

# SIGMA - Normalization

Isaac Griffith and Rosetta Roberts  
Empirical Software Engineering Laboratory  
Informatics and Computer Science  
Idaho State University  
Pocatello, Idaho, 83208  
Email: {grifisaa@isu.edu, roberose@isu.edu}

## **Abstract—Introduction:**

### **Objective:**

### **Methods:**

### **Results: What are the main findings? Practical implications?**

### **Limitations: What are the weaknesses of this research?**

### **Conclusions: What is the conclusion?**

### **Index Terms—Island Grammars, Automated Grammar Formation, Software Language Engineering**

## I. INTRODUCTION

Multilingual parsing is an open problem that is currently being worked on. One method of multilingual parsing that has been developed is by creating island grammars for the combined grammars [1]. An automated method for doing this is currently being developed [SIGMA REFERENCE HERE]. One challenge to automating the creation of island grammar based multilingual parsers is detecting similar parts of grammars and combining them. This process becomes easier with an appropriate normalization process and normal form.

### **Outline:**

1. Mention current SIGMA tool.
2. Normalization section is untested.

**RG** Design a normalization process for grammars that is conducive to identifying and merging similar parts across grammars to be used for merging grammars for the automated creation of multilingual parsers.

## *Organization*

## II. BACKGROUND AND RELATED WORK

### *A. Theoretical Foundations*

testhateh ## Related Studies {#sec:related}

### *B. Research Contributions*

## III. APPROACH

### *A. Normalization*

Next, the trivially merged grammar is normalized to a unique normal form wherein every production is one of the following two forms:

$$\langle A \rangle ::= \langle B \rangle 'a' \dots$$
$$\langle B \rangle ::= \langle A \rangle \mid 'b' \mid \dots \mid \epsilon$$

Where production  $\langle A \rangle$  represents form  $F_1$  and production  $\langle B \rangle$  represent form  $F_2$ . These forms were selected to ease comparison of similar productions. Note, as per this normalization, neither form may be recursively defined. The rational behind

this, is to ensure that regardless of how productions are nested they will normalize to the same form, for example: Given grammar  $EG_1$

$$\langle A \rangle ::= 'a' \langle B \rangle$$
$$\langle B \rangle ::= 'b' 'c'$$

and grammar  $EG_2$

$$\langle A \rangle ::= \langle B \rangle 'c'$$
$$\langle B \rangle ::= 'a' 'b'$$

both  $EG_1$  and  $EG_2$  normalize to the same grammar, as follows:

$$\langle A \rangle ::= 'a' 'b' 'c'$$

Normalization continues, repeatedly, through the following six processes until stabilization: eliminating unused rules, simplifying productions, merging equivalent rules, eliminating unit rules, expanding productions, and collapsing compatible productions.

1) *Eliminating Unused Rules:* Remove all rules that are not produced, directly or indirectly, from the start rule. This is accomplished by marking all rules enumerated via a depth first search, and then dropping unmarked rules. When applied to  $G_3$  the grammar is transformed into grammar  $G_4$ , as depicted in Fig. 1a.

2) *Simplifying Productions:* To simplify in-memory grammar representations, the process removes  $\epsilon$  embedded inside terms. Next, productions containing one term are replaced with that term. When this step is applied to grammar  $G_4$ , it is transformed into grammar  $G_5$ , as depicted in Fig. 1b

3) *Merging Equivalent Rules:* Rules that have identical productions are replaced by a single rule. This new rule is given a name derived from the rules that were merged to create it. In the following example of this step, rules a and b are merged into the rule a+b:

$$\langle s \rangle ::= \langle a \rangle \mid \langle b \rangle$$
$$\langle a \rangle ::= 'a' 'b' \langle a \rangle$$
$$\langle b \rangle ::= 'a' 'b' \langle a \rangle$$

Which after equivalent rules are merged yields:

$$\langle s \rangle ::= \langle a+b \rangle$$
$$\langle a+b \rangle ::= 'a' 'b' \langle a+b \rangle$$

When this step is applied to grammar  $G_5$ , it is transformed into grammar  $G_6$ , as depicted in Fig. 1c.

