7D-1

ガーベッジコレクションを必要としない文字列処理機構

山之上卓　安在弘幸　吉田将
九州大学　九州工業大学

1．はじめに

ガーベッジコレクションを必要としない文字列処理機構を提案する。この機構は、階層構造をもったlong-term memoryと線形のshort-term memory及びそれに付随するアルゴリズムにより構成される。この機構は文字列のパターン認識とデータ圧縮を同時に行う。繰り返しパターンを持った文字列の記憶に必要なlong-term memoryの容量は、文字列の長さをkとすると、O(logk)である。本稿では、long-term memoryや、関数”記憶する”、”思い出す”及び述語”発見”、”クリシエ”、等の形式的定義および、この文字列処理機構の性質を示す。

2．形式的定義

2．1 Long-term memory (Structure)

$S \triangleq \{h_1, h_2, \cdots, h_m\}$

$$h_i = \begin{cases} \{(a,b) \mid a \in \Sigma,\ b \in \Sigma \cup \lambda\} & \text{if } i=1 \\ \{(u,v) \mid u \in PI(h_{i-1}),\ v \in PI(h_{i-1}) \cup \lambda\} & \text{if } i>1 \end{cases}$$

(Σ:Alphabet, λ:an empty string)

Partial Identifications

$PI(h_i)=\{x \in Nn \mid x=pid(h_i,u,v),(u,v) \in h_i\}$

$pid : S \times (Nn \cup \Sigma) \times (Nn \cup \Sigma \cup \lambda) \vdash\!\!\!\rightarrow Nn$

$\forall (x,y),(s,t) \in h_i,$

$(x,y)=(s,t) \equiv pid(h_i,x,y)=pid(h_i,s,t).$

(Nn : A set of Natural numbers)

Identifications

$I(S)=\{(i,j) \mid h_i \in S, j \in PI(h_i)\}$

2．2 記憶する.

Memorize : $\Sigma^+ \times \{S\} \vdash\!\!\!\rightarrow I(S) \times \{S\}$

Memorize$(\omega,S) \triangleq (\delta,S')$

$\delta=(k,j) \in I(S'), k=\min\{i \mid |\alpha_i(\omega,S)|=1\},$

$j=\alpha_k(\omega,S),$

$S'=\{h'_1, h'_2, \cdots, h'_m\}$

$(h'_{k+1}=h_{k+1}, h'_{k+2}=h_{k+2}, \cdots, h'_m=h_m).$

$\alpha : Nn \times \Sigma^+ \times \{S\} \vdash\!\!\!\rightarrow Nn^+$

$\alpha_i(\beta,S) \in PI(h'_i)^+$

$$= \begin{cases} pid(h'_i,a_1,a_2) \cdot pid(h'_i,a_3,a_4) \cdot \cdots \\ \quad \cdots \cdot pid(h'_i,a_{n-1},a_n) & \text{if n=even} \\ pid(h'_i,a_1,a_2) \cdot pid(h'_i,a_3,a_4) \cdot \cdots \\ \quad \cdots \cdot pid(h'_i,a_n,\lambda) & \text{if n=odd} \end{cases}$$

$$h'_i = h_i \cup \begin{cases} \{(a_1,a_2),(a_3,a_4),\cdots,(a_{n-1},a_n)\} & \text{if n=even} \\ \{(a_1,a_2),(a_3,a_4),\cdots,(a_n,\lambda)\} & \text{if n=odd} \end{cases}$$

$$h_i \in S,\ a_1 \cdot a_2 \cdot \cdots \cdot a_n = \begin{cases} \beta & \text{if } i=1 \\ \alpha_{i-1}(\beta,S) & \text{if } i>1 \end{cases}$$

2．3 思い出す.

Remind : $I(S) \times \{S\} \vdash\!\!\!\rightarrow \Sigma^+$

Remind$(Z,S) \triangleq \omega, Z \in I(S), Z \overset{*}{\underset{G(S)}{\Rightarrow}} \omega \in \Sigma^+$

$G(S)=(N(S),T(S),P(S),Z)$

$Z \in I(S),\ N(S)=I(S),$

$T(S)=\{\ a \mid (a,b) \in h_1 \cup (b,a) \in h_1, h_1 \in S\ \}$

$P(S)=\{(i,j) \to (i-1,u) \cdot (i-1,v) \mid$

$j=pid(h_i,u,v),\ (u,v) \in h_i \in S\ \}$

$\cup\ \{(0,a) \to a \mid a \in T(S)\}$

$\cup\ \{(i,\lambda) \to \lambda \mid (x,\lambda) \in h_i \in S\}$

A Garbage Correction Free String Manipulation System

Takashi YAMANOUE+, Hiroyuki ANZAI*, Sho YOSHIDA*
+Kyushu univ.　*Kyusyu Institute of Technology

2．4 Long-term memoryの記憶量

Size : {S}⊢Nn

$$Size(S)\triangleq 2\sum_{i=1}^{n}|hi\in S|$$

2．5 発見

Discovery : $\Sigma^{+}\times\{S\}$⊢boolean

Discovery(ω,S)≜

Memorize(ω,S)=(δ,S')⊃Size(S')>Size(S)

2．6 クリシエ

Cliché : $\Sigma^{+}\times\{S\}$⊢boolean

Cliché(ω,S)≜～Discovery(ω,S)

3 諸性質

○ Remind(Memorize(ω,S))=ω

○ Memorize(ω,S)=(δ,S'), δ=(i,j)

⊃$2^{i-1}\leqq|\omega|\leqq 2^{i}$

○ Memorize(ω^{n},{})=(δ,S), $|\omega|=l$

⊃Size(S)≦2(l+1)($\log_2$(nl)+1)

4 例

○Memorize('abcdefg',{})=(δ,S)

S={h_1,h_2,h_3}

h_1 ={(a,b),(c,d),(e,f),(g,λ)}
pid(h_1,*,*)= 1 , 2 , 3 , 4
h_2 ={(1,2),(3,4)}
pid(h_2,*,*)= 1 , 2
h_3 ={(1,2)}
pid(h_3,*,*)= 1

δ=(3,1)

Size(S)=14

○Memorize('aaaaaaaa',{})=(δ,S)

S={h_1,h_2,h_3}

h_1 ={(a,a)}
pid(h_1,*,*)=1
h_2 ={(1,1)}
pid(h_2,*,*)=1
h_3 ={(1,1)}
pid(h_3,*,*)=1

δ=(3,1)

Size(S)=6

5 おわりに

ガーベッジコレクションを必要としない文字列処理機構の基本的な関数として”記憶する”、”思い出す”などを定義した。今後”連想する”、”考える”、”推定する”、”予測する”などを定義する予定である。また2次元図形認識系への応用や、専用パラレルハードウェアの導入等も検討している。

この文字列処理機構は、言語処理系の生成系MYLANGにインプリメントされており、ハッシュ関数などに用いている。

参考文献

1）河村知行：自動的なパターン抽出によるデータ圧縮法の提案、情報処理学会論文誌、Vol.25,No.6, pp.1089-1094(1984).

2）山之上、安在：属性付構文指示翻訳系の生成系MYLANG、情報処理学会論文誌、Vol.26,No.1, pp.195-204(1985).