アルゴリズム 33−13
(1993. 5. 29)

サーキットボード穴開け問題

味園 真司, 岩野 和生
日本IBM 東京基礎研究所

製造工場のサーキットボード穴開け問題という実社会の問題に取り組む機会に恵まれたので、その結果を報告する。この問題を巡回セールスマン問題としてとらえ、数種類の発見的方法を適用した。特に、疑似乱数生成技術を応用した新たな発見的方法を導入したのでそれを紹介する。また特殊な経路設計を必要とする穴開け問題を紹介する。

Circuit board drilling problem

Shinji Misono, Kazuo Iwano
IBM Research, Tokyo Research Laboratory, IBM Japan
1623-14 Shimo-tsuruma, Yamato-shi, Kanagawa-ken 242, Japan

We report our experience to solve real-world problem, a circuit board drilling problem which occurred at a manufacturing plant. We consider it as the traveling salesman problem, and tried several heuristics to TSP. We explain new heuristics which utilize pseudo-random number generation techniques. We also show real world examples in the circuit board drilling problem which require special drilling-route scheduling.

1 はじめに

生産工場で発生した実社会の問題に対して、計算機科学の理論研究を適用する機会に恵まれたので、その報告をする。今回取り組んだのは、回路基盤穴開け経路設計問題である。この問題は基本的には巡回セールスマン問題（TSP）であり、我々は既存及び新たな発見的方法でTSPの近似解を求めた。

次章でまず、この回路基盤穴開け経路設計問題を紹介し、TSPの発見的方法を用いた結果、つまり作業時間の短縮の度合を述べる。3節では、TSPを解く際に用いた様々な発見的方法を説明する。今回の経路設計問題には直接必要ではないが、TSPの発見的方法の性能評価には巡回路長の下限値を評価する必要がある。4節でHeldとKarpの下限値評価法の改良を試みたのでその結果を報告する。5節では、回路基盤穴開け経路設計問題で出てくる特殊な穴開け順序設計という問題を紹介する。

2 回路基盤穴開け経路設計問題

日本IBM野洲工場には、パーソナルコンピューターやハードディスク用の回路基盤に部品装着用の穴を開ける工程がある。現在この工程で処理する基盤数が非常に多いため、装置を２４時間稼働させて穴開け作業を行なっている。そのため穴開け作業時間を短縮させ、より多くの基盤を処理したいという要望があった。

穴開け作業は、回路基盤１２枚分に相当するパネルを単位に行なわれる。１枚のパネルには、直径の異なる１０種類程度の穴を開ける。種類毎の穴の数は、１０個から１００００個まで様々である。図1に１１２９個の穴の例を示す。

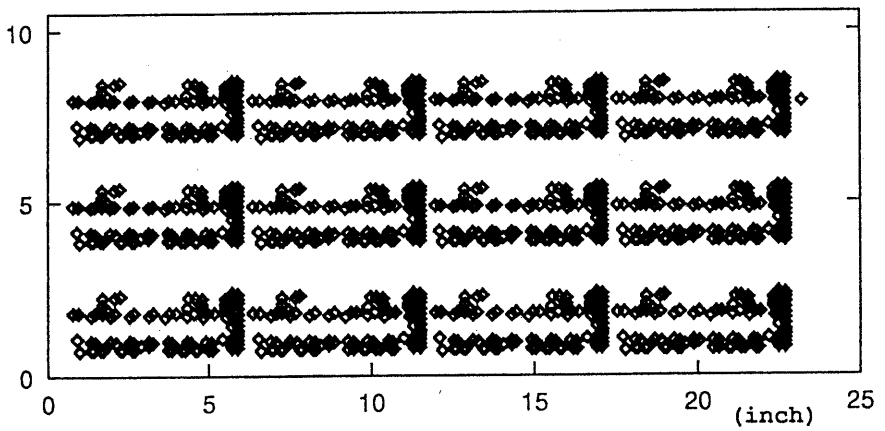

図 1: 回路基盤上の穴の配置例（穴の数１１２９）

穴開け作業は、まずパネルを穴開け装置に設置することから始まる。次に開けるべき穴の種類に応じた針をドリルに装着し、定められた位置にドリルを移動させては穴を開けていく。１種類の穴を開け終えると作業途中で針折れが生じていないかどうか調べる。針折れが生じていなければ針置き場に戻り、

