Hargi (191524027), D4-2A

# Imperative Programming Research

# Executing and verifying higher-order functional-imperative programs in Maude

VladRusua,*, AndreiArusoaieb

a*Inria Lille Nord Europe, France*

b*"Alexandru Ioan Cuza" University of Iasi, Romania*

# Introduction

- Kontribusi Dalam makalah ini kami menggunakan sistem modul dan meta-level untuk menggabungkan fungsi tingkat tinggi dan monad negara di Maude. Maude dari bahasa pemrograman fungsional tingkat tinggi dengan fitur-fitur penting. Dalam bahasa yang dihasilkan, seseorang dapat memprogram dalam semangat, misalnya, Haskell , sambil tetap memiliki akses ke fitur bahasa dan sistem Maude yang sudah dikenal . Kami mengilustrasikan bahasa yang dihasilkan dengan program fungsional-imperatif sederhana.
- Kelayakan eksekusi simbolik, verifikasi program dan verifikasi program-ekuivalen secara resmi dibuktikan. Monad negara bagian juga telah diperkenalkan di asisten bukti Coq untuk membuktikan program yang ditulis dengan gaya imperatif . Integrasi monad negara bagian di Maude adalah pengetahuan terbaik kami yang baru dalam kerangka ini. Pekerjaan terkait lainnya adalah verifikasi program, khususnya, sehubungan dengan Reachability-Logic rumus.

- rl adalah formalisme untuk mendefinisikan semantik bahasa pemrograman dan logika spesifikasi program yang dapat dilihat sebagai generalisasi bahasa-parametrik dari logika Hoare. Salah satu kontribusi dari makalah ini menunjukkan bahwa pendekatan sebelumnya juga dapat digunakan pada bahasa yang didefinisikan oleh embedding dangkal, sedangkan karya yang ada digunakan embeddings dalam. Alat termasuk eksekusi simbolik yang dikombinasikan dengan bahan lain termasuk Java PathFinder , DART , IMUT , EXE , PEX . Alat yang dibangun di atas eksekusi simbolik untuk melakukan analisis dan verifikasi program termasuk .
- Perbedaan pertama adalah di sini kita menerapkan simbolik eksekusi, verifikasi program dan program ekuivalen dengan bahasa yang tertanam dangkal di Maude, dengan fitur-fitur fungsional-imperatif tingkat tinggi, sedangkan di versi sebelumnya kami hanya menerapkan verifikasi formal, ke program dalam bahasa imperatif sederhana yang tertanam dalam di Maude. Perbedaan lainnya adalah bahwa di versi awal kami bersikeras inkrementitas dalam buktinya, sedangkan di sini fitur ini tidak dominan

# Background

- Kami menggabungkan fungsi tingkat tinggi dan monad status di Maude, sehingga menyematkan bahasa fungsional tingkattinggi dengan fitur-fitur penting dalam kerangka kerja Maude. Kami mengilustrasikan, melalui program sederhana dalambahasa yang dihasilkan: eksekusi program yang konkret dan simbolis; verifikasinya berkenaan dengan properti yangdiekspresikan dalam Reachability Logic, generalisasi bahasa-parametrik dari Hoare Logic; dan verifikasi properti kesetaraanprogram. Pendekatan kami terbukti bagus dan diimplementasikan di Full Maude dengan memanfaatkan fitur reflektif dansistem modulnya.

# Contents

- Program-program imperatif fungsional tingkat tinggi di Maude
- Verifikasi program
- Kesetaraan program
- Kesimpulan

# Program-program imperatif fungsional tingkat tinggi di Maude

# 1 Menambahkan fungsi tingkat tinggi ke Maude

- Teori TRIV

  fthTRIVis

  sortElt.***thisisacomment

  endfth

- Module arrow

  fmodARROW{X::TRIV,Y::TRIV}is***formalparameters

  protectingSUBST.***importedmodule

  sortArrow{X,Y}.***sortdeclaration

  op__:Arrow{X,Y}X$Elt->Y$Elt.***operationdeclarations

  oplambda_:_:X$EltY$Elt->Arrow{X,Y}[ctor].

