Probabilistic Optimization for FPGA Board Level Routing Problems

Fei He, Xiaoyu Song, Ming Gu, Guowu Yang, William N. N. Hung, and Jiaguang Sun

***Abstract*—Field programmable gate arrays (FPGAs) are an enabling technology in circuit designs. We consider the board-level multi-terminal net assignment in the FPGA-based logic emulation. A novel probabilistic optimization method is devised for solving the net assignment problem. The approach incorporates randomized rounding, genetic algorithm, and solution-improvement strategies. Experimental results demonstrate promising performance.**

***Index Terms*—Board-level routing, Chernoff bound, field programmable gate array (FPGA), randomized rounding.**

I. Introduction

VERIFICATION is an indispensable step in hardware design. An important verification method is hardware emulation. A widely used hardware emulator is realized by field programmable gate array (FPGA) [1]. In the current VLSI technology, FPGA-based hardware emulation plays a crucial role in system validation. Since FPGA based emulation is generally much faster than software simulation, designers can run a lot more tests on emulators than simulators.

An FPGA-based emulator consists of many FPGAs. The FPGAs are typically connected in two ways: *directly* or *indirectly*. In direct interconnection architectures, each FPGA has to serve two functions: logic and interconnection. In indirect architectures, FPGAs are connected through field-programmable interconnection chips (FPICs) [2], [3]. The FPIC serves only interconnection function but not logic function. The indirect approaches provide uniformed net delay between FPGAs, which is strongly needed in today's performance-driven design process, where interconnect delays raise critical issues. Uniform delays between FPGAs also make it easier to partition and synthesize large designs into several FPGAs.

In this paper, we consider the set of FPGAs for implementing the logic function interconnected by a set of FPICs, where each FPIC is used as a crossbar and can only connect to the FPGAs, but not to each other. Given a set of FPGA chips, a set of crossbars and a set of inter-chip nets, we determine the possible assignment of nets connecting the designated pins through these crossbars. The problem is called the *board-level net routing problem* (*BLRP*) [4]–[7].

An optimal algorithm for two-terminal case was proposed in [4]. The algorithm was devised based on the iterative computation of Euler circuits in graphs of BLRPs. They also proved that the multi-terminal routing problem is NP-hard. In [5], an interesting satisfiability-based method for solving the multi-terminal case was presented. The approach transformed the FPGA board-level routing task into a single Boolean equation. Any assignment of Boolean variables that satisfies the equation specifies a valid routing. Empirical results show that it is one of the best in existing solvers for BLRP.

We devise a novel probabilistic optimization algorithm that incorporates randomized rounding, genetic algorithm and solution improvement strategies. With this approach, we solve the problem of board-level multi-terminal net assignment in FPGA-based logic emulation. The techniques based on Chernoff bounds are used to establish the feasibility of randomized rounding. Experimental results demonstrate the promising performance of the approach. Compared to [5], our approach is more applicable to large routing instances.

The paper is organized as follows. In Section II, we formulate the problem as an integer programming. Section III introduces the general ideas in terms of randomized rounding and application in solving our problem. A mathematical technique, Chernoff bound, is used to analyze the feasibility of randomized rounding. In Section IV, a novel probabilistic optimization algorithm combining randomized rounding, genetic algorithm and solution improvement method is devised. The experimental results of our approach are presented in Section V. Section VI concludes this paper.

II. Problem Formulation

We refer to FPGAs as *chips* and assume all chips are identical. Let CHIPS be the set of all P chips, NETS be the set of all N multi-terminal nets, and CROSSBARS be the set of all K crossbars. A chip has a set of I/O pins. Divide I/O pins of each chip evenly into K groups of same size M (assuming the number of all I/O pins in a chip can be divided exactly by K): $S_1, S_2, \ldots, S_K$. We consider every group as a subset type. Let $S = \{S_1, S_2, \ldots, S_K\}$ be the set of all subset types. Notice the kinds of subset types equal to the number of crossbars. Actually the subset types and the crossbars are related one by one. For every subset type S_i, there is a corresponding crossbar i. Only nets coming from subset type S_i can be connected to crossbar i. Let $\mathrm{NETS} = \{n_1, n_2, \ldots n_N\}$ be a netlist. We represent each net $n_i (i = 1, 2, \ldots, N)$ by a P-bit vector, i.e., $n_i = \langle b_1, b_2, \ldots, b_P \rangle$, where b_j $(j = 1, 2, \ldots, P)$ is a 0–1

Manuscript received October 24, 2004; revised June 27, 2005. This work is supported in part by the Chinese National 973 Plan under Grant 2004CB719406 and by the Chinese National 863 Plan under Grant 2003AA414031. This paper was recommended by Associate Editor T. Ueta.

