設計自動化28−2
(1985・9・17)

インターラクティブDRCの一手法

An Interactive DRC Method

鈴木 五郎　国友 佳男　浜田 直
G.Suzuki　Y.Kunitomo　N.Hamada

(株)日立製作所 日立研究所
Hitachi Ltd. Hitachi Res. Lab.

1.緒 言

汎用VLSIの実装設計においては、ハンド・クラフト設計からオンライン設計への転換が求められている。 オンライン設計の実現手段としてスペ-シング機能を持ったシンボリック・レイアウト・システム[1]がある。 ところが、汎用VLSIはチップ・サイズに関して非常に厳しい制約があるため、実用化されたという報告はいまだに聞かれない。そこで効果的な手段として、設計規則に関するコレクト・バイ・コンストラクションの機能を持たせることが考えられる。すなわち設計者が一個の図形を入力する毎に、それが設計規則に違反するか否かをインターラクティブに計算機がチェックし、常に設計規則に違反しない保証をとりながら設計を行う方式である。 この機能を実現するにはDRC(設計規則チェック)を高速化するという技術課題がある。

主な設計規則は表1のように5項目あり、Space, Enclosure,Incursionのように図形間の距離を規定した第一のグループと、WidthやNotchのように自分自身の形状に関する規則を規定した第二のグループに大別される。 インターラクティブにDRCをしようとした場合、第二のグループはチェックの対象となる図形が既に認識されているのに対して、第一のグループは大量に存在する既入力図形の中からチェックの対象になるものを抽出しなければならない。つまり第一グループのチェックのほうががなり処理時間を必要とすることが予想される。 そこで今回、第一グループの規則に注目し、チェック対象図形の高速抽出法と図形間距離計算法を検討した。

2.チェック対象図形高速抽出法

チェック対象となる図形を高速に抽出するには、各図形が図面中でどこに存在しているかを管理する図面情報管理方式が必要である。 従来から各種の管理方式が考案されているが、VLSIのような莫大な図形量(10万トランジスタ-のオ-ダ-)を管理しようとすると、その効果を十分に発揮できないことが分かった。 従来からある図面情報管理方式の代表例である領域分割法[2]の問題点を述べ、次に我々の提案する管理方式フィールド・ブロック法を説明する。

図形番号giか割り当てられた6個の矩形(多角形の場合には外接する矩形を考える)が配置されている図1(a)を例題とする。 図1(b)は各図形の実体情報を格納する実体情報テ-ブルを示しており、補助記憶装置との間でスワッピング処理を行うという前提に立ちスワッピング処理の単位となるブロックにグ

ループ化しておく。 実体情報テーブルの1列が1ブロックに対応している。 入力された各図形の実体情報Giは入力順にこのテーブルに格納され、Giのヘッダーとして矩形の左下、右上座標(x_{min},y_{min}),(x_{max},y_{max})が格納されている。 以下では図面情報管理方式のいかんにかかわらず実体情報テーブルの構造は変らないと仮定する。

2.1 領域分割法

図2(a)のように図面をあらかじめ格子状の領域Riに分割しておき、各図形がどこの領域に存在しているかを管理する方式である。 例えば図2(b)のように各領域毎にそこに存在する図形情報か格納されている実体情報テーブルのブロック番号を管理しておく。 この方式では図面の規模が大きくなると次の点が問題となる。

(1) 管理情報の作成・変更

例えば図2(c)のように図形g_3とg_4とを移動し場合、図形が存在しなくなった領域と新たに存在するようになった領域の両方の管理情報を変更しなければならない。 (図2(d))

(2) 管理情報量が莫大

必要な実体情報の抽出を高速化しようとして分割を細かくすると、複数の領域にまたがる図形の個数が増加し管理情報量が莫大になってしまう。 そこで管理情報が実体情報を圧迫して実体情報のスワッピング頻度が高くなり、実体情報の抽出に時間がかかることになる。

