

H26 ⑦ アルゴリズムとプログラミング

(1) 43行目: 5 6 10

46行目: 2 5 5 7 13 18 20

(2) front = 3, rear = 5

(3) データ列の中央値と新たに挿入するデータとを比較し、新たに挿入するデータのほうが小さいならデータ列の前半を走査範囲とし、そうでなければデータ列の後半を走査範囲とする。
 挿入データと一致するデータが存在する場合は、探索を終了する。
 以上のような走査範囲の半減を繰り返している。

(4) insert関数: $O(\log n)$ ∵ (3)で説明した通り、15~19行目のループ1周につき対象となるデータ列のサイズ n を半減しているため

delete関数: $O(1)$ ∵ 変数 front で示された番地を参照するだけなので、データ数 n に依存しない。実行に要するステップ数が、いかなる変数にも依存しない。

(5) データ挿入回数が SIZE (= 20) 回に達した時点で、変数 rear の値が SIZE の値と等しくなる。この状態で insert関数が呼び出されると 9 行目の if 文の条件が真となり異常終了となる。
 つまり、データ挿入回数が配列 A の大きさ SIZE より大きくなると挿入に失敗する。

(6) 25 行目で変数 rear をコメントしている部分を以下のように変更する

[変更前] rear ++;

[変更後] rear = (rear + 1) % SIZE;

ただし、% は剰余演算子である。このようにすることで、SIZE を超えるデータ挿入回数であっても変数 rear の値を SIZE 未満に抑えることができる。

同様に変数 front についても 31 行目を以下のように変更する。

[変更前] front ++;

[変更後] front = (front + 1) % SIZE;

ただし、配列 A に挿入されているデータ数が SIZE を超える場合はエラーと判定しなければならぬので、9 行目の if 文の条件を以下のように変更する。

[変更前] if (rear > SIZE - 1)

[変更後] if ((rear + 1) % SIZE == front)

↳ <別解> (こっちの方が better)

変数 front, rear の値は増えはなし。

ただし、front, rear が関与する演算には常に % SIZE を付与する。

配列 A に格納されているデータ数 N を管理しおき、それか SIZE を超えるとエラーを出す。

「データ列」

$m = (\text{left} + \text{right}) / 2$

理由


left と right の位置関係が逆だとおかしい

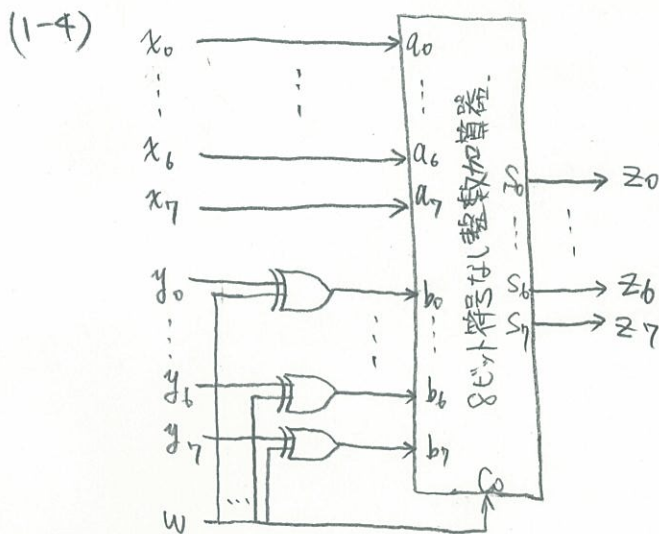
H26 ② 計算機システムとシステムプログラム

(1) (1-1)

i	90	01011010
j	-40	11011000
k	50	00110010
m	-126	10000010
n	126	01111110
d	0.390625	00011001
e	1.125	00110010

(1-2) ~~減算を加算として表現~~ ~~びますが、減算回路が不要になる。~~
 ↳ 減算をビット反転および1を加算で実現できるため。

(1-3) y_i
 w  y'_i (y_i, w, y'_i の対応表を書けば、XOR が良いことを説明できます)



- (2) (2-1) ☐ a ~~unlocked~~ (?) 非専有
☐ b ~~locked~~ (?) 専有
☐ c 割り込み

(2-2) プロセス X がテストを行い、共有資源 Z が ~~unlocked~~ ^{非専有} 状態だったとする。
 テストとセットが不可分操作でない場合、このタイミングでプロセス Y がテストを
 実行することが考えられる。すると、プロセス Y が ~~見ても共有資源 Z が~~
~~unlocked 状態であるので、プロセス X とプロセス Y が同時に共有資源 Z を~~
~~使用し、競合が生じる。~~ このとき、Y が ~~見ても~~ 資源 Z が非専有
 なので、引き続きセットを行うことで、Z は専有状態となる。ここで処理が
 X に戻ると、X は Z が非専有状態であると判断するため、セットコマンドで
 Z は専有状態となる。よって、両プロセスが Z を専有するため、排他制御に失敗する。