ドリルの針を付け変えて次の種類の穴明けを開始する。針折れが生じていた場合には針を付け直した後、もう一度同じ種類の穴を開け直す。

現在穴開け作業には、1パネルあたり約１１０分かかっている。典型的なパネルを例にして作業別の所要時間の割合を見ると、パネル交換に５％、針の交換に７％、ドリルの垂直方向の動きである穴開けに７０％、そしてドリルの水平方向への動きである穴から穴への移動に１８％という具合になっている（ちなみにドリルの垂直方向移動速度は水平方向移動速度の約 1/100)。パネル交換、針交換そして穴開けの時間は固定されたものであるから、結局穴開け作業時間を短縮するには、穴から穴への移動距離を最短化する以外に方法がない。

これまで野洲工場では、次のような方法で経路を設計してきた。まずパネル上には同一の基盤が１２枚のっているので、１つの基盤について経路を求めた後、１２個の経路をつないでいく。１つの基盤については、穴の座標の辞書式順序で経路を決めていく。

ところが穴の種類毎にパネル全体で見ると、ドリルはまず針置き場から出発して、各穴を１度ずつ訪れ、再び針置き場に戻ってくる。つまり穴から穴への移動距離最短化問題とは、巡回セールスマン問題に他ならない（実際には5節で述べるように特殊な経路設計が必要になる場合がある)。ここで穴開け装置のドリル部の水平移動速度は x 軸、y 軸それぞれ独立でかつ等しい。そこで穴 (x_1, y_1) と穴 (x_2, y_2) の距離は、L_∞ 距離、つまり $\max(|x_1 - x_2|, |y_1 - y_2|)$ で計ることになる。

我々は次節で紹介する TSP の発見的方法を用いて移動経路の短縮化を試みた。図 2に従来の方法に対する我々の方法の短縮率を穴の数の関数として示した。この結果、穴開け作業の全体の時間を１０％強短縮することができた。

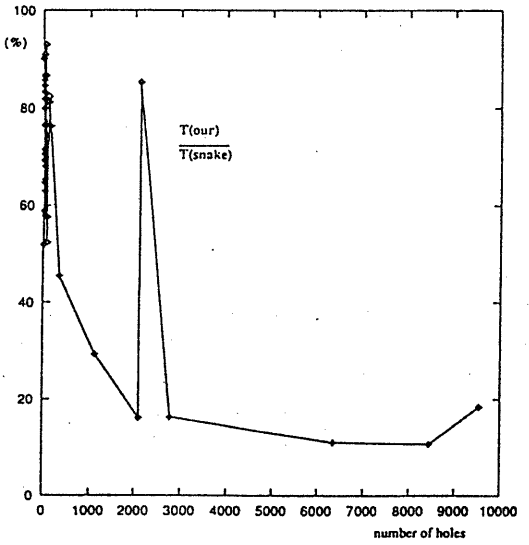

図 2: 従来手法（ snake ）の経路長 (T(snake)) と我々の方法の経路長 (T(our)) の比較

3 TSPの発見的方法

TSPの数多くの発見的方法の比較評価はBentley[1, 2]が精力的に行なっている。またシミュレーテッドアニーリング法も含めた評価をJohnson[5]が行なっている。我々の数値実験は小数の発見的方法に限られているため、彼らの結果には遠く及ばない。しかし均一な都市分布に対しての実験という従来の評価結果に対して、回路基盤の実際の例に対して評価という新たな情報を付与できるものと考えている。

ここではまず今回実験した既存の発見的方法を簡単に紹介する。既存のTSPの発見的方法は、巡回路生成に関するものと、初期巡回路から局所最適解を求めるものに分けられる。巡回路生成の発見的方法としてはつぎの4つを試みた。

Nearest addition 任意の1都市から出発して、まだ訪れていない都市の中で、現訪問地に最も近い都市を次の訪問地とする。全都市を訪れ終えたら、最後の訪問地と出発地とを結んで巡回路を完成させる。

Furtheset addition Nearest additionと反対に、未訪問の都市の中で、現訪問地から最も遠い都市を次の訪問地として、巡回路を決定していく。

