electronics

Article

SLDPSO-TA: Track Assignment Algorithm Based on Social Learning Discrete Particle Swarm Optimization

Huayang Cai [1,2,3], Ruping Zhou [1,2,3], Pengcheng Huang [1,2,3], Yidan Jing [1,2,3] and Genggeng Liu [1,2,3,*]

[1] College of Computer and Data Science, Fuzhou University, Fuzhou 350116, China; 210310001@fzu.edu.cn (H.C.); 231016004@fzu.edu.cn (R.Z.); 231027160@fzu.edu.cn (P.H.); 200327040@fzu.edu.cn (Y.J.)
[2] Engineering Research Center of Big Data Intelligence, Ministry of Education, Fuzhou University, Fuzhou 350116, China
[3] Fujian Key Laboratory of Network Computing and Intelligent Information Processing, Fuzhou University, Fuzhou 350116, China
* Correspondence: liugenggeng@fzu.edu.cn

Abstract: In modern circuit design, the short-circuit problem is one of the key factors affecting routability. With the continuous reduction in feature sizes, the short-circuit problem grows significantly in detailed routing. Track assignment, as a crucial intermediary phase between global routing and detailed routing, plays a vital role in preprocessing the short-circuit problem. However, existing track assignment algorithms face the challenge of easily falling into local optimality. As a typical swarm intelligence technique, particle swarm optimization (PSO) is a powerful tool with excellent optimization ability to solve large-scale problems. To address the above issue, we propose an effective track assignment algorithm based on social learning discrete particle swarm optimization (SLDPSO-TA). First, an effective wire model that considers the local nets is proposed. By considering the pin distribution of local nets, this model extracts and allocates more segments to fully leverage the role of track assignment. Second, an integer encoding strategy is employed to ensure that particles within the encoding space range correspond one-to-one with the assignment scheme, effectively expanding the search space. Third, a social learning mode based on the example pool is introduced to PSO, which is composed of other particles that are superior to the current particle. By learning from various objects in the example pool, the diversity of the population is improved. Fourth, a negotiation-based refining strategy is utilized to further reduce overlap. This strategy intelligently transfers and redistributes wire segments in congested areas to reduce congestion across the entire routing panel. Experimental results on multiple benchmarks demonstrate that the proposed SLDPSO-TA can achieve the best overlap cost optimization among all the existing methods, effectively reducing congestion in critical routing areas.

Keywords: conflict minimization; track assignment; example pool; particle swarm optimization; rip-up and reroute

check for updates

Citation: Cai, H.; Zhou, R.; Huang, P.; Jing, Y.; Liu, G. SLDPSO-TA: Track Assignment Algorithm Based on Social Learning Discrete Particle Swarm Optimization. *Electronics* **2024**, *13*, 4571. https://doi.org/10.3390/electronics13224571

Academic Editor: Gemma Piella

Received: 29 September 2024
Revised: 13 November 2024
Accepted: 18 November 2024
Published: 20 November 2024

1. Introduction

Physical design currently represents a key area of research in the field of Very Large Scale Integration (VLSI) computer-aided design [1]. Among the various phases of physical design, routing plays a crucial role, directly impacting the ultimate performance of the chip [2,3]. The traditional routing flow is typically divided into two phases: global routing and detailed routing. Traditionally, detailed routing is guided by global routing results. However, global routing provides a rough overview from a global perspective and may not accurately capture local congestion, making it challenging to detect detailed routing problems. To better guide the detailed routing, physical designs typically introduce an intermediate phase, known as track assignment, between global routing and detailed routing [4]. This phase needs to consider factors such as congestion to achieve high

routability. However, the decreasing chip size and increasing integration complexity pose challenges for traditional routing algorithms, making them less effective in addressing these tasks and introducing a novel set of challenges for routing.

Track assignment serves as a crucial link between global routing and detailed routing. Most existing track assignment algorithms focus on generating conflict-free routing solutions by maximizing the number of assigned wires. Nevertheless, there are unassigned wires that are not considered, potentially leading to routing conflicts. This type of track assignment problem is referred to as the Conflict-free Track Assignment (CFTA) problem. On the other hand, there are a few works that seek conflict-minimizing routing solutions by assigning all wires, thus solving another type of problem called the Conflict-minimizing Track Assignment (CMTA) problem. Although these works consider more routing resources and provide better guidance for detailed routing, they tend to fall into local optimality.

Swarm intelligence (SI) is an important category of optimization techniques and provides new ideas for solving various complex operation optimization problems [5–9]. This field draws inspiration from the straightforward behaviors and self-organized interactions observed among intelligent individuals in nature, including ant colony foraging, bird flocking, flock effects, bacterial growth, and fish swarming [10–14]. Each SI technique has its unique advantages. In particular, the particle swarm optimization (PSO) technique stands out among SI techniques due to its minimal control parameters, simple implementation, and excellent optimization capability. It has been widely used in various fields, including combinatorial optimization [15–18], image matching and enhancement [19,20], and data mining [21,22]. Furthermore, the PSO algorithm has been successfully applied to some routing problems with positive results [23,24]. On the other hand, there has been some recent work on improving the convergence rate or speed of PSO algorithms [25–27]. For example, Ref. [25] improved the convergence speed as well as the global exploration capabilities of PSO algorithms by targeted position-mutated elitism. Ref. [26] proposed a random sampling strategy of control parameters and applied stochastic correction to each dimension for the population optimum value, thereby improving the convergence rate while maintaining a higher convergence accuracy of the algorithm. Ref. [27] combined the Firefly Algorithm with PSO to further improve the effectiveness of the algorithm. Therefore, the PSO technique stands out as one of the effective methods for overcoming the limitations of traditional routing algorithms.

To effectively address the CMTA problem, we propose an effective track assignment algorithm based on social learning discrete particle swarm optimization, called SLDPSO-TA, which considers the congestion problem in the local net, while taking the conflict and wirelength as the optimization objectives. The major contributions of this paper are listed as follows:

- An efficient integer encoding method is designed to enhance the adaptability of the proposed algorithm, the SLDPSO-TA.
- A novel greedy strategy-based initial assignment method is introduced to generate high-quality initial particles. Additionally, a genetic operator is applied to both expand the population diversity and filter out high-quality initial particles.
- A simple and efficient cost function is formulated for the optimization objectives, encompassing routing conflicts (including conflicts among wires and conflicts between wires and blockages) and wirelength. Moreover, a preprocessing strategy is employed to significantly reduce the number of cost calculations, thereby accelerating the proposed algorithm.
- In the particle updating phase, a social learning mode based on the example pool is implemented to enable particles to learn from better particles, which can maintain the population diversity. Furthermore, multi-point mutation is introduced to enhance the exploration capability of the algorithm.
- To further optimize conflicts between wires, a negotiation-based refining strategy is executed through the rip-up and reroute operation on the solution.

The rest of the paper is organized as follows. Section 2 analyzes related works. Section 3 presents the problem formulation and evaluation metrics. Section 4 introduces details of the SLDPSO-TA. Section 5 shows the performance of the proposed algorithm. Conclusions are given in Section 6.

2. Related Work

2.1. PSO

Many scholars have noticed the excellent optimization, fast convergence, and simple parameter setting of the PSO algorithm, and applied it to solve VLSI problems.

An interactive adaptive PSO algorithm is proposed in [28] to improve the search efficiency and accuracy of the population to minimize the area of floor planning. Experiments show that the search quality and speed of this method are the best. In particular, there are many applications of PSO in routing problems. Ref. [29] uses the PSO algorithm to calculate the optimal number and size of buffers. In [30], reinforcement learning is employed to optimize the search of PSO, and a routing scheme with higher quality and safety is generated based on the improved PSO. In [31], the PSO algorithm with gradient descent is adopted to construct the routing tree with the minimum number of vias and wirelength, thereby reducing the chip manufacturing cost and power consumption. In [32], PSO is combined with a meta-heuristic algorithm based on invasive weed optimization to enhance the global optimization capability of the algorithm, thereby reducing the wirelength cost of global routing. Ref. [33] utilizes a multi-objective particle optimization algorithm to study the tradeoff between the power and delay of high-speed circuits, and generates the non-dominated optimal solutions of dynamic power and delay. Ref. [34] proposes a particle swarm optimization method that combines logistic regression and adaptive granularity learning to optimize decoupling capacitance, thereby reducing the cumulative impedance of the power delivery net.

The above work demonstrates the significant potential of PSO for addressing VLSI routing problems. In addition, to further improve the optimization ability of the algorithm, we introduce the Social Learning Dynamic Particle Swarm Optimization (SLDPSO) algorithm to tackle VLSI routing challenges. Specifically, the proposed algorithm leverages the PSO technique and incorporates a social learning model based on the example pool. As shown in Figure 1, the example pool for the current particle X_i consists of particles, represented by dashed circles, with better fitness than X_i. X_i then randomly selects one particle from this example pool as its learning target. Compared to the traditional approach of learning from a single global optimum, *gbest*, the social learning model increases the diversity of the population, thereby enhancing the exploration ability of the algorithm. Therefore, SLDPSO can overcome the problem of premature convergence of PSO due to loss of diversity, thus searching for the unknown optimal design solution.

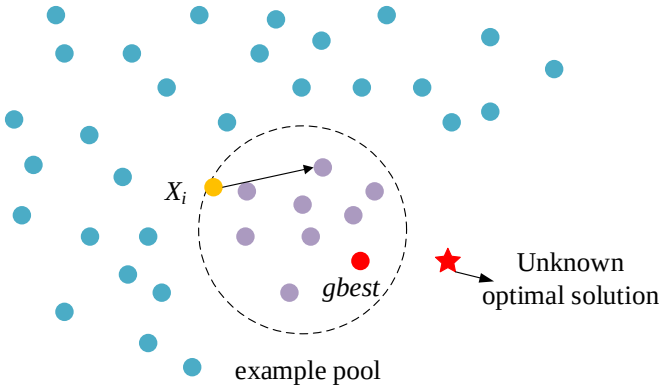

Figure 1. The social learning model.

2.2. Track Assignment

The track assignment phase is usually added between global routing and detailed routing to speed up detailed routing while improving the accuracy of global routing results. Existing track assignment algorithms are divided into two main categories: CFTA [35–43] and CMTA [44–52].

