

並列局所探索法における近傍サイズの最適決定方式とその組合せ回路テスト生成への応用

夏目 幸一郎† 島山 一実† 伊達 博†

† (株)日立製作所 日立研究所

あらまし 局所探索法は組合せ最適化問題を解く発見的手法の一つであり、可能解 x の近傍にあるより良い可能解 y を x に代入する、という処理を行う。本論文では、特別な組合せ最適化問題を局所探索法で解く場合において、可能解 x をできるだけ速く最適解に近づけることを考える。その速度を理論的に求め、それを最大にする近傍サイズの近似式を導く。さらに、その結果を応用した組合せ回路の並列テストパターン生成手法を提案する。

Optimization of neighborhood size for parallel local search and its application to test generation of combinational circuits

Koichiro Natsume † Kazumi Hatayama † Hiroshi Date †

† Hitachi Research Laboratory, Hitachi, Ltd.

Abstract The local search is a heuristic approach to solve a combinational optimization problem. It includes the step of searching a better feasible solution y in the neighborhood of a feasible solution x and substituting y for x . In this paper, we consider an approach to bring a feasible solution x close to the optimum solution as fast as possible in the case of solving specific combinational optimization problem by the local search. We analyze the speed theoretically and present an approximation of optimal neighborhood size which maximize the speed. Moreover we apply the result to the parallel test generation of combinational circuits.

1 はじめに

局所探索法は組合せ最適化問題を解く発見的手法の一つである。初期可能解 x_0 を生成して x に代入し、 x の近傍 $N(x)$ 内により良い可能解 y があれば y を x に代入して同様の処理をくり返す。 x の近傍 $N(x)$ 内により良い可能解がない場合には x を出力して処理を終了する。

これを並列化する手法には、最良優先探索法とマルチ山登り法がある⁽¹⁾。最良優先探索法では、近傍探索を並列実行し、より良い可能解のうち最良のものを x に代入する。マルチ山登り法では、各プロセッサが独立に局所探索法を実行し、それらの結果のうち最良のものを最適解として出力する。

本論文では、特別な条件を満たす組合せ最適化問題を複数台のプロセッサにより最良優先型局所探索法で解く場合において、可能解 x をできるだけ速く最適解に近づけることを考える。その速度を理論的に求め、それを最大にする近傍サイズの近似式を導く。さらに、その結果を応用した組合せ回路の並列テストパターン生成手法を提案する。

2 組合せ最適化問題とそのためのハードウェア構成およびアルゴリズム

以下、本論文で議論の対象とする組合せ最適化問題と、それを解くためのハードウェア構成およびアルゴリズムを示す。

(1) 組合せ最適化問題

目的関数

$$f(x) = f(x_1, x_2, \dots, x_n) \rightarrow \text{最小}$$

制約条件 $x \in \{0, 1\}^n$

問題に関する仮定

(a) 目的関数値 $f(x)$ を最小にする最適解はただ一つしかない。

(b) 可能解 $x = (x_1, x_2, \dots, x_n)$ と最適解とのハミング距離 h (以下、FO距離と呼ぶ) と、目的関数値 $f(x)$ との間には、次式の関係がある (g は単調非減少関数)。

$$h = g(f(x)) \quad (2-1)$$

(c) 処理時間の大部分は、目的関数値の計算が

占めている。したがって、プロセッサ間の通信など、それ以外のことに要する時間は無視できる。

(2) ハードウェア構成

使用するプロセッサは全部で $(p+1)$ 台であり、そのうち1台は並列処理全体を管理するマスタープロセッサ、それ以外は近傍探索用のスレーブプロセッサである (図1)。

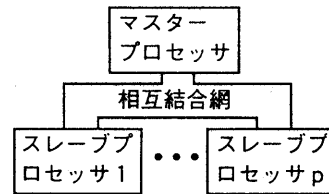


図1. ハードウェア構成

(3) アルゴリズム

(S1) マスタープロセッサは、 $x_0 \in \{0, 1\}^n$ をランダムに生成して $x \leftarrow x_0$ とする。 x の目的関数値 $f(x)$ を計算して $\text{Min} \leftarrow f(x)$ とする。

