Machine Translation

Doug Arnold
University of Essex
doug@essex.ac.uk

February 16, 2006

Outline

1. Translation
2. Approaches/Architectures
3. Implementation (English↔Japanese)
4. Why Translation is difficult
5. Why Machine Translation is hard
6. References
7. Code Listing

Outline (1)

⇒ 1 **Translation**

2 Approaches/Architectures

3 Implementation (English↔Japanese)

4 Why Translation is difficult

5 Why Machine Translation is hard

6 References

7 Code Listing

1 Translation

Take a text in one language (the *source* language) and output an “equivalent” text in another (the *target* language).

Machine Translation is the attempt to automate all or part of the translation activity. Hutchins (1986)

Outline (2)

1 Translation

⇒ 2 **Approaches/Architectures**

3 Implementation (English↔Japanese)

4 Why Translation is difficult

5 Why Machine Translation is hard

6 References

7 Code Listing

2 Approaches/Architectures

Classical

- Direct
- Interlingual
- Transfer

Recent

- Analogical Approaches:
  - Statistical
  - Example Based
- (Constraint Based Approaches)

Outline (3)

1 Translation

2 Approaches/Architectures

⇒ **3 Implementation (English↔Japanese)**

4 Why Translation is difficult

5 Why Machine Translation is hard

6 References

7 Code Listing

Outline (3)

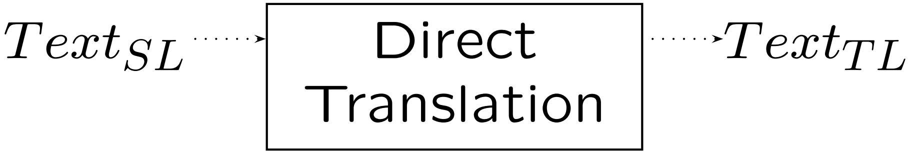


Figure 1: Direct Translation

Outline (3)

