BottleneckNet:一种面向大规模数字芯片的逻辑综合后阶段时序瓶颈预测的GNN模型

何广鹏[1],邸志雄[2,3*],邓雨姣[1],陈 旋[4],张泽涛[4],刘 洋[4]

(1. 西南交通大学信息科学与技术学院,四川成都 611730;2. 西南交通大学集成电路学院,四川成都 611730;
3. 西安电子科技大学集成电路学部,陕西西安 710126;4. 华为技术有限公司海思半导体业务部,广东深圳 518129)

摘 要: 在超大规模数字芯片(Very Large Scale Integration,VLSI)设计流程中,逻辑综合阶段是连接架构设计与物理实现的桥梁. 然而,逻辑综合工具分析得出的初始时序瓶颈路径与布局布线(Placement and Routing,P&R)完成后的真实时序瓶颈之间往往存在显著差异. 这种"时序不一致性"主要源于两个维度:首先,在综合阶段,由于物理布局信息缺失,电子设计自动化(Electronic Design Automation,EDA)工具通常采用线网负载模型估算互连线时延,难以捕捉深亚微米工艺下复杂的寄生参数效应;其次,现代芯片内部逻辑门种类繁多且拓扑连接高度复杂,这显著增加了在逻辑综合阶段进行准确静态时序分析(Static Timing Analysis,STA)的计算难度. 为了解决上述挑战,本文提出一种面向大规模数字芯片的时序瓶颈预测模型——BottleneckNet. 首先,在特征工程方面,针对大规模设计网表规模过大导致深度学习模型难以训练的问题,本文提出了一种基于寄存器子图(Register Sub-Graph,RSG)概念的网表特征提取方法. 该方法通过对网表拓扑进行结构化剪枝,仅保留寄存器间的关键组合逻辑信息. 该特征提取流程可以在$O(n)$的时间复杂度下完成,从而满足了工业级大规模网表的处理需求. 在模型架构方面,本文设计了一种基于图神经网络(Graph Neural Network,GNN)的双通道特征传播模型,该模型能够同时捕捉电路的全局特征与局部逻辑网络拓扑信息. 通过融合双通道特征,BottleneckNet能够实现对布局布线后时序瓶颈的精准感知. 本文基于多组开源大规模设计构建了完备的测试数据集,实验结果表明,本文提出的BottleneckNet模型展现出优异的综合性能. 在处理效率方面,该方法能够在分钟级时间内完成百万门级设计的特征提取与推理任务. 在预测精度方面,针对时序最差的5%~20%时序瓶颈路径,本文方法的预测准确率不仅显著优于轻量级梯度提升机(Light Gradient Boosting Machine,LightGBM)模型,且远高于工业界主流逻辑综合工具Synopsys公司的Design Compiler(DC)在综合阶段给出的计算结果. 该研究成果对于指导逻辑综合阶段后的时序优化、缩短芯片研发迭代周期具有重要的理论意义与工程应用价值.

关键词: 逻辑综合;布局布线;静态时序分析;寄存器子图;图神经网络;双通道特征;时序瓶颈

基金项目: 国家自然科学基金(No.62374138)

中图分类号: TN402 **文献标识码:** A **文章编号:** 0372-2112(2025)12-4518-09

电子学报URL:http://www.ejournal.org.cn **DOI:**10.12263/DZXB.20250798

BottleneckNet: A Graph Neural Network for Post-Synthesis Timing Bottleneck Prediction in Large-Scale Digital ICs

HE Guang-peng[1], DI Zhi-xiong[2,3*], DENG Yu-jiao[1], CHEN Xuan[4], ZHANG Ze-tao[4], LIU Yang[4]

(1. *School of Information Science and Technology*, *Southwest Jiaotong University*, *Chengdu*, *Sichuan* 611730, *China*;
2. *School of Integrated Circuits*, *Southwest Jiaotong University*, *Chengdu*, *Sichuan* 611730, *China*;
3. *Faculty of Integrated Circuits*, *Xidian University*, *Xi'an*, *Shaanxi* 710126, *China*;
4. *Hisilicon*, *Huawei Technologies Co.*, *Ltd.*, *Shenzhen*, *Guangdong* 518129, *China*)

Abstract: In the design flow of very large scale integration (VLSI), the logic synthesis stage serves as a bridge between architectural design and physical implementation. However, there is often a significant discrepancy between the initial timing bottleneck paths analyzed by logic synthesis tools and the actual timing bottlenecks after placement and routing (P&R). This "timing inconsistency" mainly originates from two dimensions: first, in the synthesis stage, due to the lack