F. He is with the Department of Computer Science and Technology and the School of Software, Tsinghua University, Beijing 100084, China (e-mail: hef02@mails.tsinghua.edu.cn).

X. Song, G. Yang, and W. N. N. Hung are with the Department of Electrical and Computer Engineering, University of Portland, OR 97203 USA.

M. Gu and J. Sun are with the School of Software, Tsinghua University, Beijing 100084, China.

Digital Object Identifier 10.1109/TCSII.2005.859569

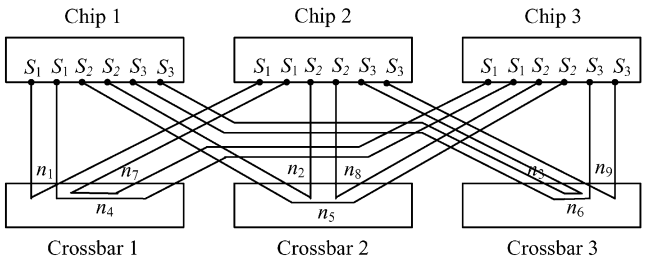

Fig. 1. Instance of BLRP.

variable. Define $b_j = 1$ if n_i has a terminal on chip $j, b_j = 0$ otherwise.

Fig. 1 shows an instance of BLRP with $P = 3, K = 3, M = 2$, and $N = 9$. The nets connected to crossbar 1 belong to subset type S_1, while nets connected to crossbar 2 and crossbar 3 come from subset type S_2 and S_3 respectively. Nets n_1, n_2 and n_3 are represented as $< 1, 1, 0 >$ nets n_4, n_5, and n_6 are represented as $< 1, 0, 1 >$, while nets n_7, n_8, and n_9 are represented as $< 0, 1, 1 >$.

Given a BLRP instance, we determine a function

$$F : \text{NETS} = \{n_1, n_2, \ldots, n_N\} \rightarrow S = \{S_1, S_2, \ldots, S_K\}$$

such that in each chip there are no more than M nets assigned to the same pin subset S_i [4]. If we view each subset type as a bin, each net as an object, then the BLRP can be considered as a partitioned bin-packing problem [7].

We present a new mathematical model for BLRP. Let $A = (n_1^T, n_2^T, \ldots, n_N^T)$ with $n_i = \langle b_1, b_2, \ldots, b_P \rangle$. Namely, $A = (a_{ti})_{P \times N}$ is a $P \times N$ matrix where a_{ti} is the b_t of net n_i.

Define the decision variable as

$$x_{ij} = \begin{cases} 1, & \text{if net } n_i \text{ is assigned to subset type } S_j \\ 0, & \text{otherwise.} \end{cases} \tag{1}$$

and let $X = (x_{ij})_{N \times K}$.

A feasible solution to BLRP satisfies the following two constraints.

1) Every net must be assigned to one and only one subset type, i.e.,

$$\sum_{j=1}^{K} x_{ij} = 1, \qquad \text{for } 1 \le i \le N. \tag{2}$$

2) For each subset on each chip, the number of nets assigned to it cannot exceed M, i.e.,

$$\sum_{i=1}^{N} a_{ti} x_{ij} \le M, \qquad \text{for } 1 \le t \le P, \qquad 1 \le j \le K. \tag{3}$$

The matrix form of (3) is: $AX \le M + 0_{P \times K}$, where $0_{P \times K}$ is a zero matrix with P rows and K columns.