For the CFTA problem, plenty of works have emerged. Ref. [35] proposes a heuristic algorithm based on Weighted Bipartite Matching (WBM) to guide detailed routing. As the scale of the problem expands, the runtime of WBM increases significantly. With the continuous progress and development of the VLSI manufacturing process, factors such as crosstalk, yield, and others become increasingly important for chip performance, and receive growing attention. Refs. [36,37] consider the yield factor in track assignment. Considering both the timing and yield, Ref. [36] proposes a heuristic algorithm to greatly optimize the minimum timing slack, which transforms the mixed linear geometric programming problem of optimizing wire width and spacing into a convex optimization problem. Ref. [37] proposes a yield-driven track routing algorithm, which minimizes the critical region of the short circuit by finding the smallest Hamiltonian path to sort wires. Then it uses the second-order cone programming to optimize the wire width and spacing to achieve the minimum failure rate. Refs. [38–41] solve the crosstalk problem generated by the wires assigned to adjacent nets. Ref. [38] proposes a track assignment algorithm based on Integer Linear Programming (ILP) to optimize the crosstalk problem with a high runtime. Ref. [42] proposes a track assignment method to avoid native conflicts. Ref. [43] designs a track assignment strategy in the proposed initial routing algorithm to minimize initial routing width while avoiding conflicts in assigning PTL segments.

The above CFTA works have a large number of unassigned iroutes, which will cause serious short-circuit problems in detailed routing, and, thus, many CMTA works are proposed. Due to the limited routing resources, congestion has become an essential indicator of the quality evaluation of the track assignment solution. In order to perform rapid routability evaluation, Ref. [44] proposes a Negotiation-based Track Assignment (NTA) algorithm, which uses a greedy method to obtain an initial assignment under the premise of allowing conflicts. As a result, iterative negotiation is used to produce a routing solution with the least conflict. Ref. [45] proposes a Track Assignment Algorithm Based on Discrete Particle Swarm Optimization (DPSO-TA), which combines genetic operation and a negotiation-based refining strategy. Refs. [46,47] consider local nets, via location, and pin access in track assignment. Ref. [48] proposes the first track router with yield-driven wire planning to optimize yield loss induced by random defects. Ref. [49] uses the output of track assignment with local corrections combined with global shortest path search as the input of detailed routing to obtain efficient routing results. Ref. [50] first assigns all wires to tracks, then proposes an ILP model to minimize the overlap between iroutes. Ref. [51] creates tracks for each internal node, bump pad, and IO pad based on the flow passing through it, and then builds non-crossing routing paths by connecting tracks into track pairs. In addition, there are some works performed in their specific backgrounds. Ref. [53] proposes a new supervised learning approach to train a policy model to solve the track assignment-based detailed routing problems, but only for two-pin nets. Ref. [54] proposes a maze routing method, which does not apply to signal wire assignment in VLSI.

In general, most existing work tends to fall into local optima because of greedy strategies, or to be inefficient in large-scale problems. Therefore, considering the efficiency and quality of the solution, this paper utilizes the PSO algorithm with a social learning strategy to solve the CMTA problem and to find a better track assignment solution to improve the accuracy of the routability evaluation. The research gap between the SLDPSO-TA and previous research is shown in Table 1.

Table 1. Research gap.

Category	References	Characteristics
CFTA	[29–39]	There may be unassigned iroutes, which may lead to routing conflicts.
CMTA	[40–47]	The algorithms easily fall into local optima because of greedy strategies, or are inefficient in large-scale problems.
	[48,49]	The algorithms only work in specific backgrounds.
	the proposed SLDPSO-TA	PSO based on social learning can avoid premature convergence while maintaining high efficiency.

3. Problem Formulation and Evaluation Metrics

3.1. Problem Formulation

Figure 2a presents a multi-layer global routing model, where each dot is called a Global Routing Cell (*GRC*). In Figure 2b, the routing area consists of several sub-regions, each of which is a GRC.

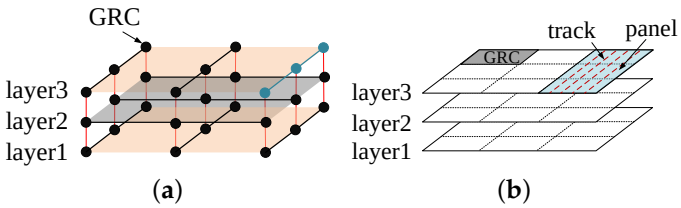

Figure 2. Multi-layer track assignment model. (**a**) Multi-layer global routing model. (**b**) Corresponding multi-layer track assignment model.

Definition 1 (Panel)**.** *In multi-layer routing problems, the routing direction is uniform on each layer, i.e., either horizontally or vertically. The routing directions of adjacent layers are intersected. A horizontal (vertical) panel consists of a set of GRCs in the horizontal (vertical) direction (the panels of layer 1 and 3 shown in Figure 2b are vertical, while the panels of layer 2 are horizontal).*

Definition 2 (Iroute)**.** *An iroute is defined as the straight-line path between the centers of two GRCs within a set covered by the routing area of a net.*

Definition 3 (Track)**.** *A track is the designated location for placing iroutes. In a horizontal (vertical) panel, a set of hypothetical horizontal (vertical) trajectory lines is referred to as tracks (as shown in Figure 2b).*

The track assignment problem is described as follows. Let L denote the number of layers, and n denote the number of panels on layer l. All information about each net, including the set of pins and iroutes, is provided. For each panel, given the set of tracks and iroutes, the objective is to assign all iroutes to tracks with the minimum cost. The cost of the track assignment comprises conflict cost and wirelength cost.

3.2. Evaluation Metrics

In this section, the specific calculation process of two primary metrics for evaluating the quality of track assignment solutions, i.e., conflict cost (including overlap cost and blockage cost) and wirelength cost, is presented.

3.2.1. Conflict Cost

Definition 4 (Conflict cost)**.** *Conflicts are composed of both overlap and blockage conflicts. An overlap conflict occurs when iroutes share the same track, while a blockage conflict arises when*

iroutes overlap with blockages. The conflict cost comprises the overlap cost due to overlap conflicts and the blockage cost resulting from blockage conflicts.

Overlap cost calculation: For a track assigned with m iroutes, it is represented by an interval that consists of a set of unit intervals, as shown in Figure 3. The value of each unit interval is defined as the number of iroutes in the interval, denoting a common subinterval of k $(2 \leq k \leq m)$ iroutes. For each common interval generated by k iroutes, the overlap cost is the product of the length l of this common interval and $(k-1)$. Specially, when $k = 0$ or $k = 1$, its overlap cost is 0. The overlap cost of a track is the sum of the overlap costs for all intervals comprising the track (the overlap cost of the track shown in Figure 3 is $l_1(k_1-1)+l_2(k_2-1)+l_3(k_3-1)=6$). The overlap cost of a panel is the sum of the overlap costs of all tracks in that panel. The total overlap cost of a track assignment solution is the sum of the overlap costs of all panels.

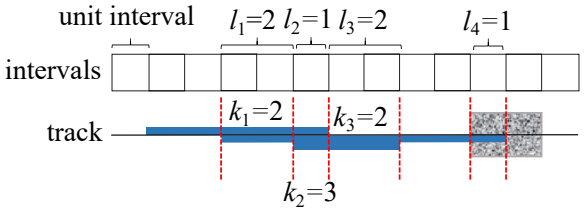

Figure 3. The conflict cost calculation for one track.

Blockage cost calculation: For an assigned iroute, its blockage cost is the sum of the overlap lengths between all blockages that overlap (partially or entirely) with it. The blockage cost of each track is the sum of the blockage costs of all iroutes on that track (e.g., the blockage cost of the track shown in Figure 3 is 1). In a panel, the blockage cost is the sum of blockage costs for all tracks. The blockage cost of the final track assignment result is the sum of the blockage costs of all panels.

3.2.2. Wirelength Cost

Definition 5 (Wirelength cost)**.** *The wirelength cost of a net is defined as the total length resulting from connecting all components, including pins and iroutes, within the net.*

A diagram illustrating the calculation of the wirelength cost for a net, which includes three iroutes and two pins, is shown in Figure 4. In Figure 4a, the top view of the net is depicted, where v_1 and ir_1 are located in the upper layer, v_2 and ir_3 are located in the lower layer, and ir_2 is located in the middle layer with vertical panels. Figure 4b shows a weighted complete graph with these five components as nodes, and the weight of each edge represents to the shortest Manhattan distance between the two corresponding components. A minimum spanning tree is then constructed to connect all components within the net, and its length represents the wirelength cost of the net.

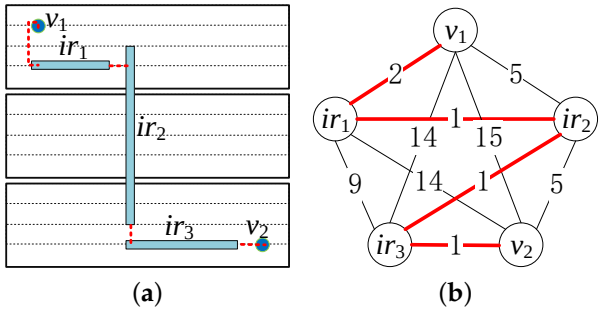

Figure 4. The wirelength cost calculation of a net. (**a**) The top view of the net. (**b**) The weighted completed graph corresponding to the net.

4. Details of SLDPSO-TA

We propose the SLDPSO-TA for addressing the CMTA problem, considering wirelength and conflict in local nets as optimization objectives. The proposed algorithm initially employs a hybrid approach, combining a greedy strategy with a genetic operator to generate high-quality "leader particles" for the population. Then, mutation and crossover operators are introduced to implement the particle updating phase. Moverover, a multi-point mutation strategy and a social learning mode based on an example pool are used to enhance the diversity of the population during the particle updating phase, thus strengthening the global search capability of the algorithm. Finally, a negotiation-based rip-up and reroute strategy is applied to further optimize the overlap cost of the track assignment.

4.1. Overall Design Flow

The overall design flow of the SLDPSO-TA is shown in Figure 5, which mainly consists of three stages, including the initialization stage, the particle update stage, and the refining stage. In the initialization stage, the global routing results are first loaded and tracks are created for each panel. Then, the iroutes are extracted from the global and local nets, and the initial population is generated using the initial assignment method based on the greedy strategy. In the particle update stage, velocity and position update operations are performed on the particles, and the fitness of the particles is calculated. Afterwards, the *pbest* of each particle, the example pool, and the *kbest* of the particles are updated based on the fitness of the particles, where hte *kbest* is the historical optimal solution for a particle in the sample pool that is better than the current particle. Next, the optimal solution, denoted as *gbest*, is updated. Specifically, the current *gbest* is compared with the *pbests* of all particles. If a *pbest* is better than the current *gbest*, then the current *gbest* is updated to that *pbest*. Once the termination condition (maximum number of iterations) is satisfied in the particle update stage, it proceeds to the next stage. Otherwise, velocity and position update operations are performed on particles. In the refining stage, the overlapping conflicts of the optimal solution *gbest* are optimized using the rip-up and reroute refining strategy. Finally, a high-quality track assignment solution can be generated.