(S2) マスタープロセッサは、式(2-1)を用いて可能解 x のFO距離を計算する。 $h=0$ ならば(S8)へ。

(S3) マスタープロセッサは、近傍サイズ r を求める。可能解 x と近傍サイズ r を各スレーブプロセッサに送信する。

(S4) 各スレーブプロセッサは、可能解 x の成分をランダムに r 個選んで0になっているものを1に、1になっているものを0に変更した、変更解 y を t 個生成する。それらの目的関数値を計算する。

(S5) 各スレーブプロセッサは、 t 個の変更解のうち、目的関数値が最小のものを一つ選び、目的関数値と共にマスタープロセッサに送信する。

(S6) マスタープロセッサは、送信された p 個 (p はスレーブプロセッサ数) の変更解のうち、目的関数値が最小のもの b を一つ選ぶ。 $f(b) < f(x)$ ならば $x \leftarrow b$, $\text{Min} \leftarrow f(b)$ とする。

(S7) (S2)へ。

(S8) マスタープロセッサは、 x を出力する。

終わり。

3 近傍サイズの最適値の決定

3.1 平均・FO距離減少速度の定義

本アルゴリズムにおいては、可能解 x ができるだけ速く最適解に近づく、すなわち、(S2)と(S7)におけるFO距離 h の差の期待値が、(S2)から(S7)までの所要時間の割に大きくなることが望ましい。(S2)から(S7)までの所要時間は、前章の仮定のもとでは可能解更新頻度 t に比例する。そこで、次のように定義する。

<定義1>

近傍サイズ r と可能解更新頻度 t は整数で、 $1 \leq r \leq n$, $t \geq 1$ とする。このとき、次式の値を平均・FO距離減少速度と呼ぶ。

$$v = v(r, t) =$$

((S2)におけるFO距離 h - (S7)におけるFO距離 h_{NEW} の期待値) / t 。

以下、定義1による平均・FO距離減少速度 v の値を最大にするような近傍サイズ r の近似値と可能解更新頻度 t の値を決定することを目標にして議論を進める。

3.2 平均・FO距離減少速度の計算

本節では、前節で定義した平均・FO距離減少速度 v の表式を導く。ただし、 $j < 0$ または $j > i$ の場合には ${}_i C_j = 0$ とする。また、 $[x]$ は x を越えない最大の整数を表す。

まず、本アルゴリズムの(S4)において、可能解 $FS = (FS_1, FS_2, \dots, FS_n)$ の成分をランダムに r 個選んで0になっているものを1に、1になっているものを0に変更することにより、変更解 $CS = (CS_1, CS_2, \dots, CS_n)$ を生成したとする。このとき、最適解 $OS = (OS_1, OS_2, \dots, OS_n)$ に対して、 $FS_i = OS_i$ かつ $CS_i \neq OS_i$ である添字 i の個数を X_{CS} として $X_{CS} = k$ となる確率 $P(X_{CS} = k)$ を求める。

n 個 (n は決定変数の個数) の添字のなかから r 個を選び出す組合せは ${}_n C_r$ 通りある。 $FS_i = OS_i$ である添字は $(n-h)$ 個あり、それら

のなかから k 個の添字を選び出す組合せは

${}_{n-h} C_k$ 通りある。また、 $FS_i \neq OS_i$ である添字は h 個あり、それらのなかから $(r-k)$ 個の添字を選び出す組合せは ${}_h C_{r-k}$ 通りある。したがって、求める確率は

$$P(X_{CS} = k) = \frac{{}_{n-h} C_k \cdot {}_h C_{r-k}}{{}_n C_r}$$

となる。また、このとき、変更解 CS のFO距離は $h - (r - 2k)$ となる。

次に、 pt 個 (p はスレーブプロセッサ数、 t は可能解更新頻度) 生成される変更解 CS のうち、目的関数値が最も小さいものを $BS = (BS_1, BS_2, \dots, BS_n)$ とする。 $FS_i = OS_i$ かつ $BS_i \neq OS_i$ である添字 i の個数を X_{BS} として、 $X_{BS} = k$ となる確率 $P(X_{BS} = k)$ を求める。 BS は、 pt 個生成される変更解のうちFO距離が最も小さいものであるのもので、

$$P(X_{BS} = k) = P(X_{BS} \geq k) - P(X_{BS} \geq k+1)$$

$$= \{P(X_{CS} \geq k)\}^{pt} - \{P(X_{CS} \geq k+1)\}^{pt}$$

$$= \left(\sum_{i=k}^r \frac{{}_{n-h} C_i \cdot {}_h C_{r-i}}{{}_n C_r} \right)^{pt} - \left(\sum_{i=k+1}^r \frac{{}_{n-h} C_i \cdot {}_h C_{r-i}}{{}_n C_r} \right)^{pt}$$

