ログ検索向け正規表現検索条件の自動生成方式

松本 良央† 加藤 守† 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、ログなどの非定型テキストデータの利活用の要望が高まっているが、ログを検索する際に、正規表現による検索条件を人手で誤りなく作成するのは難しいという課題があった。

本稿では、10進数値などの範囲や検索開始位置を指定する正規表現の自動生成方式について提案し、その実装および評価結果について報告する。

2. 背景と課題

筆者らは多様で大規模なログを形式にとらわれず完全に復元可能な形式で保存、管理することができるログ専用データベース管理システム「ログデータベース」を開発した[1]。

ログデータベースでは、検索時にログの形式を判定するために正規表現を用いるが、正規表現による検索条件を生成する時に、一定の範囲の属性値を検索したり検索位置を指定するための正規表現を簡単に誤りなく記述生成することは困難である。

例えば、ある範囲の正規表現をその範囲に含まれる要素を全て“|”を使って表すことは、長大な範囲の場合膨大な式となってしまい、検索時にメモリを大量に消費し、照合に時間がかかるという課題があった。

3. 正規表現検索条件の自動生成実現方式

ここでは、「10 進数値の範囲指定」と「位置指定」を例にその実現方式について説明する。これらは、範囲指定の単独、または位置指定と範囲指定の組合せによる利用が可能である。

3.1 10進数値の範囲指定

ある 10 進数値の下限値から上限値の範囲を含む値の指定において、効率的に正規表現を生成するために下限値と上限値の間に含まれる値を下位、中位、上位の3領域に分割する。それぞれの領域において正規表現を生成し、それらのデータを結合することにより、その範囲の正規表現を生成する。

3.1.1 各領域の定義

中位領域は、下限値から上限値の範囲の中で[0-9]という正規表現で定義できる最大限の範囲のことを指す。例えば 1～1000 の正規表現は、[1-9]|[1-9][0-9]{1,2}|100[0-0] で表され、“[1-9][0-9]{1,2}”が中位領域に該当する。

下位領域、上位領域は中位領域の範囲を算出することで定義できる。下位領域、上位領域は、それぞれ属性範囲条件の下限値～中位領域の下限値、中位領域の上限値～属性範囲条件の上限値の範囲のことを指す。各領域の算出方法については以降で詳しく説明する。

まず、図1のように変数を定義する。

E：求める正規表現、L：下位領域、
M:中位領域、H：上位領域
n：下限値の桁数、m：上限値の桁数、
$A_1A_2\cdots A_n$：下限値、$B_1B_2\cdots B_m$：上限値、
k：下限値と上限値で最上位桁から数字が同じ桁数

図1. 変数の定義

この時、正規表現の初期状態は、次のように定義できる。

E=L+″|″+M+″|″+H・・・(1)

本稿では 10 進数値の範囲を例に説明するが、日付や時刻に関しても、同様に 3 つの領域に分けて正規表現に変換する方法の適用が可能である。これらの場合は、年、月、日、時、分、秒の単位毎に最下位桁が[0-9]という正規表現で定義できる最大限の範囲の中位領域を単位毎の繰り上がりを考慮した上で 10 進数値の範囲を求める形で算出してから、下位領域、上位領域を算出する。

3.1.1.1 中位領域の定義

中位領域の正規表現は次のように定義できる。

1) m-n≧2
M=″[1-9][0-9]{n,m-2}″・・・(2)

2) m-n=1
M=″″・・・(3)

3) n=m
a) 下限値と上限値の下一桁のみの数値が違う場合と下限値と上限値が同じ場合
M=″″・・・(4)
b) a)以外
$M=″A_1A_2..A_k[(A_{(1+k)}+1)-(B_{(1+k)}-1)][0\text{-}9]\{n\text{-}k\text{-}1\}″\cdots(5)$

Automatic Creation of Regular Expression for Searching Log
Yoshio Matsumoto†, Mamoru Kato†, Takaaki Nakamura†, Mitsunori Kori†
Information Technology R&D Center, Mitsubishi Electric Corporation.

