

1. Record Nr.	UNINA9910637722203321
Titolo	Machine learning applications in electronic design automation // edited by Haoxing Ren, Jiang Hu
Pubbl/distr/stampa	Cham, Switzerland : , : Springer, , [2022] ©2022
ISBN	3-031-13074-X
Descrizione fisica	1 online resource (585 pages)
Disciplina	929.374
Soggetti	Engineering Disseny de circuits electrònics Automatització Aprenentatge automàtic Aplicacions industrials Llibres electrònics
Lingua di pubblicazione	Inglese
Formato	Materiale a stampa
Livello bibliografico	Monografia
Nota di bibliografia	Includes bibliographical references and index.
Nota di contenuto	Intro -- Preface -- Contents -- About the Editors -- Part I Machine Learning-Based Design Prediction Techniques -- 1 ML for Design QoR Prediction -- 1.1 Introduction -- 1.2 Challenges of Design QoR Prediction -- 1.2.1 Limited Number of Samples -- 1.2.2 Chaotic Behaviors of EDA Tools -- 1.2.3 Actionable Predictions -- 1.2.4 Infrastructure Needs -- 1.2.5 The Bar for Design QoR Prediction -- 1.3 ML Techniques in QoR Prediction -- 1.3.1 Graph Neural Networks -- 1.3.2 Long Short-Term Memory (LSTM) Networks -- 1.3.3 Reinforcement Learning -- 1.3.4 Other Models -- 1.4 Timing Estimation -- 1.4.1 Problem Formulation -- 1.4.2 Estimation Flow -- 1.4.3 Feature Engineering -- 1.4.4 Machine Learning Engines -- 1.5 Design Space Exploration -- 1.5.1 Problem Formulation -- 1.5.2 Estimation Flow -- 1.5.3 Feature Engineering -- 1.5.4 Machine Learning Engines -- 1.6 Summary -- References -- 2 Deep Learning for Routability -- 2.1 Introduction -- 2.2 Background on DL for Routability -- 2.2.1 Routability Prediction Background -- 2.2.1.1 Design Rule Checking (DRC) Violations -- 2.2.1.2 Routing Congestion

