Pad Assignment for Die-Stacking System-in-Package Design

Wai-Kei Mak, Yu-Chen Lin, Chris Chu, and Ting-Chi Wang

***Abstract*—Wire bonding is the most popular method to connect signals between dies in System-in-Package (SiP) design nowadays. Pad assignment, which assigns inter-die signals to die pads so as to facilitate wire bonding, is an important physical design problem for SiP design because the quality of a pad assignment solution affects both the cost and performance of a SiP design. In this paper, we study a pad assignment problem, which prohibits the generation of illegal crossings and aims to minimize the total signal wirelength, for die-stacking SiP design. We first consider the two-die cases and die-stacks with a bridging die, and present a minimum-cost flow based approach to optimally solve them in polynomial time. We then describe an approach, which uses a modified left edge algorithm and an integer linear programming technique, for pyramid die-stacks with no bridging die. Finally, we discuss extensions of the two approaches to handle additional design constraints. Encouraging experimental results are shown to support our approaches.**

***Index Terms*—System-in-Package, pad assignment, wire bonding, die-stack.**

I. INTRODUCTION

Comparing System-in-Package (SiP) [2]–[4] with System-on-Chip (SoC), SiP is a more economical option than SoC in many consumer electronic products because of the high process complexity and cost associated with SoC. On the other hand, compared with traditional system integration where multiple dies with separate packaging are mounted on a PCB, SiP has the advantages of smaller size, lower cost, higher performance, lower power, and shorter time to market. So, today SiP is already widely used in consumer electronics such as cell phones. Currently, SiP design is mostly done by ad hoc methods and the quality of a design is heavily dependent on the designers' expertise. Tool support specific to SiP design is still inadequate [3], [5]–[9].

Wire bonding [10] is the most popular method to connect signals between different dies in SiP nowadays. As shown in Figure 1, a die-stacking SiP design using wire bonding has the following properties: (1) Dies with different sizes are stacked and pad signals are connected by bonding wires. (2) Die pads can only be located on die boundaries.

This work was partially supported by National Science Council of Taiwan under Grant NSC ???. A preliminary version of the paper appeared in ICCAD'2009 [1].

Wai-Kei Mak and Ting-Chi Wang are with the Department of Computer Science, National Tsing Hua University, Taiwan. Yu-Chen Lin is with Synopsys, Taiwan. Chris Chu is with the Department of Electrical and Computer Engineering, Iowa State University, USA. (emails: wkmak@cs.nthu.edu.tw, tcwang@cs.nthu.edu.tw, kenter78@gmail.com, cnchu@iastate.edu)

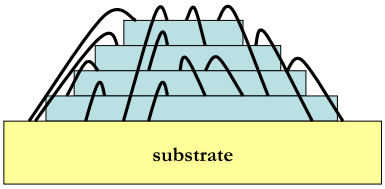

Fig. 1. Wire bonding for die-stacking SiP design.

In the simplest type of SiP design, all the dies in a die-stack are individually wire bonded to the package substrate and the substrate can take care of interconnecting the dies to each other and the outside world. Assembly companies usually also have the capability to make direct die-to-die wire bond connections. But die-to-die wire bonding can only be made when a die is sufficiently larger than the die above it to allow enough room for the die-to-die wire bond connections. In conventional pyramid stacking, a smaller die is always stacked on top of a larger one as in Figure 1.

When it is required to make a stack of same sized dies or stack a larger die on top of a smaller one, a spacer (a dummy layer of silicon) is used to provide space for the loop height of the lower wire bonds under the upper die. An example is shown in Figure 2. In such case, the use of spacers between dies will increase the total package thickness. The impact force during bonding can also cause die deflection of the overhang unsupported die edges of the upper die. Moreover, die-to-die wire bond connections are also impossible.

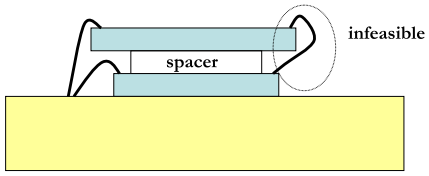

Fig. 2. Stacking a die on top of another die of equal or smaller size requires spacer insertion and prohibits die-to-die wire bonds.

An important stage during the SiP physical design flow is *pad assignment* which assigns inter-die signals to die pads. Pad assignment is typically invoked after the partitioning of the components of a system (or sub-system) into different dies. After pad assignment, the subsequent floorplanning/placement and routing stages can then be carried out. When we do pad assignment, some wire crossings might be produced as in Figure 3(a), Figure 4, and Figure 5. Nevertheless, some types of crossings are in fact tolerable. Suppose the dies are indexed from top to bottom, U_i and L_i denote the indexes of the dies on

which the upper and lower end points of wire w_i are located. Then, if the interval of wire w_i is properly contained in the interval of wire w_j (i.e., $U_i > U_j$ and $L_i < L_j$), then a crossing between between w_i and w_j is legal. For example, Figure 3(a) shows one such legal crossing and Figure 3(b) shows why the two wires actually will not touch each other by viewing it from a different angle.

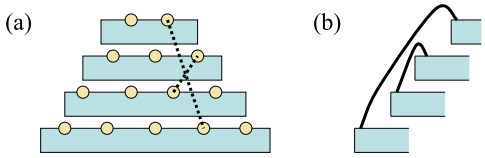

Fig. 3. (a) A legal crossing. The interval of one wire is properly contained in the interval of another wire. (b) The corresponding wires can have enough clearance between them.

On the other hand, two types of crossing are considered *illegal* in this paper. The first type of illegal crossing is a crossing of two wires in which the upper pads of both wires are co-located on a die and the lower pads of both wires are co-located on another die. Figure 4 shows such a crossing. The other type of illegal crossing is shown in Figure 5 in which $U_i < U_j$ and $L_i < L_j$, and the four pads almost align into a straight line when projected onto the *x*-*z* plane. To make it more precise, one can introduce a user-specified constant *dis* such that if either distance d_1 or d_2 shown in Figure 5(b) is less than *dis*, then the two wires are considered too close. The usual practice of design house is to avoid these two types of crossing, as they present a hard task to the wire bonding machine and they are likely to cause the final bonding wires to touch.

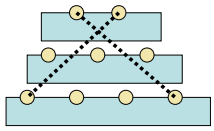

Fig. 4. First type of illegal crossing. Two wires' upper pads are at the same level and their lower pads are also at the same level.

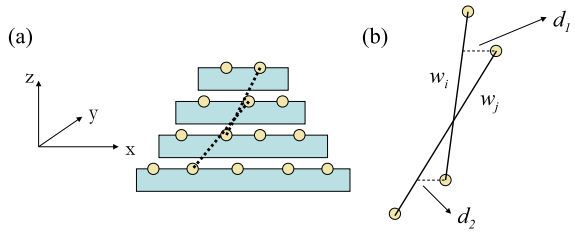

Fig. 5. Second type of illegal crossing.

In this paper, we study a pad assignment problem which prohibits the generation of illegal crossings and aims to minimize the total signal wirelength. We first consider the two-die cases and die-stacks with a bridging die[1] for which we present a minimum-cost flow [11] based approach to optimally solve them in polynomial time. We then describe an approach, which consists of two stages, for pyramid die-stacks with no bridging die. In the first stage, the left edge algorithm (which was originally designed for the classical channel routing problem) [12] is modified to assign as many signals as possible to die pads. If there are remaining signals whose pads cannot be determined in the first stage, our approach proceeds to assign them to die pads using the integer linear programming (ILP) technique [13] in the second stage. Extensive experiments are conducted and encouraging results are reported to support our approaches. We also discuss how to extend them to handle practical design constraints. To the best of our knowledge, our work is the first one which addresses a pad assignment problem for die-stacking SiP design.

