Heuristics For Backplane Ordering

James P. Cohoon
Department of Computer Science
University of Virginia

Sartaj Sahni
Department of Computer Science
University of Minnesota

Computer Science Report No. TR-85-16
August 7, 1985

Heuristics For Backplane Ordering[†]

James P. Cohoon
Department of Computer Science
University of Virginia

Sartaj Sahni
Department of Computer Science
University of Minnesota

Abstract: The Board Permutation Problem, a backplane ordering problem, has been previously shown to be NP-hard. We develop here several heuristics for the Board Permutation Problem. These heuristics produce solutions that are locally optimal with respect to some nontrivial transforms. The heuristics are analytically shown to be $m/3$-approximate, where m is the number of nets in a problem instance. The heuristics have been shown experimentally to have quite acceptable behavior. Several of the heuristics make use of a Statistical Mechanics technique (simulated annealing) for thermal equilibrium analysis in producing their solution.

Keywords and Phrases: board permutation, backplane wiring, design automation, heuristics, local optimality, transforms, statistical mechanics and thermal equilibrium.

† - This research was supported in part by NSF grant MCS 80-005856.

1. INTRODUCTION

Complex digital systems are often decomposed into functional units that are individually designed and implemented. The result is a collection of boards that when properly interconnected serve as the desired digital system. Prior to wiring the interconnections, the boards are arranged in some linear arrangement or permutation. Besides the interconnections required to directly implement the nets, other interconnections are then introduced. For example, if a connection must cross a board or boards that are not involved in the net then a terminal must be placed on each of these intermediary boards to allow the signal to pass through.

The permutation of boards together with all their interconnections is a *backplane*. The size of the backplane is a function of both the size of the individual boards and the interconnections between the boards. If the boards themselves have been constructed to minimize their size then the backplane area may be minimized by minimizing the space required for the interconnections. The goal of the *Board Permutation problem* or *BP* is to determine a permutation of the boards that minimizes the backplane area. The input to the BP problem is a hypergraph (B,L), where B is set of n boards $\{b_1, \ldots, b_n\}$ and L is a set of m nets $L=\{N_1, \ldots, N_m\}$ on B. In Figure 1.a a BP instance is given. There are eight boards and five nets in the example. In Figure 1.b a permutation π of B is given. The arcs in the figure are graphic representations of the nets in L.

There are several methods for minimizing the backplane area. One method is to find a permutation that minimizes the maximum number of interconnections among the boards. The premise for this method is that each interconnection requires space, and by minimizing the total number of interconnections, the backplane area is minimized [ADOL72, RUTM64, SANG75, STEI61].

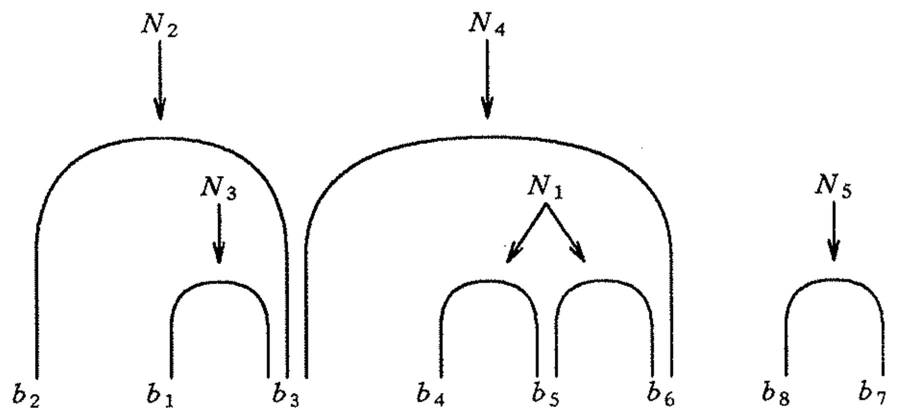
Another method of minimizing the backplane area is explored here. Before proceeding a definition must be given. The $density(\pi)$, where π is a permutation of B, is

$$\max_{1 \leqslant i \leqslant n-1} \left\| \left(\bigcup_{j=1}^{i} S_{\pi_j} \right) \cap \left(\bigcup_{j=i+1}^{n} S_{\pi_j} \right) \right\| .$$

where $S_i=\{a \mid a \in L, b_i \in a\}$, $1 \leqslant i \leqslant n$. Informally, the $density(\pi)$ is the maximum number of interconnections between any two adjacent boards in π. Cederbaum [CEDE74] showed that minimizing the interconnection density is equivalent to minimizing the backplane area.

$$\begin{aligned}
B &= \{b_1,b_2,b_3,b_4,b_5,b_6,b_7,b_8\} \\
L &= \{N_1,N_2,N_3,N_4,N_5\} \\
N_1 &= \{b_4,b_5,b_6\} \\
N_2 &= \{b_2,b_3\} \\
N_3 &= \{b_1,b_3\} \\
N_4 &= \{b_3,b_6\} \\
N_5 &= \{b_7,b_8\}
\end{aligned}$$

(a) - BP Instance



(b) - Permutation For Above BP Instance

Figure 1

Therefore, to minimize the backplane area it suffices to choose a permutation π from Π whose density is minimal (Π is the set of all permutations of B) This method of minimizing the backplane area has also been previously examined by Goto et al. [GOTO77] and it is the subject of further exploration here.

Given the above terminology and remarks, the BP problem may be formally expressed:

Input: A set of boards $B=\{b_1,\ldots,b_n\}$; a set of nets $L=\{N_1,\ldots,N_m\}$ on B such that $N_i \subseteq B$, $1 \leqslant i \leqslant m$ and $\bigcup_{i=1}^{m} N_i = B$

Output: A permutation π of B such that $density(\pi)=\min_{\omega\in\Pi}\left|density(\omega)\right|$

In Figure 1.b the permutation has density 2. The density of 2 occurs in several places in the permutation, for example between boards b_4 and b_5. An examination of the problem instance shows that this permutation has optimal density.

The algorithms that are developed here and elsewhere to minimize the backplane area by minimizing the maximum density of interconnection, have a variety of other applications. One application is in the *Gate Array Layout problem* or *GAL* [TODD82]. In the GAL problem the input is a set of cells, where a cell is some basic electrical and logical unit. The cells are to be laid out and interconnected in a regular fashion, typically in a matrix-like form. This structure, a *gate array*, may be built both economically and with fast turnaround. The gate array has both interconnections among the columns of cells and intraconnections within a column of cells. A part of the output solution is a gate array whose size is minimal. Algorithms for the BP problem may be applied to each individual column to order the column's cells in a manner that minimizes the impact of intra-column connections. The BP algorithms can then be applied on a higher hierarchical level to order the columns themselves in a manner that minimizes the impact of inter-column connections.

