Normalization of AsmL programs

P. Cegielski – I. Guessarian

15-06-09

OUTLINE

- Introduction: ASM *versus* AsmL
- Definitions: syntax and semantics
- Normalization

ASM

ASM

- Algorithmically complete language
- Purely functional
- Purely parallel

AsmL

AsmL

- Implement ASM
- Closer to usual languages
- Easier to program with

ASM versus AsmL

ASM:

basic constructs (rules)

- Updates
- Tests
- Parallel composition

More Data Structures

AsmL:

basic constructs (rules)

ASM plus

- sequential composition
- UNTIL and WHILE loops

More Control Structures

ASM Syntax

ASM Syntax: rules are

- Updates: $f(t_1, \ldots, t_n) := t_0$
- Tests: IF φ THEN R_1 ELSE R_2 ENDIF
- Parallel composition:

par
R_1
$\vdots$
R_k
endpar

ASM Semantics

ASM Semantics: rules are executed in parallel until a fixed point is reached

Example: compute gcd(a,b)

IF $(a \mod b = 0)$ THEN $g := b$ ELSE

par

$a := b$

$b := a \mod b$

endpar

ENDIF

AsmL Syntax

AsmL Syntax: rules are

- ASM rules
- Sequential composition: step rule

  par
  step R_1
  $\vdots$
  step R_k
  endpar

- Iterations:

  step until φ R
  step while φ R
  step until fixpoint R

AsmL Semantics

AsmL Semantics:

As usual, The step rule is interpreted as the sequential composition of R_1, followed by R_2, etc. , ending with R_k.

par

step R_1

$\vdots$

step R_k

endpar

AsmL Semantics: example

```
step
        avg := 0
        i := 0
step    until (i = n)
        avg := avg + grade[i]
        i := i+1
step
        avg := avg/n
        i := 0
        nb := 0
step    until (i = n)
        if (grade[i] > avg) then nb := nb+1
        i := i+1
```

Compute the number of grades whose value is greater than the average grade.

Normalization

Goal

Translate the control structures of AsmL into ASM

- Simulate sequential composition.
- Simulate nested iterations with single fixed point iteration.

Normalization: example

```
if (mode = 0) then   avg := 0
                     i := 0
                     mode := 1
if (mode = 1) then   avg := avg + grade[i]
                     i := i+1
                     if (i = n) then mode := 2
if (mode = 2) then   avg := avg/n
                     i := 0
                     nb := 0
                     mode := 3
if (mode = 3) then   if (grade[i] > avg) then nb := nb+1
                     i := i+1
                     if (i = n) then mode := 4
```

Normalization

In the normal form

- *sequential composition* is simulated by controlling the parallelism via a new constant symbol *mode*.
- iteration *until fixpoint* will be simulated by a new control constant *c* which is set to 1 if non-trivial updates are performed (0 otherwise).

A Normal form AsmL program is a program of the form

step until fixpoint
par
if (mode = 0) then R_1
...
if (mode = j) then R_j
...
if (c=0) $\wedge$ (mode = i) then R_i
if (c=1) $\wedge$ ($M_i \geq$ mode $\geq$ i) then
mode:=i
c:=0
...
if (mode = k) then R_k
endpar

Construction of normal forms

By structural induction

AsmL rule	Normal form
ASM rule R R	if (mode $= 0$) then R mode $:= 1$
AsmL step rule step R_1 step R_2	par R_1'' $R_2^{+fin_1}$ endpar

Construction of normal forms

Iterations

AsmL rule	Normal form
UNTIL STEP UNTIL φ R	if (mode $= 0$) then if $\neg\varphi$ then mode $:= 1$ if φ then mode $:=$ M+1 R^{+1}

UNTIL FIXPOINT
 	
step until fixpoint R
	if (c=1) $\wedge$ ($M \geq$ mode ≥ 0) then
	mode:=0
	c:=0
	if (c=0) $\wedge$ (mode $= 0$) then R^c

Theorem

For every AsmL program P, there exists a normal form ASM program P_n such that for every $\mathcal{L}$-state $\mathcal{A}$ there exists an $\mathcal{L}_m$-state $\mathcal{A}_m$ in which P_n computes the same function as P.

Complexity counted by the number of updates: the complexity of P is of the form $\prod n_i$ and the complexity of P_n is of the form $\prod(1 + n_i) > C \times \prod n_i$

Summary

- for each AsmL program P, there is an ASM program P_n in normal form computing the same function
- if the nesting of iterations is bounded by constant k in P, then the complexity of P_n is at most $2k$ times the complexity of P
- can we do better?

THANKS FOR YOUR ATTENTION