2.2 フィールド・ブロック法

我々が提案する管理方式では管理情報量が極めて少なく且つ管理情報の作成・変更が非常に容易になるという特徴を持っている。

2.2.1 基本概念

フィールドブロック法は実体情報テーブルのブロック毎にその物理的な広がりの情報を持たせることにより管理を行う方式である。 ブロックの物理的な広がりは次に定義する4個の座標(Xmin,Ymin),(Xmax,Ymax)で表現する。

$$Xmin = \min\{x_{min}\}$$

$$Ymin = \min\{y_{min}\}$$

$$Xmax = \max\{x_{max}\}$$

$$Ymax = \max\{y_{max}\}$$

ここで(x_{min},y_{min}),(x_{max},y_{max})は該当するブロック内に格納されている各図形Giのヘッダー(hi)である。このように物理的な広がりの情報を持ったブロックのことをフィールド・ブロックと呼ぶことにする。例題図面では図3(a)のように破線で表現した2つのフィールド・ブロックが存在する。 管理情報としては図3(b)のように(Xmin,Ymin),(Xmax,Ymax)だけである。

図面中のある矩形領域にかかる図形は次の手順で抽出される。

(P1) 矩形の左下、右上座標が指定される。

(P2) 管理情報テーブルと格納されている各フィールド・ブロックの(Xmin,Ymin)、(Xmax,Ymax)から、指定された矩形にかかっている全てのフィールド・ブロックの番号を抽出し、実体情報が格納されている主記憶装置または、補助記憶装置上の位置を求める。

(P3) (P2)で求めた一つのフィールド・ブロックに注目。 それが主記憶装置上と存在している場合には(P4)へ、存在していない場合には補助記憶装置上の該当ブロックと主記憶装置上の不要なブロックとの間でスワッピング処理を行う。 不要なブロックは例えばLRU法[3](Least Recently Used法)により決定する。

(P4) フィールド・ブロック内に格納されている各図形に注目し、そのヘッダー情報(x_{min},y_{min})(x_{max},y_{max})から矩形にかかる図形を判定する。

(P5) (P2)で求めた全てのフィールド・ブロックに関して(P3),(P4)を繰り返す。

2.2.2 領域分割法との比較

(1) 管理情報の作成・変更

例えば図3(c)のように図形g_3とg_4とを移動した場合、管理情報の変更は図3(d)のように移動した図形が属しているフィールド・ブロックの広がりに変化があるときのみ行われる。 変更する内容は(Xmin,Ymin),(Xmax,Ymax)だけであるから極めて簡単な処理となる。 図形を追加・回転・ミラー反転・消去した場合でも管理情報の変更は同じ要領で行われる。

(2) 管理情報量

図2(b)と図3(b)を比較すれば、フィールド・ブロック法における管理情報が極めて少量であることが分かる。 つまり量的に管理情報が実体情報を圧迫し、スワッピングの頻度を高めてしまうような事態は生じない。

以上の比較から、フィールド・ブロック法は大量の図形情報をオンライン設計する場合に極めて有効な図面情報管理方式であるといえる。 但し図形の入力順序によっては、図4(a)のようにフィールド・ブロックが大きく広がってしまう場合がある。 このような事態を回避するためには、フィールド・ブロックの自動分割が必要となる。 フィールド・ブロックがある大きさ以上になった場合にはその広がりができるだけ小さくなるように複数のブロックに自動分割している(図4(b))。

3. 図形間距離計算法

ここではフィールド・ブロック法によってチェック対象図形を高速に抽出した後、Space規則を例に上げ、図形間の距離計算をどのように行うかを述べる。

3.1 DRC処理手順の概要

図形の入力後、どのような手順でDRCを行うかについて図5で説明する。

(P1) 入力図形Gと既入力図形gの間のSpace規則の最大値Dを求める。

(P2) Ge(Gの外接矩形)をDだけ拡大した$\widetilde{Ge}$を作る。

(P3) $\widetilde{Ge}$と共通領域を持つフィールド・ブロックを抽出する。

(FBのしぼり込み)

(P4) $\widetilde{Ge}$と共通領域を持つge(gの外接矩形)を抽出する。

(図形のしぼり込み)

