Network Flow Modeling for Escape Routing on Staggered Pin Arrays *

Pei-Ci Wu and Martin D. F. Wong
Department of Electrical and Computer Engineering
University of Illinois at Urbana-Champaign

Abstract— Recently staggered pin arrays are introduced for modern designs with high pin density. Although some studies have been done on escape routing for hexagonal arrays, the hexagonal array is only a special kind of staggered pin array. There exist other kinds of staggered pin arrays in current industrial designs, and the existing works cannot be extended to solve them. In this paper, we study the escape routing problem on staggered pin arrays. Network flow models are proposed to correctly model the capacity constraints of staggered pin arrays. Our models are guaranteed to find an escape routing satisfying the capacity constraints if there exists one. The correctness of these models lead to an optimal algorithm.

I. Introduction

Escape routing is an important problem in package and printed circuit board (PCB) design. Its purpose is to route from specific pins inside a pin array to the boundary of the array. According to the recent studies on PCB routing, escape routing can be further classified into three categories [7]: unordered escape, ordered escape, and simultaneous escape. These three types of escape routing problems all have many applications in package and PCB routing. The definition of the unordered escape routing problem is to route from pins inside one pin array to the boundary of the array without considering the pin ordering along the boundary. In this paper, we focus on the unordered escape routing problem.

The pin array discussed in the unordered escape routing problem usually refers to the traditional square pin array. The traditional square pin array is constructed by placing pins in a pin grid (see Fig. 1). However, since the constantly evolving technology continues to push the complexity of package and PCB design to a higher level, these complicated designs lead to very dense designs with high pin counts. Thus, staggered pin arrays are introduced to enable such designs with high pin density. A staggered pin array is constructed by shifting the specific columns of the traditional square pin array by a constant distance. The distance in the x-dimension between two adjacent columns of pins is a fixed value, while the distance in y-dimension between two adjacent rows of pins is another fixed value (see the left of Fig. 2). Moreover, Δx and Δy do not need to correlate with each other. We consider a *tile* as a diamond which is formed by four adjacent pins. The design rules then limit the number of wires between each of the boundaries, the horizontal diagonal, and the vertical diagonal in the tile. We call such constraints *boundary capacity*, *horizontally diagonal capacity*, and *vertically diagonal capacity*, respectively, or *B-cap*, *H-cap*, and *V-cap* for short (see the right of Fig. 2). Since Δx has no specific relationship with Δy, the values of *B-cap*, *H-cap*, and *V-cap* can be all different.

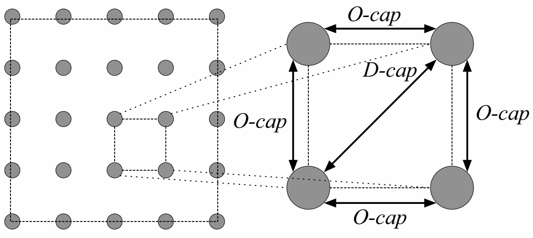


Fig. 1. An example of a traditional square pin array (left) and the enlarged view of a tile (right). *O-cap* and *D-cap* are two capacity constraints within a tile.

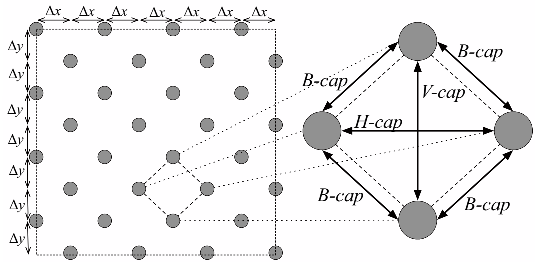


Fig. 2. An example of a 4 × 8 staggered pin array (left) and the enlarged view of a tile (right). Δx is the distance in the x-dimension between two adjacent columns of pins, and Δy is the distance in the y-dimension between two adjacent rows of pins.

Currently, escape routing problem for staggered pin arrays is solved manually in industry. Ho et al. [3] recently presented an escape routing algorithm for staggered pin arrays. Note that the staggered pin array focused on [3] is the hexagonal array, where ($2 \cdot$ *B-cap* $=$ $2 \cdot$ *H-cap* $=$ *V-cap*) or ($2 \cdot$ *B-cap* $=$ *H-cap* $=$ $2 \cdot$ *V-cap*). However, there are other kinds of staggered pin array in current industrial designs. According to the correlation between *B-cap*, *H-cap*, and *V-cap*, staggered pin arrays can be classified into three different types:

- Hexagonal arrays where $2 \cdot$ *B-cap* $=$ $2 \cdot$ *H-cap* $=$ *V-cap* or $2 \cdot$ *B-cap* $=$ *H-cap* $=$ $2 \cdot$ *V-cap* (see Fig. 3(a)).
- Rotated square pin arrays where *H-cap* $=$ *V-cap* (see Fig. 3(b)).
- Staggered pin arrays which do not belong to the two types described above. An example is as shown in Fig. 3(c).