  varf:Arrow{X,Y}.***variabledeclarations

  varsxz:X$Elt.

  vary:Y$Elt.

  opERR:->[Y$Elt].***constantdeclaration

  eq(lambdax:y)(z)=***equation

  downTerm(subst(upTerm(x),upTerm(z),upTerm(y)),ERR).

  endfm

- Dengan menghubungkan parameter aktual ke parameter formal dari unit berparameter. Terdapat sejumlah unit yang telah ditentukan sebelumnya untuk struktur data umum (Boolean,integer,string,…), adalah bersyarat

# 2 State Monad

state monads dengan state sort S dan outputnya berupa sort A adalah fungsi dari s untuk (A,S), sehingga definisi dari state monads sendiri adalah mengambil state dan mengembalikan state berikutnya yang berupa hasil komputasi yang bergantung pada state, state monad memberikan alternatif untuk melakukan komputasi

Contoh program :

```
(fmodRET{A::TRIV,S::TRIV}is
protectingARROW{A,Monad{A,S}}.
vara:A$Elt.
vars:S$Elt.
opret:->Arrow{A,Monad{A,S}}.
eqret(a)(s)=(a,s).
endfm)
```

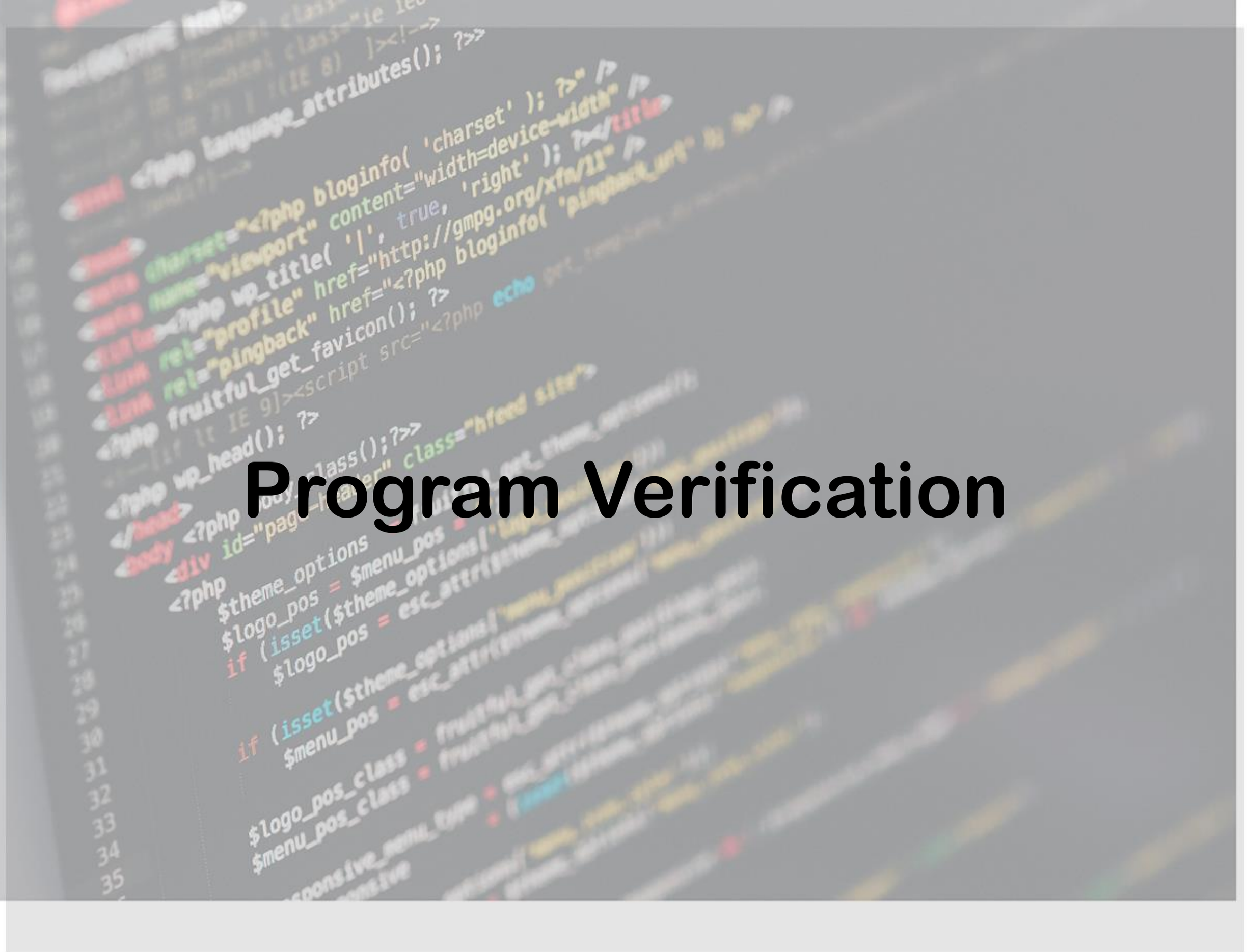
State monad : opret:->Arrow{A,Monad{A,S}}