The rest of this paper is organized as follows. Section II states the assumptions and the problem formulation. Section III describes our minimum-cost flow based approach. Section IV gives the details of the two-stage approach. Section V presents extensions of the two approaches to satisfy additional design constraints. The experimental results are reported in Section VI, and we conclude the paper in Section VII.

II. Assumptions and Problem Formulation

A. Assumptions

For a SiP design, we assume that the dies are arranged as a stack where the die order and orientation are pre-determined. A die has a rectangular shape, and its pads are positioned along its four sides. We also assume each signal needs to be assigned to exactly two pads on different dies, and each die has adequate pads to accommodate associated signals. Note that it is possible that some signal may only have a pad on a die which needs to be connected to a finger on the package substrate. For this case, we can treat the substrate as a die sitting at the bottom of the die stack and each finger as a pad. A pad assignment result for a signal must be one such that both upper and lower pads assigned to the signal are located on the same side but on different dies. Therefore, the pad assignment result for the one shown on the left of Figure 6 is feasible for the signal while the one on the right is disallowed. For a signal involving more than two dies, we assume that it has been converted to two-pad signals in advance.

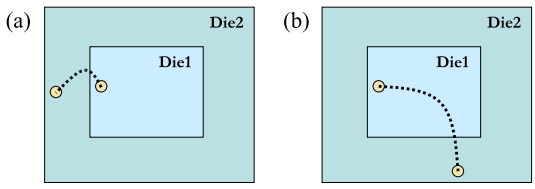

Fig. 6. (a) An allowed assignment. (b) A disallowed assignment.

B. Problem Formulation

The pad assignment problem considered in this paper has the following inputs. A set of dies, numbered by $1, 2, ...$, from top to bottom in the stack, and a set of signals, $w_1, w_2, ...$,

[1]The definition of bridging die will be introduced in section III-B. A die-stack with die-to-substrate bonding wires only can be treated as a die-stack with a bridging die.

are given. Each signal w_i is associated with two die numbers U_i and L_i, representing the dies on which its upper pad and lower pad should be located. For each die, a set of pads on each side and their associated locations in the 3-dimensional space are also given.

The pad assignment problem asks to assign each signal w_i to two pads on the same side of dies U_i and L_i such that no illegal crossing is created and the sum of the wirelengths of all signals (i.e., the total signal wirelength) is minimized. After pad assignment, we know which two pads are assigned to a signal and therefore the wirelength of this signal can be calculated. For simplicity, we use the Euclidean distance between the two pads to approximate the actual length of the bonding wire.

III. Network Flow Based Approach for Two-die or Bridging Die Cases

In this section, we show that the pad assignment problem for any die-stack with only two dies or die-stack containing a bridging die can be solved optimally in polynomial time by a minimum-cost flow based approach.

A. Two-die Cases

For the case with only two dies, there is only one wire type, i.e., each wire is from the top die to the bottom die. We can reduce the two-die pad assignment problem to the minimum-cost flow problem [11] as follows.

Suppose we are given k signals, and the pad sets P and Q on the two dies, respectively. We will construct a flow network $G = (V, E)$, where V and E are the node and edge sets, respectively. For each pad in $P \cup Q$, there is a node in V. A source node s and a sink node t are also added to V. For each pad $p_i \in P$ and each pad $q_j \in Q$ (i.e., a pair of pads on different dies), if they are on the same side, there is a directed edge from p_i to q_j in E with the capacity being 1 and the cost being the wirelength between p_i and q_j. In addition, there is a directed edge from s to each pad p_i in P and there is a directed edge from each pad q_j in Q to t; the capacity and cost of each of these edges are 1 and 0, respectively. Finally, the supply of node s is set to k and the demand of node t is also set to k.

Figure 7 shows the flow network for an instance of the two-die pad assignment problem, where each node p_i ($1 \leq i \leq 12$) represents a pad on the upper die, each node q_j ($1 \leq j \leq 16$) represents a pad on the lower die, the number of signals is six, and $WL(p_i, q_j)$ denotes the wirelength between two pads p_i and q_j. In this example, it is also assumed that pads in the set $\{p_1, p_2, p_3, q_1, q_2, q_3, q_4\}$ ($\{p_4, p_5, p_6, q_5, q_6, q_7, q_8\}$, $\{p_7, p_8, p_9, q_9, q_{10}, q_{11}, q_{12}\}$, $\{p_{10}, p_{11}, p_{12}, q_{13}, q_{14}, q_{15}, q_{16}\}$, respectively) are on the same side.

After the flow network G is built, we proceed to find a minimum-cost flow of G. Note that one can regard this as the problem of computing a minimum cost bipartite matching from P to Q with fixed cardinality k. It is well known that for any minimum-cost flow problem instance with integral edge capacities, if it is feasible, then there exists an integral optimum solution and the network simplex algorithm is guaranteed to find an integral optimum flow in polynomial time [14]. Since all edges of G have integral capacities and we assume that each die has adequate pads to accommodate associated signals, an integral minimum-cost flow F can be found. We now explain how to produce the corresponding pad assignment solution from F. It is not hard to see that there will be k paths of the form: $s \rightarrow p_i \rightarrow q_j \rightarrow t$ such that all edges on each path are saturated (i.e., having a flow of 1); in addition, except the starting node s and ending node t, these paths are all node-disjoint. As a result, we can find k node-disjoint saturated edges (p_i, q_j)'s from these paths, and assign the given k signals to the corresponding pads of these edges one by one in an arbitrary order. For example, in Figure 7, if there are six signals to be assigned, and the bold edges are saturated edges found in a minimum-cost flow, then we can assign the six signals to the pairs of pads $(p_3, q_1), (p_5, q_5), (p_6, q_6), (p_7, q_9), (p_{10}, q_{14}), (p_{11}, q_{16})$.

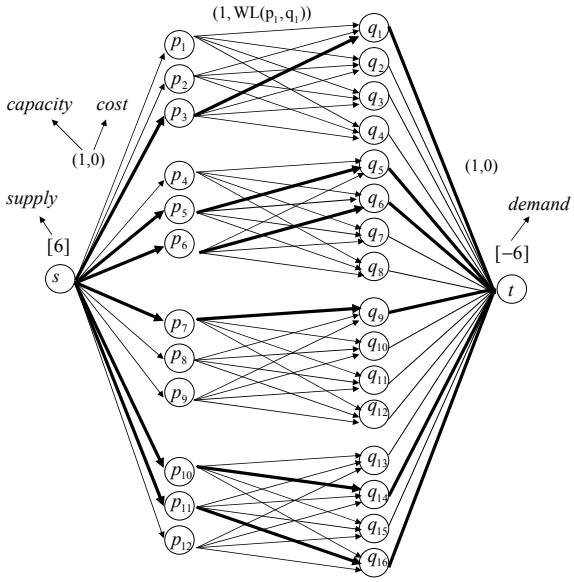

Fig. 7. Flow network for a two-die instance. A fictitious minimum-cost flow is shown with bold edges.

We call the minimum-cost flow based approach above MCF. We now state the optimality of MCF in the following theorem.

Theorem 1. *The two-die pad assignment problem can be optimally solved by MCF.*

Proof: Let S denote a two-die pad assignment problem instance, k be the number of signals, and G be the corresponding flow network. Based on the way we build G, it is easy to see that each pad assignment solution (which may have illegal crossings) of S corresponds to a feasible flow of G, and each feasible flow of G corresponds to $k!$ pad assignment solutions (because there are $k!$ different ways to assign the k signals to the pads of k saturated edges (p_i, q_j)'s). Moreover, the total wirelength of a pad assignment solution is equal to the cost of the corresponding feasible flow of G. Therefore, if we can prove that the pad assignment solution produced from a minimum-cost flow of G by MCF does not have any illegal crossing, we can conclude that this pad assignment solution is

an optimal one because it also has the shortest wirelength.