The industrial data sets that we used in our experiment contain these three types of staggered pin arrays.

[*] This work was partially supported by the National Science Foundation under grant CCF-1017516.

978-1-4673-3030-5/13/$31.00 ©2013 IEEE

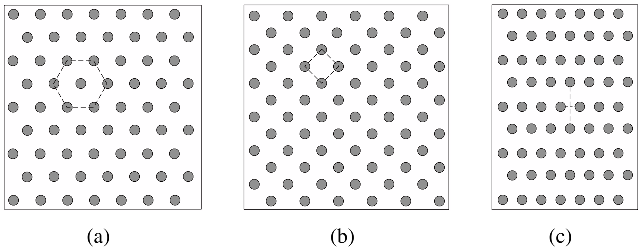


Fig. 3. Examples of three types of staggered pin array that are available in industrial PCB designs. (a) A hexagonal array where the dashed polygon is a hexagon; (b) a rotated square pin array where the dashed polygon is a rotated square; (c) a staggered pin array that is neither a hexagonal array nor a rotated square pin array. In this example, *H-cap* $= 1$ and *V-cap* $= 3$.

Given the capacity constraints, the escape routing problem is to find an escape routing satisfying the capacity constraints. The previous work for escape routing on traditional square pin arrays is to find an escape routing satisfying the orthogonal and diagonal capacity constraints defined by two adjacent pins ([6]). A valid escape routing must satisfy the capacity constraints, otherwise, the design rules are violated. Hence, for the escape routing problem on staggered pin arrays, the objective is to find an escape routing that satisfies the *B-cap*, *H-cap*, and *V-cap*constraints. Although [3] shows that for a hexagonal array even capacity constraints are met, the wire spacing between two adjacent tiles can be violated under some condition, such condition never exists for all the industrial boards that we have seen, which means satisfying capacity constraints is enough to guarantee a design rule correct escape routing for all those industrial boards. Note that we have to consider any two tiles rather than two adjacent tiles only when checking the condition. Therefore, in this paper, the escape routing problem we focus is to find an escape routing satisfying the capacity constraints in the staggered pin arrays.

Network flow is pervasively used to solve the unordered escape routing problem. The idea is to view each routing path as a unit flow from the pin to the boundary. Since no pin ordering is considered, a flow solution is able to be transformed into some non-crossing routing. Yan and Wong [6] proposed a network flow model to correctly model the capacities inside a square tile for the traditional square pin array. For the traditional square pins array, a tile contains two capacity constraints (*O-cap* and *D-cap* in the right of Fig. 1), so the proposed model can be rotated to solve rotated square pin arrays which contain two capacity constraints as well. However, it cannot model the other two types of staggered pin arrays. Ho et al. [3] used a network based linear programming formulation to solve escape routing problem on the hexagonal array, and the formulation cannot be extended to solve the other two types of staggered pin arrays. In summary, none of the previous works can completely solve the escape routing on staggered pin arrays. Particularly, none of them can handle the third type of staggered pin arrays (other than hexagonal arrays and rotated square pin arrays), since it does not exist a specific correlation between the capacity constraints, unlike the other two types of staggered pin arrays, which makes it the most difficult one to model. Network flow models proposed in this paper can solve all three types of staggered pin arrays.

In this paper, we introduce the problem formulation of escape routing on staggered pin arrays. Then network flow models are proposed to correctly model the capacity constraints of staggered pin arrays. Our models guarantee to give a legal routing if there exists one routing solution satisfying the *B-cap*, *H-cap*, and *V-cap* capacity constraints. An optimal escape routing algorithm is then built upon the models. Experimental results show that our algorithm has very short runtime.

The rest of this paper is organized as follows: Section II formulates the escape routing problem for staggered pin arrays. Section III presents our network flow models and escape routing algorithm. Experimental results are given in Section IV. Finally, Section V concludes the paper.

II. Problem Formulation

The input to the problem is an $m \times n$ staggered pin array with p pins specified as to-be-escaped pins. The definition of an $m \times n$ staggered pin array is as follows: each row has m pins and there is a total of n rows. Fig. 2 illustrates a 4×8 staggered pin array. *B-cap*, *H-cap*, and *V-cap* are given to specify the boundary capacity, horizontally diagonal capacity, and vertically diagonal capacity in a tile.

The expected output of the problem is an octilinear routing from the to-be-escaped pins to the boundary of the array satisfying the capacity constraints. We also would like the total length of the routing to be minimized.

