Post-floorplanning Power Optimization for MSV-driven Application Specific NoC Design

Kan Wang, Sheqin Dong
Department of Computer Science & Technology
Tsinghua University, Beijing, China 100084
Email: wangkan09@mails.tsinghua.edu.cn, dongsq@mail.tsinghua.edu.cn

Abstract—In this paper, a transitive closure graph (TCG)-based post-floorplanning repacking algorithm is proposed for power optimization of Multiple Supply Voltages (MSV)-driven application specific 2D NoC. Through a five-stage processing including floorplan representation, preconditioning, objective insertion point calculation, TCG modification and ILP-based incremental repacking, the total communication power can be optimized without changing much of the original floorplan. Experimental results show that the proposed method can reduce the communication power by 28.3%.

I. INTRODUCTION

Network-on-chip (NoC) has become more and more important as it can effectively reduce the communication cost on chip [1]-[3]. However, with the increasing integration of intellectual properties, high power density is created, which leads to power waste, thermal problems, and even hardware mistakes. Power consumption has become one of the major challenges in current NoC design.

The power consumption of NoC mainly consists of two parts: core power and communication power. The core power is caused by the computing of cores and operating of network components while the communication power is caused by communications between cores and network components. One effective technology to reduce the core power is Multiple Supply Voltage (MSV) [4]-[7], which partitions the whole circuit into domains of different voltage levels. Each voltage domain is called a Voltage Island (VI). By assigning lower voltage to some cores, the total power can be reduced.

Designers in recent years applied MSV to 2D NoC [8]-[10] to reduce total power. However, all of them paid attention to regular NoC design and few of them considered the MSV technology in the design of application specific 2D NoC (ASNoC). Different from regular NoC, ASNoC design is more difficult due to irregular core sizes, which involves issues of both floorplanning and network components (NC) assignment. Although the researches [11]-[13] proposed voltage island-driven floorplanning algorithms, they only focused on SoC and did not consider the assignment of NCs. A recent work [14] proposed a voltage island-driven floorplanning considering NC assignment for ASNoC. However, as a result of the randomness of disturbing during floorplanning, the positions of network components cannot be optimized. Although a post-floorplanning dead-space re-allocation algorithm was used to improve the positions of network components in [14], it still cannot optimize the communication power because it just adjusted the positions of network components without changing the original floorplan. Since the network components are placed in the dead-space, the positions will be highly restricted by the floorplan and a bad floorplan can lead to large communication power. Fig.1 shows an example. The dashed line in figure stands for the bounding box of voltage island with core 1, 2, 3 and 4. Network components are required to be inserted into the dead-space in this bounding box. If the floorplan is a good one, as shown in Fig.1 (a), NCs can be inserted to a good position. But if a floorplan is a tighter one, as shown in Fig.1 (b), the NCs will be inserted far from the best position, leading to larger communication distance and larger communication power. However, it is hard to directly generate such good floorplans from traditional floorplanner.

To address this issue, in this paper, a transitive closure graph (TCG)-based post-floorplanning repacking algorithm (TPRA) is proposed for the design of MSV-driven ASNoC. Through a five-stage processing including floorplan representation, preconditioning, objective insertion point calculation, TCG modification and ILP-based incremental repacking, the positions of NCs and the total communication power can be optimized without changing much of the original floorplan.

The rest of the paper is organized as follows. Section II introduces the overall design flow. Section III describes the proposed algorithm in detail. Experiment results are shown in Section IV and conclusions are provided in Section V.

II. OVERALL DESIGN FLOW

The main idea of TPRA is to formula the relative positions of cores and network components by using a representation
we just need to modify the hgraph by this, to insert components into the dead-space among cores, then the corresponding floorplan will be also changed. Inspired the corresponding TCG are one-to-one correspondence. If we direction and vertical direction respectively. A floorplan and which represent the relative positions of cores in horizontal extent that the core can be moved towards the right or upward edge from

Fig. 2: Overall flow of TPRA

and incrementally repack inserting the network components to good positions without overlaps with cores. For this, the TCG representation is used because it can easily represent the relative positions between cores [15].

