Interchangeable Pin Routing with Application to Package Layout

Man-Fai Yu   Joel Darnauer   Wayne Wei-Ming Dai
Board of Studies in Computer Engineering
University of California, Santa Cruz, CA 95064

Abstract

Many practical routing problems such as BGA, PGA, pin redistribution and test fixture routing involve routing with interchangeable pins. These routing problems, especially package layout, are becoming more difficult to do manually due to increasing speed and I/O. Currently, no commercial or university router is available for this task. In this paper, we unify these different problems as instances of the Interchangeable Pin Routing (IPR) problem, which is NP-complete. By representing the solution space with flows in a triangulated routing network instead of grids, we developed a min-cost max-flow heuristic considering only the most important cuts in the design. The heuristic handles multiple layers, prerouted nets, and all-angle, octilinear or rectilinear wiring styles. Experiments show that the heuristic is very effective on most practical examples. It had been used to route industry designs with thousands of interchangeable pins.

1 Introduction

Package layout has been a missing link in design automation — it is done manually because packages have been simple enough and layout tools cannot produce all-angle, non-uniform width wires. Nowadays, array I/O packages like Ball Grid Arrays (BGA) and Pin Grid Arrays (PGA) are complex structures with multiple routing layers, power and ground planes and built-in decoupling capacitors. High I/O count and high-speed performance requirements make manual design more costly and time-consuming. Although some package routers exist[1, 2], they have a number of limitations that prevent them from being adopted by industry for practical use. This paper proposes a more general router that does not have these limitations and is suitable for a wide variety of applications, ranging from BGA and PGA routing to test probe card design.

In BGA or PGA package design, we wish to connect each chip pad to a single package pin, but we may not care which pad is connected to which pin. This same type of problem also occurs in the design of test fixtures, footprint escape patterns, and to some extent in routing the signals inside an ASIC to one of the available I/O pads on its periphery. Figure 1 shows a test probe card design. We want to connect all pins in set A to set B while avoiding all the obstacles in set O. We do not care which pin in B is connected to which pin in A as long as there is *some* pin connected.

Conventional routers cannot address this problem because they require a predefined pin assignment in the form of a netlist before the routing process can be started. In addition to adding an extra step to the design process, the choice of pin assignment is critical to the quality of the routing. Often a pin assignment is chosen that cannot be realized. Rather than performing pin assignment and routing as two separate steps, what is needed is a router that performs these two steps simultaneously.

Specific package routers like PGA or BGA routers[1, 2, 3, 4] have been developed that take advantage of special geometries and symmetries of their respective problems and the freedom of interchangeable pins. Although these methods are exceptionally efficient, they have a number of limitations:

Geometry dependent They rely on the symmetry of the arrays and rings to generate solutions and cannot cope with missing, skewed, off-grid or arbitrarily placed pads.

Routability guarantee These algorithms cannot

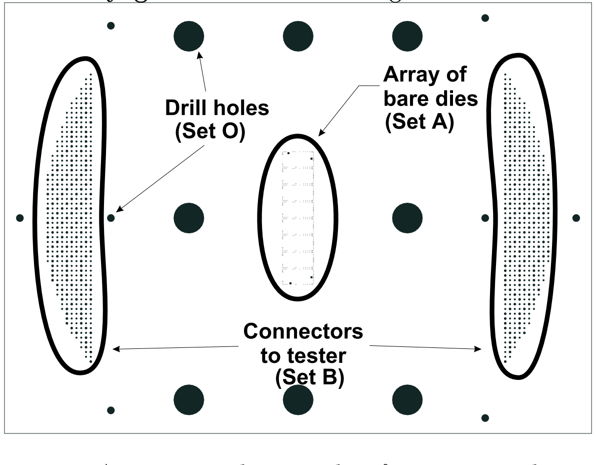

Figure 1: An unrouted example of routing with exchangeable pins. Each pad in the center must be connected to one pad on the edge connector.

ICCAD '96
1063-6757/96 $5.00 © 1996 IEEE

always tell if they have generated a routable solution, and have no strategy for changing an unroutable solution into a routable solution.

Unequal sets They require the two pin sets to be the same size. In actual packages the number of pins available may be larger than the number of chip I/O. The router should have the freedom to select which pins to use.

Obstacles They do not take into account the presence of obstacles.

Multilayer It is not easy to extend Yu and Dai's[1, 2] routers to multiple layers.

