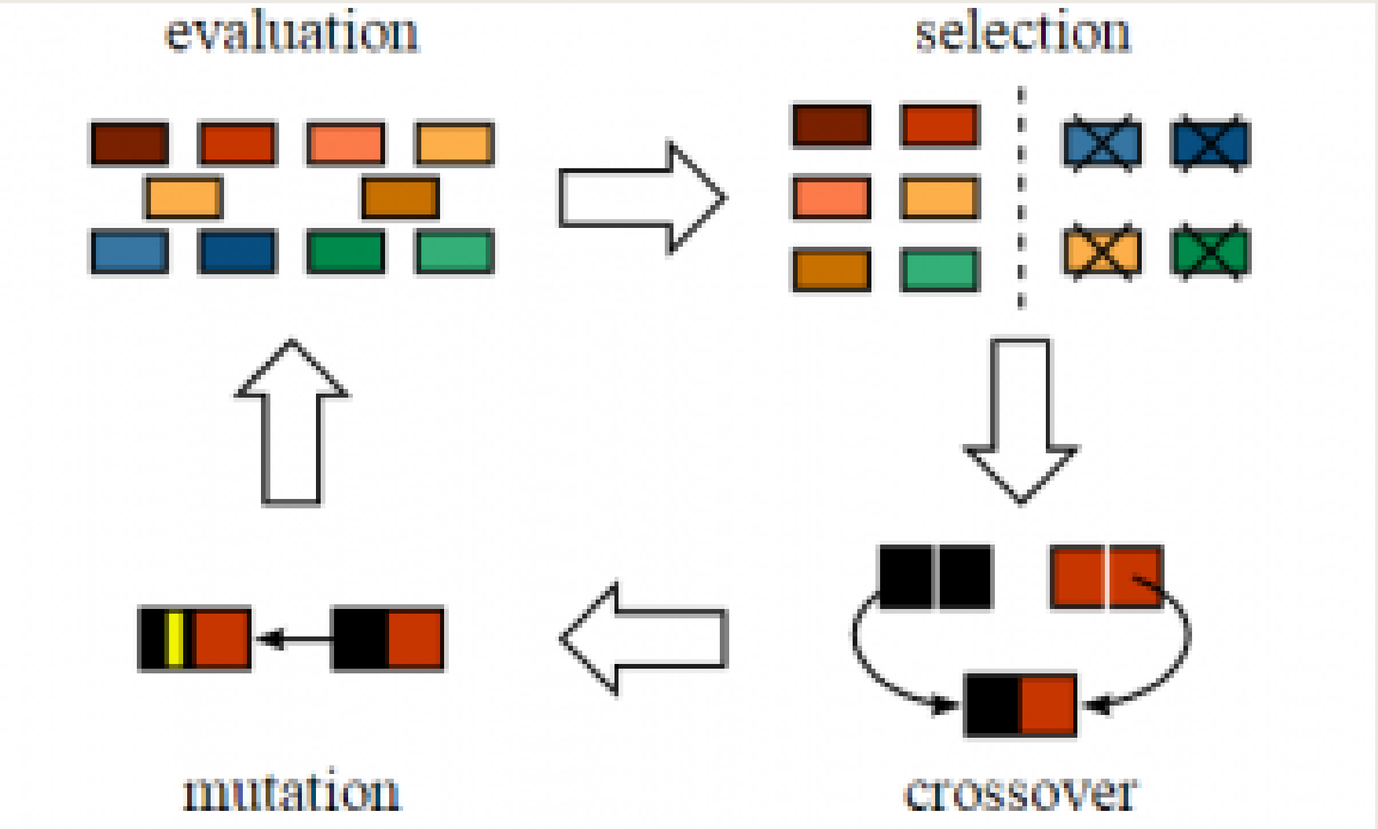
A Novel Methodology for Optimizing Direct Response Display Advertising Campaigns