3.1.1.2 下位領域と上位領域の定義

下位領域と上位領域の正規表現は次のように定義できる。

L="explower($A_1 \cdots A_n$, n)"・・・(6)

H="expupper($B_1 \cdots B_m$, m)"・・・(7)

下位領域と上位領域の正規表現への変換について擬似プログラムコードを使って具体的に説明する。

上記式(6)の下位領域を求める explower の動作は図2の通り。

```
Ex="";
for(i=2;i<=n;i++){
  if((A(n-(i-1)))=!9){
Ex=Ex+"A1..A(n-i)[(A(n-(i-1))+1)-9][0-9]{i-1}|";}}
L=Ex"(A1...A(n-1)[An-9]|)("+Ex+")";
```

図2. 下位領域の擬似プログラムコード

上記式(7)の上位領域を求める expupper の動作は図3の通り。

```
Ex="";
for(j=2;j≦m;j++){
Ex=Ex+"B1...B(m-j)[0-(B(m-(j-1))-1)][0-9]{j-1}|";}
H=Ex"(B1...B(m-1)[0-Bm]|)("+Ex+")";
```

図3. 上位領域の擬似プログラムコード

3.2 位置指定

本方式では、カンマ区切り CSV やスペース区切り形式のログを対象とし、カンマやスペースなどの区切り文字によってフィールドに区切られた行の検索開始位置を指定し、指定された位置以降のすべての文字列を検索対象とする。

3.2.1 正規表現変換

位置指定は図4のような条件設定で単純に正規表現への変換が可能である。ここでは連続した区切り文字のカンマを一文字として扱わない場合の例を示し、上段がフィールド位置指定、下段が検索方法を表す。

```
・フィールド番号=1：^
・フィールド番号=2：^(([^,¥"]*,)|(¥"(([^¥"]*)|
  (([^¥"]*¥"¥"[^¥"]*)*))¥",))
・フィールド番号>2："^(([^,¥"]*,)|
  (¥"(([^¥"]*)|(([^¥"]*¥"¥"[^¥"]*)*))¥",))
  {検索対象のフィールド番号-1}"
```

```
・前方一致：なし（ヌル文字列）
・中間一致：.*
・フィールド内中間一致：(([^,¥"]*)|
  (¥"([^¥"]|(¥"¥"))*))
```

図4. 正規表現変換

前方一致は、指定されたフィールドの最初の文字から照合する。中間一致は、指定されたフィールドから行末までの間の任意の位置から照合する。フィールド内中間一致は、指定されたフィールドから次のフィールドまでの間の任意の位置から照合する。

位置指定正規表現 EP は次のように定義できる。

EP="フィールド位置指定+検索方法"・・・(8)

4. 正規表現検索条件自動生成方式の実装

本稿で提案した正規表現検索条件の自動生成方式を Java および C 言語プログラムに実装した。

筆者らは、DFA（Deterministic Finite Automaton）のメモリ消費量を大幅に削減することにより、大規模な検索条件の高速な照合方式 sDFA（size-reduced DFA）を提案し、ログデータベースに適用した[1][2]。本評価では、sDFA におけるメモリ消費量を決める支配的要因となる状態遷移数と、実際のログに対する照合速度を評価し、その有効性を確認した。

図5に評価環境を、表1に評価に使用した数値範囲の正規表現を示す。

```
OS：Windows Server 2003 SP2
CPU：Xeon 3.20GHz、Memory：4GB
```

図5. 評価環境

表1 正規表現例

数値範囲	本方式による正規表現
123456789 ～ 500000000	12345678[9-9]|1234567[9-9][0-9]{1}|123456[8-9][0-9]{2}|12345[7-9][0-9]{3}|1234[6-9][0-9]{4}|123[5-9][0-9]{5}|12[4-9][0-9]{6}|1[3-9][0-9]{7}|[2-4][0-9]{8}|50000000[0-0]

本方式により生成した表1の正規表現による状態遷移数および入退室管理装置のログ（約 2100 万件）に対する照合速度は表2のとおりで動作上問題ないことを確認した。

表2 状態遷移数と照合速度

状態遷移数	照合速度[万文字／秒]
80507	10066

5. おわりに

本稿では、ログ検索向け正規表現検索条件の自動生成方式について提案した。また、実際にその方式をプログラムに実装し、複雑な正規表現を自動的に生成できることを確認した。さらに、評価により、その有効性を確認した。

参考文献

[1] 中村 他，大規模ログデータベースの実現，第68回情報処理学会全国大会，1D-2，2006

[2] 中村 他，大規模正規表現の高速照合方式，第67回情報処理学会全国大会，4F-5，2005