

# Relatório Final - Lógica Computacional 1<sup>1</sup>

Marcelo Araújo - (150016794), Augusto Brandão - (160024366), Ian Nery - (150129670)

<sup>1</sup>Departamento de Ciência da Computação – Universidade de Brasília (UNB)

## Introdução

A lógica computacional, extenso campo de pesquisa e aplicações práticas, é primordialmente utilizada para a comprovação do funcionamento de sistemas críticos modernos, que a torna essencial tanto como especialização na área, quanto como introdução para outros âmbitos de pesquisa, como a de inteligência artificial. O objetivo do projeto em questão é principalmente introduzir os alunos a ferramentas de verificação e formalização para a melhor absorção e entendimento das técnicas de manipulação lógica apresentadas em sala, além de demonstrar como tais conceitos aprendidos são aplicados na prática.

No projeto contido nesse relatório, é apresentado para a prova questões sobre propriedades envolvendo a inserção e ordenação de sequências. A ferramenta utilizada para a especificação e verificação formal da lógica por detrás de tais conjecturas é o PVS. O software mencionado é de grande importância para as aplicações práticas descritas acima e outras, como é possível constatar pelo diretório da NASA, que abarca diversas bibliotecas acerca de diversas conjecturas similares a apresentada no trabalho, com a finalidade de auxiliar os pesquisadores da área a ter um entendimento similar ao proposto pelo projeto.

## Explicação das soluções

### Questão 1

A conjectura da questão 1, *fs\_insert\_in\_sorted\_preserves\_sort* propõe que para toda  $s$  sequência finita e  $x$  natural, têm-se que a ordenação de  $s$  implica na ordenação de  $s$  após uma inserção de  $x$  em si. Para o início de sua prova, foi utilizado um passo de indução forte para a sequência " $s$ " e para a variável " $\text{length}(s)$ ", sendo essa a variável que define o tamanho da sequência finita " $s$ ". A estratégia primordial para a prova de tal conjectura foi a utilização da skolemização e instanciação, relatadas a regras referentes aos quantificadores universais da lógica de Gentzen; manipuladores equacionais com a finalidade de dissecar o problema para torná-lo trivial, como enunciado pelos comandos **lift-if**, **replace** e **expand**; manipuladores lógico-proposicionais, como **prop**; e também comandos com a função de cut, como **case** e **lemma**.

### Questão 2

A questão 2, *fs\_insertionsort\_is\_sorted*, é uma conjectura cuja proposição é: para toda sequência finita natural " $s$ " têm-se que a inserção ordenada em " $s$ " torna-se em uma sequência ordenada. Para a prova da conjectura, foi utilizado como na questão 1 a estratégia de indução forte, para as mesmas " $s$ " e " $\text{length}(s)$ ", e além dos comandos de skolemização e instanciação, foram utilizados novamente os manipuladores equacionais **lift-if**, **replace** e **expand**; o manipulador lógico-proposicional **prop**; e por fim os comandos que exprimem a função cut, nesse caso **case**, **rewrite** e **lemma**.

### Questão 3

A questão 3, *fs\_ins\_and\_add\_in\_perm\_is\_perm*, consiste na proposição que para quaisquer as sequências "s1" e "s2", que são permutações entre si, e para qualquer variável "x", têm que a inserção de x no primeiro elemento da sequência s1 e a inserção de x dada pelo método *insertion* em s2, s1 e s2 ainda são permutações (ACHO Q O CONCEITO TA ERRADO ME CORRIJAM AI CLÃ). O método para prova da conjectura foi o uso dos mesmos mecanismos da questão 2, além do comando **typepred**, cuja função é de apresentar restrições de tipo em expressões previamente estabelecidas.

