On Optimal Board-Level Routing for FPGA-based Logic Emulation*

Wai-Kei Mak and D. F. Wong
Department of Computer Sciences
University of Texas at Austin
Austin, TX 78712

Abstract—In this paper, we consider a board-level routing problem which is applicable to FPGA-based logic emulation systems such as the Realizer system [3] and the Enterprise Emulation System [5] manufactured by Quickturn Systems. For the case where all nets are two-terminal nets, we present an $O(n^2)$-time optimal algorithm where n is the number of nets. Our algorithm guarantees 100% routing completion if the number of inter-chip signal pins on each FPGA chip in the logic emulation system is less than or equal to the number of I/O pins on the chip. Our algorithm is based on iteratively finding Euler circuits in graphs. We also prove that the routing problem with multi-terminal nets is NP-complete.

1 Introduction

Introduced in the mid-1980's, FPGAs [1,2] combine the programmability of programmable logic devices and the scalable interconnection structure of traditional gate arrays. This combination results in programmable devices with much higher logic density. Compared with traditional ASIC technologies, FPGAs have the advantages of rapid customization, negligible non-recurring engineering cost, and reprogrammability. Such advantages have led to increasing interest in the FPGA technology for a wide variety of applications, such as logic emulation and system prototyping.

Logic simulation is indispensable for the verification of digital system designs. However, due to the high computational complexity of the problem, logic simulation by software oftentimes cannot completely verify the behavior of large digital systems. Recently several logic emulation systems using multiple-FPGAs have been developed [3-7]. These systems are capable of emulating complex digital system designs several orders of magnitude faster than software simulators. As a result, FPGA-based logic emulators can verify large designs that otherwise are not possible by software simulators.

*This work was partially supported by the Texas Advanced Research Program under Grant No. 003658459, by a DAC Design Automation Scholarship, and by a grant from AT&T Bell Laboratories.

32nd ACM/IEEE Design Automation Conference ®

A logic emulator consists of a set of FPGAs (for implementing the logics) and a set of FPICs (for interconnections between the FPGAs). For logic emulation, we first partition the design into parts each of which can fit inside a single FPGA on the logic emulator. Typical CAD tools (e.g. technology mapper, placer, router etc) developed for FPGAs can then be used to complete the internal design of each individual FPGA. Finally, we need to perform board-level routing to connect signals between the FPGA chips. The problem of partitioning a large design into multiple FPGAs is currently an active research area [8,9]. The design of CAD tools for FPGAs is comparatively a more mature subject [1]. On the other hand, very few results have been reported for the board-level routing problem.

In this paper, we address the board-level routing problem using a routing model which is applicable to logic emulation systems such as the Realizer system [3] and the Enterprise Emulation System [5] manufactured by Quickturn Systems. In particular, we consider the case where each FPIC is a small full crossbar and these crossbars only connect to the FPGAs but not to each other. The I/O pins of each FPGA are evenly divided into proper subsets, using the same division on each one. The pins of each crossbar chip are connected to the same subset of pins from each FPGA chip. Thus crossbar chip x is connected to subset x of each FPGA's pins. As many crossbar chips are used as subsets, and each crossbar chip has as many pins as the number of pins in the subset times the number of FPGA chips. (See Figure 1 for an illustration of the architecture.) All existing routing algorithms [4,5] are based on greedy heuristic and do not guarantee 100% routing completion even when such a solution exists.

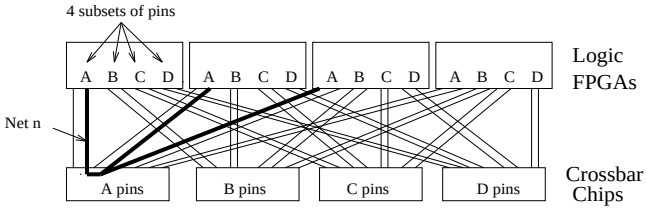

Figure 1: Interconnect architecture.

We present a surprisingly simple result for the case where all nets are two-terminal nets and each I/O pin subset size is even. (Note that the architectural requirement of even I/O pin subset size can be easily met. We also note that most of the nets in a circuit design are two-terminal nets.) We show that as long as the number of *net pins* inside each FPGA chip is less than or

equal to the number of I/O pins on the chip, the routing problem is always routable. Moreover, we also present an efficient polynomial time algorithm, based on iteratively finding Euler circuits in graphs, to achieve 100% routing completion when the above conditions are satisfied. Finally, we also prove that the routing problem with multi-terminal nets is NP-complete.