The *B-cap* constraint limits the number of wires between one side of a tile which implies that at most $2 \cdot$*B-cap* wires can pass through a tile. Since the *H-cap* and *V-cap* constraints limit the number of wires between two diagonals of a tile, there are at most (*H-cap* + *V-cap*) wires that can put inside a tile. If *B-cap* is set such that $2 \cdot$ *B-cap* is larger than (*H-cap* + *V-cap*), there are only (*H-cap* + *V-cap*) wires allowed to pass through a tile. Therefore, it is natural that $2 \cdot$ *B-cap* $\leq$ *H-cap* + *V-cap* for all our inputs, and we call this the *capacity inequality*. The capacity constraints of the staggered pin arrays we have seen in industrial designs does satisfy the capacity inequality.

The capacity inequality is further analyzed as follows:

- Consider *H-cap* and *V-cap* are both even. Then,

$$\begin{aligned} & 2 \cdot \textit{B-cap} && \leq 2 \cdot \lfloor \textit{H-cap}/2 \rfloor + 2 \cdot \lfloor \textit{V-cap}/2 \rfloor \\ \Rightarrow \quad & \textit{B-cap} && \leq \lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor \end{aligned} \tag{1}$$

- Consider *H-cap* is odd and *V-cap* is even, and vice versa. Then,

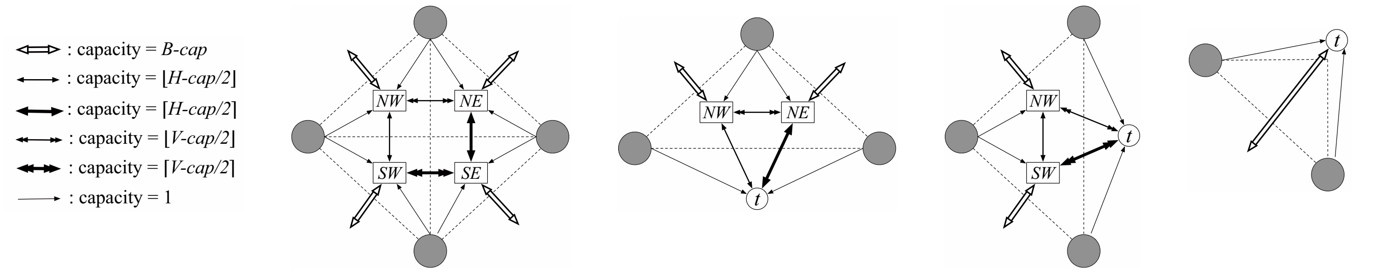
$$\begin{aligned} & 2 \cdot \textit{B-cap} && \leq 2 \cdot \lfloor \textit{H-cap}/2 \rfloor + 2 \cdot \lfloor \textit{V-cap}/2 \rfloor + 1 \\ \Rightarrow \quad & \textit{B-cap} && \leq \lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor + 1/2 \\ & && \leq \lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor \end{aligned} \tag{2}$$

- Consider *H-cap* and *V-cap* are both odd. Then

$$\begin{aligned} & 2 \cdot \textit{B-cap} && \leq 2 \cdot \lfloor \textit{H-cap}/2 \rfloor + 2 \cdot \lfloor \textit{V-cap}/2 \rfloor + 2 \\ \Rightarrow \quad & \textit{B-cap} && \leq \lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor + 1 \end{aligned} \tag{3}$$

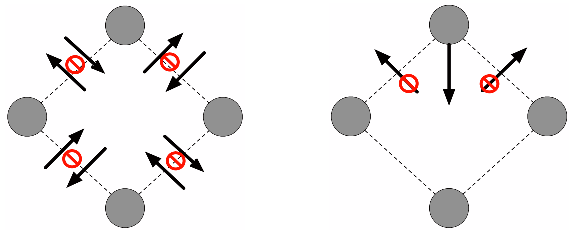
According to (1-3), for all our inputs, we can conclude that, only the inputs, where $2 \cdot$ *B-cap* is equal to (*H-cap* + *V-cap*) and both of *H-cap* and *V-cap* are odd,

$$\textit{B-cap} = \lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor + 1 \tag{4}$$



(a) (b) (c) (d) (e)

Fig. 4. Our network model for Equation (5). (a) The capacity of each edges; (b) a tile; (c) a $1/2$-tile along the bottom boundary of the array; (d) a $1/2$-tile along the right boundary of the tile; (e) a $1/4$-tile in the northeast corner of the array. t is the super sink.



