Problem C: Microarchitecture Design Space Exploration

(DAMO Academy)

Q&A

Q1. Regarding the problem C, I have the following concept.
I want to build an offline dataset with objective functions like power, performance, area, congestion score with early physical estimation. I have encountered some issues with building the entire VLSI flow(Chipyard, Hammer and ASAP7,.etc). Since the contest also builds the entire flow and collects the data for the contest. Could you please help to give some suggestions? Thank you very much.
A1. We will provide this dataset later soon. The data collection process is time-consuming and resource-consuming. Please consider using the provided dataset for this contest.

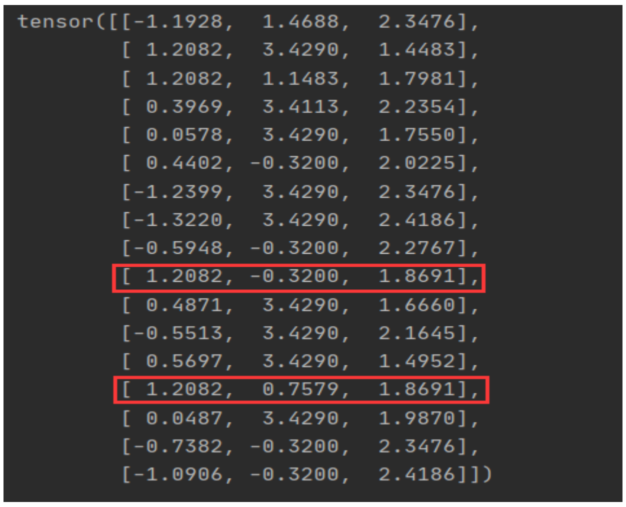
Q2. I have a question about Problem C since I noted that in the instruction document, some example optimizers are referenced. However, I could only find the source code of the random optimizer in the instruction doc, and there is no related code in the provided package installed via pip. May I know whether it is possible to provide the code of the other example optimizers?
Thank you.
A2. Please refer to the released case on 5/31.

Q3. Could you please help to advise if the data return to us for each test is a single ADRS value or PPA values? Thank you.
A3. Yes, the result after running a test is a ADRS value, not PPA.

Q4. We have two additional questions about problem C. Could you please help to check? Thank you.
1) In the dataset to be released in the future, for a specific design, are all the feasible configurations in the corresponding design space (along with PPA & times) provided in the dataset? Otherwise, it could happen that we may query a configuration that is feasible but not included in the dataset.
2) Is there a way to perform "early stopping" using the given codebase? Since we notice that the number of queries is provided as an constant argument when launching the script, but it could be possible that an optimization strategy wants to determine this number dynamically and early stop if needed.
A4. Below please find our responds.
1) All configurations are provided in the dataset, and you should make sure your inquiry is valid, i.e., a valid configuration within the design space.

2) You can configure the number of queries provided, i.e., you can specify the number of queries in your optimizer so that you can terminate an optimization strategy w.r.t. your "early stopping" criterion.
You can specify the number of queries through '-q' or ' --num-of-queries' option when you launch your optimizer.

Q5. Could you please help to check the following questions? Thank you very much!
1). Is there a proportional or inverse proportional independent relationship between ADRS and the normalized value of P, P, and A in this problem, respectively?
2). When we try to check the relation between ADRS and the normalized PPA, we find a seemingly unexpected PPA appearing in the reference pareto frontier after filtering by get_non_dominated() (please refer to the attachment). Is it proper that these two PPAs appear simultaneously in the same pareto frontier? Will it affect ADRS if the predicted pareto frontier is worse than the reference pareto frontier?



```
tensor([[-1.1928,  1.4688,  2.3476],
        [ 1.2082,  3.4290,  1.4483],
        [ 1.2082,  1.1483,  1.7981],
        [ 0.3969,  3.4113,  2.2354],
        [ 0.0578,  3.4290,  1.7550],
        [ 0.4402, -0.3200,  2.0225],
        [-1.2399,  3.4290,  2.3476],
        [-1.3220,  3.4290,  2.4186],
        [-0.5948, -0.3200,  2.2767],
        [ 1.2082, -0.3200,  1.8691],
        [ 0.4871,  3.4290,  1.6660],
        [-0.5513,  3.4290,  2.1645],
        [ 0.5697,  3.4290,  1.4952],
        [ 1.2082,  0.7579,  1.8691],
        [ 0.0487,  3.4290,  1.9870],
        [-0.7382, -0.3200,  2.3476],
        [-1.0906, -0.3200,  2.4186]])
```