The constraints (1)–(3) characterize the exact solution that we need. We say a solution is *legal* if and only if it satisfies constraints (1)–(3). However, the solution space limited by constraints (1)–(3) is too small. In order to extend the feasible solution space, such that our search process can be performed efficiently, we relax the constraint (2) to be

$$\sum_{j=1}^{K} x_{ij} \le 1, \qquad \text{for} \quad 1 \le i \le N. \tag{4}$$

Our objective function is

$$\text{Maximize} \sum_{i=1}^{N} \sum_{j=1}^{K} x_{ij}. \tag{5}$$

Then, our integer-programming (IP) model is defined by the objective function (5) and the constraints (1), (3), and (4). Let w_0 be the optimal value of (IP), obviously $w_0 \le N$. If $w_0 = N$, the optimal solution satisfies constraints (1)(3), and the solution is legal. If the corresponding integer-programming model has a legal solution, we say the BLRP problem is feasible.

III. Randomized Rounding for BLRP

To solve the integer-programming problems, Raghavan [8] proposed an effective technique of solving a corresponding linear programming problem first and then randomly rounding it. They applied their approach to solving the global wiring problem with promising performance [8]. The technique was also applied to the MAX-SAT problem [9]. Using Raghavan's technique, we do not need to solve an integer programming, which is shown to be NP-hard.

The Chernoff bound is an important technique in designing and analyzing randomized algorithms. The basic ideas behind the Chernoff bound are presented in [10]. Raghavan [8] uses the Chernoff Bound to analyze the randomized rounding. Motwani [11] provides a detailed survey on the Chernoff Bound.

A. Linear Program Relaxation

For the integer programming (IP) of the BLRP, we replace the integrality constraint in (1) with

$$0 \le x_{ij} \le 1, \qquad \text{for} \quad 1 \le i \le N, \qquad 1 \le j \le K.$$

Then, we obtain a *linear program relaxation* of IP. We can use several efficient methods to solve it (such as the simplex method and so on). Let LP denote the linear programming. Let $\widehat{x_{ij}}$, $1 \le i \le N$, be the optimal solutions of LP and let $\widehat{w}$ be its optimal value. Since LP is a relaxation of IP, there is $\widehat{w} \ge w_0$.

B. Randomized Rounding

The optimal solutions of LP may be fractional values, and therefore may not constitute a feasible solution to our integer program IP. We must therefore round these values to 0's and 1's to obtain a feasible BLRP solution. The idea of randomized rounding is to interpret the fractional solutions provided by the linear program as probabilities for the rounding process [11].

Denote by $\overline{x_{ij}}$ the rounded value of $\widehat{x_{ij}}$. Randomized rounding is as follows: independently for each i and each j, set $\overline{x_{ij}}$ to 1 with the probability $\widehat{x_{ij}}$; otherwise set $\overline{x_{ij}}$ to 0. Thus, for each i and each $j, \Pr[\overline{x_{ij}} = 1] = \widehat{x_{ij}}$ and $\Pr[\overline{x_{ij}} = 0] = 1 - \widehat{x_{ij}}$.

C. Analysis by Chernoff Bound

Intuitively, we can always obtain a legal solution by a sufficient number of randomized rounding. But what is the approximate upper bound to the number of necessary randomized roundings? We answer the question in this section.

Theorem 1 [11]: Let $X_1, X_2, \ldots, X_n$ be *independent Poisson trials* such that, for $1 \le i \le n, \Pr[X_i = 1] = p_i$, where $0 < p_i < 1$. Let $X = \sum_{i=1}^{n} X_i$, $\mu = E[X] = \sum_{i=1}^{n} p_i$. Then, for any δ, if $\delta > 0$

$$\Pr[X > (1+\delta)\mu] < \left(\frac{e^{\delta}}{(1+\delta)^{(1+\delta)}}\right)^{\mu} \tag{6}$$

if $0 < \delta \le 1$

$$\Pr[X < (1-\delta)\mu] < \exp(-\mu\delta^2/2). \tag{7}$$

Definition 1 [11]: Define

$$F^{+}(\mu,\delta) \triangleq [e^{\delta}/(1+\delta)^{(1+\delta)}]^{\mu} \tag{8}$$

$$F^{-}(\mu,\delta) \triangleq \exp(-\mu\delta^2/2). \tag{9}$$

For any positive μ and $\varepsilon, \Delta^{+}(\mu,\varepsilon)$ is the value of δ that satisfies