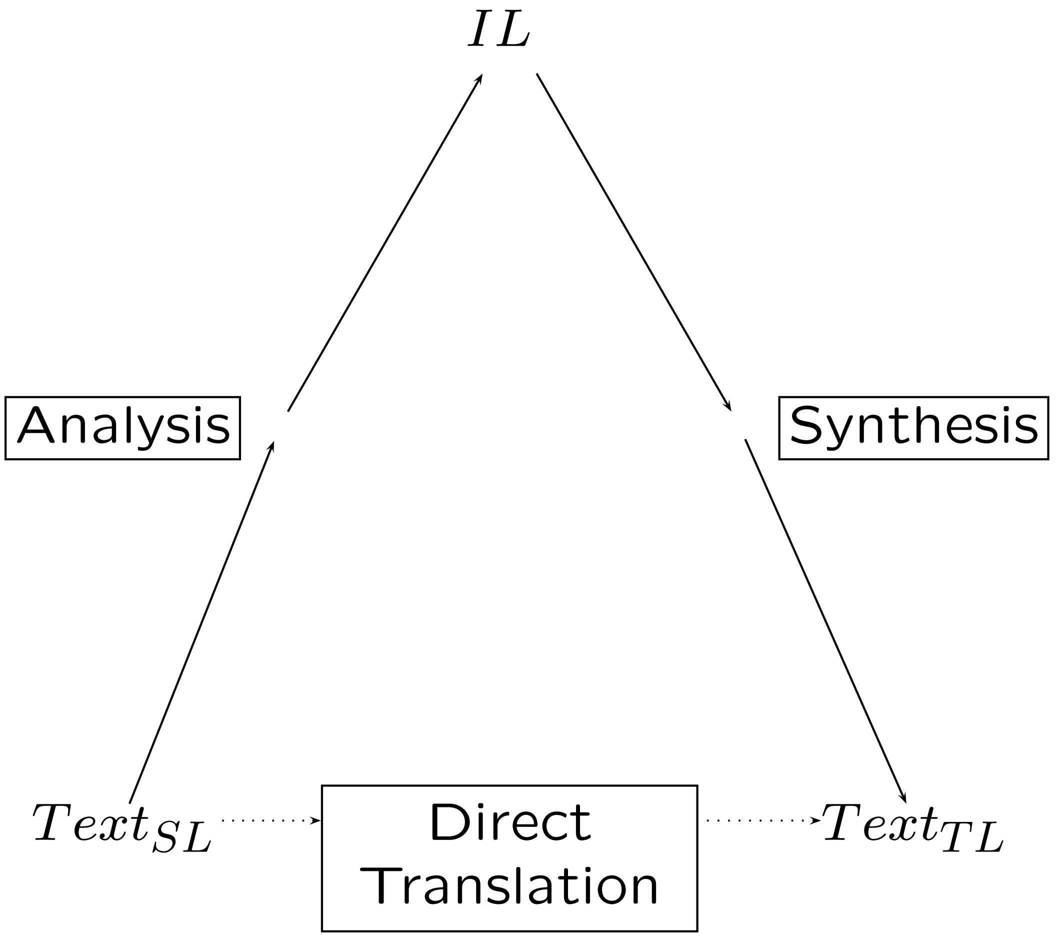


Figure 2: Interlingual Translation

Outline (3)

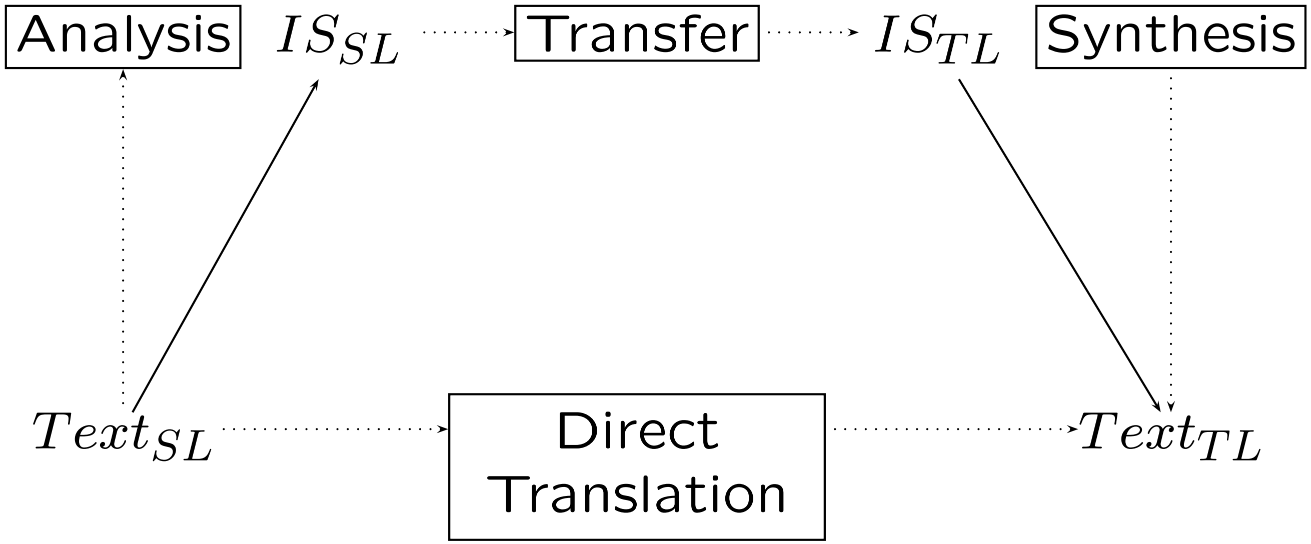


Figure 3: Transfer

Outline (3)

