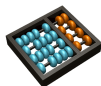


# Criação de uma Biblioteca Padrão para a Linguagem HasCASL

Glauber Módolo Cabral – Orientando  
Prof. Dr. Arnaldo Vieira Moura – Orientador

Universidade Estadual de Campinas  
Instituto de Computação



26 de Abril de 2010

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos

# Roteiro

## Introdução

### Métodos Formais e Linguagens de Especificação

## Contextualização

### Linguagens Utilizadas

### Ferramentas Utilizadas

## Motivação e Justificativa

## Objetivos

## Desenvolvimento

## Contribuições

## Questões em aberto

## Conclusões

## Trabalhos futuros

## Agradecimentos

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

## Roteiro

### Introdução

Métodos Formais e  
Linguagens de Especificação

### Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

### Motivação e Justificativa

### Objetivos

### Desenvolvimento

### Contribuições

### Questões em aberto

### Conclusões

### Trabalhos futuros

### Agradecimentos

## Métodos Formais

- ▶ Ferramentas de Engenharia de Software que empregam formalismos matemáticos na construção de programas;
- ▶ Compostos por uma ou mais linguagens de especificação e algumas ferramentas auxiliares;

## Linguagens de Especificação Formal

- ▶ Linguagens existentes são baseadas em diversos formalismos: Extended ML, Z, MÉTODO B, MAUDE, LARCH, CASL, ...);
- ▶ Apresentam variado nível de suporte à verificação automática de propriedades auxiliada por ferramentas.
- ▶ Possuem códigos de exemplo e bibliotecas para reuso.

# CASL: Common Algebraic Specification Language

- ▶ Linguagem de especificação algébrica que permite extensões e sub-linguagens;
- ▶ Possui uma biblioteca padrão com especificações para reuso.

## HASKELL

- ▶ Linguagem de programação funcional;
- ▶ Implementa conceitos de lógica de segunda ordem: tipos que são funções, polimorfismo e construtores de tipos;
- ▶ Avaliação preguiçosa: argumentos de função são avaliados apenas quando são usados;
- ▶ HASKELL PRELUDE: biblioteca padrão com tipos básicos e funções de manipulação de listas, de textos e de E/S em tela e arquivo.

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos

## HASCASL: HASKELL + CASL

- ▶ Extensão de CASL com conceitos de lógica de segunda ordem;
- ▶ Tem a linguagem de programação funcional HASKELL como sub-conjunto;
  - ▶ Facilita a transformação da especificação para código HASKELL executável;
- ▶ Possui avaliação estrita: função com parâmetros indefinidos possui valor de retorno indefinido;
- ▶ No entanto, avaliação preguiçosa pode ser emulada por uma combinação de tipos de dados;
- ▶ **Não possui biblioteca padrão com especificações reutilizáveis.**

# HETS: Heterogeneous Tool Set

- ▶ Analisador sintático para CASL e sub-linguagens;
- ▶ Utiliza o editor *Emacs* como interface.
- ▶ Gera um *Grafo de Desenvolvimento*:
  - ▶ Nós: especificações;
  - ▶ Arcos: dependência entre especificações;
  - ▶ Cores indicam o estado das necessidades de prova;
- ▶ Gerencia as provas realizadas com o provador de teoremas ISABELLE.

## Isabelle

- ▶ Provador de teoremas semi-automático;
- ▶ Lógicas para escrita de provas: HOL e HOLCF, dentre outras;
- ▶ HETS traduz especificações em HASCASL para HOL e HOLCF.

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

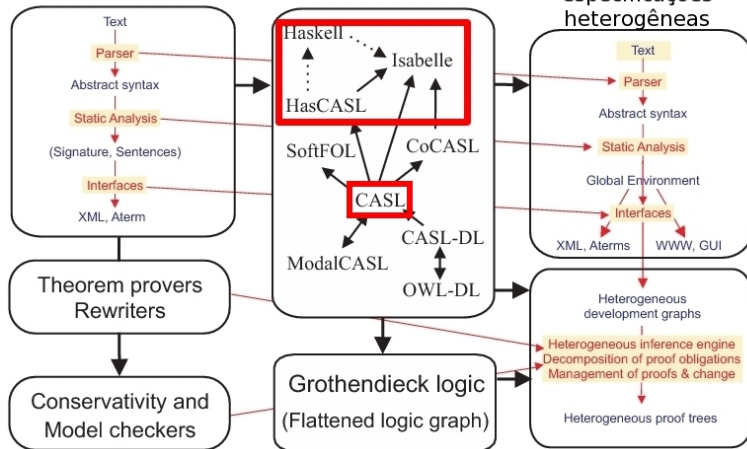
Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos

## Ferramentas para lógicas Grafo de Lógicas



Fonte:

[http://www.informatik.uni-bremen.de/agbkb/forschung/formal\\_methods/](http://www.informatik.uni-bremen.de/agbkb/forschung/formal_methods/)

[CoFI/hets/index\\_e.htm](http://CoFI/hets/index_e.htm)

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos

# Motivação e Justificativa

## Criação de uma Biblioteca Padrão para HASCASL

- ▶ Contribuiria para difundir a linguagem;
- ▶ Permitiria o reuso de especificações;
- ▶ Considerada uma premissa para uso da linguagem em problemas reais.

## Biblioteca PRELUDE como Ponto de Partida

- ▶ Possui tipos de dados amplamente utilizados em programas Haskell;
- ▶ Permitiria o uso de tipos existentes em HASKELL nas especificações escritas em HASCASL;
- ▶ Facilitaria a transformação de especificações em HASCASL para código executável em HASKELL;



## Objetivo Principal

Especificar uma biblioteca para a linguagem HASCASL com tipos de dados de segunda ordem segundo a biblioteca PRELUDE da linguagem HASKELL.

## Objetivo Secundário

Provar teoremas criados pela ferramenta Hets durante a análise da especificação utilizando o provador de teoremas ISABELLE.

# Escolhas iniciais

## Avaliação Estrita

- ▶ Emprega construções mais simples de HASCASL;
- ▶ Precisa de conhecimento básico de HOL para escrever as provas;
- ▶ Possui alguma documentação e alguns exemplos;
- ▶ Escolhida como ponto de partida para a especificação.

## Avaliação Preguiçosa

- ▶ Emprega construções mais avançadas de HASCASL;
- ▶ Precisa de profundo conhecimento prévio das linguagens HOL e HOLCF para escrever as provas;
- ▶ Possui pouca documentação e exemplos;
- ▶ Introduzida em refinamento posterior da especificação.

## Especificação da Biblioteca em HASCASL

- ▶ Modela tipos e funções através de axiomas em HASCASL e CASL;
- ▶ Possui propriedades a serem verificadas com ISABELLE.
- ▶ Constituída por 18 especificações.

## Verificação de Teoremas com ISABELLE

- ▶ Garante que as especificações se comportem como esperado;
- ▶ Necessita que axiomas sejam reescritos para que ISABELLE consiga usá-los;
- ▶ Axiomas são reescritos na forma de lemas e também precisam ser provados para garantir consistência;

# Estado Inicial da Verificação das Especificações

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas

Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

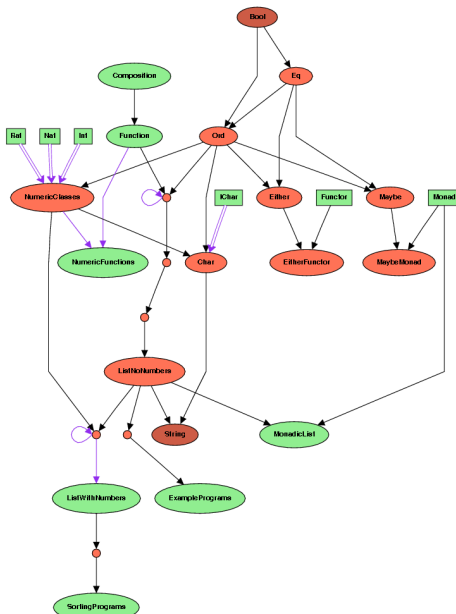
Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos



# Especificação Ord em HASCASL

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos

```
1 spec Ord = Eq and Bool then
2 free type Ordering ::= LT | EQ | GT
3 type instance Ordering: Eq
4 . (LT == LT) = True    %(IOE01)% %implied
5 . (LT == EQ) = False  %(IOE04)%
6 . (LT /= EQ) = True   %(IOE07)% %implied
7 class Ord < Eq {
8   fun __<__ : a * a -> Bool
9   var x, y, z, w: a
10  . (x == y) = True => (x < y) = False
11                                     %(LeIrreflexivity)%
12  . (x < y) = True => y < x = False
13                                     %(LeTAsymmetry)% %implied
14  . (x < y) = True /\ (y < z) = True
15    => (x < z) = True                %(LeTTransitive)%
16  . (x < y) = True \/ (y < x) = True
17    \/ (x == y) = True              %(LeTTTotal)%
18 }
```