$$F^{+}(\mu, \Delta^{+}(\mu,\varepsilon)) = \varepsilon. \tag{10}$$

Similarly, $\Delta^{-}(\mu,\varepsilon)$ is that value of δ that satisfies

$$F^{-}(\mu, \Delta^{-}(\mu,\varepsilon)) = \varepsilon. \tag{11}$$

Using the Chernoff bound, we can prove the feasibility of randomized rounding to the BLRP instances. First, since the randomized roundings are performed independently to each other, so $\overline{x_{ij}}, 1 \le i \le n$ and $1 \le j \le K$ are independent Poisson trials. Define $\overline{X} = \sum_{i=1}^{N}\sum_{j=1}^{K} \overline{x_{ij}}$ and $\widehat{X} = \sum_{i=1}^{N}\sum_{j=1}^{K} \widehat{x_{ij}}$, then we have

$$E(\overline{X}) = \sum_{i=1}^{N}\sum_{j=1}^{K} E(\overline{x_{ij}}) = \sum_{i=1}^{N}\sum_{j=1}^{K} \widehat{x_{ij}} = \widehat{X}.$$

Theorem 2: Let ε be a real number such that $0 < \varepsilon < 1$. With probability $1-\varepsilon$, the BLRP solution S produced by randomized rounding satisfies

$$\overline{X_s} \le \widehat{X}(1 + \Delta^{+}(\widehat{X},\varepsilon)).$$

Proof: Omitted due to the page limitation [12]. □

Theorem 3: Let ε be a real number such that $0 < \varepsilon < 1$. With probability $1-\varepsilon$, the BLRP solution S produced by randomized rounding satisfies

$$\overline{X_s} \ge \widehat{X}(1 - \Delta^{-}(\widehat{X},\varepsilon)).$$

Proof: Omitted due to the page limitation [12]. □

Theorem 4: In the set of n BLRP solutions repeatedly generated by randomized rounding, with probability $1-\varepsilon^n$, where ε is a real number such that $0 < \varepsilon < 1$, there exists at least one solution that satisfies

$$\overline{X_{s_i}} \le \widehat{X}(1 + \Delta^{+}(\widehat{X},\varepsilon)).$$

Proof: Omitted due to the page limitation [12]. □

Theorem 5: In the set of n BLRP solutions repeatedly generated by randomized rounding, with probability $1-\varepsilon^n$, where ε is a real number such that $0 < \varepsilon < 1$, there exists at least one solution that satisfies

$$\overline{X_{s_i}} \ge \widehat{X}(1 - \Delta^{-}(\widehat{X},\varepsilon)).$$

Proof: Omitted due to the page limitation [12]. □

Suppose we want to find a solution that satisfies

$$0.99 \cdot \widehat{X} \le \overline{X_{s_i}} \le 1.01 \cdot \widehat{X} \tag{12}$$

with the probability 0.999. We first substitute the value of 0.999 to Theorem 4 and 5, and get $\varepsilon^n = 0.001$ and $\delta = 0.01$.

In order to satisfy the former part of (12), we substitute the values of ε and δ into (8) and (10), then

$$10^{-3/n} = (e^{0.01}/1.01^{1.01})^{\widehat{X}}.$$

From the above equation, we can easily calculate $n = 13\,861$ in the case of $\widehat{X} = 10$ and $n = 1386$ in other case of $\widehat{X} = 100$. On the other hand, for satisfying the latter part of (12), we substitute the values of ε and δ into (9) and (11), and then with similar calculation, we can get $n = 13\,816$ when $\widehat{X} = 10$, and $n = 1382$ when $\widehat{X} = 100$.

In conclusion, to find a satisfying solution, we are likely to perform the randomized rounding about 13 861 times in the case of $\widehat{X} = 10$ and about 1386 times in the case of $\widehat{X} = 100$.

IV. Our Approach

In Raghavan's approach, the legal solution of IP is achieved by applying randomized rounding repeatedly. After applying the randomized rounding sufficient times, we can always get a legal solution. However, this process is very time consuming.

A. Optimization Procedure

In our new approach, we adopt the efficient genetic algorithm (GA). The randomized rounding is integrated into the process of GA. To accelerate the searching process, a *solution improvement method* is designed. The solution improvement method maintains a short tabu list and searches the local optimal solution in a neighbor field. The method can greatly reduce the time needed for finding the optimal solution, thus making our algorithm practical. Our optimization procedure is given as follows.