The BP problem is a generalization of the NP-Hard *Minimum Cut Linear Arrangement problem* or *MCLA* [GARE79]. MCLA restricts the size of the N_i's in L such that $|N_i|=2$, $1 \leqslant i \leqslant n$. Such a restriction reduces the input from a hypergraph to a graph. As any algorithm for BP is also an algorithm for MCLA, BP is NP-hard. Contemporary analysis strongly suggests that there is no deterministic, polynomial-time algorithm for any NP-hard problem. So, alternative methods of dealing with BP must then be explored. One alternative is to develop low-order, polynomial-time algorithms for special cases of BP problem. For example, we have developed two algorithms *Bpd1* and *Bpd2*, that determine whether the optimal density is 1 or 2 respectively [COHO83]. Another special case that can be considered is the reduced BP problem, MCLA. Though MCLA as stated earlier is NP-hard, Gurari and Sudburough have developed a dynamic programming algorithm for MCLA that determines whether there is a permutation with density k in $O(n^k)$ time [GURA81]. It is an open question whether their algorithm can be modified to produce solutions for the general BP problem in $O(n^{ck})$ time, where c is some constant.

The alternative explored here for the BP problem is a relaxation of the optimization criterion of the problem to accept a feasible solution whose value is *reasonably close* to the optimal solution's value. Algorithms that produce reasonably close solutions are called *heuristics*. An heuristic is *f(n)-approximate* if for all problem instances I

$$\left| \frac{F^*(I) - F'(I)}{F^*(I)} \right| \leqslant f(n)$$

where $F^*(I)$ is the value of an optimal solution for I, $F'(I)$ is the value of the solution generated by the heuristic, and n is a measure of the size of I. With the incorporation of *Bpd1* and *Bpd2*, every algorithm for BP is trivially $\frac{m}{3}$-approximate. Since solutions with density 1 and 2 are found by *Bpd1* and *Bpd2*, a lower limit for $F^*(I)$ not equal to $F'(I)$ is 3. An upper limit for $|F^*(I)-F'(I)|$ is m. Therefore, $\frac{m}{3}$ is an upper-bound for the ratio.

The first heuristic (Figure 2) we consider below is an $O(n^2m)$ heuristic developed by Goto et al. [GOTO77]. The heuristic, *Heuristic.1*, produces solutions that are locally optimal with respect to the transform

$$s(\pi,i,j)=\begin{cases}(\pi_1,\ldots,\pi_{i-1},\pi_j,\pi_i,\ldots,\pi_{j-1},\pi_{j+1},\ldots,\pi_n) & 1\leqslant i<j\leqslant n\\ \pi & \text{otherwise}\end{cases}$$

(If i is 1 then $\pi_1,\ldots,\pi_{i-1}$ is null. Similarly, if j is n then $\pi_{j+1},\ldots,\pi_n$ is null.) Informally, a solution π produced by *Heuristic.1* has the property that if a board b has been assigned to the j^{th} position then bringing b forward to the i^{th} position, does not decrease the density.

Besides analyzing *Heuristic.1*, several new heuristics are proposed below to solve the general BP problem. These same heuristics are combined to produce several other heuristics. These new heuristics are named *Heuristic.2* through *Heuristic.13*. The new heuristics all require low-order, polynomial-time to execute. In addition, the heuristics, like *Heuristic.1*, generate solutions that are locally optimal with respect to some nontrivial transform. It is

```
1.  Algorithm Heuristic.1($\pi$)
2.      $T \leftarrow B$
3.      $R \leftarrow \emptyset$
4.      for $i \leftarrow 1$ to $n$ do
5.          choose $b$ from $T$ such that $b$ achieves
                $\min_{t \in T}\left(\left|\left(\bigcup_{i \in R} S_i \cup S_t\right) \cap \left(\bigcup_{i \in T-\{t\}} S_i\right)\right|\right)$
6.          $\pi_i \leftarrow b$
7.          $T \leftarrow T - \{b\}$
8.          $R \leftarrow R + \{b\}$
9.      end for
10. end
```

Figure 2 - *Heuristic.1*

also shown for the heuristics that, like *Heuristic.1*, the ratio $\frac{m}{3}$ can be achieved for arbitrarily large problem instances when *Bpd1* and *Bpd2* are used in conjunction with the heuristics. Although the heuristics do not achieve a worst case performance improvement over *Heuristic.1*, experimental results indicate that several of the heuristics perform in a superior fashion to *Heuristic.1*. In addition, we interpret the experimental results to produce a combined heuristic, *Heuristic.14*.

Several of our heuristics have roots in Statistical Mechanics. The BP problem is interpreted as a system with n degrees of freedom in low temperature thermal equilibrium. We then adapt the Metropolis algorithm [METR53] for approximate numerical solution of thermal systems to run with the heuristics. This technique has been similarly used by Kirkpatrick et al. [KIRK83] to achieve impressive results for the Traveling Salesperson Problem and several NP-hard design automation problems.

In comparing our heuristics, we only consider the quality of their solutions and their run-times. All of our heuristics use at most $O(n)$ space in addition to the space required for the input. This additional space is not considered significant in evaluating the heuristics.

2. HEURISTICS

Heuristic.1

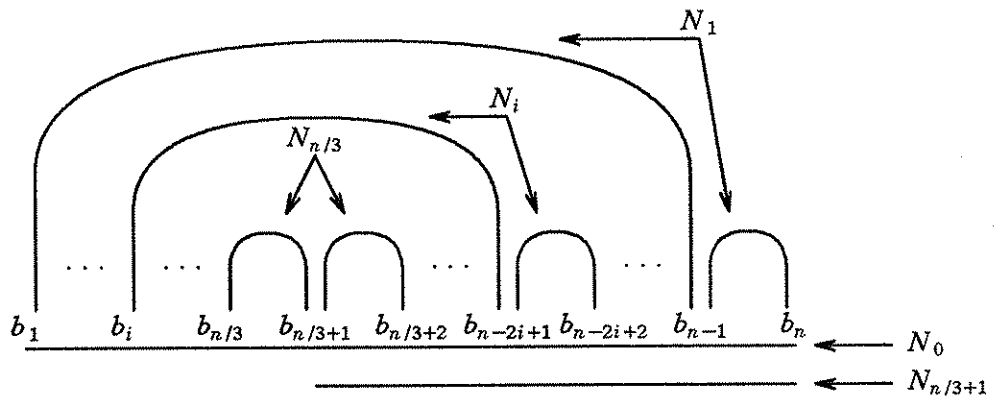
We show below that there exist arbitrarily large problem instances for *Heuristic.1* where it produces a permutation with density m when there exists an optimal permutation with density 3. Such a demonstration is sufficient to show that the heuristic is $\frac{m}{3}$-approximate as $\frac{m-3}{3} < \frac{m}{3}$.

In Figure 3.a, an instance is given that has the property that *Heuristic.1* may produce a permutation with density m. Such a permutation π is given in Figure 3.b. A permutation σ with density 3 exists, and is given in Figure 3.c. The permutation σ is optimal for the instance. For this instance n is a multiple of 3.

