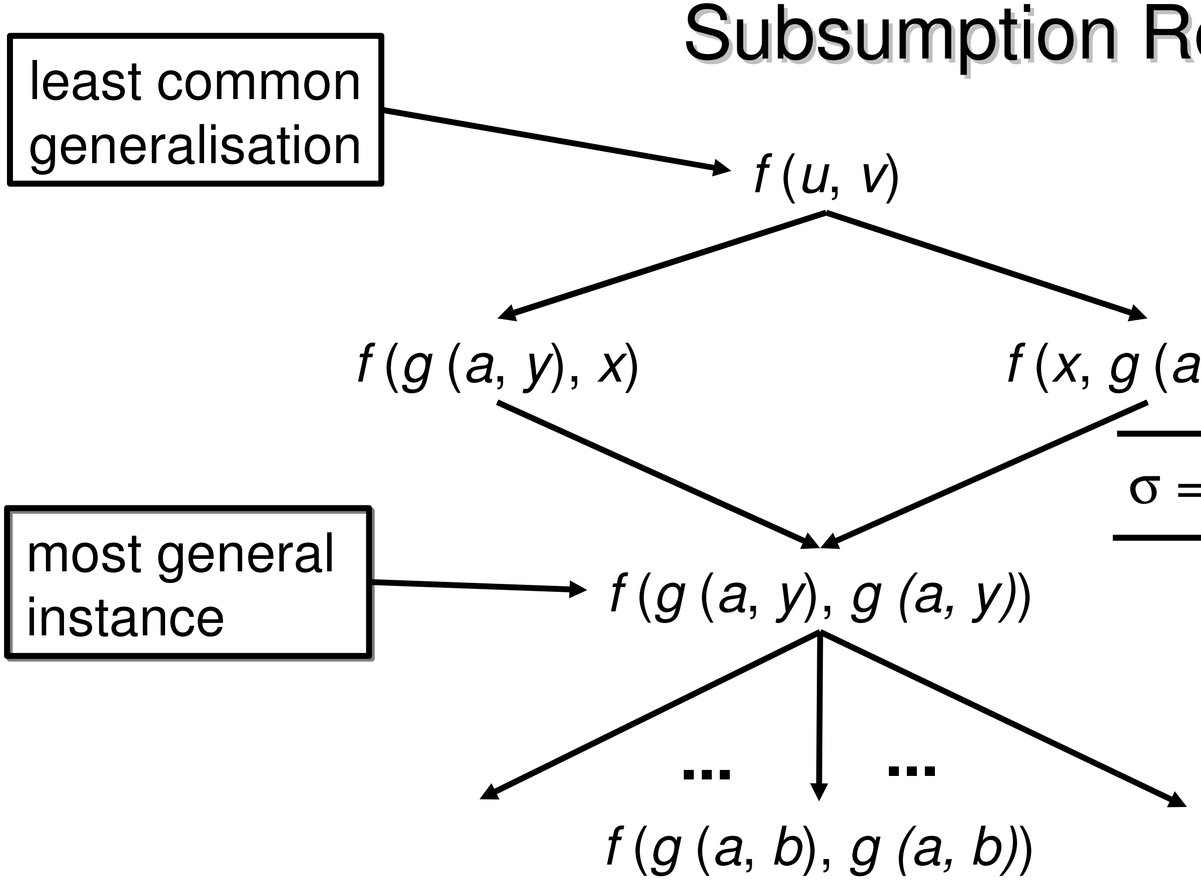
Variables: Equality on Subterms: Coreference

:Example *“ John´s wife is his best friend “*

<u>**Predicate Caculus**</u>: person(John, y, z, spouse(w, x)) and bestfriend(John,w)

$$\left[\begin{array}{ll}
\texttt{type:} & \texttt{person} \\
\texttt{name:} & \texttt{John} \\
\texttt{age:} & \\
\texttt{height:} & \texttt{[]} \\
\texttt{spouse:} & \boxed{\texttt{1}} \left[\begin{array}{ll} \texttt{name:} & \texttt{mary} \\ \texttt{age:} & \end{array}\right] \\
\texttt{bestfriend:} & \boxed{\texttt{1}}
\end{array}\right]$$

The Central Importance of the Subsumption Relation!



least common generalisation

$f\,(u, v)$

$f\,(g\,(a, y), x)$

$f\,(x, g\,(a, z))$

most general instance

$f\,(g\,(a, y), g\,(a, y))$

...

...

