

Section 4.2: Isomorphism of Grammars

In the section we study the isomorphism of grammars.

Copyright © 2003–4 Alley Stoughton

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled “GNU Free Documentation License”.

The L^AT_EX source of these slides, the associated book, and the distribution of the Forlan toolset are available on the WWW at <http://people.cis.ksu.edu/~stough/forlan/>.

1

(4.2) Introduction (Cont.)

Suppose G is the grammar with variables A and B , start variable A and productions:

$$A \rightarrow 0A1 \mid B,$$

$$B \rightarrow \% \mid 2A.$$

And, suppose H is the grammar with variables B and A , start variable B and productions:

$$B \rightarrow 0B1 \mid A,$$

$$A \rightarrow \% \mid 2B.$$

Question: how are G and H related?

Answer: H can be formed from G by renaming the variables A and B of G to B and A , respectively.

As a result, we say that G and H are isomorphic.

2

(4.2) On the Subtlety of Isomorphism

Suppose G is as before, but that H is the grammar with variables 2 and A , start variable 2 and productions:

$$2 \rightarrow 021 \mid A,$$

$$A \rightarrow \% \mid 22.$$

Then H can be formed from G by renaming the variables A and B to 2 and A , respectively.

Question: should we consider G and H to be isomorphic?

Answer: no—the symbol 2 is in both $\text{alphabet}(G)$ and Q_H . G and H generate different languages.

A grammar's variables (e.g., A) can't be renamed to elements of the grammar's alphabet (e.g., 2).

3

(4.2) Definition of Isomorphism

An *isomorphism* h from a grammar G to a grammar H is a bijection from Q_G to Q_H such that:

- h turns G into H ;
- $\text{alphabet}(G) \cap Q_H = \emptyset$, i.e., none of the symbols in G 's alphabet are variables of H .

We say that G and H are *isomorphic* iff there is an isomorphism between G and H .

As expected, we have that isomorphism implies equivalence.

4

(4.2) Renaming Variables

Let $X = \{(G, f) \mid G \in \mathbf{Gram}, f \text{ is a bijection from } Q_G \text{ to some set of symbols, and } \{f(q) \mid q \in Q_G\} \cap \mathbf{alphabet}(G) = \emptyset\}$. The function $\mathbf{renameVariables} \in X \rightarrow \mathbf{Gram}$ takes in a pair (G, f) and returns the grammar produced from G by renaming G 's variables using the bijection f . Then, if G is a grammar and f is a bijection from Q_G to some set of symbols such that $\{f(q) \mid q \in Q_G\} \cap \mathbf{alphabet}(G) = \emptyset$, then $\mathbf{renameVariables}(G, f)$ is isomorphic to G .

5

(4.2) Renaming Variables (Cont.)

The following function is a special case of $\mathbf{renameVariables}$. The function $\mathbf{renameVariablesCanonically} \in \mathbf{Gram} \rightarrow \mathbf{Gram}$ renames the variables of a grammar G to:

- A, B , etc., when the grammar has no more than 26 variables (the smallest variable of G will be renamed to A , the next smallest one to B , etc.); or
- $\langle 1 \rangle, \langle 2 \rangle$, etc., otherwise.

These variables will actually be surrounded by a uniform number of extra brackets, if this is needed to make the new grammar's variables and the original grammar's alphabet be disjoint.

6

(4.2) Finding and Processing Isomorphisms in Forlan

The Forlan module `Gram` contains the following functions for finding and processing isomorphisms in Forlan:

```
val isomorphism           : gram * gram * sym_rel -> bool
val findIsomorphism      : gram * gram -> sym_rel
val isomorphic           : gram * gram -> bool
val renameVariables      : gram * sym_rel -> gram
val renameVariablesCanonically : gram -> gram
```

The function `findIsomorphism` is defined using a procedure that is similar to the one used for finding isomorphisms between finite automata, and `isomorphic` is defined using `findIsomorphism`.

7

(4.2) Forlan Examples

Suppose the identifier `gram` of type `gram` is bound to the grammar with variables `A` and `B`, start variable `A` and productions:

$$\begin{aligned} A &\rightarrow 0A1 \mid B, \\ B &\rightarrow \% \mid 2A. \end{aligned}$$

Suppose the identifier `gram'` of type `gram` is bound to the grammar with variables `B` and `A`, start variable `B` and productions:

$$\begin{aligned} B &\rightarrow 0B1 \mid A, \\ A &\rightarrow \% \mid 2B. \end{aligned}$$

And, suppose the identifier `gram''` of type `gram` is bound to the grammar with variables `2` and `A`, start variable `2` and productions:

$$\begin{aligned} 2 &\rightarrow 021 \mid A, \\ A &\rightarrow \% \mid 22. \end{aligned}$$

8

(4.2) Forlan Examples (Cont.)

Here are some examples of how the above functions can be used:

```
- val rel = Gram.findIsomorphism(gram, gram');
val rel = - : sym_rel
- SymRel.output("", rel);
(A, B), (B, A)
val it = () : unit
- Gram.isomorphism(gram, gram', rel);
val it = true : bool
- Gram.isomorphic(gram, gram'');
val it = false : bool
- Gram.isomorphic(gram', gram'');
val it = false : bool
```