STOCHASTIC CONGESTION MODEL FOR VLSI SYSTEMS

Patrick Hung and Michael J. Flynn

Technical Report No. CSL-TR-97-737

October 1997

This work was supported by NSF under contract MIP93-13701.

STOCHASTIC CONGESTION MODEL FOR VLSI SYSTEMS

by

Patrick Hung and Michael J. Flynn

Technical Report No. CSL-TR-97-737

October 1997

Computer Systems Laboratory

Stanford University

Gates Building 3A, Room 332

Stanford, California 94305-9030

pubs@shasta.stanford.edu

Abstract

Designing with deep submicron feature size presents new challenges in complexity, performance, and productivity. Information on routing congestion and interconnect area are critical in the pre-RTL stage in order to forecast the whole die size, define the timing specifications, and evaluate the chip power consumption.

In this report, we propose a stochastic model for VLSI interconnect routing, which can be used to estimate the routing congestion and the interconnect area in the pre-RTL stage. First, we define the uniform and geometric routing distributions, and introduce a simple and efficient algorithm to calculate the routing probabilities. We then derive the routing probabilities among multiple functional blocks, and investigate the effects of routing obstacles. Finally, we map the chip to a Cartesian coordinate system, and model routability based on the supply and demand distributions of routing channels.

Key Words and Phrases: VLSI, pre-RTL, Uniform Routing Distribution, Geometric Routing Distribution, Probabilistic Routing Algorithm

Copyright © 1997
by
Patrick Hung and Michael J. Flynn

Contents

List of Figures

1 INTRODUCTION

1.1 MOTIVATION

Designing with deep submicron feature size presents new challenges in complexity, performance, and productivity. Modern state-of-the-art microprocessors contain more than ten million transistors and the cycle times are generally less than five nanoseconds. For example, the Intel Pentium II microprocessor has more than 17.5 million transistors and runs at 233MHz (4.3ns) [Mic97, Ass94].

While device delay decreases with smaller feature size, interconnect delay does not necessarily go down because the die size remains roughly the same. Consequently, interconnect delay plays a dominant role in the timing specifications, and chip floorplanning is crucial in determining the chip performance. The most alarming point is that with conventional design methodology, it takes more and more iterations between logic design and layout to fix the timing and area problems.

Information on the routing congestion and the interconnect lengths are critical in the pre-RTL stage in order to forecast the whole die size, define the timing specifications, and evaluate the chip power consumption. Early accurate estimation can dramatically reduce the number of iterations to fix the timing problems [Fle94, Goe94].

To reduce the complexity of the design, the chip is first partitioned into functional blocks. The connections among the functional blocks are defined but the port locations are usually not fixed because the functional blocks are not yet designed. At this point the design is by no means cast in concrete, but we would attempt to answer the following questions:

- Is there any routing congestion areas?
- What is the total die size?
- What are the impacts when technology changes?

1.2 CONVENTIONAL APPROACH

In the pre-RTL stage, chip floorplanning is traditionally done manually by system architect. The sizes of the functional blocks are estimated from the previous designs or some area model [Fly95]. The functional blocks are then placed based on experience and the timing requirements. In order to estimate whether the chip is routable, the locations of the ports are arbitrarily assigned in each functional block. The global router is then run to see if the chip is routable [PL88, SK94, SBP95].

The traditional approach is simple and straightforward, but it suffers from several major drawbacks. First, since the port locations are arbitrarily assigned in the pre-RTL stage, the actual routing may be completely different from the pre-RTL routing. The chip may be routable in the pre-RTL stage, but not routable in the final design. Second, it does not provide any useful insights when technology changes. Third, this approach depends heavily on the routing sequence. In other words, the chip may be routable if some wires are routed before the other wires. This is undesirable because the actual routing sequence in the final stage will likely to be different. Finally, the pre-RTL routing can be very time consuming and does not provide immediate feedbacks to help system architects optimize their floorplan.

1.3 RELATED WORK

In 1960 E. F. Rent proposed a simple relationship between “number of pins” versus “number of circuits” in a logic design based on empirical data collected in IBM [LR71].

$$N_p = K_p {N_g}^{\pi}$$

This equation is called Rent's rule, where N_p is the number of pins, N_g is the number of gates, π is Rent's constant, and K_p is a proportionality constant.

Rent's rule was later applied to derive the average interconnection length and expected wiring area of a design given the number of gates and gate pitch [Don79, Gam81, Sch82, HHM84, Fer85]. Recently, Rent's rule was used to estimate the optimum interconnect dimensions for low power applications [DDM96].

Since the parameters used in Rent's rule are determined from empirical data, it is useful to predict the pin requirements and average interconnection lengths of future designs provided that the architecture remains unchanged. On the contrary, if the architecture or design methodology changes, the results will not be accurate [Bak87].

Other researchers have developed analytical expression for the routability of the circuits in the Field-Programmable Gate Arrays (FPGA) [SP86, BRV93]. However, their models are limited to the FPGA and do not apply to general VLSI design. In particular, the problems of floorplanning and block placement were not addressed. These issues are crucial in deep submicron design.

1.4 OUR APPROACH

In this report, we propose a stochastic model for VLSI interconnect routing.

First, we define the routing probabilities between two points in a Cartesian coordinate system. The routing probabilities are derived based on uniform and geometric routing distributions. The uniform routing distribution assumes all possible routes between the two points are equally probable. However, most modern routers prefer routing with minimum vias in order to reduce interconnect delay. The geometric routing distribution takes the router preference into consideration. A simple and efficient algorithm is proposed to calculate the routing probabilities.

Second, we derive routing probabilities among multiple functional blocks, and investigate the effects of routing obstacles. Our algorithm is enhanced to handle the multiple blocks and routing obstacles.

Finally, we compress multiple metal layers into a single logical layer and map the chip into a Cartesian coordinate system. We model routability based on the supply and demand distributions of routing channels. The idea is that a chip is routable if and only if the supply exceeds the demand everywhere on the die.

2 UNIFORM ROUTING DISTRIBUTION

2.1 ROUTING BETWEEN TWO POINTS

In a Cartesian coordinate system, consider a route from point A at $(0,0)$ to point B at (m,n). Assuming that there is no routing obstacles between A and B and the route can either run horizontally or vertically at each grid point, the shortest routing distance is equal to $m+n$. This is called the Manhattan distance between A and B.

If the routing is limited to the Manhattan distance, there are ${}_{m+n}C_m$ different number of ways to route from A to B. The combination ${}_{m+n}C_m$ is derived from the fact that there are $m+n$ "tracks" between A and B, out of which m "tracks" have to go horizontal [Fel70].

As shown in Figure 1, there are ${}_4C_2 = 6$ different ways to route from A to B for $m = n = 2$.

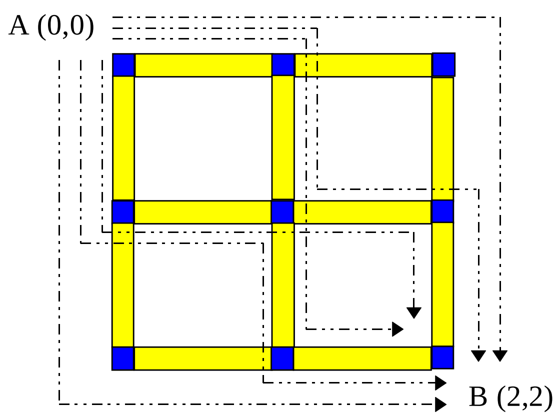


Figure 1: Routing between Two Points

The number of routes between A and B increases when A and B are further apart. Actually, the number of routes forms a two-dimensional Pascal triangle (Figure 2).

2.2 DERIVATION

Suppose all the ${}_{m+n}C_m$ routes have **equal** probabilities. The routing from point A to point B is said to follow uniform routing distribution.