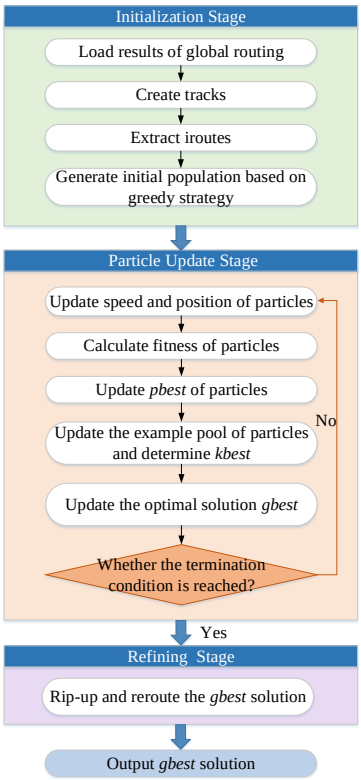

Figure 5. The flow chart of the SLDPSO-TA.

4.2. Wire Model Considering Local Nets

Definition 6 (Global net)**.** *A net is called a global net if it contains more than one GRC (e.g., n_1 in Figure 6a is a global net).*

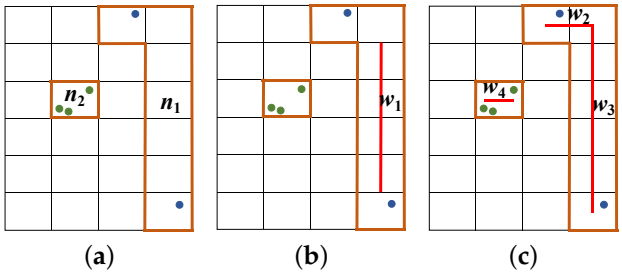

Figure 6. Wire model of SLDPSO-TA. (**a**) Global routing results of two nets. (**b**) Iroutes extracted by CFTA works. (**c**) Iroutes extracted by SLDPSO-TA.

Definition 7 (Local net)**.** *If all pins are located in the same GRC, such a net is called a local net (e.g., n_2 in Figure 6a is a local net).*

Extracting iroutes is a crucial phase in track assignment. The majority of existing track assignment algorithms overlook the routing information of local nets, and only focus on extracting iroutes from global nets, leading to significant information loss. Therefore, the proposed algorithm considers the pin distribution of local nets to enhance the extraction and assignment of iroutes.

For global nets, all straight lines connecting the centers of one GRC to another GRC are extracted as iroutes, effectively utilizing the routing information at the corners, as shown in Figure 6. For local nets, two rectilinear Steiner trees need to be constructed, namely the Steiner tree with a single horizontal or vertical trunk (as shown in Figure 7). The single vertical (horizontal) Steiner tree is constructed by initially constructing the vertical (horizontal) trunk. Then, all pins in the local net are connected to the trunk by horizontal (vertical) lines. The x-coordinate (y-coordinate) value of the vertical (horizontal) trunk is determined by the median of the x-coordinate (y-coordinate) values of all pins. The upper and lower y-coordinate values (left and right x-coordinate values) of the vertical (horizontal) trunk are set as the maximum and minimum values of the y-coordinate (x-coordinate) values of all the pins, respectively. Once Steiner trees in two different directions are constructed, the iroute extracted from this local net is determined by selecting the trunk of the shorter tree based on a comparison of their lengths.

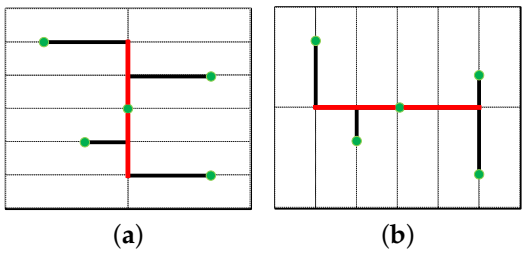

Figure 7. Wire model of the local net. (**a**) Steiner tree with a single vertical trunk. (**b**) Steiner tree with a single horizontal trunk.

4.3. Particle Encoding

To enhance the resolution of CMTA problems using PSO, an integer encoding method is proposed. In this particle encoding method, each particle corresponds to a track assignment solution for one panel. The length of the particle encoding is equal to the number of iroutes on that panel, and each bit in the encoding represents the track where the iroute is located. Both iroutes and tracks are encoded with integers.

As shown in Figure 8a, the panel has three tracks, nine iroutes, and one blockage, where the blockage is located on track 2. Figure 8b illustrates the specific conflict area of the panel (marked by the red dashed line). Noted that a blockage might be located on more than one track or span multiple adjacent tracks, depending on the size of the blockage. According to the aforementioned encoding method, the particle encoding corresponding to the track assignment solution in Figure 8a is "2 1 1 3 1 3 2 2 1". The value of the first bit of the encoding is 2, indicating that iroute ir_1 is assigned to track 2.

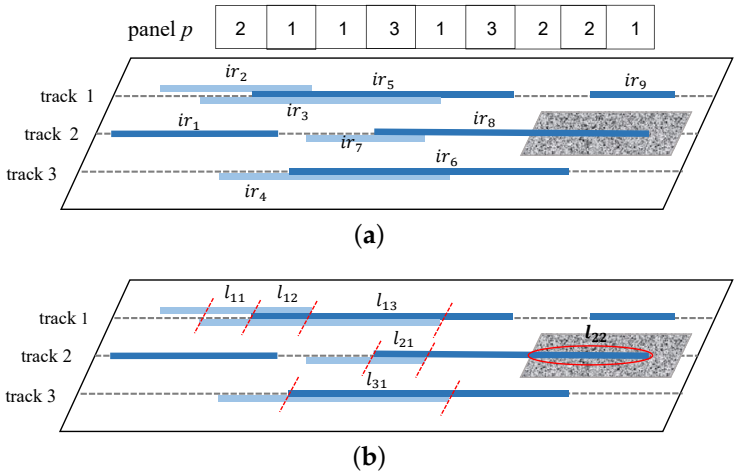

Figure 8. The track assignment solution of a panel (including 3 tracks, 9 iroutes, and a blockage). (**a**) Particle encoding. (**b**) The specifics of conflicts.

The integer encoding method introduced is able to satisfy the integrity, soundness, and non-redundancy criteria simultaneously [55]. The length of the particle is set to guarantee that each iroute within the panel can be assigned, and the range of the i-th encoding value of the particle includes all possible tracks to which iroute i can be assigned. Therefore, this particle encoding method can cover the entire solution space. For each solution, a unique particle can be found to correspond to it, so that the particles in the range of the encoding space align one-to-one with the track assignment solutions on the panel. This encoding method provides an intuitive representation of the assignment within each panel, offering simplicity in implementation. It can effectively expand the search space compared with the greedy method of NTA, thus improving the performance of the algorithm.

4.4. A Greedy Strategy-Based Initial Assignment Method

The time-consuming cost calculation limits the number of iterations of the particle swarm to a certain extent, which affects the convergence speed of the algorithm. Therefore, a greedy strategy-based initial assignment method is designed to generate initial particles with higher quality in the following steps:

(1) Track assignment is performed with different assignment orders based on the greedy criterion, resulting in four types of initial solutions.
(2) The mutation operator is introduced to generate the initial population incorporating the four types of initial solutions.
(3) The selection operator based on the fitness value evaluation is introduced to screen out high-quality particles for the population as the initial solution.

Table 2 compares the effects of different assignment orders on the track assignment results [44], where WL represents wirelength cost, and OC represents overlap cost. "Order 1", "order 2", "order 3", and "order 4" indicate that the initial track assignment is conducted in the order of iroute length from long to short, iroute length from short to long, the number of components from large to small, and the number of components from small to large,

respectively. As shown in Table 2, assigning in "order 1" yields the solution with the least overlap cost, while sacrificing a small amount of wirelength.

Table 2. Track assignment with four assignment orders.

Assignment Orders	Order 1		Order 2		Order 3		Order 4	
	WL	OC	WL	OC	WL	OC	WL	OC
Ratio	1	1	0.98	2.01	0.99	2.01	0.95	2.02

To explore more potential solutions, the SLDPSO-TA generates four distinct populations based on the above four assignment orders. Each population is generated by introducing mutations to particles corresponding to a specific order. The specific mutation phase is described in Section 4.6. Then, a selection operator is introduced to identify more promising particles for the population. The proposed algorithm enhances the exploration of potential solutions by assigning twice the number of particles to "order 1" compared to the other three orders. The pseudocode of the initial solution generation is provided in Algorithm 1, where Line 1 to Line 5 involve sorting iroutes using four different assignment orders. Lines 6 to 15 determine the particle counts for each order. Line 16 represents the mutation operation, and Line 18 involves sorting particles based on fitness values. Finally, Line 19 initializes the initial swarm, and set the initial population size is $M \times ratio$.

Algorithm 1 Initialization

Require: Routing panel *panel*, The scale of population M, Screening ratio *ratio*
Ensure: Initial particle swarm *swarm*[M]
1: **Begin**
2: *initAns*_1 = AssignByLltos(*panel* $\rightarrow$ *irouteList*);
3: *initAns*_2 = AssignByLstol(*panel* $\rightarrow$ *irouteList*);
4: *initAns*_3 = AssignByNltos(*panel* $\rightarrow$ *irouteList*);
5: *initAns*_4 = AssignByNstol(*panel* $\rightarrow$ *irouteList*);
6: **for** each particle p_i in the swarm **do**
7: **if** $i \geq 0$ && $i < M \times \frac{2}{5}$ **then**
8: Initialize p_i by *initAns*_1;
9: **else if** $i < M \times \frac{3}{5}$ **then**
10: Initialize p_i by *initAns*_2;
11: **else if** $i < M \times \frac{4}{5}$ **then**
12: Initialize p_i by *initAns*_3;
13: **else**
14: Initialize p_i by *initAns*_4;
15: **end if**
16: Mutation(p_i);
17: **end for**
18: SortParticlesByFit(*swarm*);
19: SetInitSwarm(*swarm*, *ratio*);
20: **End**

4.5. Fitness Calculation

Since the greedy strategy-based initial assignment method can produce a routing solution with a minimal wirelength cost, and the solution undergoes minimal changes during the particle swarm iterations, the proposed algorithm focuses on the conflict cost during the design of the fitness function, which is formulated as follows:

$$\text{fitFunc}(p) = overlapCost_p + \beta \times blkCost_p \quad (1)$$

where $overlapCost_p$ and $blkCost_p$ are the overlap cost and blockage cost of panel p, respectively, and β is a weighting factor. Since the overlap of the iroute with the blockage has a significant impact on routability, β is set to be 100,000 to strictly control the blockage cost.