Luis Miralles-Pechu´an, Hiram Ponce, Lourdes Mart´ınez-Villase˜nor



- 21522678 - Phạm Trung Tín
- 21522798 - Lương Triệu Hoàng Vũ

NỘI DUNG TRÌNH BÀY

I Tóm tắt

II Online Advertising Campaigns

III CTR Model

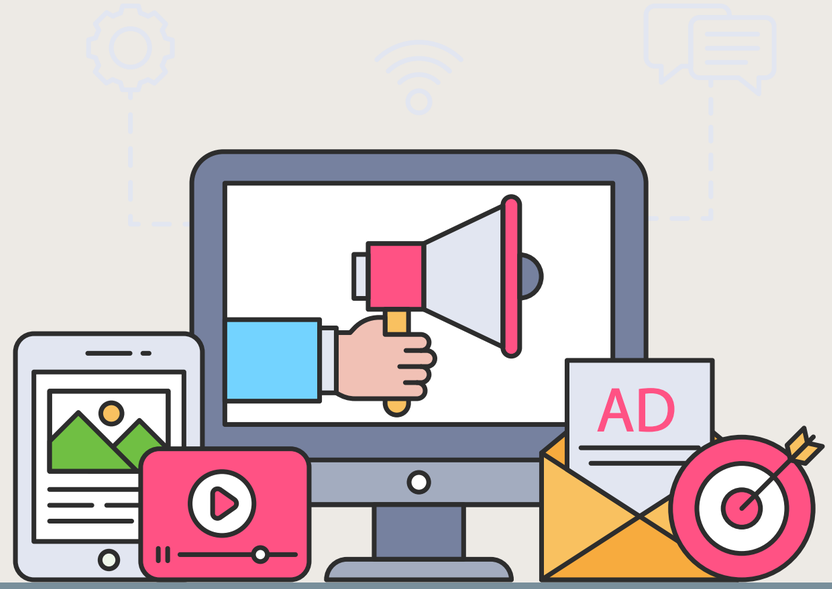
IV Genetic Algorithm

I. TÓM TẮT

- Các chiến dịch quảng cáo trực tuyến đã thu hút sự chú ý của nhiều nhà quảng cáo mong muốn quảng bá hoạt động kinh doanh của họ trên Internet
- Một trong những vấn đề chính là cấu hình các chiến dịch của họ một cách hiệu quả
  - Lựa chọn mục tiêu phù hợp
  - Đảm bảo tỷ lệ chấp nhận của người xem

Bài báo này trình bày một phương pháp mới để tối ưu hóa kỹ thuật trong các chiến dịch quảng cáo trực tuyến bằng sử dụng thuật toán di truyền làm mô hình tối ưu hóa cơ sở và mô hình CTR dựa trên học máy.

II. ONLINE ADVERTISING CAMPAIGNS



2.1

Ecosystem for Online Advertising

2.2

Display Advertising Optimization

2.1. ECOSYSTEM FOR ONLINE ADVERTISING

- Theo Interactive Advertising Bureau, quảng cáo kỹ thuật số, bao gồm quảng cáo trực tuyến và trên thiết bị di động, đã có mức tăng trưởng mạnh mẽ qua từng năm