1. Solve the linear program LP to obtain an optimal solution, assume the corresponding optimal objective value is w_{LP}.
2. **if** the linear program has no feasible solution or $w_{\mathrm{LP}} < N$, **return** false.
3. X_best = NULL;
4. **for** ($iter1 = 1$ to $MaxIter1$)

a) Initialize the $population$ with the chromosomes generated by randomized rounding;

b) **for** ($iter2 = 1$ to $MaxIter2$) **do**

i. Update the chromosomes by crossover, mutation and improvement operations;

ii. Calculate the objective value for each chromosome;

iii. Sort chromosomes in increasing order of their objective value; if the best chromosome is better than X_best, substitute it for X_best.

iv. If the objective value of X_best equals N, **return** true.

v. Select the chromosomes by spinning the roulette wheel;

End for
End for
Return false.

There are many efficient methods for solving LP, such as simplex method. The time needed by step 1 greatly depends on the LP solver, so it is not considered into the total runtime taken by our algorithm. X_best stores a copy of the best solution found so far. It is initialized in step 3, and is updated in each iteration cycle (step iii). In step a), we initialize the population. If no legal solution found after *MaxIter2* iterations, the algorithm would be redirected to step a), and then reinitialize the population.

B. Chromosome Coding

Assume that the nets are labeled from 1 to n, and net i is denoted as n_i. Assume the subset types are labeled from 1 to K, and subset type k is denoted as S_k. We encode an assignment of nets to subset types into a chromosome $V = (v_1, v_2, \ldots, v_n)$, where $v_i = j$ represents that net n_i is assigned to subset type S_j.

To suit the form of chromosome coding, we transform the constraint (1)–(3) into the following optimization model:

$$\text{Minimize} \sum_{t=1}^{P} \sum_{j=1}^{K} f\Big(\sum_{i=1}^{N} a_{ti} x_{ij} - M\Big) \tag{13}$$

subject to constraints (1) and (2), where $f(\cdot)$ is the penalty function for unsatisfying constraint (3). Because the constraints (1) and (2) are satisfied naturally by the structure of the chromosome, we only need to consider this objective function (13) in our search process.

C. Initialization

We generate $\mathrm{pop\_size}$ chromosomes by randomized rounding to initialize the population. Assume the solutions of (LP) is $\widehat{x_{ij}}$, for $1 \le i \le N$, and $1 \le j \le K$. We consider the value of $\widehat{x_{ij}}$ as the probability of assigning net i to subset type j, i.e., $\Pr[v_i = j] = \widehat{x_{ij}}$. Thus, the process of randomly rounding the solution of (LP) to be a chromosome is: set v_i independently to be j with probability $\widehat{x_{ij}}, 1 \le j \le K$.

D. Mutation

We define a parameter Pm as the probability of mutation. The number of chromosomes undergoing the mutation operation may most likely to be $\mathrm{Pm} * \mathrm{pop\_size}$. We adopt the two-point mutation. For traditional two-point mutation, it randomly selects two points r_1 and r_2 in the chromosome, and then replaces the value of every character between site r_1 and r_2 with a random number between 1 to K. In our heuristic two-point mutation, the mutation sites r_1 and r_2 are selected similarly, however the value of each character between site r_1 and r_2 are replaced with a randomized rounding number from $\widehat{\mathrm{x}_{ij}}$ by $\Pr[v_i = j] = \widehat{x_{ij}}, r_1 \le i \le r_2$.

Improvement Method:
1、 Set all nets unlocked;
2、 Find the best move: reassign the unlocked net n_i to subset type S_j, such that for all feasible moves, this move reduces objective value mostly;
3、 Perform the best move found in step 2, and set net n_i as locked;
4、 Repeat step 2 and step 3, until no move can be found.

Fig. 2. Improvement method outline.

TABLE I
PERFORMANCE DISTRIBUTION OF OUR APPROACH

Benchmark	N	GA+R+I		GA+R		GA+I	
		Iter	Runtime	Iter	Runtime	Iter	Runtime
p003_k2_m2_n05	3	1	0	1	0	1	0
p005_k4_m5_n05	40	3	0	MAX	1.562	15	0.015
p010_k6_m2_n02	60	1	0	1	0	2	0
p020_k5_m2_n04	76	152	0.25	MAX	7.984	761	1.969
p020_k5_m2_n05	62	8	0.016	MAX	6.766	330	0.672
p020_k5_m2_n06	59	35	0.047	MAX	6.359	44	0.094
p020_k5_m2_n07	49	234	0.218	MAX	5.406	442	0.703
p020_k5_m2_n08	52	29	0.031	MAX	5.688	83	0.141

E. Improvement

We define a parameter Pi as the probability of improvement. When a new chromosome is generated, it is improved with the probability Pi. The number of chromosomes undergoing the improvement operation is most likely to be $\mathrm{Pi} * \mathrm{pop\_size}$. The improvement method is shown in Fig. 2.