The overall flow of TPRA is shown in Fig. 2. Given the initial floorplan, generates the corresponding TCG first. Then calculate the slacks of all cores and generate dead-spaces. After that, calculate the objective insertion points for network components. Then for each network component, find the border nodes in TCG and update the corresponding TCG by inserting the nodes of network components into it. Finally, an ILP-based repacking algorithm is proposed to incrementally insert the components to the objective insertion positions. More details are introduced in the following sections.

III. DETAILED ALGORITHM

Before we present the detailed algorithm, some notations are defined for the algorithm first, as shown in Table I.

A. Floorplan representation

The TCG representation [15] describes the geometric relations among cores based on two graphs of hgraph and vgraph which represent the relative positions of cores in horizontal direction and vertical direction respectively. A floorplan and the corresponding TCG are one-to-one correspondence. If we modify the relative positions of cores in hgraph or vgraph, then the corresponding floorplan will be also changed. Inspired by this, to insert components into the dead-space among cores, we just need to modify the hgraph and vgraph by inserting additional nodes and the corresponding directed edges in graphs. More details about how to modify TCG and how to pack with a given TCG can be obtained from [15].

B. Preconditioning

1) Core slack calculation: The slack of a core denotes the extent that the core can be moved towards the right or upward without changing the maximum width or maximum height of the floorplan. For each core \( i \), there are two slacks: \( hslack_i \) in horizontal direction and \( vslack_i \) in vertical direction. To calculate \( hslack_i \), first find all cores which have a directed edge from \( i \) in hgraph, referred as core set \( HOUT_i \), and then calculate the \( hslack_i \) as follows:

\[
hslack_i = \min_{k \in HOUT_i} \{hslack_k + x_k - rx_i\}. \tag{1}\]

Similarly, \( vslack_i \) can be obtained according to:

\[
vslack_i = \min_{k \in VOUT_i} \{vslack_k + y_k - ry_i\} \tag{2}\]

where \( VOUT_i \) is the set of cores which have a directed edge from \( i \) in vgraph. The core slack calculation is performed from right to left in hgraph and from top to bottom in vgraph. If there are no cores in \( HOUT_i \) or \( VOUT_i \), then use the border line of the floorplan instead.

2) Dead-space generation: After core slack calculation, the on-chip dead-spaces are then searched from the given floorplan. This is easy to achieve by using a linear scanning algorithm. In order not to generate narrow dead-space, we add a constraint that the ratio of width/height of the dead-space should be between 0.1 and 10 when combining small dead-spaces into large ones.

C. Objective insertion point calculation

The objective insertion points are the positions for each component so that the total communication power is optimized. They are calculated by the following objective function:

\[
\min \sum_{i \in N_c} \sum_{j \in N_c} com_{i,j} \cdot length_{ij} + \sum_{k \in N_c} \sum_{l \in N_c} com_{k,l} \cdot length_{kl}. \tag{3}\]

By solving the LP problem, the coordinates of each objective insertion point can be obtained. We do not discuss the details of the problem due to the page limitation.

D. TCG modification

The objective insertion points are the ideal positions for each component. However, the positions are not available if they are occupied by cores. In this section, the actual insertion positions in TCG are determined.

Considering that the objective insertion points may be within a dead-space or a core, the inserting process satisfies the following strategies: 1) if the objective position \( P_c \) of component \( c \) is in a dead-space \( ds \), then we find all cores adjacent with \( ds \). Let \( cleft_{ds} \), \( cright_{ds} \), \( cdrown_{ds} \) and \( cup_{ds} \) denote the set of cores which are adjacent with \( ds \) and are on the left, right, below and above of \( ds \) respectively. Then for each core \( i \) in \( cleft_{ds} (cdrown_{ds}) \), insert a directed edge from \( i \) to \( c \) in hgraph (vgraph); for each core \( j \) in \( cright_{ds} (cup_{ds}) \), insert a directed edge from \( c \) to \( j \) in hgraph (vgraph). 2) if \( P_c \)