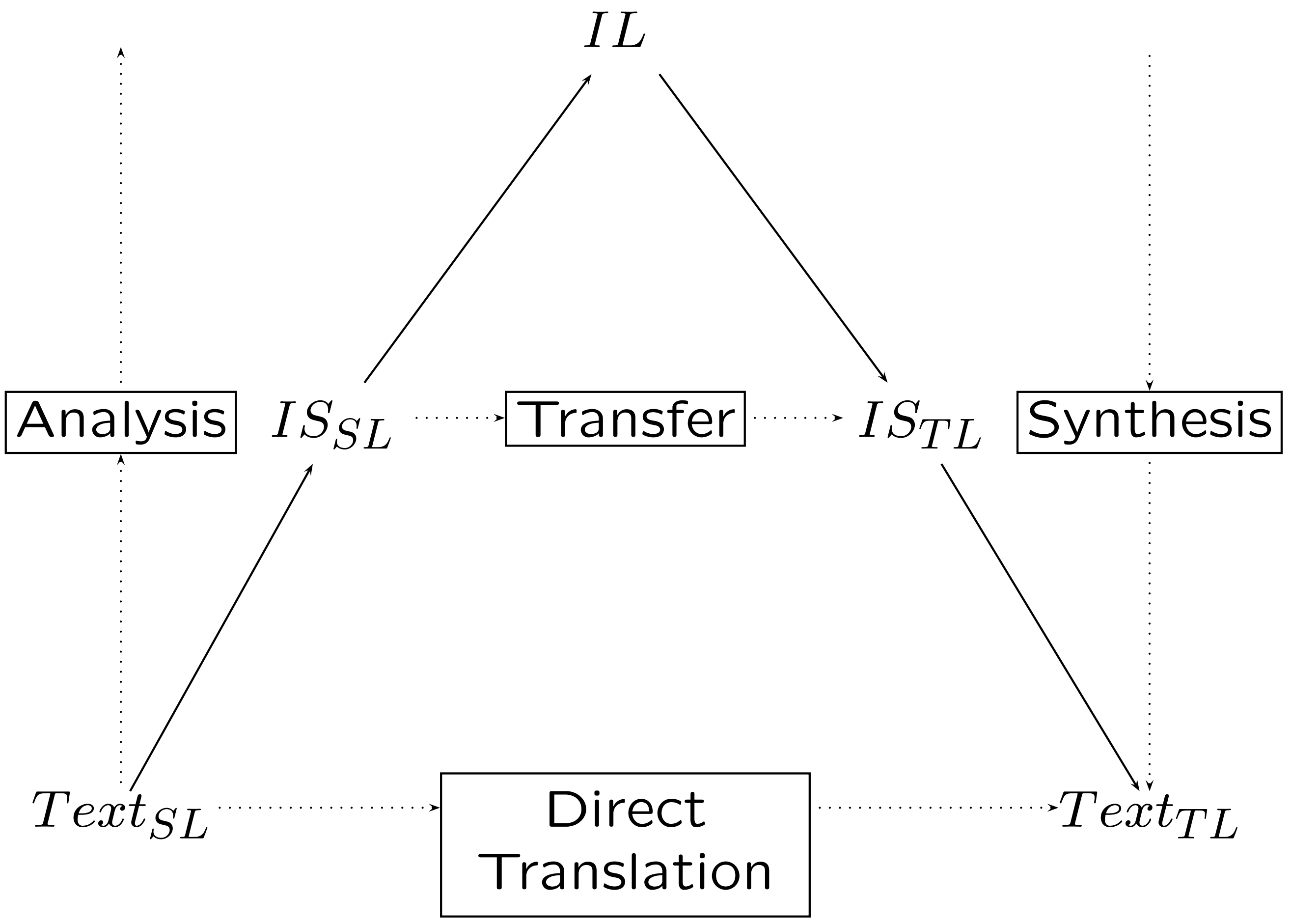


Figure 4: Vauquois' Triangle

3 Implementation (English↔Japanese)

- Interlingual approach;
- Transfer approach

3.1 Interlingual

- parse English sentence to produce IL;
- generate Japanese sentence from IL

We need:

- Interlingua (i.e. a meaning representation language)
- Grammar of English (relating English to the IL);
- Grammar of Japanese (relating Japanese to the IL);
- Parser;
- Generator

In outline:

```
translate(E,J) :-
        parse(English,IL_Rep),
        generate(IL_Rep,Japanese).
```

3.2 Transfer

- parse English sentence to produce IS$_e$;
- transfer IS$_e$ to IS$_j$
- generate Japanese sentence from IS$_j$

We need:

- IS Language(s): IS$_e$ and IS$_j$ (still a meaning representation language, but a bit less abstract than an IL);
- Grammar of English (relating English to the IS$_e$);
- Grammar of Japanese (relating Japanese to the IS$_j$);
- Parser;
- Generator;
- transfer component: rules to relate IS$_e$ to IS$_j$.