### Questão 4

Já a questão 4, *fs\_insertionsort\_is\_permutations*, representa que para cada sequência "s", a inserção ordenada em s e a sequência original s sofrem permutação. Na prova feita, foi utilizado o passo de indução forte como na questão 1, os métodos descritos nas questões 2 e 3, e também o comando de manipulação equacional **decompose-equality**, cuja função é decompor igualdades em componentes mais triviais para prova, como a decomposição de uma igualdade de sequência a separa em uma igualdade de sequências junto a uma igualdade de seus tamanhos.

### Especificação do problema e explicação do método de solução

Nesta etapa mostraremos o desenvolvimento das provas e suas relações com a dedução natural no cálculo de Gentzen. Omitiremos certos passos com a finalidade de manter a prova legível, como certos comandos de *expand*, passos omitidos serão sinalizados adequadamente quando necessários, além disso, fórmulas escondidas com a mesma finalidade estarão representadas como  $\Gamma$  do lado esquerdo do sequente e como  $\Delta$  do lado direito do sequente.

### Questão 1

Usaremos algumas abreviação com o intuito de diminuir os tamanhos das provas assim, *s?* abrevia *is.sorted?(x : finseq)*, *i* abrevia *insertion(x : nat, y : finseq)*, *l* abrevia *length(x : finseq)*, *s* abrevia *seq(.)*, *af* abrevia *addfirst(x : nat, y : finseq)*.

$$\frac{\frac{\frac{\frac{\nabla_1}{\Gamma, s?(x_1) \Rightarrow \Delta_1} (prop)}{\Gamma, s?(x_1) \Rightarrow s?(i(x, x_1))} (R_{=})}{\Gamma \Rightarrow s?(x_1) \rightarrow s?(i(x, x_1))} (R_{\rightarrow})}{\frac{\frac{\forall_y \forall_x (l(y) < l(x_1)) \rightarrow s?(y) \rightarrow s?(i(x, y)) \Rightarrow \forall_x s?(x_1) \rightarrow s?(i(x, x_1))}{\Rightarrow \forall_{s,x} s?(s) \rightarrow s?(i(x, s))} (R_{\forall}) \text{ e } (LW)} (mi+)$$

Onde  $\Delta_1$  foi usado para abreviar a igualdade usada por  $(R_{=})$ , aplicada pelo comando *expand*, as regras de  $(R_{\forall})$  e  $(R_{\rightarrow})$  foram aplicadas com o comando *sheep*.

$$\begin{aligned} \Delta_1 = & IF \ l(x_1) = 0 \ THEN s?(af(x, x_1)) \\ & ELSE IF \ x \leq f(x_1) \ THEN s?(af(x, x_1)) \\ & ELSE s?(af(f(x_1), i(x, rest(x_1)))) \end{aligned} \quad (1)$$

Agora podemos explicar o uso do comando *prop* em nossa prova para obtermos os ramos  $\nabla_{1,2,3}$ , a estrutura de *if, then else* pode ser interpretada como uma série de implicações, como no exemplo de  $\Delta_1$ , temos,  $l(x_1) = 0 \rightarrow s?(af(x, x_1))$ , como uma primeira implicação,  $(\neg(l(x_1) = 0) \wedge x \leq f(x_1)) \rightarrow s?(af(x, x_1))$ , e temos uma terceira implicação que corresponde ao caso que nenhuma das duas condições dos *if's* são verdadeiras,  $(\neg((l(x_1) = 0) \wedge x \leq f(x_1))) \rightarrow s?(af(x, x_1))$ .