\begin{table}[h]
\centering
\caption{Notations of TPRA}
\begin{tabular}{|c|c|}
\hline
\textbf{N} & The set of all cores \\
\hline
\textbf{N_c} & The set of all network components \\
\hline
\textbf{x}, \textbf{y} & The coordinate of left lower corner of core or component \( i \) \\
\hline
\textbf{w}, \textbf{h} & The width and height of core or network component \( i \) \\
\hline
\textbf{rx}, \textbf{ry} & The coordinate of right upper corner of core or component \( i \) \\
\hline
\textbf{length}_{ij} & The distance between \( i \) and \( j \) \\
\hline
\textbf{hgraph} & The transitive closure graph in horizontal direction \\
\hline
\textbf{vgraph} & The transitive closure graph in vertical direction \\
\hline
\end{tabular}
\end{table}
is not in dead-space but overlapped with core oc, then search for all dead-spaces adjacent with oc and select the one which has enough space and is nearest to the objective position. 3) If there is no dead-space around oc, then select the direction nearest to Pc and with largest slack and then insert c to the direction. 4) if ds or oc is next to the border of the floorplan, then the edges from s (next to the left or bottom border) to c or the edges from c to t (next to the right or top border) are inserted to the corresponding graphs.

Fig. 3 shows an example. In Fig. 3(a), the objective insertion position of component c is in the dead-space among 1, 2, 3 and 5, then the component c will be inserted into the dead-space directly. The corresponding TCG is shown in Fig. 3(b). Edge 1 → c and c → 5 are added in hgraph and edge 2 → c and c → 3 are added in vgraph. Fig. 4 shows another example. In Fig. 4(a), Pc is within core 4 and next to the bottom left corner of 4. As vslack4 is larger than hslack4, instead of being inserted to the left of core 4 (Fig. 4(b)), c is inserted to the lower space of core 4 by pushing up by 4 the height of c (Fig. 4(c)). For the case that Pc is next to the top right corner of oc, similar methods can be used and we do not discuss in detail here.

E. ILP-based repacking

After that, an ILP-based repacking algorithm is proposed to incrementally insert the network components to the ideal positions according to the modified TCG.

There are several constraints which should be considered in the repacking phase. First, the inserted network components should satisfy the boundary constraint derived from the modified TCG. Second, the relative positions between cores should be maintained because the modified relative positions can greatly change the floorplan and even enlarge the voltage islands resulting in the increase of design cost. Besides, the cores and network components should satisfy the non-overlap constraint. Therefore, the repacking problem can be formulated as the following optimization problem:

\[
\begin{align*}
M \text{ in } & \omega \cdot \sum_{c \in \mathcal{N}_c} (|x_c - px_c| + |y_c - py_c|) \\
& + \sum_{i \in \mathcal{N}} (|x_i - ix_i| + |y_i - iy_i|) \\
\text{subject to:} \\
& x_i + w_i \leq x_c, \forall e_{i,c} \in E_h, i \in \mathcal{N}, c \in \mathcal{N}_c \\
& x_c + w_c \leq x_j, \forall e_{c,j} \in E_h, c \in \mathcal{N}_c, j \in \mathcal{N} \\
& y_k + h_k \leq y_c, \forall e_{k,c} \in E_v, k \in \mathcal{N}, c \in \mathcal{N}_c \\
& y_c + h_c \leq y_l, \forall e_{c,l} \in E_v, c \in \mathcal{N}_c, l \in \mathcal{N} \\
& x_i + w_i \leq x_j, \forall e_{i,j} \in E_h, i \in \mathcal{N}, j \in \mathcal{N} \\
& y_k + h_k \leq y_l, \forall e_{k,l} \in E_v, k \in \mathcal{N}, l \in \mathcal{N} \\
& X_{i,j} + X_{j,i} + Y_{i,j} + Y_{j,i} \geq 1, \forall i,j \in \mathcal{N} \bigcup \mathcal{N}_c.
\end{align*}
\]