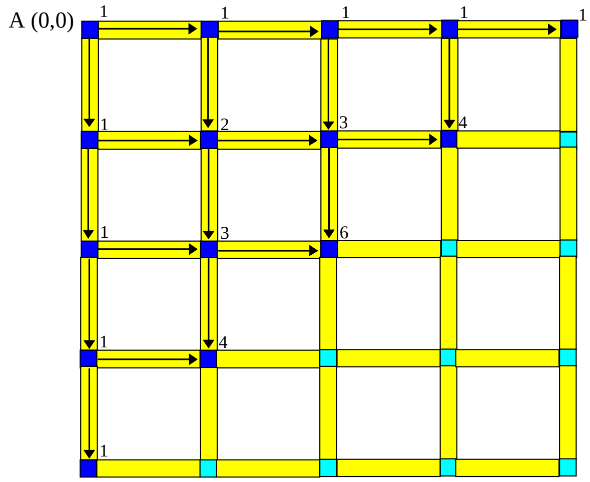


Figure 2: Routing Combinations

First, we can derive the probability that the route goes through a point P at coordinate (x, y) where $0 \leq x \leq m$ and $0 \leq y \leq n$ (Figure 3).

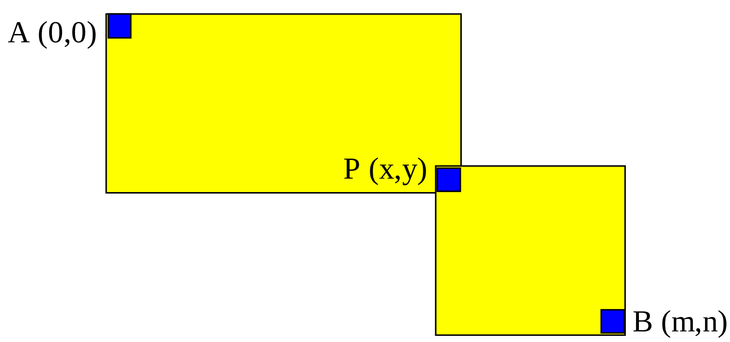


Figure 3: Routing through a Point (Uniform Routing Distribution)

The number of routes from A to B that pass through point P is equivalent to the number of routes from A to P multiplied by the number of routes from P to B, which is equal to:

$$_{x+y}C_x \times {}_{m+n-x-y}C_{m-x}$$

Hence, the probability that the route goes through point P is:

$$_{x+y}C_x \times {}_{m+n-x-y}C_{m-x} \div {}_{m+n}C_m$$

Second, we can derive the probability that the route passes through any particular track (Figure 4).

The number of routes from point A to point B that pass through (x, y) and then $(x + 1, y)$ is:

$_{x+y}C_x \times {}_{m+n-x-y-1}C_{m-x-1}$ or

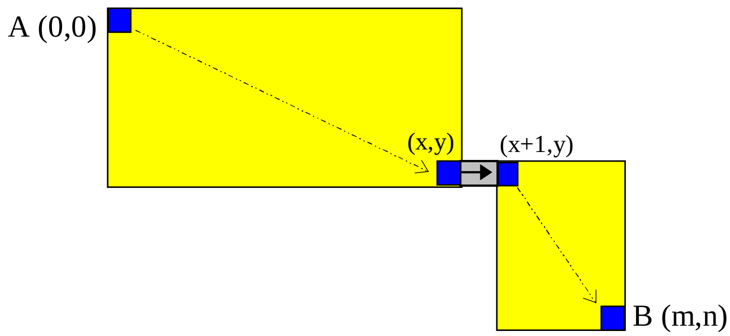


Figure 4: Routing through a Horizontal Track (Uniform Routing Distribution)

$$ {}_{x+y}C_x \ \times \ {}_{m+n-x-y-1}C_{n-y} $$

Hence, the probability that the route passes through (x, y) and then $(x + 1, y)$ is:

$$ {}_{x+y}C_x \ \times \ {}_{m+n-x-y-1}C_{n-y} \ \div \ {}_{m+n}C_m $$

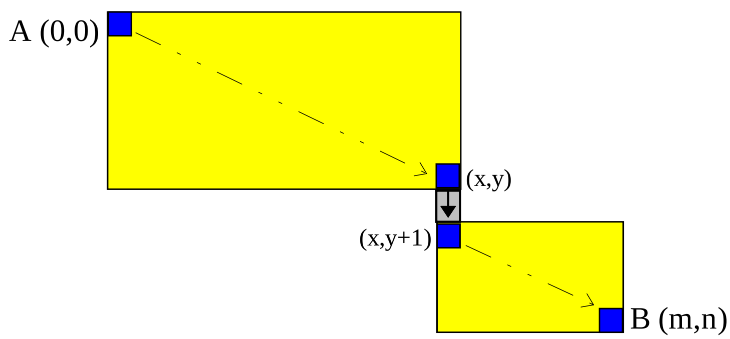


Figure 5: Routing through a Vertical Track (Uniform Routing Distribution)

Similarly, the probability that the route passes through (x, y) and then $(x, y + 1)$ is:

$$ {}_{x+y}C_x \ \times \ {}_{m+n-x-y-1}C_{m-x} \ \div \ {}_{m+n}C_m $$

2.3 ALGORITHM

It is very computationally intensive to calculate each routing probability individually. There is a faster and easier way to calculate the routing probabilities.

At each grid point the route has to go either right or down. Suppose ${P_{00}}'$ and ${P_{00}}''$ are the conditional probabilities that the routing will go right and down respectively (Figure 6).

$$ \begin{array}{lcccc} {P_{00}}' & = & {}_0C_0 \ \times \ {}_{m+n-1}C_n \ \div \ {}_{m+n}C_m & = & m \div (m+n) \\ {P_{00}}'' & = & {}_0C_0 \ \times \ {}_{m+n-1}C_m \ \div \ {}_{m+n}C_m & = & n \div (m+n) \end{array} $$

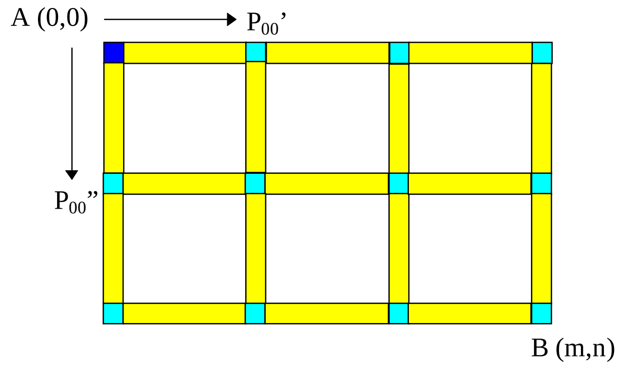


Figure 6: Conditional Routing Probabilities (Uniform Routing Distribution)

$$P_{00}{}' \div P_{00}{}'' \quad = \quad m \div n$$

From Section 2.2, we can derive the above equations. The equations imply that the conditional probabilities depend solely on the ratio of m and n. We can apply this property to find out the routing probabilities for all the horizontal and vertical tracks. The step-by-step algorithm is shown below.

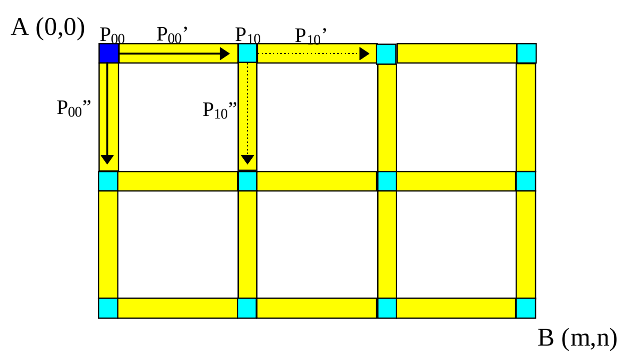


Figure 7: Routing for the First Row (Uniform Routing Distribution)

Step 1: Set the probability at point A (P_{00}) to 1.0.