(a) (b)

Fig. 5. The properties of a directed routing R with the minimum crossings with the tile boundaries. (a) Property 1; (b) Property 2.

is satisfied. Otherwise, the inputs satisfy

$$B\text{-}cap \leq \lfloor H\text{-}cap/2 \rfloor + \lfloor V\text{-}cap/2 \rfloor . \qquad (5)$$

III. Our Network Flow Models

In this section, we will first present a network flow model for Equation (5), and then another one for Equation (4) to cover all the staggered pin arrays.

A. Modeling Equation (5)

For each tile, we create four nodes, which are NW-node, NE-node, SE-node, and SW-node, respectively (see Fig. 4(b)). Bidirectional edges (which are realized by two directed edges: a forward edge and a backward edge) are created between the node pairs (NW, NE), (NE, SE), (SE, SW), and (SW, NW), and then give them capacity, $\lfloor V\text{-}cap/2 \rfloor$, $\lceil H\text{-}cap/2 \rceil$, $\lceil V\text{-}cap/2 \rceil$, and $\lfloor H\text{-}cap/2 \rfloor$, respectively. For connections between tiles, four directed edges are used to connect the NW-node of a tile to the SE-node of its northwest neighboring tile, the NE-node of a tile to the SW-node of its northeast neighboring tile, the SE-node of a tile to the NW-node of its southeast neighboring tile, and the SW-node of a tile to the NE-node of its southwest neighboring tile, respectively. We call such edges *inter-tile edges* and give each of them capacity *B-cap*. In order to escape from the to-be-escaped pins, we create a pin node for each pin, and then connect them to the nodes inside a tile. The pin node in the north corner of a tile is connected to the NW-node and NE-node, the pin node in the east corner of a tile is connected to the NE-node and SE-node, and the pin node in the south corner of a tile is connected to the SW-node and SE-node, and finally the pin node in the west corner of a pin is connected to the NW-node and SW-node. All these edges have capacity 1.

Although we define a tile as a diamond formed by four adjacent pins, we can see those tiles along the boundary of the array can only contain two or three adjacent pins. We call a boundary tile with three pins as a $1/2$-*tile*, and a boundary tile with only two pins as a $1/4$-*tile*.

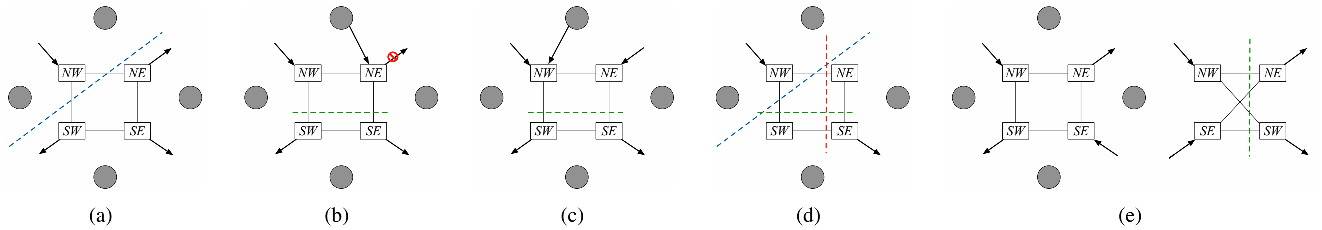
For the $1/2$-tiles along the bottom boundary, each of them has two nodes, which are the NW-node and NE-node, respectively (see Fig. 4(c)). Bidirectional edges with capacity $\lfloor V\text{-}cap/2 \rfloor$ are created between the NW-node and NE-node, and the NW-node and NE-node both are connected to the outside of the array by bidirectional edges with capacity $\lfloor H\text{-}cap/2 \rfloor$ and $\lceil H\text{-}cap/2 \rceil$, respectively. A $1/2$-tile along the right boundary is as shown in Fig. 4(d). Then the construction of the $1/2$-tiles along the top(left) boundary is symmetrical to that of the $1/2$-tiles along the bottom(right) boundary. Finally, the edges connecting a $1/2$-tile to the adjacent tiles and connecting the pin nodes to other nodes are the same as those for a complete tile. Note that those pins lying on the boundary are directly connected to the outside of the array.

The $1/4$-tiles can only be sitting in the three corners of the array, which are the northeast, southeast, and southwestern corners, respectively. For a $1/4$-tile, there is an inter-tile edge with capacity *B-cap*. According to Equation (5), *B-cap* limits the number of wires that can pass through the $1/4$-tile. Use the inter-tile edges to connect to the outside of of the array is sufficient for the capacity constraints. The model of a $1/4$-tile in the northeast corner of the array is shown in Fig. 4(e).

A super source s and a super sink t are also created in the network. We connect s to the pin nodes of all to-be-escaped pins by edges with capacity 1. Finally, all edges from the boundary tiles to the outside of the array are connected to the sink t.

