順序回路自動設計システムにおける状態割当てについて

STATE ASSIGNMENT APPLIED TO THE AUTOMATIC DESIGN SYSTEM FOR SEQUENTIAL MACHINES

吉川 良一　　相磯 秀夫　　所 眞理雄
Ryoichi Yoshikawa　　Hideo Aiso　　Mario Tokoro
慶應義塾大学
Keio Univ. Depertment of Engineering

This paper describes a method to solve the state assignment problem of asynchronous sequential machines, which is applied to the Automatic Design System for Sequential Machines now developped by the authors. The method is based on the idea proposed by Saucier[1)], who has formulated the problem in graph theoretic terms of embedding graph into n-cube, where the weight associated with edge is used to decrease the transition time. However, the method introduces a new algorithm associated with the determination of weight, which employs transition probability via edge. Since the algorithm gives an adequate weight, the method improves the quality of design.

1、緒言

非同期式順序回路の状態割当てをするアルゴリズムの一つがSaucier[1)]により示されている。それは、状態をvertex、1 bit の状態遷移をedgeに対応させ、グラフとして扱うことによりtotally sequentialな状態割当てを実現している。我々が開発中の順序回路自動設計システムは、状態割当てにこのアルゴリズムを採用している。

グラフとして考えると、状態割当ては、2進数に対応したn-cube 内のvertexに各状態の対応づけをすることである。特にtotally sequential な場合は、遷移が常に1 bit ずつであるので、状態の遷移はグラフ上のedgeをたどりながら行なわれると考えられる。そこでSaucierは、この状態割当てを、重みのついたedgeをn-cubeに埋め込む問題として扱い、これをもとに状態遷移の道を決定している。

本論文は、このアルゴリズムを実現する上でのいくつかの問題点を指摘している。その主なものは

i) Saucierにより示された重みのつけかたでは、実際的でない場合が存在する。
ii) 重みのついたグラフをn-cubeに埋め込むアルゴリズムの最適化に関する考察がない。

である。ここではSaucierにより示された重みのつけかたを改善するために、edgeを遷移する確率から重みを与えることを提案し、その大きさを求めるアルゴリズムを示し、その正当性を論じる。

2、Saucierのアルゴリズム

2.1 グラフで考えた場合のcritical race free

各状態をvertex、1 bit の遷移をedgeで表わし、グラフとして考える。この場合、状態遷移はグラフ上のedgeをたどりながら行なわれると考えられる。ここで、Saucierと同様の定義を導入すると、

$s_i \in S$ ：状態

$I_j \in I$ ：入力

$\delta(s_i, I_j)$ ：入力 I_j による遷移で、s_i から到達する安定状態を示す関数（ $\delta(s_i, I_j) = s_i$ のとき、s_i は安定状態）

S_{I_j} ：$\delta(s_i, I_j)$ の定義されている S の部分集合 $S_{I_j} \subseteq S$

$\pi_i = (B_{i1}, \cdots, B_{ij}, \cdots)$：入力 I_i による状態の分割
ここで分割された各ブロック：B_{ij} は次の条件をみたすような状態の集合である
$s_1, s_2 \in B_{ij} \iff \delta(s_1, I_i) = \delta(s_2, I_i)$
$B_{ij} \subseteq S_{I_i}$

$I(B_{ij})$ ：B_{ij} に対応した n-cube 内の vertex

G_{ij} ：B_{ij} に対応する n-cube の subgraph

このとき、次の条件をみたすとき、critical race のない（totally sequential な）状態割当てが実現できる（状態遷移の道を指定することができる）。

i) $I(B_{ij}) \subseteq \{\text{vertices of } G_{ij}\}$

ii) G_{ij} ： connected graph

iii) $G_{ij} \cap G_{ik} = \phi \iff j \neq k$

ここで、

$X_{ij} = \{\text{vertices of } G_{ij}\} - I(B_{ij})$

X_{ij} ：補足のvertex

$\delta(s_i, I_j)$ の定義されていない状態に対応したvertexが、X_{ij} として使われる。

2.2 状態割当てのアルゴリズム

Saucierは、次のような2段階からなるアルゴリズムによって状態割当てを実現している。

(1) 各状態を最小の n-cube 内に遷移時間が短かくなるように配置する。
ここでは、edgeに遷移時間が短かくなるような重みをつけ、重みの大きなグラフを最小の n-cube 内に実現することでこれを行なっている。

