The Relation Between Deterministic Pushdown Automata and Context-Free Languages

Feng Zonghao[1]

1. Deterministic Pushdown Automata

As the name suggests, a **deterministic pushdown automaton** (**DPDA**) is a variation of the **pushdown automaton** (**PDA**). To follow the principles of determinism, the DPDA has at most one legal transition at each step of its computation, which differs it from the PDA.

We all know that **deterministic finite automata** (**DFA**) and **nondeterministic finite automata** (**NFA**) are equivalent in language recognition power, but does this property still holds when it comes to the DPDA and the PDA? To figure out the answer, we could check the formal definition of the DPDA [3] first:

Definition 1.

A **deterministic pushdown automaton** is a 6-tuple (Q, Σ, Γ, δ, q_0, F), where Q, Σ, Γ, and F are all finite sets, and

1. Q is the set of states,
2. Σ is the input alphabet,
3. Γ is the stack alphabet,
4. $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to (Q \times \Gamma_\varepsilon) \cup \{\emptyset\}$ is the transition function,
5. $q_0 \in Q$ is the start state, and
6. $F \subseteq Q$ is the set of accept states.

The transition function δ must satisfy the following condition. For every $q \in Q$, $a \in \Sigma$, and $x \in \Gamma$, exactly one of the values

$$\delta(q, a, x), \delta(q, a, \varepsilon), \delta(q, \varepsilon, x), \text{and } \delta(q, \varepsilon, \varepsilon)$$

is not $\emptyset$.

Notice that ε-moves are allowed in the DPDA's transition function, even though ε-moves are prohibited in DFAs. But comparing to the PDA, we add the restriction that ε-moves are mutually exclusive to non-ε-moves in a certain situation of DPDA.

2. Deterministic Context-Free Languages

We call the language that can be accepted by DPDA as the **deterministic context-free language** (**DCFL**). The DPDA is unable to choose between different state transition alternatives, so that it cannot recognize all CFL. That is, DPDA and PDA are not equivalent in language recognition power. Formally, if $L(A)$ is a language accepted by a PDA A, it can also be accepted by a DPDA if and only if there is a unique computation routine from the initial configuration to an accepting one for all strings belonging to $L(A)$.

[1]Computer Science Class 2013, Taishan College, Shandong University. Student number: 201300130013

To help understanding, consider this example: The language of even-length palindromes on the alphabet of 0 and 1 has the context-free grammar $S \to 0S0|1S1|\varepsilon$. An arbitrary string of this language cannot be parsed without reading all its letters first, which means that a PDA has to try several alternative state transitions to select from different possible lengths of a half-parsed string. [1] Not all context-free languages are deterministic, which makes the DPDA a strictly weaker device than the PDA.

DCFL has great applications under various computer science research areas, such as compiler design, etc. For instance, DCFL can be accepted in $O(n)$ time by a LR(k) parser [2], while CFL need approximately $O(n^{2.378})$ time to parse even if we use the best method so far.

3. Conclusion

From previous discussion, we could draw the conclusion that the relation between **deterministic pushdown automata** and **context-free languages** is: the DPDA accepts the DCFL, which is a proper subset of CFL. The relation could be illustrated by the graph below:

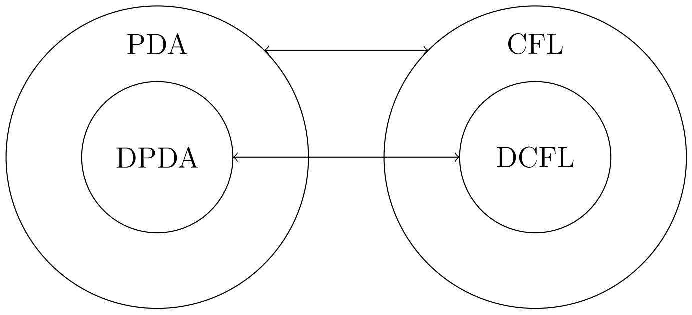

Figure 1. Relation between DPDA and CFL

References

1. John E Hopcroft, Rajeev Motwani, and Jeffrey D Ullman. Introduction to automata theory, languages, and computation (2 ed.). pages 249–253, 2001.

2. Donald E Knuth. On the translation of languages from left to right. *Information and control*, 8(6):607–639, 1965.

3. Michael Sipser. Introduction to the theory of computation (3 ed.). pages 130–131, 2012.