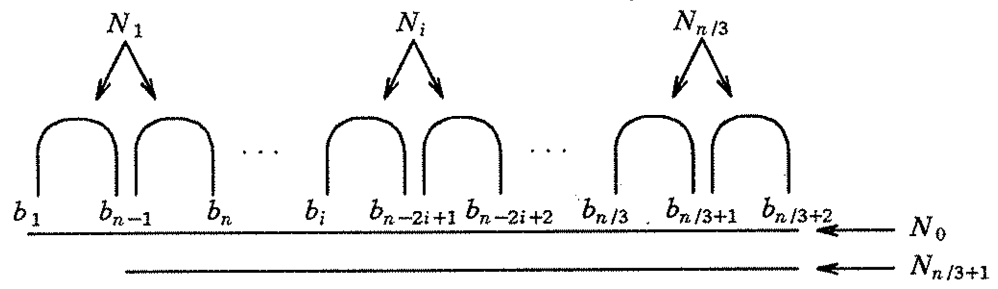
A demonstration that π is a valid output for the instance by *Heuristic.1* is straightforward. As all boards appear in at least 2 nets, the first board in π can be any board b where $|S_b|=2$, therefore π_1 can be assigned b_1. As the size of each net in S_{b_1} is greater than 2, and as all remaining boards must introduce at least one new net, any board that

$$\begin{aligned}
B &= \{b_1, \ldots, b_n\} \\
L &= \{N_0, \ldots, N_{\frac{n}{3}+1}\} \\
N_0 &= \{b_1, \ldots, b_n\} \\
N_i &= \{b_i, b_{n-2i+1}, b_{n-2i+2}\},\ 1 \leqslant i \leqslant \frac{n}{3} \\
N_{\frac{n}{3}+1} &= \{b_{\frac{n}{3}+1}, \ldots, b_n\}
\end{aligned}$$

(a) - BP Instance For *Heuristic.1*



(b) - Permutation With Worst Case Density For Above BP Instance



(c) - Permutation With Optimal Density For Above BP Instance

Figure 3

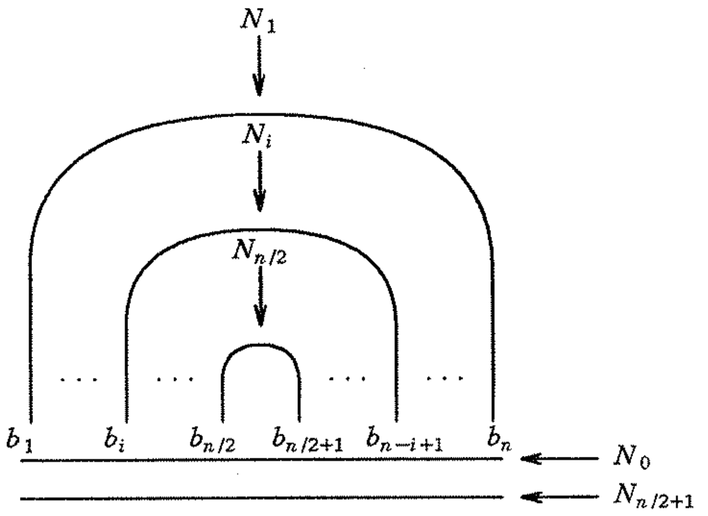
would introduce only one new net is valid for π_2. Therefore, π_2 can be assigned b_2. A similar examination for the remaining boards shows that they can be assigned as in Figure 3.b.

Heuristic.2

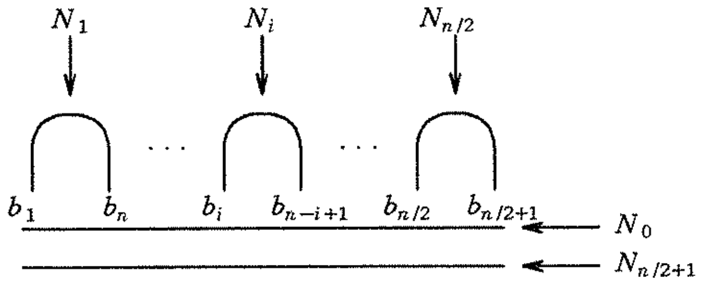
Heuristic.2 constructs its solution π *from the outside towards the middle* (i.e. the boards are assigned in the order $\pi_1, \pi_n, \pi_2 \pi_{n-1}, , \ldots, \pi_{\frac{n}{2}}$). The assignment is performed in a

$$
\begin{aligned}
B &= \{b_1, \ldots, b_n\} \\
L &= \{N_0, \ldots, N_{\frac{n}{2}+1}\} \\
N_0 &= \{b_1, \ldots, b_n\} \\
N_i &= \{b_i, b_{n-i+1}\},\ 1 \leqslant i \leqslant \frac{n}{2} \\
N_{\frac{n}{2}+1} &= \{b_1, \ldots, b_n\}
\end{aligned}
$$

(a) - BP Instance For *Heuristic.2*



(b) - Permutation With Worst Case Density For Above BP Instance



(c) - Permutation With Optimal Density For Above BP Instance

Figure 4

manner that ensures π is locally optimal with respect to the transform

$$t(\pi,i,j)=\begin{cases}(\pi_1,\ldots,\pi_{i-1},\pi_j,\pi_i,\ldots,\pi_{j-1},\pi_{j+1},\ldots,\pi_n) & 1\leqslant i<j\leqslant\lfloor\frac{n}{2}\rfloor\\(\pi_1,\ldots,\pi_{j-1},\pi_{j+1},\ldots,\pi_i,\pi_j,\pi_{i+1},\ldots,\pi_n) & \lfloor\frac{n}{2}\rfloor\leqslant j<i\leqslant n\\ \pi & \text{otherwise}\end{cases}$$

Informally, a solution π produced by *Heuristic.2* has the property that if a board in the first half is moved forward or a board in the second half is moved backward then the density does not decrease. A straight-forward implementation allows the heuristic to run in $O(n^2m)$ time.

Figure 4.a provides an instance that has the property that *Heuristic.2* may produce a permutation with density m as its solution. Such a permutation π is given in Figure 4.b. However, a permutation σ with optimal density 3 exists is given in Figure 4.c. Therefore, *Heuristic.2* is $\frac{m}{3}$-approximate.

Heuristic.3

Heuristic.3 constructs a solution π that is locally optimal with respect to the order τ that the boards are considered. The heuristic operates by placing the i^{th} board being considered in a position optimal with respect to the sub-solution generated by the $i-1$ previously placed boards. The heuristic is given in Figure 5. By calculating the density of $(\tau_i,\pi_1,\ldots,\pi_{i-1})$ and saving its $i-1$ individual densities at step 4 and then using the