となる。

次に、平均・FO距離減少速度 v を求める。

$X_{BS} > r/2$ の場合には、(S7)における可能解 x は(S2)におけるものと同じなので $h_{NEW} = h$ となり、 $X_{BS} = r/2$ の場合にもやはり $h_{NEW} = h$ となる。したがって、これらの場合は平均・FO距離減少速度 v の値に寄与しない。

一方、 $X_{BS} < r/2$ すなわち $X_{BS} \leq [(r-1)/2]$ の場合には、 $h_{NEW} = h - (r - 2X_{BS})$ となる。したがって、

$$v = v(r, t) = \frac{1}{t} \sum_{k=0}^{[(r-1)/2]} (r-2k) P(X_{BS} = k)$$

$$= \frac{1}{t} \sum_{k=0}^{[(r-1)/2]} (r-2k) \left\{ \left(\sum_{i=k}^r \frac{{}_{n-h} C_i \cdot {}_h C_{r-i}}{{}_n C_r} \right)^{pt} - \left(\sum_{i=k+1}^r \frac{{}_{n-h} C_i \cdot {}_h C_{r-i}}{{}_n C_r} \right)^{pt} \right\}. \quad (3-1)$$

ここに, v : 平均・F O 距離減少速度,
 r : 近傍サイズ,
 t : 可能解更新頻度,
 n : 決定変数の個数,
 p : スレーブプロセッサ数,
 h : 現在解の F O 距離.

となる.

3. 3 最適近傍サイズに関する考察の準備

以下, 近傍サイズ r と可能解更新頻度 t の最適値について理論的な考察を行うのに必要な, 記号を定義し補題を述べる.

<定義 2 >

n, p, h を整数の定数とし, $n \geq 1, p \geq 1, 1 \leq h \leq n$ とする. このとき, x と y の 2 変数関数 $A(x, y), B(x, y), C(x, y)$ を次式で定義する. ただし, x と y は整数で, $1 \leq x \leq n, y \geq 0$ とする. また, $y > x$ の場合には $B(x, y) = 0$ とする.

$$A(x, y) = \frac{{}^{n-h}C_y \cdot {}^h C_{x-y}}{{}^n C_x} \quad (3-2)$$

$$B(x, y) = \sum_{z=y}^x A(x, z) \quad (3-3)$$

$$C(x, y) = 1 - \{B(x, y)\}^p \quad (3-4)$$

<補題 1 >

$M \geq 0$ とする. $X_m (0 \leq m \leq M)$ と $Y_m (0 \leq m \leq M)$ は $(M+1)$ 項から成る有限数列で, 次の条件を満たすものとする.

$$X_m > 0 \quad (0 \leq m \leq M) \quad (3-5)$$

$$\sum_{j=0}^M X_j = \sum_{j=0}^M Y_j = S \quad (3-6)$$

$$\frac{Y_0}{X_0} \geq \frac{Y_1}{X_1} \geq \dots \geq \frac{Y_M}{X_M} \quad (3-7)$$

(a) このとき,

$$\sum_{j=m}^M X_j \geq \sum_{j=m}^M Y_j \quad (1 \leq m \leq M) \quad (3-8)$$

が成り立つ.

(b) さらに, $Y_0/X_0 > Y_M/X_M$ ならば,

$$\sum_{j=m}^M X_j > \sum_{j=m}^M Y_j \quad (1 \leq m \leq M) \quad (3-9)$$

が成り立つ.

3. 4 可能解更新頻度の最適値

本節では, 可能解更新頻度 t の最適値 (平均・F O 距離減少速度 v を最大にする値) t_{opt} = 1 であることを理論的に示す.

<定理 1 >

$r (1 \leq r \leq n)$ を定数と見た場合, $v(r, t)$ は $t=1$ のときに最大値を取る, すなわち, $t_{opt} = 1$ となる.