(P5) $\Omega=\widetilde{Ge}\cap ge$を求める。

(計算個所のしぼり込み)

(P6) Ωに注目し、図形間の距離を計算する。

(図形間距離計算)

(P7) (P5),(P6)を、$\widetilde{Ge}$と共通領域を持つ全てのgeに関して繰り返す。

(P8) (P4)～(P7)を$\widetilde{Ge}$と共通領域を持つ全てのフィールド・ブロックに関して繰り返す。

ここで(P3)及び(P4)は2.2.1で述べた手順で処理される。

実際の図形間距離計算処理に入る前処理として、対象図形及び計算個所を3段に渡ってしぼり込むことにより高速化を図っている。

3.2 エッジ法と頂点法

2つの図形間の距離を計算する代表的な方法として、辺どうしの距離を計算するエッジ法と、凸頂点(内角が180° 以下の頂点)に注目し、頂点と辺との距離を計算する頂点法[4]がある。 表2にその比較を示す。

ケース1 エッジ法では(辺ab,辺1) (辺ac,辺1)の組み合わせで辺間の計算を行うが、実際には(頂点a,辺1),(頂点b,辺1),(頂点a,辺1),(頂点c,辺1)の4個所を計算する。 一方頂点法では(頂点a,辺1)の1個所だけの計算で済む。

ケース2 エッジ法では辺l_1,辺l_2両端点の4つの組合せを計算するのに対して、頂点法では1個所だけの計算で済む。

以上2つのケースから分かるように凸頂点に注目した図形間距離計算法のほうが効率が良い。特に頂点数が多い図形どうしが複雑にからみ合っているような場合、エッジ法はかなり不利になると予想されるためここでは頂点法を採用し、その高速化を図った。

3.3 Space チェック処理手順

3.1で説明したDRC処理手順のうち(P5)及び(P6)の処理について詳細に説明する。 図6にSpaceチェック手順を示す。

(P1) 2つの図形の各辺をベクトル化し、方向の情報を持たせる。

(P2) Ω(=$\tilde{G}e \cap 9e$)内に存在する入力図形Gと既入力図形gの凸頂点$V_\Omega^{\downarrow}$,$V_\Omega^{\uparrow}$及び辺E_Ω,e_Ωを抽出する。

(P3) E_Ω,e_Ωに関して両端のX,Y座標でソート。

(P4) $V_\Omega^{\uparrow}$の一つに注目し、それを中心とする一辺2Dの正方形φを発生させる。

(P5) φ内に存在するgの辺e_ϕ($e_\phi \subset e_\Omega$)を次の判定基準で抽出する。

(P5.1) e_Ωの方向に注目し、図7(a)に示した表によって対象となる辺をしぼり込む。 上段は注目している$V_\Omega^{\uparrow}$の前後の辺の方向を、下段この頂点に対して距離計算を行う候補となる辺の持つべき方向を表わしている。

(P5.2) (P5.1)でしぼり込んだ辺に関してφと辺との位置関係により最終的に距離計算を行う辺をしぼる。図7(b)において

$(X_{min}^{\phi}, Y_{min}^{\phi})$
$(X_{max}^{\phi}, Y_{max}^{\phi})$:φの左下、右上座標

$(X_{min}^{\ell}, Y_{min}^{\ell})$
$(X_{max}^{\ell}, Y_{max}^{\ell})$:辺の左下、右上座標

右下り斜線の場合には、それを対角線とする矩形の左下、右上座標。

水平線の場合には $Y = Y_{min}^{\ell} = Y_{max}^{\ell}$

垂直線の場合には $X = X_{min}^{\ell} = X_{max}^{\ell}$

を表わす。 辺が斜、水平、垂直の場合それぞれ(a),(b),(c),(d)の条件を満足しているものを拾い上げる。 この際(P3)でソートした情報を使い判定を高速化する。