Step 2: Calculate the probabilities coming out from point A.

$$P_{00}{}' \quad = \quad P_{00} \times m \div (m+n)$$
$$P_{00}{}'' \quad = \quad P_{00} \times n \div (m+n)$$

Step 3: Calculate the probabilities at $(1,0)$.

$$P_{10} \quad = \quad P_{00}{}'$$
$$P_{10}{}' \quad = \quad P_{10} \times (m-1) \div (m+n-1)$$
$$P_{10}{}'' \quad = \quad P_{10} \times n \div (m+n-1)$$

Step 4: Similarly, calculate the probabilities at $(2,0)$, $(3,0)$, ..., $(m,0)$ on the first row.

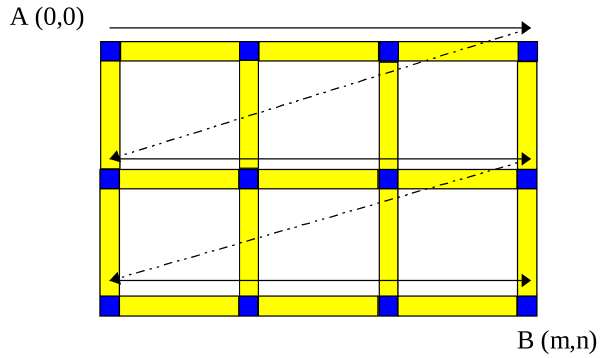


Figure 8: Routing for the Remaining Rows (Uniform Routing Distribution)

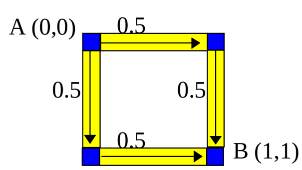
Step 5: Calculate the probabilities at $(0,1)$, $(1,1)$, ..., $(m,1)$ on the second row.

$$
\begin{aligned}
P_{01} &= P_{00}{''} \\
P_{01}{'} &= P_{01} \times m \div (m+n-1) \\
P_{01}{''} &= P_{01} \times (n-1) \div (m+n-1) \\
P_{11} &= P_{01}{'} + P_{10}{''} \\
P_{11}{'} &= P_{11} \times (m-1) \div (m+n-2) \\
P_{11}{''} &= P_{11} \times (n-1) \div (m+n-2) \\
&\cdots
\end{aligned}
$$

Step 6: Repeat step 5 for the 3^{rd}, 4^{th}, ..., n^{th} rows.

2.4 EXAMPLES

Example 2.1: Routing probabilities for $m = 1$ and $n = 1$.



Example 2.2: Routing probabilities for $m = 3$ and $n = 2$.

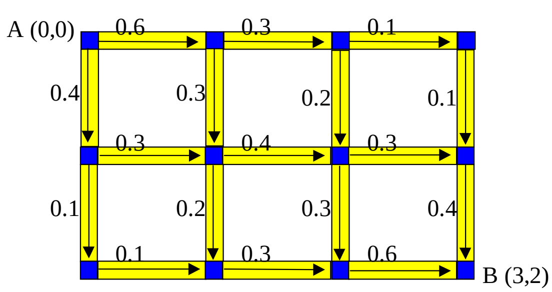


Figure 9: Uniform Routing Distribution Examples

3 GEOMETRIC ROUTING DISTRIBUTION

In uniform routing distribution, we model all possible routes between the two end points to have equal probabilities. So we can write:

Prob[route has 2 bends] $\div$ Prob[route has 1 bend] =
(Number of routes with 2 bends) $\div$ (Number of routes with 1 bend)

Prob[route has 3 bends] $\div$ Prob[route has 2 bend] =
(Number of routes with 3 bends) $\div$ (Number of routes with 2 bends)

However, most modern routers do not follow uniform routing distribution. The routers typically prefer wires with less vias (or bends) because vias consume a lot of space and their electrical impedance can cause significant wiring delay. The geometric routing distribution takes the router preference in consideration, by using a geometric scaling factor α . α is between zero and one.

Prob[route has 2 bends] $\div$ Prob[route has 1 bend] =
$\alpha \times$ (Number of routes with 2 bends) $\div$ (Number of routes with 1 bend)

Prob[route has 3 bends] $\div$ Prob[route has 2 bend] =
$\alpha \times$ (Number of routes with 3 bends) $\div$ (Number of routes with 2 bends)

....

Prob[route has N+1 bends] $\div$ Prob[route has N bend] =
$\alpha \times$ (Number of routes with N+1 bends) $\div$ (Number of routes with N bends)

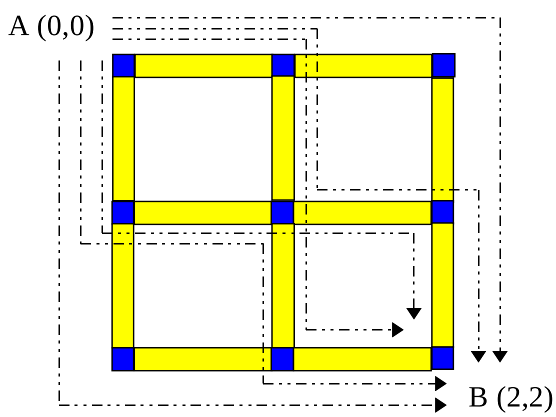


Figure 10: Routing from (0,0) to (2,2) (Geometric Routing Distribution)

For example, suppose we want to route from (0,0) to (2,2). There are two routes with 1 bend, two routes with 2 bends, and two routes with 3 bends (Figure 10).

$$\begin{array}{lcl}
\text{Prob[route has 1 bend]} & = 2 \div (2 + 2\,\alpha + 2\,\alpha^2) & = 1 \div (1 + \alpha + \alpha^2) \\
\text{Prob[route has 2 bends]} & = 2\,\alpha \div (2 + 2\,\alpha + 2\,\alpha^2) & = \alpha \div (1 + \alpha + \alpha^2) \\
\text{Prob[route has 3 bends]} & = 2\,\alpha^2 \div (2 + 2\,\alpha + 2\,\alpha^2) & = \alpha^2 \div (1 + \alpha + \alpha^2)
\end{array}$$

3.1 DERIVATION

If either m or n is equal to zero, there is only one way to route from A to B. But if both m and n are non-zero, the routing will have at least one bend. The maximum number of bends is equal to $(2 \times m - 1)$ if m and n are the same; otherwise the maximum number is $2 \times min(m, n)$. For instance, the maximum number of bends is 4 for $m = 3$, $n = 2$.

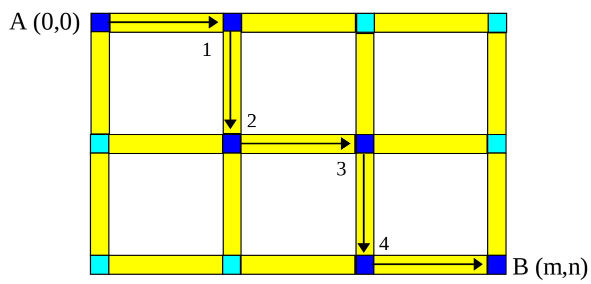


Figure 11: Routing from (0,0) to (3,2) (Geometric Routing Distribution)

As described in Section 2.1, there are ${}_{m+n}C_m$ different ways to route from A to B if the routing is limited to Manhattan distance. Out of the ${}_{m+n}C_m$ different routes, there are