2 The Board-Level Routing Problem

We will refer to the FPGAs on the logic emulator simply as *chips* and refer to each I/O pin subset on a FPGA simply as a *pin subset*. We assume that all the chips are identical. Let m be the number of I/O pins in each pin subset. Let K be the total number of pin subsets in each chip. It follows that the total number of I/O pins on each chip is mK. Let $\{n_1, n_2, ..., n_l\}$ be the set of signal nets interconnecting the chips. Let $\{T_1, T_2, ..., T_K\}$ be the set of pin subset types.

The board-level routing problem, which will be referred to as *BLRP*, is the problem of assigning the net pins in each chip to the I/O pins on the chip such that all pins belonging to the same net have to be assigned to I/O pins of the same type. Also, at most one net pin can be assigned to each I/O pin. Figure 2 shows a routing solution to a problem with six nets and three chips where m=2 and K=2. Equivalently, BLRP can be viewed as the problem of assigning the nets to the set of pin subset types satisfying the condition that in any chip no more than m nets are assigned to the same pin subset type. Thus BLRP can be formally decribed as follows: Determine a function $F : \{n_1, n_2, \ldots, n_l\} \rightarrow \{T_1, T_2, \ldots, T_K\}$ such that in each chip there are no more than m nets get mapped to T_i for all i. Clearly, $F(n_j) = T_k$ means that we assign all pins of net n_j to I/O pins of type T_k. A routing problem is *feasible* if such function F exists.

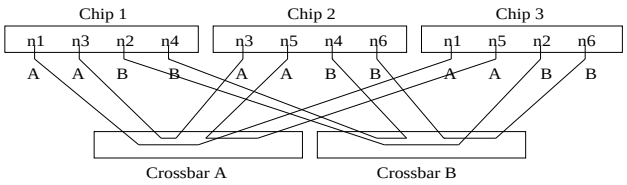

Figure 2: A routing solution.

All existing algorithms [4,5] for BLRP are based on greedy heuristic. We now give an example to show that greedy heuristic may not find a routing solution even when one exists. Consider the following heuristic: the nets are arranged and then processed according to some order; when a net is processed, it is assigned to the first type of pins that is still available at all the chips sharing the net.

For the instance in Figure 2 we may process the nets in the order n_1, n_2, n_3, n_4, n_5, n_6. First, n_1 and n_2 are assigned to pin subset type A. Then n_3 cannot be assigned to pin subset type A as no type A pin is available on chip 1, and it is assigned to pin subset type B. Similarly n_4 is assigned to pin subset type B. But when n_5 and n_6 are processed, they cannot be assigned to pin subset type A as all type A pins in chip 3 are used, and cannot be assigned to pin subset type B as all type B pins in chip 2 are used. So the heuristic only routed two-third of the nets. On the other hand, Figure 2 already shows a feasible solution which routed all nets by assigning net n_i to pin subset type A if i is odd, and to pin subset type B if i is even.

3 Algorithm

In this section, we assume that all nets are two-terminal nets. We also assume m, the pin subset size, is even. For a routing problem to be feasible, clearly a necessary condition is that the number of net pins inside each chip is less than or equal to the number of I/O pins on the chip. (In this case, we say that the routing problem satisfies the *I/O pin capacity constraint*.) Surprisingly, we can prove that the above condition is also sufficient. Moreover, we designed an efficient polynomial-time algorithm to successfully route all nets if the I/O pin capacity constraint is satisfied.

Given an assignment of nets to the pin subset types. We define the degree of imbalance $\Phi(v)$ at a chip v to be the minimum number of nets that need to be re-assigned so that no more than m nets in v are assigned to the same pin subset type, i.e., $\Phi(v) = \sum_{T_i(v)>m}(T_i(v) - m)$, where $T_i(v)$ denotes the number of nets in chip v that are assigned to pin subset type T_i. When the degree of imbalance at every chip is 0, we have a feasible solution. Our strategy is to reduce the degree of imbalance at each chip iteratively until it becomes 0 without increasing the degree of imbalance at other chips.