and Pin Accessibility -- 2.2.1.3 Relevant Physical Design Steps -- 2.2.1.4 Routability Prediction -- 2.2.2 DL Techniques in Routability Prediction -- 2.2.2.1 CNN Methods -- 2.2.2.2 FCN Methods -- 2.2.2.3 GAN Methods -- 2.2.2.4 NAS Methods -- 2.2.3 Why DL for Routability -- 2.3 DL for Routability Prediction Methodologies -- 2.3.1 Data Preparation and Augmentation -- 2.3.2 Feature Engineering -- 2.3.2.1 Blockage -- 2.3.2.2 Wire Density -- 2.3.2.3 Routing Congestion -- 2.3.2.4 Pin Accessibility -- 2.3.2.5 Routability Label -- 2.3.3 DL Model Architecture Design -- 2.3.3.1 Common Operators and Connections -- 2.3.3.2 Case Study: RouteNet 2:xie2018routenet -- 2.3.3.3 Case Study: PROS 2:chen2020pros -- 2.3.3.4 Case Study: J-Net 2:liang2020drc. 2.3.3.5 Case Study: Painting 2:yu2019painting -- 2.3.3.6 Case Study: Automated Model Development 2:chang2021auto -- 2.3.4 DL Model Training and Inference -- 2.4 DL for Routability Deployment -- 2.4.1 Direct Feedback to Engineers -- 2.4.2 Macro Location Optimization -- 2.4.3 White Space-Driven Model-Guided Detailed Placement -- 2.4.4 Pin Accessibility-Driven Model-Guided Detailed Placement -- 2.4.5 Integration in Routing Flow -- 2.4.6 Explicit Routability Optimization During Global Placement -- 2.4.7 Visualization of Routing Utilization -- 2.4.8 Optimization with Reinforcement Learning (RL) -- 2.5 Summary -- References -- 3 Net-Based Machine Learning-Aided Approaches for Timing and Crosstalk Prediction -- 3.1 Introduction -- 3.2 Backgrounds on Machine Learning-Aided Timing and Crosstalk Estimation -- 3.2.1 Timing Prediction Background -- 3.2.2 Crosstalk Prediction Background -- 3.2.3 Relevant Design Steps -- 3.2.4 ML Techniques in Net-Based Prediction -- 3.2.5 Why ML for Timing and Crosstalk Prediction -- 3.3 Preplacement Net Length and Timing Prediction -- 3.3.1 Problem Formulation -- 3.3.2 Prediction Flow -- 3.3.3 Feature Engineering -- 3.3.3.1 Features for Net Length Prediction -- 3.3.3.2 Features for Timing Prediction -- 3.3.4 Machine Learning Engines -- 3.3.4.1 Machine Learning Engine for Net Length Prediction -- 3.3.4.2 Machine Learning Engine for Preplacement Timing Prediction -- 3.4 Pre-Routing Timing Prediction -- 3.4.1 Problem Formulation -- 3.4.2 Prediction Flow -- 3.4.3 Feature Engineering -- 3.4.4 Machine Learning Engines -- 3.5 Pre-Routing Crosstalk Prediction -- 3.5.1 Problem Formulation -- 3.5.2 Prediction Flow -- 3.5.3 Feature Engineering -- 3.5.3.1 Probabilistic Congestion Estimation -- 3.5.3.2 Net Physical Information -- 3.5.3.3 Product of the Wirelength and Congestion -- 3.5.3.4 Electrical and Logic Features. 3.5.3.5 Timing Information -- 3.5.3.6 Neighboring Net Information -- 3.5.4 Machine Learning Engines -- 3.6 Interconnect Coupling Delay and Transition Effect Prediction at Sign-Off -- 3.6.1 Problem Formulation -- 3.6.2 Prediction Flow -- 3.6.3 Feature Engineering -- 3.6.4 Machine Learning Engines -- 3.7 Summary -- References -- 4 Deep Learning for Power and Switching Activity Estimation -- 4.1 Introduction -- 4.2 Background on Modeling Methods for Switching Activity Estimators -- 4.2.1 Statistical Approaches to Switching Activity Estimators -- 4.2.2 "Cost-of-Action"-Based Power Estimation Models -- 4.2.3 Learning/Regression-Based Power Estimation Models -- 4.3 Deep Learning Models for Power Estimation -- 4.4 A Case Study on Using Deep Learning Models for Per Design Power Estimation -- 4.4.1 PRIMAL Methodology -- 4.4.2 List of PRIMAL ML Models for Experimentation -- 4.4.2.1 Feature Construction Techniques in PRIMAL -- 4.4.2.2 Feature Encoding for Cycle-by-Cycle Power Estimation -- 4.4.2.3 Mapping Registers and Signals to Pixels -- 4.5 PRIMAL Experiments -- 4.5.1 Power Estimation Results of PRIMAL -- 4.5.2 Results Analysis -- 4.6 A Case Study on Using Graph Neural Networks for Generalizable Power Estimation -- 4.6.1 GRANNITE Introduction --