Nearest insertion with random city selection まず2都市からなる巡回路を作る。これに都市を次々に加えて巡回路を成長させていく。巡回路を成長させる際、巡回路長の増加が最も少なくなるように順番に都市を加える。最初の2都市及び加える都市は、全都市からランダムに選び出す。

Multiple fragment 1都市からなるフラグメントの集合から出発して、最も近い2つのフラグメントを次々とつないでいく。この際ある都市の次数が2以上になったり、また部分的な巡回路が形成されないようにフラグメントの対を選ぶ。この操作をフラグメントが1つにまとまるまで続ける。つまり都市間の距離をソートしておき、短いものから順に巡回路の性質を壊さないものを加えていくgreedy methodに他ならない。

Bentleyの実験では、Multiple fragmentの方法が最も近似解の精度が良くまた計算時間も短いとの結果が出ている。

以上のような発見的方法で得られた巡回路を初期値として、そこからたどり着ける局所最適解を求めるのが、次の2-OPT, 3-OPTである。いずれも現在の巡回路の隣接巡回路を求め、もしもこの隣接巡回路の順路長の方が現在値よりも短ければ、これを新たな巡回路に採用する。この操作を、現在値よりも短い順路長の隣接巡回路が存在しなくなるまで続ける。

2-OPTと3-OPTでは、隣接巡回路の定義が異なる。

2-OPT 現巡回路の中の2辺 $A-B$、$C-D$、を $A-C$、$B-D$に変え、残りの部分を適当に反転したものを隣接巡回路とみなす。

3-OPT 現巡回路の中の3辺 $A-B$、$C-D$、$E-F$, を $A-E$、$D-B$, $C-F$, に変え、残りの部分を適当に反転したものを隣接巡回路とみなす。

いずれも現巡回路に対し一般に隣接巡回路は多数存在する。多数の候補の中から一つを選ぶ方法にも様々なものが考えられる。今回は最も原始的に（予めつけておいた）都市の番号順に隣接巡回路を探

索する方法を用いた。またBentleyらのように$K-d$ tree等のデータ構造を利用していないので、非常に時間のかかるものに留まっている。

Bentleyの実験結果では、巡回路生成発見的方法で得られる解は、厳密解より約２０％長い。これに2-OPTを適用すると１０％程度に、3-OPTでは６％程度にまで近似解の精度を改善することができる。このように局所最適化を行なえば、どの生成発見的方法の解よりも良い近似解が得られるが、局所最適化は時間がかかる。また精度の良い初期解ほど、局所最適化の結果も良い傾向にある。これらのことからできるだけ高速に、しかも質の良い巡回路を生成する発見的方法が望まれることになる。

我々は２種類の新たな生成発見的方法を実験した。その方法を説明する前に、これまでの生成発見的方法の問題点を考えてみる。Nearest addition, Furtherst addition, Nearest insertion with random city selectionともに局所的な情報しか使用していいない所に問題がある。例えばNearest additionでは、最初の内は距離的に近い都市を選び続けることができるが、最後の都市と最初の都市を結ぶ辺が極端に大きくなってしまったりしてしまう。

このような問題を避けるには、都市全体の分布という広域的な情報を利用する必要がある。その利用の一つの例が、Multiple fragment法であると言えよう。都市分布を辺の長さつまり都市間距離の分布を通じて利用している。今回我々は広域的情報を取り入れる発見的方法を２種類試みた。

Nearest insertion with uniform city selection 巡回路の生成の方法は従来のNearest insertion法と同じ。ただし以下に述べる方法で加えるべき都市を選択していく。まず全都市を辞書式順序に並べて１次元的番号（１～N Nは都市数）をつけておく。次にVan der Corput列[3]、つまり１次元区間上に動的な意味で一様な（ディスクレパンシーが最小な）点列をN-1個分生成し、その点列をソートして番号づけする。この番号に対応する都市をVan der Corput列の生成順に次々と巡回路に加えていく。

この方法では、本来２次元の都市分布を強引に１次元的尺度に変換してしまっいる。このため実際には近傍に位置する２都市が、辞書的順序では遠い位置であるように見なされてしまうという問題がある。