# Especificação Ord traduzida para HOL

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

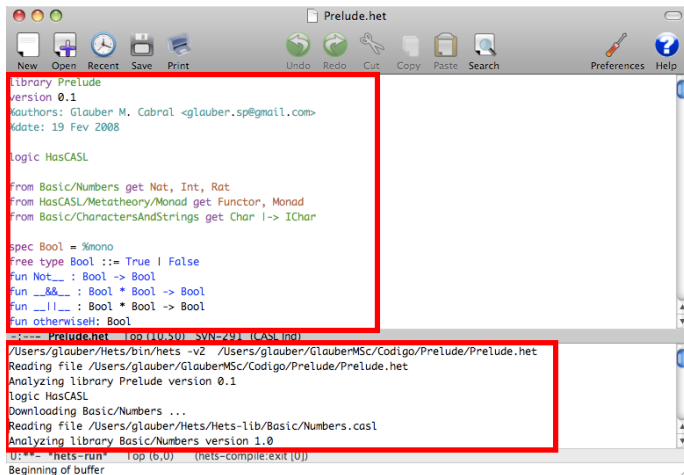
Trabalhos futuros

Agradecimentos

```
1 LeIrreflexivity [rule_format] :
2 "ALL (x :: 'a). ALL (y :: 'a).
3  x == ' y = True' --> x < ' y = False'"
4
5 lemma LeIrreflContra : " x < ' x = True' ==> False"
6 by auto
7
8 theorem LeTAsymmetry :
9 "ALL (x :: 'a). ALL (y :: 'a).
10  x < ' y = True' --> y < ' x = False'"
11 apply(auto)
12 apply(rule ccontr)
13 apply(simp add: notNot2 NotTrue1)
14 apply(rule_tac x="x" in LeIrreflContra)
15 apply(rule_tac y = "y" in LeTTransitive)
16 by auto
```

# Passo a passo da Especificação e Verificação

1. Realizar verificação sintática com a ferramenta HETS no arquivo da especificação (extensão *.casl* ou *.het*);



```
library Prelude
version 0.1
%authors: Glauber M. Cabral <glauber.sp@gmail.com>
%date: 19 Feb 2008

Logic HasCASL

From Basic/Numbers get Nat, Int, Rat
From HasCASL/Metatheory/Monad get Functor, Monad
From Basic/CharactersAndStrings get Char l-> IChar

spec Bool = %mono
free type Bool ::= True | False
fun Not__ : Bool -> Bool
fun __&&__ : Bool * Bool -> Bool
fun __||__ : Bool * Bool -> Bool
fun otherwiseH: Bool

U:***- Prelude.het Top (10.50) SVN-291 (CASL ind)
/Users/glauber/Hets/bin/hets -v2 /Users/glauber/GlauberMSc/Codigo/Prelude/Prelude.het
Reading file /Users/glauber/GlauberMSc/Codigo/Prelude/Prelude.het
Analyzing library Prelude version 0.1
Logic HasCASL
Downloading Basic/Numbers ...
Reading file /Users/glauber/Hets/Hets-lib/Basic/Numbers.casl
Analyzing library Basic/Numbers version 1.0
U:***- "hets-run" Top (b,U) (hets-compile:exit [U])
Beginning of buffer
```