Let us call an escape routing is $legal$ if it satisfies all the *B-cap*, *H-cap*, and *V-cap* constraints within all tiles. A flow of the network is called $legal$ if the flow on every edge does not exceed the edge capacity. Theorem 1 guarantees the correctness of our network flow model. The proof of the theorem will be provided in the following section.

Theorem 1. *Given a staggered pin array with k to-be-escaped pins that satisfies Equation (5), there exists a legal routing of k pins if and only our network flow model has a legal flow of value k.*



(a) (b) (c) (d) (e)

Fig. 6. (a)-(e) Analysis of the possible flow configurations on the inside-tile network for Equation (5). The dotted lines represent a min-cut in the inside-tile network.

A.1 Proof of Correctness

In order to prove Theorem 1, we have to show the following two lemmas, Lemma 1 and Lemma 2, are both correct.

Lemma 1. *If a legal routing of k pins exists, there must exist a legal flow of value k in our model.*

Proof. We prove this by construction. Let R be a legal routing of k pins that the wires of the routing have the minimum crossings with the tile boundaries. By assigning a direction from the pin to the outside of the array, R has the following two properties: (1) a tile boundary cannot have the wires with opposite directions passing it (see Fig. 5(a)); (2) if a pin is routed into one tile, no wires can exit the tile from the two neighboring boundaries of the pin (see Fig. 5(b)). These two properties can be proved by contradiction. For any R that violates either property, some wires in R can always be reconnected so that the property holds and the number of crossings is further reduced, which contradicts R has the minimum crossings. The complete proof is omitted here due to the page limit.

The number of wires that cross each tile boundary is equal to the flow that cross the tile boundary. We call the flow that enters a tile incoming flow, IF, and the flow that exits a tile outgoing flow, OF. Because of the continuity of the routing, the incoming flow is equal the outgoing flow for each tile. Therefore, we can apply the flow algorithm to each tile to obtain a flow solution on the inside-tile network. So, next we will show we can always obtain a legal flow solution for any incoming flow and outgoing flow configurations.

For the ease of presentation, let us denote the NW-node, NE-node, SW-node, and SE-node as the peripheral nodes. All the possible of configurations of the incoming/outgoing flow are as follows:

(1) The flow enters(exits) only one peripheral node and exits(enters) three peripheral nodes (see Fig. 6(a)). So $IF \leq$ *B-cap*, and the minimum cut, MC, is equal to $\lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor$. With Equation (5), we have $IF \leq \textit{B-cap} \leq MC$, and thus, $IF \leq MC$.

(2) The flow enters only one peripheral node and there is another flow from the pin at the the north corner. We assign the flow from the pin to a node that has no flow (see Fig. 6(b)). Due to Property 2, there is no flow coming out at from the NE-node. So the wires must cross the horizontal cut to exit the tile. Thus, IF cannot exceed the *H-cap*, otherwise the routing is illegal. Thus, we have $IF \leq \textit{H-cap} = MC$.

(3) The flow enters two adjacent peripheral nodes and exits two adjacent peripheral nodes (see Fig. 6(c)). The wires must cross the horizontal/vertical cut of the tile depending on the locations of the two adjacent nodes. So, $IF \leq \textit{H-cap} = MC$.

(4) The flow enters only one peripheral node and exits one peripheral node (see Fig. 6(d)). Since the routing is legal, $IF \leq \mathsf{min}(\textit{B-cap}, \textit{H-cap}, \textit{V-cap})$. And, $MC \leq \mathsf{min}(\lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor, \textit{H-cap}, \textit{V-cap})$. Thus, we have $IF \leq MC$.

(5) The flow exits two nonadjacent peripheral nodes (see the left of Fig. 6(e)). According to Property 2, there must not exist any flow from a pin to the peripheral nodes. So, we know $IF \leq \textit{B-cap} + \textit{B-cap} = 2\textit{B-cap}$. The minimum cut of this inside-tile network is shown as the right of Fig. 6(e), and $MC = \lfloor \textit{H-cap}/2 \rfloor + \lceil \textit{H-cap}/2 \rceil + \lfloor \textit{V-cap}/2 \rfloor + \lceil \textit{V-cap}/2 \rceil = \textit{H-cap} + \textit{V-cap}$. So, with the capacity inequality, we have $IF \leq 2\textit{B-cap} \leq (\textit{H-cap} + \textit{V-cap}) = MC$.