Assume that there exists one wire crossing or more in the pad assignment solution produced from a minimum-cost flow F of G by MCF. Since S has only two dies, each crossing must be an illegal crossing of the first type. For each pair of wires which induce a crossing (see the left part of Figure 8), we can always swap the signals for the top pads of the wires (see the right part of Figure 8) to remove the crossing. In addition, after the swap, the new wirelength will become shorter than the old wirelength, based on triangle inequality. We can apply the swapping technique to each crossing until all crossings are eliminated. The resultant pad assignment solution will correspond to a feasible flow with cost smaller than that of F. This contradicts that F is a minimum-cost flow, and thus the pad assignment solution produced by MCF has no illegal crossing. This completes our proof. ■

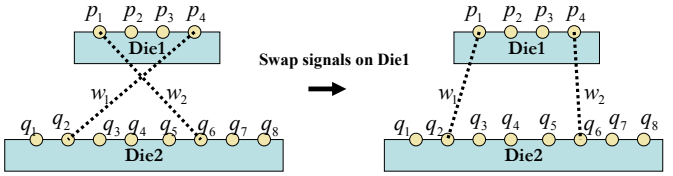

Fig. 8. Swapping signals for top pads to get shorter wirelength.

B. Bridging Die Cases

It is not uncommon for a multi-die SiP design to have a bridging die. In such a SiP design, there is one die which we call a *bridging die* such that all the other dies are connected to it only but are not connected to one another. We call the remaining dies non-bridging dies. For example, Figure 9 gives a three-die pad assignment instance where die Q is the bridging die. We show that the pad assignment problem for a multi-die SiP design with a bridging die can also be cast as a minimum-cost flow problem.

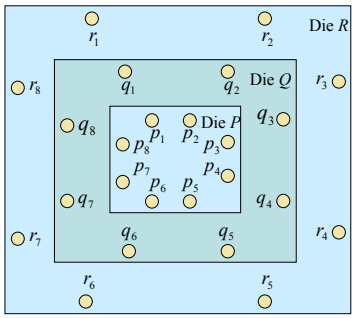

Fig. 9. A pad assignment problem instance with a bridging die Q.

We can construct a flow network as shown in Figure 10 for the three-die instance of Figure 9. A source node s_Q is created for the bridging die Q and two sink nodes t_P and t_R are created for the two non-bridging dies P and R, respectively. Node s_Q is connected to each node q_i which corresponds to a pad in the bridging die with a directed edge of capacity 1 and cost 0. Each node p_i which corresponds to a pad in die P is connected to node t_P with a directed edge of capacity 1 and cost 0. Similarly, each node r_i which corresponds to a pad in die R is connected to node t_R with a directed edge of capacity 1 and cost 0. If pad q_i and pad p_j are on the same side, then we add a directed edge from node q_i to node p_j of capacity 1 and cost equal to the wirelength between the two pads. Similarly, if pad q_i and pad r_j are on the same side, then we add a directed edge from node q_i to node r_j of capacity 1 and cost equal to the wirelength between the two pads. But there is no edge between any pair of nodes p_i and r_j because there is no signal connecting the two non-bridging dies. Finally, the supply at node s_Q is set to the number of signals associated with the bridging die which is 7 in our instance. The demand at each sink node t_α is set to the number of signals associated with non-bridging die α. In our instance, the demands of nodes t_p and t_q are 3 and 4 since dies P and Q have three and four signals, respectively.

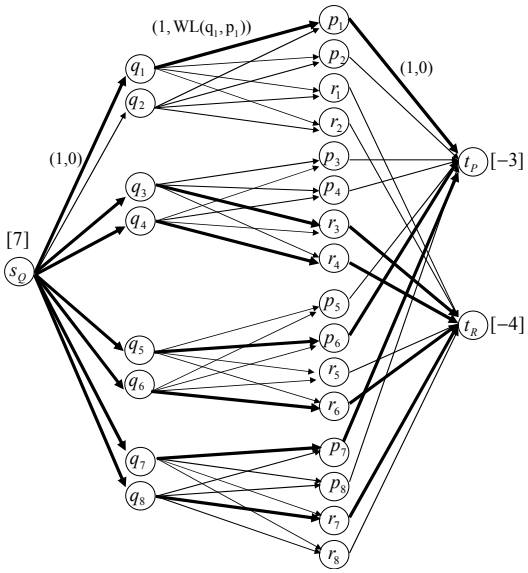

Fig. 10. Flow network for the example in Figure 9. A fictitious minimum-cost flow is shown with bold edges.

Similar to the two-die case, we can obtain an optimal pad assignment solution from a minimum-cost flow of the corresponding flow network. For example, in the minimum-cost flow in Figure 10, there are three saturated edges of the form (q_i, p_j) and four saturated edges of the form (q_i, r_j), all of which are node-disjoint. We can assign the three signals between dies Q and P to the three pairs of pads corresponding to the three saturated edges of the form (q_i, p_j) and assign the four signals between dies Q and R to the four pairs of pads corresponding to the four saturated edges of the form (q_i, r_j).

It is not hard to see that we can generalize the above

construction even if there are more than two non-bridging dies. Similar to the proof of Theorem 1, we can show that the resultant pad assignment solution will not have any illegal crossing of the first type. In addition, as all connections have one end at the bridging die, illegal crossing of the second type is impossible. Thus, the solution must have no illegal crossing. Moreover, any two-die instance can be seen as a case with one non-bridging die by regarding one of its two dies as a bridging die and the other die as a non-bridging die. Consequently, any n-die $(n \geq 2)$ pad assignment problem instance with a bridging die can be transformed into an instance of the minimum-cost flow problem and solved optimally. Thus we have our second theorem below.

Theorem 2. *The pad assignment problem with a bridging die can be optimally solved by MCF.*

Finally, we note that all SiP designs utilizing die-to-substrate wire bonding only can be regarded as instances with a bridging die by treating the substrate as the bridging die. In other words, we can use the minimum-cost flow based approach to optimally handle all SiP designs utilizing die-to-substrate wire bonding only. Since the MCF method has no restriction on the sizes of the dies, it also covers those SiP designs made up of non-pyramid die-stack which normally employ die-to-substrate wire bonding only.

IV. An Approach for Pyramid Die-Stacks with No Bridging Die

In this section, we propose a pad assignment heuristic for SiP designs made up of pyramid die-stack that has no bridging die. Our approach consists of two stages. To reduce the problem complexity, it focuses only on a certain subset of the solution space in the first stage such that the left edge algorithm [12] can be modified and applied to assign as many signals as possible to die pads. If there are remaining signals whose pads cannot be determined in the first stage, an integer linear programming (ILP) based method is invoked in the second stage. Both stages are guaranteed not to generate any illegal crossing. This approach is called MLE+ILP, and its overall flow is shown in Figure 11. The details of each stage are explained in the next two subsections. The experimental results in section VI show that the two-stage approach can find a feasible solution with near optimal wirelength for all test cases within a short time.

A. First Stage: Modified Left Edge (MLE) Algorithm

For each side of a die, we label all its pads[2] from the center towards the two ends, starting with 1. The pads on different dies but on the same side and with the same label will form an *imaginary track*, and the length of the track is determined by the difference of the largest and smallest die numbers among all the dies covered by this track (see Figure 12). In the first stage, our approach tries to assign as many signals as possible to these imaginary tracks. That is, a signal can be assigned only to two pads with the same label. The tracks are constructed in this way for two reasons. First, the pads on the same track will nearly fall in a straight line perpendicular to the corresponding side. So assigning signals to the tracks will not result in unnecessarily long connections. Second, any assignment of signals to the tracks will not form the first type of illegal crossing.