Prerouting These routers cannot accept prerouted wires. Prerouting is very important in package design because the designer wants to be able to route all the critical nets before other non-critical nets.

In response to these limitations, we propose a general problem formulation called *Planar Two-Terminal Interchangeable Pin Routing (P2TIR)*. Although we can take advantage of the freedom of interchangeable pins, P2TIR is NP-complete so no polynomial-time algorithm is likely to exist that guarantees routability[5]. Despite this, we offer an efficient heuristic which handles *arbitrary* pin placements, *unequal sets and obstacles*, can be extended to *multiple layers* and observes all *prerouting*. Experiments show that this heuristic, which we called the *flow router*, is very effective on practical examples, including some provided by the industry. We also showed that the router scales well to handle production-sized jobs from industry. The router successfully completed our largest example of 4000 interchangeable pins without manual intervention. No commercial or university router is capable of handling interchangeable pin problems of this size.

Cho and Sarrafzadeh[6, 7, 8] formulated the Pin Redistribution Problem for routing redistribution layers in Ceramic Multichip Modules (MCM-C). This problem on a single layer is similar to P2TIR except that all objects are on a fixed grid and all wires are Manhattan. Chang *et al*[9] uses a flow approach similar to ours for solving the Pin Redistribution Problem. Since the problem formulation is less general it is not easy to generalize the solution to other technologies. In addition, every grid cell corresponds to a node in their flow graph so the number of nodes in the graph is $O(m^2)$ where m is the dimension of the design. For a test probe card, the length of the card is in the order of inches and the chips to be tested has pad pitch in the order of several mils so grid-based approach is practically impossible. In this work we consider the most general case of *all-angle* wiring and our solution works for Euclidean, octilinear or rectilinear wiring styles. Our routing network is not based on grids but on triangulation, which scales linearly with the number of pins. Therefore our solution is more technology-independent and computationally more efficient.

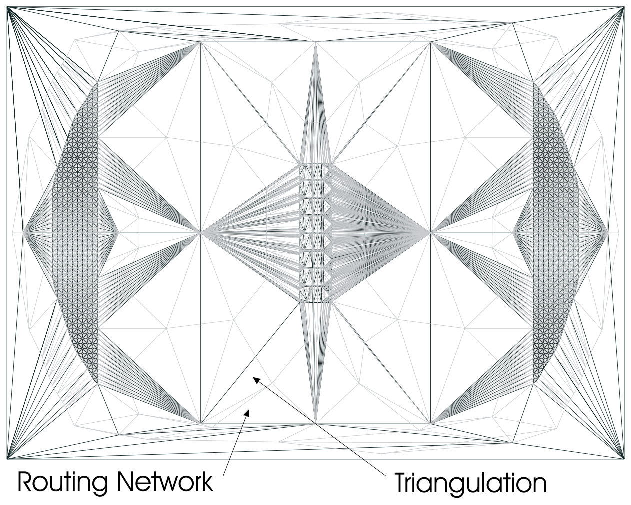

Figure 2: The routing network of the probe card

1.1 Formal Problem Definition

The fundamental features of an instance of P2TIR are two sets of pins placed arbitrarily in a plane that we wish to connect to one another. We also need to model the routing area and all obstacles, as well as the particular design rules permitted by the wiring. Accordingly, we define an instance of the Planar 2-Terminal Interchangeable Pin Routing (P2TIR) problem as follows:

Definition 1 Planar 2-Terminal Interchangeable Pin Routing

Instance *A 6-tuple of* (b, A, B, O, w, s) *where:*

b *is a polygon representing the routing area boundary.*

A *is a set of polygons for one class of pins.*

B *is a set of polygons for the other class of pins.*

O *is a set of polygons representing obstacles.*

w *is a positive integer for the minimum wire width.*

s *is a positive integer for the minimum wire spacing.*

A*,* B*,* O *are non-overlapping polygons inside* b*. Without loss of generality* $|A| \leq |B|$.

Output *A* detailed routing *of the design, i.e a set of wire paths that connects each pad in* A *to a unique pad in* B *avoiding all obstacles in* O *and obey the width and spacing rules.*

Maley[10] showed that the routability of topological routing (or homotopic routing) can be determined in polynomial time. A topological routing of a 2-terminal net is the equivalence class of all detailed routing of the net under homotopic transformation. A topological routing T is routable if and only if there exists a detailed routing in T that satisfies all design rules. Maley also showed that a topological routing is routable if the total number of wires flowing through straight cuts between any pair of features (the *flow of the cut*) is less than the maximum number of wires that could be accommodated in the best case (*the capacity*).