In outline:

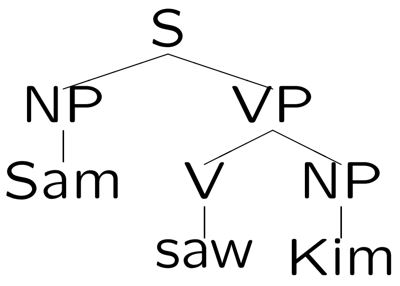
```
translate(E,J) :-
        parse(English,IS_e),
        transfer(IS_e,IS_j),
        generate(IL_j,Japanese).
```

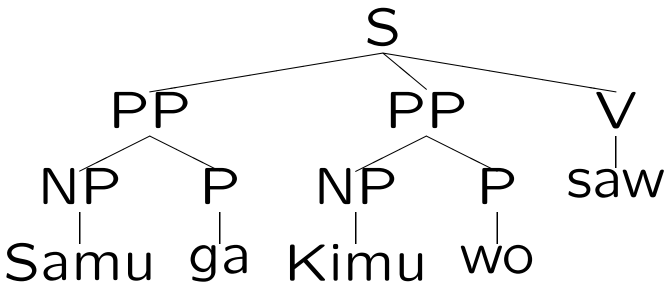
3.3 Japanese/English Grammar

(1) Sam saw Kim

(2) Samu wa Kimu wo mitta.
Sam TOPIC Kim OBJECT saw

(3) Samu ga Kimu wo mitta.
Sam SUBJECT Kim OBJECT saw


S
NP
VP
Sam
V
NP
saw
Kim


S
PP
PP
V
NP
P
NP
P
saw
Samu
ga
Kimu
wo

3.4 Needs

3.4.1 For Interlingual System

IL: Just predicate logic-like: $saw(k, s)$

Parser : ordinary DCG;

Generator : ordinary DGC;

3.4.2 For Transfer System

Parser : ordinary DCG;
Generator : ordinary DGC;
IS: Just surface structure
Transfer : In general:

```
s(Xe,Ye, ..., Ze)  translates_as  s(Xj,Yj, ..., Zj) :-
          Xe translates_as Xj,
          Ye translates_as Yj,
          ...,
          Ze translates_as Zj.
```

With these assumptions, we can build a couple of trivial MT systems.

See section 7 (Prolog code).

. . . DEMO . . .

Outline (4)

1 Translation

2 Approaches/Architectures

3 Implementation (English↔Japanese)

⇒ 4 **Why Translation is difficult**

5 Why Machine Translation is hard

6 References

7 Code Listing

4 Why Translation is difficult

In general, this is a *creative* activity:

- translators have to act as ‘cultural mediators’;
- translations should be “good” (e.g. persuasive, interesting) in their own right;
- novel terms and uses in the source language;

Moreover

- notion of “equivalence” varies;
- even ‘same content’ is hard to get:

French	English
ami	boyfriend, male friend
—	friend
amie	girlfriend, female friend
tu	—
—	you
vous	—

But MT is *possible*:

- METEO — controlled language, near perfect;
- Many Commercial Systems. . . — broad coverage, draft quality:
  - okay for information acquisition;
  - for dissemination revision is needed

. . . And useful (cost, speed, consistency, clarity, less tedium).

Outline (5)

1 Translation

2 Approaches/Architectures

3 Implementation (English↔Japanese)

4 Why Translation is difficult

⇒ **5 Why Machine Translation is hard**

6 References

7 Code Listing

5 Why Machine Translation is hard

- Form under-determines content (Analysis Problem);
- Content under-determines form (Synthesis Problem);
- Languages differ;
- Descriptive Problems:
  - Describing the facts
  - Getting the facts

5.1 The Analysis Problem: form under-determines content

(4) You'll never recognize our Sam — she's grown another foot.

(5) a. Cleaning fluids can be dangerous.
b. Cleaning fluids are dangerous.
c. Cleaning fluids is dangerous.

(6) a. crocodile shoes
b. horse shoes
c. brake shoes

(7) the pumpkin bus

(8) a. Will you have some tea?
b. Thank you.

(9) a. Will you stay at the Otani Hotel?
b. Probably.

(10) pregnant women and mothers (who are nursing) . . .

(11) I saw the soldiers aim at the women, and I saw several fall.

(12) The council refused the women a permit because they advocated violence.

(13) The patients in question were all pregnant at the time. They were asked about their eating habits....

Observation

Any fact at all can be crucial to translation.

5.2 Synthesis Problem: content under-determines form

(14) Sam has a white cat.

(15) a. Sam has a cat. It is white.
b. Sam has a cat which is white.
c. There is a white cat. It's Sam's.
d. There is a white cat. It belongs to Sam.
e. . . .