$\langle S \rangle ::= \langle S_1 \rangle \mid \langle S_2 \rangle$	$\langle S \rangle ::= \langle S_1 \rangle \mid \langle S_2 \rangle$
$\langle S_1 \rangle ::= \langle A \rangle \mid \langle B \rangle$	$\langle S_1 \rangle ::= \langle A \rangle \mid \langle B \rangle$
$\langle A \rangle ::= 'a' \varepsilon \langle B \rangle \langle C_1 \rangle$	$\langle A \rangle ::= 'a' \langle B \rangle \langle C_1 \rangle$
$\langle C_1 \rangle ::= 'c'$	$\langle C_1 \rangle ::= 'c'$
$\langle B \rangle ::= 'b' 'd'$	$\langle B \rangle ::= 'b' 'd'$
$\langle S_2 \rangle ::= \langle C_2 \rangle \mid \langle D \rangle$	$\langle S_2 \rangle ::= \langle C_2 \rangle \mid \langle D \rangle$
$\langle C_2 \rangle ::= 'c'$	$\langle C_2 \rangle ::= 'c'$
$\langle D \rangle ::= 'a' 'd' ('e' \mid 'c')$	$\langle D \rangle ::= 'a' 'd' ('e' \mid 'c')$
$\mid \langle C_2 \rangle \mid 'b')$	$\mid \langle C_2 \rangle \mid 'b')$
(a) Grammar $G_4$ .	(b) Grammar $G_5$ .
$\langle S \rangle ::= \langle S_1 \rangle \mid \langle S_2 \rangle$	$\langle S \rangle ::= \langle S_1 \rangle \mid \langle S_2 \rangle$
$\langle S_1 \rangle ::= \langle A \rangle \mid \langle B \rangle$	$\langle S_1 \rangle ::= \langle A \rangle \mid \langle B \rangle$
$\langle A \rangle ::= 'a' \langle B \rangle \langle C \rangle$	$\langle A \rangle ::= 'a' \langle B \rangle 'c'$
$\langle C \rangle ::= 'c'$	$\langle B \rangle ::= 'b' 'd'$
$\langle B \rangle ::= 'b' 'd'$	$\langle S_2 \rangle ::= 'c' \mid \langle D \rangle$
$\langle S_2 \rangle ::= \langle C \rangle \mid \langle D \rangle$	$\langle D \rangle ::= 'a' 'd' ('e' \mid 'c')$
$\langle D \rangle ::= 'a' 'd' ('e' \mid 'c')$	$\mid ('c' \mid 'b')$
$\mid \langle C \rangle \mid 'b')$	(d) Grammar $G_7$ .
(c) Grammar $G_6$ .	
$\langle S \rangle ::= \langle S_1 \rangle \mid \langle S_2 \rangle$	$\langle S \rangle ::= \langle A \rangle \mid \langle B \rangle \mid 'c' \mid$
$\langle S_1 \rangle ::= \langle A \rangle \mid \langle B \rangle$	$\langle D \rangle$
$\langle A \rangle ::= 'a' \langle B \rangle 'c'$	$\langle A \rangle ::= 'a' 'b' 'd' 'c'$
$\langle B \rangle ::= 'b' 'd'$	$\langle B \rangle ::= 'b' 'd'$
$\langle S_2 \rangle ::= 'c' \mid \langle D \rangle$	$\langle D \rangle ::= \langle D_1 \rangle \mid 'c' \mid 'b'$
$\langle D \rangle ::= \langle D_1 \rangle \mid \langle D_3 \rangle$	$\langle D_1 \rangle ::= 'a' 'd' \langle D_2 \rangle$
$\langle D_1 \rangle ::= 'a' 'd' \langle D_2 \rangle$	$\langle D_2 \rangle ::= 'e' \mid 'c'$
$\langle D_2 \rangle ::= 'e' \mid 'c'$	(f) Grammar $G_9$ .
$\langle D_3 \rangle ::= 'c' \mid 'b'$	
(e) Grammar $G_8$ .	