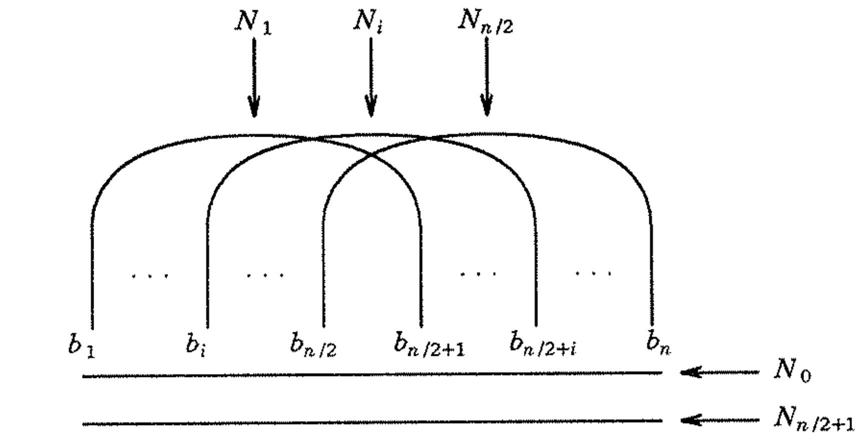
```
1.  Algorithm Heuristic.3($\pi$,$\tau$)
2.      for $i \leftarrow 1$ to $n$ do
3.          $\lambda \leftarrow (\tau_i,\pi_1,\ldots,\pi_{i-1})$
4.          cost $\leftarrow$ density($\lambda$)
5.          for $j \leftarrow i-1$ downto 1 do
6.              $\rho \leftarrow (\pi_1,\ldots,\pi_j,\tau_i,\pi_{j+1},\ldots,\pi_{i-1})$
7.              if cost $>$ density($\rho$) then
8.                  $\lambda \leftarrow \rho$
9.                  cost $\leftarrow$ density($\lambda$)
10.             end if
11.         end for
12.         $\pi \leftarrow \lambda$
13.     end for
14. end
```

Figure 5 - *Heuristic.3*

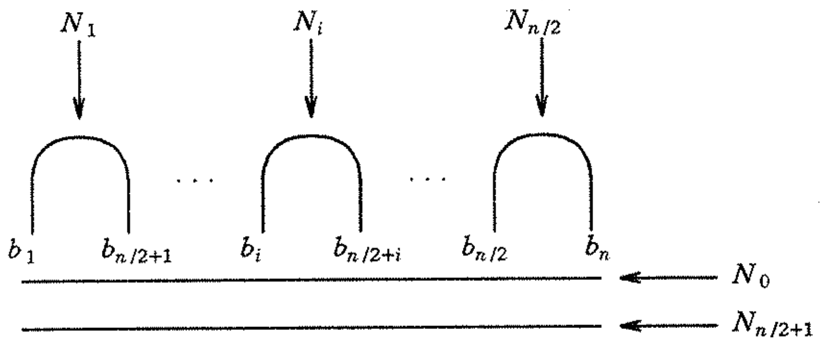
results for the next iteration of step 5, the time spent in determining the density of ρ at step 6 may be reduced to $O(m)$. Therefore, the total time required by *Heuristic.3* is $O(n^2m)$.

$$\begin{aligned}
B &= \{b_1, \ldots, b_n\} \\
L &= \{N_0, \ldots, N_{\frac{n}{2}+1}\} \\
N_0 &= \{b_1, \ldots, b_n\} \\
N_i &= \{b_i, \ldots, b_{\frac{n}{2}+i}\},\ 1 \leqslant i \leqslant \frac{n}{2} \\
N_{\frac{n}{2}+1} &= \{b_1, \ldots, b_n\}
\end{aligned}$$

(a) - BP Instance for *Heuristic.3*



(b) - Permutation With Worst Case Density For Above BP Instance



(c) - Permutation With Optimal Case Density For Above BP Instance

---

Figure 6

To specify a worst case instance for *Heuristic.3*, the definition of density must be augmented to include partial solutions. Below, a net i is not considered in calculating the density of a partial solution of size k if $\left|N_i \cap \{\pi_1, \ldots, \pi_k\}\right| \leqslant 1$.

The instance in Figure 6 demonstrates that *Heuristic.3* has arbitrarily large instances where a valid output π with density m may be produced when there exists an optimal permutation σ with density 3. If $\tau=(b_1, \ldots, b_n)$ then a valid output of *Heuristic.3* is $\pi=\tau$ as in Figure 6.b. As for any boards b_i and b_j, $i \neq j$, we have $S_i \cap S_j = \{N_0, N_{\frac{n}{2}+1}\}$, $1 \leqslant i, j \leqslant \frac{n}{2}$, $(\beta_1, \ldots, \beta_{\frac{n}{2}})$ is a valid partial solution for *Heuristic.3*. No matter where board $b_{\frac{n}{2}+1}$ is placed the density after its assignment will be 3. Thus $(\beta_1, \ldots, \beta_{\frac{n}{2}+1})$ is a valid partial solution for *Heuristic.3*. A similar examination shows that the remaining boards can be assigned as in Figure 6.b. Therefore, *Heuristic.3* has an arbitrarily large instance where it can produce a permutation with density m when there is an permutation σ, as in Figure 6.c, that has density 3.

Heuristic.4

Heuristic.4 examines its input permutation τ to see if it is possible to achieve a reduction in density by interchanging two boards in the permutation. The heuristic then repeats this process to see if two more boards can be interchanged to produce another density improvement. This pairwise interchanging is continued until no interchange of boards would result in a density improvement. By its method of operation *Heuristic.4*'s solutions are locally optimal with respect to the transform

$$u(\pi, i, j) = \begin{cases} (\pi_1, \ldots, \pi_{i-1}, \pi_j, \pi_{i+1}, \ldots, \pi_{j-1}, \pi_i, \pi_{j+1}, \ldots, \pi_n) & 1 \leqslant i < j \leqslant n \\ (\pi_1, \ldots, \pi_{j-1}, \pi_i, \pi_{j+1}, \ldots, \pi_{i-1}, \pi_j, \pi_{i+1}, \ldots, \pi_n) & 1 \leqslant j < i \leqslant n \\ \pi & \text{otherwise} \end{cases}$$

A straight-forward implementation of *Heuristic.4* allows its solution to be calculated in $O(n^3m^2)$ time.

In Figure 7.a, an instance is given that has the property that *Heuristic.4* can produce a permutation with density m. Such a permutation π is given in Figure 7.b. If π of Figure 7.b is also the permutation τ that is given to *Heuristic.4* to examine, then no improvement is possible with a single pairwise interchange. This follows as the maximal density in π occurs between boards $b_{\frac{n}{2}}$ and $b_{\frac{n}{2}+1}$. Hence, if an improvement is to be made it must be

