

Program Transformation

Application examples

- Converting to a new language dialect
- Migrating from a procedural language to an object-oriented one, e.g. C to C++
- Adding code comments
- Requirement upgrading, e.g. using 4 digits for years instead of 2 (Y2K)
- Structural improvements, e.g. changing GOTOs to control structures
- Pretty printing

Simple program transformation

- Modify all arithmetic expressions to reduce the number of parentheses using the formula:

$$(a + b) * c = a * c + b * c$$

x := (2+5) * 3

becomes

x := 2 * 3 + 5 * 3

Transformation tools

- There are many transformation tools
- Program-Transformation.org lists about 90 of them
- We will look at one of the most mature tools, TXL

TXL

- A generalized source-to-source translation system
- Uses a context-free grammar to describe the structures to be transformed
- Rule specification uses a by-example style
- Has been used to process billions of lines of code for Y2K purposes

TXL programs

- TXL programs consist of two parts:
 - Grammar for the input language
 - Transformation Rules
- Let's look at some examples...

Calculator.Txl - Grammar (Part 1)

```
define program
  [expression]
end define

define expression
  [term] | [expression] [addop] [term]
end define

define term
  [primary] | [term] [mulop] [primary]
end define
```

Calculator.Txl - Grammar (Part 2)

```
define primary
  [number] | ( [expression] )
end define

define addop
  '+' | '-'
end define

define mulop
  '*' | '/'
end define
```

Calculator.Txl - Transformation Rules (Part 1)

```
rule main
  replace [expression]
    E [expression]
  construct NewE [expression]
    E [resolveAddition]
      [resolveSubtraction]
      [resolveMultiplication]
      [resolveDivision]
      [resolveParentheses]
  where not
    NewE [= E]
  by NewE
end rule
```

Calculator.Txl - Transformation Rules (Part 2)

```
rule resolveAddition
  replace [expression]
    N1 [number] + N2 [number]
  by
    N1 [+ N2]
end rule

rule resolveParentheses
  replace [primary]
    ( N [number] )
  by N
end rule
```

DotProduct.Txl (Part 1)

```
define program
  ([repeat number]) . ([repeat number])
  | [number]
end define

rule main
  replace [program]
    ( V1 [repeat number] ) .
    ( V2 [repeat number] )
  construct Zero [number]
    0
  by
    Zero [addDotProduct V1 V2]
end rule
```

DotProduct.Txl (Part 2)

```
function addDotProduct V1 [repeat number]
  V2 [repeat number]

  deconstruct V1
    First1 [number] Rest1 [repeat number]
  deconstruct V2
    First2 [number] Rest2 [repeat number]
  construct ProductOfFirsts [number]
    First1 [* First2]
  replace [number]
    N [number] by
    N [+ ProductOfFirsts]
  [addDotProduct Rest1 Rest2]
end function
```

```
define program
  [repeat number]
end define

rule main
  replace [repeat number]
    N1 [number] N2 [number]
    Rest [repeat number]
  where
    N1 [> N2]
  by
    N2 N1 Rest
end rule
```

- Guided Tour
- Many examples
- Reference manual
- Download TXL for many platforms

Example uses

- HTML Pretty Printing of Source Code
- Language to Language Translation
- Design Recovery from Source
- Improvement of security problems
- Program instrumentation and measurement
- Logical formula simplification and interpretation