Permutation-based Optimization using a Generative Adversarial Network

Sami Lemtenneche
Kasdi Merbah University
Ouargla, Algeria
lemtenneche.sami@univ-ouargla.dz

Abdelhakim Cheriet
Kasdi Merbah University
Ouargla, Algeria
abdelhakim.cheriet@univ-ouargla.dz

Bensayah Abdellah
Kasdi Merbah University
Ouargla, Algeria
bensayah.abdallah@univ-ouargla.dz

ABSTRACT

This paper presents a new method to deal with permutation-based problems using a Generative Adversarial Network. Generative adversarial Networks (GANs) are used in many fields as models that mimic the trained data distribution. However, dealing with permutation space in GANs has not been yet investigated in detail. To reach this objective, we propose using GANs to solve the permutation-based optimization problem using an Estimation of Distribution Algorithm. We use a Random Key method to translate the generated data from continuous values to permutations. The Experiment aims to investigate the quality of solutions provided by the proposed method in two ways. Firstly, we evaluate the solutions obtained directly from the GANs, and in the second one, we use a local search method to improve the quality of the solution. In the end, We perform a set of experiments on instances of the known TSP problems. Also, we notice that the obtained results promise to extend the work on other classes of problems.

CCS CONCEPTS

• **Mathematics of computing → Evolutionary algorithms**; *Combinatorial optimization*; • **Computing methodologies** → Neural networks; *Generative and developmental approaches.*

KEYWORDS

Generative Adversarial Networks, permutation-based problem, estimation of distribution algorithm, evolutionary algorithm.

ACM Reference Format:
Sami Lemtenneche, Abdelhakim Cheriet, and Bensayah Abdellah. 2021. Permutation-based Optimization using a Generative Adversarial Network. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459423

1 INTRODUCTION

Estimation of Distribution Algorithms (EDAs) are a subset of evolutionary algorithms (EAs). EDAs explore the search space by building and sampling explicit probabilistic models of promising solutions at each generation. EDAs were successfully applied to many optimization problems[1–3]. A pseudo-code of an EDA is:

Algorithm 1: The general pseudo code of EDAs

```
Generate initial population;
while Termination Criteria not met do
    Select some promising individuals;
    Build probabilistic model M using selected individuals;
    Sample M to generate new individuals;
    Replace old population by new individuals;
```

On the other hand the generative adversarial networks (GANs) are widely used in Computer Vision [6]. It aims to learn the probability distribution of trained data and use it to generate new data. GANs have been used to estimate the probabilistic model in the field of evolutionary multiobjective optimization [5]. Motivated by this success of GANs, we propose to introduce GANs in EDAs to solve permutation-based problems. First, we explain what a Generative Adversarial Network is. The next section introduces the proposed algorithm then we present the results of the experiments conducted. Finally, we end this work with a conclusion that contains the main results and some perspectives.

2 GENERATIVE ADVERSARIAL NETWORKS

Generative adversarial networks (GANs) are a class of generative deep learning models introduced by Ian Goodfellow et al.[4]. Generally, the generative adversarial networks are a model which combine two neural networks, a **generator G** and a **discriminator D**.

The generator receives as inputs a random vector and generates fake data to fool the discriminator, while the discriminator receives real samples from data training or generated samples from the generator and outputs the probability of this input coming from the real data instead of the generator. The term adversarial means that G and D compete with each other to achieve their individual goals. The GAN model converges when the discriminator and the generator reach a Nash equilibrium. This is the optimal point for the objective function.

3 PROPOSAL

In our work, we train a GAN to learn the distribution of a given population then use the trained GAN to sample new individuals (see Algorithm 2). We repeat this process in each iteration of the algorithm. In other words, the generator creates a probabilistic model of the data set training(selected individuals) across the training phase and creates a new population using this generator (sampling phase).

We start by generating an initial population randomly, then the selected individuals will be normalized between [0,1] and labeled as

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459423

Algorithm 2: A pseudo code of the proposed GAN-EDA

Generate initial population;
while *Termination Criteria not met* **do**
  Select the promising individuals and normalize their gens between [0,1];
  Train GAN using the selected individuals;
  Generate new samples using the trained generator;
  Apply random key to get individuals;
  Replace old population by the new individuals;

real data. We use the generator to produce a set of offspring labeled as fake data. The discriminator is trained with real and fake data and tries to distinguish between them. The generator is updated by the generated samples, while its input is a random vector sampled from a normal distribution.