The overlap cost $overlapCost_p$ in Equation (1) is computed as the sum of the overlap lengths of two iroutes on each track. In contrast to the calculation in Section 3.2, the calculation in this section distinctly emphasizes the overlap within the congested area. The overlap and blockage costs of each track, as well as the fitness values of the panel p in Figure 8b, are calculated as follows:

track 1: overlap cost: $l_{11} + 2l_{12} + l_{13}$, blockage cost: 0
track 2: overlap cost: l_{21}, blockage cost: l_{22}
track 3: overlap cost: l_{31}, blockage cost: 0
panel p: fitness value: $l_{11} + 2l_{12} + l_{13} + l_{21} + l_{31} + \beta \times l_{22}$

The proposed fitness calculation aims to mitigate severe conflicts by adjusting the weight of the overlap cost and blockage cost in congested areas.

Property 1. *The preprocessing strategy of the SLDPSO-TA can effectively reduce the number of blockage cost calculations, a claim that can be empirically substantiated in Section 5.2.*

The computation of blockage cost $blkCost_p$ at each iteration involves traversing all blockages in a single computation, proving time-consuming for large-scale problems. To address this challenge, a preprocessing strategy is introduced in this section to reduce the number of blockage cost calculations. After the initialization of the particle swarm, a look-up table of size $|I_p \times T_p|$ is promptly generated for each panel to store the overlap information between iroutes and blockages. Table 3 illustrates the blockage cost associated with placing iroute i on track t. By generating a look-up table of size 6×3 for the panel to pre-store the potential blockage cost of each iroute, the particle requires only $O(1)$ time to obtain the blockage cost of the iroute during the iterative process.

Table 3. A look-up table for conflict information of a panel.

Blockage Cost	ir_1	ir_2	ir_3	ir_4	ir_5	ir_6
tr_1	$b_{1,1}$	$b_{2,1}$	$b_{3,1}$	$b_{4,1}$	$b_{5,1}$	$b_{6,1}$
tr_2	$b_{1,2}$	$b_{2,2}$	$b_{3,2}$	$b_{4,2}$	$b_{5,2}$	$b_{6,2}$
tr_3	$b_{1,3}$	$b_{2,3}$	$b_{3,3}$	$b_{4,3}$	$b_{5,3}$	$b_{6,3}$

4.6. Particle Updating Formulation

In the proposed algorithm, we mainly incorporate crossover and mutation operators from a GA into the DPSO framework. These operators introduce genetic diversity into the particle population, thus enhancing the exploration capabilities of the algorithm. The crossover operator combines information from multiple particles, leading to better mixing of superior solutions and improving the ability of the algorithm to effectively explore the search space. On the other hand, the mutation operator helps to maintain the diversity of the particle population, thus preventing premature convergence to a local optimum and ensuring continued exploration for new regions of the solution space. Furthermore, since the SLPSO algorithm fails to apply directly to discrete track assignment problems due to its continuity, the crossover and mutation operators from the GA are applied to the particle updates of SLPSO. In this way, we can discretize the SLPSO algorithm and make it more suitable for solving the CMTA problem in this paper.

The particle update process of the SLDPSO-TA follows the following equation:

$$X_i^t = \mathrm{TF}_3(\mathrm{TF}_2(\mathrm{TF}_1(X_i^{t-1}, \omega), c_1), c_2) \tag{2}$$

where ω is the inertial weight, c_1 and c_2 are the acceleration factors. TF_1 is the inertial component, while TF_2 and TF_3 are the individual cognitive component and the social cognitive component, respectively. Similarly, the inertial component of the SLDPSO-TA is realized by the mutation operator, while the individual cognitive component and the social cognitive component are enacted by the crossover operator. Additionally, we propose a

novel encoding method and incorporate a social learning mode based on the example pool to address the CMTA problem.

Based on Equation (2), we propose the particle updating model as follows:

(1) Inertial component

TF_1 represents the inertial component of the particle and is achieved through the mutation operator as follows:

$$W_i^t = \mathrm{TF}_1(X_i^{t-1}, \omega) = \begin{cases} \mathrm{M_T}(X_i^{t-1}), & r_1 < \omega \\ X_i^{t-1}, & \text{otherwise} \end{cases} \tag{3}$$

where ω is the inertial weighting factor, $\mathrm{M_T}(\cdot)$ is a multi-point mutation operation used for track assignment, and r_1 is a randomly generated number distributed on the interval $[0, 1)$.

The implementation process of the multi-point mutation operation, denoted as MT$(\cdot)$ in Equation (3), is detailed below. The algorithm randomly selects num positions and updates the encoded values at these positions to integers within the interval $\left[1, T(p_{i,j})\right]$, where $T(p_{i,j})$ represents the total number of tracks in a panel, which is computed based on the wire width at each layer and the minimum spacing requirement between the wires. These details can be obtained from the global routing results of the test circuit.

Note that the variable num decreases linearly with the progression of iterations, leading to a more pronounced mutation effect on particles during the early stages of iteration. This effect gradually diminishes as the iteration progresses, thereby speeding up the convergence of the algorithm. Figure 9 demonstrates the mutation operation of the SLDPSO-TA. It shows the specific encoding conditions before and after the particle mutation when $num = 2$. It can be seen that the mutation operation leads to the iroute ir_4 and iroute ir_7, originally located on tracks 3 and 2, being reassigned to tracks 2 and 1 after mutation. Algorithm 2 shows the pseudocode of the multi-point mutation operation. Lines 1 to 6 set the number of mutation points, where *ir_size* is the number of iroutes. Lines 7 to 20 describe the mutation operation, where *tr_size* is the number of tracks.

Algorithm 2 Multi-point_Mutation_TA

Require: Particle p to be mutated
Ensure: The mutated particle p

```
1: Begin
2: num_init = 0.2 × ir_size;
3: num = DecFunc(num_init, 0, t);
4: if num < 1 then
5:     num = 1;
6: end if
7: while num > 0 do
8:     ir = Random(1, ir_size);
9:     while true do
10:        tr = Random(1, tr_size);
11:        if tr ≠ p[ir] then
12:            Update(p[ir], tr);
13:            break;
14:        else
15:            continue;
16:        end if
17:    end while
18:    num = num − 1;
19: end while
20: End
```

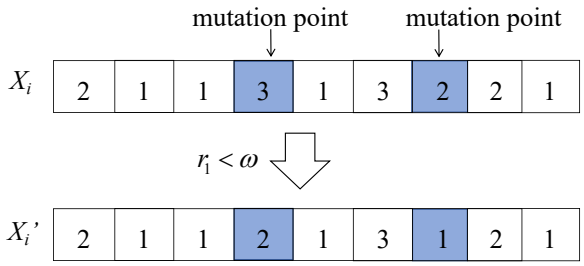

Figure 9. The multi-point mutation operation of the SLDPSO-TA.

(2) Individual cognitive component

TF_2 represents the individual cognitive component of the particle and is implemented by the crossover operator as follows:

$$S_i^t = \mathrm{TF}_2(W_i^t, c_1) = \begin{cases} \mathrm{C_T}(W_i^t, X_i^P), & r_2 < c_1 \\ W_i^t, & \text{otherwise} \end{cases} \tag{4}$$

where c_1 determines the probability that the particle crosses with its historical optimal solution X_i^P, $\mathrm{C_T}(\cdot)$ is a crossover operator with X_i^P as its crossover object, and r_2 is a randomly generated number distributed on the interval $[0, 1)$.

(3) Social cognitive component

TF_3 represents the social cognitive component of the particle and is implemented by the crossover operator as follows:

$$X_i^t = \mathrm{TF}_3(S_i^t, c_2) = \begin{cases} \mathrm{C_T}(S_i^t, X_k^P), & r_3 < c_2 \\ S_i^t, & \text{otherwise} \end{cases} \tag{5}$$

where c_2 determines the probability of the particle crossing with the historical optimal solution (X_k^P, where $1 \le k \le i-1$) of any particle in the learning example pool, $\mathrm{C_T}(\cdot)$ is a crossover operator with X_k^P as its crossover object, and r_3 is a randomly generated number in the interval $[0, 1)$.

Figure 10 shows the crossover operation of the SLDPSO-TA. The first line represents the learning object of the particle, which can be either the *pbest* X_i^P of particle i or the *pbest* X_k^P of any outstanding particle k in the example pool. The second and third lines represent the specific encodings before and after the crossover, respectively. The particle undergoes social learning by updating values in the crossover interval of length $clen (1 \le clen \le |I(p_{i,j})|)$ to match the values in the corresponding interval of the learning object, bringing it closer to either X_k^P or X_i^P. Based on the example pool, the social learning objects of all particles are intentionally diversified in the same iteration. This mechanism provides particles with the opportunity to explore a broader range of solutions. Algorithm 3 shows the pseudocode of the crossover operation, where Lines 1 to 6 define the crossover interval, and Lines 7 to 10 perform the crossover operation.

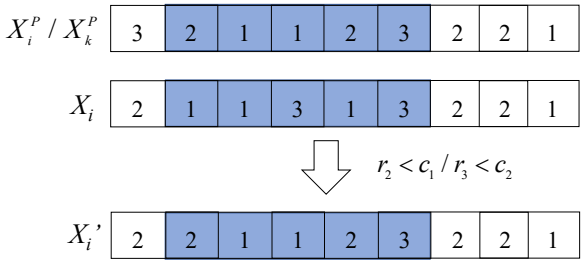

Figure 10. The crossover operation of the SLDPSO-TA.

Algorithm 3 Crossover_TA

Require: Particle p to be crossed, Learning object q
Ensure: Particle after crossover
1: **Begin**
2: $ir_1 = \text{Random}(1, ir\_size)$;
3: $ir_2 = \text{Random}(1, ir\_size)$;
4: **if** $ir_1 > ir_2$ **then**
5: Swap(ir_1, ir_2);
6: **end if**
7: **for** $ir : ir_1$ to ir_2 **do**
8: Update$(p[ir], q[ir])$;
9: **end for**
10: **End**

4.7. Refining Strategy

Overlap conflict is a critical metric in track assignment. To further improve the quality of the track assignment solution, the SLDPSO-TA employs a negotiation-based rip-up and reroute strategy to refine the solution. This refining strategy can reasonably rip up and reassign iroutes in congested areas to reduce the congestion across the entire panel.