Because there are efficient algorithms for finding a correct detailed routing from a topological routing[11], we will consider a routable topological routing a solution for P2TIR.

2 Network flow formulation of P2TIR

2.1 The Routing Network

In the case of 2-terminal nets, we may recognize some similarities between a topological routing and a network flow. For example, a net can be modeled as a flow from a source (a terminal) to a sink (the other terminal). Flow is conserved at nodes that are neither sources nor sinks. Nets are also conserved because they only terminate at pins. Each source originate one unit of flow and each sink terminate one unit of flow. Similarly, each pin either originates or terminates a net. To make these ideas more concrete, we consider the *routing network* of a design.

The routing network $T(V, E, s, t)$ is a directed graph with a source s and a sink t. We first shrink each pin in sets A and B and each obstacle in O into a point. Each representing point has to be within the boundary of its object. Let these sets of points be $\tilde{A}$, $\tilde{B}$ and $\tilde{O}$ respectively. Then we build a Delaunay triangulation D on $\tilde{A} \cup \tilde{B} \cup \tilde{O} \cup b$, where b is the set of points of the bounding polygon. From the triangulation graph, we define T as follows:

Definition 2 *If Δ is the dual of D, a* Routing Network *$T(V, E, s, t)$ is a network where $V = \Delta \cup \tilde{A} \cup \tilde{B}$. E consists of the following arcs:*

- *A pair of opposite arcs for each edge in Δ.*
- *A pair of opposite arcs between each point in $\tilde{A} \cup \tilde{B}$ and each of its incident triangles.*

Each arc has a capacity and a non-negative cost.

The arcs originating from a point in $\tilde{A}$ (the source set) have unit capacities and arcs to a point in $\tilde{A}$ have zero capacities. The opposite is true for $\tilde{B}$ (the sink set). Finally, the pair of arcs connecting two triangles have capacities equal to the maximum number of wires that can cross between the corresponding pins or obstacles in both directions. The capacities should account for the finite size of pins or obstacles. The supersource s has an outgoing arc to each vertex in $\tilde{A}$ with capacity equal to 1 and cost equal to 0. Similarly, the supersink t has an incoming arc from each vertex in $\tilde{B}$.

Note that the vertices in Δ are triangles. Since there are many possible triangulations (besides Delaunay) for each instance of P2TIR, there are many possible routing networks for a given problem instance. Fig. 2 shows a routing network and a triangulation of the same probe card in Fig. 1. Each line segment in the figure represents a pair of opposite arcs in the network. Any flow assignment on this network has the following properties.

- The flow entering a vertex in $\tilde{A}$ must be -1.
- The flow entering a vertex in $\tilde{B}$ must be either 0 or $+1$. (Some pins in B may not be used.)
- The net flow entering a vertex in Δ must be 0. (Wires end at pads, not in triangles.)

We can show that flow assignment on the routing network of a design can be transformed into a topological routing of the design.

Theorem 1 *A flow assignment on a routing network of a design can be transformed into a topological routing of the same design.*

2.2 Routability of the topological routing

Maley[10] showed that only a certain set of cuts, the *decisive cutset*, needs to be checked for a topological routing to determine its routability. If there are N objects, we need to check at most $N(N-1)/2$ cuts. Thus, the number of cuts in the decisive cut set is bounded by $O(|A + B + O|^2)$.

In the routing network only *some but not all* cuts are represented as constraints. These cuts are the edges of the triangulation. There are only $O(N)$ cuts in a triangulation of N points while there are $O(N^2)$ cuts. It is obviously likely that many important cuts are not explicitly represented by the triangulation. We call these *implicit cuts.* A flow solution only guarantees that all explicit cuts are safe but says nothing about implicit cuts. We have shown that P2TIR is in fact NP-complete (by reduction from 3-SAT) so no polynomial time algorithm that finds a routable topological routing is likely to exist[5].

3 The Flow Router

In this section we describe in detail the **flow router** as a heuristic to solve P2TIR. We use the min-cost formulation. The router has three steps:

1. Building the routing network.
2. Solving the min-cost max-flow problem.
3. Transforming the solution into a topological routing.