$2 \times {}_{m-1}C_0 \times {}_{n-1}C_0$	routes have 1 bend
${}_{m-1}C_1 \times {}_{n-1}C_0 + {}_{m-1}C_0 \times {}_{n-1}C_1$	routes have 2 bends
$2 \times {}_{m-1}C_1 \times {}_{n-1}C_1$	routes have 3 bends
${}_{m-1}C_2 \times {}_{n-1}C_1 + {}_{m-1}C_1 \times {}_{n-1}C_2$	routes have 4 bends
$2 \times {}_{m-1}C_2 \times {}_{n-1}C_2$	routes have 5 bends
...	
$2 \times {}_{m-1}C_{k-1} \times {}_{n-1}C_{k-1}$	routes have 2k-1 bends
${}_{m-1}C_k \times {}_{n-1}C_{k-1} + {}_{m-1}C_{k-1} \times {}_{n-1}C_k$	routes have 2k bends (if $m > k, n > k$)
${}_{m-1}C_k \times {}_{n-1}C_{k-1}$	routes have 2k bends (if $n = k$)
${}_{n-1}C_k \times {}_{m-1}C_{k-1}$	routes have 2k bends (if $m = k$)

Proof:

- *Number of routes which have 2k-1 bends*

  Assume the first segment is horizontal and the last segment is vertical. Excluding the first and the last segments, there are $k - 1$ horizontal segments and $k - 1$ vertical segments. The positions of the $k-1$ vertical segments are chosen from $m-1$ positions, whereas the positions of the $k - 1$ horizontal segments are chosen from $n - 1$ positions.

If the first segment is horizontal, the number of routes is ${}_{m-1}C_{k-1} \times {}_{n-1}C_{k-1}$. Similarly, if the first segment is vertical, the number of routes is also ${}_{m-1}C_{k-1} \times {}_{n-1}C_{k-1}$.

Hence, the total number of routes is

$$2 \times {}_{m-1}C_{k-1} \times {}_{n-1}C_{k-1}.$$

- *Number of routes which have 2k bends*

  Assume the first and the last segments are horizontal. Excluding the first and the last segments, there are $k-1$ horizontal segments and k vertical segments. The positions of the k vertical segments are chosen from $m-1$ positions, whereas the positions of the $k-1$ horizontal segments are chosen from $n-1$ positions.

  If the first segment is horizontal, the number of routes is ${}_{m-1}C_{k} \times {}_{n-1}C_{k-1}$. Similarly, if the first segment is vertical, the number of routes is ${}_{m-1}C_{k-1} \times {}_{n-1}C_{k}$.

  Hence, the total number of routes is

  $${}_{m-1}C_{k} \times {}_{n-1}C_{k-1} + {}_{m-1}C_{k-1} \times {}_{n-1}C_{k}.$$

  Please note that when $m = k$, the first and the last segments cannot be horizontal and the first term does not exist. Similarly, when $n = k$, the first and the last segments cannot be vertical and the second term does not exist.

For instance, for m = 3, n = 2, there are ${}_5C_3 = 10$ different routes.

$2 \times {}_2C_0 \times {}_1C_0$	=	2 routes have 1 bend
${}_2C_1 \times {}_1C_0 + {}_1C_0 \times {}_1C_1$	=	3 routes have 2 bends
$2 \times {}_2C_1 \times {}_1C_1$	=	4 routes have 3 bends
${}_2C_2 \times {}_1C_1$	=	1 routes have 4 bends
Total	=	10 routes

3.2 ALGORITHM

In this section, we will derive an algorithm to calculate the probability that the routing passes through each horizontal and vertical track.

- Horizontal Track

  As described in Section 2, there are ${}_{x+y}C_x \times {}_{m+n-x-y-1}C_{n-y}$ possible routes from A to B that pass through (x, y) and then $(x+1, y)$. First, we need to find out how many of those routes have one bend, two bends, etc.

  Consider the routing from point A to point P at (x, y), we can apply the same arguments in Section 3.1. The only difference is that there is an additional bend if the last segment is vertical.

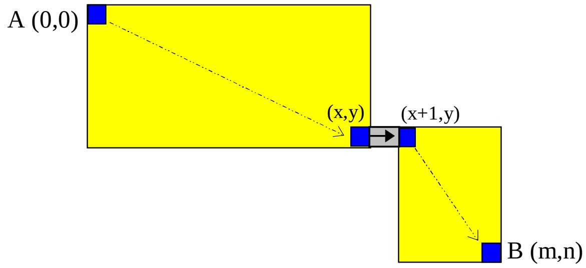


Figure 12: Routing through a Horizontal Track (Geometric Routing Distribution)

$$\begin{array}{ll}
{}_{x-1}C_0 \times {}_{y-1}C_0 & \text{routes have 1 bend} \\
{}_{x-1}C_1 \times {}_{y-1}C_0 + {}_{x-1}C_0 \times {}_{y-1}C_0 & \text{routes have 2 bends} \\
{}_{x-1}C_1 \times {}_{y-1}C_1 + {}_{x-1}C_0 \times {}_{y-1}C_1 & \text{routes have 3 bends} \\
{}_{x-1}C_2 \times {}_{y-1}C_1 + {}_{x-1}C_1 \times {}_{y-1}C_1 & \text{routes have 4 bends} \\
{}_{x-1}C_2 \times {}_{y-1}C_2 + {}_{x-1}C_1 \times {}_{y-1}C_2 & \text{routes have 5 bends} \\
\ldots & \\
{}_{x-1}C_{k-1} \times {}_{y-1}C_{k-1} + {}_{x-1}C_{k-2} \times {}_{y-1}C_{k-1} & \text{routes have 2k-1 bends} \\
{}_{x-1}C_k \times {}_{y-1}C_{k-1} + {}_{x-1}C_{k-1} \times {}_{y-1}C_{k-1} & \text{routes have 2k bends}
\end{array}$$

From point $(x+1, y)$ to point B, we can derive the following equations.

$$\begin{array}{ll}
{}_{m-x-2}C_0 \times {}_{n-y-1}C_0 & \text{routes have 1 bend} \\
{}_{m-x-2}C_1 \times {}_{n-y-1}C_0 + {}_{m-x-2}C_0 \times {}_{n-y-1}C_0 & \text{routes have 2 bends} \\
{}_{m-x-2}C_1 \times {}_{n-y-1}C_1 + {}_{m-x-2}C_0 \times {}_{n-y-1}C_1 & \text{routes have 3 bends} \\
{}_{m-x-2}C_2 \times {}_{n-y-1}C_1 + {}_{m-x-2}C_1 \times {}_{n-y-1}C_1 & \text{routes have 4 bends} \\
{}_{m-x-2}C_2 \times {}_{n-y-1}C_2 + {}_{m-x-2}C_1 \times {}_{n-y-1}C_1 & \text{routes have 5 bends} \\
\ldots & \\
{}_{m-x-2}C_{k-1} \times {}_{y-1}C_{k-1} + {}_{m-x-2}C_{k-2} \times {}_{n-y-1}C_{k-1} & \text{routes have 2k-1 bends} \\
{}_{m-x-2}C_k \times {}_{y-1}C_{k-1} + {}_{m-x-2}C_{k-1} \times {}_{n-y-1}C_{k-1} & \text{routes have 2k bends}
\end{array}$$

The total number of bends is equal to the number of bends from A to (x, y) plus the number of bends from $(x+1, y)$ to B.