[2]A pyramid die-stack is assumed and the number of pads on each side of an upper die is always less than or equal to that on the same side of a lower die.

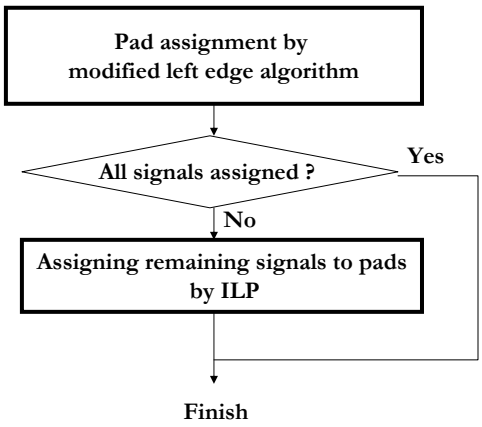

Fig. 11. The overall flow of our MLE+ILP approach.

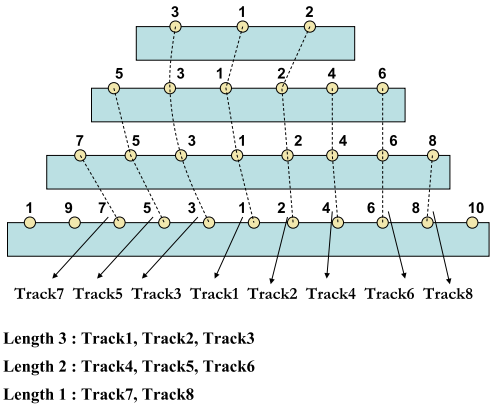

Fig. 12. Forming imaginary tracks.

The original left edge algorithm [12] is for solving the channel routing problem, and therefore in order to modify and apply it to solve the pad assignment problem in the first stage, we need to decide the set of available tracks and the set of wires to be routed. In our pad assignment problem, each die has four sides, and therefore we will consider pad assignment for four sides simultaneously. For the example in Figure 13, it will form 12 tracks with length 3, 4 tracks with length 2, and 4 tracks with length 1 at the beginning, as shown in Figure 14, when four sides are considered together. Since each signal w_i is to be assigned to two pads on dies U_i and L_i, the signal forms a wire to be routed with its two end points on dies U_i and L_i. Throughout the rest of this paper, signal and wire will be used interchangably. For the example in Figure 13, signal b is on Die 1 and Die 3, so there will be a wire from Die 1 to Die 3. Figure 14 shows all tracks and all wires of the example in Figure 13. Note that the channel routing problem assumes that the available tracks are unlimited and have equal length, and the left edge algorithm tries to route

the wires using as few tracks as possible. However, in our pad assignment problem, the tracks are limited and could have different lengths, thus making our problem different from the channel routing problem.

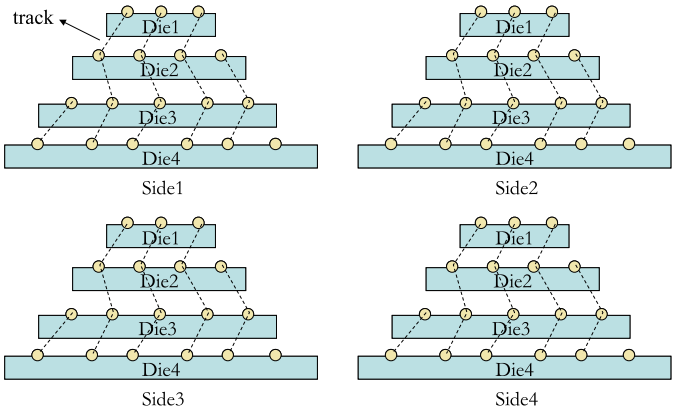

Fig. 13. A SiP pad assignment instance.

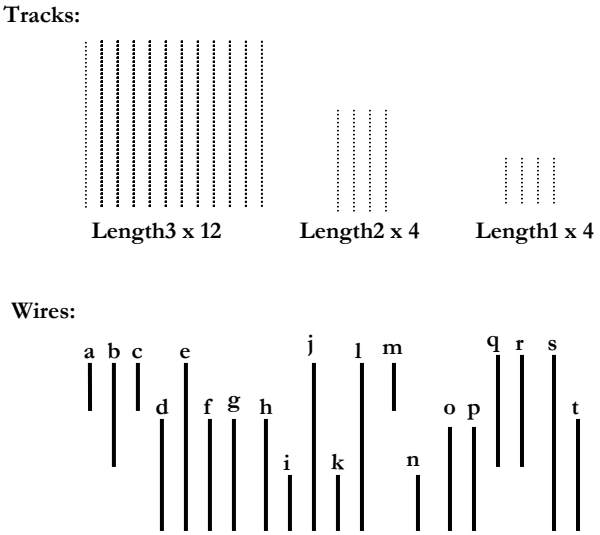

Fig. 14. Tracks and wires for the instance in Fig.13.

After all the tracks and wires are created, we sort all the wires to form an order for assignment. Wires are sorted in increasing order of their upper end points as in the original left edge algorithm. If there is a tie for the upper end points, a wire with higher lower end point (i.e., a shorter wire) is ordered first which is not required in the original left edge algorithm. In this way, if we cannot route all signals that have their upper end points at the same die, we will choose to route the ones that are shorter which will leave more resources for routing subsequent wires. We use this heuristic strategy because we want to route as many signals as possible if not all signals can be routed in this stage.

We start the assignment process and assign one wire at a time according to the sorted order of wires. When we process a wire, we look for a track that can accommodate it. If no track has room for the current wire, the wire fails to be assigned in this stage. Every time after a wire from Die i to Die j $(i < j)$ is assigned to a track, the pads from Die $(i+1)$ to Die $(j-1)$ in the track can still be used later by other wires (see Figure 15), hence we create a new track from Die $(i+1)$ to Die $(j-1)$ and add it to the end of the set of tracks. Note that the original left edge algorithm does not create such a track. We also note that when we assign a new wire to a particular track, there is a small chance that it will produce an illegal crossing with a previously assigned wire as in Figure 16. In this case, we will choose another track to assign the new wire.

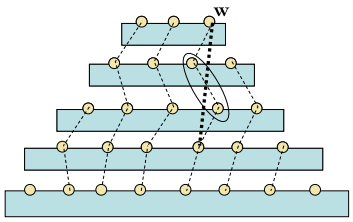

Fig. 15. After assigning wire w, the circled portion of the track between but excluding w's two end points can still be used by other wire.

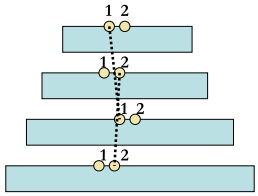

Fig. 16. A situation where illegal crossing can arise. Suppose a wire between dies 1 and 3 have been assigned to track 1, then assigning a wire between dies 2 and 4 to track 2 can result in an illegal crossing.

The wire assignment result produced for the example in Figure 13 is shown in Figure 17. Mapping the wire assignment result back to the original SiP instance yields the pad assignment shown in Figure 18. We call the method used in this stage the modified left edge (MLE) algorithm. As shall be seen in Section VI, the majority of the signals can have their die pads assigned in the first stage.

B. Second Stage: ILP

If there are wires which cannot be assigned to die pads after the first stage, we will assign them by an integer linear programming (ILP) based method in the second stage. In this ILP formation, we have the following constants and variables.