3.1 Building the routing network

The Delaunay triangulation of $N = |\tilde{A} \cup \tilde{B} \cup \tilde{O}|$ points can be constructed in $O(N \log N)$ time[12]. The dual of the triangulation can be constructed in $O(N)$ time. Additional edges can be added in $O(|A \cup B|)$ time. The capacity of each edge is set as follows:

- If the edge is in the dual of the triangulation, then it represents a cut between two vertices. The

Algorithm 1 (BUILDUP)

```
Algorithm BUILDUP(Routing network $T$)
    for totalflow $\leftarrow$ 1 to $|A|$
        Path $p \leftarrow$ SHORTESTPATH($T$)
        if path is not found, return "$T$ is unroutable".
        Increment flow on all edges of $p$ by 1.
    endfor
```

Figure 3: Algorithm buildup

capacity of the edge is

$$\lfloor \frac{\text{Length of cut} - \text{Pad sizes} - \text{Wire spacing}}{\text{Wire spacing} + \text{Wire width}} \rfloor.$$

This is the number of wires that can intersect this cut without overflowing it.

- If the edge terminate at a pin, then the capacity is set to 1.
- The capacities of all edges of the supersink t and the supersource s are set to 1.

There is more flexibility in choosing the cost function. We choose the cost of an edge to approximate the wire length of the final topological routing. Since the position of a wire intersecting a cut is equally likely along the cut, we choose the edges that represent the cut in the routing network to be the perpendicular bisector of the cut. The intersection of the three perpendicular bisectors of a triangle is the center of the circumcircle of the triangle. We therefore define the cost of an edge to be the distance between the circumcenters where the edge terminate. Other points in the triangle, such as the centroid, can be used too. Experiments show that the solutions obtained by using the centroid and the circumcenter are not much different. This means that both are reasonably good estimators of real wire length.

3.2 The Min-cost Max-flow Algorithm

After the routing network is constructed, we run a min-cost max-flow algorithm on the network. The algorithm we used is based on the 'buildup' algorithm described in Papadimitrou and Steiglitz[13]. Informally, we try to find a minimum total cost assignment of flows for a given flow. In this case, the given flow is the maximum flow because the flow is equal to the number of connections, i.e. $|A|$. This flow is maximum because the sum of capacities of edges of the supersource is $|A|$. If we cannot push $|A|$ flow across the network, the design is unroutable. This is because a bottleneck of cuts in the triangulation has overflowed.

This algorithm requires a shortest path algorithm SHORTESTPATH that handles negative-cost edges. We used the algorithm described in Tarjan[14]. This algorithm runs in $O(|V||E|)$ time. Note that the flow on an edge can be negative. The run time of the flow assignment algorithm is $O(|V|^3)$ since the number of

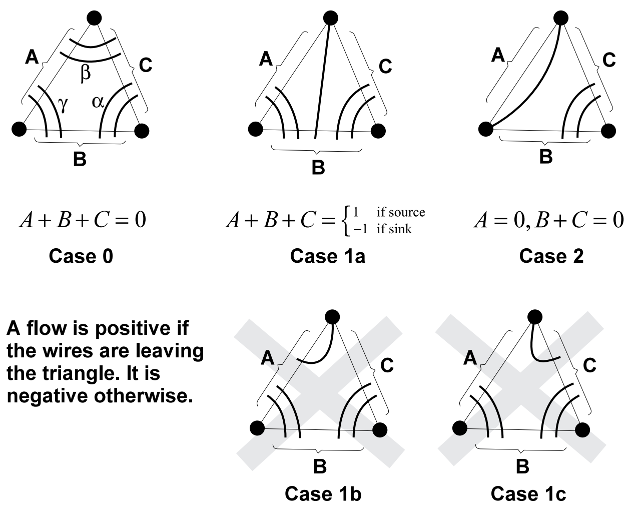
A
β
C
γ
α
B
$A+B+C=0$
Case 0
A
C
B
$A+B+C=\begin{cases}1 & \text{if source}\\ -1 & \text{if sink}\end{cases}$
Case 1a
A
C
B
$A=0, B+C=0$
Case 2
A flow is positive if the wires are leaving the triangle. It is negative otherwise.
A
C
B
Case 1b
A
C
B
Case 1c

Figure 4: Three cases of mapping flows in a triangle to a topological routing

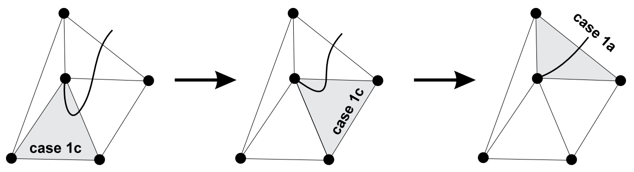

Figure 5: A Case 1c triangle transformed homotopically to a Case 1a triangle

edges in the triangulation is linearly proportional to the number of vertices, $|V| = |A \cup B \cup O|$. Fig. 3 shows the basic algorithm.