1126 ③ 離散構造

- (1) (1-1) R_2 : (i) $\forall x \in V$ について $(x, x) \in R_2$ より反射律が成立.
 (ii) $(x, y) \in R_2$ とする. このとき, y と x を共に含む有向閉路が存在すること自明である. よって対称律が成立.
 (iii) $(x, y) \in R_2 \wedge (y, z) \in R_2$ とする.
 ① x と y を共に含む有向閉路に z が含まれる場合, $(x, z) \in R_2$ は自明
 ② x と y を共に含む有向閉路に z が含まれない場合,
 $x \rightarrow \dots \rightarrow y \rightarrow \dots \rightarrow z \rightarrow \dots \rightarrow y \rightarrow \dots \rightarrow x$ という有向閉路が存在するので
 $(x, z) \in R_2$. (ただし " \rightarrow " は有向辺を表すものと可也)
 ①, ② より $(x, y) \in R_2 \wedge (y, z) \in R_2 \Rightarrow (x, z) \in R_2$ となり推移律が成立.
 (i), (ii), (iii) より R_2 は同値関係である.

- (1-2) R_3 : (i) $\forall x \in V$ について $(x, x) \in R_3$ は自明であるので反射律が成立
 (ii) $\forall x, y \in V$ について $(x, y) \in R_3 \wedge (y, x) \in R_3$ とすると $x=y$ は自明(?) であるので
 反対称律が成立.
 (iii) $(x, y) \in R_3 \wedge (y, z) \in R_3$ とする.
 このとき, S から z へ行く有向経路には y が含まれ, y から x へ行く有向経路には x が含まれる. つまり S から z へ行く有向経路には x と y が含まれるといえる.
 よって $(x, z) \in R_3$ となるので推移律が成立.
 (i), (ii), (iii) より R_3 は半順序関係である.

- (1-3) R_1, R_2, R_3 (結局, $R_2 = \{(x, x) | x \in V\}$ になる)

- (2) (2-1) (a) $r(h, x, y) \rightarrow \forall x \in X \ m(h, x, y)$

(b) $m(h-1, x, y) \rightarrow \forall x \in X \ m(h, x, y)$

(c) $m(h, x, y) \rightarrow (m(h-1, x, y) \vee r(h, x, y))$

- (2-2) 「 A もとで得られる242個の, 両方カントにおいて安定であること」は以下で表された.
 $(CX1 \wedge CX2 \wedge CX3 \wedge CX4 \wedge CX5 \wedge CY1 \wedge CY2 \wedge CY3 \wedge CY4 \wedge CY5 \wedge CZ1 \wedge CZ2) \rightarrow H$
 論理式 P はこの否定である.

$P = CX1 \wedge CX2 \wedge CX3 \wedge CX4 \wedge CX5 \wedge CY1 \wedge CY2 \wedge CY3 \wedge CY4 \wedge CY5 \wedge CZ1 \wedge CZ2 \wedge \neg H$

- (2-3) (2-3-1) (d) $\neg r(1, u1, u2)$

(e) $\neg r(1, u2, u2)$

(f) $\neg r(1, u1, u1) \vee m(1, u1, u1) \vee m(1, u2, u1)$

(g) $\neg m(1, u1, u2) \vee r(1, u1, u2)$

(h) $\neg m(1, u2, u2) \vee r(1, u2, u2)$

- (2-3-2) $B = \neg m(1, u1, u1) \wedge \neg m(1, u1, u2) \wedge m(1, u2, u1) \wedge \neg m(1, u2, u2)$ と可.

「 $A1 \wedge A2 \wedge A3 \wedge A4 \wedge A5 \wedge A6 \wedge A7 \wedge A8 \wedge \neg B$ 」が充足不能であることを示す.

$A1 \wedge A3$ より $A9$: $r(1, u1, u1)$ が導かれる.

$A2 \wedge A4$ より $A10$: $r(1, u2, u1)$ が導かれる.

$A5 \wedge A9 \wedge A10$ より $A11$: $\neg m(1, u1, u1)$ が導かれる.

$A6 \wedge A9 \wedge A11$ より $A12$: $m(1, u2, u1)$ が導かれる.

$A3 \wedge A7$ より $A13$: $\neg m(1, u1, u2)$ が導かれる.