Hence, the probability that the routing passes through (x, y) and $(x+1, y)$ is:

$$\frac{[{}_{x-1}C_0 \cdot {}_{y-1}C_0 + ({}_{x-1}C_1 \cdot {}_{y-1}C_0 + {}_{x-1}C_0 \cdot {}_{y-1}C_0) \cdot \alpha + \ldots][{}_{m-x-2}C_0 \cdot {}_{n-y-1}C_0 \cdot \alpha + ({}_{m-x-2}C_1 \cdot {}_{n-y-1}C_0 + {}_{m-x-2}C_0 \cdot {}_{n-y-1}C_0) \cdot \alpha^2 + \ldots]}{2 \cdot {}_{m-1}C_0 \cdot {}_{n-1}C_0 + ({}_{m-1}C_1 \cdot {}_{n-1}C_0 + {}_{n-1}C_1 \cdot {}_{m-1}C_0) \cdot \alpha + 2 \cdot {}_{m-1}C_1 \cdot {}_{n-1}C_1 \cdot \alpha^2 + \ldots}$$

- Vertical Track

  Similarly, the probability that the routing passes through (x, y) and $(x, y+1)$ is:

$$\frac{[{}_{x-1}C_0 \cdot {}_{y-1}C_0 + ({}_{x-1}C_1 \cdot {}_{y-1}C_0 + {}_{x-1}C_0 \cdot {}_{y-1}C_0) \cdot \alpha + \ldots][{}_{m-x-1}C_0 \cdot {}_{n-y-2}C_0 \cdot \alpha + ({}_{m-x-1}C_0 \cdot {}_{n-y-2}C_1 + {}_{m-x-1}C_0 \cdot {}_{n-y-2}C_0) \cdot \alpha^2 + \ldots]}{2 \cdot {}_{m-1}C_0 \cdot {}_{n-1}C_0 + ({}_{m-1}C_1 \cdot {}_{n-1}C_0 + {}_{n-1}C_1 \cdot {}_{m-1}C_0) \cdot \alpha + 2 \cdot {}_{m-1}C_1 \cdot {}_{n-1}C_1 \cdot \alpha^2 + \ldots}$$

3.3 EXAMPLES

For $m = 3$ and $n = 2$, the routing probabilities on all the tracks are shown for various α in Figure 14.

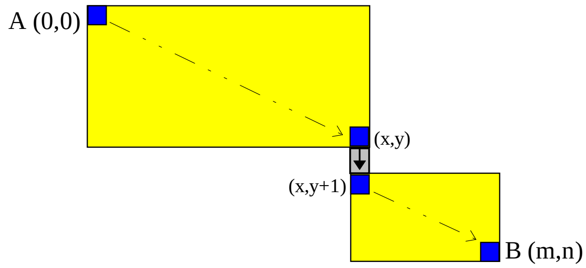


Figure 13: Routing through a Vertical Track (Geometric Routing Distribution)

4 PROBABILISTIC ROUTING ALGORITHM

The geometric routing distribution models real routers pretty well. However, the formulas to calculate the routing probabilities are complex and the computational time is long. There is a much simpler and quicker way.

At each point, we can assume that the routing probability is divided into two components. The first component follows the uniform routing distribution described in Section 2. The second component always goes straight. We define β as the fraction of the routing probability belonged to the first component. Obviously, $(1-\beta)$ is the fraction of the routing probability belonged to the second component.

The algorithm are described in Section 4.1 and Section 4.2, and the results are then presented and analyzed in Section 4.3 and Section 4.4.

4.1 DERIVATION

Consider a point P at (x, y) between point A and point B. There are two incoming routing probabilities ($p1$ and $p2$) and two outgoing routing probabilities ($p3$ and $p4$). Probability that the route passes through point $P = p1 + p2 = p3 + p4$.

Now, we divide $p1$ into two components ($p1_A$ and $p1_B$), where

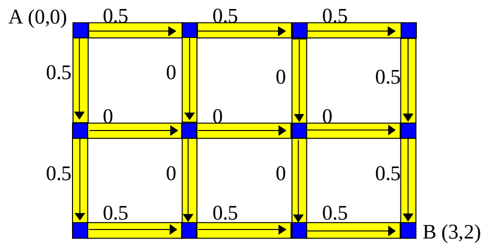
$$\begin{aligned} p1_A &= p1 \times \beta \\ p1_B &= p1 \times (1-\beta) \end{aligned}$$

The first component ($p1_A$) follows the uniform routing distribution as described in Section 2. Similarly, $p2$ is divided into $p2_A$ and $p2_B$. The second component ($p1_B$) always avoid bends and goes straight unless the routing hits the right or the bottom edge.

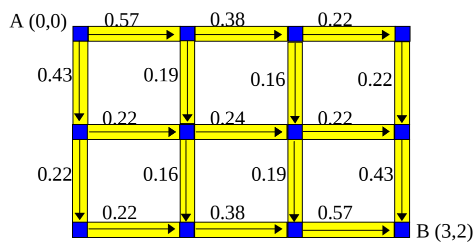
So, we get the following equations.

$$p1_A = p1 \times \beta$$

Example 3.1: Routing probabilities for $\alpha = 0$.



Example 3.2: Routing probabilities for $\alpha = 0.5$.



Example 3.3: Routing probabilities for $\alpha = 1.0$.

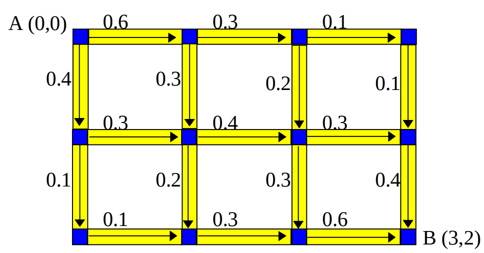


Figure 14: Geometric Routing Distribution Examples

$$
\begin{aligned}
p1_B &= p1 \times (1-\beta) \\
p1 &= p1_A + p1_B \\
\\
p2_A &= p2 \times \beta \\
p2_B &= p2 \times (1-\beta) \\
p2 &= p2_A + p2_B \\
\\
p3_A &= (p1_A + p2_A) \times (n-y) \div (m+n-x-y) \\
p3_B &= p1_B \\
p3 &= p3_A + p3_B \\
&= (p1 + p2) \times \beta \times (n-y) \div (m+n-x-y) + p1 \times (1-\beta) \\
\\
p4_A &= (p1_A + p2_A) \times (m-x) \div (m+n-x-y) \\
p4_B &= p2_B \\
p4 &= p4_A + p4_B
\end{aligned}
$$

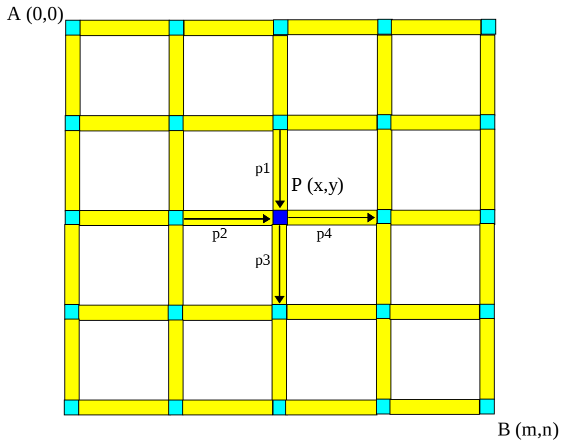


Figure 15: Probabilistic Routing Algorithm Equations

$$= \quad (p1 + p2) \times \beta \times (m - x) \div (m + n - x - y) + p2 \times (1 - \beta)$$

If we start from point A and calculate all the routing probabilities from point A to point B. The routing probabilities may not be symmetrical, i.e., $P_{xy} \neq P_{(m-x)(n-y)}$. Typically, these two probabilities are close but not identical.

One simple solution is to assume we route from point A to point B half of the time and from point B to point A half of the time. After we have calculated all the routing probabilities, we need to average out (P_{xy} and $P_{(m-x)(n-y)}$), ($P_{xy}{}'$ and $P_{(m-x-1)(n-y)}{}'$), ($P_{xy}{}''$ and $P_{(m-x)(n-y-1)}{}''$) as shown in Figure 16.