3.3 Transforming a flow solution to a topological routing

The last step in the flow router is to convert a min-cost flow solution to a topological routing. This can be done on a triangle-by-triangle basis. Fig. 4 shows the three possible cases of flow assignments on a triangle. Note that each edge of the triangle corresponds to a pair of opposite arcs in the routing network. In a min-cost flow solution, the flow of at least one arc in a pair has zero flow. Regardless of the direction of the flows across a cut, we can only have three cases. Case 0 has no connection to any of the pins in the triangle. Case 1 has one connection and Case 2 has two. Since a topological routing is actually an equivalence class of homotopically equivalent detailed routings, Case 1b and Case 1c are redundant. Fig. 5 shows a homotopic transformation of a detailed routing involving a Case 1c triangle to a routing that does not use Case 1b or Case 1c triangles.

In a case 0 triangle, we can compute the subflows α, β, γ from the flows A, B, C by noting $\gamma + \beta = |A|$, $\alpha + \gamma = |B|$ and $\beta + \alpha = |C|$. We can also show that α, β and γ can only be non-negative integers.

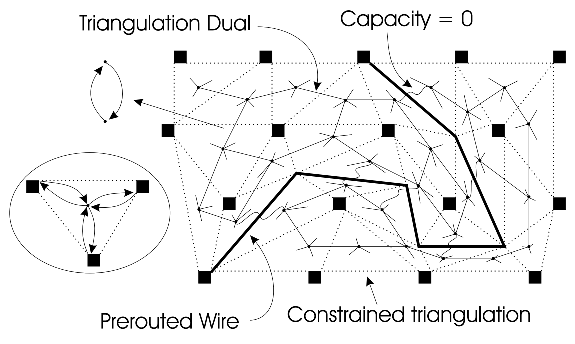

Figure 6: A prerouted wire as constrained edges and its (partial) routing network

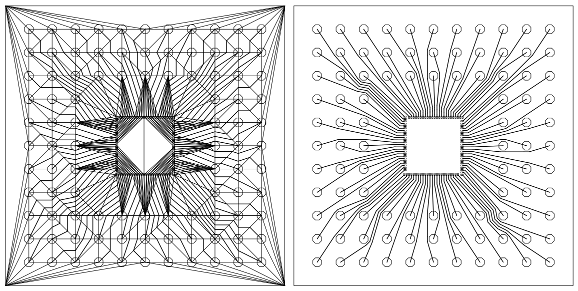

Figure 7: 96 pin PGA with the triangulation graph and wiring

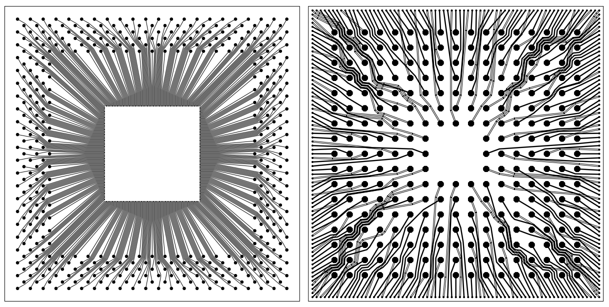

Figure 8: A 444 staggered pin PGA package and a two layer 280-pin BGA package

It is a simple matter to find the subflows in Case 1a and Case 2. Finally, we stitch the transformations of all triangles together to obtain the final topological routing.

3.4 Handling Prerouted Nets and Extension to Multiple Layers

We handle prerouted nets by constrained edges. We assume that each prerouted net is piecewise-linear. Then we embed each wire as a set of constrained edges into the triangulation. The capacities of corresponding routing network arcs are set to 0 so that wires cannot intersect each other. In effect the prerouted wires become barriers of flows. Since the prerouted wires are directly embedded into the triangulation, their exact shape affects the flow solution. Fig. 6 shows a prerouted wire embedded in the triangulation and the resulting dual.