Fig. 1. Transformed grammars produced during the normalization of grammar  $G_3$ .

4) *Eliminating Unit Rules*: All non-terminals with productions of one of the following two forms will have their non-terminal symbols replaced by their rules, and their productions eliminated.

$$\begin{aligned} \langle a \rangle &::= \langle b \rangle \\ \langle a \rangle &::= 'a' \end{aligned}$$

Elimination of productions of the first form, is derived from Chomsky Normal Form (CNF) [X]. Eliminations of productions of the second form, a derivation from CNF, allows the simplification process to simplify rules of the following form:

$$\begin{aligned} \langle a \rangle &::= \langle b \rangle 'a' 'b' \\ \langle b \rangle &::= \varepsilon \end{aligned}$$

When this step is applied to grammar  $G_6$ , it is transformed into grammar  $G_7$ , as depicted in Fig. 1d.

5) *Expanding Productions*: Productions that have nested rules have all nested content replaced by with a non-terminal. The new non-terminal defines a production pointing to their content. When this step is applied to grammar  $G_7$ , it is transformed into grammar  $G_8$ , as depicted in Fig. 1e.

6) *Collapsing Compatible Productions*: The final step of the normalization process combines productions that are com-

patible with each other. This ensures that any non-terminal symbols referenced in a rule will not define a duplicate production. The following provides an example:

$$\begin{aligned} \langle A \rangle &::= 'a' \langle B \rangle \\ \langle B \rangle &::= 'b' 'c' \\ \langle C \rangle &::= 'c' \mid \langle D \rangle \\ \langle D \rangle &::= 'd' \mid 'e' \end{aligned}$$

would then collapse to form:

$$\begin{aligned} \langle A \rangle &::= 'a' 'b' 'c' \\ \langle C \rangle &::= 'c' \mid 'd' \mid 'e' \end{aligned}$$

When this step is applied to grammar  $G_8$ , it is transformed into grammar  $G_9$ , as depicted in Fig. 1f.

## IV. EXPERIMENTAL DESIGN

### A. Goals, Hypotheses, and Variables

Goals: - Evaluate each step of normalization

Hypotheses: - Null hypothesis: each step has no effect on the halstead effort for MCC.

Variables: - Size of grammar as chosen in SIGMA - Blocking, - Halstead/MCC - The steps that the normalization goes through. ( $2^{\text{numSteps}}$ )

### B. Design

Blocked factorial design

Reasons - Effect from size found in SIGMA - Interactions from size also seen in SIGMA

### C. Experimental Units

### D. Data Collection Procedures

### E. Analysis Procedures

### F. Evaluation of Validity

*Conclusion Validity:*

*Internal Validity:*

*Construct Validity:*

*External Validity:*

## V. RESULTS

- A. *Data Set Reduction*
- B. *Descriptive Statistics*
- C. *Hypothesis Testing*

## VI. INTERPRETATION

- A. *Evaluation of Results*
- B. *Limits of the Study*
- C. *Inferences*
- D. *Lessons Learned*

## VII. THREATS TO VALIDITY

## VIII. CONCLUSIONS AND FUTURE WORK

- A. *Summary of Findings*
- B. *Relation to Existing Evidence*
- C. *Impact*
- D. *Limitations*

## ACKNOWLEDGEMENTS

## REFERENCES

- [1] N. Synytskyy, J. R. Cordy, and T. R. Dean, "Robust multilingual parsing using island grammars," in *Proceedings of the 2003 Conference of the Centre for Advanced Studies on Collaborative Research*. IBM Press, 2003, pp. 266–278.