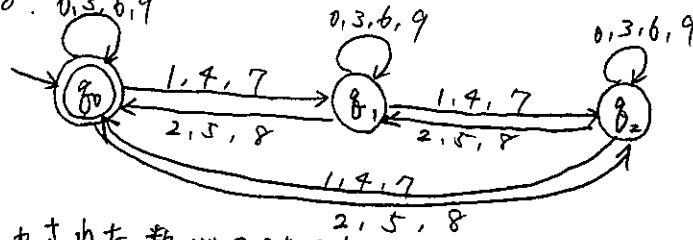
$A4 \wedge A8$ より $A14$: $\neg m(1, u2, u2)$ が導かれる.

$A11 \wedge A12 \wedge A13 \wedge A14 \wedge \neg B$ より 定数 m が導かれるので 題意が示された.

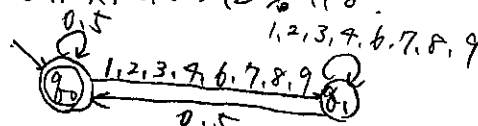
H26 ④ 計算理論

(1) (1-1) 現在入力されている語を3で割った余りが 0, 1, 2 のいずれかで 3 状態に分ける。

現在まわに入力された数を x , 次に入力される数を y とすると、次は $10x+y$ を3で割った余りを考えればよい。ここで「 $x \bmod 3 = 10x \bmod 3$ 」という関係を踏まえてオートマトン B は以下のように構築できる。



(1-2) 最後に入力された数から 0 または 5 のとき、5 で割り切れると判断できるので、オートマトン C は以下のように書ける。



$B = (Q_B, \Sigma, \delta_B, q_0^B, F_B)$, $C = (Q_C, \Sigma, \delta_C, q_0^C, F_C)$ とする。

B が C で受理されるとき、その入力はいずれも割り切れるので、

$D = (Q_B \times Q_C, \Sigma, \delta_D, (q_0^B, q_0^C), F_D)$

ただし、 $\delta_D((q_i^B, q_j^C), a) = (\delta_B(q_i^B, a), \delta_C(q_j^C, a))$ ただし $a \in \Sigma$

また、 $F_D = \{(q_i^B, q_j^C) \mid q_i^B \in F_B \wedge q_j^C \in F_C\}$

(1-3) 受理状態 F_D を以下の F_D' に変更する。

$F_D' = \{(q_i^B, q_j^C) \mid q_i^B \in F_B \vee q_j^C \in F_C\}$

(2) (2-1) $G_1 = (N_1, T_1, P_1, S_1)$ は以下のように書ける。

$N_1 = \{A, B\}$

$T_1 = \{c\}$

$P_1 = \{A \rightarrow cB, B \rightarrow cB, B \rightarrow \epsilon\}$

$S_1 = A$

(2-2) まず、言語 $L' = \{a^n b^n \mid n \geq 1\}$ を生成する文脈自由文法 G' を示し、 L' が文脈自由言語であることを示す。

$G' = (N', T', P', S')$ は以下のように書ける

$N' = \{A, B\}$

$T' = \{a, b\}$

$P' = \{A' \rightarrow aB'b, B' \rightarrow aB'b, B' \rightarrow \epsilon\}$

$S' = A$

L_1 と L' を用いると、 $L_2 = L' \cdot L_1$, $L_3 = L_1 \cdot L'$ のように L_2 および L_3 は文脈自由言語の連接演算で書ける。補題 1 より L_2 および L_3 は文脈自由言語である。

(2-3) 積演算 \cap に関心していると仮定すると、 $L_2 \cap L_3$ は文脈自由言語になる。

しかし、 $L_2 \cap L_3 = L_4$ であるが、補題 2 より L_4 は文脈自由言語ではないので矛盾する。よって文脈自由言語全体の集合は積演算 \cap について閉じていない。

5

(1)

(1-1) 出力が過去の情報(出力)に依存しない情報源.

(マルコフ情報源は過去の出力に影響を受ける情報源)

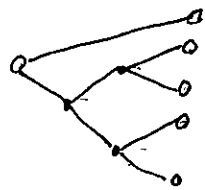
1-2) (a) g (i) c (j) e (k) b

1-3)

1-3-1) $p_3 + p_4 > p_1$

1-3-2) $p_4 + p_5 > p_2$
 $p_1 > p_2 + p_3$) ... (x)

証明) 題意を満たす復合木の形は以下の通りのみ.



(x) が必要十分条件であることを示すため.

(i) (x) の条件式 \rightarrow 復合木

(ii) 復合木 \rightarrow (x) の条件式 を示す.

(i) $p_1 > p_2 > \dots > p_5$ よりはじめに生起確率が p_4 と p_5 の記号が縮約される
 ここで $p_4 + p_5 > p_2$ より、この時点で生起確率が最も小さい p_2, p_3 に対応する記号が縮約される.