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas

Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

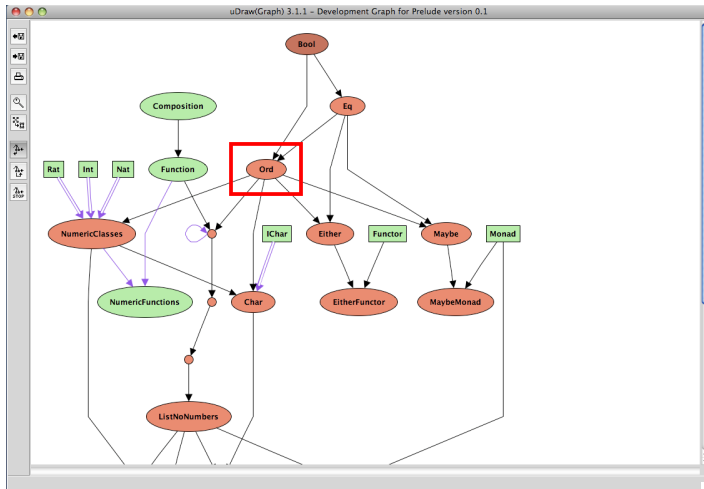
Trabalhos futuros

Agradecimentos



# Criação de uma Biblioteca Padrão para a Linguagem HasCASL

2. HETS gera o grafo de desenvolvimento da especificação analisada;



## Introdução

## Contextualização

## Motivação e Justificativa

## Objetivos

## Desenvolvimento

## Contribuições

Questões em aberto

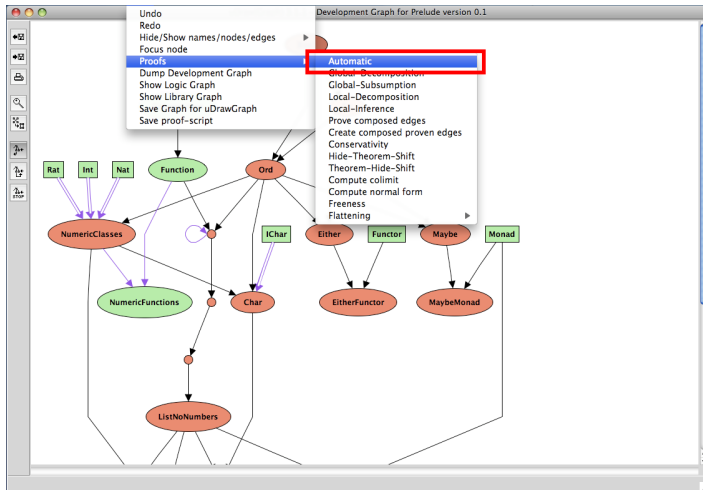
## Conclusões

## Trabalhos futuros

## Agradecimentos

# Passo a passo da Especificação e Verificação

3. Executar o comando de prova automático para que ele interprete o grafo e as necessidades de prova;



Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas

Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

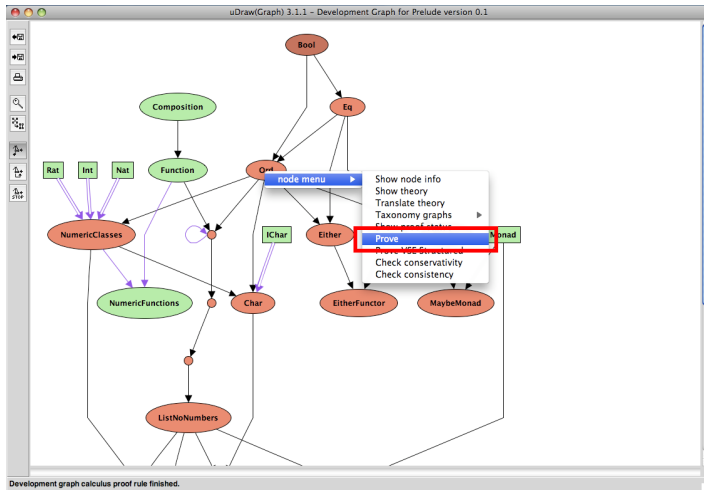
Conclusões

Trabalhos futuros

Agradecimentos

# Passo a passo da Especificação e Verificação

4. Selecionar um nó vermelho (com provas em aberto) para verificar seus teoremas;



Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

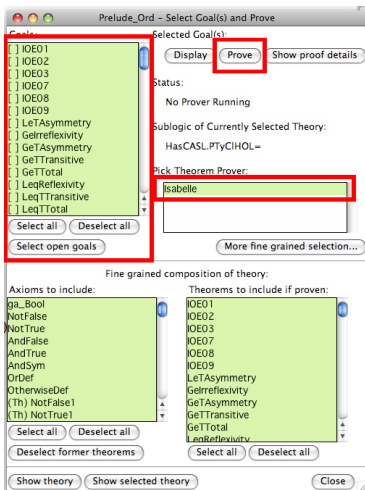
Conclusões

Trabalhos futuros

Agradecimentos

# Passo a passo da Especificação e Verificação

5. Escolher os teoremas a serem provados, o provador de teorema e iniciar a prova;



Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

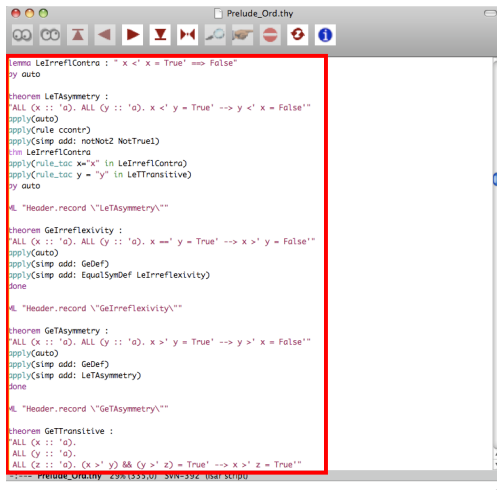
Conclusões

Trabalhos futuros

Agradecimentos

# Passo a passo da Especificação e Verificação

## 6. Escrever as provas utilizando a interface *ProofGeneral* para o provador ISABELLE;



```

Prelude_Ord.thy

lemma LeIrreflContra : "x <= y = True ==> False"
  by auto

theorem LeTSymmetry :
  "ALL (x :: 'a). ALL (y :: 'a). x <= y = True ==> y <= x = False"
  apply(auto)
  apply(rule ccontr)
  apply(simp add: notNot2 NotTrue1)
  then LeIrreflContra
  apply(rule_tac x="x" in LeIrreflContra)
  apply(rule_tac y = "y" in LeTTransitive)
  by auto

ML "Header.record \"LeTSymmetry\""

theorem GeIrreflexivity :
  "ALL (x :: 'a). ALL (y :: 'a). x == y = True ==> x > y = False"
  apply(auto)
  apply(simp add: GeDef)
  apply(simp add: EqualSymDef LeIrreflexivity)
  done

ML "Header.record \"GeIrreflexivity\""

theorem GeTSymmetry :
  "ALL (x :: 'a). ALL (y :: 'a). x > y = True ==> y > x = False"
  apply(auto)
  apply(simp add: GeDef)
  apply(simp add: LeTSymmetry)
  done

ML "Header.record \"GeTSymmetry\""

theorem GeTTransitive :
  "ALL (x :: 'a).
  ALL (y :: 'a).
  ALL (z :: 'a). (x > y) && (y > z) = True ==> x > z = True"

```

Criação de uma Biblioteca Padrão para a Linguagem HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e Linguagens de Especificação