(P6) $V_\Omega^{\downarrow}$とe_ϕとの間で距離計算を行う。

(P7) (P4)(P5)(P6)を全ての$V_\Omega^{\downarrow}$に関して繰り返す。

(P8) $V_\Omega^{\downarrow}$に注目し、立場を換えて(P4)~(P7)を繰り返す。

4. プログラム評価

フィールド・ブロック法と頂点法をプログラム化してその評価を行なった。 プログラム言語はFORTRAN 77であり、次の条件を設定した。

(1)使用計算機 HITAC M200H (約8MIPS)。

(2)実体情報テーブル・1ブロックの大きさは6144バイト。

(3)主記憶装置上の実体情報テーブルは61440バイト(10ブロック)。

(4)図形は斜線を含む15本の辺から構成されており、実体情報は約200バイトである。 一つのフィールド・ブロックには約30個の図形が存在。

4.1 フィールド・ブロック法の評価

4.1.1 管理情報量の評価

図8は横軸が図形数を、縦軸が管理情報量と実体情報量を表わしている。 10K個の図形を扱った場合でも実体情報が2Mバイトなのに対して管理情報は20Kバイトで済む。 その比率は1%であるからほとんど無視できる管理情報量である。

4.1.2 図形抽出処理時間

図9は図面中に存在する図形数と、その中からチェックの対象となる図形を一個抽出するのに必要なCPU時間の関係を表わしている。 処理速度は図形数nに対して$O(n^{\frac{1}{2}})$程度の変化である。 1K個の図形の

中からの抽出でも0.022秒程であり、極めて高速に処理できることが分かる。

4.1.3 領域分割法との比較

0.6K個の図形を配置した図面を用い、領域分割法[5]とフィールド・ブロック法とで管理情報量と図形抽出時間がどう異なるかを実験した。表3がその結果である。フィールド・ブロック法は領域分割法に比べて図形抽出時間が約5倍、管理情報量が約13%で済むことが明らかとなった。これらのデータはフィールド・ブロック法が領域分割法と比べて優れていることを表している。

4.2 頂点法の評価

図9から2個の図形間の距離計算には、約0.002秒程度かかっている。また、30頂点程度の図形どうしが複雑にからみあっている場合でも、高々0.004秒程で図形間距離計算が行えることが分かった。

4.3 Spaceチェックの評価

2K個の図形が配置されている図面を使い、Space規則違反をおこす図形数とチェック時間の関係を調べた。図10のグラフから4,5個の図形と規則違反をおこしたとしても0.05秒程度でDRCが行えることが分かる。実用上ほとんど問題のない処理時間である。

5. 結言

インタラクティブDRC実現手段の一つを検討し、テスト・プログラムを作り評価した。今回検討したフィールド・ブロック法及び頂点法は大量の図形情報に関してかなり有効であり、これらの手法を組み合わせることにより高速なDRCが行えることが分かった。

参考文献

[1] S. Taylor,"Symbolic Layout",VLSI Design, March,1984

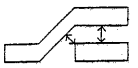
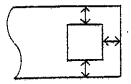
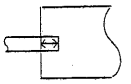
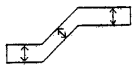
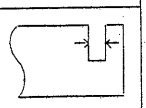
[2] J.A.Wilmore,"The Design of Efficient Data Base to Support an Interactive LSI Layout System, 16th DAC, June,1979

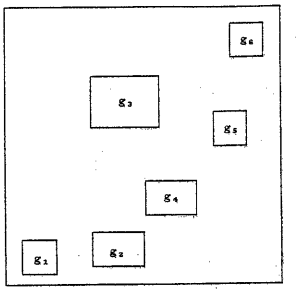
[3] 益田"ページング・マシンにおけるスワッピング・アルゴリズム",情報処理 Vol.13,No.2,1972