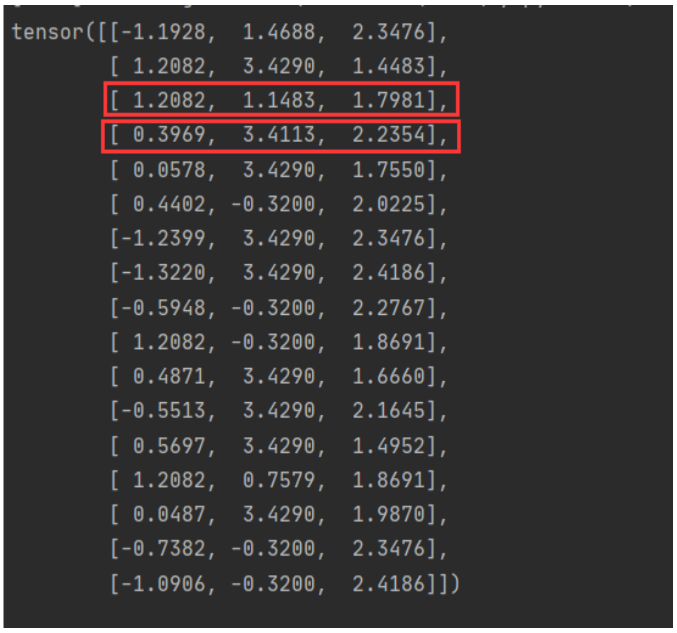
3). Can we import get_adrs from iccad_contest.functions.problem and directly call function get_adrs() in the implement of function observe()?
A5. Below please find our responds.
1) A higher normalized performance value, a lower normalized power value, and a lower normalized area value suggest that the design is on the true Pareto frontier with a higher possibility. Suppose your optimizer can explore designs with a higher normalized performance value, lower normalized power value, and lower normalized area value as much as possible. In that case, the ADRS of your optimizer will be lower, indicating better your optimizer is. Simultaneously, it would be best to consider the overall running time of your optimizer. A lower overall running time of your optimizer is expected.
2) It is normal that these two PPA values are on the true Pareto frontier.
Firstly, you can inspect their PPA values by printing them using higher precision, e.g., 8 significant digits. Secondly, the designs corresponding to these two PPA values indicate approximately the same performance. However, one design dissipates more power and has a smaller area, while the other one dissipates less power but has a larger area. Both of them cannot dominate each other.
3) You can leverage all functions provided by the contest platform.

Q6. We have a question about problem C and would like a clarification.
Can we pretrain a model locally using provided or self-generated data, and submit the obtained model parameter files together with other code files to the online evaluation system for further operations (such as fine-tuning for the specific dataset)? Or is this not allowed and all training that might be involved has to be done online from scratch? We think the latter seems more reasonable, otherwise it would be difficult to maintain a fair evaluation of ORT.
Thank you.

A6. Yes, you are right. It's not allowed to submit the obtained model parameter files to the online evaluation system. Any model leveraged in your optimizer should be trained online, and the training time is also corporated into the ORT automatically by the contest platform.
The submission includes:

1. **your implemented Python scripts,**
2. **the execution commands,**
3. **related JSON configurations leveraged by your algorithm, if any, and**
4. **dependency packages list for supporting your algorithm execution, if any.**