(2) ブロック内の各状態から安定状態まで状態が遷移する道を指定する。
ここで行なうことは、条件をみたすようなsubgraph：G_{ij} を決定することに他ならないが、ここではブロックごとに可能な道をすべて調べてこれを行なっている。このとき、条件は

i), ii) ブロックに属するすべてのvertexは道でつながれている。

iii) ブロック内の状態遷移として指定される道は、他のブロックの道と交わってはいけない。

という意味になる。

2.3 問題点

(1) 重みのつけかた

Saucierは、$B_{ij} = (x_0, x_1, \cdots, x_n)$ があるとき、edge(x_a, x_b), $0 \leq a \leq n, 0 \leq b \leq n, a \neq b$ を

考え、次のような重みをつけている。($edge(x_a,x_b) \equiv edge(x_b,x_a)$ は、$edge(I(x_a), I(x_b))$ を表わしている。) ここで、B_{ij} の n 個組の左端に書かれるものが安定状態であると約束する。

$$f_i(x_a,x_b) = \begin{cases} w_1 \cdots B_{ij}=(x_a,x_b) & \text{のとき} \\ w_2 \cdots B_{ij}=(x_a,x_b,x_c,\cdots) & \text{のとき} \\ w_3 \cdots B_{ij}=(x_z,x_a,x_b,\cdots) & \text{のとき} \end{cases}$$

ただし、$w_1 > w_2 > w_3$ (定数)

このとき、$edge(x_a,x_b)$ の重みは各入力について総和をとり

$$F_w(x_a,x_b) = \sum_{I_i} f_i(x_a,x_b) \qquad \text{———} \quad (1)$$

として計算される。

しかしながら、上で示したような重みのつけかたでは、w_2, w_3 が B_{ij} の要素の数にかかわりなく一定であるために、実際的でない場合が生じている。Saucier は例として、$w_1=3, w_2=2, w_3=1$ という値を示しているが、

$$B_{ij}=(x_a,x_b)$$
$$\begin{cases} B_{kp}=(x_0,x_1,x_2,\cdots,x_{n_{kp}}) \\ B_{lp}=(x_0,x_1,x_2,\cdots,x_{n_{lp}}) \\ B_{mp}=(x_0,x_1,x_2,\cdots,x_{n_{mp}}) \end{cases}$$

があるとき、$F_w(x_a,x_b) = F_w(x_1,x_2)$ ということになる。しかし、$n_{kp}, n_{lp}, n_{mp} >> 1$ のとき $edge(x_1,x_2)$ を実現することは、$edge(x_a,x_b)$ に比べてほとんど意味がないことであり、ある $n_{kp}, n_{lp}, \cdots$ に対して w_3 の大きさをどの程度にするのが良いか問題となる。また同様に

$$B_{ij}=(x_a,x_b)$$
$$\begin{cases} B_{kp}=(x_1,x_2,\cdots x_{n_{kp}}) \\ B_{lp}=(x_1,x_2,\cdots x_{n_{lp}}) \end{cases}$$

があるとき、$F_w(x_a,x_b) < F_w(x_1,x_2)$ となる。しかし、$n_{kp}, n_{lp} >> 1$ のとき $edge(x_1,x_2)$ を実現するということの意味は薄くなり、$F_w(x_a,x_b) < F_w(x_1,x_2)$ は実際的ではない。やはりここでも、ある $n_{kp}, n_{lp}, \cdots$ に対して、w_1 を基準とした w_2 の大きさがどうあるべきか問題となる。しかし、Saucier は w_1, w_2, w_3 の値については例を示したに留まり、具体的にどのような値を与えるべきかということには触れていない。