Mesh selection with Sobol sequence まず全都市を$\sqrt{N}$個のメッシュに分割する。次に２次元上に動的な意味で一様な点を分布させるSobol列[7]を$\sqrt{N}$個だけ生成させる。Sobol列の生成順に従ってメッシュを選択し、そのメッシュから代表都市を選択し合計$O(\sqrt{N})$個の都市を選び出す。この代表としについて局所最適化を施して小巡回路を作成する。再びSobol列の生成順にメッシュから残りの都市を選び出していき、この小巡回路をNearest insertionの方法で成長させていく。

この方法では２次元分布をメッシュ単位で管理し、その分布情報を均一に取り出す際にSobol列を利用している。以上２種類の発見的方法の性能評価を実データについて行なっている。

4 TSPの下限値評価

TSPの発見的方法の評価には巡回路長の下限値を計算しておく必要がある。代表的な下限値の評価方法としてHeldとKarp[4]によるものがある。今回Held-Karpの降下法にニ分探索方を組み合わせた評価法を試みた。

Held-Karpの降下法では，1-木を利用する。1-木とは、グラフ上の任意の１点を除いた点に対するスパニングツリーと、除いた１点から出る２本の辺からなる構造体である。最小コスト1-木を求める問

題はTSPの緩和問題であるから、1-木最小コストは常にTSPの解(C^*(TSP))以下である。また各点iにポテンシャルπ_iを与えた場合も、TSPの最適巡回路は不変であるから、k種類目の1-木のコスト（都市間距離の総和）をc_k, 1-木の点iの次数をd_{ik}とすると、

$$C^*(\mathrm{TSP}) \geq \min_k \{c_k + \sum_i \pi_i d_{ik}\} - 2\sum_i \pi \tag{1}$$

が成り立つ。よって

$$w(\pi) = \min_k c_k + \sum_i \pi_i (d_{ik} - 2) \tag{2}$$

とした時、ポテンシャルの中で$w(\pi)$を最大にするものを求めることで、TSPの下限を評価することができる。

Held-Karpの降下法は、ゼロポテンシャルから始めて、ポテンシャルを繰り返し変更していき、$\max_k w(\pi)$を求める。つまり、現ポテンシャルπ^mでの最小コスト1-木を求め、その時の点の次数から次のポテンシャルπ^{m+1}を

$$\pi_i^{m+1} = \pi_i^m + t^m (d_i^m - 2) \tag{3}$$

のように決定していく。ここでt^mは

$$t^m = \alpha \frac{(\tilde{w} - w(\pi^m))}{\sum_i (d_i^m - 2)^2} \tag{4}$$

($0 < \alpha < 2$)で与えられるスカラー量であり、$\tilde{w}$は$w(\pi^m)$以上のあるポテンシャルに対する最小コスト1-木のコストである[6]。以上のポテンシャル更新を、最小コスト1-木の次数が全て2になるか、あるいはコストが$\tilde{w}$を越えるまで繰り返す。

この降下法では$\tilde{w}$が入力パラメータとなっており、つぎの観察結果を得た。もしも$\tilde{w}$が$w(\pi^m)$に非常に近い場合、すぐに$\tilde{w}$を越える最小コスト1-木が見つかる。反対にUBとして大きな値、たとえばTSPの発見的方法の値、を入れた場合にはポテンシャルが毎回大きく変化するために、降下法の収束が非常に悪くなる。

以上の観察に基づき、$\tilde{w}$を二分探索的に決定する方法を試みている。まずTSPの発見的方法の結果を$\tilde{w}$の明白な上限、ゼロポテンシャルに対する最小コスト1-木のコストが明白な下限とする。次に上・下限の中点を$\tilde{w}$とし降下法を行ない、ある一定期間に収束するか動かを調べる。もしもこの上限値を越える値に収束した場合には、ここを新たな下限値にする。それ以外の場合にはこの値と下限値との中点を新たな$\tilde{w}$とする。この下限値と上限値がある一定の精度以下に近付いた段階で探索を終了する。