- Constants
  - T_i: $1 \leq i \leq n$
    The i-th wire type. The wire type of a signal is determined by its two associated die numbers. If two signals have the same associated die numbers,

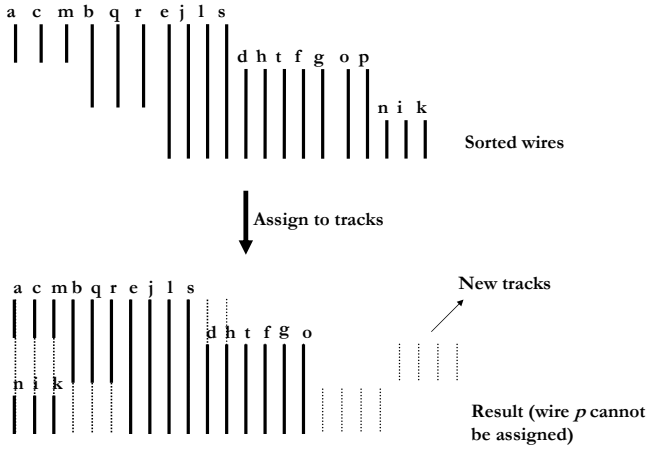

Fig. 17. Wire assignment by the modified left edge algorithm.

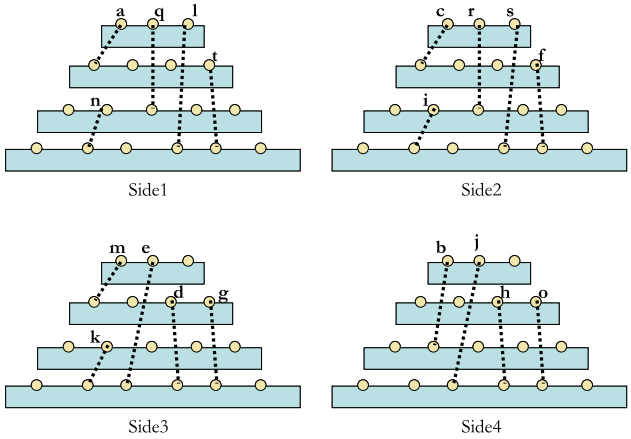

Fig. 18. Pad assignment result.

they have the same wire type. n is the total number of different wire types for the remaining signals.

– N_{T_i}: $1 \leq i \leq n$
The number of remaining signals with wire type T_i.

– C_{T_i}: $1 \leq i \leq n$
The number of wire candidates for wire type T_i. This amount can be determined from the remaining die pads. If a wire candidate causes an illegal crossing with an already assigned wire, then we will not create this wire candidate.

– $W_j^{T_i}$: $1 \leq j \leq C_{T_i}$, $1 \leq i \leq n$
The wirelength of the j-th wire candidate for wire type T_i.

– P_k: $1 \leq k \leq m$
The k-th unassigned pad. Here we only consider those pads which may be used by unassigned signals. m is the total number of such unassigned pads.

– $E_j^{T_i}$: $1 \leq j \leq C_{T_i}$, $1 \leq i \leq n$
$E_j^{T_i}$ is the set of the two pads which are used to realize the j-th wire candidate for wire type T_i.

- Variables
  – $x_j^{T_i}$: $1 \leq j \leq C_{T_i}$, $1 \leq i \leq n$
  $x_j^{T_i} \in \{0, 1\}$. If $x_j^{T_i}$ is 1 in an ILP solution, it means that the j-th wire candidate of wire type T_i is selected; if $x_j^{T_i}$ is 0, the wire candidate is not selected.

For each wire type T_i, it needs to select exactly N_{T_i} wire candidates, so we have the following constraints:

$$\sum_{j=1}^{C_{T_i}} x_j^{T_i} = N_{T_i}, 1 \leq i \leq n$$

Each unassigned pad can only be used by a wire candidate or none, so we have the following constraints:

$$\sum_{\forall E_j^{T_i}: P_k \in E_j^{T_i}} x_j^{T_i} \leq 1, 1 \leq k \leq m$$

For any two wire candidates, say the j-th wire candidate of wire type T_i and the j'-th wire candidate of wire type $T_{i'}$, if they cause an illegal crossing, we need to add the following constraint to avoid the two wire candidates to be selected simultaneously:

$$x_j^{T_i} + x_{j'}^{T_{i'}} \leq 1$$

The objective is to minimize the total wirelength and hence is stated as follows:

$$min \sum_{i=1}^{n} \sum_{j=1}^{C_{T_i}} x_j^{T_i} \times W_j^{T_i}$$

It is worth mentioning that this ILP formulation exactly models our pad assignment problem, if the first stage of our approach is skipped.

The advantage of formulating the pad assignment problem as an integer linear program is that it is easily extendable to consider additional constraints in pad assignment. But a disadvantage of ILP is that potentially it can be very time consuming. To effectively control the size of the ILP, we may reduce the number of wire candidates generated by each unassigned pad. For the wire candidates of a wire type that can be generated by a pad, we only keep at most R wire candidates which have shorter wirelengths than the others. Now if we can find a solution to this new ILP problem, it must also be a solution to the original ILP problem but may not be optimal. On the other hand, if there is no solution found for this new ILP problem, the value of R will be increased by a fixed amount to get another new ILP problem to be solved. The whole process is iterated until a solution is found, or the value of R reaches the upper limit but no solution is found.

V. Extensions

Our proposed SiP pad assignment approaches are very flexible. In this section, we show how they can be modified if there are additonal constraints on the pad assignment solution. We

will consider side constraints and pre-assignment constraints as examples. A designer may specify on which side of a SiP design a signal should go in or out of the dies that it connects, which we call a *side constraint*. If a signal has one terminal pre-assigned to certain pad on a die, we call this a *pre-assignment constraint*.[3] One special case is that if we reuse a legacy die in a SiP design, then all signals in the legacy die are pre-assigned to specific pads on the legacy die.

A. Extending the MCF Approach

Let us consider the 3-die instance in Figure 9 again. Suppose a desiger wants signals a and b to be assigned to the top side and signal c to be assigned to the right side, and the rest of the signals can be assigned to any of the four sides. Then a minimum-cost flow of the network in Figure 10 will not guarantee to give a feasible pad assignment satisfying the side constraints.

To solve the above problem instance with side constraints, we need to modify the right hand side of the network in Figure 10. We replace node t_α for each non-bridging die α and their incident edges as follows. In general, we can distinguish five types of signals which are (i) signals that must be assigned to the top side, (ii) signals that must be assigned to the bottom side, (iii) signals that must be assigned to the right side, (iv) signals that must be assigned to the left side, and (v) signals that can be assigned to any side. For each non-bridging die α, we introduce nodes top_α, $bottom_\alpha$, $right_\alpha$, $left_\alpha$, and any_α corresponding to the five possible types of signals associated with die α. Node top_α ($bottom_\alpha$/$right_\alpha$/$left_\alpha$) only has unit-capacity incoming edges from nodes corresponding to pads on the top (bottom/right/left) side of die α. All these edges have cost 0. Node any_α only has incoming edges from nodes top_α, $bottom_\alpha$, $right_\alpha$, and $left_\alpha$. Each incoming edge of node any_α has infinite capacity[4] and zero cost. Finally, the demand of node top_α ($bottom_\alpha$/$right_\alpha$/$left_\alpha$) is equal to the number of signals in die α that need to be assigned to the top (bottom/right/left) side. The demand of node any_α is equal to the number of signals in die α that can be assigned to any side.

The new flow network assuming signals a and b have to be assigned to the top side, and signal c has to be assigned to the right side is shown in Figure 19. Note that signals a and c are associated with non-bridging die P while signal b is associated with non-bridging die die R, so the number of signals in die P that need to be assigned to the top and the right sides are both 1, and the number of signals in die R that need to be assigned to the top side is 1. It is easy to see that by construction, each node corresponding to a pad (nodes q_i's, p_i's, r_i's in Figure 19) either has exactly one incoming edge and the capacity of that incoming edge is 1, or one outgoing edge and the capacity of that outgoing edge is 1. Hence, each pad will be used by at most one signal when a minimum-cost flow is computed. In addition, the demands of nodes top_α, $bottom_\alpha$, $right_\alpha$, $left_\alpha$, and any_α are set in such a way that a sufficient number of pad pairs will be chosen on each side to satisfy the given side constraints.