Suppose we have an assignment of nets to pin subsets for which some chip v_0 has a pin subset T_i assigned more than m nets, i.e., $T_i(v_0) > m$. Then chip v_0 must also have a pin subset T_j assigned less than m pins, i.e., $T_j(v_0) < m$. We show how to balance the number of nets assigned to pin subset types T_i and T_j to reduce the degree of imbalance at chip v_0.

Consider those nets assigned to pin subset types T_i and T_j. We form a graph G_{ij} as follows. We represent each chip as a distinct vertex. And for each net assiged to pin subset types T_i or T_j, we represent it by a distinct edge that connect the two vertices corresponding to the chips sharing the nets. So G_{ij} is a multigraph and the number of edges between any two vertices is equal to the number of nets shared by the two corresponding chips that are assigned to pin subset types T_i or T_j. Let H_{ij} be the connected component of G_{ij} that contains vertex v_0 (the vertex corresponding to chip v_0). The idea is to traverse all the edges of H_{ij} once and label the edges alternately with i and j. If an edge is labeled i (j), we re-assign the corresponding net to pin subset type T_i (T_j). In the following, we prove two useful lemmas before presenting our algorithm. When we say "a chip in H_{ij}" and "a net in H_{ij}", we mean a chip that has a corresponding vertex in H_{ij} and a net that has a corresponding edge in H_{ij} respectively.

Lemma 1 *We can label the edges of $H_{ij} = (V, E)$ with i and j such that*

(i) *$|i(v) - j(v)| = 0$ for all $v \in V$, if $|E|$ is even and all vertices have even degree;*

(ii) $|i(v) - j(v)| \leq 1$ *for all* $v \in V$, *if some vertex has odd degree;*

(iii) *for an arbitrary vertex* $u \in V$, $|i(u) - j(u)| = 2$, *and* $|i(v) - j(v)| = 0$ *for all* $v \in V - \{u\}$, *if* $|E|$ *is odd and all vertices have even degree*

where $i(v)$ *and* $j(v)$ *denote the number of edges on vertex* v *labeled* i *and the number of edges on vertex* v *labeled* j, *respectively.*

Proof: First we prove (i) and (iii). If all vertices have even degree, then H_{ij} has an Euler circuit. We can traverse an Euler circuit starting and ending at an arbitrary vertex u and label the edges alternately with i and j along the way. It is clear that $|i(v) - j(v)| = 0$ for all $v \in V - \{u\}$. And if $|E|$ is even, the first and the last traversed edges will get different labels, so $|i(u) - j(u)| = 0$. This proves (i). On the other hand, if $|E|$ is odd, the first and the last traversed edges will get the same label, so $|i(u) - j(u)| = 2$. This proves (iii).

Now we prove (ii). If some vertex in H_{ij} has odd degree, the number of odd-degree vertices must be even (since $\sum_{v \in V} degree(v) = 2|E|$). So we can add a new vertex w and join every odd-degree vertex in H_{ij} to w by an edge to get a graph $H'_{ij} = (V', E')$ whose vertices are all of even degree. We consider two cases.
Case 1: $|E|$ is even. Then $|E'|$ is also even. So case(i) applies to H'_{ij} and we can label the edges of H'_{ij} so that $|i(v) - j(v)| = 0$ for all $v \in V'$. Removing all the edges on w, we obtain an edge-labeling for H_{ij} such that $|i(v) - j(v)| = 0$ for all even-degree vertex v in H_{ij}, and $|i(v) - j(v)| = 1$ for all odd-degree vertex v in H_{ij}.
Case 2: $|E|$ is odd. Then $|E'|$ is also odd. So case(iii) applies to H'_{ij} and we can label the edges of H'_{ij} so that $|i(w) - j(w)| = 2$ (pick w as the arbitrary vertex) and $|i(v) - j(v)| = 0$ for all $v \in V' - \{w\}$. Removing all the edges on w, we obtain an edge-labeling for H_{ij} such that $|i(v) - j(v)| = 0$ for all even-degree vertex v in H_{ij}, and $|i(v) - j(v)| = 1$ for all odd-degree vertex v in H_{ij}.
In both cases, $|i(v) - j(v)| \leq 1$ for all vertex $v \in V$. □

Lemma 2 *We can balance the number of nets assigned to pin subset types* T_i *and* T_j *so that for any chip* v *in* $H_{ij} = (V, E)$,