Definition 8 (History cost)**.** *The history cost is defined as the number of overlaps between iroutes or between iroutes and blockages, resulting from the reassignment operation. Specifically, for an iroute on a track, the history cost is the sum of the history costs associated with all unit intervals covered by that iroute. Each of these unit intervals has a history cost whose value is the number of overlaps caused by the reassignment operation on that interval.*

The history cost is a crucial metric to accurately reflect the congestion. Therefore, the history cost is considered a vital component in the cost calculation of the refining strategy. At the beginning of the refining strategy, the history cost of each unit interval on each track is initialized to 0. When an iroute is reassigned and conflicts occur, the value of each unit interval corresponding to the overlapping part is increased by 1. The schematic of the history cost calculation is shown in Figure 11. It can be seen from Figure 11 that the coded values of the unit intervals between the dashed lines are all increased by 1.

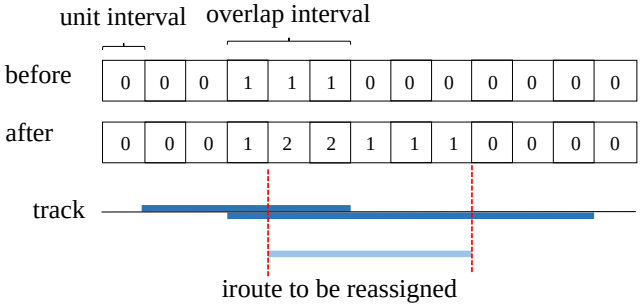

Figure 11. The history cost calculation.

The refining strategy of the SLDPSO-TA is organized into the following two phases:

(1) Select an iroute for reassignment. An iroute with the maximum cost is selected by using the following cost function:

$$\text{removeCost}(ir, tr) = overlapCost_{ir,tr} + historyCost_{ir,tr} \quad (6)$$

where $overlapCost_{ir,tr}$ and $historyCost_{ir,tr}$ are the overlap cost reduced when the iroute ir is removed from the track tr and the history cost of ir, respectively.

(2) Determine the target track. The track to which the iroute is reassigned is determined using the following cost function:

$$\begin{aligned} \text{addCost}(ir, tr) &= 0.1 \times wlCost_{ir,tr} + \alpha \times overlapCost_{ir,tr} \\ &\quad + \beta \times blkCost_{ir,tr} + historyCost_{ir,tr} \end{aligned} \tag{7}$$

where $wlCost_{ir,tr}$ represents the wirelength cost when iroute ir is on track tr. Additionally, $overlapCost_{ir,tr}$ and $blkCost_{ir,tr}$ denote the increased overlap cost and blockage cost, respectively, resulting from assigning ir to track tr. The coefficients α and β are utilized to adjust the weights of the overlap cost and blockage cost.

Note that the value of α increases with the number of iterations. This ensures the optimization of iroutes in the early phases of iterations, gradually shifting the focus towards the optimization of overlap conflicts. Meanwhile, β is still set to 100,000 to prevent an increase in blockage conflicts.

4.8. Time Complexity Analysis

Lemma 1. *Given a panel with I iroutes, N pins, and B blockages, let the population size be M, the number of iterations of the particles be T, and the number of iterations of the refining strategy be r. Then, the approximate time complexity of the SLDPSO-TA is* $O(MT \times (I^2 + M\log_2 M) + r \times (I^2 + N + B))$.

Proof. We analyze the time complexity of the SLDPSO-TA in three parts: the greedy strategy-based initial assignment, the iteration of particles, and the refining strategy. □

(1) In *Step 2*, the initial population solution of the SLDPSO-TA is generated, and the iroutes are first sorted with the time complexity of $O(I\log_2 I)$. Then, the greedy strategy-based assignment method is performed in four different orders. This assignment process involves calculating the cost associated with the assignment to determine the assigned track.

We analyze time complexity from two parts, i.e., the calculation of the wirelength cost and conflict cost, respectively. First, we consider the wirelength cost incurred by iroutes when they are placed on the individual tracks. Since the wirelength cost of local nets takes only linear time, the time complexity of this part is mainly influenced by iroutes in global nets. Specifically, the calculation involves determining the distance between the iroute placed on each track and all components of its net, resulting in a time complexity of $O(I + N)$. Second, we consider the conflict cost caused by an iroute being placed on a track. This part requires traversing all blockages and assigned iroutes on the track where the iroute is located, resulting in a time complexity of $O(I + B)$. Consequently, the time complexity of the greedy assignment phase to assign all iroutes is $O(I \times (I + N + B))$. The mutation and selection operators are used to screen the initial population, where the complexity of the mutation operation is constant time and the complexity of the selection operation depends on the sorting algorithm as $O(M\log_2 M)$. In summary, the approximate time complexity of this phase is $O(I \times (I + N + B) + M\log_2 M)$.

(2) In *Steps 3–6*, the iterative part of the particles includes the mutation operation, the crossover operation, and the fitness calculation. The time complexity of both mutation and crossover operations is constant time. The fitness calculation includes the calculation of the blockage cost and the overlap cost. Since the blockage cost is calculated using a look-up table-based preprocessing strategy, which takes only constant time, the time complexity of blockage cost calculation for the whole track assignment solution is $O(I)$. For overlap cost calculation, all iroutes on each track are traversed twice, leading to a complexity of $O(I^2)$. Updating the example pool for each particle is mainly dependent on the sorting algorithm, resulting in a complexity of $O(M\log_2 M)$. Considering the population size M and the number of iterations T, the overall time complexity is thus $O(MT \times (I^2 + M\log_2 M))$.

(3) In *Step 8*, as the history cost of the relevant iroutes on the original track needs to be updated at the end of each rip-up and reroute operation, its time complexity is approximately $O(I)$. In each iterative operation, iroutes to be reassigned are first selected according to Equation (6). The time of this part is mainly determined by the overlap cost calculation time for ripping up iroutes, with the time complexity of $O(I^2)$. Next, a track

is selected for the ripped-up iroute using Equation (7). It is required here to calculate the wirelength cost after reassignment, the increased overlap cost, the blockage cost, and the sum of the history cost, with a time complexity of $O(I+N+B)$. Thus, the time complexity of the refining strategy for r iterations is approximated $O(r \times (I^2+N+B))$.

The sum of the time complexity of the above three parts is $O(I \times (I+N+B) + M\log_2 M + MT \times (I^2+M\log_2 M) + r \times (I^2+N+B))$. Compared with the other two parts, the time complexity of the first part is negligible due to its small order of magnitude. Therefore, the overall time complexity of the SLDPSO-TA is $O(MT \times (I^2+M\log_2 M) + r \times (I^2+N+B))$.

5. Experimental Results

The proposed algorithm, the SLDPSO-TA, was implemented in the C/C++ language and tested on a PC with a 2.6GHz CPU and 64GB of memory. To obtain global routing results as benchmarks for the SLDPSO-TA, we adopted the global router NCTUgr [56] to route the placement results generated by NTUplace4 [57] for the test circuit DAC2021 [58]. The benchmarks sb2 to sb19 are representative of modern industrial ASIC designs, with numerous placement and routing blockages, more metal layers, varying width and spacing across layers, etc. By experimenting with these benchmarks, this paper demonstrates that the SLDPSO-TA can provide an excellent solution to real-world circuit problems and further advance research in routability track assignment.

Due to the large scale of the problem, we configured the parameters as follows: $M = 20$ and $iter = 400$. The inertia weight factor linearly decreases from 0.95 to 0.4, the learning factor c_1 linearly decreases from 0.9 to 0.15, and the learning factor c_2 linearly increases from 0.4 to 0.9. The parameter settings for the refining strategy align with those specified in [44].

5.1. Validation of Wire Model Considering Local Nets

To validate the importance of the local net, the extraction of iroutes is detailed in Table 4. This table presents the following metrics for each benchmark: the total number of nets (TN), the number of local nets (LN), the ratio of LN to TN (LN/TN), the number of extracted total iroutes (TI), the number of iroutes extracted from local nets (LI), the ratio of LI to TI (LI/TI), and the ratio of the number of assigned iroutes (TI/(TI-LI)). The last column in the table displays the ratio of the number of assigned iroutes, indicating the ratio of the iroutes extracted and assigned by the SLDPSO-TA to those in the previous work (without considering local nets).

Table 4. The nets and extracted iroutes of each benchmark.

Bench	TN	LN	LN/TN (%)	TI	LI	LI/TI (%)	TI/(TI-LI)
sb2	990,899	304,025	30.68	2,134,703	304,025	14.24	1.17
sb3	898,001	339,078	37.76	2,017,313	339,078	16.81	1.20
sb6	1,006,629	350,741	34.84	2,037,526	350,741	17.21	1.21
sb7	1,340,418	440,442	32.86	2,883,168	440,442	15.28	1.18
sb9	833,808	317,925	38.13	1,671,043	317,925	19.03	1.23
sb11	935,731	275,351	29.43	1,811,827	275,351	15.20	1.18
sb12	1,293,436	361,900	27.98	2,719,712	361,900	13.31	1.15
sb14	619,815	178,114	28.74	1,261,573	178,114	14.12	1.16
sb16	697,458	227,932	32.68	1,371,803	227,932	16.62	1.20
sb19	511,685	187,695	36.68	995,611	187,695	18.85	1.23
Avg			32.98			16.07	1.19

It can be seen from Table 4 that local nets constitute a significant proportion of the total nets, with an average of 32.98% of the total nets. Moreover, iroutes extracted from local nets account for 16.07% of the total iroutes on average. The SLDPSO-TA makes full use of information from local nets to significantly increase the number of assigned iroutes. Compared with the works that do not consider local nets, it can extract an additional 19% more iroutes on average, thereby enhancing the accuracy of the track assignment solution.

5.2. Validation of Preprocessing Strategy

The SLDPSO-TA employs a look-up table-based preprocessing strategy to reduce the number of blockage cost calculations during particle iteration. Table 5 shows the number of blockage cost calculations before and after implementing the preprocessing strategy. The second and third columns of Table 5 depict the total number of extracted iroutes (TI) and the total number of blockages (TB) for each test circuit, respectively. The last column depicts the ratio of the number of blockage cost calculations before and after implementing the preprocessing strategy. The experimental results demonstrate that the number of blockage cost calculations is 297 times higher without the preprocessing strategy, indicating the effectiveness of the look-up table-based preprocessing strategy in reducing redundant computations.