# 3 Sorting stack

- Contoh program :

```
opsort:Arrow{X,Arrow{X,Bool}}->Monad{Unit,State{X}}.
eqsort(leq)st(t,emptyStack)=(tt,st(t,emptyStack)).
eqsort(leq)st(t,n)=(tt,st(t,n)).
eqsort(leq)st(t,n::m::stk)=
(min(leq);;
(dox:=popin(sort(leq);;push(x)))
)
st(t+1,n::m::stk)
ifx:=freshVar(t).
endfm)
```

Operasi menyortir Operasi monadik ini adalah melakukan penyortiran tumpukan menurut pesanan leq itu penerima sebagai parameter. Kasus yang menarik adalah tumpukan dengan setidaknya dua elemen, yang ditangani oleh persamaan ketiga. Elemen ini kemudian muncul dan disimpan dalam sebuah variabel x



# Program Verification

## Contoh :

crl min (leq) st (t, true, true, (q :: stk)) => (tt, st (t ', true, (leq q 'least (leq, q' :: stk ')) dan sameElements (q :: stk, q' :: stk '), q' :: stk '))) jika t ': = freshNatVar (t) / \ q': = freshVar (t) / \ stk ': = freshStackVar (t).

Rumus diatas merupakan rumus dari min fungsi yang mana fresh variable dalam kondisi dikuantifikasi secara eksistensial (Nat, X $ Elt dan Stack {X}. Fungsi least dan sameElements return, respectively, elemen paling sedikit w.r.t. urutan leq di stack dan nilai asli dari predikat menyatakn bahwa kedua stack memiliki element yang sama.

# Program equivalence

Dengan menggunakan eksekusi simbolik dapat memverifikasi program sehubungan dengan rumus torl. Penulis menunjukkan bahwa eksekusi simbolik dan verifikasi program memungkinkan penulis untuk membuktikan kesetaraan antara program. Dua program yang lemah lebih memiliki ketertarikan sendiri. Dua program dikatakan sama jika dimasukkan input yang sama dan diberhentikan di waktu yang sama maka akan memiliki output yang sama

# Conclusion

- Dalam makalah ini telah menunjukkan bahwa Maude adalah kerangka kerja all-in-one di mana seseorang dapat menulis, menjalankan, dan memverifikasi program-program imperatif fungsional tingkat tinggi. Ada beberapa perkembangan yang ditinggalkan Prosedur itu sendiri membutuhkan implementasi meta-level.

# Credit

- Journal of Logical and Algebraic Methods in Programming
- Executing and verifying higher-order functional-imperative programs in Maude
- VladRusua,∗, AndreiArusoaieb
- a*Inria Lille Nord Europe, France*
- b*“Alexandru Ioan Cuza” University of Iasi, Romania*

- Pembuat PPT :
- Muhammad Hargi Muttaaqin
- 191524027
- D4 TIF
- 2A-D4
- Muhammad.hargi.tif419@polban.ac.id