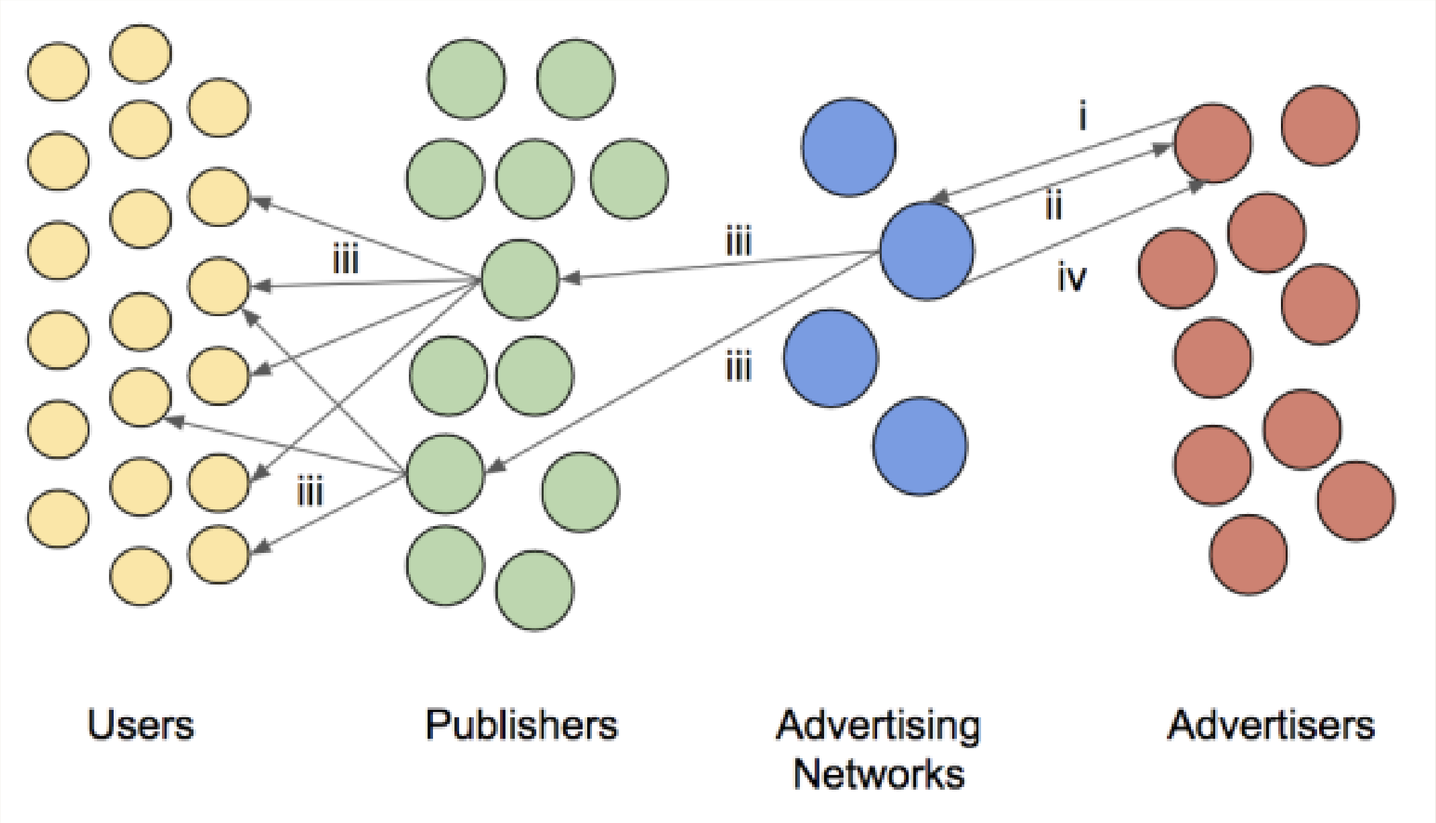


Figure 1: Các bước chính của quá trình đăng tải quảng cáo

- (i): Nhà quảng cáo cấu hình chiến dịch bằng cách chọn và cấu hình một bộ tham số;
- (ii): Mạng quảng cáo ước tính giá và lưu lượng truy cập cho từng cấu hình
- (iii): Nhà quảng cáo đặt mức giá và ngân sách tối đa cho chiến dịch
- (iv): Quảng cáo được hiển thị cho đến khi hết ngân sách

2.1. ECOSYSTEM FOR ONLINE ADVERTISING

Cách kết nối liên tục giữa
người dùng & nhà quảng cáo

01

Quảng cáo dựa trên
tìm kiếm bằng các từ khóa

Các mô hình định giá phổ biến là:

- Giá theo lượt nhấp (CPC)
- Giá theo hành động (CPA)
- Giá theo khách hàng tiềm năng (CPL)