Utilizando o comando *prop* nessa estrutura obtemos 3 ramos, prove nientes da utilização de comando como *flatten*, *split*, assim expandindo o *prop* utilizado acima podemos ver as seguintes provas:

$$\frac{\frac{\nabla'_1}{\Gamma, l(x_1) = 0, s?(x_1) \Rightarrow s?(af(x, x_1))} \quad \nabla_2 \quad \nabla_3}{\Gamma, s?(x_1) \Rightarrow \Delta_1} (prop)$$

$$\frac{\nabla_1 \quad \frac{\nabla'_2}{\Gamma, x \leq f(x_1), s?(x_1) \Rightarrow s?(af(x, x_1)), l(x_1) = 0} \quad \nabla_3}{\Gamma, s?(x_1) \Rightarrow \Delta_1} (prop)$$

$$\frac{\nabla_1 \quad \nabla_2 \quad \frac{\nabla'_3}{\Gamma, s?(x_1) \Rightarrow s?(af(x, x_1)), l(x_1) = 0, x \leq f(x_1)}}{\Gamma, s?(x_1) \Rightarrow \Delta_1} (prop)$$

Assim, isso nos dá 3 objetivos para provar, o primeiro representado por  $\nabla'_1$ , onde queremos provar que para uma sequência  $x_1$  de tamanho 0, vazia, a adição de um natural  $x$  em uma lista vazia resultam em uma lista ordenada, no caso a lista resultante terá apenas um elemento, estando ordenada, a prova deste ramo pode ser concluída sem maiores dificuldades com os comando apresentados.

Para o  $\nabla'_2$  temos o objetivo de provar que para uma lista  $x_1$  que está ordenada, com tamanho diferente de 0, a adição de um elemento menor ou igual ao primeiro elemento de  $x_1$  no índice 0 resulta em uma lista ordenada. Podemos ver que se a lista está ordenada e o elemento a ser inserido é menor ou igual ao primeiro de  $x_1$  adicioná-lo na primeira posição mantém a ordenação.

$$\frac{\frac{\nabla_4}{\Gamma \Rightarrow k = 0, \Delta} \quad \frac{\nabla_5}{\Gamma, k = 0 \Rightarrow \Delta}}{\Gamma, \forall_k k \leq l(x_1) - 2 \rightarrow x_1.s(k) \leq x_1.s(k+1) \Rightarrow af(x, x_1).s(k) \leq af(x, x_1).s(k+1), \Delta} (Cut)$$

$$\frac{\vdots}{\Gamma, x \leq f(x_1), s?(x_1) \Rightarrow s?(af(x, x_1)), l(x_1) = 0}$$

Interessante desta prova é a utilização da regra de *Cut*, no caso utilizamos  $k = 0$ , assim teremos um ramo provado por  $\nabla_5$  que indica que o valor de  $k$  é 0 oque nos dá realizando expansões que o primeiro elemento,  $af(x, x_1).s(0) \leq af(x, x_1).s(1)$ , expandindo temos  $x \leq x_1.s(0)$  do lado direito do sequente, e temos exatamente isso do lado esquerdo também, dentro do  $\Gamma$ , assim fechamos este ramo.

$$\Gamma, x \leq x_1.s(0) \Rightarrow x \leq x_1.s(0), \Delta \quad (Ax) \tag{2}$$

## Descrição da formalização

### Conclusões

A partir da prova lógica das quatro conjecturas auxiliares apresentadas, é possível provar a funcionalidade da função *fs\_insertion\_sort* empregada como lema no apêndice do trabalho, visto que as questões posteriormente demonstradas são propriedades referentes ao mecanismo de inserção e identificação da ordenação das sequências apresentadas, a prova aplicada da função citada se compõe dos comandos de skolemização da expressão seguida da instanciação do quantificador existencial em variáveis já presentes na expressão sucedida, e então o comando de cut **rewrite** para aplicação dos lemas *fs\_insertionsort\_is\_sorted* e *fs\_insertion\_sort\_is\_permutations*; sendo tais lemas as conjecturas provadas posteriormente pelas questões 2 e 4.

Apesar dos problemas do trabalho designado serem de relativa simplicidade para alguém especializado na área, estes foram fator determinante para a percepção dos integrantes sobre a importância e abrangência de usos da dedução natural e cálculo sequente, independente do software utilizado para a aplicação da teoria.

### Referências

Bibliographic references must be unambiguous and uniform. We recommend giving the author names references in brackets, e.g. [?], [?], and [?].