The above analysis can be easily extended into the boundary tiles. So, we know $IF \leq MC$ for all tiles. According to max-flow min-cut theorem [1], if $IF \leq MC$, then there must exist a legal flow solution in the network. There we can always obtain a legal flow on any inter-tile network by the maximum flow algorithm. □

Lemma 2. *A legal flow of value k can be transformed int a legal routing of k pins.*

Proof. A procedure proposed in [8] can transform a flow of value k into a planar topology of routing with k pins. Although our network flow model is different from the one in [8], such procedure still can be applied on our flow solution, as long as the flow solution is legal. Since the *B-cap* constraint can be captured by the inter-tile edge, *H-cap* constraint can be captured by the two vertical edges in a tile with capacity $\lfloor \textit{H-cap}/2 \rfloor$ and $\lceil \textit{H-cap}/2 \rceil$, and the *V-cap* constraint can be captured by the two horizontal edges in a tile with capacity $\lfloor \textit{V-cap}/2 \rfloor$ and $\lceil \textit{V-cap}/2 \rceil$, our network flow model can correctly capture the capacity constraints within each tile. The routing transformed from a legal flow must be legal as well. □

B. Modeling Equation (4)

The model for Equation (4) is mostly identical to the former model. For each tile, we give both vertical edges a capacity of $\lfloor \textit{H-cap}/2 \rfloor$ and both horizontal edges a capacity of $\lfloor \textit{V-cap}/2 \rfloor$ (see Fig. 7(b)). Each tile contains one more node than that in the former model. The node is a center node, called C-node. The center node has a capacity 1. Node capacity can be realized by splitting the node into two nodes and adding an edge with the same capacity between them. Bidirectional edges are created to connect the C-node with the NW-node, NE-node, SE-node, and SW-node. We give such edges infinite capacity.

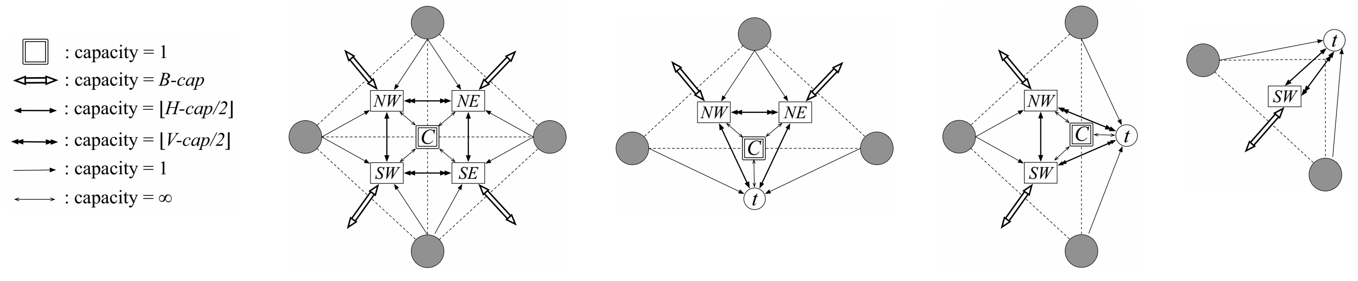


Fig. 7. Our network model for Equation (4). (a) The capacity of each edges; (b) a tile; (c) a $1/2$-tile along the bottom boundary of the array; (d) a $1/2$-tile along the right boundary of the tile; (e) a $1/4$-tile in the northeast corner of the array. t is the super sink.

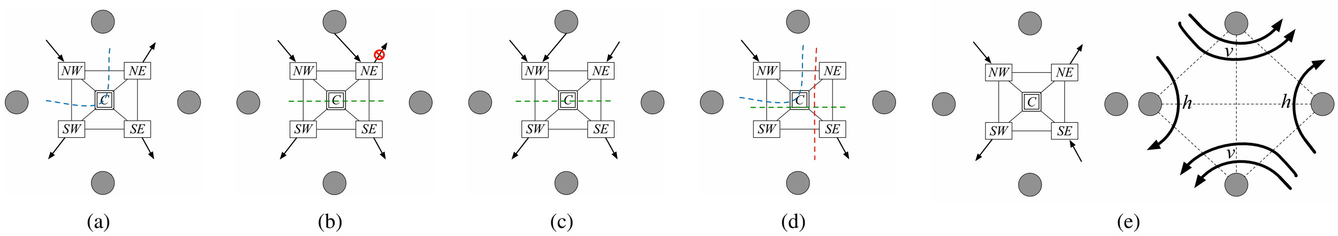


Fig. 8. (a)-(e) Analysis of the possible flow configurations on the inside-tile network for Equation (4). The dotted lines represent a min-cut in the inside-tile network. $h = \lfloor \textit{H-cap}/2 \rfloor$ and $v = \lfloor \textit{V-cap}/2 \rfloor$.

For the $1/2$-tiles, a center node with a capacity 1 is created as well (see Fig. 7(c) and 7(d)). Bidirectional edges with infinite capacity are also used to connect the center node to other nodes. Edges with capacity $\lfloor \textit{H-cap}/2 \rfloor$ and $\lfloor \textit{V-cap}/2 \rfloor$ are created between the nodes inside them and connect to the outside of the array.