Q7. We find a PPA with higher performance, lower power, and smaller area and another one with lower performance, higher power, and larger area appear simultaneously in the reference pareto frontier. According to the previous clarification, the former shall dominate the latter, right?



```
tensor([[-1.1928,  1.4688,  2.3476],
        [ 1.2082,  3.4290,  1.4483],
        [ 1.2082,  1.1483,  1.7981],
        [ 0.3969,  3.4113,  2.2354],
        [ 0.0578,  3.4290,  1.7550],
        [ 0.4402, -0.3200,  2.0225],
        [-1.2399,  3.4290,  2.3476],
        [-1.3220,  3.4290,  2.4186],
        [-0.5948, -0.3200,  2.2767],
        [ 1.2082, -0.3200,  1.8691],
        [ 0.4871,  3.4290,  1.6660],
        [-0.5513,  3.4290,  2.1645],
        [ 0.5697,  3.4290,  1.4952],
        [ 1.2082,  0.7579,  1.8691],
        [ 0.0487,  3.4290,  1.9870],
        [-0.7382, -0.3200,  2.3476],
        [-1.0906, -0.3200,  2.4186]])
```

A7. We update and release the contest platform.
Please update the contest platform on your local machine.
The variable `problem.pareto_frontier` is an 'nx3' matrix, denoting 'n' points of objective values on the Pareto frontier.
Each point is a '1x3' vector, with each element representing the metric score of performance, power, and area values, respectively.
A better design has higher performance, less power, and a smaller area.

Hence, a better design has larger metric scores, i.e., the larger each element of a '1x3' vector, the better the design is.
We negate the power and area values to calculate the Pareto frontier, aligning with the definition of a better design.

Q8. We suspect that there is a bug in the provided codebase. When computing the golden Pareto frontier of the design space, the power and area values are negated (problem.py, line 110-114).
However, when computing the Pareto frontier of the explored set, these values are not negated (design_space_exploration.py, line 208-228).
A8. We update and release the contest platform. Please check the latest version. If you have any questions, comments, or bug reports, please do not hesitate to contact us.

Q9. Could you please clarify that for each of the three values of the evaluated PPA (the variable objective_values, or y of the observe method), is it higher is better or lower is better?
A9. Each element of the variable \`objective_values\` or \`y\` defined in the contest platform denotes normalized performance, power, and area values, respectively. The design is better if it has a higher normalized performance value and lower normalized power and area values. In conclusion, in each element of \`objective_values\` or \`y\`, if the first value should be larger, the second and the third value should be smaller, then the microarchitecture is better.

Q10. For the alpha submission of Problem C, should we submit to the TSRI machines, or there is another server for Problem C?
A10. Please submit to the TSRI machines.

Q11. Does the given dummy dataset meet the condition "A higher normalized performance value, a lower normalized power value, and a lower normalized area value suggest that the design is on the true Pareto frontier with a higher possibility" stated in A5?
A11. The dummy dataset meets the condition "A higher normalized performance value, a lower normalized power value, and a lower normalized area value suggest that the design is on the true Pareto frontier with a higher possibility".

Q12. Is the component "time of the VLSI flow" of the dummy case reasonable, i.e., the evaluation of each microarchitecture embedding costs about 30,000+ seconds?
A12. It is reasonable. We leverage a series of commercial electronic design automation (EDA) tools to evaluate and verify each microarchitecture. It does consume much time.

Q13. In the final calculation of ADRS, any two PPAs employed to determine the same Pareto frontier shall never dominate each other. Otherwise, it will affect the calculated result of ADRS. Is it right?

A13. If two objective values fall into the same Pareto frontier, they cannot dominate each other.