<略証 >

(a) $1 \leq r < h$ の場合

$$v(r, t) = 2 \sum_{k=1}^{[(r-1)/2]} \frac{1 - \{B(r, k)\}^{pt}}{t + \{r - 2[(r-1)/2]\} \frac{1 - \{B(r, [(r-1)/2] + 1)\}^{pt}}{t}}$$

と表示できる. $1 \leq k \leq [(r-1)/2] + 1$ である全ての k に対して $0 \leq B(r, k) < 1$ なので, この式は $(1 - s^t) / t (0 \leq s < 1)$ の形をした項の重み付きの和となっている. 各項は $t=1$ のときに最大値を取る, v も $t=1$ のときに最大値を取る.

(b) $h \leq r \leq n$ の場合

$$v(r, t) = 2 \sum_{k=1}^{[(2h-r-1)/2]} \frac{1 - \{B(r, k+r-h)\}^{pt}}{t + \{r - 2[(r-1)/2]\} \frac{1 - \{B(r, [(r-1)/2] + 1)\}^{pt}}{t}}$$

と表示できる. $r - h + 1 \leq k \leq [(r-1)/2] + 1$ である全ての k に対して $0 \leq B(r, k) < 1$ なので, この式は $(1 - s^t) / t (0 \leq s < 1)$ の形をした項の重み付きの和となっている. 各項は $t=1$ のときに最大値を取る, v も $t=1$ のときに最大値を取る. (証明終)

3. 5 最適近傍サイズについての理論的検討

本節では, 可能解更新頻度 $t=1$ としたときの近傍サイズ r の最適値 (平均・F O 距離減少速度

vを最大にする値) r_{opt} についての理論的結果を示す。ただし、最適な近傍サイズがただ一つに確定しない場合には、それらのうち最小のものを選ぶものとする。

<定理2>

nとpを任意の正の整数とする。

$1 \leq h \leq n/2$ である全てのhに対して、

$r_{opt} \leq h$ となる。

<略証>

hとr ($1 \leq h \leq n/2, h < r < 2h$) を定数とすると、 $\min(n-r, h)$ をM,

$A(r, i+r-h)$ を X_i , $A(2h-r, i)$ を Y_i とみなすと、これらは補題1の(a)の仮定において、jをiと読みかえたものを満たすことを示す。

式(3-5)と(3-6)が満たされることは容易に分かる。

$0 \leq i \leq \min(n-r, h)$ として、iの関数D(i)を次式で定義する。

$$D(i) = \frac{A(2h-r, i)}{A(r, i+r-h)}$$

すると、計算により、

$$D(i) = \begin{cases} \frac{(2h-r)!(n+r-2h)!}{r!(n-r)!} \cdot \prod_{j=1}^{r-h} \frac{2h-r-i+j}{n-r-i+j} \\ (0 \leq i \leq 2h-r), \\ 0 \quad (2h-r < i \leq \min(n-r, h)), \end{cases}$$

となる。

D(i)は $0 \leq i \leq 2h-r$ の範囲では正の値を取る単調非増加関数 ($r-h$) 個と正の定数との積の形をしており、 $2h-r < i \leq \min(n-r, h)$ の範囲では0である。ゆえに、 $0 \leq i \leq \min(n-r, h)$ の範囲において単調非増加であり、式(3-7)が満たされることが分かる。

ゆえに、補題1の(a)より、 $1 \leq k \leq$

$\min(n-r, h)$ であるkに対して、

$$\sum_{i=k}^{\min(n-r, h)} A(r, i+r-h) \geq \sum_{i=k}^{\min(n-r, h)} A(2h-r, i)$$

となり、さらに、

$$C(r, k+r-h) \leq C(2h-r, k)$$

となることも容易に分かる。

一方、 $h < r < 2h$ であるrに対して、

$$v(r, 1) = 2 \sum_{k=1}^{[(2h-r-1)/2]} C(r, k+r-h) + \{r-2[(r-1)/2]\} C(r, [(2h-r-1)/2]+1+r-h),$$

$$v(2h-r, 1) = 2 \sum_{k=1}^{[(2h-r-1)/2]} C(2h-r, k) + \{r-2[(r-1)/2]\} C(2h-r, [(2h-r-1)/2]+1),$$