The $1/4$-tile has a node inside it, and edges with capacity $\lfloor \textit{H-cap}/2 \rfloor$ and $\lfloor \textit{V-cap}/2 \rfloor$ are used to connect to the outside of the array. Similarly, a super source s and a super sink t are created as well, and they are connect with the to-be-escaped pins and the outside edges from the boundary tiles, respectively.

The following theorem guarantees the correctness of the network flow model. The proof will be presented in the next section.

Theorem 2. *Given a staggered pin array with k to-be-escaped pins that satisfies Equation (4), there exists a legal routing of k pins if and only if our network flow model has a legal flow of value k.*

B.1 Proof of Correctness

We have to show that Lemma 3 and Lemma 4 are both correct.

Lemma 3. *If a legal routing of k pins exists, there must exist a legal flow of value k in our model for Equation (4).*

Proof. The construction from a routing to the inside-tile networks and the possible configurations of the incoming/outgoing flow for a tile are both identical to those shown in Section A.1. An example of all the possible configurations is given in Fig. 8. Since the detailed analysis of the configuration (1)-(4) is similar to the one ins Section A.1, the analysis is omitted. Here, we only focus on the configuration (5):

(5) The flow exits two nonadjacent peripheral nodes (see the left of Fig. 8(e)). Suppose a tile has $IF = 2 \cdot \lfloor \textit{H-cap}/2 \rfloor + 2 \cdot \lfloor \textit{V-cap}/2 \rfloor$, which is shown in the right of Fig. 8(e). Since $\textit{B-cap} = \lfloor \textit{H-cap}/2 \rfloor + \lfloor \textit{V-cap}/2 \rfloor + 1$, only one more flow can enter/exit the tile, otherwise the routing is illegal. So, IF is bounded by $2 \cdot \lfloor \textit{H-cap}/2 \rfloor + 2 \cdot \lfloor \textit{V-cap}/2 \rfloor + 1$. Thus, $IF \leq MC$. □

Lemma 4. *A legal flow of value k in our model for Equation (4) can be transformed int a legal routing of k pins.*

Proof. As described in the proof for Lemma 2, an existing approach can transform a flow of value k into a planar routing with k pins. In our network flow model, the horizontally diagonal capacity can be ensured by the two vertical edges and the center node with a total capacity of $2 \cdot \lfloor \textit{H-cap}/2 \rfloor + 1 = \textit{H-cap}$, while the vertically diagonal capacity can be ensured by the two horizontal edges and the center node with a total capacity of $2 \cdot \lfloor \textit{V-cap}/2 \rfloor + 1 = \textit{V-cap}$. □

C. Escape Routing Algorithm

Based on the input of the problem, we can decide a network flow model and then apply the maximum flow algorithm on it. The max-flow solution can be transformed into a topological routing by splitting the nodes and edges of the flow solution [6]. Then the topological routing can be converted into an octilinear routing by existing algorithms [4, 5]. An example that a flow of total value 4 turns into an octilinear routing is shown in Fig. 9. We can see that the zigzag wires in the octilinear routing effectively utilizes the routing area.

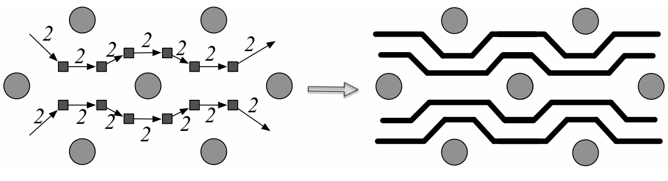


Fig. 9. A flow solution (left) is transformed into an octilinear routing(right). The zigzag wires obtained from the octilinear routing style effectively utilize the routing area.

TABLE I
EXPERIMENTAL RESULTS

	array	escape	Eqn(4)	capacities			our results	
	$m\times n$	pin no.	or Eqn(5)	*B*	*H*	*V*	routability	runtime
case_1	35×13	157	(5)	2	3	4	100%	0.12*s*
case_2	25×17	163	(5)	2	4	4	100%	0.13*s*
case_3	35×13	160	(5)	2	4	4	100%	0.12*s*
case_4	55×17	374	(5)	2	4	4	100%	0.29*s*
case_5	17×34	86	(5)	2	2	4	100%	0.16*s*
case_6	17×34	113	(5)	2	2	4	100%	0.16*s*
case_7	8×180	704	(5)	2	4	4	100%	0.48*s*
case_8	35×13	140	(4)	2	3	1	100%	0.12*s*
case_9	11×27	42	(5)	2	2	3	100%	0.08*s*
case_10	17×34	197	(5)	2	4	4	100%	0.17*s*
case_11	25×34	172	(5)	2	2	4	100%	0.26*s*

We can also assign cost 1 to the inter-tile edges, and zero cost to all other edges. We can compute a min-cost max-flow solution to minimize the number of tiles each wire traverses and thus the total wire length can be minimized.