In our method, a move is to reassign a net n_i to a subset S_j. The neighborhood of a given point is defined as the set of solutions which can be reached in one move from the given solution. Once a move (which reassign net n_i to a subset S_j) has been done, the opposite move (which assign back net n_i to pre-assigned subset) will be considered as tabu for a given number of T iterations (in this method, T is limited to be 1).

V. EXPERIMENTAL RESULTS

To test our algorithm, we coded our algorithm in C++ language and ran on a PC with Intel® Celeron® 2.4 GHz CPU and 512M RAM. The parameters are set as: $\mathrm{pop\_size} = 50, \mathrm{MaxIter1} = 10, \mathrm{MaxIter2} = 1000, \mathrm{Pm} = 0.05, \mathrm{Pi} = 0.95$, and $\mathrm{Pc} = 0.5$, where Pc is the probability of crossover.

Our approach is an integration of GA and two techniques: randomized rounding and solution improvement strategy. Table I compares the performance contributions made by the two techniques respectively. *Benchmark* is the name of the tested benchmark. The benchmark's name implies some parameters. For example, the name "p003_k2_m2_n05" indicates that the number of chips is 3, the number of pin types is 2, the number of pins in each type is 2, and the maximum number of pins for generated net is 5. Column N lists the number of nets. $GA + R + I$ denotes our algorithm, $GA + R$ denotes the algorithm integrated by GA and the randomized rounding only, $GA + I$ denotes the algorithm integrated by GA and the improvement strategy only.

TABLE II
PERFORMANCE OF OUR ALGORITHUM VERSUS SAT SOLVERS

Benchmark	N	Pin	zChaff	DLM	RA
p020_k5_m3_n12	48	15	3.532	4.14	0.016
p020_k5_m3_n10	53	15	0.547	0.95	0.171
p020_k5_m3_n09	66	15	0.093	0.76	0.328
p020_k5_m3_n08	68	15	0.109	0.44	0.297
p020_k5_m3_n07	77	15	0.188	0.62	0.109
p020_k5_m3_n06	88	15	3.64	0.54	0.765
p020_k5_m3_n05	99	15	4.391	0.36	0.203
p020_k5_m3_n04	114	15	4.406	0.32	0.062
p020_k3_m3_n08	36	9	0.016	0.1	0
p020_k4_m3_n08	67	12	0.016	0.17	0.031
p020_k5_m4_n08	97	20	34.344	N/A	0.984
p020_k5_m5_n08	107	25	N/A	N/A	0.11
p020_k6_m3_n08	82	18	3.047	2.93	0.328
p020_k7_m3_n08	105	21	396.781	5.26	1.094
p020_k8_m3_n08	111	24	373.891	12.21	61.469
p050_k5_m3_n08	171	15	151.313	2.02	15.421
p100_k5_m2_n08	249	10	7.094	0.85	177.157
p20_k5_m15_n10	200	75	EXPLODE	EXPLODE	0.008
p40_k5_m15_n25	200	75	EXPLODE	EXPLODE	0.015
p60_k5_m15_n35	200	75	EXPLODE	EXPLODE	0.04
p80_k5_m15_n45	200	75	EXPLODE	EXPLODE	0.039
p100_k5_m15_n50	200	75	EXPLODE	EXPLODE	0.063
p15_k8_m16_n8	100	128	EXPLODE	EXPLODE	0.016
p15_k8_m18_n8	100	144	EXPLODE	EXPLODE	0.008
p15_k8_m20_n8	100	160	EXPLODE	EXPLODE	0.004
p15_k10_m20_n12	100	200	EXPLODE	EXPLODE	0.016
p15_k15_m20_n12	100	300	EXPLODE	EXPLODE	0.015

Notice that the algorithm $GA+R$ could not find the optimum solution (the case that *Iter* column is recorded as *MAX*) for most of the tested benchmarks, while the other two (with improvement strategy) can find the legal solution in all cases. This demonstrates that the improvement method is the main contributor to the performance improvement.