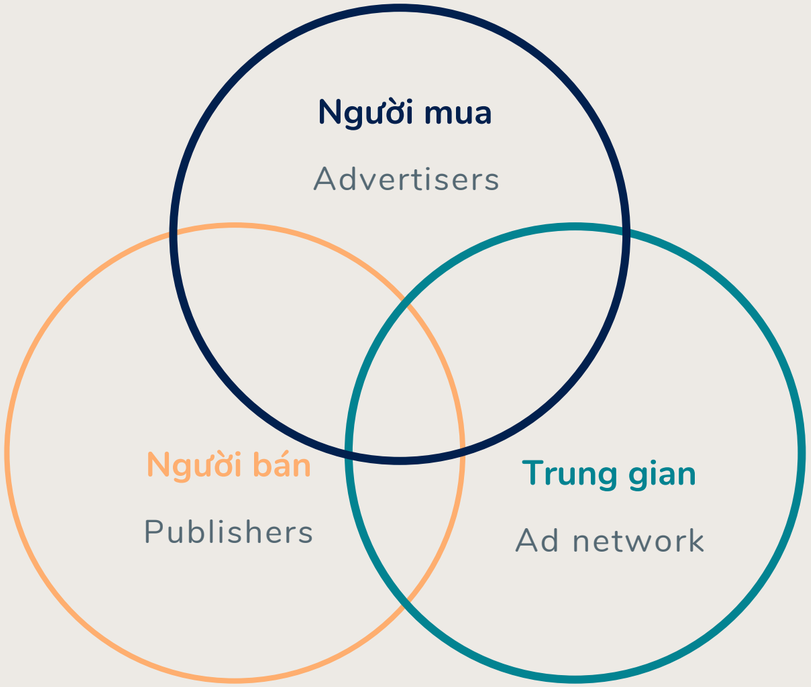
02

Quảng cáo
hiển thị hình ảnh

Các mô hình định giá phổ biến là:

- Định giá chi phí trên mỗi nghìn lần hiển thị (CPM)
- Cũng bao gồm CPC, CPA và CPL.

2.2. DISPLAY ADVERTISING OPTIMIZATION



- Tập trung vào góc nhìn của nhà quảng cáo
- Mối quan hệ giữa nhà quảng cáo và mạng quảng cáo

IN

CSDL các chiến dịch QC

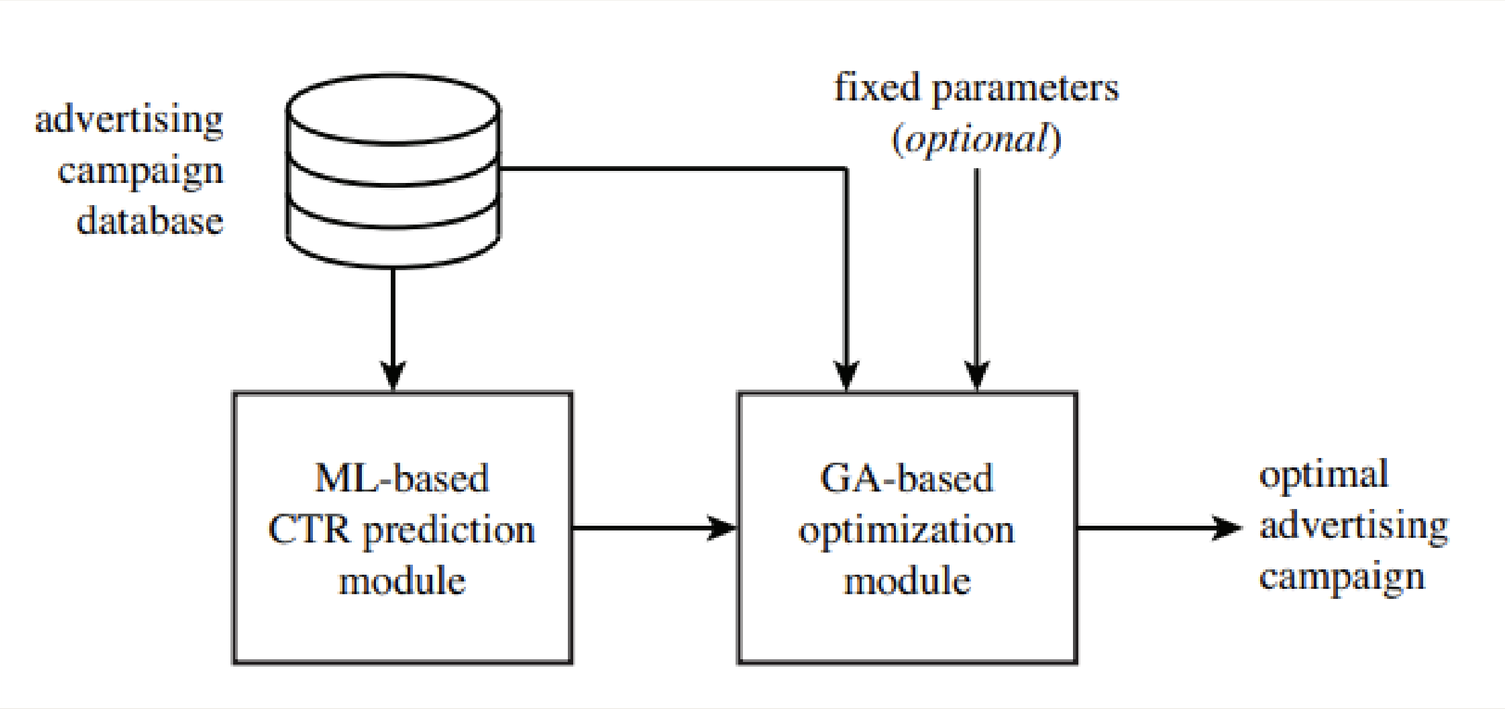
- Đối tượng, content QC, tỉ lệ CTR,...