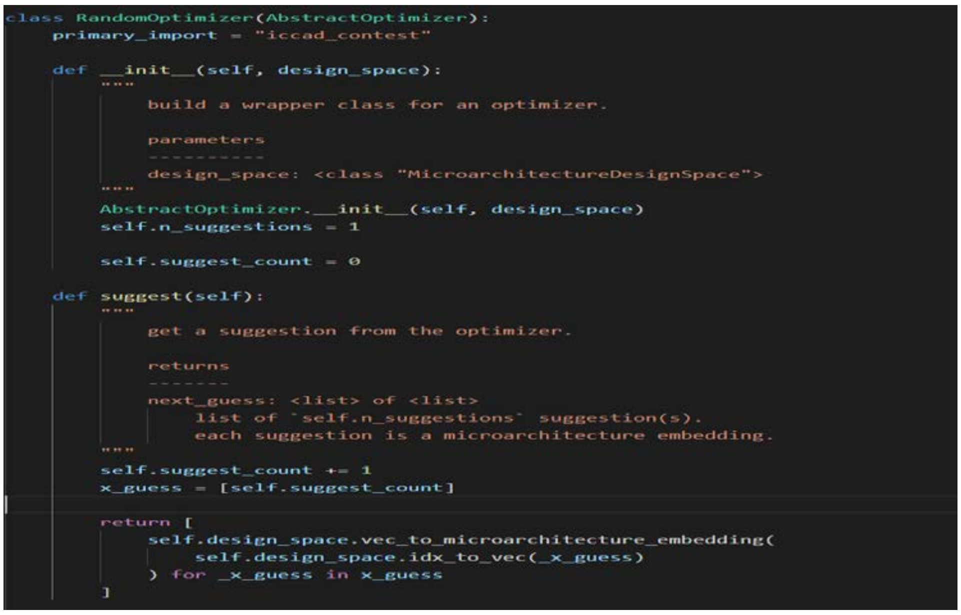
Q14. As the number of predicted PPAs grows with our suggestion, ADRS between the predicted Pareto frontier and the golden Pareto frontier will only keep unchanged or decrease but never increase, right?

A14. If your optimizer is reasonable and works well, ADRS will decrease monotonically as your suggestion increases.

Q15. We notice that the dummy design space consists of several sub-embeddings such as ISU and IFU. Does every combination of these sub-embeddings' candidate options always compose a valid microarchitecture? Will the design space of the real dataset also consist of sub-embeddings?

A15. Every combination of these sub-embeddings' candidate options composes a valid microarchitecture. The real dataset consists of these sub-embeddings.

Q16. Following up our previous Q&A, in fact we have observed some abnormality when using the provided codebase of Problem C. As a simple example, we modified the random_search optimizer as shown in the bottom fig, which essentially suggests the configurations in the design space sequentially without randomness. We found that the reported ADRS firstly decreases and then increases (set -q > 100). According to the definition, ADRS should not increase when adding new points to an existing set. Hence we suspect that there might be something wrong within the codebase.
Could you please kindly look into this issue?



```
class RandomOptimizer(AbstractOptimizer):
    primary_import = "iccad_contest"

    def __init__(self, design_space):
        """
            build a wrapper class for an optimizer.

            parameters
            ----------
            design_space: <class "MicroarchitectureDesignSpace">
        """
        AbstractOptimizer.__init__(self, design_space)
        self.n_suggestions = 1

        self.suggest_count = 0

    def suggest(self):
        """
            get a suggestion from the optimizer.

            returns
            -------
            next_guess: <list> of <list>
                list of `self.n_suggestions` suggestion(s).
                each suggestion is a microarchitecture embedding.
        """
        self.suggest_count += 1
        x_guess = [self.suggest_count]

        return [
            self.design_space.vec_to_microarchitecture_embedding(
                self.design_space.idx_to_vec(_x_guess)
            ) for _x_guess in x_guess
        ]
```

A16. Thanks for your question. However, we cannot reproduce this error.
We updated the platform yesterday, can you please use the latest version (0.0.3) and let us know if this issue still exists. Thanks.

Q17. We have a question about problem C. Could you please help to check?
We notice that the number of queries (i.e., -q) should be provided when launching the script.   It is determined before we learn about the design space,which seems inappropriate. Is there any method we could use to change the number of queries during the optimization? We would appreciate it if any information could be provided. Thank you.

A17. Thanks for your suggestions. We have updated the contest platform. Please update it with the command: pip3 install iccad-contest==0.0.4
You can apply the early stopping criterion conveniently with the latest platform.