收稿日期:2025-09-12;录用日期:2025-12-22;责任编辑:朱梅玉

*通讯作者:邸志雄

of physical layout information, electronic design automation (EDA) tools usually employ wire load models to estimate interconnect delays, which makes it difficult to capture complex parasitic effects under deep sub-micron processes; second, modern chips contain a wide variety of logic gates and highly complex topological connections, which significantly increases the computational difficulty of accurate static timing analysis (STA) during logic synthesis. To address these challenges, this paper proposes BottleneckNet, a timing bottleneck prediction model for large-scale digital chips. First, in terms of feature engineering, to address the issue where large-scale design netlists are too large for deep learning models to train, a netlist feature extraction method based on the concept of register sub-graph (RSG) is proposed. This method performs structural pruning on the netlist topology, retaining only the critical combinational logic information between registers. This feature extraction process can be completed with a time complexity of $O(n)$, thereby meeting the processing requirements for industrial large-scale netlists. In terms of model architecture, a dual-channel feature propagation model based on graph neural network (GNN) is designed. This model can simultaneously capture global features and local logic network topological information of the circuit. By fusing dual-channel features, BottleneckNet achieves accurate perception of post-P&R timing bottlenecks. A complete test dataset was constructed based on multiple sets of open-source large-scale designs. Simulation experiment results show that the proposed BottleneckNet model demonstrates excellent comprehensive performance. In terms of processing efficiency, the method can complete feature extraction and inference tasks for million-gate level designs within minutes. In terms of prediction accuracy, for the worst 5%~20% timing bottleneck paths, the prediction accuracy of the proposed method is not only significantly better than the light gradient boosting machine (LightGBM) model, but also much higher than the calculation results given by Synopsys design compiler (DC), a mainstream industrial logic synthesis tool, during the synthesis stage. The research results have important theoretical significance and engineering application value for guiding timing optimization after the logic synthesis stage and shortening the chip R&D cycle.

Key words: logic synthesis; placement and routing; static timing analysis; register sub-graph; graph neural networks; dual-channel features; timing bottleneck

Foundation Item(s): National Natural Science Foundation of China (No.62374138)

1 引言

时序收敛是专用集成电路(Application-Specific Integrated Circuit，ASIC)设计中的必要环节，而时序瓶颈是进行时序优化进而提升设计性能的关键切入点. 从逻辑综合到布局布线，工程师期望各个阶段的电子设计自动化(Electronic Design Automation，EDA)工具都能够提供高度准确的时序瓶颈分析结果，以便在后续阶段实施更有针对性的优化措施. 因此，提前预测时序瓶颈，对于提升整体时序质量具有重要意义. 然而，在逻辑综合阶段，由于缺少单元的物理位置信息以及精确的互连延时数据，综合工具所报告的时序瓶颈与布线完成后所揭示的真实时序瓶颈之间存在明显差异，这种偏差在大规模设计中尤为严重. 因此，在缺乏物理信息的情况下，在逻辑综合阶段获得与布线后高度一致的时序瓶颈分析结果仍然是一个重大挑战.

目前，大多数研究工作主要关注布局或布线阶段的延迟预测或时序瓶颈预测. 早期针对综合后时序预测或延迟计算的研究通常采用启发式的线网负载模型[1,2]来近似布局布线过程中可能引入的寄生效应. 近年来，机器学习方法逐渐成为准确预测物理验证指标的一种有效手段[3~8]. 例如，文献[9]利用支持向量机(Support Vector Machine，SVM)在布图规划之后预测布局布线阶段可能发生的时序失败；文献[10]则采用随机森林方法在布局阶段预测布线后的时序裕量. 在更近期的研究中，图神经网络(Graph Neural Network，GNN)因其能够从网表超图中提取拓扑结构信息而被广泛用于预测任务. 例如，文献[11]提出了一种基于时序引擎驱动的GNN模型，用于在后端布局阶段端到端预测数据到达时间和时序裕量. 然而，现有的基于机器学习的预测方法都需在布图规划和布局阶段产生的物理信息作为输入，因此无法解决在缺乏物理信息的大规模设计综合后阶段的时序瓶颈预测问题.

上述方法依赖于在布图规划或布局阶段产生的物理信息，需要设计首先完成这些阶段以便进行完整的特征提取. 此外，相关研究中使用的实验数据集很少涉及真正的大规模设计. 相比于小规模设计，在大规模设计上构建数据集并进行时序预测主要面临两个问题：(1)大规模设计的网表节点和超边数量显著增加，采用传统特征提取方式或直接基于网表构建图进行模型训练会带来极高的计算开销；(2)在实际工业流程中，由于内部逻辑单元和时序路径数量庞大，大规模设计在综合后阶段与物理实现阶段往往呈现出明显不同的时序特性. 此外，在大规模设计流程中，综合后阶段的物理设计工程师更关注由最差的时序路径引起的潜在时序违例，而非某一条具体路径的时序裕量或数据到达时间. 因此，在大规模设计中进行时序瓶颈预测不仅更具必要性，也更加具有挑战性.

为了解决上述问题,本文提出了一种面向大规模ASIC设计的基于GNN的综合后时序瓶颈预测模型. 该模型以快速估计从某个寄存器出发的所有时序路径中的最差的数据到达时间为目标,从而在早期提供对设计综合后阶段时序瓶颈的评估. 本文的主要贡献总结如下:

(1)提出了一种反向的图遍历方法,用于加速在大规模设计中提取寄存器子图(Register Sub-Graph,RSG). 与直接遍历方法,如广度优先搜索(Breadth-First Search,BFS)方法相比,该方法可以将时间复杂度由$O(n^2)$降至$O(n)$. 通过这种方法可以加速数据集的构建和模型训练.

(2)搭建了一种结合GNN与多层感知机(Multi-Layer Perceptron,MLP)的双通道特征传播模型. 在该模型中,Reg节点经过特征提取后嵌入了上下游的网络拓扑信息作为特征,之后Reg节点特征通过GraphSAGE(Graph Sample and Aggregate)进行聚合和传播,以学习局部特征信息. 同时,与整体网表规模相关的全局设计特征则通过线性层提取. 随后,这两个通道的输出被拼接并输入至最终的线性层进行最后的预测之后的输出. 该方法使得模型能够学习更为全面的设计信息,从而提升预测性能与泛化能力.