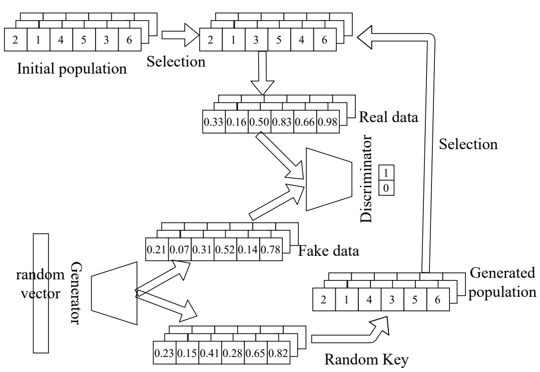

Figure 1: General scheme of the proposed algorithm

The generator can not generate a permutation directly (discrete vector and mutual exclusivity condition), but it generates real values. So we should encode their outputs to get a permutation, to do that we use the traditional Random Keys algorithm by ranking the position of generated values. We train the GAN for some number of epochs. After that, we use the generator to create the next generation, and so on. An overview of the process is presented in Figure 1.

4 EXPERIMENTAL

In this section, we study the results of the proposed algorithm using selected instances of the TSP. For this, we have used a generator and a discriminator that has both a Multilayer Perceptron architecture with two hidden layers. The input layer of the discriminator and the output layer of the generator has the same size, which is equal to the individual's size. The table 1 present the best find parameters of the GAN .

The results of the experiments shown in Table 2 present the best results obtained by the proposed algorithm. Our proposed algorithm has two variants, with a local search 2-opt method and without a 2-opt method. The quality of the solutions found in the proposal using the 2-opt algorithm is better for all used benchmarks.

Parameter	Value
Maximum number of generation	100
Population size	500
Selection size	50
Number of epochs	50
Learning rate	1e-3
Activation function (output layer of D)	Sigmoid
Activation function (output layer of G)	Tanh
Activation function in hidden layer	Relu
Batch size	32

Table 1: Parameters of the proposed algorithm and the Generative Adversarial Network

Benchmark	Best known	With 2-opt	Without 2-opt
gr17	2085	2128,2	2808,8
bayes29	2020	2806,4	4389,8
fri26	937	1110	1334,6
DANTZIG42	699	929,2	1327,4

Table 2: The obtained results of the proposed algorithm compared to the best know solutions

5 CONCLUSION

In this paper, we proposed a new estimation of distribution algorithm to solve the permutation-based problems. The main component of an EDA is how it estimates the model of the best solutions found in every generation. We used a generative adversarial network as an estimator because of his power to estimate models in the continuous space. However, using GAN to learn and generate permutation has not been investigated yet. We proposed a new way to train a GAN from a set of permutations and then use it to generate a new permutation to guide the search during the optimization process. The Experiments conducted in this paper aim to investigate how close the GAN can generate samples with the same proprieties of input permutations and use them in the optimization process. The results obtained for the used benchmarks are promising but need more investigation, essentially the number of permutations used to train the GAN.

REFERENCES

[1] Abdelhakim Cheriet. Vine copula-based EDA for dynamic multiobjective optimization. *Evolutionary Intelligence*, pages 1–25, 2020.
[2] Abdelhakim Cheriet and Roberto Santana. Modeling dependencies between decision variables and objectives with copula models. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '18, page 175–176, New York, NY, USA, 2018. Association for Computing Machinery.
[3] Abdelhakim Cheriet and Roberto Santana. Optimizing permutation-based problems with a discrete vine-copula as a model for EDA. In *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, GECCO '19, page 143–144, New York, NY, USA, 2019. Association for Computing Machinery.
[4] Ian J Goodfellow, Jean Pouget-abadie, Mehdi Mirza, Bing Xu, and David Warde-farley. Generative Adversarial Nets. pages 1–9, 2014.
[5] C. He, S. Huang, R. Cheng, K. C. Tan, and Y. Jin. Evolutionary multiobjective optimization driven by generative adversarial networks (gans). *IEEE Transactions on Cybernetics*, pages 1–14, 2020.
[6] Tzu-An Song, Samadrita Roy Chowdhury, Fan Yang, and Joyita Dutta. PET image super-resolution using generative adversarial networks. *Neural Networks*, 125:83–91, 2020.