A fictitious minmum-cost flow F is shown in Figure 19 and we can obtain an optimal pad assignment as follows. The pad pairs chosen by F are (q_1, p_1) and (q_2, r_2) at the top side, (q_3, p_3) and (q_4, p_4) on the right side, (q_5, r_5) and (q_6, r_6) at the bottom side, and (q_8, r_7) on the left side of the SiP design. For each non-bridging die α, first we arbitrarily assign each of its side-constrained signals to any chosen pad pair associated with the corresponding side of the die without repetition. Then each signal of die α with no side constraint is assigned arbitrarily to any remaining chosen pad pair associated with the die without repetition. For example, for non-bridging die P, its signals are a, c and e. The pad pairs chosen for die P by F are (q_1, p_1), (q_3, p_3) and (q_4, p_4). Signals a and c are assigned first as they are side-constrained. Signal a can be assigned to pad pair (q_1, p_1) at the top side while signal c can be assigned to either pad pair (q_3, p_3) or pad pair (q_4, p_4) on the right side. Finally, signal e can be assigned to the remaining chosen pad pair for die P.

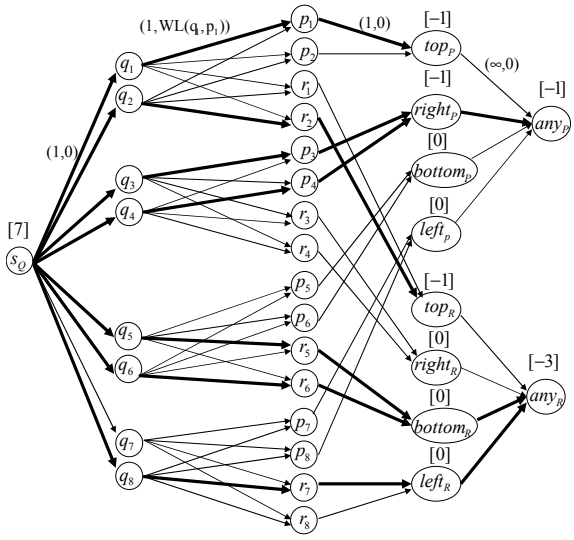

Fig. 19. Flow network incorporating side constraints. A fictitious minimum-cost flow is shown with bold edges. (Edge $(bottom_R, any_R)$ carries a flow of 2 while all other bold edges carry a unit flow.)

The minimum-cost flow approach can be further extended to handle pre-assignment constraints. Suppose in addition to the side constraints for signals a, b and c, the designer has pre-assigned die R's signal d to pad r_8 and die Q's signal e to pad q_5. Then a minimum-cost flow of the network in Figure 19 is no longer guaranteed to correspond to a feasible pad assignment satisfying the pre-assignment constraints. It is because a minimum-cost flow in Figure 19 may not use pad r_8 and/or pad q_5. In order to enforce that pads r_8 and q_5 will be used as prescribed by the designer, we can make some slight modifications in the previous flow network as below. First, we note that pad r_8 belongs to the non-bridging die R while pad q_5 belongs to the bridging die Q. So, we set a demand

[3] If both pads of the same wire w are pre-assigned, it is a fixed wire and it is not necessary to assign pads for it. But we have to remove all wire candidates that will produce an illegal crossing with w.

[4] The capacity can be set to any value larger than the number of non-side-constrained signals in die α.

of 1 for node r_8, remove its only outgoing edge, and reduce the demand of node any_R by 1. On the other hand, we set a supply of 1 for node q_5, remove its only incoming edge, and reduce the supply of node s_Q by 1. The resultant flow network is shown in Figure 20.

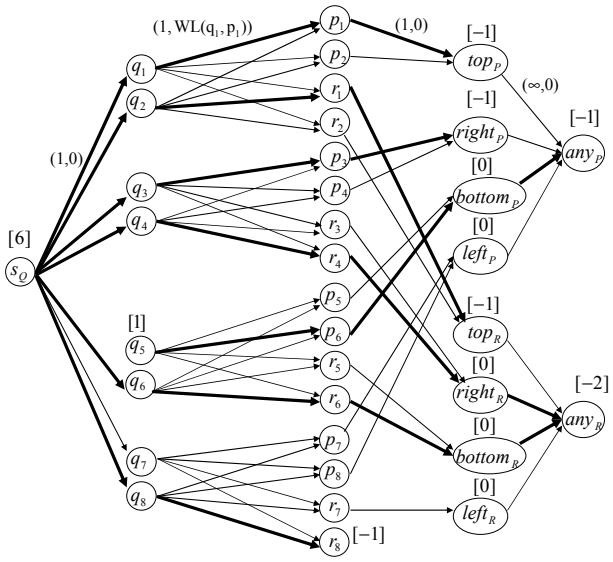

Fig. 20. Flow network incorporating side constraints and pre-assignment constraints. A fictitious minimum-cost flow is shown with bold edges.

A fictitious minmum-cost flow F is shown in Figure 20 and we can obtain an optimal pad assignment as follows. The pad pairs chosen by F are (q_1, p_1) and (q_2, r_1) at the top side, (q_3, p_3) and (q_4, r_4) on the right side, (q_5, p_6) and (q_6, r_6) at the bottom side, and (q_8, r_8) on the left side of the SiP design. We assign signals to appropriate pad pairs in the following order: (i) signals with pre-assigned pads, (ii) signals with side constraints, and (iii) any other signals. First, we single out the chosen pad pairs that include a pre-assigned pad and assign them to the designated signals. In our example, the chosen pad pair (q_5, p_6) includes pre-assigned pad q_5 which is designated for signal e, and (q_8, r_8) includes pre-assigned pad r_8 which is designated for signal d. So, (q_5, p_6) is assigned to signal e and (q_8, r_8) to signal d. Second, for each non-bridging die α, we arbitrarily assign each of its side-constrained signals to any remaining chosen pad pair associated with the corresponding side of the die without repetition. Finally, each remaining signal of die α is assigned arbitrarily to any remaining chosen pad pair associated with the die without repetition. In our example, for non-bridging die P, we can assign its side-constrained signals a and c to (q_1, p_1) and (q_3, p_3), respectively. For non-bridging die R, we can assign its side-constrained signal b to (q_2, r_1). Lastly, we can assign the two remaining signals of die R to (q_4, r_4) and (q_6, r_6) arbitrarily.

We have shown that side constraints and pre-assignment constraints can be handled at the same time. Note that the above construction is independent of the number of non-bridging dies. Hence, any n-die $(n \geq 2)$ pad assignment problem instance with a bridging die can be transformed into an instance of minimum-cost flow problem and solved optimally even if there are side constraints and/or pre-assignment constraints. The extended MCF approach is summarized in Procedure 1.

Procedure 1 Extended MCF Approach

1: create a capacitated network model that includes nodes top_α, $right_\alpha$, $bottom_\alpha$, $left_\alpha$, any_α, for each non-bridging die α;
2: **if** instance is feasible **then**
3: compute a minimum cost flow solution;
4: E = set of all saturated edges of the form (u, v) where u is a pad on the bridging die and v is a pad on a non-bridging die;
5: assign each signal with pre-assigned pad to the edge in E with matching pad;
6: assign each side constrained signal to any unassigned edge in E on the required side;
7: assign remaining signals to the remaining unassigned edges in E;
8: **else**
9: report no feasible solution;
10: **end if**

B. Extending the MLE+ILP Approach

In this subsection, we see how to revise the MLE+ILP approach introduced in section IV to satisfy side constraints and/or pre-assignment constraints. For the first stage in the MLE+ILP approach, we can design a modified left edge algorithm that honors side constraints as shown in Procedure 2. We take note of which side (either TOP, BOTTOM, LEFT, or RIGHT) each track belongs to (line 2). When a wire w with side constraint (either TOP, BOTTOM, LEFT, or RIGHT) is processed, we will skip all tracks not on the correct side (line 5). We assume that $side(w)$ is ANY if there is no side constraint for wire w, otherwise it is either TOP, BOTTOM, LEFT, or RIGHT which corresponds to one of the four possible sides. After finishing the wire assignment by Procedure 2, each track is put back to the original side it comes from.