以上のようなことから、ここでは w_2, w_3 を B_{ij} の要素の数：n_{ij} の関数とし、w_1, w_2, w_3 の値を n_{ij} の関数として決定するためのアルゴリズムについて以下の章で考察を行なう。

(2) 最大の重みをもつグラフを実現するアルゴリズム

このアルゴリズムは、edge の集まりとして構成されるグラフの中から、決められた最小の n-cube に埋め込むことができ、かつ、最大の重みをもつグラフを選びだすアルゴリズムのことである。しかし、構成されるすべてのグラフについて重みを計算し、すべての埋め込みの可能性を調べるようなアルゴリズムは、処理時間の点で問題がある。最大の重みをもつという条件はそれほど厳密なものではなく、計算時間とのかねあいで適当なところで妥協することになるが、これを行なうためには、そのアルゴリズムが最終的にどの程度良い結果を与えるかを知る必要がある。しかし、それを知るためには、実際にシステムに組み入れて、多くの

データーについて結果を調べる必要がある。
現在は、処理時間がある程度増すことになるが、Saucierのアルゴリズムに判断の数を増し、少し複雑にすることを考えている。この点については、現在考察中でありいずれ発表する予定である。

3、重みのつけかた

3.1　重みの意味

重みをつけるのは、遷移時間が短かくなるように各状態をn-cubeのvertexに割当てるためである。その意味から、重みに対する必要条件としては次のようなことが考えられる。

i) 重みの大きなedgeをdirect transitionで実現した場合、全体としての遷移時間が短かくなる。

ii) 上記の場合、重みの大きさは線形（グラフの重みの総和の順序関係と、全体としての遷移時間の短かさの順序関係が等しい）か、それに近いこと。

また、重みを定式化する考えかたとしては、次のようなものがある。

i) そのedgeをdirect transitionで実現した場合に、全体の遷移時間に与える影響を数値で表わす。

ii) 状態割当てを実現した場合に、そのedgeを遷移する確率の大きさを重みとする。

iii) その他、力学的なモデルによる方法。

この中で、確率として考えた場合に最も良い結果が得られたので、以下に順を追って述べることにする。

3.2　edge(X_a, X_b)を遷移する確率

はじめに、状態遷移は一様であり、各内部状態をとる確率は一定であり、同様に各入力の生起確率も一定であると仮定する。

重みを、edgeを遷移する確率と考える場合に、edge：$E=\{e_1, e_2, \cdots, e_m\}$についてそれぞれのedgeを遷移する確率を$P(e_1), P(e_2), \cdots, P(e_m)$とすれば、全体としての平均遷移時間は、

$$T = \Delta t \cdot \sum_{i=1}^{m} P(e_i) \qquad \text{———} \quad (2)$$

Δt：1遷移時間

として表わされる。すなわち、edgeを遷移する確率は、前節で考えた重みとしての必要条件i), ii)を間接的にみたしている。

(1)　$B_{ij}=(X_a, X_b, \cdots, X_{n_{ij}})$であるとき、入力$I_i$による遷移でedge$(X_a, X_b)$を通る確率は、

$$P_0(n_{ij}) = \frac{1+\frac{1}{N_0}(n_{ij}-1)}{n_{ij}} \qquad \text{———} \quad (3)$$

となる。ただし、N_0は状態変数の数である。

(2)　$B_{ij}=(X_0, X_a, X_b, \cdots, X_{n_{ij}})$　であるとき、

$f_A(n_{ij})$　：　$I(X_0)$と$I(X_i)$, $1 \leqq i \leqq n$　の平均距離（ハミングの距離）。

r　：ハミングの距離。

$f_N(r)$　：　N_0-cube　のvertexが、ある基準のvertexからどのような距離に分布するかを示す関数。

$f_x(r)$ ： $I(x_i) \in I(B_{ij})$, $0 \leqq i \leqq n$ が $I(x_0)$からどのような距離に分布するかを示す関数。