(i) $T_i(v) \leq m$ *and* $T_j(v) \leq m$ *when* $T_i(v) + T_j(v) \leq 2m$;

(ii) $T_i(v) \geq m$ *and* $T_j(v) \geq m$ *when* $T_i(v) + T_j(v) \geq 2m$

if m *is even.*

Proof: If $|E|$ is even or there exists an odd-degree vertex in H_{ij}, we can label all the edges of H_{ij} with i and j such that $|i(v) - j(v)| \leq 1$ by Lemma 1(i)(ii). Assigning all the nets in H_{ij} that have a corresponding edge labeled i (j) to pin subset type T_i (T_j), the result follows.

Now, consider the case $|E|$ is odd and all vertices in H_{ij} have even degree. Then there exists a vertex w such that $degree(w) \leq 2m - 2$ or $degree(w) \geq 2m + 2$. Otherwise, all vertices must be of degree $2m$ (since all vertices have even degree); this implies $2|E| = \sum_{v \in V} degree(v) = 2m|V|$, i.e., $|E| = m|V|$, which is impossible as m is even but $|E|$ is odd. So if we pick w with $degree(w) \leq 2m-2$ or $degree(w) \geq 2m+2$ as the arbitrary vertex in Lemma 1(iii), and assign all the nets in H_{ij} that have a corresponding edge labeled i (j) to pin subset type T_i (T_j), the result follows. □

Theorem 1 *For any BLRP, there exists a feasible solution using* $\lceil \Delta/m \rceil$ *pin subset types if* m *is even, where* Δ *denotes the maximum number of nets in a chip.*

Proof: Assum m is even. We show how to change an arbitrary assignment of nets to pin subsets that uses $\lceil \Delta/m \rceil$ pin subset types, so that each chip satisfies the condition that no more than m nets in it are assigned to the same pin subset.

Suppose we have an assignment of nets to pin subsets that uses $\lceil \Delta/m \rceil$ pin subset types for which some chip v_0 does not satisfy the condition, i.e., $T_i(v_0) > m$ for some pin subset T_i. Then we must have $T_j(v_0) < m$ for some other pin subset T_j among the $\lceil \Delta/m \rceil$ types used.

Consider those nets assigned to pin subset types T_i and T_j. We form a graph G_{ij} as follows. We represent each chip as a distinct vertex. And for each net assigned to pin subset types T_i or T_j, we represent it by a distinct edge that connect the two vertices corresponding to the chips that share the net. Let H_{ij} be the connected component of G_{ij} that contains vertex v_0 (the vertex corresponding to chip v_0). By Lemma 2, we can re-assign the nets in H_{ij} so that for any chip in H_{ij} the number of nets assigned to pin subset type T_i and the number of nets assigned to pin subset type T_j are both no more than m or both no less than m.

We can repeat this balancing process until no more than m nets in chip v_0 are assigned to the same pin subset type. (This happens in at most $\Delta - m$ iterations. Since $\Phi(v_0) = \sum_{T_i(v_0) > m} (T_i(v_0) - m)$ decreases each time and this happens when $\Phi(v_0)$ is zero.)

Now, if all chips satisfy the condition, we are done. Else we can repeat the same argument for a chip not satisfying the condition. Notice that nets in chip v_0 may then be re-assigned but v_0 will continue to satisfy the condition by Lemma 2(i). So repeating in this way, we will obtain an assignment which uses $\lceil \Delta/m \rceil$ pin subset types and in every chip there are no more than m nets assigned to the same pin subset type. □

Corollary 1 *Any BLRP with even subset size* m *is feasible.*

Now, we present the algorithm for solving the two-terminal net BLRP when the pin subset size m is even.

ALGORITHM 1

1 Assign all the nets to pin subset type T_1.
So for any chip v, $T_1(v) :=$ number of nets in v; and
$T_i(v) := 0$ for $i = 2, 3, \ldots, \lceil \Delta/m \rceil$ where Δ is the maximum number of nets in a chip.

