

大規模正規表現の高速照合方式

中村 隆顕 郡 光則

三菱電機株式会社 情報技術総合研究所

1. はじめに

近年、ネットワークセキュリティなどの分野において、ネットワークを流れる電子データを、そのデータに含まれるテキストを照合することによってフィルタリングする技術が重要になっている。その一方で、特に検索条件が大規模な場合に、電子データの転送速度の高速化に照合速度が追従できなくなりつつある。

本稿では、正規表現を含んだ検索条件によって、テキストを高速に照合する sDFA 方式を提案する。本方式は、DFA (Deterministic Finite Automaton) 方式のメモリ消費量を大幅に削減することにより、大規模な検索条件に対して高速な照合を実現した。本稿では、sDFA 方式の有効性の評価結果についても報告する。

2. 文字列照合

本稿で提案する sDFA 方式は、テキストの中から、検索条件として指定された正規表現に一致する文字列の出現位置を、全て出力することを目的とする。特に、検索条件が大規模な場合や、テキストにマルチバイト文字が含まれる場合に有効な方式を提案する。

3. sDFA 方式

3.1. 課題

従来、正規表現による文字列照合方式として、状態遷移機械の NFA (Non-Deterministic Finite Automaton) や DFA を利用した方式がある[1]。NFA 方式は、メモリの消費量が少ないものの、検索条件規模の増大に従って、照合速度性能が大幅に低下する課題がある。DFA 方式では、安定して高速な照合が可能であるが、検索条件の規模が増大するに従って、状態数や状態遷移数が指数的に増加する。そのために、メモリを圧迫し照合が不可能になる課題がある。いま、検索条件に含まれるユニークな文字数を m 、全入力文字種のを n とすると、DFA の状態数は $O(2^m)$ 、状態遷移数は $O(2^m n)$ である[2]。

これらの文字列照合方式は、欧米において文字種の少ない文字を対象として発達してきた。それらと比較して、文字種が非常に多いマルチバイト文字をテキストに含む場合に、特に高速な照合が困難となっている[3]。

3.2. 状態遷移の削減

本稿で提案する sDFA 方式では、高速な照合が可能な DFA 方式を基本とする。DFA 方式の課題であった状態遷移数を削減するため、Default 遷移と AnyOther 遷移の 2 種類の状態遷移を導入した。

Default 遷移は、現在の状態から入力文字による状態遷移先が無い場合に、文字を読み進めずに初期状態に遷移する状態遷移である。

1. 入力文字について、状態遷移先が定義されている場合は、定義されている状態遷移先へ遷移する。
2. 状態遷移先が定義されていない場合は、Default 遷移する。

Default 遷移により、ある入力文字に対して状態遷移先が定義されていない場合に、初期状態への状態遷移を削減することができる。同時に、初期状態からその文字によって遷移可能な状態への状態遷移も削減することができる。

AnyOther 遷移は、現在の状態から、状態遷移先が定義されていない任意の文字に対して、同一の状態に遷移する状態遷移である。

1. 入力文字について、状態遷移先が定義されている場合は、定義されている状態遷移先へ遷移する。
2. 状態遷移先が定義されていない場合は、AnyOther 遷移する。

AnyOther 遷移は、正規表現に除外文字や任意文字を含む場合に、特に有効である。

以上のように、sDFA では、Default 遷移や AnyOther 遷移を適用することにより、状態遷移数を大幅に削減することができる。状態遷移の削減によって疎となった状態遷移表において、状態遷移が定義されているエントリだけをメモリ上に置くことにより、検索時のメモリ消費量を大幅に削減できる。

4. 性能評価

本稿で提案する sDFA 方式を実装し、その有効性を評価した。

4.1. 状態遷移数

sDFA 方式と、DFA 方式の状態遷移表に含まれる有効な状態遷移の数を比較した。DFA の状態遷移数は、sDFA の状態数 × 検索条件中のユニークな文字数 (=m) とした場合 (DFA1) と、sDFA の状態数 × 日本語 EUC の全文字数 (=n) とした場合 (DFA2) の 2 通りを考慮する。

まず、検索条件(1)を以下に示す正規表現(日付)としたときの、sDFA 方式と DFA 方式の状態遷移数を比較した。この正規表現のユニークな文字数は m=21 である。

```
((明治|大正|昭和|平成)?[0-9]{1,2})|[0-9]{4})(年|/)(0|1)?[0-9](月|/)(0|1|2)?[0-9]
```