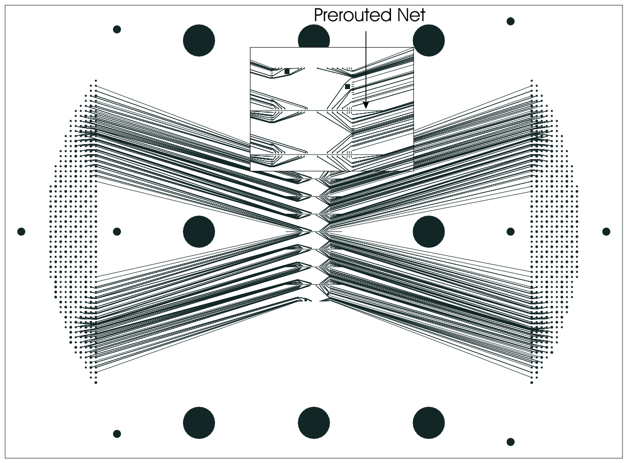

Figure 9: 400 connection test probe card with prerouted nets. These nets constrained the routing so that wires to each chip is grouped together.

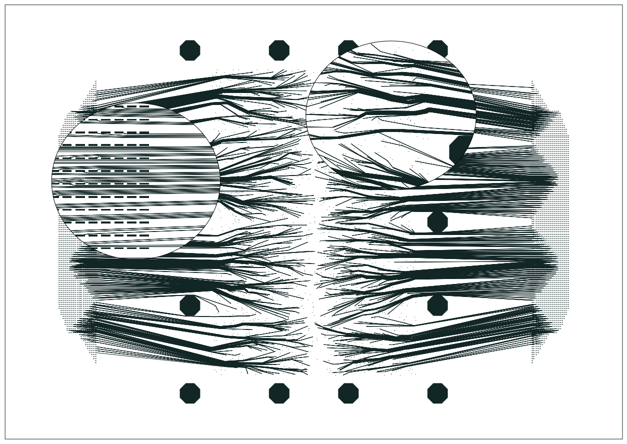

Figure 10: Two layer test probe card with 2148 randomly generated pins (one layer shown)

For multiple layers, we build triangulation and create the dual graphs on each layer as in the case of single layers. Then we combine these graphs together. Each pin is now a pin stack that spans contiguous, but not necessarily all, layers. So we extend the definition of the routing network as follows. We first choose a point to represent each pin stack. We call, as before, these sets of points $\tilde{A}, \tilde{B}$ and $\tilde{O}$ respectively for sets A, B and O. Let D_i be the triangulation of the ith layer, for $i = 1, \ldots, N$ layers and extend the definition of V in a routing network $T(V, E, s, t)$ to $V = \tilde{A} \cup \tilde{B} \cup \bigcup_{i=1}^{N} \Delta_i$, where Δ_i is the dual of D_i.

Note that each pin stack corresponds to only one (source or sink) vertex in the routing network. But each sink or source vertex has a pair of arcs to their incident triangles on *all* layers. The capacity of each cut is set according to the specific design rules of the layer the cut belongs to. Therefore each layer can have different design rules. The cost of each arc can also be adjusted so that some layer has higher cost per unit wire length than others. The min-cost algorithm will find a solution with minimum weighted wire length if one exists.

4 Experimental Results

Fig. 7 shows a small PGA with its triangulation and its routing done by the flow router. After routing, the intermediate points can be relaxed using methods proposed by Dai *et al.* [15].

Figure 8 is a single-layer 444 pin PGA package with staggered pins and a two layer 280 pin BGA package. Yu and Dai[1] cannot handle either case. The router automatically distributes the wires between the two layers and can accommodate different design rules of each layer. Figure 9 is a 400 pin test probe card with prerouted nets. The router connects chip I/Os at the center to two arrays of pads at the periphery. The flow router completed a 2148-pin, randomly-generated example on two layers with less than 10 design rule violations in 5 hours of CPU time on an HP9000 Model 735/99 workstation (Fig. 10). The router completed our largest example, a 4000 pin test probe card, with only 140 design rule violations.

