Alunos:
Davi de Moura Amaral :: 200016750
Marcelo Junqueira Ferreira :: 200023624
Vinícius Lima Passos :: 200028545

Projeto LC1
Equivalência entre noções distintas de permutação

1-Introdução e contextualização do problema.

A noção de permutação é essencial para ordenação, busca e diversos outros algoritmos computacionais. Logo, se mostra necessário analisar se há uma equivalência entre noções de permutações, permitindo uma análise mais ampla na correção lógica. Para que seja possível averiguar o problema é necessário primeiramente entender o que é uma permutação.

Uma noção básica de permutação é a de que "Tendo em mãos uma sequência ordenada qualquer com um número n de elementos distintos, qualquer outra sequência formada pelos mesmos n elementos reordenados é chamada de permutação"[1]. Contudo há inúmeras outras formas de se definir formalmente o conceito de permutação, das quais foram escolhidas duas específicas que serão comparadas ao longo desse trabalho:
A primeira se baseia em uma definição quantitativa do número de ocorrências, a qual diz que duas listas serão permutações uma da outra caso o número de ocorrências dos elementos em uma seja o mesmo na outra.

```
Fixpoint num_oc n l  :=
  match l with
    | nil => 0
    | h :: tl =>
      if n =? h then S(num_oc n tl) else  num_oc n tl
  end.
```

Enquanto a segunda segue uma noção indutiva com as seguintes regras:

$$\frac{}{\texttt{perm}\ l\ l}\ (\texttt{perm\_refl}) \qquad \frac{\texttt{perm}\ l\ l'}{\texttt{perm}\ (x :: l)\ (x :: l')}\ (\texttt{perm\_hd})$$

$$\frac{\texttt{perm}\ l\ l'}{\texttt{perm}\ (x :: y :: l)\ (y :: x :: l)}\ (\texttt{perm\_swap}) \qquad \frac{\texttt{perm}\ l\ l' \qquad \texttt{perm}\ l'\ l''}{\texttt{perm}\ l\ l''}\ (\texttt{perm\_trans})$$

2-Explicação das Soluções.

Para examinar a equivalência entre as noções de permutação apresentadas foi utilizado o processo de dedução natural auxiliado pela ferramenta de software COQ [2].Por meio do software, foi escrito um arquivo de provas “PermEquiv.v” o qual contém as definições necessárias e todo o sequenciamento lógico desenvolvido.

A equivalência foi separada em dois ramos de implicação que foram provados:

- Primeiro uma verificação se uma permutação “perm” indutiva de duas listas l e l’ implica em uma permutação “permutation” das mesmas listas l e l’.
- Segundo, o sentido contrário, uma verificação se uma permutação “permutation” quantitativa de duas listas l e l’ implica em uma permutação “perm” indutiva das mesmas listas l e l’.

Para melhor analisar esses ramos, houve outra separação em 4 “Questões” que envolvem a prova de lemas principais e auxiliares para essas implicações, além da criação de outras lemas a parte que também foram proveitosos.

3-Especificação do problema e explicação do método da solução.

O principal método de resolução do processo de dedução natural foi a realização de induções em hipóteses e em estruturas com definições indutivas ao longo dos lemas propostos.Avançando para as especificações, serão detalhadas as questões e os lemas auxiliares anteriormentes mencionados:

-Questão 1: Prova direta do primeiro ramo de implicação, verificando que uma permutação “perm” implica em uma permutação “permutation”.

Para realizar essa prova, houve a utilização de indução na hipótese da permutação “perm” entre duas listas. Devido ao fata de “perm” ser uma construção indutiva, foi considerado cada regra que compõe a definição indutiva, sendo essas “perm_refl”, “perm_hd”, ”perm_swap”, “perm_trans”.

Essa prova mostra-se simples devido a existência dos lemas anteriormente provados “permutation_refl”, “permutation_hd”, “permutation_trans” e da criação do lema “permutation_2head”, os quais possuem conjecturas similares as regras indutivas da definição de “perm”.

-Questão 2: Prova do lema de apoio “permuation_nil”.
O lema auxiliar consiste na afirmação de que se uma lista l é permutação (seguindo a noção permutation definida pelo número de ocorrências) com “nil”(lista vazia), então l é uma lista vazia.
Devido ao fato da definição da hipótese não ser indutiva, o processo de indução realizado na lista l, havendo uma análise de caso para as regras definição, quando a lista l é construida como uma lista vazia e quando se adiciona um elemnento em uma lista qualquer.
O primeiro caso é trivial, enquanto para o segundo basta observar a contradição da hipótese de indução.

-Questão 3: Prova do lema de apoio “num_oc_cons”.
O lema de apoio trata-se de uma definição mais técnica utilizando a noção de “append” do COQ, na qual afirma que “ se uma lista l possui pelo menos uma ocorrência do elemento x, então l pode ser escrita na forma l1 ++ x :: l2” . Para isso, também foi realizado o processo de indução na lista l, que é definida indutivamente, nisso houve a análise da regra de definição de lista vazia , que igual anteriormente, basta analisar a contradição da hipótes, e da regra de definição de elemento adicionado em uma lista, a qual requiriu maior manipulação lógica para ser provada.

-Questão 4: Prova direta do segundo ramo de implicação, verificando que uma permutação “permutation” implica em uma permutação “perm”.
Para realização dessa prova, houve a utilização da tática indutiva “elim” na lista l. Com isso, no primeiro caso de uma lista vazia houve a utilização do lema de apoio “permuation_nil” anteriormente abordado na “Questão 2”, facilitando a prova. Já para o segundo caso, foi utilizado o lema “num_oc_cons” abordado na “Questão 3” e outros lemas adicionais como “permutation_cons_num_oc”, “permutation_app_cons”, “perm_app_cons” que envolvem a notação de append e outros como “permutation_syn” que envolvem a manipulação de listas que seguem a noção de permutação definida.

4-Descrição da formalização.

Todo o processo descrito ao longo desse texto foi realizado utilizando a lógica de primeira ordem, para que dessa forma houvesse aplicação do processo indutivo nas estruturas lógicas. Processo o qual foi realizado com o já mencionado COQ, que possui diversas funcionalidades para asserções matemáticas, não sendo um proavdor de teoremas automatizado, mas sim um auxiliador de provas, e que utiliza a teoria dos tipos do “Calculus of constructions” [3] , a qual foi desenvolvida em paralelo ao COQ.

5-Conclusões

Primeiramente, com as dois sentidos de implicação provados, é possível, dessa forma, concluir a equivalência entre as duas noções de permutação denotadas, o que mostra que é possível provar a equivalência entre uma definição formal indutiva e uma não indutiva . Ademais foi possível concluir que o processo de prova por indução é facilitado quando a análise decorre em estruturas indutivamente definidas e se complica quando há estruturas não indutivas no contexto, como foi o caso da definição de “permutation” e a questão 4. Também foi possível perceber que o processo de provas computacionalmente necessita de um rigor maior que o desenvolvimento pela linguagem natural. Por fim, foi possível extrair diversos conhecimentos a respeitos dos processos de formalização de provas, dedução natural e softwares de auxílio de provas.

6-Referências

[1] https://brasilescola.uol.com.br/o-que-e/matematica/o-que-e-permutacao.htm

[2] https://coq.inria.fr/

[3] https://en.wikipedia.org/wiki/Calculus_of_constructions