Detailed routing using GNN-based multi-agent reinforcement learning

Roozmehr Jalilian, Bruce Xi
Department of Electrical & Computer Engineering
The University of British Columbia
{roozmehr.jalilian,xsd99}@ece.ubc.ca
Vancouver, BC V6T 1Z4

Abstract

Routing has always been one of the most challenging stages of digital logic design for field-programmable gate arrays (FPGAs) and application-specific integrated circuits (ASICs), both in terms of complexity and time consumption. Designs with a large number of nets can take hours and maybe days to be routed, even using state-of-the-art commercial CAD tools. Thus, there has been a surge of interest to use machine-learning-based approaches to tackle these difficulties. We propose a novel multi-agent reinforcement learning (RL) approach that incorporates a graph neural network (GNN) as its policy network. Having multiple agents enables us to route multiple nets simultaneously, while using a GNN as the policy network enables the agents to generalize their learned routing strategy to new, unseen designs.

1 Introduction

Digital logic design involves three major stages: **RTL**[1]**design**, **synthesis**, and **physical design**. An engineer first decides on the architecture of a digital circuit, and describes it with the help of a *hardware description language* (HDL). The described architecture is then fed into a CAD tool, which first synthesizes the design using different logic gates, and then maps the gates onto the chip canvas.

Physical design is perhaps the most challenging stage of the whole process, as it involves two sophisticated and time-consuming phases: **placement** and **routing**. The goal of this stage is to place a large number of logic gates on the chip and route them in such a way that the final circuit satisfies multiple constraints (timing, area, etc.). Routing, in general, is the more complex, as the wiring resources on the chip are limited and routing **congestion** must also be taken into account.

Due to the problem's complexity, routing is generally broken into two sub-phases: **global** and **detailed**. Global routing first partitions the chip into routing regions and searches for region-to-region paths for all signal nets; this is followed by detailed routing, which determines the exact tracks and vias of these nets based on their region assignments [1].

Obtaining a valid solution which satisfies all constraints might usually take up to several days for large circuits, leading to a surge of interest to use machine learning techniques, aiming to find solutions much faster while improving or maintaining the same level of solution quality.

[1] Register transfer level

Preprint. Under review.

2 Related work

TODO: Mention 1 or more analytical routers, and mention some routers that also incorporate RL and/or GNNs. At the end, mention that no other work has used a multi-agent RL approach which also involves a GNN (no one even has used vanilla multi-agent for that matter). I suggest you also include a very brief outline of RL and GNNs in here, before going into detail for the papers that have used them.

3 Methodology

TODO: Briefly explain what is multi-agent RL and its advantages for our given problem. Also mention the paper which used a GNN as the policy network for the RL agent [2]. Also discuss the overall architecture of the GNN that we're going to use, and mention which libraries we're using (i.e., PyTorch, PyG, and RLLib). Adding a figure depicting the routing problem in a grid would also be very helpful.
Also, don't forget to come up with a rough timeline and replace the 'X' values below!

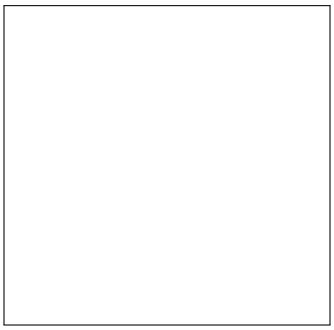

Figure 1: A simple routing scheme

Here is a list of tasks that needs to be accomplished in order:

- Extended literature review (X days)
- Getting familiar with the required Python libraries (X days)
- Finding appropriate benchmarks and converting them into the same format (X days)
- Implementing the RL agent (X days)
- Training the RL agent and tuning hyperparameters (X days)
- Testing the RL agent on benchmarks both similar to and different from the training designs (X days)
- Creating figures and tables (X hours)
- Writing the project report (X days)

4 Experiments

As stated in Section 3, our goal is to develop a model that not only routes different nets simultaneously, but is also able to generalize to unseen designs based on its learned policy.

4.1 Testing on benchmarks with similar circuit size to the training data

The goal of this experiment is to ensure that the model has learned a good policy based on its training data. We would test the model on circuits with canvas sizes, number of nets, and total number of pins per net similar to that of the training benchmarks. We would then compare the results to other routers in terms of runtime, memory usage, and solution quality.

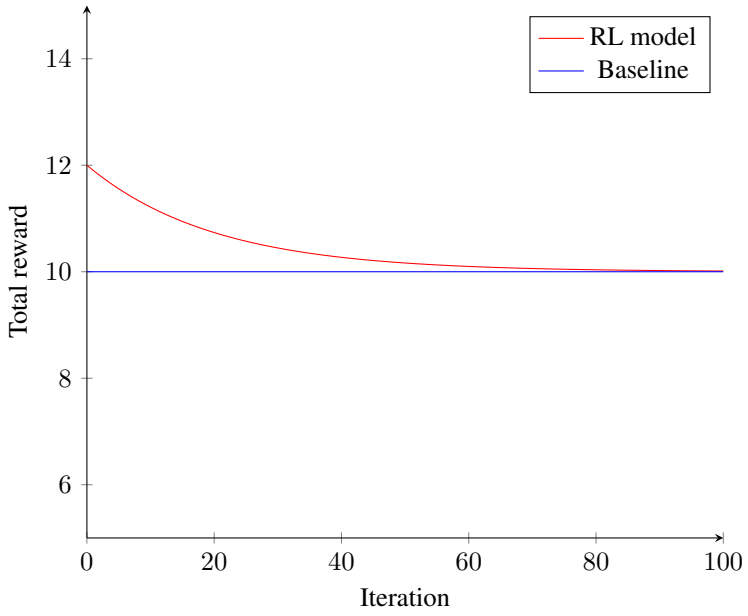

Figure 2: Placeholder figure for total reward vs. time

Table 1: Placeholder table for the experiments

Benchmark specs				Results		
Name	Canvas size	# of nets	Avg. # of pins per net	Runtime (s)	Max. memory usage (MB)	Total wire-length
test1	8×8	5	2.5	1.25	250	100
test2	6×6	3	1	0.60	150	70

4.2 Testing on benchmarks with different circuit size than the training data

We aim to test the *generalizability* of the model in this experiment, by applying to designs with different number of nets, chip canvas size, and number of pins per net. However, we're won't use benchmarks that are *significantly* different from the training designs, as the complexity of routing increases exponentially within very large designs. The chip canvas size for the test designs won't be much larger than the training designs, and we're more interested in observing how our model handles *different net topologies* in similarly-sized circuits.

References

[1] A. B. Kahng, J. Lienig, I. L. Markov, and J. Hu, "Global routing," *VLSI Physical Design: From Graph Partitioning to Timing Closure*, pp. 131–169, 2022. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-96415-3_5

[2] P. Almasan, J. Suárez-Varela, K. Rusek, P. Barlet-Ros, and A. Cabellos-Aparicio, "Deep reinforcement learning meets graph neural networks: Exploring a routing optimization use case," *Computer Communications*, vol. 196, pp. 184–194, 12 2022.