5.3 Languages Differ

A. Lexical Mismatches

English	Japanese
	kiru (clothes in general)
	haku (shoes)
wear (‘activity’)	kakeru (glasses)
put on (‘action’)	kaburu (hats)
	hameru (gloves)
	haoru (coats)
	shimeru (scarves, ties)

B. Structural Mismatches

(16) a. entrer la salle en courant
‘enter the room in running’
b. run into the room

(17) a. Sam wa Kim sika minakatta.
Sam TOP Kim APART saw-not
Sam did not see anyone apart from Kim.
b. “there is no seeing event, where Sam is the see-er, which is not a seeing of Kim.”

(18) a. Sam saw only Kim.
b. “For every event e, if e is a seeing by Sam, then e is a seeing of Kim.”

5.4 Interlingual Approach

The problems:

- Every distinction made in any language must be represented — this makes the ambiguity problem worse.
- Representations are very abstract — this makes the Synthesis problem worse.
- We still cannot avoid the fact that languages differ — the problem of ‘inference’.

The Ambiguity Problem:

- Japanese uses different words for older/younger sister;
- French has different words for male/female cousin;
- Dutch has different words for ‘runway’ (*startbaan*, *landingsbaan*)

(19) a. The plane took off from the runway.
b. Het vliegtuig steeg op van de startbaan/*landingsbaan.

- The distinctions must be represented;
- The ambiguities must be resolved

The Inference Problem

We cannot guarantee that Target Language Synthesis will be able to handle everything that is produced by source language Analysis:

(20) a. Sam wa Kim sika minakatta.
Sam TOP Kim APART saw-not
Sam did not see anyone apart from Kim.
b. “there is no seeing event, where Sam is the see-er, which is not a seeing of Kim.”

(21) a. Sam saw only Kim.
b. “For every event e, if e is a seeing by Sam, then e is a seeing of Kim.”

We need to be able to perform *inferences*:

- One of the great unsolved problems of modern logic;
- (And why natural languages make very bad Interlinguas)

5.5 Transfer Approach

- Some of the advantages of both Direct and Interlingual Systems
- Some of the disadvantages too:
  - Bilingual rules tend to be *ad hoc* and can be very complex;
  - Transfer tends to preserve the source language structure producing "Translationese";
  - $n * (n - 1)$ transfer components for n languages;
  - Abstract representations, hence....

5.6 Recent Developments

- Example Based Approaches
- Statistical Approaches

5.6.1 Example Based Approaches

Instead of transfer rules, use a database of *examples*:

Julie bought a notebook	Julie compró una libreta
Ann read a book on economics	Ann leyó un libro de economíca

(22) a. Julie bought a book on economics
b. Julie compró un libro de economíca

- a matching algorithm to get the best matches
- an algorithm that ‘de-constructs’ the source side;
- an algorithm that constructs the target side

This poses some difficult problems

5.6.2 Statistical Approaches

‘Noisy channel’ communication, e.g.

- telephony;
- speech recognition;
- cryptography; (!)
- translation. . .

- The ‘noisy channel’ has deformed the English input into French;
- The problem is to recover the English text the French speaker ‘had in mind’

To translate a sentence f into English, we try to find the English expression (e.g. sentence) e which maximizes:

(23) $\hat{e} = argmax_e Pr(e) \times Pr(f|e)$

Intuitively:

- $Pr(f|e)$ is high for English sentences that very often give rise to the French sentence we are to translate;
- $Pr(e)$ is higher for more common sentences of English

This leaves three problems

1. finding data from which to estimate $Pr(e)$ — the *language modeling problem*;
2. finding data from which to estimate $Pr(f|e)$ — *the translation modeling problem*;
3. finding an effective and efficient suboptimal search procedure for the English string that maximizes the product: the *search problem*

The first two need *data* (corpora), alignment algorithms, and training algorithms.

5.7 Some Fun: Mistranslations

(24) a. Les soldats sont dans le café.
b. !The soldiers are in the coffee.

(25) a. Les avocats sont partis.
b. !The avocados have gone.

(26) a. Nous les avions.
b. !We the planes.

Outline (6)

1 Translation

2 Approaches/Architectures

3 Implementation (English↔Japanese)

4 Why Translation is difficult

5 Why Machine Translation is hard

⇒ **6 References**

7 Code Listing

6 References

Arnold, D.J. 2003. Why Translation is difficult for Computers. In Harold Somers (ed.), *Computers and Translation: A translator's guide*, pages 119–142, Amsterdam: John Benjamins.