(3)本文基于三款开源且经过流片验证的RISC-V处理器(OpenC906、OpenC910和Xiangshan)、两个开源的大规模ASIC设计(3DES和swerv_wrapper)以及两个CNN加速器设计(icb1024和icb2048)构建数据集. 以Candence公司的布局布线工具Innovus生成的布线后时序瓶颈信息作为标签. 实验结果表明,在规模最大的设计C910上,对于前5%的最差时序瓶颈,本文所提出的预测模型BottleneckNet能够达到70%的预测准确率,显著优于主流商业逻辑综合EDA工具DC(Design Compiler)的评估准确率.

2 研究背景与动机

2.1 时序瓶颈的定义

在芯片设计中,时序瓶颈通常指的是可能导致时序违例的关键路径. 在设计流程的不同阶段,物理实现工具可以生成时序报告,用于评估设计中当前存在的时序瓶颈. 本文所预测的时序瓶颈是通过在布线完成后,利用Innovus工具生成的时序报告提取得到的,其提取过程由Tcl脚本实现. 其详细定义如下:

对于网表中的所有起点(startpoints),本文考虑与每个起点相关的时序路径(TimingPaths). 对于一个给定的起点s_i,其最差的到达时间t_i^{worst}定义为

$$t_i^{\text{worst}} = \max_{p_{ij} \in P_i} t_{ij} \tag{1}$$

其中,P_i表示所有以s_i为起点的时序路径的集合,而t_{ij}表示时序路径p_{ij}的到达时间. 由此本文可以得到整个设计的最差到达时间的集合T_{worst}如式(2)所示:

$$T_{\text{worst}} = \left\{ t_i^{\text{worst}} | s_i \in S \right\} \tag{2}$$

其中,S是网表中所有起始节点集合,而T_{worst}表示设计中的所有时序瓶颈.

T_{worst}集合中的时序瓶颈表示的是设计中最差的一些到达时间的集合. 其中"最差"可以是前5%或10%等. 在本研究中,本文着重关注5%~20%的时序瓶颈.

2.2 GraphSAGE和LightGBM概述

GraphSAGE是图神经网络领域的重要进展,其相关工作最早由William等人[12]提出. 其核心思想不是直接学习每个节点的嵌入表示,而是通过学习一个聚合函数,从节点的局部邻居中进行采样并聚合特征信息,从而生成该节点的嵌入表示. 总体而言,GraphSAGE在大规模的图的网络训练中具有更高的适用性.

GraphSAGE的采样和聚合操作可分为以下几个步骤:

(1)邻居采样. 对于每个节点,从其领域中采样固定数量的邻居.

(2)信息聚合. 利用聚合函数对采样邻居的特征信息进行聚合,得到该节点的聚合表示.

(3)表示更新. 将节点的原始表示与聚合后的表示进行结合,并施加非线性变换,得到更新后的节点表示.

(4)迭代更新. 在多层聚合结构中重复上述步骤,使得节点表示能够逐步融合来自更远邻居的信息.

在最后的实验部分本文还使用了另一个高效的传统的机器学习方法——轻量级梯度提升机(Light Gradient Boosting Machine,LightGBM)模型[13]与本文采用的GNN模型进行对比,LightGBM基于梯度提升决策树,与GNN不同,无法捕捉节点之间的依赖关系. 本文进行对比实验主要是为了验证GNN是否更加适合处理网表相关的预测任务.

3 本文工作

3.1 基于寄存器子图的特征表示

在大规模设计中用图神经网络进行时序瓶颈预测时,特征工程的构建主要面临以下两个挑战:

(1)在大规模设计的网表超图中,节点数量庞大且关系复杂,若直接将原始网表作为模型输入,将导致模型训练过程极为困难,并在特征工程环节显著增加时间开销.

(2)在综合后阶段,如果仅依赖网表数据进行特征提取,就无法获取电路单元的实际布局坐标. 由于缺少物理布局参数,这必然会增加预测的困难程度.

为了解决第一个问题,本文提出了一种新的概念——RSG. 一个RSG包含了在原始网表中从特定寄存器节

点s1出发的所有时序路径信息(包括所有组合逻辑单元c1~c3),以及从s2出发所包含的c4~c7. 在3.2节中,本文将详细说明如何从RSG的下游向上游传播信息,以及如何将特征聚合回s1. 最终得到的特征图仅包含诸如s1这样的起始点作为节点,例如e1~e4作为s1的对应的时序路径的终点,也可能在其他时序路径中作为起始点. 它们之间的拓扑信息已经嵌入到了每个对应的节点上,它们之间的连接边取直连的方式. 也就是说只要在原始网表中可以从s1到达e1,在处理后的图中就保持一条边的直接连接. 通过将原始网表抽象成一个个RSG,本文能够在尽量保留几乎全部原始网表的拓扑信息的同时,有效缩减用于模型训练的图的规模,从而加速训练过程.

为了解决第二个问题,本文构建了一条完整的特征工程流程,以实现更为准确的预测. 该流程通过从网表中提取全局特征与从RSG中提取局部节点特征来实现. 详细的特征信息如表1所示. 全局特征包括设计整体规模相关的信息,例如电路中的单元总数以及网络边的数量. 从宏观角度看,这些特征能够反映不同设计规模在布局布线阶段对实际网络延迟的影响.

表1　局部特征和全局特征汇总

特征类型	特征名称	向量大小
局部	子图大小	1
	邻居节点数	1
	图密度	1
	平均度	1
	最长路径长度	1
	终点数	1
	路径单元面积	1
	单元层级	100
	路径单元类型	16
全局	设计规模	1
	IO端口数	1
	buffers/单元数	1
	总边数	1
总特征维度		127

局部特征主要指由RSG内所有单元构成的网络的拓扑信息. 本文不仅提取了常见的图论指标(如图密度、平均度数),还着重关注那些实际形成时序路径的组合逻辑链. 对于设计中的起始点,其对应的时序路径在经过物理实现过程后,受到的不仅是组合逻辑链本身的影响,还可能受到周围其他单元在布局阶段引入的潜在干扰. 为此,本文提出了两个自定义特征,以捕捉这两类因素对时序路径的影响.