Table 5. The number of blockage cost calculations with and without the preprocessing strategy.

Bench	TI	TB	The Calculation Times of Blockage Cost (10^8)		Ratio (Without Preprocessing Strategy/ with Preprocessing Strategy)
			Without Preprocessing Strategy	With Preprocessing Strategy	
sb2	2,134,703	30,975	60,898. 40	202.76	300
sb3	2,017,313	16,323	20,929.20	69.96	299
sb6	2,037,526	16,588	25,897.59	86.25	300
sb7	2,883,168	11,095	37,793.37	125.71	301
sb9	1,671,043	9261	17,311.78	57.61	300
sb11	1,811,827	12,660	25,162.48	83.81	300
sb12	2,719,712	3660	12,430.54	43.76	284
sb14	1,261,573	10,337	6622.52	22.25	298
sb16	1,371,803	8891	1503.28	5.20	289
sb19	995,611	9666	11,110.70	37.18	299
Avg					297

5.3. Validation of Refining Strategy

The negotiation-based refining strategy employs rip-up and reroute techniques to alleviate congestion by relocating iroutes from congested areas to unoccupied tracks, which significantly affects the further optimization of overlap conflicts. Table 6 shows the comparison results of the proposed algorithms with and without the refining strategy. It shows that the refining strategy achieves an average optimization of 26.61% in overlap cost, with only a 6.95% increase in wirelength cost, while ensuring that the blockage conflicts are not degraded. Note that the most optimized overlap conflict is sb12, where the overlap cost can be reduced by 43.88%. Even in the least optimized benchmark, namely sb1, there is still a 16.34% reduction in overlap cost.

Table 6. Comparison of optimization performance of SLDPSO-TA with and without refining strategy.

Bench	Without Refining Strategy			With Refining Strategy			Optimization Rate (%)		
	WL (10^6)	OC (10^6)	BC (10^6)	WL (10^6)	OC (10^6)	BC (10^6)	WL	OC	BC
sb2	25.3153	3.2743	0.337	26.8770	2.7393	0.337	−6.17	16.34	0.00
sb3	24.0629	2.4034	0.3251	25.3798	1.9507	0.3251	−5.47	18.84	0.00
sb6	24.9646	2.1221	0.2305	26.7851	1.6457	0.2305	−7.29	22.45	0.00
sb7	37.7512	2.1602	0.2114	40.3764	1.5342	0.2114	−6.95	28.98	0.00
sb9	20.8546	1.3781	0.1459	22.6642	0.9656	0.1459	−8.68	29.93	0.00
sb11	22.2498	1.1579	0.2235	23.7883	0.7856	0.2235	−6.91	32.15	0.00
sb12	36.1468	1.4251	0.0351	39.1348	0.7997	0.0351	−8.27	43.88	0.00
sb14	15.4885	1.0563	0.2596	16.2734	0.8168	0.2596	−5.07	22.67	0.00
sb16	16.6469	1.5372	0.1422	17.6732	1.2453	0.1422	−6.17	18.99	0.00
sb19	12.2924	0.7302	0.0863	13.3398	0.4972	0.0863	−8.52	31.91	0.00
Avg							−6.95	26.61	0.00

Since the overlap conflict is a critical issue in routing, we prioritize the optimization of the overlap cost as the primary objective for routability. The experimental results demonstrate that while the refining strategy involves a tradeoff in wirelength, it leads to a substantial reduction in overlapping conflicts.

5.4. Validation of Social Learning Mode

To enhance the exploration capability of the algorithm, the social learning model based on the example pool is applied to the discrete PSO for the CMTA problem, thereby improving the quality of the solution. To verify the effectiveness of the social learning model, Table 7 shows the comparison results of the DPSO-TA [45] without the social learning model and the SLDPSO-TA with this model in terms of wirelength cost (WL), overlap cost (OC), and blockage cost (BC). Note that the DPSO-TA realizes the social cognition of particles by cross-learning with *gbest* during the particle swarm iteration. It can be seen that the proposed algorithm is able to reduce the overlap cost by 0.36% on average while keeping the wirelength cost and blockage cost almost unchanged. The experimental results demonstrate that the proposed algorithm is capable of searching for track assignment solutions with high quality by using the social learning model based on the example pool.

Table 7. Comparison of SLDPSO-TA and DPSO-TA.

Benchmark	DPSO-TA			SLDPSO-TA			Optimization Rate (%)		
	WL (10^6)	OC (10^6)	BC (10^6)	WL (10^6)	OC (10^6)	BC (10^6)	WL	OC	BC
sb2	25.3148	3.2958	0.337	25.3153	3.2743	0.337	0.00	0.65	0.00
sb3	24.0618	2.4123	0.3251	24.0629	2.4034	0.3251	0.00	0.37	0.00
sb6	24.9597	2.1298	0.2305	24.9646	2.1221	0.2305	−0.02	0.36	0.00
sb7	37.7483	2.1693	0.2114	37.7512	2.1602	0.2114	−0.01	0.42	0.00
sb9	20.8509	1.3846	0.1459	20.8546	1.3781	0.1459	−0.02	0.47	0.00
sb11	22.2445	1.1653	0.2235	22.2498	1.1579	0.2235	−0.02	0.64	0.00
sb12	36.1442	1.4307	0.0351	36.1468	1.4251	0.0351	−0.01	0.39	0.00
sb14	15.4887	1.0591	0.2596	15.4885	1.0563	0.2596	0.00	0.26	0.00
sb16	16.6473	1.5401	0.1423	16.6469	1.5372	0.1422	0.00	0.19	0.07
sb19	12.2924	0.7291	0.0863	12.2924	0.7302	0.0863	0.00	−0.15	0.00
Average							−0.01	0.36	0.01

5.5. Comparison with Existing Track Assignment Algorithms

The proposed algorithm employs the SLDPSO method based on the example pool to find a track assignment solution with higher quality, and then adopts a negotiation-based refining strategy to further reduce overlap conflicts. To validate the effectiveness of the proposed algorithm, in this section, we compare the SLDPSO-TA with two existing track assignment algorithms, namely WBM-TA [35] and NTA [44]. The comparison results are shown in Table 8 and Table 9, respectively.

Table 8 shows the comparison results between the proposed algorithm and WBM-TA in terms of WL, OC, and BC. It can be seen that the performance of the proposed algorithm outperforms WBM-TA, with an average optimization of 8.94% and 76.09% for wirelength cost and overlap cost, respectively. In particular, benchmark sb12 has the best overlap cost optimization, which can reach up to 87.72%, while benchmark sb3 has the best wirelength optimization, which can reduce the wirelength cost by 12.52%.

Moreover, Table 9 likewise shows the comparison results of the proposed algorithm with NTA in terms of WL, OC, and BC. It can be seen from Table 9 that the SLDPSO-TA still produces better results across all the benchmarks, with an average optimization of 1.85% in overlap cost, while maintaining the wirelength cost and blockage cost almost unchanged. The results above demonstrate the effectiveness of the SLDPSO-TA and the excellent optimization performance in terms of overlap cost.

Table 8. Comparison of SLDPSO-TA and WBM-TA.

Benchmark	WBM-TA			SLDPSO-TA			Optimization Rate (%)		
	WL (10^6)	OC (10^6)	BC (10^6)	WL (10^6)	OC (10^6)	BC (10^6)	WL	OC	BC
sb2	29.1351	12.2716	0.337	26.8770	2.7393	0.337	7.75	77.68	0.00
sb3	29.0131	5.7738	0.3251	25.3798	1.9507	0.3251	12.52	66.21	0.00
sb6	29.7168	5.7673	0.2305	26.7851	1.6457	0.2305	9.87	71.46	0.00
sb7	43.9719	6.78	0.2114	40.3764	1.5342	0.2114	8.18	77.37	0.00
sb9	24.6343	3.967	0.1459	22.6642	0.9656	0.1459	8.00	75.66	0.00
sb11	26.0517	4.0935	0.2235	23.7883	0.7856	0.2235	8.69	80.81	0.00
sb12	41.0377	6.5105	0.0351	39.1348	0.7997	0.0351	4.64	87.72	0.00
sb14	18.3229	3.3449	0.2596	16.2734	0.8168	0.2596	11.19	75.58	0.00
sb16	19.5534	4.2229	0.1422	17.6732	1.2453	0.1422	9.62	70.51	0.00
sb19	14.6558	2.2504	0.0863	13.3398	0.4972	0.0863	8.98	77.91	0.00
Average							8.94	76.09	0.00

Table 9. Comparison of SLDPSO-TA and NTA.

Benchmark	NTA			SLDPSO-TA			Optimization Rate (%)		
	WL (10^6)	OC (10^6)	BC (10^6)	WL (10^6)	OC (10^6)	BC (10^6)	WL	OC	BC
sb2	26.8796	2.7856	0.337	26.8770	2.7393	0.337	0.01	1.66	0.00
sb3	25.3666	1.9713	0.3251	25.3798	1.9507	0.3251	−0.05	1.04	0.00
sb6	26.7965	1.6645	0.2305	26.7851	1.6457	0.2305	0.04	1.13	0.00
sb7	40.3769	1.5572	0.2114	40.3764	1.5342	0.2114	0.00	1.48	0.00
sb9	22.6652	0.9801	0.1459	22.6642	0.9656	0.1459	0.00	1.48	0.00
sb11	23.7821	0.8133	0.2235	23.7883	0.7856	0.2235	−0.03	3.41	0.00
sb12	39.1232	0.8177	0.0351	39.1348	0.7997	0.0351	−0.03	2.20	0.00
sb14	16.2738	0.8398	0.2596	16.2734	0.8168	0.2596	0.00	2.74	0.00
sb16	17.6729	1.265	0.1422	17.6732	1.2453	0.1422	0.00	1.56	0.00
sb19	13.3439	0.5065	0.0863	13.3398	0.4972	0.0863	0.03	1.84	0.00
Average							0.00	1.85	0.00

5.6. Routability Evaluation

Track assignment serves as a crucial intermediate stage, connecting global routing and detailed routing, leveraging information from global routing to guide the detailed routing. Moreover, compared with global routing, the routability evaluation can be performed more efficiently through track assignment considering local nets, leading to accurate prediction of congested routing areas.