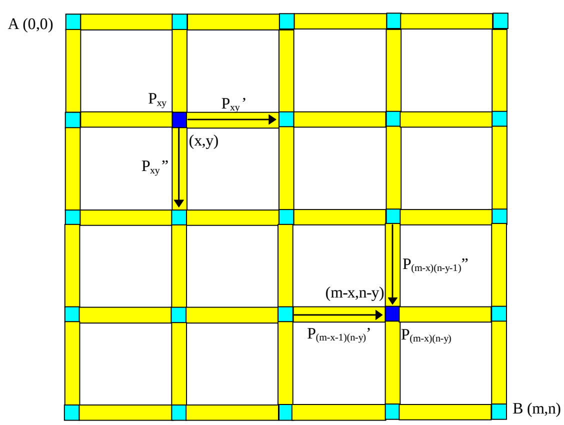


Figure 16: Symmetry of Routing Probabilities

4.2 ALGORITHM

Step 1: Set the probability at point A (P_{00}) to 1.0.

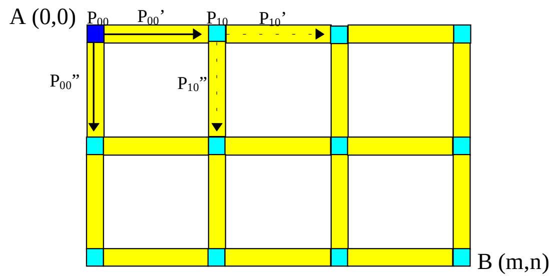


Figure 17: Probabilistic Routing Algorithm (Step 1 to 3)

Step 2: Calculate the probabilities coming out from point A.

$$
\begin{aligned}
P_{00}{}' &= P_{00} \times m \div (m+n) \\
P_{00}{}'' &= P_{00} \times n \div (m+n)
\end{aligned}
$$

Step 3: Calculate the probabilities at $(1,0)$.

$$
\begin{aligned}
P_{10} &= P_{00}{}' \\
P_{10}{}' &= P_{10} \times \beta \times (m-1) \div (m+n-1) + \; P_{00}{}' \times (1-\beta) \\
P_{10}{}'' &= P_{10} \times \beta \times n \div (m+n-1)
\end{aligned}
$$

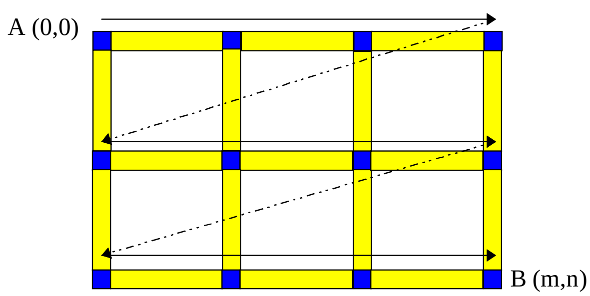


Figure 18: Probabilistic Routing Algorithm (Step 4 to 6)

Step 4: Similarly, calculate the probabilities at $(2,0)$, $(3,0)$, ..., $(m,0)$ on the first row.

Step 5: Calculate the probabilities at $(0,1)$, $(1,1)$, ..., $(m,1)$ on the second row.

$$
\begin{aligned}
P_{01} &= P_{00}{}'' \\
P_{01}{}' &= P_{01} \times \beta \times m \div (m+n-1) \\
P_{01}{}'' &= P_{01} \times \beta \times (n-1) \div (m+n-1) + P_{00}{}'' \times (1-\beta) \\
P_{11} &= P_{01}{}' + P_{10}{}'' \\
P_{11}{}' &= P_{11} \times \beta \times (m-1) \div (m+n-2) + P_{01}{}' \times (1-\beta) \\
P_{11}{}'' &= P_{11} \times \beta \times (n-1) \div (m+n-2) + P_{10}{}'' \times (1-\beta)
\end{aligned}
$$

$\ldots$

Step 6: Repeat step 5 for the 3^{rd}, 4^{th}, ..., n^{th} rows.

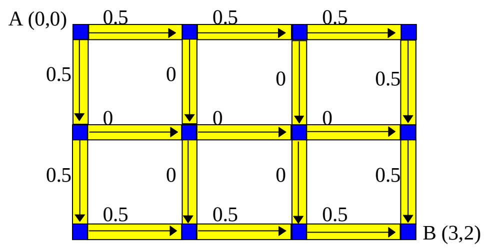
Step 7: To fix the problem of symmetry, recalculate all routing probabilities using:

$$P_{new\ xy} = P_{new\ (m-x)(n-y)} = (\ P_{xy} + P_{(m-x)(n-y)}\ ) \div 2$$

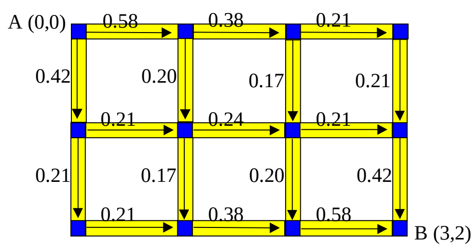
4.3 EXAMPLES

In these examples, $m = 3$ and $n = 2$. The routing probabilities on all the tracks are shown for various β in Figure 19.

Example 4.1: Routing probabilities for β = 0.



Example 4.2: Routing probabilities for β = 0.667.



Example 4.3: Routing probabilities for β = 1.0.

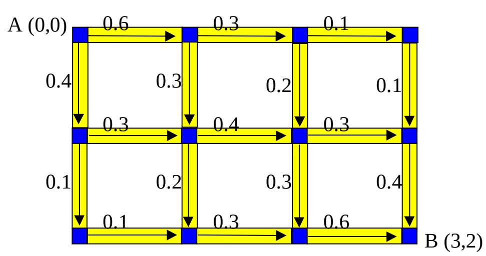


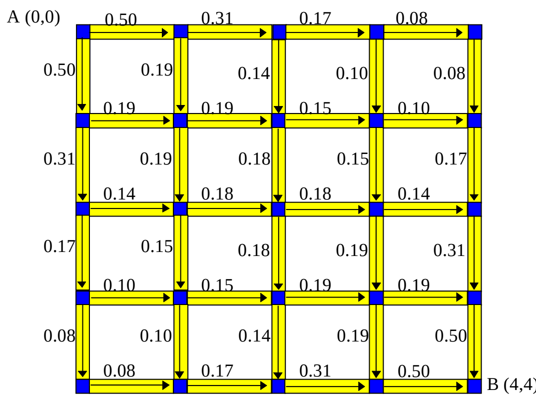
Figure 19: Probabilistic Routing Algorithm Examples

4.4 COMPARISON WITH GEOMETRIC ROUTING DISTRIBUTION

The geometric routing distribution is closely related to the probabilistic routing algorithm. When $\alpha = \beta \div (2 - \beta)$, the routing probabilities derived from the geometric routing distribution and the probabilistic routing algorithm are almost identical. In most cases, the percentage difference between the two derivations is below 2%.

As shown in Figure 20, the percentage difference is 1.4% for $m = n = 4$, $\alpha = 0.5$, $\beta = 0.667$.