Then in the second stage, an integer linear program is constructed to assign the remaining signals not assigned in stage one. Let $S = \{\text{TOP, BOTTOM, LEFT, RIGHT}\}$. To take the side constraints into account, we first add the following constants.

- $N_{T_i}^{\sigma}$: $1 \leq i \leq n$, $\sigma \in S$
  The number of remaining signals that are of wire type T_i and must be assigned to the side σ.

- $X_{\sigma}^{T_i}$: $1 \leq i \leq n$, $\sigma \in S$
  A variable $x_j^{T_i}$, $1 \leq j \leq C_{T_i}$, is in the set $X_{\sigma}^{T_i}$ if and only if its corresponding wire candidate is on the side σ.

For each wire type T_i, we need to select at least $N_{T_i}^{\sigma}$ wire candidates for each side σ, so we add the following inequalities to consider the side constraints:

Procedure 2 MLE Algorithm honoring side constraints

```
sort all wires;
record which side each track comes from;
for each wire w in sorted order do
  for each track t do
    if (side(w) = ANY or side(t) = side(w)) and t has room for w that does not cross with any previously assigned wire then
      assign w to t;
      // let w be from die i to die j
      if i + 1 < j − 1 then
        create new track t′ from die i + 1 to die j − 1;
        put t′ at the end of the track list;
      end if
      break;
    end if
  end for
end for
```

$$\sum_{\forall x_j^{T_i} \in X_\sigma^{T_i}} x_j^{T_i} \geq N_{T_i}^{\sigma}, 1 \leq i \leq n, \sigma \in S$$

To address pre-assignment constraints, we can perform an extra pre-processing step to assign all wires with pre-assignment constraints at the very beginning. Suppose wire w is pre-assigned to a pad p on some die. Then we will assign wire w to the particular track that contains pad p before excecuting Procedure 2 and the ILP stage to assign other wires.

VI. EXPERIMENTAL RESULTS

Our approaches were implemented in C++ and run on an Intel 2.4GHz Linux machine with 8GB memory. We used CPLEX [15] to solve the ILP instances and LEDA [16] to solve the minimum-cost flow instances.

First, we compared the efficiency of our minimum-cost flow approach MCF against an optimal ILP-based method on several instances of the special case with two dies only. The ILP method is to run our MLE+ILP approach directly from the second stage (i.e., skipping the MLE stage) and without setting the value of R to control the amount of wire candidates. Hence, it is also an optimal method. The test cases were randomly generated assuming the pad pitch is 50um, the thickness of a die is 6 mil (1 mil=25.4um), and the thickness of the film between adjacent dies is 1 mil. The details of the test cases and the results are shown in Table I, where the WL columns show the total wirelength results. An optimal pad assignment solution produced from MCF could be computed quickly in less than a second for every instance. On the other hand, when the pad assignment problem was modeled as an integer linear program, too many ILP constraints were generated and exceeded the memory limit of our system except for the three smallest instances. The numbers of ILP variables and constraints are shown in the last column.

Next, we experimented on three real SiP designs obtained from the industry. We note that they are instances with a bridging die as discussed in Section III-B, so their pad assignments can be optimally determined by the proposed minimum-cost flow approach. The details of the three designs are given in Table II, where wire type (i, j) means that it is from Die i to Die j. Table II also compares the assignment results by the minimum-cost flow approach against the original assignments. Column Original shows the original wirelengths. Column MCF shows the results by the minimum-cost flow method. It can be seen that MCF efficiently reduced the total wirelength by up to 36.2% (see the results in the Imp. column of Table II). The MCF approach is very fast and the run time was well under a second for each case.

We also tried imposing side constraints and pre-assignment constraints for the three industrial cases. Firstly, we used the minimum-cost flow approach to optimally re-assign all signals assuming each signal must stay on its original given side. The results are shown in Table III under MCF-S. We can see that the total wirelengths are significantly improved compared to the original assignments. Secondly, we re-ran the minimum-cost flow method when one die (a flash memory die) has fixed pad assignments for all its signals. We report the results under the MCF-P columns in the same table. Again we can obtain significant wirelength reduction compared to the original assignments.

In addition, we experimented on ten randomly generated instances of pyramid die-stacks that have no bridging die and their characteristics are listed in Table IV. Table V shows the detailed results in each stage of our MLE+ILP approach for the instances. Stage 1 is assignment by the modified left edge algorithm and stage 2 is additional assignment by ILP. After stage 1, the majority of the wires were assigned to pads and the assignment ratio was 79.31% on average. The run time for stage 1 was less than 0.01 second for all cases. The remaining wires were all assigned to pads in stage 2 by ILP. In each case, we created an initial ILP instance with the wire candidate range $R = 5$. And if an ILP instance was infeasible, we would increase R by 2 iteratively until a feasible assignment could be found. The number of iterations taken for all cases were listed in the last column of Table V. The second stage was also very fast and could finish in a few seconds for all cases.

Finally, we checked if it was computationally feasible to perform pad assignment by ILP directly without reducing the problem size by the modified left edge algorithm for the test cases in Table IV. We tried the ILP approach without limiting the wire candidate range R. We also tried the ILP approach with R set to 5 initially and increased it by 2 iteratively if the corresponding ILP was infeasible. The results are shown in Table VI. Column ILP shows the results when no range was used while column ILP-R shows the results with range. When no range was used, the number of ILP constraints were very large and we ran out of memory in seven out of ten cases. The numbers of ILP variables and constraints for each approach are shown for reference.[5] We set a time-limit of 10,000 seconds for the ILP approach with no range and also for each iteration of ILP-R. Experimental results show that without the help of the modified left edge algorithm, the ILP instances with or

[5]When ILP-R or MLE+ILP took multiple iterations, the numbers of ILP variables and constraints for the last iteration are reported.

TABLE I
COMPARISON OF MCF AND ILP ON TWO-DIE TEST CASES; "-" DENOTES OUT OF MEMORY

test case	# of dies	# of wires	# of pads	MCF WL (um)	MCF Time (s)	ILP WL (um)	ILP Time (s)	ILP #var/#const
case1	2	40	96	7181.96	<0.01	7181.96	0.31	572/34514
case2	2	80	176	14363.92	<0.01	14363.92	4.48	1932/425394
case3	2	160	336	28727.84	0.04	28727.84	103.7	7052/5924354
case4	2	240	496	43091.76	0.07	-	-	-
case5	2	320	656	57455.68	0.13	-	-	-
case6	2	400	816	71819.60	0.20	-	-	-
case7	2	480	976	86183.52	0.27	-	-	-

TABLE II
RESULTS ON REAL DESIGNS

test case	# of dies	# of wires	# of pads	# of wires of each wire type (1,2)	(1,3)	(2,3)	Original WL(um)	MCF WL(um)	MCF Imp.	MCF Time(s)
case8	3	58	197	26	0	32	55483.88	42416.03	23.6%	0.02
case9	3	38	155	15	23	0	212760.68	135641.88	36.2%	0.01
case10	3	141	483	41	0	100	221244.62	169771.38	23.3%	0.08