とすると、$I(x_a)$が$I(x_0)$から平均距離：$\bar{r} = f_x(n_{ij})$ にあると考え、$I(x_a)$より遠くにある$I(x_i) \in I(B_{ij})$の数は

$$\int_{\bar{r}}^{N_0} f_x(r)\,dr \qquad (4)$$

となる。さらに、そのうちで $1/f_n(\bar{r})$ がx_aを通る（$x_i \in B_{ij}$からx_0への遷移で通る）と考えれば、すべての$x_i \in B_{ij}$について1回ずつ、x_0に向かって遷移が起こるときにx_aを通る数は、

$$\frac{1}{f_n(\bar{r})} \cdot \int_{\bar{r}}^{N_0} f_x(r)\,dr \qquad (5)$$

となる。

次に、$I(x_a)$から後につながっているedgeの数は $N_0 - \bar{r}$ で、そのうち$I(x_a)$から$I(x_i) \in I(B_{ij})$につながっているものは $f_x(\bar{r})/f_n(\bar{r})$ である。そこで、$I(x_a)$から1 distance 後の$I(x_i) \in I(B_{ij})$につながっているedgeの数は

$$\frac{f_x(\bar{r})}{f_n(\bar{r})}(N_0 - \bar{r})$$

となり、言い換えるとx_aを通るもののうち

$$\frac{1}{N_0 - \bar{r}} \cdot \frac{f_n(\bar{r})}{f_x(\bar{r})} \qquad (6)$$

がx_bを通ることになる。そこで、x_a, x_bを通る数は

$$\frac{1}{N_0 - \bar{r}} \cdot \frac{f_n(\bar{r})}{f_x(\bar{r})} \cdot \frac{1}{f_n(\bar{r})} \cdot \int_{\bar{r}}^{N_0} f_x(r)\,dr$$

$$= \frac{1}{(N_0 - \bar{r}) \cdot f_x(\bar{r})} \cdot \int_{\bar{r}}^{N_0} f_x(r)\,dr \qquad (7)$$

ということになる。

上式から、x_a, x_bを通る確率は

$$P_1(n_{ij}) = \frac{1}{n_{ij}(N_0 - \bar{r}) \cdot f_x(\bar{r})} \cdot \int_{\bar{r}}^{N_0} f_x(r)\,dr \qquad (8)$$

となる。

さらに、入力I_iに対してedge(x_a, x_b)を通る確率を

$$P_i(x_a, x_b) = \frac{\sum_{B_{ij} \subseteq \pi_i} n_{ij} \cdot P(n_{ij})}{\sum_{B_{ij} \subseteq \pi_i} n_{ij}} \qquad (9)$$

ただし、$B_{ij} = (x_0, x_1, \cdots, x_{n_{ij}})$ において

$$P(n_{ij}) = \begin{cases} P_0(n_{ij}) & \cdots\ x_a\text{または}x_b\text{が安定状態のとき} \\ P_1(n_{ij}) & \cdots\ x_a, x_b\text{が不安定状態のとき} \\ 0 & \cdots\ x_a, x_b\text{が同時に含まれる}B_{ij}\text{がないとき} \end{cases}$$

として表わすと、すべての入力に対してedge(x_a, x_b)を通る（遷移する）平均の確率は

$$P(x_a, x_b) = \frac{\sum_{i=1}^{m} P_i(x_a, x_b)}{m} \qquad (10)$$

となる。ただし、mは入力の数である。

3.3　$B_{ij}=(x_0, x_1, \cdots, x_{n_{ij}})$　における平均距離：$f_t(n_{ij})$

(1)　$f_n(r)$　：　N_0-cube　の vertex の分布関数

確率密度関数としての２項分布を正規分布で近似する。すなわち、２項分布

$$b(x)=\binom{n}{x}\cdot\theta^{x}(1-\theta)^{n-x} \qquad (11)$$

について、確率変数：x の分布は十分に大きい n に対して近似的に正規分布 $N(n\theta,\ n\theta(1-\theta))$　に等しくなり、この方法を用いて２項分布の確率の和を、正規分布の適当な積分で計算できる。