Case 1 (Geometric Routing Distribution - $\alpha = 0.5$):



Case 2 (Probabilistic Routing Algorithm - $\beta = 0.667$):

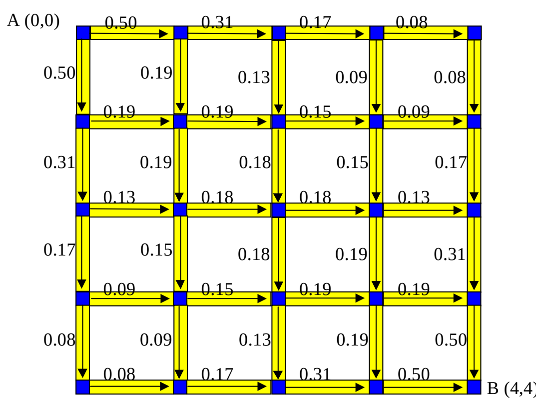


Figure 20: Probabilistic Routing Algorithm Results

The relationship between α and β is:

$$\begin{aligned} \alpha &= \beta \div (2 - \beta) \\ \beta &= 2 \times \alpha \div (1 + \alpha) \end{aligned}$$

Proof:

Consider only probability $p1$ and assume probability $p2$ is zero for now. The same argument can also apply to $p2$.

$$\begin{aligned} p3 &= \text{Prob[route does not have bend at P]} \\ &= p1 \times (1 - \beta) + p1 \times \beta \times (n - y) \div (m + n - x - y) \end{aligned}$$

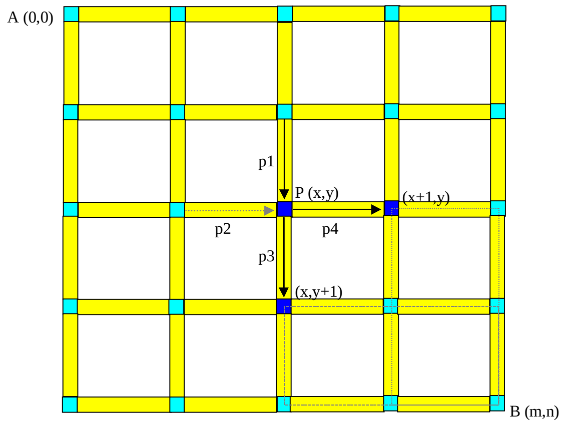


Figure 21: Comparison with Geometric Routing Distribution

$$\begin{aligned} p4 &= \text{Prob[route has bend at P]} \\ &= p1 \times \beta \times (m-x) \div (m+n-x-y) \end{aligned}$$

First, examine the case when $(m-x) = (n-y)$. The vertical distance between P and B is equal to the horizontal distance. The possible routes from $(x+1, y)$ to B are similar to those from $(x, y+1)$ to B because the two rectangles as shown in the diagram are identical. The only difference is that routes passing through $(x+1, y)$ have an extra bend.

$$\begin{aligned} p4 &= p1 \times \beta \times (m-x) \div (m+n-x-y) \\ &= p1 \times \beta \div 2 \\ &= p1 \times \alpha \div (1+\alpha) \end{aligned}$$

Hence, we get

$$\begin{aligned} \alpha &= \beta \div (2-\beta) \\ \beta &= 2 \times \alpha \div (1+\alpha) \end{aligned}$$

When $(m-x) \neq (n-y)$, we need to take the effect of the geometry of the rectangles into account.

$$\begin{aligned} p4 &= p1 \times \beta \times (m-x) \div (m+n-x-y) \\ &\approx p1 \times \alpha \div (1+\alpha) \times 2 \times (m-x) \div (m+n-x-y) \end{aligned}$$

This is only an approximation because the probabilistic routing algorithm only consider the routing from P to B, whereas the geometric routing distribution consider the whole routing from point A to point P and then to B. Thus,

$$\begin{aligned} \alpha &\approx \beta \div (2-\beta) \\ \beta &\approx 2 \times \alpha \div (1+\alpha) \end{aligned}$$

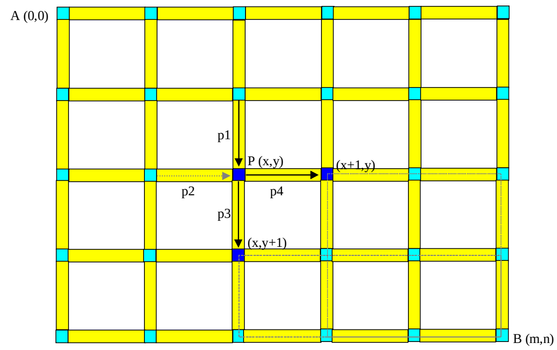


Figure 22: Probabilistic Routing Algorithm Scaling Factor

5 OTHER CONSIDERATIONS

5.1 ROUTING BETWEEN TWO BLOCKS

In the pre-RTL stage, we know the dimensions of the functional blocks and the number of connections among the functional blocks. But we do not know the exact port locations. Assume the probability density function of the port location is uniformly distributed on the functional block. We can divide each functional block into small rectangular tiles, and distribute the number of connections among the tiles. Note that the functional block can be of any shape and does not need to be rectangular.

For example, we want to connect 360 wires between block A and block B. Now, block A is divided into 6 tiles and block B is divided into 6 tiles. Each tile in block A has to connect to each tile in block B with $360 \div (6 \times 6)$ or 10 wires.

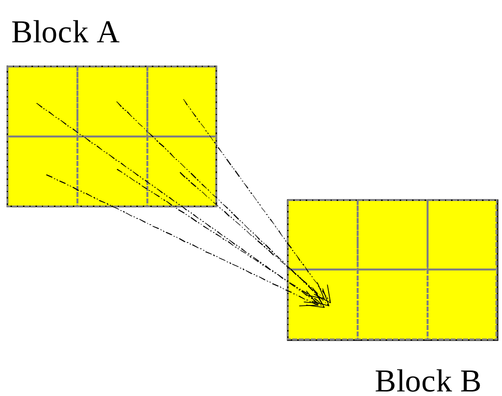


Figure 23: Routing from Block A to Block B

We can apply the same algorithm even if the port location is not uniformly distributed on the functional block. For instance, if we know that the port is located near the left edge of the functional block, we just need to distribute the wires among the tiles in that region and safely ignore all the other tiles.

5.2 ROUTING OBSTACLES

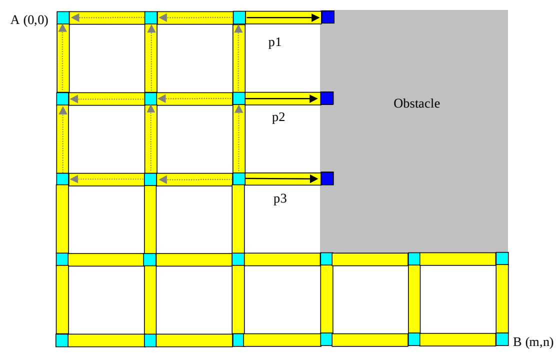


Figure 24: Routing Obstacle Overview

Assume there is a routing obstacle between point A and point B.

Step 1: Use the original algorithm but ignore all the routing probabilities going into the obstacle.

Step 2: After we have finished all routing probabilities from A to B, we have to remove all the routing probabilities going into the obstacle, i.e., $p1$, $p2$ and $p3$. These probabilities are routed backward to point A. These backward routing probabilities are deducted from the original forward routing probabilities.

Step 3: Finally, all routing probabilities have to be multiplied by $1 \div (1 - p1 - p2 - p3)$ to compensate for the "missing" probabilities.

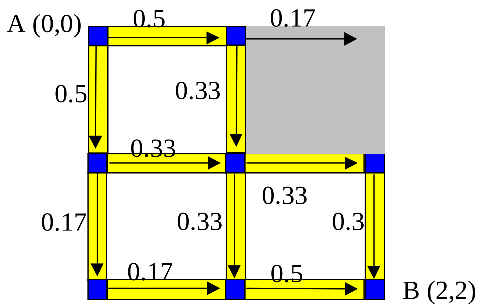
A routing obstacle example is shown in Figure 25.

5.3 COMMON CONNECTIONS AMONG MULTIPLE BLOCKS

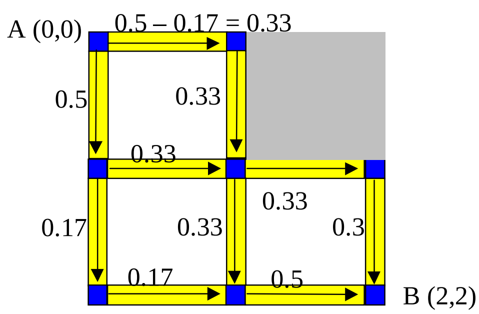
Suppose we need to connect M wires among block A, B and C. We can assume that we need to connect $M \times f$ wires from A to B, $M \times f$ wires from B to C and $M \times f$ wires from C to A, where f is a scaling factor depending on the number of blocks and the block locations.