(1)路径单元类型特征. 表示RSG内最长时序路径中不同类型单元的数量. 该特征反映了最长组合逻辑链的长度、单元类型和数量的变化对面积和单元延时的影响.

(2)单元层级特征. 统计设计网表中所有组合逻辑单元在网表中所处的深度,记录范围为1~100的节点深度. 本文假设,网表中层级不一样的单元在布局布线过程中受到的影响程度也不同.

特征需要进行降维处理,如式(3)所示:

$$\boldsymbol{z}=\ln \boldsymbol{X}\boldsymbol{w} \tag{3}$$

其中,$\boldsymbol{X}\in\mathbf{R}^{n\times m}$,$\boldsymbol{w}\in\mathbf{R}^{m}$,$n$为样例数,$m=16$或100.

总的来说,通过同时从设计网表中提取全局特征以及基于RSG的局部特征,模型能够学习物理实现过程中不同设计的网表对时序路径的影响,从而实现对时序瓶颈的准确预测.

3.2　特征嵌入流程

原始网表中包含了所有需要的信息,但在大规模设计中,网表中的单元数量可能达到数十万甚至上百万. 若直接采用遍历方法来提取寄存器子图特征,在最坏情况下其时间开销会呈二次增长,时间复杂度为$O\left(n^2\right)$. 如图1所示,两条时序路径{s2-c4-c6-c7-e3}和{s2-c5-c6-c7-e3}都会重复遍历到单元c6与c7,而这种重复遍历是没有必要的.

为了解决这一问题,本文采用图1中的方法,避免组合逻辑单元的重复遍历. 具体步骤可以分为过滤、拓扑排序以及基于反向广度优先搜索(Reverse Breadth-First Search,RBFS)进行回溯.

(1)过滤. 在原始网表中,寄存器节点始终存在上下游的层次关系,但其中可能包含许多环路和孤立单元. 第一步是从原始网表中移除所有Reg节点,从而得到多个相互独立的有向无环子图(Directed Acyclic Subgraph,DAG). 这些子图通常对应于一组寄存器与另一组寄存器之前的连接网络,相较于原始大规模网表,它们更易于遍历和处理.

(2)拓扑排序. 对由组合逻辑单元构成的有向无环图反转边的方向后,需要对每个单元进行拓扑排序. 排序的目的是建立单的遍历顺序. 具体规则为:对于一个有向无环图G,对其所有的顶点进行排序,生成一个线性序列T,要求每个节点只出现一次,且对于任意边$(u,v)\in E(G)$,顶点u必须出在v之前. 基于拓扑排序的结果,为所有组合逻辑单元分配层级. 从1开始编号,终点被视为最低层(level 0);起点节点视为最高层,记为顶层.

(3)RBFS. 通过拓扑排序得到序列T之后,可以按照顺序对所有单元进行反向回溯遍历. 通常从靠近终点的单元开始,将其作为当前节点,并使用BFS遍历其上游节点. 在遍历过程中,会记录下游网络的拓扑信

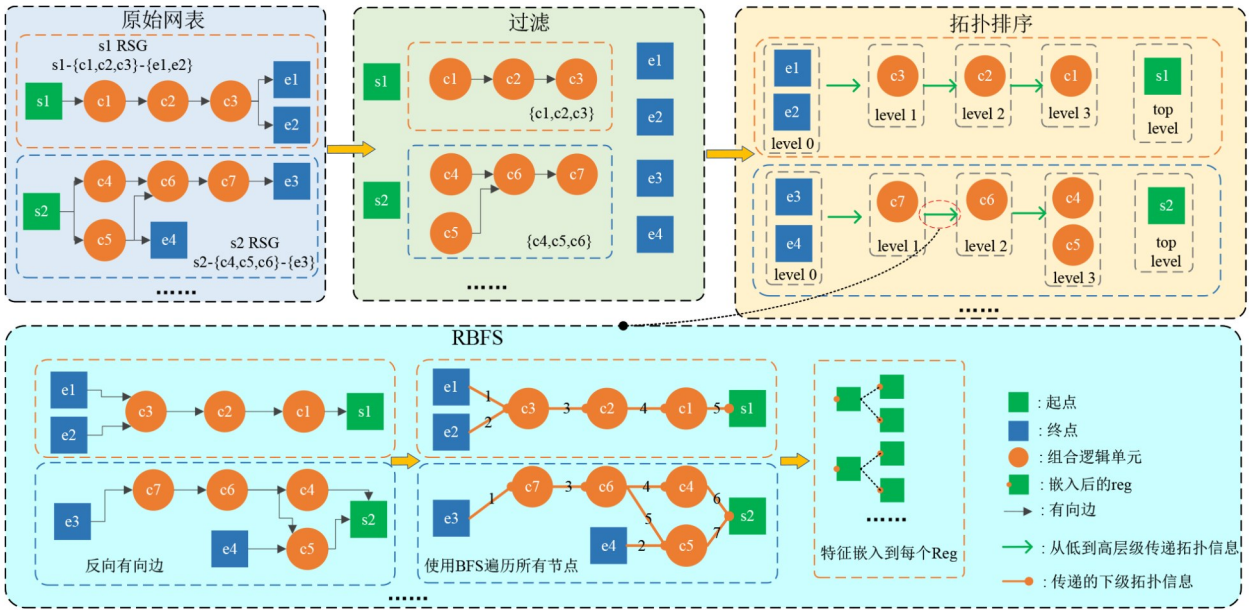


图1 基于RSG的特征嵌入流程

息,例如单元数量、路径长度等,并将这些信息向上传递至原始图的前驱节点. 当某个节点的所有上游节点均已遍历完成后,删除该节点以避免重复遍历. 该过程将不断迭代,直至回溯到起始节点完成特征嵌入.