と表示できるので、そのようなrに対しては $v(r, 1) \leq v(2h-r, 1)$ である。

また、 $r \geq 2h$ のときには $v(r, 1) = 0$ となる。

ゆえに、 r_{opt} の定義より、 $h < r_{opt} < 2h$ とも $r_{opt} \geq 2h$ ともなりえないので、 $r_{opt} \leq h$ である。

以上より、定理は証明された。(証明終)

<定理3>

nとpを任意の正の整数とする。

$n/2 < h \leq n$ である全てのhに対して、

$r_{opt} \geq h$ となる。

<略証>

hとr ($n/2 < h \leq n, 2h-n \leq r < h$) を定数とすると、 $\min(n-h, r)$ をM, $A(r, i)$ を X_i , $A(2h-r, i+h-r)$ を Y_i とみなすと、これらは補題1の(b)の仮定において、jをiと読みかえたものを満たすことを示す。

式(3-5)と(3-6)が満たされることは容易に分かる。

$0 \leq i \leq \min(n-h, r)$ として、iの関数E(i)を次式で定義する。

$$E(i) = \frac{A(2h-r, i+h-r)}{A(r, i)}$$

すると、計算により、

$$E(i) = \begin{cases} \frac{(2h-r)!(n+r-2h)!}{r!(n-r)!} \cdot \prod_{j=1}^{h-r} \frac{n+r-2h-i+j}{r-i+j} \\ \quad (0 \leq i \leq n+r-2h), \\ 0 \quad (n+r-2h < i \leq \min(n-h, r)), \end{cases}$$

となる。

$E(i)$ は $0 \leq i \leq n+r-2h$ の範囲では正の値を取る単調非増加関数 $(h-r)$ 個と正の定数との積の形をしており、 $n+r-2h < i \leq \min(n-h, r)$ の範囲では0である。ゆえに、 $0 \leq i \leq \min(n-h, r)$ の範囲において単調非増加であり、式(3-7)が満たされることが分かる。

さらに、 $i=0$ のとき $E(i) > 0$ 、 $i = \min(n-h, r)$ のとき $E(i) = 0$ である。

ゆえに、補題1の(b)より、 $1 \leq k \leq \min(n-h, r)$ である k に対して、

$$\sum_{i=k}^{\min(n-h, r)} A(r, i) > \sum_{i=k}^{\min(n-h, r)} A(2h-r, i+h-r)$$

となり、さらに、

$C(r, k) < C(2h-r, k+h-r)$ となることも容易に分かる。

また、 $\min(n-h, r) < k \leq [(r-1)/2] + 1$ である k に対しては、 $C(r, k) = C(2h-r, k+h-r) = 1$ となることを示すのもたやすい。

一方、 $2h-n \leq r < h$ である r に対して、

$$v(r, 1) = 2 \sum_{k=1}^{[(r-1)/2]} C(r, k) + \{r-2[(r-1)/2]\} C(r, [(r-1)/2+1]),$$

$$v(2h-r, 1) = 2 \sum_{k=1}^{[(r-1)/2]} C(2h-r, k+h-r)$$

$$+ \{r-2[(r-1)/2]\} C(2h-r, [(r-1)/2+1+h-r]),$$

と表示できるので、 $[(r-1)/2] + 1 \leq \min(n-h, r)$ の場合にも

$[(r-1)/2] + 1 > \min(n-h, r)$ の場合にも $v(r, 1) < v(2h-r, 1)$ で

ある。

また、容易に分かるように、 $n/2 < h \leq n$ とすると、 $r=n$ のときには $v(r, 1) = 2h-n$ 、 $1 \leq r < 2h-n$ のときには $v(r, 1) \leq r$ となる。

ゆえに、 r_{opt} の定義より、 $2h-n \leq r_{opt} < h$ とも $1 \leq r_{opt} < 2h-n$ ともなりえないので、 $r_{opt} \geq h$ である。

以上より、定理は証明された。(証明終)

<定理4>

n を任意の正の整数とする。 p が十分大きい場合には、 $1 \leq h \leq n$ である全ての h に対して $r_{opt} = h$ となる。

<略証>

まず、 $B(x, y)$ の値を調べる。容易に分かるように、

(a)、 $x \geq h$ の場合

$0 \leq y \leq x-h$ のときには $B(x, y) = 1$ 、 $x-h < y \leq \min(n-h, x)$ のときには $0 < B(x, y) < 1$ 、 $y > \min(n-h, x)$ のときには $B(x, y) = 0$ である。