2 Balance the number of nets assigned to each pin subset type.
For each chip v **do**
while there exists pin subset types T_i and T_j
s.t. $T_i(v) > m$ and $T_j(v) < m$ **do**
Represent each chip as a vertex, and
represent each net assigned to pin subset
types T_i or T_j as a distinct edge
joining the vertices corresponding to the chips
that share the net to obtain a multigraph;
Let H_{ij} be the connected component containing v;
if H_{ij} has some odd-degree vertex **then**
add a new vertex w to H_{ij} and join every
odd-degree vertex in H_{ij} to w by an edge;
$u := w$
else if H_{ij} has an odd number of edges **then**
$u :=$ a vertex in H_{ij} whose degree is not $2m$
else $u := v$;
Find an Euler circuit in H_{ij};
Traverse the Euler circuit starting at vertex u and
label the edges alternately with i and j along the way;
Re-assign each net that has a corresponding edge in H_{ij}
to pin subset type T_i if the edge is labeled i,
and to T_j otherwise;
Discard w (if it has been added) and all edges on it.

Figure 4(a) shows an instance of the BLRP. We show how our algorithm works on it. In this example, m, the pin subset size is 2; and Δ, the maximum number of nets in a chip is 6. So $\lceil \Delta/m \rceil = 3$. A solution is feasible if each chip satisfies the condition that no more than two nets in it are assigned to the same pin subset.

First, we assign all nets to pin subset type A as shown in Figure 4(b). We process the chips in the order v_1, v_2, v_3, v_4. Chip v_1 already satisfies the required condition. But $A(v_2) > 2 > B(v_2)$, so we form and label graph H^2_{AB} as shown in Figure 4(c). Then we re-assign the corresponding nets as in shown in Figure 4(d). After that, both chips v_1 and v_2 satisfy the required condition. For chip v_3, $A(v_3) > 2 > C(v_3)$, so we form and label graph H^3_{AC} as shown in Figure 4(e). And re-assign the corresponding nets as shown in Figure 4(f). Chip v_3 requires further processing since now $B(v_3) > 2 > A(v_3)$. So we form and label graph H^3_{BA} as shown in Figure 4(g), and re-assign the corresponding nets as shown in Figure 4(h). Now, chip v_3 also satisfies the required condition that no more than two nets in it are assigned to the same pin subset (note that this property is preserved at chips v_1 and v_2). Since chip v_4 already satisfies the condition, we are done.

Theorem 2 *ALGORITHM 1 finds a feasible solution for any two-terminal net BLRP with even subset size m in $O(n^2)$-time where n is the total number of nets.*

Proof: In each iteration, we can construct graph H_{ij} and find an Euler circuit in it in $O(n)$-time. And at most $2n$ iterations are required because the sum of the degrees of imbalance at all chips is no more than $2n$ at the beginning. Hence the result. □

Now, we consider the BLRP with odd pin subset size m. Note that we used only $\lceil \Delta/m \rceil (\leq K)$ types of pins in our algorithm. So some types of pins may not be used at all. If m is odd, we may limit ourselves to use at most $m-1$ pins of each pin subset type in every chip and use our algorithm to find a routing that uses $\lceil \Delta/(m-1) \rceil$ types of pins. So when m is odd, we can still apply our algorithm in this way to find an optimal solution if $\lceil \Delta/(m-1) \rceil \leq K$.

Theorem 3 *For any two-terminal net BLRP with odd subset size m such that $\lceil \Delta/(m-1) \rceil \leq K$, we can modify ALGORITHM 1 to find a feasible solution.*

We do not apply our algorithm directly when m is odd because in step 2, if H_{ij} has an odd number of edges, it is possible that all vertices in H_{ij} have degree $2m$ when m is odd. Moreover, not all instances of the BLRP with odd pin subset size m have a feasible solution. It is obvious that the simple instance in Figure 3 with m equals to 1 has no feasible solution.

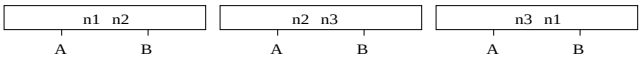

Figure 3: An instance of BLRP with odd subset size m

4 NP-completeness

In this section, we prove that the board-level routing problem with multi-terminal nets is NP-complete [11,12]. First, we use polynomial-time reduction from the Graph K-Colorability problem [11], a known NP-complete problem described below, to prove that the BLRP with pin subset size of 1 is NP-complete. Then, we show that the BLRP with pin subset size of 1 is polynomial-time reducible to the BLRP with pin subset size of m for any $m \geq 1$. Hence, for any $m \geq 1$, the BLRP with pin subset size of m is NP-complete.