Arnold, D.J., Balkan, Lorna, Meijer, Siety, Humphreys, R.Lee and Sadler, Louisa. 1993. *Machine Translation: an Introductory Guide*. London: Blackwells-NCC.

Dorr, Bonnie Jean. 1993. *Machine Translation: A View from the Lexicon*. Cambridge, Mass: The MIT Press.

Hutchins, W J and Somers, H L. 1992. *An Introduction to Machine Translation*. Academic Press.

Hutchins, W. John. 1986. *Machine Translation: Past, Present, Future*. Chichester/New York: Ellis Horwood/Wiley.

Kay, Martin, Gawron, Jean Mark and Norvig, Peter. 1994. *Verbmobil: a translation system for face to face dialog*. CSLI Lecture Notes, No. 33, Stanford, CA: CSLI.

King, Margaret (ed.). 1987. *Machine Translation Today: The State of the Art*. Edinburgh: Edinburgh University Press, proceedings of the Third Lugano Tutorial, 1984.

McCord, Michael C. 1989. Design of LMT: A Prolog-Based Machine Translation System. *Computational Linguistics* 15(1), 33–52.

Nirenberg, S. (ed.). 1987. *Machine Translation: Theoretical and Methodological Issues*. Cambridge: CUP.

Rosetta, M.T. 1994. *Compositional Translation*. Dordrecht: Kluwer Academic Publishers.

Trujillo, Arturo. 1999. *Translation Engines: techniques for Machine Translation*. Berlin: Springer-Verlag.

Whitelock, P.J. and Kilby, K.J. 1995. *Linguistic and Computational Techniques in Machine Translation System Design*. London: UCL Press.

Outline (7)

1 Translation

2 Approaches/Architectures

3 Implementation (English↔Japanese)

4 Why Translation is difficult

5 Why Machine Translation is hard

6 References

⇒ **7 Code Listing**

7 Code Listing

Program 1 *transfer.pl*

```
%% -*- mode:prolog; mode:font-lock -*-
%% ----------------------------------------------------
%% File:    English-Japanese transfer based MT
%% Author:   <doug@s2159.essex.ac.uk>
%% Date:    Tue Feb 29 2000
%% Time-stamp: <03/02/21 12:55:46 doug serlinux33.essex.ac.uk transfer.pl>
%% ----------------------------------------------------

% All categories bear a ‘language’ feature (e or j)
% and a (syntactic) ‘representation’ feature
% Vs have a subcat feature 1 or 2
% Japanese PPs have a ‘Pform’ feature.

               % English Grammar
s(e,s(NP,VP)) -->
        np(e,NP),
        vp(e,_,VP).
```

```
np(e,np(sam))  --> [sam].
np(e,np(kim))  --> [kim].

vp(e,1,vp(V)) -->
        v(e,1,V).
vp(e,2,vp(V,NP)) -->
        v(e,2,V),
        np(e,NP).

v(e,1,v(go))   --> [went].
v(e,1,v(come)) --> [came].

v(e,2,v(see))  --> [saw].
v(e,2,v(love)) --> [loved].

% S = [kim,loved,sam], s(e,R,S,[]).
% S = [sam,came], s(e,R,S,[]).
% R = s(np(sam),vp(v(love),np(kim))), s(e,R,S,[]).
% R = s(np(sam),vp(v(go))), s(e,R,S,[]).
```

Code Listing

```
               % Japanese Grammar
s(j,s(NP1,NP2,V)) -->
        pp(j,ga,NP1),
        pp(j,wo,NP2),
        v(j,2,V).
s(j,s(NP,V)) -->
        pp(j,ga,NP),
        v(j,1,V).

pp(j,X,NP) -->
        np(j,NP),
        p(j,X,_).
p(j,wo,_) --> [wo].
p(j,ga,_) --> [ga].

np(j,np(samu))  --> [samu].
np(j,np(kimu))  --> [kimu].

v(j,1,v(iku))  --> [ikimashita].
v(j,1,v(kuru)) --> [kimashita].
```