２項分布を正規分布で近似すると

$N(n\theta,\ n\theta(1-\theta))$ ：

$$f(x)=\frac{1}{\sqrt{n\theta(1-\theta)}\sqrt{2\pi}}\,e^{-\frac{1}{2n\theta(1-\theta)}(x-n\theta)^2} \qquad (12)$$

となり、N_0-cube　の場合を考えると、$\theta=\frac{1}{2}$，$x=r$，$n=N_0$　であるので

$$f(r)=\sqrt{\frac{2}{N_0\pi}}\cdot e^{-\frac{2}{N_0}(r-\frac{N_0}{2})^2} \qquad (13)$$

となる。

上記 $f(r)$ は、N_0-cube　の vertex の確率密度関数を表わす。そこで、vertex の分布は、

$$\begin{aligned} f_n(r) &= 2^{N_0}\cdot f(r) \\ &= 2^{N_0}\sqrt{\frac{2}{N_0\pi}}\,e^{-\frac{2}{N_0}(r-\frac{N_0}{2})^2} \end{aligned} \qquad (14)$$

となる。

(2)　$f_x(r)$：$I(B_{ij})$　の分布関数

$B_{ij}=(x_0, x_1, \cdots, x_{n_{ij}})$　に属する vertex は、安定状態に対応した $I(x_0)$　の付近に多く分布することを予想して減衰関数を考える。

ここでは、減衰関数を e^{-Kr^2} と仮定する。そうすると vertex：$I(B_{ij})$ の分布は

$$\begin{aligned} f_x(r) &= e^{-Kr^2}\cdot 2^{N_0}\cdot\sqrt{\frac{2}{N_0\pi}}\cdot e^{-\frac{2}{N_0}(r-\frac{N_0}{2})^2} \\ &= 2^{N_0}\cdot\sqrt{\frac{2}{N_0\pi}}\cdot e^{-(K+\frac{2}{N_0})r^2+2r-\frac{N_0}{2}} \end{aligned} \qquad (15)$$

ここで、$K+\frac{2}{N_0}=a$ とおくと

$$\begin{aligned} f_x(r) &= 2^{N_0}\cdot\sqrt{\frac{2}{N_0\pi}}\cdot e^{-ar^2+2r-\frac{N_0}{2}} \\ &= 2^{N_0}\cdot\sqrt{\frac{2}{N_0\pi}}\cdot e^{\frac{1}{a}-\frac{N_0}{2}}\cdot e^{-a(r-\frac{1}{a})^2} \end{aligned} \qquad (16)$$

となる。また、このときの定数 K の値は

$$\int_{-\infty}^{\infty} f_x(r)\,dr = n_{ij}+1 \qquad (17)$$

から得られる。ここで、積分の範囲は本来 $0\sim N_0$ であるが、２項分布を正規分布で近似したときに

$$\int_{0}^{N_0} f_x(r)\,dr \simeq \int_{-\infty}^{\infty} f_x(r)\,dr \qquad (18)$$

という仮定が含まれているので、$-\infty\sim+\infty$ へ拡張した。その結果、積分公式

$$\int_0^\infty e^{-A^2x^2}dx = \frac{\sqrt{\pi}}{2A} \qquad \text{———} \quad (19)$$

が使える。

$$\int_{-\infty}^{\infty} f_x(r)\,dr = 2^{N_0}\cdot\sqrt{\frac{2}{N_0\pi}}\cdot e^{\frac{1}{\alpha}-\frac{N_0}{2}}\cdot\int_{-\infty}^{\infty} e^{-\alpha(r-\frac{1}{\alpha})^2}dr \qquad \text{———} \quad (20)$$

公式から

$$\int_{-\infty}^{\infty} e^{-\alpha(r-\frac{1}{\alpha})^2}dr = \sqrt{\frac{\pi}{\alpha}} \quad (\alpha = K+\tfrac{2}{N_0}>0) \qquad \text{———} \quad (21)$$

$$\therefore \int_{-\infty}^{\infty} f_x(r)\,dr = 2^{N_0}\cdot\sqrt{\frac{2}{N_0\alpha}}\cdot e^{\frac{1}{\alpha}-\frac{N_0}{2}} \qquad \text{———} \quad (22)$$
$$= n_{ij}+1$$