总的来说,通过上述流程,本文能够加速在大规模设计网表中对RSG的局部拓扑信息的提取. 相比于最坏情况下正向BFS遍历的$O(n^2)$,本文的方法仅具有$O(n)$的时间复杂度,并且只引入了一些空间复杂度的增加.

3.3 双通道特征传播模型

基于GraphSAGE的双通道特征传播模型如图2所示,本文将特征传播通道分为两部分:局部特征通道和全局特征通道.

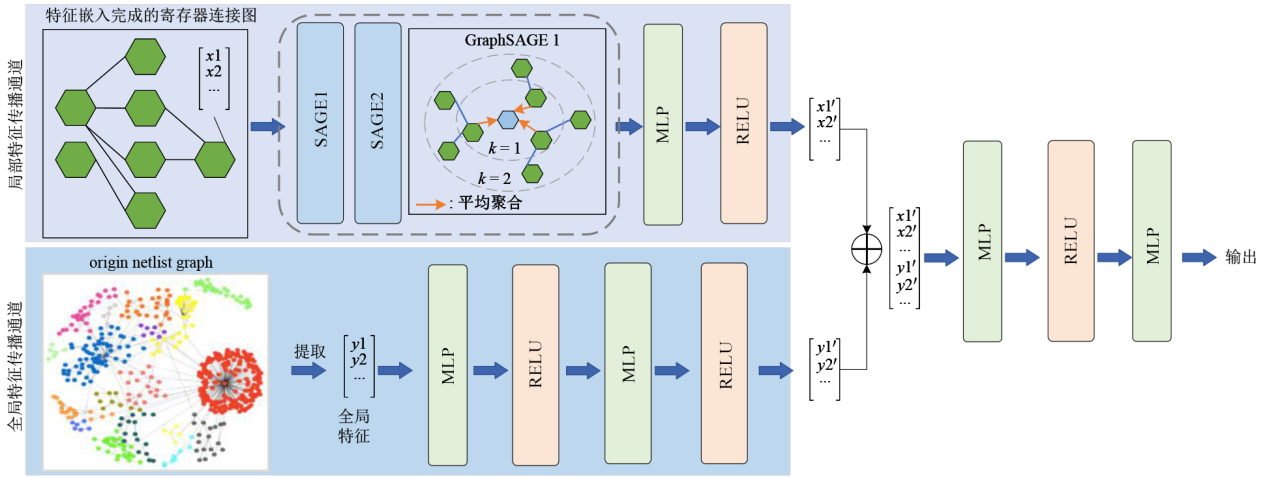


图2 双通道特征传播模型框架

在局部特征通道,嵌入好的节点特征会通过两层GraphSAGE网络对节点特征进行采样和聚合. 而在全局特征通道中,本文首先对原始网表进行分析,提取全局特征,并将其输入到由两个MLP组成的线性层中进行传播. 最后,将局部特征通道的输出与全局特征通道的结果进行拼接,并输入到最终的线性层中输出预测结果.

正如在前文中讨论的,本文认为仅依赖RSG中的局部信息进行特征提取来预测最终的时序瓶颈是不够的. 大规模设计在物理实现过程中会受到更严格的约束,例如约束脚本中关于芯片面积和时序一致性的限制. 这些约束会影响局部时序路径在布局布线阶段的实际实现,从宏观层面作用于物理实现工具.

因此,本文将设计网表中的宏观信息抽象为全局

特征,并构建一个独立于局部特征的特征传播通道. 这样,输出中既包含从RSG中提取的有用局部信息,又包含来自不同设计的整体规模信息,使得模型能够同时利用宏观和局部特征进行学习,从而提升预测的准确性和泛化能力.

4　实验结果

本文使用C++和Python完成代码编写,以Pytorch以及深度图谱库(Deep Graph Library, DGL)[14]作为框架. 本文服务器的硬件是i7-13700k@2.2Ghz和Nvidia RTX4090.

4.1　测试数据集

本文采用的数据集信息如表2所示,包含了一些开源的设计,其中包含三个开源RISC-V处理器:OpenC906、OpenC910[15]和香山处理器[16]. 从开源OpenLane[17]项目中选取了两个大规模ASIC设计:3DES和swerv_wrapper. 而icb1024和icb2048是两款来自工业界的CNN硬件加速器.