$B=\{b_1,\ldots,b_n\}$

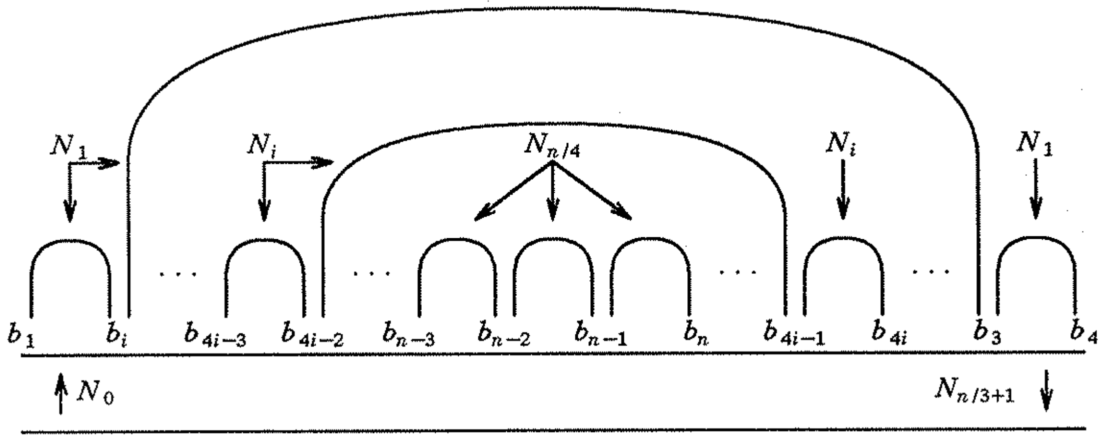
$N=\{N_0,\ldots,N_{\frac{n}{4}+1}\}$

$N_0=\{b_1,\ldots,b_n\}$

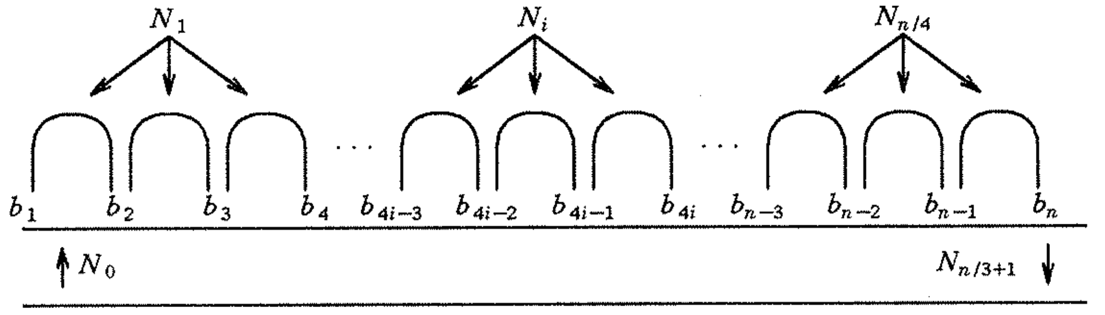
$N_i=N_{2i}=\{b_{4i-3},b_{4i-2},b_{4i-1,4i}\},\ 1\leqslant i\leqslant \frac{n}{4}$

$N_{\frac{n}{4}+1}=\{b_1,\ldots,b_n\}$

(a) - BP Instance For *Heuristic.4* And *Heuristic.9*



(b) - Permutation With Worst Case Density For Above Permutation



(c) - Permutation With Optimal Density For Above Permutation

Figure 7

done with a board b_i and a board b_j, where $1\leqslant i\leqslant \frac{n}{2}$ and $\frac{n}{2}+1\leqslant j\leqslant n$. But as all boards in each half of the permutation are part of a net with another board in the same half, moving a board to the other half of the permutation does not reduce the density. However, a permutation σ with optimal density 3 exists, and is given in Figure 7.c.

Therefore, *Heuristic.4* has an arbitrarily large instance where it may produce a solution with density m when the optimal solution has density 3.

Heuristic.5

Metropolis, et al. [METR53] developed a statistical mechanics algorithm that can be adapted to produce a heuristic for the BP problem. The Metropolis algorithm simulates the behavior of a thermal system in low temperature equilibrium. The behavior of such thermal systems is determined by the displacement of atoms over time. Through the use of random variates and a probabilistic model, the Metropolis algorithm successfully simulated the displacement of atoms. Their probabilistic model determines whether to accept as valid, a random variate that represents a change in the thermal energy. A random variate that represents a decrease in thermal energy is accepted with probability 1; a probability function $p(\cdot)$ is used to decide whether to accept a variate that represents an increase in thermal energy. The exact nature of the function p is dependent upon the thermal system being simulated. The Metropolis algorithm suggests a way that we might modify *Heuristic.4* to produce a better heuristic -- *Heuristic.5*. *Heuristic.5* will use a probability model for board interchanges in the same way the Metropolis algorithm uses a probability model for the displacement of atoms. If we allow the possibility of an interchange even though it may increase the density then we may be able to avoid a local optimum that is not a global optimum.

Heuristic.5 is formally presented in Figure 8. Its function *reject* is a boolean function that determines whether to perform a board interchange that would increase the density. The function *reject* operates in the following manner: draw a variate v from the uniform distribution on the interval $(0,1]$, if $v \leqslant \min\left\{\frac{density(\pi)}{(m+5)}, 0.9\right\}$ then perform some board interchange otherwise no more board interchanges are to be performed. The intuition behind *reject* is to favor continued board interchanges when the density is high with the greater expectation that there is a permutation with less density. Analysis of the relative merits of the function *reject* is deferred to the experimental analysis section.

As *Heuristic.5* always chooses a local optimum, it is locally optimal with respect to transform u. And as there is nonzero probability that a board interchange that would result in increased density is rejected, worst case BP instances for *Heuristic.4* are also worst case instances for *Heuristic.5*. Therefore, *Heuristic.5* is also $\frac{m}{3}$-approximate.

```
1.  Algorithm $Heuristic.5(\pi,\tau)$
2.      $\pi \leftarrow \tau$
3.      repeat
4.          $cost \leftarrow density(\tau)$
5.          repeat
6.              $\sigma \leftarrow \tau$
7.              $i \leftarrow 0$
8.              $min \leftarrow cost$
9.              repeat
10.                 $i \leftarrow i + 1$
11.                 $j \leftarrow i$
12.                 while $(j < n)$ and $(cost \geqslant min)$ do
13.                     $j \leftarrow j + 1$
14.                     $\tau \leftarrow interchange(\sigma,i,j)$
15.                     $cost \leftarrow density(\tau)$
16.                 end while
17.             until $(i = n)$ or $(cost < min)$
18.         until $cost \geqslant min$
19.         if $density(\sigma) < density(\pi)$ then
20.             $\pi \leftarrow \sigma$
21.         end if
22.         if $reject$ then
23.             return
24.         else
25.             Arbitrarily interchange the position of two boards in $\sigma$
26.             $\tau \leftarrow \sigma$
27.         end if
28.     forever
29. end
```

Figure 8 - *Heuristic.5*

Heuristic.6, Heuristic.7, and Heuristic.8

Heuristic.6, *Heuristic.7*, and *Heuristic.8* are each two-phase heuristics. The heuristics use respectively *Heuristic.1*, *Heuristic.2*, and *Heuristic.3* in their first phase to construct an initial solution. Their second phase uses *Heuristic.5* as a post-processor to improve the solution (if possible). As the solutions of *Heuristic.6* through *Heuristic.8* are last examined by *Heuristic.5*, the solutions are locally optimal with respect to transform u. The decision to use *Heuristic.5* in *Heuristic.6* through *Heuristic.8* instead of *Heuristic.4* is based on experimental analysis that is presented in a subsequent section.