予備的な実験では、この方法は従来の方法よりも同じ時間内ではより精度の高い下限値を見つけている。現在より詳しい実験を行なっている。

5 特殊経路設計問題

基盤の穴開けには特殊な経路設計を要するものがある。互いに重なり合う複数の穴を連ねて直線上の穴を開ける場合である。例えば図3(b)のように5個の穴（左から順に1、2、…、5と番号を付けることにする）を開けることで、図3(a)のような直線上の穴を作る場合を考える。

この時、1 → 2 → 3の順に穴を開けていくと、穴2を開ける時、図3(c)のように針の左側は既に開けられた穴1と重なるため、針にかかる圧力が左右で異なってしまう。このように左右非対称な圧

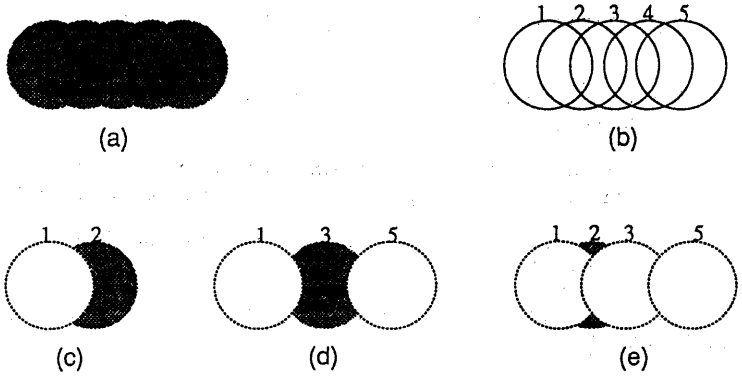

図 3: ステップドリル (a) 開けるべき穴の形状 (b) (a) を開けるのに用いる 5 個の針穴 (c) 針折れを起こす穴の開け方 (d) ステップドリル第三段階 (e) ステップドリル第四段階

力がかかると針が高い確率で折れてしまう。針折れが生じると針の交換が必要となるが、針交換にはかなりの時間がかかってしまう。そのため、針折れを最小限に抑えるようにする必要がある。つまり、常に針圧の非対称さが閾値以下になるように穴開け順序を決めなければいけない。上の例でいえば、$1 \rightarrow 5 \rightarrow 3 \rightarrow 2 \rightarrow 4$ とすれば良い。こうすれば、穴 3 と穴 2 を開ける場合、図 3(d)(e) のように針に左右対称な負荷がかかることになる。

このような特殊な穴開け方法をエンジニアはステップドリルと呼び、手作業で設計しているのが現状である。この設計には数時間をようするので、針圧の非対称さを最小化するアルゴルズムの開発が望まれている。またこのような特殊な経路を含みつつ、全体の経路長を最短化することも次の課題として取り組むべきであろう。

6 むすび

今回取り組んだ実問題、基盤穴開け問題、の一つは TSP そのものであった。そこで実際の都市分布に対して、いろいろな発見的方法の効果を確かめることができた。また基盤穴開け問題実問題にはステップドリルのような新しい問題も含まれていた。我々は理論的にも広がりのある面白い問題を発掘すべく、現在実問題に積極的に取り組んでいる。

参考文献

[1] J. J. Bentley, Experiments on traveling salesman heuristics, pp.91–99 in *First Annual ACM–SIAM Symposium on Discrete Algorithms*, San Fransicsco, CA, 1990.

[2] J. J. Bentley, Fast algorithms for geometric traveling salesman problems, *ORSA Journal on Computing*, 4, pp.387–411, 1992.

[3] J. G. van der Corput, *Verteilungstunktionen I, II*, Nederal. Akad. Wetensh. Proc. Ser. B, 38, pp.813–821, 1058–1066, 1935.

[4] M. Held and R. M. Karp, The traveling-salesman problem and minimum spanning trees, *Mathematical Programming*, 1, pp.6–25, 1971.

[5] D. S. Johnson, Local optimization and the traveling salesman problem, pp.446-461 in *Proceedings of the Seventeenth Coloquium on Automata, Languages and Programming*, Springer–Verlag, New York, 1990.

[6] E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooy Kan and D. B. Shimoys, *The Traveling Salesman Problem*, John Wiley & Sons, New York, 1985.

[7] I. M. Sobol', The distribution of points in a cube and the approximate evaluation of integrals, *Zh. Vychisl. Mat. Fiz.*, 7, pp.784–802, 1967.