

UNIVERSITY OF COPENHAGEN  
Computer Science Department  
Data-Parallel Compilation  
Lexical analysis & Syntax Tree Construction

William Henrich Due (mcj284)  
Submitted: 5th of April 2024

**Abstract**

Abstract.

## 1 Introduction

Introduction.

## 2 Theory

Hills paper “Parallel lexical analysis and parsing on the AMT distributed array processor” [1] describes a method to obtain the path in a deterministic finite automata given a input string. This section will describe the theory of this method and extend the it for tokenization.

### 2.1 Data-parallel Lexical Analysis

To explain the theory of parallel lexical analysis we first remind the reader of the definition of a deterministic finite automaton.

**Definition 2.1** (DFA). A deterministic finite automata [2] [3] is given by a 5-tuple  $(Q, \Sigma, \delta, q_0, F)$  where.

1.  $Q$  is the set of states where  $|Q| < \infty$ .
2.  $\Sigma$  is the set of symbols where  $|\Sigma| < \infty$ .
3.  $\delta : \Sigma \times Q \rightarrow Q$  is the transition function.
4.  $q_0 \in Q$  is the initial state.
5.  $F \subseteq Q$  is the set of accepting states.

This definition is fine as is but we will need to reformualte it to develop data-parallel lexical analysis. We would want the definition to use a curried transition function. But for this to hold then the DFA would also have to be total.

---

**Definition 2.2** (Total DFA). A DFA  $(Q, \Sigma, \delta, q_0, F)$  is said to be total if and only if

$$\delta(a, q) \in Q : \forall(a, q) \in \Sigma \times Q$$

If a DFA is total we may use a curried transition function  $\delta : \Sigma \rightarrow Q \rightarrow Q$ .

This is needed since else the the function would not be fully defined in the domains  $\Sigma$  and  $Q$ .

The reason for doing so is because if we have any two functions  $g = \delta(a)$  and  $f = \delta(a')$  then it follows from composition that.

$$g(f(q)) = (g \circ f)(q)$$

This allows for an alternative way of determining if a string can be produced by an DFA. Instead of first evaluating  $f(q)$ , then  $g(f(q))$  and then checking if this state is a member of  $F$ . We could instead partially apply  $\delta$  to the symbols and then compose them to a single function which could be used to determine if a string is valid. This sets the stage for data-parallel lexing, we want to find a way to make the problem into a **map-reduce**. We want to do this because it can be computed using a data-parallel implementation unlike the normal way of traversing a DFA.

For the ability to use a data-parallel **map-reduce** we must have a monoidal structure. Here  $\Delta$  is the set of all the composed partially applied  $\delta$  functions needs to be closed under function composition.

**Proposition 2.1** (DFA Composition Closure). Given a total DFA, the set of all partially applied function compositions (Endofunctions) that is closed under function composition  $\Delta : \{Q \rightarrow Q\}$ . Is the set  $\Delta = \Delta_i$  in the recurrence relation with the smallest  $i$  such that  $\Delta_i = \Delta_{i+1}$ .

$$\begin{aligned}\Delta_1 &= \{\delta(a) : a \in \Sigma\} \\ \Delta_{i+1} &= \Delta_i \cup \{f \circ g : f, g \in \Delta_i\}\end{aligned}$$

*Proof.* We will start by showing that a solution  $\Delta$  exists. First note that the cardinality is monotonically increasing i.e.  $\Delta_i \subseteq \Delta_{i+1}$  since  $\Delta_{i+1}$  is a union of  $\Delta_i$  and another set. Secondly note that since  $|Q| < \infty$  then a finite amount of functions of the form  $Q \rightarrow Q$  can be created. Since the set is bounded and increasing then at some point  $\Delta_i = \Delta_{i+1}$  and the smallest  $i$  where it holds is the solution  $\Delta$ .

For  $\Delta$  to be closed under function composition, then for arbitray  $f, g \in \Delta$  it must hold that  $f \circ g \in \Delta$ . Since  $\Delta_1$  is the set of endofunctions that constructs  $\Delta$  then all elements of  $\Delta$  can be expressed on the form.

$$\delta(a_1) \circ \dots \circ \delta(a_n) \in \Delta$$

---

If all combinations of  $\Delta_1$  of any sequence length are members of  $\Delta$  then  $\Delta$  would be closed under function composition. Furthermore, it is known that  $\Delta$  is finite so combinations after some  $\Delta_i = \Delta_{i+1}$  would only add new sequences of compositions but no new endofunctions. Therefore it suffices to show that if all combinations of length  $k$  where  $1 \leq k \leq i$  is a subset of  $\Delta_i$  then  $\Delta$  is closed under function composition. This can be shown using a proof by induction.

Base:  $\Delta_1$  trivially holds since it only contains sequences of length one and they are the initial endofunctions.

Step: Given  $\Delta_i$  contains every combination of length  $i$  or less then we to show this implies that  $\Delta_{i+1}$  will contain every combination of length  $i+1$  or less.

By definition  $\Delta_{i+1}$  it must contain every of length  $i$  or less due to  $\Delta_i \leq \Delta_{i+1}$ . It remains to show that every sequence of length  $i+1$  is a member of  $\Delta_{i+1}$ . We know  $\Delta_i$  will contain every sequence of length 1 and  $i$  by monotonicity. It is also known that a direct product of  $\Delta_i$  is used in the definition meaning  $\{f \circ g : f, g \in \Delta_i\} \subseteq \Delta_{i+1}$ . There by a direct product between sequences of length 1 and  $i$  leading to every sequence of length  $i+1$  being a member of  $\Delta_{i+1}$ . Thereby  $\Delta$  is closed under function composition.  $\square$

Since  $\Delta$  is closed under function composition then it follows that  $\Delta$  can induce a monoidal structure.

**Corollary 2.1** (Total DFA monoid). DFA composition closure induces a semigroup which intern induces the monoid  $(\Delta \cup \{id\}, \circ)$  where  $id : Q \rightarrow Q$  and  $id(q) = q$ .

Knowing this we can establish the following algorithm

**Algorithm 2.1** (Data-parallel Lexical Analysis). It can be determined in  $O(n)$  work and  $O(\log n)$  span if a string can be produced by a DFA. First construct the total DFA  $(Q, \Sigma, \delta, q_0, F)$  from the DFA.

1. Partially apply  $\delta$  to every symbol in the input string such that it becomes a sequence of endofunctions.
2. Reduce the endofunction in to a single endofunction  $\delta' : Q \rightarrow Q$ .
3. Evaluate  $\delta'(q_0)$  and determine if  $\delta'(q_0) \in F$ .

---

## 2.2 Parallel Tokenization

## 3 Conclusion

Conclusion.

## References

- [1] Jonathan M.D Hill. “Parallel lexical analysis and parsing on the AMT distributed array processor”. In: *Parallel Computing* 18.6 (1992), pp. 699–714. ISSN: 0167-8191. DOI: [https://doi.org/10.1016/0167-8191\(92\)90008-U](https://doi.org/10.1016/0167-8191(92)90008-U). URL: <https://www.sciencedirect.com/science/article/pii/016781919290008U>.
- [2] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. USA: Addison-Wesley Longman Publishing Co., Inc., 2006. ISBN: 0321455363.
- [3] Wikipedia contributors. *Deterministic finite automaton* — *Wikipedia, The Free Encyclopedia*. [Online; accessed 4-February-2024]. 2023. URL: [https://en.wikipedia.org/w/index.php?title=Deterministic\\_finite\\_automaton&oldid=1192025610](https://en.wikipedia.org/w/index.php?title=Deterministic_finite_automaton&oldid=1192025610).