Heuristic.9

Heuristic.9 operates in a manner somewhat similar to *Heuristic.4*. Whereas *Heuristic.4* attempted to interchange two boards to improve the density, *Heuristic.9* attempts to move one board to improve the density. Its solution is locally optimal with respect to the transform

$$v(\pi,i,j)=\begin{cases}(\pi_1,\ldots,\pi_{i-1},\pi_j,\pi_i,\ldots,\pi_{j-1},\pi_{j+1},\ldots,\pi_n) & 1\leqslant i<j\leqslant n\\ (\pi_1,\ldots,\pi_{j-1},\pi_{j+1},\ldots,\pi_i,\pi_j,\pi_{i+1},\ldots,\pi_n) & 1\leqslant j<i\leqslant n\\ \pi & \text{otherwise}\end{cases}$$

Informally, a solution π produced by *Heuristic.9* has the property that if s and t are boards in B, moving s so it is adjacent to t does not decrease the density. A straightforward implementation of *Heuristic.9* allows it solution to be calculated in $O(n^2m^2)$ time.

The instance in Figure 7 also serves to demonstrate that *Heuristic.9* has arbitrarily large instances where a valid output π with density m may be produced when there exists an optimal permutation σ with density 3.

Heuristic.10

Heuristic.10 (Figure 9) is an adaptation of *Heuristic.9* using the statistical mechanics techniques presented in conjunction with *Heuristic.5*. Like *Heuristic.5*, *Heuristic.10* chooses a locally optimum solution. However, *Heuristic.10* is locally optimal with respect to transform v. Also, as there in nonzero probability that no board is moved that would increase the density, the worst case BP instances for *Heuristic.9* are also worst case instances for *Heuristic.10*. Therefore, *Heuristic.10* is $\frac{m}{3}$-approximate.

Heuristic.11, Heuristic.12, and Heuristic.13

Heuristic.11, *Heuristic.12*, and *Heuristic.13* are each two-phase heuristics. The heuristics use respectively *Heuristic.1*, *Heuristic.2*, and *Heuristic.3* in their first phase to construct an initial solution. Their second phase uses *Heuristic.10* as a post-processor to improve the solution (if possible). As the solutions of *Heuristic.11* through *Heuristic.13* are last examined by *Heuristic.10* the solutions are locally optimal with respect to transform v. The decision to use *Heuristic.10* in *Heuristic.11* through *Heuristic.13* instead of *Heuristic.9* is based on experimental analysis that is presented in the next section.

```
Algorithm Heuristic.10($\pi$,$\tau$)
    $\pi \leftarrow \tau$
    repeat
        $cost \leftarrow density(\tau)$
        repeat
            $\sigma \leftarrow \tau$
            $min \leftarrow cost$
            $i \leftarrow 0$
            repeat
                $i \leftarrow i + 1$
                $j \leftarrow 0$
                repeat
                    $j \leftarrow j + 1$
                    $\tau \leftarrow insert(\sigma,i,j)$
                    $cost \leftarrow density(\tau)$
                until $(j = n)$ or $(cost < min)$
            until $(i = n)$ or $(cost < min)$
        until $cost \geqslant min$
        if $density(\sigma) < density(\pi)$ then
            $\pi \leftarrow \sigma$
        end if
        if reject then
            return
        else
            Arbitrarily move a board within $\sigma$
            $\tau \leftarrow \sigma$
        end if
    forever
end
```

Figure 9 - *Heuristic.10*

3. EXPERIMENTAL TESTING AND ANALYSIS

The experimental analysis and testing had several phases. The results of a given phase directed the testing of successive phases. The first phase was concerned with the placement of the function *reject* in *Heuristic.5* and *Heuristic.10*. The second phase measured the relative efficiency of function *reject*. The third phase compared *Heuristic.4* with *Heuristic.5* and *Heuristic.9* with *Heuristic.10*. From the test results of this phase, it was decided to use *Heuristic.5* in *Heuristic.6* through *Heuristic.8* and *Heuristic.10* in *Heuristic.11* through *Heuristic.13*. The fourth phase was a general comparison of all thirteen heuristics. The principal result of this phase was *Heuristic.14*. *Heuristic.14* is a combined heuristic that makes use of both *Heuristic.11* and *Heuristic.12*. The fifth and final phase of testing compared *Heuristic.14*'s solutions to the optimal solutions for a set of a problem instances.

We note that all of the testing performed for this section was consistent with the testing performed by Goto et al. [GOTO77].

Placement Of The Function Reject

The use of the function *reject* (Figure 10) in *Heuristic.5* need not be limited to only after a local optimum has been found. Rather, *reject* may be used after any application of the function *interchange*. In Figure 11, such a placement is shown with *Heuristic.5a*. Testing was conducted on *Heuristic.5* and *Heuristic.5a*. As a result of this experiment, the further use of *Heuristic.5a* was considered too expensive for our purposes. The testing consisted of several small BP instances, where n ranged from 5 to 8 and m was held constant at 10. The solutions for *Heuristic.5* were all computed in less than a second, while *Heuristic.5a* for the same instances required up to several hours. Though there was a slight improvement in one of the *Heuristic.5a*'s solutions over *Heuristic.5*'s solution, the time cost was deemed too expensive for further use. A similar examination of *Heuristic.10* with the application of *reject* after each board insertion showed a slight improvement in one of the trials, but again the time cost was deemed prohibitive.

Relative Efficiency of Function Reject

In the second phase of testing the relative efficiency of the function *reject* was measured. Two alternatives, *reject.1* and *reject.2*, were considered and they are given in Figure 12. The two alternatives introduced approximately a 20% increase and decrease, respectively, in the likelihood of accepting a board interchange that would increase the density. Let *Heuristic.5b* be the use of *Heuristic.5* with function *reject.1*. Let *Heuristic.5c* be the use of *Heuristic.5* with function *reject.2*.

```
1. Function $reject(\pi)$
2.     if $random \leq \min\left(\frac{density(\pi)}{m+5}, 0.9\right)$
3.             then true
4.             else false
5.     end if
6. end
```

Figure 10 - Function *reject*

```
Algorithm Heuristic.5a($\pi,\tau$)
    $\pi \leftarrow \tau$
    $cost \leftarrow density(\tau)$
    $mincost \leftarrow$ cost
    repeat
        $\sigma \leftarrow \tau$
        $i \leftarrow 0$
        $min \leftarrow cost$
        repeat
            $i \leftarrow i + 1$
            $j \leftarrow i$
            $interchanges \leftarrow$ false
            while ($j < n$) and (not $interchanges$) do
                $j \leftarrow j + 1$
                $\tau \leftarrow interchange(\sigma,i,j)$
                $cost \leftarrow density(\tau)$
                if ($cost < min$) or (not $reject$) then
                    $interchanges \leftarrow$ true
                    if $cost < mincost$ then
                        $mincost \leftarrow cost$
                        $\pi \leftarrow \tau$
                    end if
                end if
            end while
        until ($i = n$) or ($interchanges$)
    until not $interchanges$
end
```

Figure 11 - *Heuristic.5a*

```
1. Function reject.1(π)
2.       if random ≤ min$\left(\frac{0.8 \cdot density(\pi)}{m+5}, 0.9\right)$
3.             then true
4.             else false
5.       end if
6. end
```

```
1. Function reject.2(π)
2.       if random ≤ min$\left(\frac{1.2 \cdot density(\pi)}{m+5}, 0.9\right)$
3.             then true
4.             else false
5.       end if
6. end
```

Figure 12

Comparison of Heuristic.5 to Heuristic.5b and Heuristic.5c				
	Heuristic.5b		*Heuristic.5c*	
Number of Boards	Percentage of Cases with Increased Density	Percentage Time Decrease	Percentage of Cases with Reduced Density	Percentage Time Increase
10	8	36	12	54
15	8	30	16	63
20	20	28	20	52
25	8	25	16	48
30	8	25	4	38

Figure 13

The results of this experiment are summarized in the table of Figure 13. There were 25 BP instances randomly generated for each n in the table. Several values of m were used for each n. For a given BP instance each board had a constant probability p of whether it was assigned to a particular net. Several values of p were used for each n.