[4] 佐藤"VLSI設計における計算機何学の応用",Bit, Vol.17,No.4,1985

[5] 田中"プリント基板用設計ルール・チェック・プログラムの開発",第31回情処全大,1985

表1　設計規則の例

1	Space	
2	Enclosure	
3	Incursion	
4	Width	
5	Notch	



g_1～g_6：図形番号

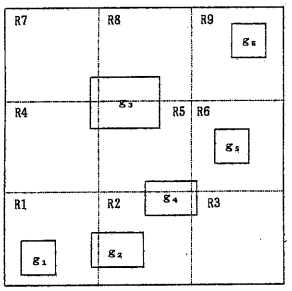
(a)　図　面

b_1	h_1	G_1	h_2	G_2	h_3	G_3
b_2	h_4	G_4	h_5	G_5	h_6	G_6

(b)　実体情報テーブル

b_1,b_2：ブロック番号
G_1-G_6：図形実体情報
h_1-h_6：ヘッダー
(Xmin,Ymin)　(Xmax,Ymax)

図1　例題図面



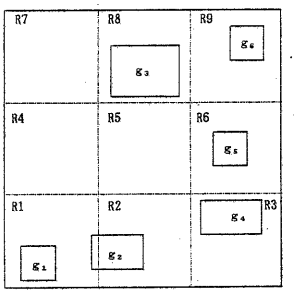
R1～R9：領域番号

(a)　図　面

R1	b1
R2	b1,b2
R3	b2
R4	b1
R5	b1,b2
R6	b2
R7	b1
R8	b1
R9	b2

b1,b2:ブロック番号

(b)　管理情報テーブル



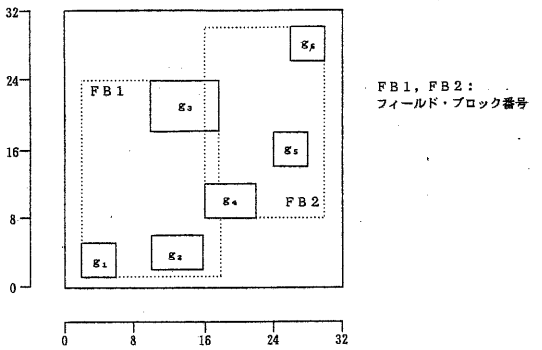
(c)　変更後の図面

R1	b1
R2	b1
R3	b2
R4	0
R5	0
R6	b2
R7	0
R8	b1
R9	b2

：変更個所

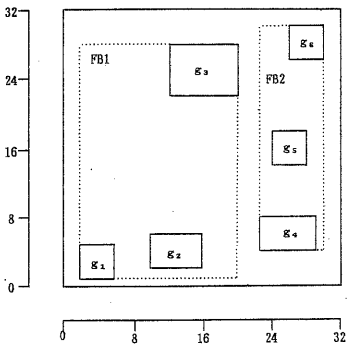
(d)　変更された管理情報テーブル

図2　領域分割法



FB1, FB2:
フィールド・ブロック番号

(a) 図 面



(c) 変更後の図面

b1	(1 , 1)	(18 , 24)
b2	(16 , 8)	(30 , 30)

b1,b2:ブロック番号

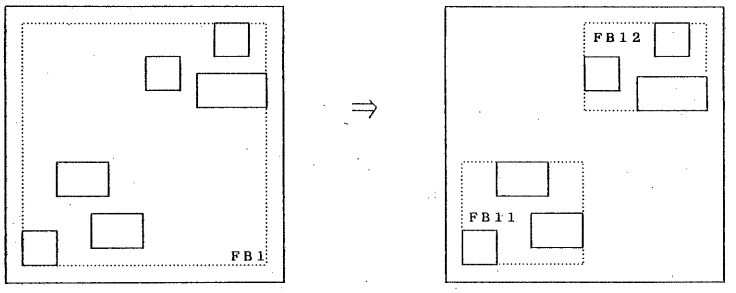
(b) 管理情報 テーブル

b1	(1 , 1)	(20 , 28)
b2	(22 , 4)	(30 , 30)