式を変形すると

$$\frac{1}{\alpha}-\frac{1}{2}\log\alpha = \frac{N_0}{2}+\log\frac{n_{ij}+1}{2^{N_0}}\cdot\sqrt{\frac{N_0}{2}} \qquad \text{———} \quad (23)$$

実際に数値を求めるためには、上式を利用し、逐次近似法を用いて $\alpha = K+\frac{2}{N_0}$ を求める。

(3) $f_t(n_{ij})$ ：平均距離

$$\bar{r} = f_t(n_{ij}) = \frac{1}{n_{ij}}\cdot\int_{-\infty}^{\infty} r\cdot f_x(r)\,dr$$
$$= \frac{2^{N_0}}{n_{ij}}\cdot\sqrt{\frac{2}{N_0\pi}}\cdot e^{\frac{1}{\alpha}-\frac{N_0}{2}}\cdot\int_{-\infty}^{\infty} r\cdot e^{-\alpha(r-\frac{1}{\alpha})^2}dr \qquad \text{———} \quad (24)$$

ここで、

$$\int_{-\infty}^{\infty} r\cdot e^{-\alpha(r-\frac{1}{\alpha})^2}dr$$
$$= \int_{-\infty}^{\infty}\left(t+\frac{1}{\alpha}\right)e^{-\alpha t^2}dt \qquad \left(t = r-\frac{1}{\alpha}\right)$$
$$= \frac{1}{\alpha}\int_{-\infty}^{\infty} e^{-\alpha t^2} = \sqrt{\frac{\pi}{\alpha^3}} \qquad \text{———} \quad (25)$$

であるから、平均距離は

$$f_t(n_{ij}) = \frac{2^{N_0}}{n_{ij}}\cdot\sqrt{\frac{2}{N_0\alpha^3}}\cdot e^{\frac{1}{\alpha}-\frac{N_0}{2}} \qquad \text{———} \quad (26)$$

$f_t(n_{ij})$ の値は、図1のようになる。

3.4 edge(x_a, x_b) を通る確率： $P_1(n_{ij})$

3.2節において、edge(x_a, x_b) を通る確率：$P_1(n_{ij})$ は次式のように与えられた。

$$P_1(n_{ij}) = \frac{1}{n_{ij}(N_0-\bar{r})\cdot f_x(\bar{r})}\cdot\int_{\bar{r}}^{N_0} f_x(r)\,dr \qquad \text{———} \quad (8)$$

上式において

$$\int_{\bar{r}}^{N_0} f_x(r)\,dr \simeq \frac{n_{ij}}{2} \qquad \text{———} \quad (27)$$

と考えると、(16)式を代入して

$$P_1(n_{ij}) = \frac{1}{n_{ij}(N_0-\bar{r})\cdot 2^{N_0}\sqrt{\frac{2}{N_0\pi}}\cdot e^{\frac{1}{\alpha}-\frac{N_0}{2}}\cdot e^{-\alpha(\bar{r}-\frac{1}{\alpha})^2}}\cdot\frac{n_{ij}}{2}$$

$$= \frac{1}{2^{N_0+1}\cdot(N_0-\bar{r})}\cdot\sqrt{\frac{N_0\pi}{2}}\cdot e^{-\frac{1}{a}+\frac{N_0}{2}}\cdot e^{a(\bar{r}-\frac{1}{a})^2} \qquad \text{———} \quad (28)$$

となる。

結果として、$B_{ij}=(x_0,x_1,\cdots,x_{n_{ij}})$ において $edge(x_a,x_b)$ を通る確率は

i) x_a または x_b が安定状態のとき

$$P_0(n_{ij}) = \frac{1+(n_{ij}-1)/N_0}{n_{ij}} \qquad \text{———} \quad (3)$$

ii) x_a,x_b が不安定状態のとき

$$P_1(n_{ij}) = \frac{1}{2^{N_0+1}\cdot(N_0-\bar{r})}\cdot\sqrt{\frac{N_0\pi}{2}}\cdot e^{-\frac{1}{a}+\frac{N_0}{2}}\cdot e^{a(\bar{r}-\frac{1}{a})^2} \qquad \text{———} \quad (28)$$