IV. EXPERIMENTAL RESULTS

We implement our network flow based escape routing algorithm in C++ and test it on several industrial data sets. The min-cost flow solution of our model is obtained by the min-cost flow solver CS2[2]. All experimental results are performed on a workstation with two 2.4GHz Intel Xeon CPU.

We test our router on eleven data sets and the result is shown in Table 1. Among the eleven data sets, *case_1* to *case_7* are actual industrial data and *case_8* to *case_11* are derived from industrial data with some modification. The first seven columns of the table give the information on the data including the name, the pin array size, the number of to-be-escaped pins, the property of the input (satisfying Equation (4) or Equation (5)), and the capacity rules (*B-cap*, *H-cap*, and *V-cap*). The last two columns show the routability of our results as well as the runtime of our router. The routability is defined as the value of the number of routed pins/the number of to-be-escaped pins.

It can be seen that all data sets satisfy the capacity inequality. Among the eleven data sets, ten data sets satisfy Equation (5) while one data set satisfies Equation (4). Our router can successfully route all data sets in very short time. Our routing solution of *case_10* is shown in Fig. 10. A dashed square is drawn on top of the routing result to show our router can handle the dense designs. It can be seen that almost all the *B-cap*, *H-cap* and *V-cap* along the square is used by our routing, which indicates that the routing is very dense.

V. CONCLUDING REMARKS

In this paper, we studied the escape routing problem for staggered pin arrays. Based on the geometry of the staggered pin array, a tile can be defined as a diamond formed by four adjacent pins and hence the corresponding capacity constraints are generated for a tile. Then we formulated the escape routing problem for staggered pin arrays. We proposed two network flow models to model the capacity constraints of staggered pin arrays, and we proved the correctness of our models. These network flow models led to an optimal algorithm. From the experimental results, it is shown that our escape routing algorithm can successfully route the industrial data in very short time.

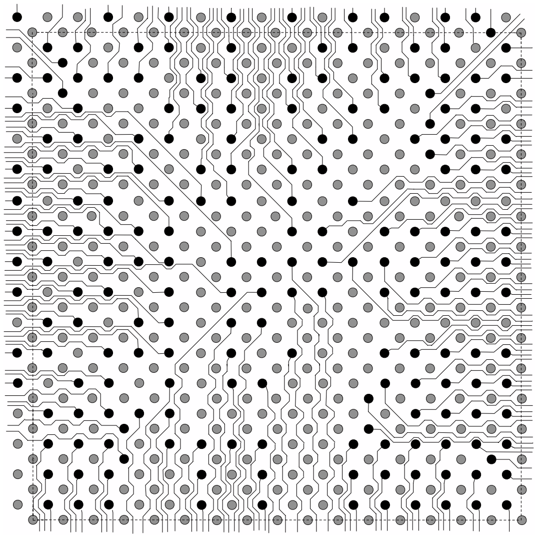


Fig. 10. Routing solution of *case_10*. The dashed square is drawn on top of the result to show that routing uses up almost all routing resources.

REFERENCES

[1] R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network Flows: Theory, Algorithms, and Applications*. Prentice Hall, Feb. 1993.

[2] CS2: min-cost flow solver. `http://www.igsystems.com/cs2/index.html`. Copyright: 1995-2006 IG Systems, Inc.

[3] Y.-K. Ho, H.-C. Lee, and Y.-W. Chang. Escape routing for staggered-pin-array PCBs. In *Proc. Int. Conf. on Computer-Aided Design*, pages 306–309, 2011.

[4] D. Staepelaere, J. Jue, T. Dayan, and W. W.-M. Dai. SURF: Rubber-band routing system for multichip modules. *IEEE Des. Test. Comput.*, 10(4):18–26, Dec. 1993.

[5] D. J. Staepelaere. Geometric transformations for a rubber-band sketch. Master's thesis, University of California at Santa Cruz, Santa Cruz, CA, USA, Sept. 1992.

[6] T. Yan and M. D. F. Wong. A correct network flow model for escape routing. In *Proc. Design Automation Conf.*, pages 332–335, 2009.

[7] T. Yan and M. D. F. Wong. Recent research development in PCB layout. In *Proc. Int. Conf. on Computer-Aided Design*, pages 398–403, 2010.

[8] T. Yan and M. D. F. Wong. Correctly modeling the diagonal capacity in escape routing. *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, 31(2):285 – 293, Feb. 2012.