: 変更個所

(d) 変更された管理情報テーブル

図3 フィールド・ブロック法



(a) 分割前

(b) 分割後

図4. フィールド・ブロックの分割

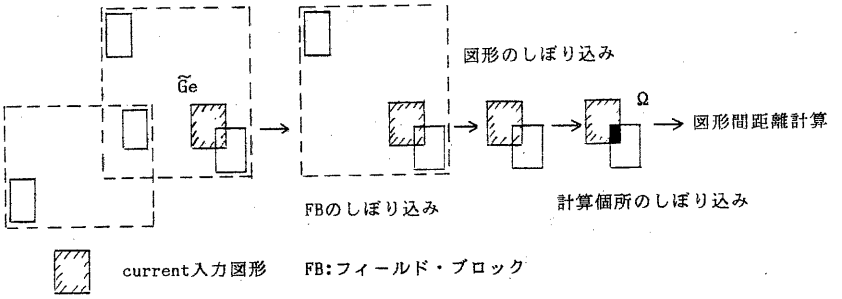
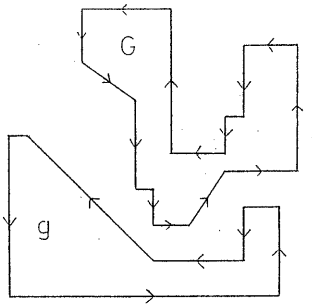
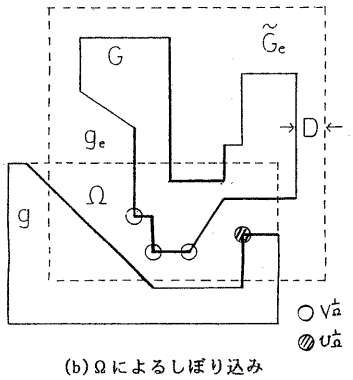


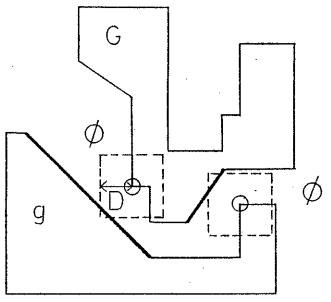
図5 DRC処理手順



(a)各辺をベクトル化

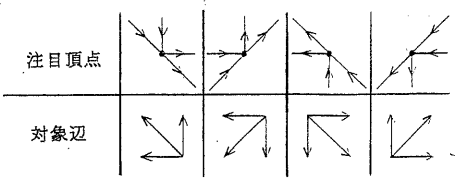


(b)Ωによるしぼり込み

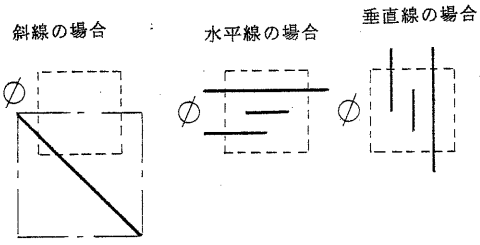


(c)φによるしぼり込み

図6 Spaceチェック手順



(a)辺の方向によるしぼり込み



(a) $X^{\phi}_{max} \geqq X^{\ell}_{min} \geqq X^{\phi}_{min}$ $Y^{\phi}_{max} \geqq y^{\ell}_{min} \geqq Y^{\phi}_{min}$

(b) $X^{\ell}_{max} \geqq X^{\phi}_{min} \geqq X^{\ell}_{min}$ $y^{\ell}_{max} \geqq Y^{\phi}_{min} \geqq y^{\ell}_{min}$

(c) $X^{\phi}_{max} \geqq X^{\ell}_{max} \geqq X^{\phi}_{min}$ $Y^{\phi}_{max} \geqq y^{\ell}_{min} \geqq Y^{\phi}_{min}$

(d) $X^{\ell}_{max} \geqq X^{\phi}_{max} \geqq X^{\ell}_{min}$ $y^{\ell}_{max} \geqq Y^{\phi}_{min} \geqq y^{\ell}_{min}$

$Y^{\phi}_{min} \leqq y^{\ell} \leqq Y^{\phi}_{max}$ $\quad X^{\phi}_{min} \leqq x^{\ell} \leqq X^{\phi}_{max}$