Contextualização

Linguagens Utilizadas

Ferramentas Utilizadas

Motivação e Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em aberto

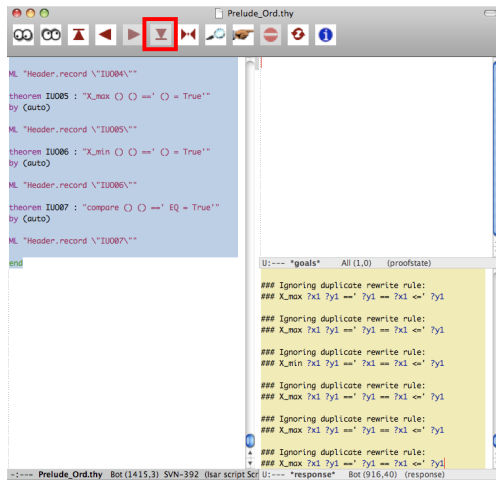
Conclusões

Trabalhos futuros

Agradecimentos

# Passo a passo da Especificação e Verificação

## 7. Executar a verificação completa do arquivo de prova e fechar a janela;



Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

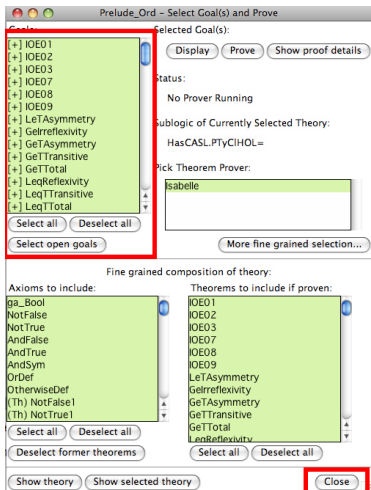
Conclusões

Trabalhos futuros

Agradecimentos

# Passo a passo da Especificação e Verificação

8. Os teoremas que foram provados tem as respectivas caixas de seleção marcadas;



Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

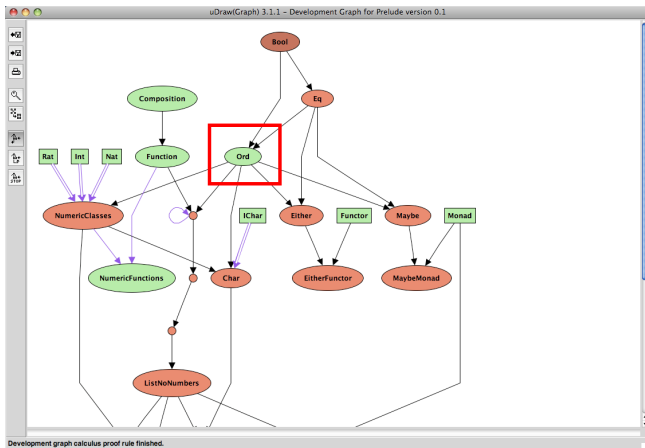
Conclusões

Trabalhos futuros

Agradecimentos

# Passo a passo da Especificação e Verificação

9. Resultado da verificação no grafo: verde (totalmente verificado), vermelho (teoremas em aberto), amarelo (falta verificação de consistência).



Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos





# Contribuições

- ▶ Biblioteca especificada possui os tipos de dados booleano, listas, caracteres e cadeias de caracteres.
- ▶ Especificações de exemplos empregam listas e booleanos;
- ▶ Duas versões para a biblioteca (aprox. 1000 LOC cada):
  - 1ª Versão Tipos com avaliação estrita devido à complexidade do uso de tipos com avaliação preguiçosa;
  - 2ª Versão Refinamento para suportar tipos com avaliação preguiçosa sem suporte a tipos infinitos;
- ▶ A maioria das necessidades de prova foram verificadas:
  - ▶ 9 especificações verificadas totalmente;
  - ▶ 8 especificações possuem alguns teoremas em aberto.

## Especificações de Tipos Numéricos

- ▶ Especificações de tipos numéricos da biblioteca da linguagem CASL não possuem todos os lemas necessários para que sejam utilizadas com o provador ISABELLE.
- ▶ Seu uso exigiria mapeamentos entre os tipos de dados da biblioteca da linguagem CASL e seus respectivos tipos de dados na linguagem HOL.
- ▶ O mapeamento para o tipo de dados `Nat` hoje existente está implementado com funções obsoletas e não pode ser usado dentro do provador ISABELLE.
- ▶ Funções envolvendo tipos numéricos não foram especificadas na versão atual da biblioteca.

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos

## Tipos Contínuos e Estruturas Infinitas

- ▶ Exige tipos de dados complexos da linguagem HASCASL e o uso da lógica HOLCF no provador de teoremas ISABELLE.
- ▶ HOLCF é mais complexa do que HOL e seu uso estava fora do escopo do trabalho.
- ▶ Suporte a estes tipos não foi implementado;
- ▶ Por consequência, ainda não é possível refinar as especificações para o subconjunto executável da linguagem HASCASL.

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

**Questões em  
aberto**

Conclusões

Trabalhos futuros

Agradecimentos

# Conclusões

## Objetivos Iniciais

- ▶ Especificar uma biblioteca para HASCASL baseada na biblioteca PRELUDE;
- ▶ Verificar propriedades das especificações criadas.

## Objetivos Alcançados

- ▶ Biblioteca possui os tipos de dados booleano, listas, caracteres e cadeias de caracteres
- ▶ Tipos numéricos, Tipos Contínuos e Estruturas Infinitas não foram especificados;
- ▶ Estado das necessidades de prova geradas:
  - ▶ 9 totalmente verificadas;
  - ▶ 8 verificadas parcialmente.

- ▶ Escrever novos mapeamentos entre os tipos de dados da biblioteca da linguagem CASL e os tipos de dados da linguagem HOL
- ▶ Especificar e verificar tipos de dados numéricos e funções que os envolvam.
- ▶ Especificar tipos de dados infinitos;
- ▶ Especificar estruturas de dados mais complexas implementadas por alguns compiladores da linguagem HASKELL.

# Agradecimentos

Obrigado!

Apoio Financeiro



Contato

glauber.sp@gmail.com

Criação de uma  
Biblioteca Padrão  
para a Linguagem  
HasCASL

Glauber M. Cabral

Roteiro

Introdução

Métodos Formais e  
Linguagens de Especificação

Contextualização

Linguagens Utilizadas  
Ferramentas Utilizadas

Motivação e  
Justificativa

Objetivos

Desenvolvimento

Contribuições

Questões em  
aberto

Conclusões

Trabalhos futuros

Agradecimentos