(b)、 $x < h$ の場合

$y=0$ のときには $B(x, y) = 1$ 、 $0 < y \leq \min(n-h, x)$ のときには $0 < B(x, y) < 1$ 、 $y > \min(n-h, x)$ のときには $B(x, y) = 0$ である。

次に、 $p \rightarrow \infty$ のときの $v(r, 1)$ の値を調べる。 $v(r, t)$ の定義より、

(c)、 $h \leq r \leq n$ の場合

$r-h \leq [(r-1)/2]$ すなわち $r < 2h$ のとき、 $v(r, 1) \rightarrow 2h-r$ となる。

$r-h > [(r-1)/2]$ すなわち $r \geq 2h$ のとき、 $v(r, 1) \rightarrow 0$ となる。

(d)、 $1 \leq r < h$ の場合

$v(r, 1) \rightarrow r$ となる。

したがって、 $p \rightarrow \infty$ としたときに $v(r, 1)$ が近づく極限関数 $g(r)$ が定義できて、

$$g(r) = \begin{cases} r & (1 \leq r < h), \\ 2h-r & (h \leq r < 2h), \\ 0 & (2h \leq r \leq n), \end{cases}$$

と表示できる。

$g(r)$ は $r=h$ のときに最大値 h を取り、
 $r \neq h$ であるどんな h に対しても、 $g(r) \leq h-1$ である。

したがって、 h に依存して決まる値である
 $p_0(h)$ をうまく選ぶことにより、次のことが
 成り立つようにできる。

$p \geq p_0(h)$ を満たす全ての p に対して、
 $v(r, 1)$ は $r=h$ のときかつそのときに限り
 最大値を取る。

ゆえに、 $p \geq \max p_0(h)$ (h の範囲は
 $1 \leq h \leq n$) である全ての p に対して、 $r_{opt} = h$ ($1 \leq h \leq n$) となる。

ゆえに、定理は証明された。(証明終)

3.6 最適傍サイズの近似式

本節では、数値実験に基づいて最適傍サイズ
 r_{opt} の近似式を導く。

図2に、決定変数の個数 $n=30$ 、スレーブ
 プロセッサ数 $p=2$ および 15 、可能解更新頻度 t
 $=1$ としてFO距離 h と最適傍サイズ r_{opt}
 との関係を示した。黒丸が $p=2$ の場合を、白丸
 が $p=15$ の場合を示している。ただし、二重丸
 は黒丸と白丸が重なっていることを示している。
 n, p および t を一定とした場合、 r_{opt} の値
 については次のような傾向が観察される。

(1)、 r_{opt} は $h=n/2$ の付近で最も急激
 に変化する。

(2)、 h が $n/2$ に上から近づくとときよりも下
 から近づくとときのほうが、 r_{opt} は $h=n/2$
 のときの値 (n が奇数の場合には
 $h=(n-1)/2$ のときと $h=(n+1)/2$
 のときの値との平均値) に速く近づく。

定理2から4および上記の性質(1)と(2)
 を満足させるため、

$$h_{st} = \frac{2h}{n} - 1 \quad (3-10)$$

によりFO距離 h を (-1) 以上 $(+1)$ 以下の
 値に変換し、それを用いた

$$r_{a-opt} = \begin{cases} \frac{n}{2} \left\{ 1 - (-h_{st})^{\frac{1}{\alpha_1 n^{\beta p \cdot \gamma + 1}}} \right\} & (h < \frac{n}{2}), \\ \frac{n}{2} \left(1 + h_{st}^{\frac{1}{\alpha_2 n^{\beta p \cdot \gamma + 1}}} \right) & (h \geq \frac{n}{2}), \end{cases} \quad (3-11)$$