Code Listing

```
v(j,2,v(miru))    --> [mitta].
v(j,2,v(ai_suru)) --> [ai,simashita].

% s(j,R,[samu,ga,kimu,o,mimashita],[]).
% s(j,R,[samu,ga,ikimashita],[]).
% s(j,s(np(samu),v(iku)),S,[]).
% s(j,s(np(samu),np(kimu),v(iku)),S,[]).

               % Transfer Rules
?- op(10,xfx, =>).

s(NP1e,vp(Ve,NP2e))
  =>
s(NP1j,NP2j,Vj) :-
        NP1e  => NP1j,
        NP2e  => NP2j,
        Ve  => Vj.

s(NP1e,vp(Ve))
  =>
s(NP1j,Vj) :-
```

Code Listing

```
        NP1e  => NP1j,
        Ve  => Vj.

np(sam)  => np(samu).
np(kim)  => np(kimu).
v(see)   => v(miru).
v(love)  => v(ai_suru).
v(come)  => v(kuru).
v(go)    => v(iku).

               % MT System
e2j(Eng,Jap) :-
        s(e,ISe,Eng,[]),  % analysis
        ISe  => ISj,      % transfer
        s(j,ISj,Jap,[]).  % synthesis
j2e(Jap,Eng) :-
        s(j,ISj,Jap,[]),   % analysis
        ISe  => ISj,       % transfer
        s(e,ISe,Eng,[]).   % synthesis

% e2j([sam,saw,kim],Jap).
% j2e([samu,ga,kimu,wo,mitta],Eng).
% e2j(Eng,Jap).
%% ----- end  ------------------------------------------
```

Program 2 *il.pl*

```
%% -*- mode:prolog; mode:font-lock -*-
%% -----------------------------------------------------
%% File:    English-Japanese Interlingual MT
%% Author:   <doug@s2159.essex.ac.uk>
%% Date:    Tue Feb 29 2000
%% Time-stamp: <03/02/21 11:46:24 doug serlinux33.essex.ac.uk il.pl>
%% -----------------------------------------------------

% All categories bear a ‘language’ feature (e or j)
% and a (semantic) representation feature
% Vs have a subcat feature 1 or 2
% Japanese PPs have a ‘Pform’ feature.

               % English Grammar
s(e,Sem) -->
```

Code Listing

```
        np(e,NP),
        vp(e,_,NP^Sem).
np(e,s)  --> [sam].
np(e,k)  --> [kim].

vp(e,1,V) -->
        v(e,1,V).
vp(e,2,Sem) -->
        v(e,2,NP^Sem),
        np(e,NP).

v(e,1,X^go(X))   --> [went].
v(e,1,X^come(X)) --> [came].

v(e,2,X^Y^see(Y,X))   --> [saw].
v(e,2,X^Y^love(Y,X))  --> [loved].

% S = [kim,loved,sam], s(e,R,S,[]).
% S = [sam,came], s(e,R,S,[]).
% R = s(np(sam),vp(v(love),np(kim))), s(e,R,S,[]).
% R = s(np(sam),vp(v(go))), s(e,R,S,[]).
```

Code Listing

```
               % Japanese Grammar
s(j,Sem) -->
        pp(j,ga,Subj),
        pp(j,wo,Obj),
        v(j,2,Obj^Subj^Sem).
s(j,Sem) -->
        pp(j,ga,Subj),
        v(j,1,Subj^Sem).

pp(j,X,Sem) -->
        np(j,NP),
        p(j,X,NP^Sem).
p(j,wo,P^P) --> [wo].
p(j,ga,P^P) --> [ga].

np(j,s)  --> [samu].
np(j,k)  --> [kimu].

v(j,1,X^go(X))   --> [ikimashita].
```

Code Listing

```
v(j,1,X^come(X)) --> [kimashita].

v(j,2,X^Y^see(Y,X))  --> [mitta].
v(j,2,X^Y^love(Y,X)) --> [ai,shimashita].

% s(j,R,[samu,ga,kimu,o,mimashita],[]).
% s(j,R,[samu,ga,ikimashita],[]).
% s(j,s(np(samu),v(iku)),S,[]).
% s(j,s(np(samu),np(kimu),v(iku)),S,[]).

               % MT System
e2j(Eng,Jap) :-
        s(e,IL,Eng,[]),  % analysis
        s(j,IL,Jap,[]).  % synthesis
j2e(Jap,Eng) :-
        s(j,IL,Jap,[]),   % analysis
        s(e,IL,Eng,[]).   % synthesis

% e2j([sam,saw,kim],Jap).
% j2e([samu,ga,kimu,wo,mitta],Eng).
% e2j(Eng,Jap).
```

Code Listing

```
%% ----- end  ----------------------------------------
```