Table II compares the performance of our algorithm to the SAT solvers used in [5]. All experiments are conducted on the same PC. Column *Pins* lists the number of pins on each chip. *DLM* is the time needed to solve the problem by *DLM* SAT solver, and z*Chaff* is the time needed to solve the problem by *zChaff* SAT solver. The time needed to solve the problem by our randomized algorithm is listed in the *RA* column.

For the same parameters of $P = 20, K = 5$ and $M = 3$, we increase N (number of nets) gradually. However, the time it takes for the *RA* does not increase correspondingly. The same phenomenon can be observed for the number of pin types (K). The reason is that the *RA* is an iterative algorithm, thus the time it takes depends not only on the problem size, but also its used iterations.

For the majority of cases, our algorithm performs better than the *SAT* solvers. Notice there is one benchmark $(\mathrm{p020\_k5\_m5\_n08})$ whose solution cannot be found by none of the *SAT* solvers, but can be found by ours. For some of the cases, our algorithm takes longer to compute than *DLM* or *zChaff*. This is understandable. Since the time needed by a SAT solver usually depends on the complexity of the *CNF* formula [5] (converted from the corresponding benchmark), and the size of the *CNF* formula is not necessarily proportional to the problem size.

Furthermore, we randomly generated some large benchmarks that mimic real-world problem size [13] (with more than 50 pins on a chip) to test our algorithm. For these benchmarks, the *SAT* solvers cannot be used because the converter (converting the benchmarks to *CNF* formulas) explodes already. However our algorithm can solve all these benchmarks with rapid speed. This demonstrates that our algorithm is more applicable to large benchmarks.

VI. CONCLUSION

In this paper, we studied the board level routing problem. A novel mathematical model has been proposed. A new randomized algorithm combining randomized rounding, genetic algorithm and solution improvement method has been presented. Experimental results showed promising performance of our approach.

REFERENCES

[1] S. D. Brown, R. J. Francis, J. Rose, and Z. G. Vranesic, *Field Programmable Gate Arrays*. Norwell, MA: Kluwer, 1992.
[2] M. Slimane-Kadi, D. Brasen, and G. Saucier, "A fast FPGA prototyping system that uses inexpensive high-performance FPIC," in *Proc. ACM/SIGDA Int. Workshop FPGAs*, Berkely, CA, 1994.
[3] Crossbar Switch Reference Design (2002). [Online]. Available: http://www.xilinx.com/esp/xbarswitch
[4] W.-K. Mak and D. F. Wong, "On optimal board-level routing for FPGA-based logic emulation," *IEEE Trans. Comput.-Aided Des. Integr. Circuits*, vol. 16, no. 3, 1997.
[5] X. Song *et al.*, "Board-level multiterminal net assignment for the partial cross-bar architecture," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 11, no. 3, pp. 511–514, Jun. 2003.
[6] M. Gu, F. He, X. Song, and J. Sun, "Multi-terminal net assignments by scatter search," *Math. Comp. Model.*, vol. 41/8–9, pp. 997–1004, Apr.-May 2005.
[7] A. Ejnioui and N. Ranganathan, "Multi-terminal net routing for partial crossbar-based multi-FPGA systems," in *Proc. ACM Int. Symp. FPGA*, Monterey, CA, 1999.
[8] P. Raghavan and C. C. Thompson, "Randomized rounding," *Combinatorica*, vol. 7, pp. 365–374, 1987.
[9] M. X. Goemans and D. P. Williamson, "New 3/4-approximation algorithms for MAX SAT," *SIAM J. Discr. Math.*, 1993.
[10] H. Chernoff, "A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations," *Ann. Math. Stat.*, vol. 23, pp. 493–509, 1952.
[11] R. Motwani and P. Raghavan, *Randomized Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 1995.
[12] F. He *et al.*, "Probabilistic Optimization for FPGA Board Level Routing Problems," Tsinghua Univ., Beijing, China, Tech. Rep., 2005.
[13] Success with Synplicity and Philips Semiconductors (2001). [Online]. Available: http://www.synplicity.com/literature/success/pdf/ss_philips.pdf