The Graph K-Colorability problem is as follows. Given a graph $G = (V, E)$ and a positive integer $K \leq |V|$, determine whether there is an assignment of colors to the vertices such that each vertex is assigned one color, and no two adjacent vertices have the same color, and the total number of color used is no more than K. Such an assignment is called a K-coloring.

Theorem 4 *The BLRP with pin subset size of 1 is NP-complete.*

Proof: It is clear that the BLRP with pin subset size of 1 belongs to NP.

To show that Graph K-colorability is polynomial-time reducible to the BLRP with pin subset size of 1. Given an instance $G = (V, E)$ of the Graph K-colorability problem, we construct an instance of the BLRP with pin subset size of 1 as follows. For each edge (u, v) in G, we form a K-pin chip that contains only two nets, n_u and n_v. There are K size-1 pin subsets per chip. It is easy to see that the graph G has a K-coloring if and only if the constructed routing problem has a feasible solution. □

Theorem 5 *The BLRP with pin subset size of m is NP-complete for any $m \geq 1$.*

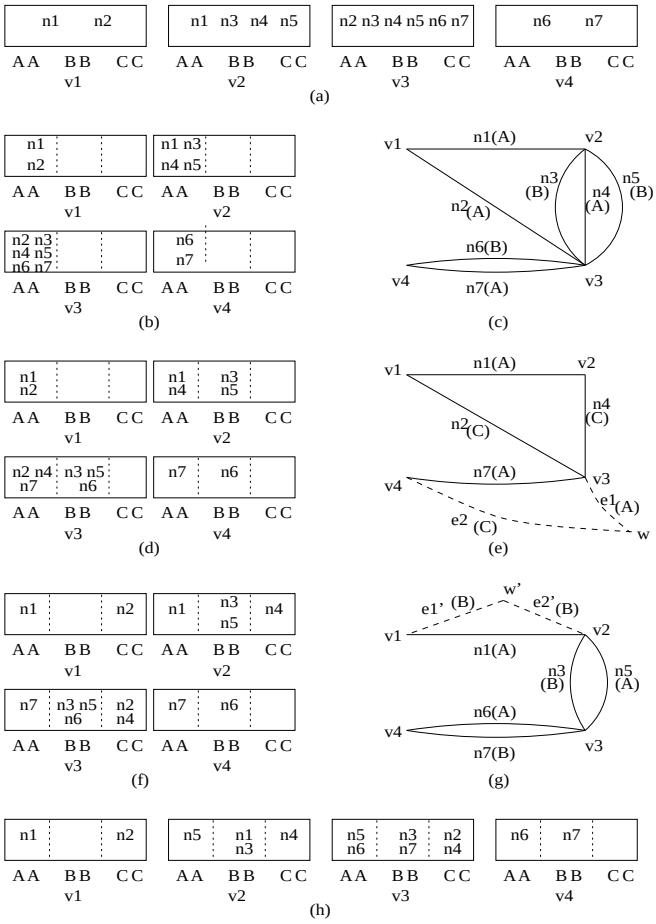

Figure 4: (a) An instance of BLRP. (b) An initial assignment of nets to pin subsets. (c) Graph H^2_{AB} with Euler circuit $n_1n_3n_4n_5n_7n_6n_2$. (d) Re-assigning some nets according to the labels in H^2_{AB}. (e) Graph H^3_{AC} with an Euler circuit $e_1n_4n_1n_2n_7e_2$. (f) Re-assigning some nets according to the labels in H^3_{AC}. (g) Graph H^3_{BA} with an Euler circuit $e'_1n_1n_3n_6n_7n_5e'_2$. (h) A feasible solution otained after re-assigning some nets according to the labels in H^3_{BA}.

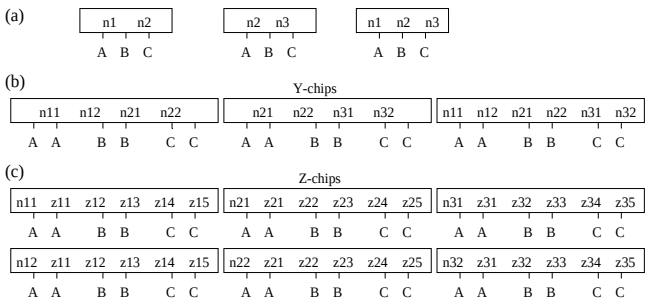

Figure 5: Reducing a routing problem with subset size 1 to one with subset size 2. (a) A problem with subset size 1. (b),(c) Construction of two kinds of chips of the corresponding problem with subset size 2.