The approximate 20% decrease in probability of acceptance with *reject.1* had a minimum percentage decrease of 8% and a maximum percentage decrease of 20% in solution performance. Overall, 10% of *Heuristic.5b*'s solutions showed an increase in density over *Heuristic.5*'s solutions. The approximate 20% increase in probability of acceptance with *reject.2* had a minimum percentage increase of 4% at $n = 30$ in the number of instances with improved density when compared to *Heuristic.5* with function *reject*. The maximum percentage increase for *reject.2* was 20% and occurred at $n = 20$. Overall, 14% of *Heuristic.5c*'s solutions showed a decrease in density over *Heuristic.5c*'s solutions.

The table in Figure 13 also indicates the relative cost in time the two functions *reject.1* and *reject.2* have with respect to *reject*. The use of function *reject.1* with *Heuristic.5* had a minimum percentage time decrease of 25% at $n = 25$ and $n = 30$ when compared to *Heuristic.5* with *reject*. The maximum percentage time decrease with the use of *reject.1* was 36% at $n = 10$. Overall, *Heuristic.5b* used 26% less time to solve the set of BP instances than *Heuristic.5*. The use of function *reject.2* with *Heuristic.5* had a minimum percentage time increase of 38% at $n = 30$ when compared to *Heuristic.5* with *reject*. The maximum percentage time increase with the use of *reject.2* was 54% at $n = 10$. Overall, *Heuristic.5c* used 45% more time to solve the set of BP instances than *Heuristic.5*.

The results of the experiment with respect to the relative efficiency of *reject* to *reject.1* and *reject.2* indicate it is somewhat sensitive to scaling. Further experiments may be in order to determine whether *reject.2* or a reject function with an even greater scaling of *reject* should be considered. In successive testing of *Heuristic.5* with the other heuristics it was deemed that the use of the function *reject* was suitable as its solution quality and run-time characteristics were both good. However, arguments can be made for the use of either *reject.1* or *reject.2*.

Preliminary Heuristic Comparison

In this phase of testing *Heuristic.4*'s performance was compared to *Heuristic.5*'s performance and *Heuristic.9*'s performance was compared to *Heuristic.10*'s performance. The testing was performed in a similar manner to the testing of the previous phase. Several values of n were used and for each value of n, 25 BP instances were run. The BP instances were randomly generated with several values of m used for each n and the same instances were given to all of the heuristics. The results of this testing are summarized for *Heuristic.4* and *Heuristic.5* in the table of Figure 14.

An examination of Figure 14 shows that for most of the instances there was a significant improvement with respect to density in the performance of *Heuristic.5* over *Heuristic.4*. The minimum improvement was found for $n = 10$ and $n = 25$ where in 16% of the instances the density of the output permutation of *Heuristic.5* was an improvement

Comparison of Heuristic.4 and Heuristic.5		
Number of Boards	Percentage of *Heuristic.5*'s Solutions with Density Improvement over *Heuristic.4*'s Solutions	Percentage Increase In Running Time of *Heuristic.5* over *Heuristic.4*
10	16	178
15	24	118
20	28	122
25	16	91
30	24	87

Figure 14

over the density of the output permutation of *Heuristic.4*. The maximum improvement was found with $n = 20$ where in 28% of the instances the density of the output permutation of *Heuristic.5* was an improvement over the density of the output permutation of *Heuristic.4*. Overall, 22% of *Heuristic.5*'s solutions were an improvement over *Heuristic.4*'s solutions. This represents a significant performance increase.

The cost of these performance increases is also presented in Figure 14. The table shows that for $n = 10$ the run-time of *Heuristic.5* was almost triple that of *Heuristic.4*. However, for $n = 30$ there was approximately a doubling of run-time with *Heuristic.5*. Overall, the average run-time for a solution by *Heuristic.5* required 95% more time than the solution calculated by *Heuristic.4*.

Though the run-time costs are significant, in the interest of developing effective tools for the BP problem it was determined to use *Heuristic.5* as the post-processor in *Heuristic.6* through *Heuristic.8* rather than *Heuristic.4*. The testing of *Heuristic.9* and *Heuristic.10* was comparable to the testing of *Heuristic.4* and *Heuristic.5*. And in a similar manner it was decided to use *Heuristic.10* as a post-processor in *Heuristic.11* through *Heuristic.13*.

General Comparison of the Heuristics

In this phase of testing, *Heuristic.1* through *Heuristic.13* were extensively tested on a set of problem instances. From this testing we were able to select two heuristics to combine to produce a new heuristic, *Heuristic.14*. The testing was performed in a manner similar to previous phases. Several values of n were again used. For each value of n, 25 BP instances were randomly generated. Several values of m were used for each n. The instances were constructed such that a board b had a constant probability p in a given instance of whether it was assigned to a net. Several values of p were used for each n. For the purposes of this experiment, *Heuristic.3*, *Heuristic.4*, *Heuristic.5*, *Heuristic.8*, and *Heuristic.9* were each given 5 starting permutations per instance. For each starting permutation there was an associated output permutation. The heuristic returned the output solution with minimal density.

The table in Figure 15 displays the number of times a heuristic produces a solution that is minimal with respect to the solutions produced by the other heuristics. An examination of the table shows that except for $n = 5$, *Heuristic.11* consistently outperformed the other heuristics. At $n = 5$, *Heuristic.11* still had 22 of its solutions being minimal. Overall, *Heuristic.11* had 66% of its solutions being minimal. If because of limited resources one must choose a single heuristic for a BP instance then *Heuristic.11* appears to be a reasonable choice.