表2　不同模型的最终预测结果性能指标对比

Model	Metric	3DES	swerv_wrapper	XiangShan	OpenE906	OpenC906	icb1024	icb2048	OpenC910
DC	Top-5%	56.29%	71.05%	71.24%	68.94%	44.11%	86.87%	65.78%	5.00%
	Top-10%	53.43%	80.70%	72.72%	80.07%	55.70%	93.11%	67.42%	9.90%
	Top-20%	71.85%	86.59%	95.70%	86.25%	80.27%	96.70%	68.24%	20.12%
	RMSE	0.087	0.89	0.13	0.97	1.10	0.11	0.30	1.43
	R^2	0.86	0.58	0.51	0.44	0.38	0.91	0.34	−0.623
	Kendall	0.82	0.83	0.75	0.71	0.70	0.91	0.42	0
LightGBM	Top-5%	49.11%	61.28%	70.83%	71.31%	62.95%	87.38%	67.34%	64.37%
	Top-10%	60.57%	75.96%	74.13%	74.56%	76.31%	94.28%	68.00%	61.84%
	Top-20%	76.65%	93.31%	95.56%	80.43%	77.25%	96.61%	68.53%	76.88%
	RMSE	0.018	0.66	0.013	0.72	0.57	0.074	0.20	0.612
	R^2	0.99	0.80	0.99	0.65	0.83	0.96	0.72	0.714
	Kendall	0.83	0.75	0.82	0.77	0.79	0.89	0.71	0.64
本文工作	Top-5%	60.64%	73.98%	79.66%	71.60%	71.71%	92.66%	73.25%	65.98%
	Top-10%	68.19%	89.91%	67.52%	71.76%	80.18%	96.15%	71.16%	69.94%
	Top-20%	79.23%	90.10%	99.74%	86.50%	86.68%	97.40%	83.17%	80.13%
	RMSE	0.016	0.47	0.012	0.76	0.52	0.04	0.15	0.57
	R^2	0.99	0.79	0.99	0.62	0.85	0.99	0.84	0.75
	Kendall	0.85	0.88	0.87	0.85	0.83	0.91	0.75	0.72

对于所有设计,本文基于28 nm工艺库并使用DC进行逻辑综合,通过Innovus完成物理实现流程直至布线完成,以生成完整的时序报告.

4.2　性能评估指标

本文采用Kendall[18]系数来评估模型预测结果与Innovus工具在布线完成后提供的时序瓶颈排序之间的相关性. 如表2所示,在评估模型性能时,本文不仅关注数据拟合指标,如RMSE和R^2,还强调模型预测的时序瓶颈结果与实际值之间的排序相关系数Kendall. Kendall越高,说明与布线后的真实时序瓶颈重合度越高,越能够帮助设计者快速识别最差时序路径并进行优化. Kendall系数用于衡量两个有序序列之间的排序相关性,其计算方法为:先定义Top$-k$%指标. 对于每个模型,本文取前k%的时序瓶颈T_{worst},并计算其与Innovus最终结果的前k%的交集.

T_{worst}详细定义在2.1节. 本文用$T_{\text{worst}}^{\text{model}}$、$T_{\text{worst}}^{\text{DC}}$和$T_{\text{worst}}^{\text{innovus}}$来分别表示本文的模型、DC以及innovus输出的时序瓶颈. 除此之外,用Top$-k$%, $k \in 5, 10, 20$来表示对于最差的5%、10%、20%的时序瓶颈的命中率.

Top$-k$%的具体计算方法如式(4)、式(5)所示:

$$\text{Top}-k\%_{\text{model}} = \frac{\left| T_{\text{Top}-k\%}^{\text{model}} \cap T_{\text{Top}-k\%}^{\text{innovus}} \right|}{\left| T_{\text{Top}-k\%}^{\text{innovus}} \right|} \tag{4}$$

$$\text{Top}-k\%_{\text{DC}} = \frac{\left| T_{\text{Top}-k\%}^{\text{DC}} \cap T_{\text{Top}-k\%}^{\text{innovus}} \right|}{\left| T_{\text{Top}-k\%}^{\text{innovus}} \right|} \tag{5}$$

使得$x=(x_1, x_2, \cdots, x_n)$, $y=(y_1, y_2, \cdots, y_n)$是两个长度相同的有序序列,对于一对数据点$(x_i, y_i)$和$(x_j, y_j)$,若$x_i \geqslant x_j$且$y_i \geqslant y_j$,则认为它们是一个一致对,否则为不一致对.

Kendall系数τ的计算公式如式(6)所示：

$$\tau=\frac{N_c-N_d}{0.5n(n-1)} \tag{6}$$

其中，N_c表示一致对的数量，N_d表示不一致对的数量.

在实际工程中，物理设计工程师通常会聚焦于最关键的时序瓶颈. 为了体现这一点，本文采用模型在最差5%~20%时序路径上的预测准确率作为专门的评估指标. 这一指标能够快速定位最严重的违例，以便进行针对性优化. 需要指出的是，5%和20%的阈值是工业实践中常用的经验值. 本文提出的方法具有灵活性，可以根据具体设计需求，应用于任意Top－k%范围.

4.3 实验结果分析

鉴于每个设计都是规模较大且彼此之间差异明显，本文采用留一法(leave-one-design-out)的交叉验证方案. 具体地说，在每一组实验中，将某个设计留作测试集，而在其余所有设计上训练模型，这样能够评估模型在不同网表结构上的泛化性能.

表3详细展示了所提方法在数据集构建和模型训练阶段的效率优势. 在特征提取阶段，对于不同规模，具有不同内部复杂度的设计. 本文的特征提取方法都能够实现加速的效果. 模型训练阶段，通过使用RSG替代全网表(Full-Graph). 本文在保持关键时序信息的同时，极大地缩减了图的规模，从而达到加速模型训练的效果.