To verify the effectiveness of the proposed algorithm for the routability evaluation of different placers, we test circuits placed by the placers NTUplace4 [57], SimPLR [59], and Ripple [60], respectively. As shown in Table 10, the last row shows the ratio of the average wirelength cost, overlap cost, and blockage cost derived from the SLDPSO-TA for the evaluation of the three placement results. It shows that routing on the results derived from SimPLR has the highest wirelength and conflicts, while the placement result using NTUplace4 has the highest routability. Therefore, routing based on NTUplace4 results is the preferred choice among the above three placers. The above results also demonstrate that the proposed algorithm, the SLDPSO-TA, can be useful in selecting placers with higher routability, thereby providing enhanced guidance for subsequent routing tasks.

Table 10. Comparison of track assignment results on NTUplace4, SimPLR and Ripple.

Benchmark	NTUplace4			SimPLR			Ripple		
	WL (10^6)	OC (10^6)	BC (10^6)	WL (10^6)	OC (10^6)	BC (10^6)	WL (10^6)	OC (10^6)	BC (10^6)
sb2	26.8770	2.7393	0.337	31.0425	3.5038	0.3794	28.472	3.3361	0.3871
sb3	25.3798	1.9507	0.3251	27.6148	2.6558	0.3537	26.5608	2.8655	0.3134
sb6	26.7851	1.6457	0.2305	27.9432	1.9509	0.2549	27.2141	2.1161	0.2359
sb7	40.3764	1.5342	0.2114	46.6648	4.6852	0.5237	42.6287	2.7672	0.2696
sb9	22.6642	0.9656	0.1459	23.789	1.3044	0.1611	23.9412	1.5628	0.1407
sb11	23.7883	0.7856	0.2235	24.7795	1.2663	0.2752	24.7921	1.1349	0.2309
sb12	39.1348	0.7997	0.0351	41.1449	0.9209	0.0821	40.8504	1.2754	0.0517
sb14	16.2734	0.8168	0.2596	17.3188	1.1231	0.2797	16.8203	1.0911	0.251
sb16	17.6732	1.2453	0.1422	18.7137	1.6751	0.2209	17.5417	1.3284	0.1803
sb19	13.3398	0.4972	0.0863	14.1174	0.6448	0.1258	14.198	0.7331	0.1032
Ratio	1	1	1	1.0766	1.5011	1.4560	1.0411	1.4311	1.1312

6. Conclusions

We have studied the CMTA problem considering the routing resources in both the global and local nets, and proposed a track assignment algorithm based on social learning discrete particle swarm optimization to solve it. By designing a simple and effective integer coding method, the search space is expanded to improve the performance of the algorithm. Moreover, an effective fitness function is designed to optimize both the conflict cost and the wirelength cost. Then, a higher-quality initial population is provided for the SLDPSO-TA based on four different routing assignment orders and a combination of mutation and selection operators. Furthermore, the social learning model based on the example pool is introduced to enhance population diversity during the particle updating process. Finally, a refining strategy is utilized to further optimize the track assignment solution. The performance of the SLDPSO-TA is validated on industrial test cases. The experimental results show that compared with existing algorithms, our algorithm has the best overlap-conflict optimization ability for industrial track assignment problems, which is of great reference significance for solving the actual chip routing problems. Moreover, the SLDPSO-TA can provide placers with more reliable routability evaluation and provide good guidance for chip design and implementation. Future research can be carried out in two directions. The first direction involves improving the performance of SLDPSO by integrating various domain topologies and mutation and crossover operators, and employing diverse cooperation and competition mechanisms. The second direction considers more routing factors, such as crosstalk and timing convergence, to provide more effective guidance for the execution of detailed routing.

Author Contributions: Conceptualization, H.C., R.Z., Y.J. and G.L.; methodology, H.C., R.Z. and Y.J.; software, H.C., R.Z. and P.H.; validation, H.C. and P.H.; formal analysis, R.Z. and G.L.; investigation, P.H. and Y.J.; resources, H.C., R.Z. and P.H.; data curation, P.H.; writing—original draft preparation, H.C., Y.J. and G.L.; writing—review and editing, H.C.; visualization, P.H. and Y.J.; supervision, G.L.; project administration, G.L.; funding acquisition, G.L. There have been updates to the authorship and contributions since the preprint paper [61] was published. Y.J., a master's student at the time of the preprint, has since graduated and is no longer involved in the ongoing work for this manuscript. H.C., a current student, has taken over responsibility for this work, including extensive revisions and refinements to the content based on the preprint version. H.C. is now handling all subsequent review, editing, and related matters for this submission, with assistance from students R.Z. and P.H. and G.L. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Fujian Natural Science Funds under Grant No. 2023J06017 and the National Natural Science Foundation of China under Grant No. 62372109.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Qiu, Y.; Xing, Y.; Zheng, X.; Gao, P.; Cai, S.; Xiong, X. Progress of Placement Optimization for Accelerating VLSI Physical Design. *Electronics* **2023**, *12*, 337. [CrossRef]
2. Huang, Z.; Huang, H.; Shi, R.; Li, X.; Zhang, X.; Chen, W.; Wang, J.; Zhu, Z. Detailed placement and global routing co-optimization with complex constraints. *Electronics* **2021**, *11*, 51. [CrossRef]
3. Liu, G.; Wei, L.; Yu, Y.; Xu, N. A High-Quality and Efficient Bus-Aware Global Router. *Chin. J. Electron.* **2024**, *34*, 1–13.
4. Jiang, Y.J.; Fang, S.Y. Pin Access-Oriented Concurrent Detailed Routing. In Proceedings of the International Symposium on Physical Design, Virtual, 26–29 March 2023; pp. 17–25.
5. Sun, Y.; Pan, J.S.; Hu, P.; Chu, S.C. Enhanced Equilibrium Optimizer algorithm applied in job shop scheduling problem. *J. Intell. Manuf.* **2023**, *34*, 1639–1665. [CrossRef]
6. Dao, T.K.; Pan, T.S.; Nguyen, T.T.; Pan, J.S. Parallel bat algorithm for optimizing makespan in job shop scheduling problems. *J. Intell. Manuf.* **2018**, *29*, 451–462. [CrossRef]
7. Liu, G.; Chen, Z.; Zhuang, Z.; Guo, W.; Chen, G. A unified algorithm based on HTS and self-adapting PSO for the construction of octagonal and rectilinear SMT. *Soft Comput.* **2020**, *24*, 3943–3961. [CrossRef]
8. Ijemaru, G.K.; Ang, K.L.M.; Seng, J.K.P. Swarm intelligence techniques for mobile wireless charging. *Electronics* **2022**, *11*, 371. [CrossRef]
9. Elmagzoub, M.; Syed, D.; Shaikh, A.; Islam, N.; Alghamdi, A.; Rizwan, S. A survey of swarm intelligence based load balancing techniques in cloud computing environment. *Electronics* **2021**, *10*, 2718. [CrossRef]
10. Mavrovouniotis, M.; Li, C.; Yang, S. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. *Swarm Evol. Comput.* **2017**, *33*, 1–17. [CrossRef]
11. Liu, X.; Du, Y.; Jiang, M.; Zeng, X. Multiobjective particle swarm optimization based on network embedding for complex network community detection. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 437–449. [CrossRef]
12. Chen, J.; Wu, Y.; Xu, X.; Zheng, H.; Ruan, Z.; Xuan, Q. PSO-ANE: Adaptive network embedding with particle swarm optimization. *IEEE Trans. Comput. Soc. Syst.* **2019**, *6*, 649–659. [CrossRef]
13. Census, C.; Wang, H.; Zhang, J.; Deng, P.; Li, T. Particle subswarms collaborative clustering. *IEEE Trans. Comput. Soc. Syst.* **2019**, *6*, 1165–1179. [CrossRef]
14. Attia, A.F.; Elaziz, M.A.; Hassanien, A.E.; El-Sehiemy, R.A. Prediction of solar activity using hybrid artificial bee colony with neighborhood rough sets. *IEEE Trans. Comput. Soc. Syst.* **2020**, *7*, 1123–1130. [CrossRef]
15. Cheng, B.; Lu, H.; Xu, X.; Shen, W. Improved local search chaotic discrete particle swarm optimization algorithm for solving traveler's problem. *J. Comput. Appl.* **2016**, *36*, 138–142. (In Chinese)
16. Mudjihartono, P.; Jiamthapthaksin, R.; Tanprasert, T. Parallelized GA-PSO algorithm for solving job shop scheduling problem. In Proceedings of the International Conference on Science in Information Technology, Balikpapan, Indonesia, 26–27 October 2016; pp. 103–108.
17. Mahmud, S.; Chakrabortty, R.; Abbasi, A.J.; Ryan, M. Switching strategy-based hybrid evolutionary algorithms for job shop scheduling problems. *J. Intell. Manuf.* **2022**, *33*, 1939–1966. [CrossRef]
18. Peng, Z.; Zhang, H.; Tang, H.; Feng, Y.; Yin, W. Research on flexible job-shop scheduling problem in green sustainable manufacturing based on learning effect. *J. Intell. Manuf.* **2022**, *33*, 1725–1746. [CrossRef]
19. Du, J.X.; Huang, D.S.; Zhang, J.; Wang, X.F. Shape matching using fuzzy discrete particle swarm optimization. In Proceedings of the Swarm Intelligence Symposium, Pasadena, CA, USA, 8–10 June 2005; pp. 405–408.
20. Sharma, A.; Kapur, R.K. Image enhancement using hybrid GSA—Particle swarm optimization. In Proceedings of the International Conference on Contemporary Computing and Informatics, Greater Noida, India, 14–17 December 2016; pp. 698–704.
21. Cohen, S.C.; de Castro, L.N. Data clustering with particle swarms. In Proceedings of the International Conference on Evolutionary Computation, Vancouver, BC, Canada, 16–21 July 2006; pp. 1792–1798.
22. Wang, C.; Zhang, Y.; Song, J.; Liu, Q.; Dong, H. A novel optimized SVM algorithm based on PSO with saturation and mixed time-delays for classification of oil pipeline leak detection. *Syst. Sci. Control. Eng.* **2019**, *7*, 75–88. [CrossRef]
23. Liu, G.; Zhou, R.; Xu, S.; Zhu, Y.; Guo, W.; Chen, Y.C.; Chen, G. Two-stage competitive particle swarm optimization based timing-driven X-routing for IC design under smart manufacturing. *ACM Trans. Manag. Inf. Syst.* **2022**, *13*, 1–26. [CrossRef]
24. Kourepinis, V.; Iliopoulou, C.; Tassopoulos, I.X.; Aroniadi, C.; Beligiannis, G.N. An Improved Particle Swarm Optimization Algorithm for the Urban Transit Routing Problem. *Electronics* **2023**, *12*, 3358. [CrossRef]
25. Shaqarin, T.; Noack, B.R. A fast-converging particle swarm optimization through targeted, position-mutated, elitism (PSO-TPME). *Int. J. Comput. Intell. Syst.* **2023**, *16*, 6. [CrossRef]
26. Sun, L.; Song, X.; Chen, T. An improved convergence particle swarm optimization algorithm with random sampling of control parameters. *J. Control. Sci. Eng.* **2019**, *2019*, 7478498. [CrossRef]
27. Hsieh, F.S. Comparison of a hybrid firefly–Particle swarm optimization algorithm with six hybrid firefly–Differential evolution algorithms and an effective cost-saving allocation method for ridesharing recommendation systems. *Electronics* **2024**, *13*, 324. [CrossRef]