The table in Figure 16 displays for *Heuristic.1* through *Heuristic.13* both their maximum and average difference in density from the solution with the minimal density. The table gives further evidence of *Heuristic.11*'s superiority. *Heuristic.11*'s maximum difference in density from the minimal solution was minimal with respect to the other

Number Of Times In 25 trials A Heuristic Produced A Solution With Minimal Density With Respect To Other Heuristics							
Heuristic	$n = 5$	$n = 10$	$n = 15$	$n = 20$	$n = 25$	$n = 30$	Total
1	21	15	7	7	7	4	61
2	21	1	5	1	0	1	29
3	25	7	0	0	0	0	32
4	24	14	11	4	3	2	58
5	25	16	12	7	4	6	70
6	23	16	10	10	8	5	72
7	23	7	7	3	7	3	50
8	25	7	2	5	0	1	40
9	25	16	6	3	6	0	56
10	25	17	6	3	6	2	59
11	22	20	13	17	16	11	99
12	24	6	13	5	5	5	58
13	25	11	4	2	1	1	44

Figure 15

Density Differences		
Heuristic	Maximum Difference In Density From Minimal Solution	Average Difference In Density From Minimal Solution
1	8	1.26
2	9	2.69
3	12	3.73
4	5	1.41
5	5	1.00
6	5	0.99
7	6	1.53
8	7	1.94
9	6	1.48
10	5	1.30
11	5	0.65
12	6	1.33
13	10	2.22

Figure 16

heuristics. Its average difference in density was also minimal. The table also shows 3 of the heuristics were on average within 1 of the minimal solution and 9 of the heuristics were on average within 2 of the minimal solution. Thus, the other heuristics, though not as effective as *Heuristic.11*, do produce solutions whose density on average is quite close to the density produced by *Heuristic.11*.

Let *Heuristic.ij* be a combined heuristic that when given a BP instance it applies both *Heuristic.i* and *Heuristic.j* to the instance and returns the solution with minimal density, $1 \leqslant i, j \leqslant 13$. The table in Figure 17 displays the number of minimal solutions these combined heuristics achieved with the BP instances generated for the general comparison table of Figure 15. The combining of *Heuristic.11* and *Heuristic.12* had the most minimal solutions -- 121. The combining of *Heuristic.5*, *Heuristic.9*, or *Heuristic.10* with *Heuristic.11* would have come within 5 of the number of minimal solutions the combining of *Heuristic.11* and *Heuristic.12* would have achieved. As the combining of *Heuristic.11* and *Heuristic.12* did achieve the greatest number of minimal solutions for combined heuristics, this combination was selected for further testing. Let this combined heuristic be called *Heuristic.14*

Number Of Times in 150 Trials A Combined Heuristic Produced A Solution With Minimal Density With Respect to Other Combined Heuristics													
Combining Heuristics	1	2	3	4	5	6	7	8	9	10	11	12	13
1	61	69	67	84	94	72	81	73	84	86	99	87	74
2	69	29	39	65	75	80	50	47	62	65	105	58	51
3	67	39	32	60	71	76	54	40	58	61	103	62	44
4	84	65	60	58	70	91	77	65	77	79	112	84	65
5	94	75	71	70	70	98	86	76	86	88	120	94	75
6	72	80	76	91	98	72	88	82	92	94	105	95	82
7	81	50	54	77	86	88	50	61	71	74	115	68	65
8	73	47	40	65	76	82	61	40	64	67	108	70	48
9	84	62	58	77	86	92	71	64	56	59	116	78	66
10	86	65	61	79	88	94	74	67	59	59	118	81	68
11	99	105	103	112	120	105	115	108	116	118	99	121	109
12	87	58	62	84	94	95	68	70	78	81	121	58	74
13	74	51	44	65	75	82	65	48	66	68	109	74	44

Figure 17

Heuristic.14

As *Heuristic.14* is a combination of *Heuristic.11* and *Heuristic.12* and as both *Heuristic.11* and *Heuristic.12* are locally optimal with respect to transform v, so is *Heuristic.14*. The testing of *Heuristic.14* was similar to the previous testing. Thirty BP instances were randomly generated with several values of n and m. The assignment of boards to nets was performed in a manner as above.

Heuristic.14's Solution Among 30 Trials Compared To Optimal Density	
Difference In Density	Frequency
0	20
1	9
2	1

Figure 18

For the 30 generated instances, both the optimal density and the density of the solution constructed by *Heuristic.14* were determined. The table in Figure 18 summarizes the differences in the densities for these instances. In two thirds of the generated BP instances, *Heuristic.14*'s solutions were optimal. In the remaining instances, the difference in density from the optimal solution was no more than 2. These results indicate that *Heuristic.14* is an effective heuristic in practice. If because of limited resources one must choose a single combined heuristic for a BP instance then *Heuristic.14* appears to be a reasonable choice.

4. CONCLUSIONS

The Board Permutation problem is an important problem in the design automation of digital systems. As the Board Permutation problem has been previously shown to be NP-hard. it is unlikely that there are fast algorithms that always return an optimal permutation. We have shown however, that there are fast heuristics that are $\frac{m}{3}$-approximate. We have also shown that these heuristics are locally optimal with respect to some nontrivial transforms. Finally, we have experimentally shown that several of the heuristics typically perform quite well. In fact, one of the heuristics generated an optimal solution 66% of the time during testing.

5. BIBLIOGRAPHY

[ADOL72] D. Adolphson and T. C. Hu, Optimal Linear Ordering, *SIAM Journal of Applied Mathematics*, **25**(3), November 1972, pp. 403-423.

[CEDE74] I. Cederbaum, Optimal Backboard Ordering through the Shortest Path Algorithm, *IEEE Transactions on Circuits and Systems*, **CAS-21**(5), September 1974, pp. 626-632.

[COHO83] J. P. Cohoon and S. Sahni, Exact Algorithms for Special Cases of the Board Permutation Problem, *21st Allerton Conference on Communication, Control, and Computing*, 1983, pp. 246-255.

[GARE79] M. R. Garey and D. S. Johnson, *Computers And Intractability - A Guide To The Theory Of NP-Completeness*, W. H. Freeman and Co., San Francisco, CA, 1979.

[GOTO77] S. Goto, I. Cederbaum and B. S. Ting, Suboptimum Solution of the Back-Board Ordering with Channel Capacity Constraint, *IEEE Transactions on Circuits and Systems*, **CAS-24**(11), November 1977, pp. 645-652.

[GURA81] E. M. Gurari and I. H. Sudborough, Cutwidth Problems in Graphs, *19th Allerton Conference on Communication, Control, and Computing*, 1981, pp. 752-761.

[KIRK83] S. Kirkpatrick, C. D. Gelatt and M. P. Vecchi, Optimization by Simulated Annealing, *Science*, **220**(4598), May 13, 1983, pp. 671-680.

[METR53] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller and E. Teller, Equation of State Calculation by Fast Computing Machines, *Journal of Chemical Physics*, **21**(6), June 1953, pp. 1087-1092.

[RUTM64] R. A. Rutman, An Algorithm for Placement of Interconnected Elements Based on Minimum Wire Length, *Proceedings of Proc. IFIPS Spring Joint Computer Conference*, 1964, pp. 477-491.

[SANG75] A. Sangiovanni-Vincentelli and M. Santomauro, A Heuristic Guided Algorithm for Optimal Backboard Wiring, *13th Allerton Conference on Circuits and Systems*, 1975.

[STEI61] L. Steinberg, The Back Board Wiring Problem: A Placement Algorithm, *SIAM Review*, **3**(1), January 1961, pp. 37-50.

[TODD82] L. F. Todd, D. J. Gilbert, J. M. Hansen, R. J. Anderson, S. V. Pantula, A. K. Biyani and J. L. Barron, CGAL - A Multi Technology Gate Array Layout System, *19th Design Automation Conference Proceedings*, Las Vegas, Nevada, 1982, pp. 792-799.