OUT

TOP những chiến dịch QC tốt nhất

2.2. DISPLAY ADVERTISING OPTIMIZATION

Sử dụng phương pháp tối ưu hóa heuristic, tức là thuật toán di truyền, thay vì lập trình tuyến tính hoặc mô hình phi tuyến tính



III. CTR MODEL

3.1

Dataset & Hasing Trick

3.2

Design of the Prediction CTR Model

3.3

Result

DATASET & HASING TRICK

DATASET

- Tập dữ liệu Avazu về chiến dịch quảng cáo dành cho thiết bị di động điện thoại (Avazu, 2015) gồm 4M samples
- Tập dữ liệu bao gồm 24 thuộc tính: ‘id', 'click', 'hour', 'C1', 'banner_pos', 'site_id', ..
- Có sự mất cân bằng khá lớn giữa 2 nhãn: label 1 (~600k / 16.98%), label 0 (~3400k / 83.02%)

HASING TRICK

- Hasing Trick là một phương pháp khéo léo để mô hình hóa các tập dữ liệu với số lượng lớn thông tin bằng cách sử dụng hàm băm
- Mã hóa các giá trị của các features thành 1 số nguyên nằm trong khoảng từ 0 đến D
- Đặt kích thước D phải lớn để tránh xung đột
- Paper set up D=2**20, mã hoá 22 features trừ ‘id’ và ‘click’

DESIGN OF THE PREDICTION CTR MODEL

TRAINING

- Thiết kế dựa trên mô hình online logistic regression. Gồm 2 mảng:
- w : mảng biểu diễn trọng số của mỗi feature
- n: mảng số nguyên thể hiện số lần xuất hiện của mỗi thuộc tính sau khi áp dụng hashing trick
- Giá trị w sẽ được updated trong quá trình training bằng công thức:

$$w[i] = w[i] - \frac{\alpha(p - y)}{\sqrt{n[i] + 1}} \quad (1)$$

PREDICT

- predited - p là kết quả sau khi áp dụng hàm sigmoid:

$$p = \frac{1}{1 + exp\left(-\sum_{i=1}^{N} w[f_i]\right)}$$

DESIGN OF THE PREDICTION CTR MODEL

Algorithm 2 Training and testing the prediction CTR model.