という式を r_{opt} の近似式として用いることに
 した。

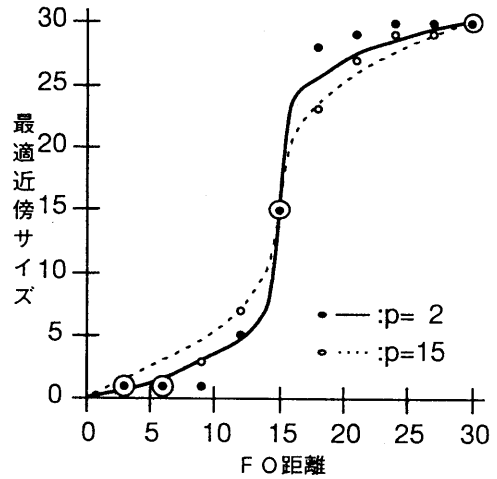


図2 FO距離と最適傍サイズ ($n=30$)

式(3-11)に含まれる $\alpha_1, \alpha_2, \beta, \gamma$
 の値は、式(3-12)で与えられる r_{a-opt}
 の誤差の二乗和 S が最も小さくなるように決めた。
 ただし、式(3-12)において、 n の範囲は
 $5 \leq n \leq 35$ を満たす5の倍数、 p の範囲は $2 \leq$
 $p \leq 20$ を満たす整数、 h の範囲は $1 \leq h \leq n$ を
 満たす整数とする。

$$S = \sum_n \sum_p \sum_h (r_{a-opt} - r_{opt})^2 \quad (3-12)$$

最初は各定数 $\alpha_1, \alpha_2, \beta, \gamma$ の変域を十分
 広く取って比較的粗い格子を設定し、その中から
 S を最小にするものを選択した。その後、徐々に
 変域を狭め、格子を細かくしていくことにより精
 度を上げていった。その結果、 $\alpha_1=0.46,$
 $\alpha_2=0.53, \beta=0.67, \gamma=0.40$ と
 求まった。

式(3-11)にこれらの値を代入した

$$r_{a-opt} = \begin{cases} \frac{n}{2} \left\{ 1 - \left(-h_{st} \right)^{\frac{1}{0.46n^{0.67}p^{-0.40}+1}} \right\} & \left(h < \frac{n}{2} \right), \\ \frac{n}{2} \left(1 + h_{st}^{\frac{1}{0.53n^{0.67}p^{-0.40}+1}} \right) & \left(h \geq \frac{n}{2} \right), \end{cases} \quad (3-13)$$

による h と r_{a-opt} の関係を図2の曲線で示す。実線が $p=2$ の場合を、点線が $p=15$ の場合を示している。

4 本手法の組合せ回路テスト生成への応用

図3に示した2つの組合せ回路C1とC2を考える。C1は通常の組合せ回路であるが、C2のNOTゲート(論理ゲートの種類については図4を参照)の出力線には故障があり、どのような場合にも信号値が1になってしまう。これらの2つの回路の間で出力信号値に違いが生じるような入力信号の組合せ(テストパターン)を求めることを考える。

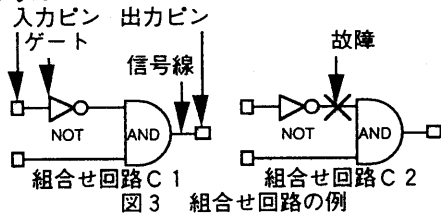


図3 組合せ回路の例

ゲート	出力信号値
	入力値が0の場合 1 入力値が1の場合 0
	全入力値が1の場合 1 その他の場合 0
	全入力値が0の場合 0 その他の場合 1

図4 論理ゲートの種類と出力信号値

この例題では、表1に示すように4通りの入力ベクトル(入力信号の組合せ)に対する出力信号値の異同を全部調べることにより、入力ベクトル




(1, 1)だけがテストパターンになっていることが分かる。

表1 入力ベクトルと目的関数値

入力ベクトル	出力信号値		テストパターンか?	目的関数値	FO距離
	C1	C2			
(0,0)	0	0	NO	10000.00	2
(0,1)	1	1	NO	101.01	1
(1,0)	0	0	NO	101.01	1
(1,1)	0	1	YES	1.02	0

組合せ回路のテストパターンを生成する問題はNP完全なので(2)、テストパターンがあれば必ず求められる厳密解法(3, 4)には回路規模の多項式時間で終了する保証がなく、また、プログラムが複雑になる。そこで、入力ベクトルのFO距離と相関のある目的関数を定義し、その値を最小にするものを発見的に探索する方法が提案されるようになった。我々が以前に提案した手法もその一つであり、組合せ回路C1およびC2における実数値の出力値に基づいて目的関数値を計算し、それを近傍サイズ $r=1$ の局所探索法により最小化している(5)。