$f\,(g\,(a, b), g\,(a, b))$

$$\sigma = \{x \to g\,(a, y), z \to y\}$$

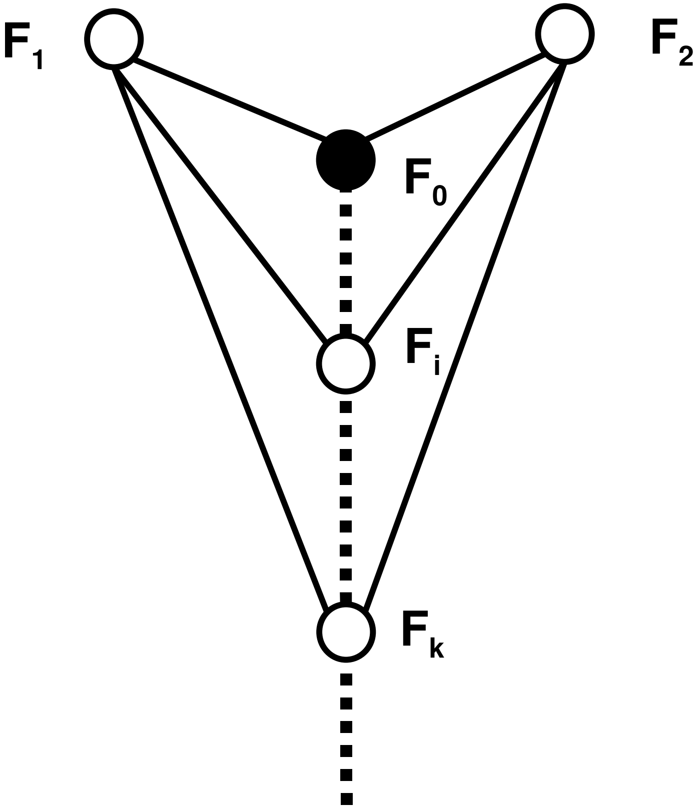
G. Plotkin,
J. Reynolds,
G. Huet

Unification

Idea: Unification is the operation that merges the Information of two features.

Definition:

A feature structure F0 is the unification of two Feature structures F1 and F2, iff F0 is the least Information type that is subsumed by both F1 and F2.



Grammar Rules by “Feature Structures“

The Grammar by Equations:

X0 ⇒ X1 X2

⟨ X0 category ⟩ = S

⟨ X1 category ⟩ = NP

⟨ X2 category ⟩ = VP

⟨ X1 head agreement ⟩ = ⟨ X2 head agreement ⟩

⟨ X0 head ⟩ = ⟨ X2 head⟩

⟨ X0 head mood ⟩ = declarative

Feature structure

$$x0: \quad \begin{bmatrix} \text{category:} & S \\ \text{head:} & 1 \end{bmatrix}$$

$$x1: \quad \begin{bmatrix} \text{category:} & NP \\ \text{head:} & 3 \; [\, \text{agreement: } 2 \,] \end{bmatrix}$$

$$x2: \quad \begin{bmatrix} \text{category:} & VP & \\ \text{head:} & 1 & \begin{bmatrix} \text{agreement:} & 2 \; [\,] \\ \text{subject:} & 3 \\ \text{mood:} & \text{declarative} \end{bmatrix} \end{bmatrix}$$

Parsing Example:
The Vermin Exterminator

Part 1:

$$\begin{bmatrix} \text{category:} & \text{NP} \\ \text{head:} & \begin{bmatrix} \text{det:} & \text{the} \\ \text{root:} & \text{man} \\ \text{agreement:} & \text{singular} \end{bmatrix} \end{bmatrix}$$

The Man

Part 2:

$$\begin{bmatrix} \text{category:} & \text{VP} \\ \text{head:} & \begin{bmatrix} \text{root:} & \text{kill} \\ \text{tense:} & \text{present} \\ \text{agreement:} & \text{singular} \\ \text{object:} & \begin{bmatrix} \text{category:} & \text{NP} \\ \text{head:} & \begin{bmatrix} \text{root:} & \text{bug} \\ \text{agreement:} & \text{plural} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Kills Bugs

Parsing Example: The Vermin Exterminator (2)

$$\begin{bmatrix} \text{x1:} & \begin{bmatrix} \text{category:} & \text{NP} \\ \text{head} & \begin{bmatrix} \text{det:} & \text{the} \\ \text{root:} & \text{man} \\ \text{agreement:} & \text{singular} \end{bmatrix} \end{bmatrix} \\ \text{x2:} & \begin{bmatrix} \text{category:} & \text{VP} \\ \text{head:} & \begin{bmatrix} \text{root:} & \text{kill} \\ \text{tense:} & \text{present} \\ \text{agreement:} & \text{singular} \\ \text{object} & \begin{bmatrix} \text{category:} & \text{NP} \\ \text{head:} & \begin{bmatrix} \text{root:} & \text{bug} \\ \text{agreement:} & \text{plural} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

→ Unification with F-Rule

One Central Operation:
Feature Unification

INPUT: NL surface structure of a sentence

Parsing with a Unification Grammer

(Shieber, Uszkoreit, ..)

The Parse Tree (Feature Structure)

Computing of Meaning

(Fentadt, et al; Kaplan, et al)

The Reprentation of Meaning with Feature Terms

(K1-ONE, SB-ONE,etc)

Natural Language Generation Generation
Based on Unifica Grammers

OUTPUT: NL answer, explanation, dialog etc.

A Feature Unification Machine