Proof: We use $R(i)$ to denote the BLRP with pin subset size of i. Suppose we have a fixed $m \geq 1$.

It is clear that $R(m)$ belongs to NP.

To show that $R(1)$ is polynomial-time reducible to $R(m)$. Given an instance $I(1)$ of $R(1)$, we construct an instance $I(m)$ of $R(m)$ as follows.

Suppose that every chip in $I(1)$ has K pin subsets, S_1 to S_K. All chips we form in $I(m)$ also have K pin subsets, T_1 to T_K, and of course each has a size of m.

For each chip X in $I(1)$, we form a chip Y in $I(m)$ so that if net n_i belongs to chip X, then nets n_{i1}, n_{i2}, $\ldots$, n_{im} belong to chip Y. So the number of nets in chip Y is m times that in chip X. See Figure 5(a)(b).

And for each net n_i that appears in $I(1)$, we form m chips, Z_{i1}, Z_{i2}, $\ldots$, Z_{im}. Chip Z_{ij} $(j = 1, \ldots, m)$ contains nets z_{i1}, z_{i2}, $\ldots$, $z_{i(mK-1)}$, and n_{ij}. See Figure 5(a)(c). Hence chips Z_{i1}, Z_{i2}, $\ldots$, Z_{im} all contain the maximum possible number of nets. Thus we are forcing n_{i1}, n_{i2}, $\ldots$, n_{im} to be assigned to the same pin subset type in any feasible solution.

It can be shown that $I(1)$ has a feasible solution if and only if $I(m)$ has a feasible solution [13]. □

References

[1] S.D. Brown, R.J. Francis, J. Rose, and Z.G. Vranesic, *Field-Programmable Gate Arrays*, Kluwer Academic Publishers, 1992.

[2] S. Trimberger (edited), *Field-Programmable Gate Array Technology*, Kluwer Academic Publishers, 1994.

[3] J. Varghese, M. Butts, and J. Baatcheller, "An Efficient Logic Emulation System", *IEEE Transactions on VLSI, Vol. 1, No. 2*, June 1993, 171-174.

[4] M. Slimane-Kadi, D. Brasen, and G. Saucier, "A Fast-FPGA Prototyping System that Uses Inexpensive High-Performance FPIC," *ACM/SIGDA International Workshop on Field-Programmable Gate Arrays*, Feb 1994.

[5] L. Maliniak, "Multiplexing Enhances Hardware Emulation," *Electronic Design*, Nov. 1992, 76-78,

[6] S. Walters, "Computer-Aided Prototyping for ASIC-based Systems," *IEEE Design and Test*, June 1991, 4-10.

[7] K. Yamada, H. Nakada, A. Tsutsui, and N. Ohta, "High-Speed Emulation of Communication Circuits on a Multiple-FPGA System," *ACM/SIGDA International Workshop on Field-Programmable Gate Arrays*, Feb 1994.

[8] N.C. Chou, L.T. Liu, C. K. Cheng, W.J. Dai, and R. Lindelof, "Circuit Partitioning for Huge Logic Emulation Systems," *31st ACM/IEEE Design Automation Conference*, 1994.

[9] H.H. Yang, and D. F. Wong, "Area/Pin-Constrained Circuit Clustering for Delay Minimization," *ACM/SIGDA International Workshop on Field-Programmable Gate Arrays*, Feb 1994.

[10] C. Berge, *The Theory of Graphs and Its Applications*, John Wiley and Sons, 1962.

[11] M.R. Garey, and D.S. Johnson, *Computers and Intractability, A Guide to the Theory of NP-completeness*, W.H. Freeman, 1979.

[12] T.H. Cormen, C.E. Leiserson, and R.L. Rivest, *Introduction to Algorithms*, McGrawHill, 1990.

[13] W.K. Mak, and D.F. Wong, *On Optimal Board-Level Routing for FPGA-based Logic Emulation*, Technical Report, TR94-30, Department of Computer Sciences, University of Texas at Austin, 1994.