TABLE III
RESULTS ON REAL DESIGNS WITH SIDE CONSTRAINTS AND PRE-ASSIGNMENT CONSTRAINTS

test case	MCF-S WL(um)	MCF-S Imp.	MCF-S Time(s)	MCF-P WL(um)	MCF-P Imp.	MCF-P Time(s)
case8	42458.14	23.5%	0.02	42612.43	23.2%	0.02
case9	135641.88	36.2%	0.01	146516.41	31.1%	0.01
case10	169822.76	23.2%	0.06	170318.45	23.0%	0.06

TABLE IV
DETAILED INFORMATION OF GENERAL TEST CASES

test case	# of dies	# of wires	# of pads	# of wires of each wire type (1,2)	(1,3)	(1,4)	(1,5)	(1,6)	(2,3)	(2,4)	(2,5)	(2,6)	(3,4)	(3,5)	(3,6)	(4,5)	(4,6)	(5,6)
case11	4	160	352	19	22	35	0	0	31	26	0	0	27	0	0	0	0	0
case12	4	320	672	43	58	55	0	0	58	58	0	0	48	0	0	0	0	0
case13	4	640	1312	87	114	110	0	0	115	115	0	0	99	0	0	0	0	0
case14	5	160	340	16	11	11	14	0	17	16	11	0	12	28	0	24	0	0
case15	5	320	660	32	28	26	30	0	37	27	28	0	26	41	0	45	0	0
case16	5	640	1300	69	61	50	61	0	56	66	61	0	70	73	0	73	0	0
case17	6	160	336	4	10	6	8	8	14	14	5	7	12	8	8	12	16	28
case18	6	320	672	12	23	18	20	19	30	27	12	19	22	16	17	20	29	36
case19	6	640	1296	46	30	49	34	37	35	51	34	38	30	56	57	42	48	53
case20	6	800	1632	59	38	65	45	45	44	60	46	51	41	64	69	52	58	63

TABLE V
DETAILED RESULTS OF EACH STAGE OF MLE+ILP

test case	stage 1 Time (s)	stage 1 # of assigned wires	stage 1 assigned %	stage 1 WL (um)	stage 2 Time (s)	stage 2 # of assigned wires	stage 2 assigned %	stage 2 WL (um)	# of Iter
case11	<0.01	134	83.75	40578.07	0.02	26	16.25	10002.07	1
case12	<0.01	262	81.88	77206.07	0.16	58	18.12	21450.04	2
case13	<0.01	521	81.40	153514.39	7.69	119	18.60	42734.46	5
case14	<0.01	131	81.88	41655.37	0.01	29	18.12	12600.69	1
case15	<0.01	264	82.50	85285.77	0.03	56	17.50	25334.00	1
case16	<0.01	508	79.38	165185.07	0.08	132	20.62	59052.82	1
case17	<0.01	130	81.25	43091.76	0.03	30	18.75	15211.93	1
case18	<0.01	252	78.75	91390.44	0.78	68	21.25	35945.53	2
case19	<0.01	455	71.09	170930.64	6.55	185	28.91	100829.31	6
case20	<0.01	570	71.25	215638.34	10.53	230	28.75	123260.75	6
Average			79.31				20.69		

without range were often too large for an optimal solution to be found in a reasonable amount of time. Timeouts occurred for most of the cases and we report the best solutions found before timeouts in Table VI. Comparing with our two-stage

approach, the total wirelength generated by ILP-R was only 0.85% smaller on average. Hence, the two-stage approach is significantly more efficient than the ILP only approach with little loss of quality.

VII. CONCLUSIONS

In this paper, we formulated the SiP pad assignment problem and presented novel approaches to optimize the bonding wirelength without creating illegal crossings. To the best of our knowledge, our work is the first that addresses the pad assignment problem for die-stacking SiP design. The experimental results demonstrated the efficiency and effectiveness of our approaches. In our formulation, we assume that the orientation of dies are fixed. But since the number of dies is limited in practice, if desired, one may try different orientation combinations for the best wirelength.

VIII. ACKNOWLEDGMENTS

We would like to thank Dr. Wang-Jin Chen at Faraday Technology Corporation for providing us with some technical suggestions and test cases. In addition, we would like to thank the anonymous reviewers for their valuable comments.

REFERENCES

[1] Y.C. Lin, W.K. Mak, C. Chu, and T.C. Wang. Pad assignment for die-stacking system-in-package design. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 249–255, 2009.

[2] S.F. Al-sarawi, D. Abbott, and P.D. Franzon. A review of 3-d packaging technology. *IEEE Trans. on Components, Packaging, and Manufacturing Technology - Part B*, 21(1):2–14, Feb 1998.

[3] A. Fontanelli. System-in-package technology: opportunities and challenges. In *Proceedings of International Symposium on Quality Electronic Design*, pages 589–593, March 2008.

[4] K.L. Tai. System-in-package (sip): challenges and opportunities. In *Proceedings of Asia and South Pacific Design Automation Conference*, pages 191–196, June 2000.

[5] R. Wilson. Sips form good soc alternative, but designer beware. EE Times, May 11, 2004.

[6] L. Golick, J. Goodelle, and T. Shilling. Sip modules call for right blend of tech. EE Times, May 11, 2004.

[7] K. Flton and J. Metcalfe. Facilitating system-in-package (sip) design. EE Times, June 5, 2006.

[8] D. Appello, P. Bernardi, M. Grosso, and M. S. Reorda. System-in-package testing: Problems and solutions. *IEEE Design & Test*, 23(3):203–211, 2006.

[9] P. Cauvet, S. Bernard, and M. Renovell. System-in-package, a combination of challenges and solutions. In *Proceedings of European Test Symposium*, pages 193–199, May 2007.

[10] Bond wire modeling standard. [Online]. Available: http://www.jedec.org/download/search/jesd59.pdf.

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

[12] A. Hashimoto and J. Stevens. Wire routing by optimizing channel assignment within large apertures. In *Proceedings of the 8th Workshop on Design Automation*, pages 155–169, 1971.

[13] R. J. Vanderbei. *Linear Programming: Foundations and Extensions*. Springer, 2007.

[14] J.B. Orlin. A polynomial time primal network simplex algorithm for minimum cost flows. In *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms*, pages 474–481, 1997.

[15] Cplex. [Online]. Available: http://www.ilog.com.

[16] Leda package. [Online]. Available: http://www.algorithmicsolutions.com.

TABLE VI
COMPARISON BETWEEN ILP, ILP-R, AND MLE+ILP ON GENERAL TEST CASES; "-" DENOTES OUT OF MEMORY

	ILP			ILP-R				MLE+ILP		
test case	WL (um)	Time (s)	#var/#const	WL (um)	# of Iter	Time (s)	#var/#const	WL (um)	Time (s)	#var/#const
case11	49972.38	161.55	11576/2580892	49972.38	1	0.79	2440/10388	50580.14	0.02	158/1850
case12	-	-	-	98217.35	1	588.39	2904/3556	98656.11	0.16	410/6930
case13	-	-	-	195913.38	1	10000	9640/41588	196248.85	7.69	1587/85708
case14	54122.13	10000	11480/1525452	54122.13	1	10000	3000/12676	54256.06	0.01	161/628
case15	-	-	-	110422.41	1	10000	6200/26756	110619.77	0.03	320/1574
case16	-	-	-	223861.00	1	10000	12600/54916	224237.89	0.08	835/4558
case17	58056.77	10000	11620/1075454	58059.08	1	10000	3500/14878	58303.69	0.03	275/1188
case18	-	-	-	126149.74	1	10000	7700/34030	127335.97	0.78	904/9448
case19	-	-	-	264067.02	2	20000	21700/223582	271759.95	6.55	3938/241544
case20	-	-	-	332138.26	1	10000	19700/88750	338899.09	10.53	5532/459542
Ratio				1.0				1.0085		