ここで

$$\bar{r} = f_*(n_{ij}) = \frac{2^{N_0}}{n_{ij}}\cdot\sqrt{\frac{2}{N_0 a^3}}\cdot e^{\frac{1}{a}-\frac{N_0}{2}} \qquad \text{———} \quad (26)$$

また、$a=K+\frac{2}{N_0}$ の値は次式から定められる。

$$\frac{1}{a}-\frac{1}{2}\log a = \frac{N_0}{2}+\log\frac{n_{ij}+1}{2^{N_0}}\cdot\sqrt{\frac{N_0}{2}} \qquad \text{———} \quad (23)$$

$P_0(n_{ij}), P_1(n_{ij})$ の値は、図2のようになる。

3.5 数値計算の結果

図1,図2の値は、N_0 をパラメータとして BASIC を用いて計算を行なった。ここで、a の値は(23)式を利用し、ニュートン法を用いて求めた。

Saucierにより示された重みは、グラフにおける各点の平均として、次式のように表わすことができる。

$$w_1 = P_0(1) \qquad \text{———} \quad (29)$$

$$w_2 = \frac{\sum_{i=2}^{n} P_0(i)}{n-2} \qquad \text{———} \quad (30)$$

$$w_3 = \frac{\sum_{i=2}^{n} P_1(i)}{n-2} \qquad \text{———} \quad (31)$$

(ここで、n は平均する n_{ij} の上限であるが、これは現われる n_{ij} の範囲で適当に選ぶ。)

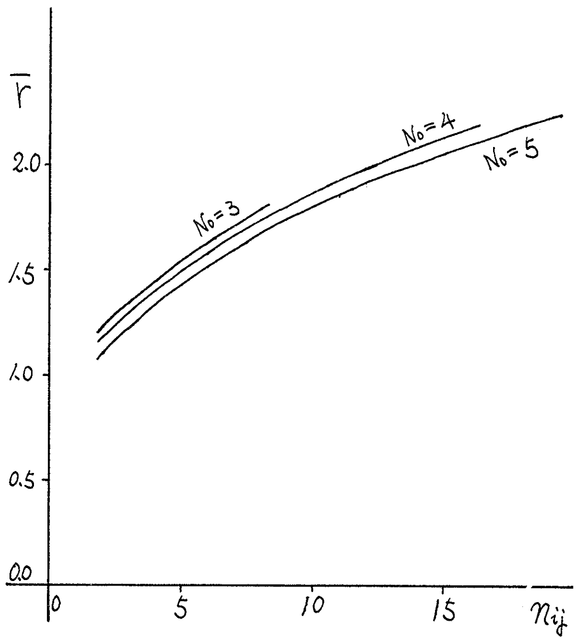

図1 平均距離：$\bar{r} = f_*(n_{ij})$

図2を見ると、w_3 として平均される各点は B_{ij} の要素の数：n_{ij} に対して大きな変化は見られない。しかし、w_2 として平均される各点は、その値が大きく変化していて、平均値と実際の値との差がかなり大きくなっている。このことから、w_3 に関しては2.3節で問題点として取り上げた、n_{ij} による変化というのは特に問題にする必要がないことがわかるが、特に w_2 に関して、Saucier による重みのつけかたは、大きく変化する重みを、平均的な一つの定数で表わそうとしていることが明らかとなった。そこで、w_2 に関しては n_{ij} の大きさでいくつかの

場合に分けるなど、重みのつけかた P_0, P_1 を改善することが考えられる。

また、W_1, W_2, W_3 について値を求めるためには、W_2 をいくつかの場合に分けるとしても、上で示したように適当な n を定めることにより、数値計算で各々の値を求めることができる。