状態遷移数の比較結果を表 1 に示す。

表 1 状態遷移数

sDFA	DFA1	DFA2
141	483	2258048

次に、検索条件(2)として 1 個以上の日本語の固定文字列キーワード(人名)を用い、そのキーワード数を変化させたときの、状態遷移数の変化を図 1 に示す。

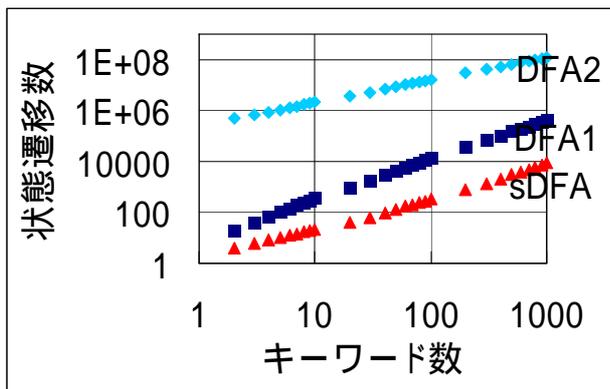


図 1 キーワード数依存状態遷移数

以上より、sDFA は、DFA と比較して状態遷移数が大幅に削減できることが確認できた。

4.2. 照合速度性能

sDFA を実装し、Linux の正規表現照合ライブラリ GNU Regex (DFA 方式)、Windows の正規表現ライブラリ .NET Regex (NFA 方式) との照合速度性能を比較した。測定条件を表 2 に示す。

表 2 性能評価条件

	sDFA	GNU Regex	.NET Regex
OS	Linux Fedora Core2 (kernel2.6.5-1)		Windows 2003 Server
CPU	Pentium4 3.2GHz		Xeon 3.2GHz
Memory	1GB		4GB
コンパイラ	gcc3.3.3-7		VC#.NET2003

照合速度性能の比較では、平均 1.6 万文字の日本語のテキスト 5,720 件に対して照合速度を測定し、その平均値を求めた。検索条件は、「4.1 状態遷移数」と同様の条件を用いた。テキストと検索条件の文字コードは、Linux が日本語 EUC、Windows が Shift-JIS である。

検索条件(1)による照合速度を表 3 に示す。

表 3 照合速度

sDFA	GNU Regex	.NET Regex
9908 万文字/秒	152	102

検索条件(2)のキーワード数を変化させたときの、照合速度の変化を図 2 に示す。

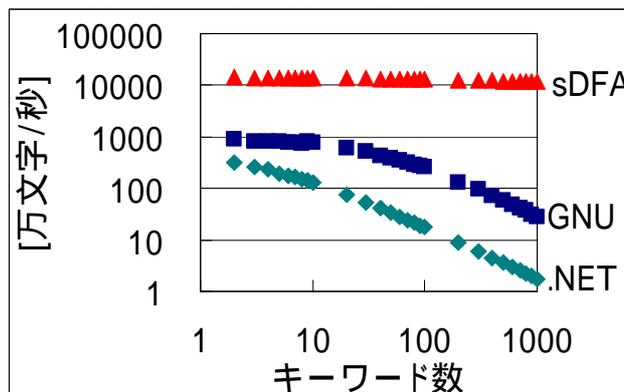


図 2 キーワード数依存照合速度

以上より、sDFA は、GNU Regex や、.NET Regex より高速であることを確認できた。特に、GNU Regex や、.NET Regex では、キーワード数の増加に従って、照合速度が大幅に低下しているのに対して、sDFA 方式では、照合速度は 1 億文字/秒で安定していることが確認できた。

5. まとめ

DFA の状態遷移数を削減することにより、テキストを正規表現によって高速に照合する方式を提案した。また、評価により、その有効性を確認した。本方式は、検索条件が大規模である場合や、テキストに日本語などのマルチバイト文字を含む場合に、特に有効である。

参考文献

- [1] E.J.Hopcroft, D.J.Ullman, Formal Languages and their Relation to Automata, Addison Wesley, 1969.
- [2] G Navarro, Pattern Matching, Journal of Applied Statistics, 31(8) 925-950, 2004.
- [3] 長谷川 勇, マルチバイトキャラクタを扱う決定性有限状態オートマトンの構成法, Linux Conference 2001, 2001.