表3 设计规模信息及特征提取和模型训练加速效果

设计	设计规模信息			特征提取时间/s			模型训练时间/s		
	Cells	Nets	Registers	BFS	RBFS	SpeedUp	Full-Graph	RSG	SpeedUp
3des	51 272	51 506	8 808	10.38	1.02	10.23×	12 378.42	243.58	50.82×
swerv_wrapper	83 672	86 596	13 986	37.44	4.68	8.00×			
XiangShan	89 970	104 762	16 812	35.40	1.62	21.85×			
OpenE906	106 350	108 161	12 080	41.85	8.04	5.21×			
OpenC906	228 840	233 384	29 488	214.70	51.61	4.16×			
icb1024	283 226	327 264	65 854	664.81	7.78	84.45×			
icb2048	559 638	646 183	129 625	3 173.14	20.02	158.50×			
OpenC910	2 199 080	2 221 913	242 706	12 446.50	236.11	52.71×			

最终的预测结果如表2所示. 从数据中可以观察到，在这些大规模设计中，本文的模型在Top－k%时序瓶颈上的命中率始终优于DC和LightGBM.

图3和图4展示了本文的模型与DC的预测结果的对比. 可以观察到，对于C906，DC在综合后阶段生成的时序报告仍有一定的参考价值. 然而，对于更大规模的设计C910，DC的时序报告几乎不再具有实际指导作用. 主要原因在于，大规模设计受到更多约束，并具有更高的网标复杂性，基于传统方法的STA工具在缺少物理信息的综合后阶段难以有效评估设计的时序状况. 相比之下，基于机器学习的方法(如LightGBM以及本文提出的BottleneckNet)在从较小规模设计过渡到大规模设计时，仍能保持相对较高的预测精度.

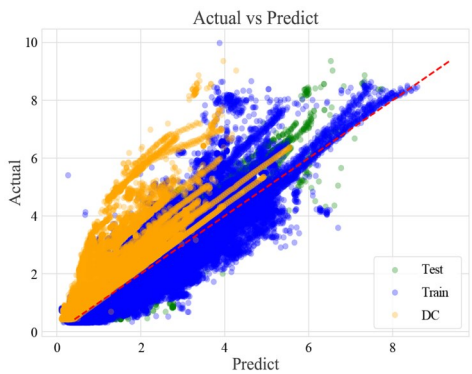


图3 C906预测效果图

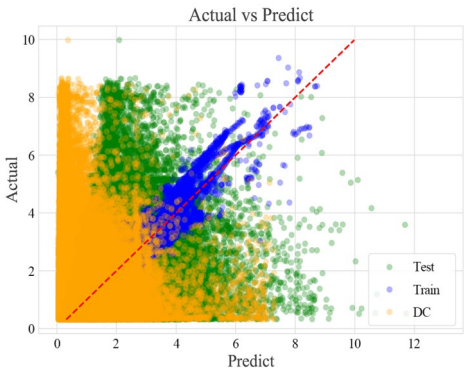


图4 C910预测效果图

表4展示的是消融实验的最终数据结果，为了验证本文模型的双通道的有效性，本文搭建了仅局部特征通道模型的对比实验. 从表4的结果可以看到，仅使用单通道进行训练时，模型最终预测的数据拟合效果以及对Top－k%的时序瓶颈的命中率都要差于双通道的预测结果. 这显著证明了全局特征的有效性. 当有全局特征参与模型训练时，模型能够学习到不同设计之间的显著差异，从而提升模型的泛化性.

表4　仅使用局部特征通道与双通道进行对比

Model	Metric	3DES	swerv_wrapper	XiangShan	OpenE906	OpenC906	icb1024	icb2048	OpenC910
仅局部特征通道	Top-5%	38.39%	69.54%	46.84%	57.38%	51.24%	79.67%	48.87%	32.09%
	Top-10%	48.81%	71.56%	57.26%	71.76%	73.62%	90.43%	61.19%	39.09%
	Top-20%	59.78%	90.10%	75.60%	80.84%	83.39%	95.07%	81.27%	52.83%
	RMSE	0.11	1.38	0.15	0.84	0.52	0.096	0.21	1.19
	R^2	0.77	0.08	0.43	0.51	0.21	0.95	0.73	0.08
	Kendall	0.63	0.81	0.41	0.73	0.77	0.70	0.64	0.17
双通道	Top-5%	60.64%	73.98%	79.66%	71.60%	71.71%	92.66%	73.25%	65.98%
	Top-10%	68.19%	89.91%	67.52%	72.27%	80.18%	96.15%	71.16%	69.94%
	Top-20%	79.23%	91.70%	99.74%	86.50%	86.68%	97.40%	83.17%	80.13%
	RMSE	0.016	0.47	0.012	0.76	0.48	0.04	0.15	0.57
	R^2	0.99	0.79	0.99	0.62	0.85	0.99	0.84	0.75
	Kendall	0.85	0.88	0.87	0.85	0.83	0.91	0.75	0.72

5　结论

实验结果表明，在多个大规模设计上的测试中，本文提出的模型在综合后阶段预测布线后时序瓶颈的性能优于DC和LightGBM. 进一步地，在大规模设计C910上的实验结果显示，针对此类超大设计的时序分析与预测仍然面临挑战，其表现为Top－k%的指标均出现显著下降. 尽管如此，与DC的STA相比，LightGBM与本文提出的模型均展现出较强的泛化能力，其预测精度在从小规模设计扩展到超大规模设计时并未出现显著衰减. 这一现象表明，基于机器学习的方法在超大规模设计的早期验证和预测中具备更可靠的应用潜力.

参考文献

[1] ALPERT C J, HU J, SAPATNEKAR S S, et al. Accurate estimation of global buffer delay within a floorplan[C]//IEEE/ACM International Conference on Computer Aided Design, 2004. ICCAD-2004. Piscataway: IEEE, 2005: 706-711.