$p_1 > p_2 + p_3$ より、この時点での生起確率は $p_1 > p_2 + p_3 > p_4 + p_5$ となり
 $(p_2 + p_3)$ と $(p_4 + p_5)$ に対応する記号が縮約されるため、復合木の形になる.

(ii) $p_1 > p_2 > \dots > p_5$ より p_4 と p_5 に対応する記号が縮約されることは自明.
 次にこれが、縮約した記号と縮約されているため.

p_4, p_5 縮約後の生起確率は少なくとも

$p_4 + p_5 > p_2 > p_3$ であり、 p_2 と p_3 縮約後では
 この時点で縮約されていない記号は p_1 のみであり、復号木より

$p_1 > p_2 + p_3 > p_4 + p_5$ となる.

よって復号木の形なるば (x) の条件式となる

以上 (i) (ii) より

(x) は必要十分条件

平成26年度 院試 5 (選択問題) ネット7-7

(1) 飛びおぼえみ...。ごめんすい.....

(2)

(2-1) (a) e (ii) d (3) + (元) d (お) e

(2-2) (元) 方式の場合、ネット7-7負荷、増加に伴い、フレーム衝突する確率が上昇するため、結果として遅延が増加する。一方(お)方式の場合、フレーム衝突は一切発生しない。(元)方式と異なり、ト-7-7巡回時間、有線であるため、フレーム再送 > ト-7-7巡回時間 のため、(お)方式の方が遅延は急激に増加しない。

② 伝搬遅延

$$\begin{cases} \text{CSMA} = \text{フレーム送信時間} + \text{再送時間} \times \text{平均} \\ \text{Token} = \text{フレーム送信時間} + \text{1177 遅延} \end{cases}$$

基本時に、ある1つのポートに着目してその評価を行えば良い

フレーム送信後に、再びTokenを受信するまでの時間

(2-3) ランダムに待ち時間、平均値を2倍にする。
 行うことは、お/広の範囲から待ち時間をランダムに選ぶため、衝突がおきる確率を1/2にするためである。

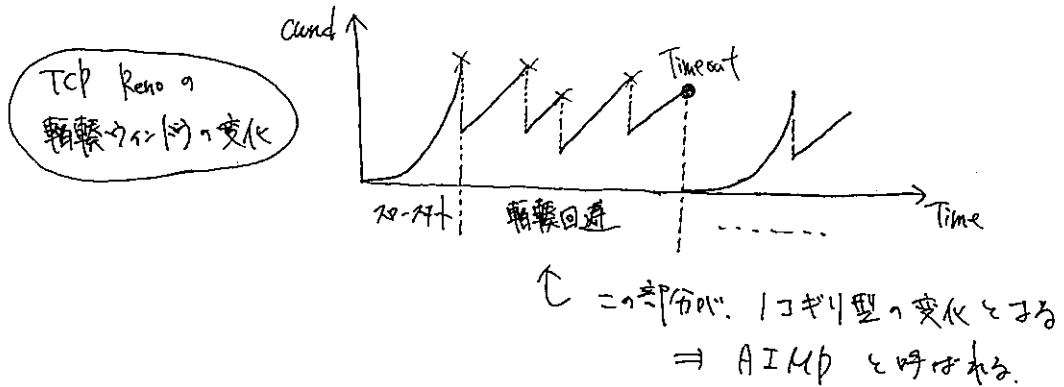
* 実際は、衝突回数 N_c に対して、

$$\begin{cases} \text{if } N_c < 10 & \text{then backoff} = \text{random}([0..2^{N_c}]) \\ \text{else if } N_c < 16 & \text{then backoff} = \text{random}([0..2^{16}]) \\ \text{else} & \text{discarding frame} \end{cases}$$

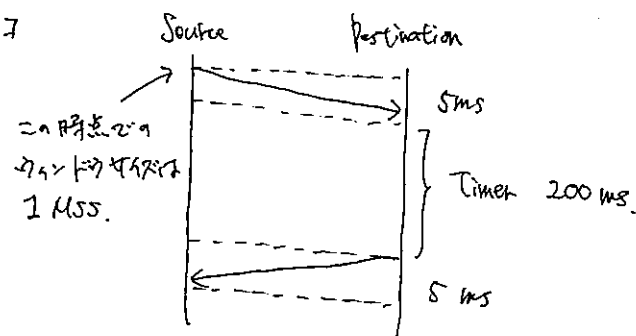
(16回目以降、衝突はあきらめる)

(3)

(3-1) (a) c (u) h (s) e (i) a (o) k



(3-2) 210 [ms]



(3-3) 240 [ms]