4.6.2 The Role of GPUs in Gate-Level Simulation and Power Estimation -- 4.6.3 GRANNITE Implementation -- 4.6.3.1 Toggle Rate Features -- 4.6.3.2 Graph Object Creation -- 4.6.3.3 GRANNITE Architecture -- 4.7 GRANNITE Results -- 4.7.1 Analysis -- 4.8 Conclusion -- References -- 5 Deep Learning for Analyzing Power Delivery Networks and Thermal Networks -- 5.1 Introduction -- 5.2 Deep Learning for PDN Analysis -- 5.2.1 CNNs for IR Drop Estimation -- 5.2.1.1 PowerNet Input Feature Representation -- 5.2.1.2 PowerNet Architecture -- 5.2.1.3 Evaluation of PowerNet -- 5.2.2 Encoder-Decoder Networks for PDN Analysis. 5.2.2.1 PDN Analysis as an Image-to-Image Translation Task -- 5.2.2.2 U-Nets for PDN Analysis -- 5.2.2.3 3D U-Nets for IR Drop Sequence-to-Sequence Translation -- 5.2.2.4 Regression-Like Layer for Instance-Level IR Drop Prediction -- 5.2.2.5 Encoder-Decoder Network Training -- 5.2.2.6 Evaluation of EDGe Networks for PDN Analysis -- 5.3 Deep Learning for Thermal Analysis -- 5.3.1 Problem Formulation -- 5.3.2 Model Architecture for Thermal Analysis -- 5.3.3 Model Training and Data Generation -- 5.3.4 Evaluation of ThermEDGe -- 5.4 Deep Learning for PDN Synthesis -- 5.4.1 Template-Driven PDN Optimization -- 5.4.2 PDN Synthesis as an Image Classification Task -- 5.4.3 Principle of Locality for Region Size Selection -- 5.4.4 ML-Based PDN Synthesis and Refinement Through the Design Flow -- 5.4.5 Neural Network Architectures for PDN Synthesis -- 5.4.6 Transfer Learning-Based CNN Training -- 5.4.6.1 Synthetic Input Feature Set Generation -- 5.4.6.2 Transfer Learning Model -- 5.4.6.3 Training Data Generation -- 5.4.7 Evaluation of OpenPDN for PDN Synthesis -- 5.4.7.1 Justification for Transfer Learning -- 5.4.7.2 Validation on Real Design Testcases -- 5.5 DL for PDN Benchmark Generation -- 5.5.1 Introduction -- 5.5.2 GANs for PDN Benchmark Generation -- 5.5.2.1 Synthetic Image Generation for GAN Pretraining -- 5.5.2.2 GAN Architecture and Training -- 5.5.2.3 GAN Inference for Current Map Generation -- 5.5.3 Evaluation of GAN-Generated PDN Benchmarks -- 5.6 Conclusion -- References -- 6 Machine Learning for Testability Prediction -- 6.1 Introduction -- 6.2 Classical Testability Measurements -- 6.2.1 Approximate Measurements -- 6.2.1.1 SCOAP -- 6.2.1.2 Random Testability -- 6.2.2 Simulation-Based Measurements -- 6.3 Learning-Based Testability Prediction -- 6.3.1 Node-Level Testability Prediction -- 6.3.1.1 Conventional Machine Learning Methods. 6.3.1.2 Graph-Based Deep Learning Methods -- 6.3.2 Circuit-Level Testability Prediction -- 6.3.2.1 Fault Coverage Prediction -- 6.3.2.2 Test Cost Prediction -- 6.3.2.3 X-Sensitivity Prediction -- 6.4 Additional Considerations -- 6.4.1 Imbalanced Dataset -- 6.4.2 Scalability of Graph Neural Networks -- 6.4.3 Integration with Design Flow -- 6.4.4 Robustness of Machine Learning Model and Metrics -- 6.5 Summary -- References -- Part II Machine Learning-Based Design Optimization Techniques -- 7 Machine Learning for Logic Synthesis -- 7.1 Introduction -- 7.2 Supervised and Reinforcement Learning -- 7.2.1 Supervised Learning -- 7.2.2 Reinforcement Learning -- 7.3 Supervised Learning for Guiding Logic Synthesis Algorithms -- 7.3.1 Guiding Logic Network Type for Logic Network Optimization -- 7.3.2 Guiding Logic Synthesis Flow Optimization -- 7.3.3 Guiding Cut Choices for Technology Mapping -- 7.3.4 Guiding Delay Constraints for Technology Mapping -- 7.4 Reinforcement Learning Formulations for Logic Synthesis Algorithms -- 7.4.1 Logic Network Optimization -- 7.4.2 Logic Synthesis Flow Optimization -- 7.4.2.1 Synthesis Flow Optimization for Circuit Area and Delay -- 7.4.2.2 Synthesis Flow Optimization for Logic Network Node and Level Counts -- 7.4.3 Datapath Logic Optimization -- 7.5 Scalability Considerations for

Reinforcement Learning -- References -- 8 RL for Placement and Partitioning -- 8.1 Introduction -- 8.2 Background -- 8.3 RL for Combinatorial Optimization -- 8.3.1 How to Perform Decision-Making with RL -- 8.4 RL for Placement Optimization -- 8.4.1 The Action Space for Chip Placement -- 8.4.2 Engineering the Reward Function -- 8.4.2.1 Wirelength -- 8.4.2.2 Routing Congestion -- 8.4.2.3 Density and Macro Overlap -- 8.4.2.4 State Representation -- 8.4.3 Generating Adjacency Matrix for a Chip Netlist -- 8.4.4 Learning RL Policies that Generalize.  
8.5 Future Directions.

---