[2] VUJKOVIC M, WADKINS D, SWARTZ B, et al. Efficient timing closure without timing driven placement and routing[C]//Proceedings of 41st Design Automation Conference, 2004. Piscataway: IEEE, 2005: 268-273.

[3] LIANG R J, XIE Z Y, JUNG J, et al. Routing-free crosstalk prediction[C]//Proceedings of the 39th International Conference on Computer-Aided Design. New York: ACM, 2020: 1-9.

[4] LIU M J, TURNER W J, KOKAI G F, et al. Parasitic-aware analog circuit sizing with graph neural networks and Bayesian optimization[C]//2021 Design, Automation & Test in Europe Conference & Exhibition. Piscataway: IEEE, 2021: 1372-1377.

[5] LU Y C, KIRAN PENTAPATI S S, ZHU L J, et al. TP-GNN: A graph neural network framework for tier partitioning in monolithic 3D ICs[C]//2020 57th ACM/IEEE Design Automation Conference. Piscataway: IEEE, 2020: 1-6.

[6] MIRHOSEINI A, GOLDIE A, YAZGAN M, et al. A graph placement methodology for fast chip design[J]. Nature, 2021, 594(7862): 207-212.

[7] REN H X, KOKAI G F, TURNER W J, et al. ParaGraph: Layout parasitics and device parameter prediction using graph neural networks[C]//2020 57th ACM/IEEE Design Automation Conference. Piscataway: IEEE, 2020: 1-6.

[8] 蔡志匡, 赵泽宇, 杨涵, 等. 基于机器学习的高效率集成电路DFT技术研究[J]. 电子学报, 2023, 51(12): 3473-3482.
CAI Z K, ZHAO Z Y, YANG H, et al. Research on high-efficiency integrated circuit DFT technology based on machine learning[J]. Acta Electronica Sinica, 2023, 51(12): 3473-3482. (in Chinese)

[9] CHAN W J, CHUNG K Y, KAHNG A B, et al. Learning-based prediction of embedded memory timing failures during initial floorplan design[C]//2016 21st Asia and South Pacific Design Automation Conference. Piscataway: IEEE, 2016: 178-185.

[10] BARBOZA E C, SHUKLA N, CHEN Y R, et al. Machine learning-based pre-routing timing prediction with reduced pessimism[C]//Proceedings of the 56th Annual Design Automation Conference 2019. New York: ACM, 2019: 1-6.

[11] GUO Z Z, LIU M J, GU J Q, et al. A timing engine inspired graph neural network model for pre-routing slack prediction[C]//Proceedings of the 59th ACM/IEEE Design Automation Conference. New York: ACM, 2022: 1207-1212.

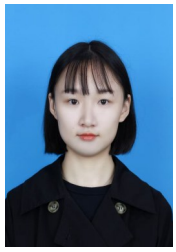
[12] HAMILTON W L, YING R, LESKOVEC J. Inductive representation learning on large graphs[EB/OL]. (2018-09-10)[2025-10-10]. https://arxiv.org/abs/1706.02216.
[13] KE G L, MENG Q, FINLEY T, et al. LightGBM: A highly efficient gradient boosting decision tree[C]//Neural Information Processing Systems. New York: Curran Associates Inc., 2017: 3149-3157.
[14] WANG M J, ZHENG D, YE Z H, et al. Deep graph library: A graph-centric, highly-performant package for graph neural networks[EB/OL]. (2020-08-25) [2025-12-30]. https://arXiv.org/abs/1909.01315.
[15] Semiconductor T-HEAD. OpenC906 and OpenC910: Open-source 64-bit RISC-V processor cores[EB/OL]. (2021-04-15) [2025-12-30]. https://github.com/XUANTIE-RV.
[16] XU Y N, YU Z H, TANG D, et al. Towards developing high performance RISC-V processors using agile methodology[C]//2022 55th IEEE/ACM International Symposium on Microarchitecture. Piscataway: IEEE, 2022: 1178-1199.
[17] SHALAN M, EDWARDS T. Building OpenLANE: A 130nm OpenROAD-based tapeout- proven flow: Invited paper[C]//Proceedings of the 39th International Conference on Computer-Aided Design. New York: ACM, 2020: 1-6.
[18] KENDALL M G. A new measure of rank correlation[J]. Biometrika, 1938, 30(1/2): 81-93.

作者简介

何广鹏 男,2001年8月出生于安徽省蚌埠市. 现为西南交通大学信息科学与技术学院硕士研究生. 主要研究方向为数字芯片物理设计算法.
E-mail: h13955244547@163.com

邸志雄 男,1984年10月出生于山西省忻州市. 现为西南交通大学信息科学与技术学院副教授. 主要研究方向为数字芯片物理设计算法和超大规模集成电路设计方法学. 中国电子学会会员编号:E190185357M.
E-mail: dizhixiong2@126.com

邓雨姣 女,1999年7月出生于四川省达州市. 现为西南交通大学信息科学与技术学院硕士研究生. 主要研究方向为数字芯片物理设计算法.
E-mail: yjiaodeng@163.com



陈 旋 女,1999年8月出生于四川省成都市. 现为华为技术有限公司海思半导体业务部工程师. 主要研究方向为物理后端设计方法学.
E-mail: chenxuan4l@huawei.com

张泽涛 男,1994年11月出生于四川省成都市. 现为华为技术有限公司海思半导体业务部工程师. 主要研究方向为物理后端设计方法学.
E-mail: zhangzetao3@huawei.com

刘 洋 男,1992年6月出生于四川省成都市. 现为华为技术有限公司海思半导体业务部工程师. 主要研究方向为物理后端设计方法学.
E-mail: liuyang169@huawei.com