The objective of Expression (4) is to minimize the total communication cost without changing much of the original floorplan, which is evaluated by the sum of all the distances deviated from the best positions of components and initial positions of cores respectively. The wire length is evaluated by using the half perimeter model. ω is used to adjust the relative weights between the two costs. In this paper, ω is set to be 0.1. The constraints (5)-(8) guarantee that the network components satisfy the boundary constraint and are inserted into the positions derived from the modified TCG. The constraints (9)-(10) ensure that the relative positions between cores remain unchanged. The constraint (11) ensures that there are no overlaps between cores and component components.

\[X_{i,j} = \begin{cases} 1, & \text{if } x_i \geq x_j \\
0, & \text{if } x_j < x_i \end{cases} \]

and can be obtained by the following linear equations:

\[
\begin{align*}
x_i - r x_j - X_{i,j} \cdot M & < 0 \\
x_j - x_i - (1 - X_{i,j}) \cdot M & \leq 0
\end{align*}
\]

where M is a very large integer. X_{i,j}, Y_{i,j} and Y_{j,i} can be obtained similarly. The constraint (11) ensures that i and j satisfy the non-overlap constraint because that at least one of the following four conditions is true:

\[x_i \geq r x_j, \quad x_j \geq r x_i, \quad y_i \geq r y_j, \quad y_j \geq r y_i. \]

The following constraints can be used to remove the absolute value signs in the objective function (4):

\[
\begin{align*}
x_c - px_c & \leq x'_c, \quad -x_c + px_c \leq x'_c \\
y_c - py_c & \leq y'_c, \quad -y_c + py_c \leq y'_c \\
x_i - ix_i & \leq x'_i, \quad -x_i + ix_i \leq x'_i
\end{align*}
\]
Then the optimization problem can be re-written by:

\[
\begin{aligned}
\text{Min } & \omega \cdot \sum_{c \in C_e} (x'_c + y'_c) + \sum_{i \in N} (x'_i + y'_i) \\
\text{Subject to } & (5) - (11), (13), (15) - (18).
\end{aligned}
\]  

By solving the ILP problem, the final positions of cores and network components will be determined.

### IV. Experimental Results

Experimental results show that the proposed algorithm is effective on power reduction and design cost optimization. All experiments are performed on a workstation with 3.0 GHz CPU and 4GB physical memory. Nine benchmarks are used from [14] in the experiments. *lp_solve* 5.5.2.0 [16] is used for LP and ILP solving. To make fair comparisons, the same power model is used as [14].

In order to show the effectiveness of our method, we compare TPRA with the algorithm in [14], as shown in Table II. The column *Com Power* means the communication power consumption and column *Wire Length* means the total wire-length for communications, while column *Hops* denotes the average number of hops and column *Dead-space Ratio* denotes the ratio of total dead-space area over the chip area. The communication power and wire-length of all benchmarks are all normalized values. Compared to [14], the proposed post-floorplanning repacking algorithm is very effective and can reduce the communication power by 50.5% at most and 28.3% on average. The dead-space is increased by 9.0% on average because of the strategy to adjust the floorplan for NCs insertion. However, for the large benchmarks such as *D_38_tvopd*, the increase of dead-space becomes very small. Due to the ILP solving, the runtime is much larger than [14]. However, for the largest benchmark, it can still solve in 10 seconds. Compared to the significant improvement, the little deteriorating on wire-length (1.5%) is acceptable.

Fig. 5 shows the generated floorplan of *D_38_tvopd*, which are composed of three voltage islands.

### V. Conclusion

In this paper, a TCG-based post-floorplanning repacking algorithm is proposed for MSV-driven ASNoC. Through a five-stage method, the total communication power can be optimized without changing much of the original floorplan.