

---

## Transformation Kit

---

Each transformation rule is in the following form.

```
(goal-object <== source-object
  (list-of-equations))
```

The list-of-equations is a list of pseudo equations written in exactly the same way as in the GenKit Pseudo Unification Grammar.

The COMPTRF function compiles each rule and produces a lisp program.

e.g.

```
(DEFUN GOAL-OBJECT-from-SOURCE-OBJECT (X1) ...
)
```

Example Rule in trans-bird.gra:

```
(jp-bird <== eng-bird
  ((x0 = x1)
   ((x1 root) = fly)
   ((x0 root) <= 'tobu)
   ((x1 subj case) = nominative)
   ((x0 subj p root) <= 'ga)
   ((x1 subj root) = bird)
   ((x0 subj root) <= 'tori)))
```

After compiling the "trans-bird.gra" file, the file,  
"trans-bird\_trf.lisp" is created:

```
(DEFUN JP-BIRD-from-ENG-BIRD (X1) (LET ((X (LIST
(LIST (LIST (QUOTE X1) X1)))))) (AND (P=P (X0) (X1))
(P=A (X1 ROOT) FLY) (P=L (X0 ROOT) (QUOTE TOBU))
(P=A (X1 SUBJ CASE) NOMINATIVE) (P=L (X0 SUBJ P
ROOT) (QUOTE GA)) (P=A (X1 SUBJ ROOT) BIRD) (P=L
(X0 SUBJ ROOT) (QUOTE TORI))) (GETVALUE* X
(QUOTE (X0))))))
```

Example Rule in the generation grammar:

```
(<start> ==> (<S>
  ((x1 <= (JP-BIRD-from-ENG-BIRD (x0))))))
```

The lisp function can be inserted anywhere in the grammar.

Use **translate** to run LR parser and Genkit when Transformation Kit is used.

```
(translate "sentence string")
```

```
> (translate "a bird flew")
```

```
>a bird flew
```

```
1 (1) ambiguity found and took 0.427894 se
```

```
;**** ambiguity 1 ***
```

```
((SUBJ  
  ((AGREEMENT 3SG) (CASE NOMINATIVE)  
                                (DEFINITENESS -)  
                                (NUMBER SG)  
                                (ROOT BIRD)))  
 (FORM PAST)  
 (ROOT FLY))
```

Generation F-structure:

```
((SUBJ ((ROOT TORI) (P ((ROOT GA)))  
  (AGREEMENT 3SG) (CASE NOMINATIVE)  
  (DEFINITENESS -) (NUMBER SG)))  
 (ROOT TOBU) (FORM PAST))
```

```

GenKit> <START> called
GenKit>   <S> called
GenKit>     <NP> called
GenKit>       <N> called
GenKit>         Rule 1 for <N> returns
                    "TORI"

GenKit>       <N> returns "TORI"
GenKit>     <P> called
GenKit>       Rule 1 for <P> returns "GA"
GenKit>     <P> returns "GA"
GenKit>     Rule 1 for <NP> returns
                    "TORI GA"

GenKit>     <NP> returns "TORI GA"
GenKit>   <VP> called
GenKit>     <V> called
GenKit>       Rule 2 for <V> returns
                    "TONDA"

GenKit>     <V> returns "TONDA"
GenKit>     Rule 1 for <VP> returns
                    "TONDA"

GenKit>     <VP> returns "TONDA"
GenKit>     Rule 1 for <S> returns
                    "TORI GA TONDA"

GenKit>     <S> returns "TORI GA TONDA"
GenKit>     Rule 1 for <START> returns
                    "TORI GA TONDA"

GenKit> <START> returns "TORI GA TONDA"

"TORI GA TONDA"

```