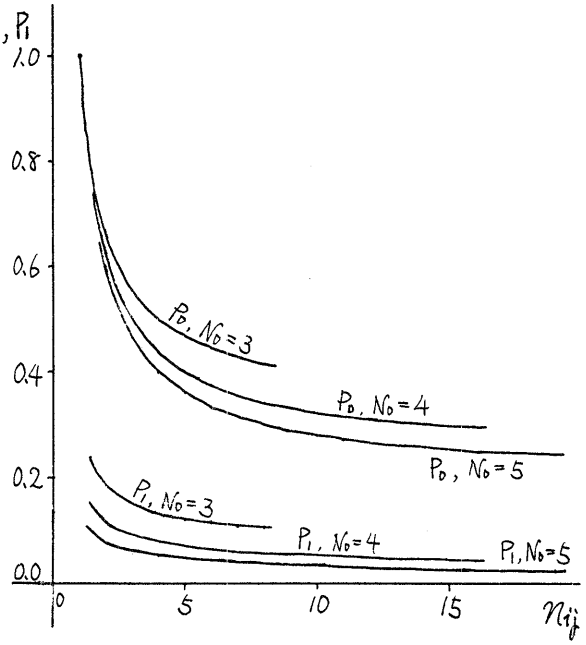

図２　遷移確率：$P_0(n_{ij}), P_1(n_{ij})$

4、設計システムの実現について

図２で表わされた重みを、システムとして実現するためには、かなり簡単にする必要がある。ここでは、$N_0 \simeq 4 \sim 5$ であることを予想して、$P_0, N_0=4$ と $P_1, N_0=4$ で全体を代表させ、さらに各々を量子化し、各 n_{ij} の点で標本化することにする。量子化の幅は、0.05〜0.1 の間で、重みの意味が十分に発揮でき、かつ、処理が容易になるものを適当に選ぶことにする。このことにより、Saucierが三つの場合に分けて重みを表わしたのに対して、ここでは、場合として分けられる数が7〜9となる。

また、実際に使用するために、次のような機能を付加している。

i)　direct edgeを指定する。

特にdirect transition で実現したい状態遷移を指定する。その結果、一部だけdirect transitionが必要であるような順序機械を実現する場合、この状態割当てを行なうと、実際問題としてSingle Transition Time Assignment と同等の性能が、少ない状態変数で（素子数が少ない）実現できる場合がある。

ii)　中間状態の数を制限する。

中間状態の数は、状態遷移に必要な時間と比例関係にあり、この数があまり大きくなることはさけなくてはいけない。

iii)　free runningを含める。

遷移する道を指定するとき、critical raceの生じない２bit 以上の遷移が可能な場合は、その部分をfree runningとして中間状態を減らす。

なお、この順序回路自動設計システムは、設計言語ADL[3)]を用いて実現中である。

5、結言

Saucierにより示された状態割当てのアルゴリズムを実現する上で指摘された問題点の一つとして、edgeにつける重みについて考察を行ない、数値計算で重みを求めるためのアルゴリズムを示した。その結果、Saucierの重みのつけかたにおける問題点が明らかになるとともに、数値計算で重みの大きさが求められることになった。

また、重みのつけかたを改善する方法としては、重みを表わすための場合分け

は、Saucierが W_1, W_2, W_3 と三つの場合であるのに対して、この数を7〜9に増すことが良いと思われる。

謝辞：
自動設計システムの実現に際し、御協力をいただいた大学院学生、山下哲男君ならびに、いろいろ御検討をいただいた相磯研究室の諸氏に感謝します。

参考文献：

[1] G. Saucier, "State Assignment of Asynchronous Sequential Machines Using Graph Techniques", IEEE Trans., C-21, 3, PP. 282-288, March 1972.

[2] J.H. Tracey, "Internal state assignments for asynchronous sequential machines", IEEE Trans., EC-15, PP. 551-560, Aug. 1966.

[3] M. Tokoro, "A Design Language ADL and its Application to Logic Design", Proc. 1st USA-Japan computer conference, 1972.