28. Vinay Kumar, S.; Rao, P.; Singh, M.K. Optimal floor planning in VLSI using improved adaptive particle swarm optimization. *Evol. Intell.* **2022**, *15*, 925–938. [CrossRef]
29. Khursheed, A.; Khare, K. Optimized buffer insertion using PSO technique for efficient interconnect designs. In *AIP Conference Proceedings*; AIP Publishing: Melville, NY, USA, 2023; Volume 2745.
30. Nath, P.; Dey, S.; Nath, S.; Shankar, A.; Sing, J.K.; Kumar Sarkar, S. VLSI Routing Optimization Using Hybrid PSO Based on Reinforcement Learning. In Proceedings of the IEEE VLSI Device Circuit and System, Kolkata, India, 26–27 February 2022; pp. 238–243.
31. Nath, S.; Shankar, A.; Sarkar, R.; Banerjee, S.; Sing, J.K.; Sarkar, S.K. Minimizing Wirelength with Bend Reduction using Gradient Descent PSO Hybrid in VLSI Global Routing. In Proceedings of the Devices for Integrated Circuit, Kalyani, India, 19–20 May 2021; pp. 401–405.
32. Nath, S.; Sing, J.K.; Sarkar, S.K. Wire length optimization of VLSI circuits using IWO algorithm and its hybrid. *Circuit World* **2024**, *50*, 205–216. [CrossRef]
33. Bansal, K.; Chaurasiya, R.K.; Chourasia, V. Solving the Trade-off between Power and Delay in High Speed Circuits using Multi Objective PSO. In Proceedings of the International Conference on Sustainable Computing and Smart Systems, Coimbatore, India, 14–16 June 2023; pp. 1139–1144.
34. Tripathi, J.N.; Junjariya, D.; Achar, R. Optimization of Decoupling Capacitors in VLSI Systems using Granularity Learning and Logistic Regression based PSO. In Proceedings of the IEEE/MTT-S International Microwave Symposium—IMS, San Diego, CA, USA, 11–16 June 2023; pp. 159–162.
35. Batterywala, S.; Shenoy, N.; Nicholls, W.; Zhou, H. Track assignment: A desirable intermediate step between global routing and detailed routing. In Proceedings of the International Conference on Computer-Aided Design, San Jose, CA, USA, 10–14 November 2002; pp. 59–66.
36. Gao, X.; Macchiarlo, L. Track routing optimizing timing and yield. In Proceedings of the Asia and South Pacific Design Automation Conference, Yokohama, Japan, 25–28 January 2011; pp. 627–632.
37. Cho, M.; Xiang, H.; Puri, R.; Pan, D.Z. Track routing and optimization for yield. *IEEE Trans.-Comput.-Aided Des. Integr. Circuits Syst.* **2008**, *27*, 872–882. [CrossRef]
38. Kay, R.; Rutenbar, R.A. Wire packing: A strong formulation of crosstalk-aware chip-level track/layer assignment with an efficient integer programming solution. In Proceedings of the International Symposium on Physical Design, San Diego, CA, USA, 9–12 April 2000; pp. 61–68.
39. Chang, Y.N.; Li, Y.L.; Lin, W.T.; Cheng, W.N. Non-slicing floorplanning-based crosstalk reduction on gridless track assignment for a gridless routing system with fast pseudo-tile extraction. In Proceedings of the International Symposium on Physical Design, Portland, OR, USA, 13–16 April 2008; pp. 134–141.
40. Zhao, Q.; Hu, J. Track assignment considering crosstalk-induced performance degradation. In Proceedings of the International Conference on Computer Design, Montreal, QC, Canada, 30 September–3 October 2012; pp. 506–507.
41. Li, Y.L.; Chang, Y.N.; Cheng, W.N. A gridless routing system with nonslicing floorplanning-based crosstalk reduction on gridless track assignment. *ACM Trans. Des. Autom. Electron. Syst.* **2011**, *16*, 1–25. [CrossRef]
42. Lai, B.T.; Li, T.H.; Chen, T.C. Native-conflict-avoiding track routing for double patterning technology. In Proceedings of the International System-on-Chip Conference, Niagara Falls, NY, USA, 12–14 September 2012; pp. 381–386.
43. Chen, X.; Fu, R.; Huang, J.; Cao, H.; Zhang, Z.; Ye, X.; Ho, T.Y.; Fan, D. JRouter: A Multi-Terminal Hierarchical Length-Matching Router under Planar Manhattan Routing Model for RSFQ Circuits. In Proceedings of the Great Lakes Symposium on VLSI, Knoxville, TN, USA, 5–7 June 2023; pp. 515–520.
44. Wong, M.P.; Liu, W.H.; Wang, T.C. Negotiation-based track assignment considering local nets. In Proceedings of the Asia and South Pacific Design Automation Conference, Macao, China, 25–28 January 2016; pp. 378–383.
45. Guo, W.; Chen, X.; Liu, G.; Chen, G. Track assignment algorithm based on hybrid discrete particle swarm optimization. *Pattern Recognit. Artif. Intell.* **2019**, *32*, 758–770. (In Chinese)
46. Liu, G.; Zhuang, Z.; Guo, W.; Wang, T.C. RDTA: An efficient routability-driven track assignment algorithm. In Proceedings of the Great Lakes symposium on VLSI, Tysons Corner, VA, USA, 9–11 May 2019; pp. 315–318.
47. Zhuang, Z.; Liu, G.; Ho, T.Y.; Yu, B.; Guo, W. TRADER: A practical track-assignment-based detailed router. In Proceedings of the Design, Automation & Test in Europe Conference & Exhibition, Antwerp, Belgium, 14–23 March 2022; pp. 766–771.
48. Cho, M.; Xiang, H.; Puri, R.; Pan, D.Z. TROY: Track router with yield-driven wire planning. In Proceedings of the Design Automation Conference, San Diego, CA, USA, 4–8 June 2007; pp. 55–58.
49. Ahrens, M.; Henke, D.; Rabenstein, S.; Vygen, J. Faster Goal-Oriented Shortest Path Search for Bulk and Incremental Detailed Routing. *Math. Program.* **2022**, *13265*, 15–28.
50. Jing, Y.; Yang, L.; Zhuang, Z.; Liu, G.; Huang, X.; Liu, W.H.; Wang, T.C. SPTA: A Scalable Parallel ILP-Based Track Assignment Algorithm with Two-Stage Partition. In Proceedings of the International Conference on Very Large Scale Integration, Patras, Greece, 3–5 October 2022; pp. 1–6.
51. Nie, S.R.; Chen, Y.T.; Chang, Y.W. Y-Architecture-Based Flip-Chip Routing with Dynamic Programming-Based Bend Minimization. In Proceedings of the ACM/IEEE Design Automation Conference, San Francisco, CA, USA, 10–14 July 2022; pp. 955–960.
52. Zhu, Z.; Huang, Z.; Chen, J.; Guo, L. Topology-aware bus routing in complex networks of very-large-scale integration with nonuniform track configurations and obstacles. *Complexity* **2021**, *2021*, 8843271. [CrossRef]

53. Chen, M.; Orailoglu, A. Deflecting crosstalk by routing reconsideration through refined signal correlation estimation. In Proceedings of the Great Lakes Symposium on VLSI, Boston Area, MA, USA, 10–12 May 2009; pp. 369–374.
54. Chen, J.; Liu, J.; Chen, G.; Zheng, D.; Young, E.F. MARCH: Maze routing under a concurrent and hierarchical scheme for buses. In Proceedings of the Design Automation Conference, Las Vegas, NV, USA 2–6 June 2019; pp. 1–6.
55. Guo, W.; Liu, G.; Chen, G.; Peng, S. A hybrid multi-objective PSO algorithm with local search strategy for VLSI partitioning. *Front. Comput. Sci.* **2014**, *8*, 203–216. [CrossRef]
56. Liu, W.H.; Kao, W.C.; Li, Y.L.; Chao, K.Y. NCTU-GR 2.0: Multithreaded collision-aware global routing with bounded-length maze routing. *IEEE Trans.-Comput.-Aided Des. Integr. Circuits Syst.* **2013**, *32*, 709–722.
57. Hsu, M.K.; Chen, Y.F.; Huang, C.C.; Chen, T.C.; Chang, Y.W. Routability-driven placement for hierarchical mixed-size circuit designs. In Proceedings of the Design Automation Conference, Austin, TX, USA, 29 May–7 June 2013; pp. 1–6.
58. Viswanathan, N.; Alpert, C.; Sze, C.; Li, Z.; Wei, Y. The DAC 2012 routability-driven placement contest and benchmark suite. In Proceedings of the Design Automation Conference, San Francisco, CA, USA, 3–7 June 2012; pp. 774–782.
59. Kim, M.C.; Hu, J.; Lee, D.J.; Markov, I.L. A SimPLR method for routability-driven placement. In Proceedings of the International Conference on Computer-Aided Design, San Jose, CA, USA, 7–10 November 2011; pp. 67–73.
60. He, X.; Huang, T.; Xiao, L.; Tian, H.; Cui, G.; Young, E.F. Ripple: An effective routability-driven placer by iterative cell movement. In Proceedings of the International Conference on Computer-Aided Design, San Jose, CA, USA, 7–10 November 2011; pp. 74–79.
61. Liu, G.; Jing, Y.; Zhou, R.; Chen, X.; Pan, J.S. SLDPSO-TA: Track Assignment Based on Social Learning Discrete Particle Swarm Optimization. 2022, *preprint*. [CrossRef]