5 Conclusion

A large number of diverse practical routing problems in ASIC, packaging and testing can be reduced to the Planar Two-terminal Interchangeable Pin Routing problem. We developed a min-cost flow router heuristic to solve this problem. The router was applied to solve an suite of routing problems in PGA, BGA and test probe cards. The router runs on multiple layers and handles prerouted nets. Experiments show that the heuristic is efficient in computing time and produced very good results, far better than what could be effectively done by hand.

6 Acknowledgement

C. K. Cheng and R. Bazylevych pointed out some previous work done by R. Bazylevych after the submission of this paper. Bazylevych *et al.* proposed a similar way of encoding topological routing[16, 17]. Although their context is not interchangeable pin routing but conventional routing, the idea of using a triangulation and recording the number of wires across the edges of the triangulation as an encoding is the same as presented in this paper.

References

[1] M.-F. Yu and W. W.-M. Dai, "Pin assignment and routing on a single-layer pin grid array," in *Proc. 1st Asia and South Pacific Design Automation Conf.*, (Makuhari, Japan), pp. 203–208, IEEE, August 1995.

[2] M.-F. Yu and W. W.-M. Dai, "Single-layer fanout routing and routability analysis for ball grid arrays," in *Proc. Intl. Conf. Computer-aided Design*, (San Jose, CA), pp. 581–586, IEEE, November 1995.

[3] J. Darnauer and W. W.-M. Dai, "Fast pad redistribution from periphery-IO to area-IO," in *Proc. IEEE Multichip Module Conf.*, (Santa Cruz, CA), pp. 38–43, March 1994.

[4] C. Ying and J. Gu, "Automated pin grid array package routing on multilayer ceramic substrates," *IEEE trans. VLSI Systems*, vol. 4, no. 1, pp. 571–575, 1993.

[5] M.-F. Yu, J. Darnauer, and W. W.-M. Dai, "Interchangeable pin routing with application to package layout," Tech. Report. UCSC-CRL-96-10, University of California, Santa Cruz, Santa Cruz, CA, April 1996.

[6] J. D. Cho and M. Sarrafzadeh, "The pin redistribution problem in multi-chip modules," in *Proc. 4th IEEE Intl. ASIC Conf. Exhib.*, (Rochester, NY), pp. P9–2.1–P9–2.4, IEEE, September 1991.

[7] J. D. Cho, K.-F. Liao, S. Rajie, and M. Sarrafzadeh, "M^2R: Multilayer routing algorithm for high-performance MCMs," *IEEE Trans. Circuits and Systems I*, vol. 41, pp. 253–265, April 1994.

[8] J. D. Cho and M. Sarrafzadeh, "An optimum pin redistribution for multichip modules," in *Proc. IEEE Multichip Module Conf.*, (Santa Cruz, CA), pp. 111–116, IEEE, Febuary 1996.

[9] D. Chang, T. F. Gonzalez, and O. H. Ibarra, "A flow based approach to the pin redistribution problem for multi-chip modules," in *Proc. 4th Great Lakes Symposium on VLSI*, (University of Notre Dame, IN), pp. 114–119, IEEE Computer Society, March 1994.

[10] F. M. Maley, *Single-layer wire routing and compaction.* Cambridge, MA: MIT Press, 1990.

[11] W. W.-M. Dai, R. Kong, and M. Sato, "Routability of a rubber-band sketch," in *Proc. 28th IEEE/ACM Design Automation Conf.*, (San Francisco, CA), pp. 45–48, IEEE Computer Society Press, 1991.

[12] F. P. Preparata and M. I. Shamos, *Computational Geometry: An Introduction.* New York: Springer-Verlag, 1985.

[13] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity.* Englewood Cliffs, NJ: Prentice-Hall, Inc, 1982.

[14] R. E. Tarjan, *Data Structures and Network Algorithms.* Phildelphia, PA.: SIAM Publications, 1983.

[15] D. Staepelaere, J. Jue, T. Dayan, and W. W.-M. Dai, "Surf: A rubber-band routing system for multichip modules," *IEEE Design and Test of Computers*, December 1993.

[16] R. P. Bazylevych, E. Zamora, and N. F. Storozenko, "The flexible routing algorithm for PCB," *Visnyk Lvivskoho Politekhnichnoho Instytutu*, vol. N76, pp. 83–88, 1973. In Ukrainian.

[17] R. P. Bazylevych, "Decomposition and topological methods for physical design automation for electronic devices," *Lviv: Vyshcha Shkola*, 1981. In Russian.