One simple way is to ignore the block locations, we can assume $f = 2 \div 3$. The reason is that we make three block connections when only two are needed. In general, if we want to connect M wires among N blocks, we will need to make ${}_NC_2$ block connections when only $(N - 1)$ are needed.

Step1: Derive the routing probabilities as before.



Step 2: Route the obstacle probabilities backward.



Step 3: Compensate for the "lost" probabilities.

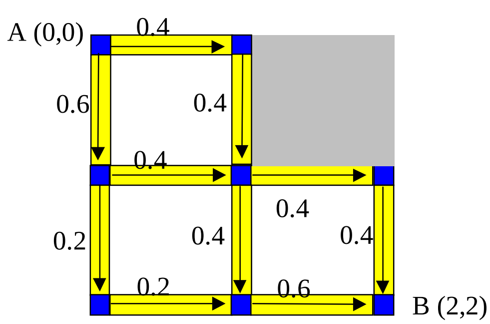


Figure 25: Routing Obstacle Algorithm

$$
\begin{aligned}
f &= (N-1) \div {}_NC_2 \\
  &= 2 \div N
\end{aligned}
$$

In reality, f is even smaller because the rectilinear length of the Steiner tree is shorter than the $(N-1)$ connections between blocks.

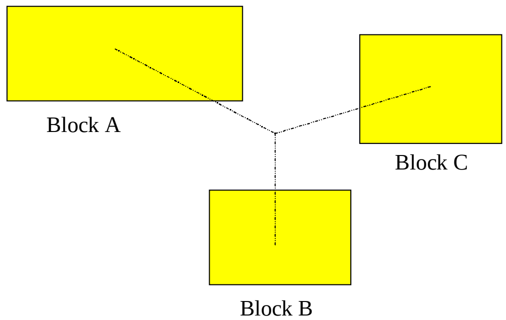


Figure 26: Connections among Multiple Blocks

5.4 CONGESTION MODEL

5.4.1 CARTESIAN COORDINATES

As shown in Figure 27, each grid point represents a rectangular tile on the die which contains a number of horizontal and vertical routing channels. The dimensions of the tile are directly proportional to the number of routing channels in each tile. It is important to choose the dimensions carefully. If the dimension is too big, we cannot locate the local routing congestion precisely. On the other hand, if the dimension is too small, the congestion model may produce a lot of false congestion alarms. From our empirical data, the tile should contain at least 100 routing channels to give a meaningful routing estimate.

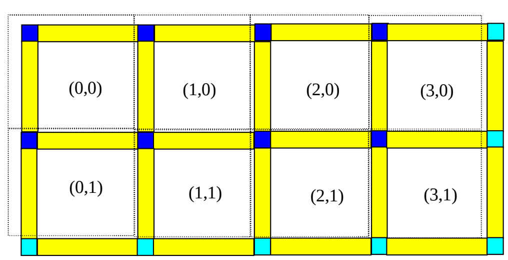


Figure 27: Stochastic Congestion Model

5.4.2 CHANNEL SUPPLY

After the tiles are mapped to the Cartesian coordinate system, we can compress the multiple metal layers into one single logical metal layer. Now, each metal layer has its own metal pitch, metal width and porosity requirements. Within each rectangular tile, we can determine the total number of horizontal and vertical channels available for routing.

Example:

M1 metal pitch	=	1 micron
M2 metal pitch	=	1 micron
M3 metal pitch	=	2 microns
M4 metal pitch	=	4 microns

Suppose each rectangular tile is 100 microns by 100 microns. Metal tracks in M1 and M3 runs horizontally, while metal tracks in M2 and M4 runs vertically. Each tile contains 150 ($= 100 \div 1 + 100 \div 2$) horizontal tracks and 125 ($= 100 \div 1 + 100 \div 4$) vertical tracks.

5.4.3 ROUTING CONGESTION

Based on the routing density model developed in Section 2 to Section 5 we can determine if there is any routing congestion in any location by simply subtracting the horizontal and vertical channel supplies from their corresponding routing densities. If the result is negative, there is routing congestion in that particular location. The layout may still be routable if the congestion is localized. However, the routing may have to extend beyond the Manhattan distance.

The idea is that the chip is routable if and only if the channel supply exceeds the channel demand in every single tile of the chip.

6 CONCLUSION AND FUTURE WORK

In this report, we proposed a new stochastic congestion model for VLSI interconnect routing. A simple and efficient algorithm was presented to calcuate the routing densities, which can be used to predict routability and estimate the total die size.

Our congestion model is actually very general and can apply to any VLSI designs. But it is particularly useful for pre-RTL floorplanning because this model does not require the actual port locations.

Based on this model, we are building a software prototype to verify the theory and look for further refinements.

References

[Ass94] Semiconductor Industry Association. The national technology roadmap for semiconductors. San Jose, CA, 1994.

[Bak87] H. B. Bakoglu. *Circuit and System Performance Limits on ULSI: Interconnections and Packaging.* PhD thesis, Stanford University, 1987.

[BRV93] Stephen D. Brown, Jonathan Rose, and Zvonko G. Vranesic. A stochastic model to predict the routability of field-programmable gate arrays. *IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems*, 12(12):1827–1838, December 1993.

[DDM96] Jeffrey A. Davis, Vivek De, and James Meindl. Optimal low power interconnect networks. *Symposium on VLSI Technology Digest of Technical Papers*, pages 78–79, 1996.

[Don79] Wilm E. Donath. Placement and average interconnection lengths of computer logic. *IEEE Transaction on Circuits and Systems*, CAS-26(4):272–277, April 1979.

[Fel70] William Feller. *An Introduction to Probability Theory and Its Applications*, volume I. John Wiley and Sons, 1970.

[Fer85] D. K. Ferry. Interconnection lengths and vlsi. *IEEE Transaction on Circuits and Devices*, pages 39–42, 1985.

[Fle94] T. Fletcher. Microprocessor technology trends. In *International Electronic Devices Meeting*, pages 269–271, 1994.

[Fly95] Michael J. Flynn. *Computer Architecture Pipelined and Parallel Processor Design.* Jones and Bartlett Publishers, 1995.

[Gam81] Abbas A. El Gamal. Two-dimensional stochastic model for interconnections in master slice integrated circuits. *IEEE Transaction on Circuits and Systems*, CAS-28(2):127–138, February 1981.

[Goe94] Ashok K. Goel. *High-Speed VLSI Interconnections: Modeling, Analysis, and Simulation.* John Wiley and Sons, 1994.

[HHM84] W. R. Heller, C. George Hsi, and Wadie F. Mikhaill. Wirability - designing wiring space for chips and chip packages. *IEEE Design and Test*, pages 43–51, August 1984.

[LR71] Bernard S. Landman and Roy L. Russo. On a pin versus block relationship for partitions of logic graphs. *IEEE Transaction on Computers*, C-20(12):1469–1479, December 1971.

[Mic97] *Microprocessor Report*, Various issues, 1994-97.

[PL88] B. Preas and M. Lorenzetti. *Physical Design Automation of VLSI Systems.* The Benjamin/Cummings Publishing Company, 1988.

[SBP95] Naveed Sherwani, Siddharth Bhingarde, and Anand Panyam. *Routing in the Third Dimension From VLSI Chips to MCMs.* IEEE Press, 1995.

[Sch82] D. C. Schmidt. Circuit pack parameter estimation using rent's rule. *IEEE Transaction on Computer Aided Design of Integrated Circuits and Systems*, CAD-1:186–192, October 1982.

[SK94] M. Sriram and S.M. Kang. *Physical Design for Multichip Modules.* Kluwer Academic Publishers, 1994.

[SP86] Sarma Sastry and Alice C. Parker. Stochastic models for wireability analysis of gate arrays. *IEEE Transaction on Computer-Aided Design*, CAD-5(1):52–65, January 1986.