C1およびC2における実数値の出力値の計算手順を説明する。まず図5(a)のように、(i)入力ピンにおいて、対応する入力ベクトルの成分が0の場合には実数値 ϵ を、1の場合には実数値 $1-\epsilon$ を与え(ただし $0.0 < \epsilon < 0.5$)、(ii)故障のある信号線には、その故障が信号値を0に固定する場合には実数値0を、1に固定する場合には実数値1を与える。次に、各論理ゲートに対して入力ピンから順番に図5(b)に示すゲート演算規則を適用して実数出力値を計算する。

実数値を 与える箇所		与える 実数値	ゲート	演算規則
入力 ピン	入力ベク トルの成 分が0	ϵ		$1-a$
	入力ベク トルの成 分が1	$1-\epsilon$		$\prod_{i=1}^m a_i$
故障 箇所	信号値を 0に固定 する	0		$1-\prod_{i=1}^m (1-a_i)$
	信号値を 1に固定 する	1		

(a) 実数値出力値計算のために与える値 (b) ゲート演算規則

図5 実数値出力値計算規則

入力ベクトルの目的関数値は、その結果を用いて次式で求められる。ただし、出力値とは実数値出力値のことである。

目的関数値＝

$$\frac{1}{\sum_{\text{全出力ピン}} |C1 \text{ の出力値} - C2 \text{ の出力値}|} \quad (4-1)$$

$\epsilon = 0.01$ として、図3の回路C1およびC2を考えた場合における、各入力ベクトルに対して決まる目的関数値とF0距離を表1に示した。例えば、入力ベクトル(0, 0)に対しては、C1の実数値出力値は $(1 - 0.01) \times 0.01 = 0.0099$ 、C2の実数値出力値は $1 \times 0.01 = 0.01$ と計算され、目的関数値は式(4-1)より10000.00と求められる。また、F0距離は2である。この結果より、目的関数値が小さい入力ベクトルはテストパターンであり、目的関数値が大きい入力ベクトルはテストパターンでないので、目的関数値を最小化することによりテストパターンを求めることができる。また、F0距離hと目的関数値cとの間には近似的に、

$$h \approx \log_{1/\epsilon} c \quad (4-2)$$

の関係があり(ただし、 ϵ は実数値出力値を計算する際に、対応する入力ベクトルの成分が0の場合

合に入力ピンに与える値)、第1章の仮定(b)が成り立っていることも分かる。これに加えて第1章の仮定(a)および(c)の成立が検証できれば、本手法が組合せ回路の並列テスト生成に有効に適用できると言える。

5 むすび

本論文では、特別な条件を満たす組合せ最適化問題を複数台のプロセッサにより最良優先型局所探索法で解く場合において、最適近傍サイズの近似式を導いた。さらに、その結果を応用した組合せ回路の並列テストパターン生成手法を提案した。

今後は、大規模な回路に対して第1章の仮定が実用上十分な精度で成り立っていることを検証し、プログラム実験により本手法の高速化効果を確認する予定である。

6 参考文献

- (1), 石川幹人, 十時 泰, 戸谷智之, 星田昌紀, 広沢 誠: “並列反復改善法によるタンパク質の配列解析”, 情報処理学会論文誌, Vol.35, No.12, Dec. 1994.
- (2), Ibarra, P. H. and Sahni, S. K.: Polynomially complete fault detection problems, IEEE Trans. Comput., Vol.C-24, No.3, pp.242-249 (1975).
- (3), Roth, J.P., Bouricius, W.G., and Schneider, P.T.: Programmed Algorithms to Compute Tests to Detect and Distinguish between Failures in Logic Circuits, IEEE Trans. Electron. Comput., EC-16, 5, pp.567-579 (1967).
- (4), Goel, P.: An Implicit Enumeration Algorithm to Generate Tests for Combinational Logic Circuits, IEEE Trans. Comput., C-30, 3, pp.215-222 (1981).
- (5), 池田光二, 畠山一実, 林 照峯: “バックトラック処理不要な組合せ回路テスト生成手法”, 信学技報, FTS89-35 (1989-10).