(a) $X^{\phi}_{max} \geqq X^{\ell}_{min} \geqq X^{\phi}_{min}$ (a) $Y^{\phi}_{max} \geqq y^{\ell}_{min} \geqq Y^{\phi}_{min}$

(b) $X^{\ell}_{max} \geqq X^{\phi}_{min} \geqq X^{\ell}_{min}$ (b) $y^{\ell}_{max} \geqq Y^{\phi}_{min} \geqq y^{\ell}_{min}$

該当辺の条件:

(a)or(b)or(c)or(d)

(b)φと辺の位置関係によるしぼり込み

図7 φ内の辺の抽出

表2　エッジ法と頂点法の比較

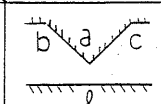
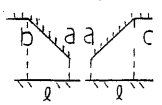
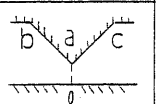
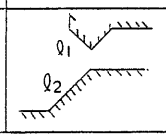
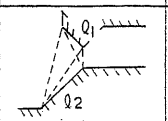
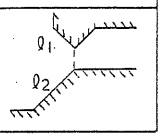
	ケース	エッジ法	頂点法
1			
2			

表3　領域分割法との比較

	管理情報量(KB)	処理時間(秒)
領域分割法	9.75	0.11
フィールド・ブロック法	1.28	0.02

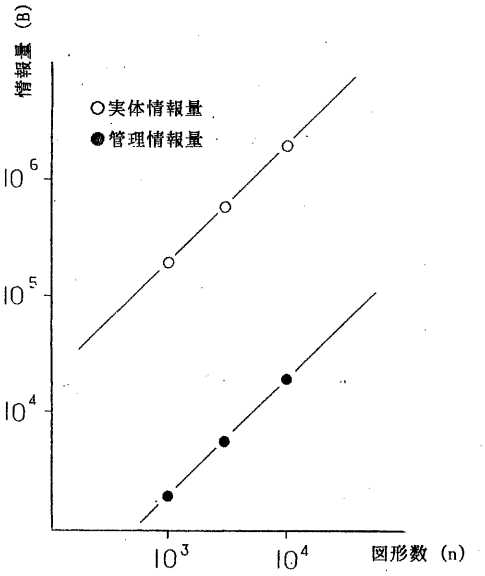


図8　実体情報量と管理情報量

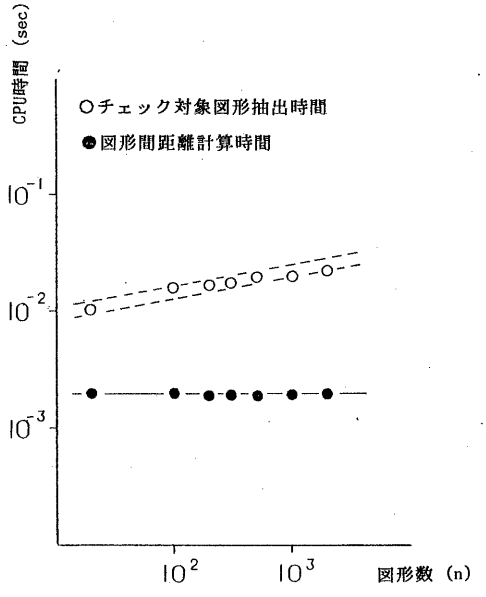


図9 図形抽出及び図形間距離計算の処理時間

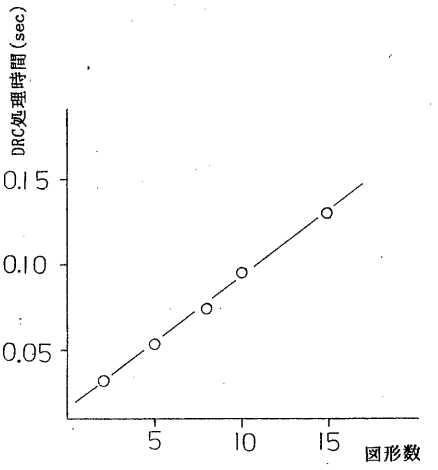


図10 設計規則違反図形数とDRC処理時間