Require: dataset
Ensure: prediction CTR model

1: $data \leftarrow$ Randomly select 12M samples visits from the original dataset
2: $data_{hash} \leftarrow$ Apply the hashing trick to $data$
3: Divide $data_{hash}$ into *training* and *testing* sets ▷ 10M *training* and 2M for *testing*
4: $D \leftarrow 2^{20}$ ▷ length of w and n
5: $\alpha \leftarrow 0.1$
6: $w \leftarrow \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$ of length D
7: $n \leftarrow \begin{bmatrix} 0 & 0 & 0 & \cdots & 0 \end{bmatrix}$ of length D
8: ▷ Training the CTR model
9: **for all** $v_k \in training$ **do** ▷ v_k is a *training* sample
10: $s \leftarrow 0$
11: **for all** $f_i \in v_k$ **do**
12: $s \leftarrow s + w[f_i + 1]$
13: **end for**
14: $p_k \leftarrow 1/(1 + exp(s))$
15: **for all** $f_i \in v_k$ **do**
16: $w[f_i] \leftarrow w[f_i] - \alpha(p_k - y_k)/(\sqrt{n[f_i] + 1})$
17: $n[f_i] \leftarrow n[f_i] + 1$
18: **end for**
19: **end for**
20: ▷ Testing the CTR model
21: **for all** $v_k \in testing$ **do** ▷ v_k is a *testing* sample
22: $s \leftarrow 0$
23: **for all** $f_i \in v_k$ **do**
24: $s \leftarrow s + w[f_i + 1]$
25: **end for**
26: $p_k \leftarrow 1/(1 + exp(s))$
27: **end for**
28: Compute the accuracy of the model

DEEP NEURAL NETWORK MODEL (DNN)

```
from tensorflow.keras.models import load_model
# load model
model=load_model('/content/drive/MyDrive/archive/dnn2.h5')
model.summary()
```

```
Model: "sequential_6"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense_30 (Dense)            (None, 60)                1380

 dense_31 (Dense)            (None, 10)                610

 dense_32 (Dense)            (None, 10)                110

 dense_33 (Dense)            (None, 10)                110

 dense_34 (Dense)            (None, 1)                 11

=================================================================
Total params: 2221 (8.68 KB)
Trainable params: 2221 (8.68 KB)
Non-trainable params: 0 (0.00 Byte)
_________________________________________________________________
```

RESULT

3.2M training and 0.8M for testing with CTR model

```
accuracy
```

```
0.817842390363353
```

```
from sklearn.metrics import log_loss
log_loss(X_test['click'], y_predd)
```

```
0.7287757483544941
```

```
from sklearn.metrics import mean_squared_error
mean_squared_error(X_test['click'], y_predd)
```

```
0.2567193211433944
```

```
from sklearn.metrics import f1_score
f1_score(X_test['click'], y_pred)
```

```
0.03217137037158721
```

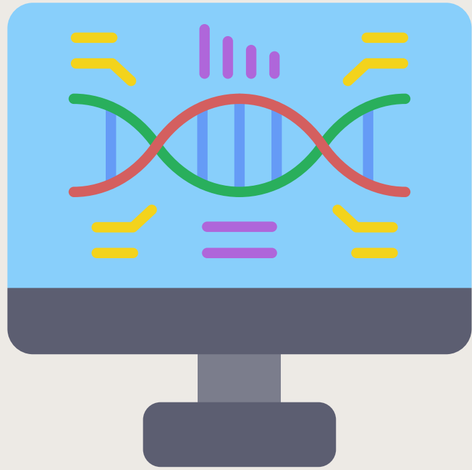
3.2M training and 0.8M for testing with DNN model

```
logarithmic loss: 0.4109585387641887
root-mean squared error: 0.1281496647286192 4
Accuracy: 83.34%
f1 score: 0.08170582902172506
```

10M training and 2M for testing with CTR model (paper)

accuracy	0.8352
logarithmic loss	0.3967
root-mean squared error (RMSE)	0.3524

IV. GENETIC ALGORITHM



4.1

Individual Encoding

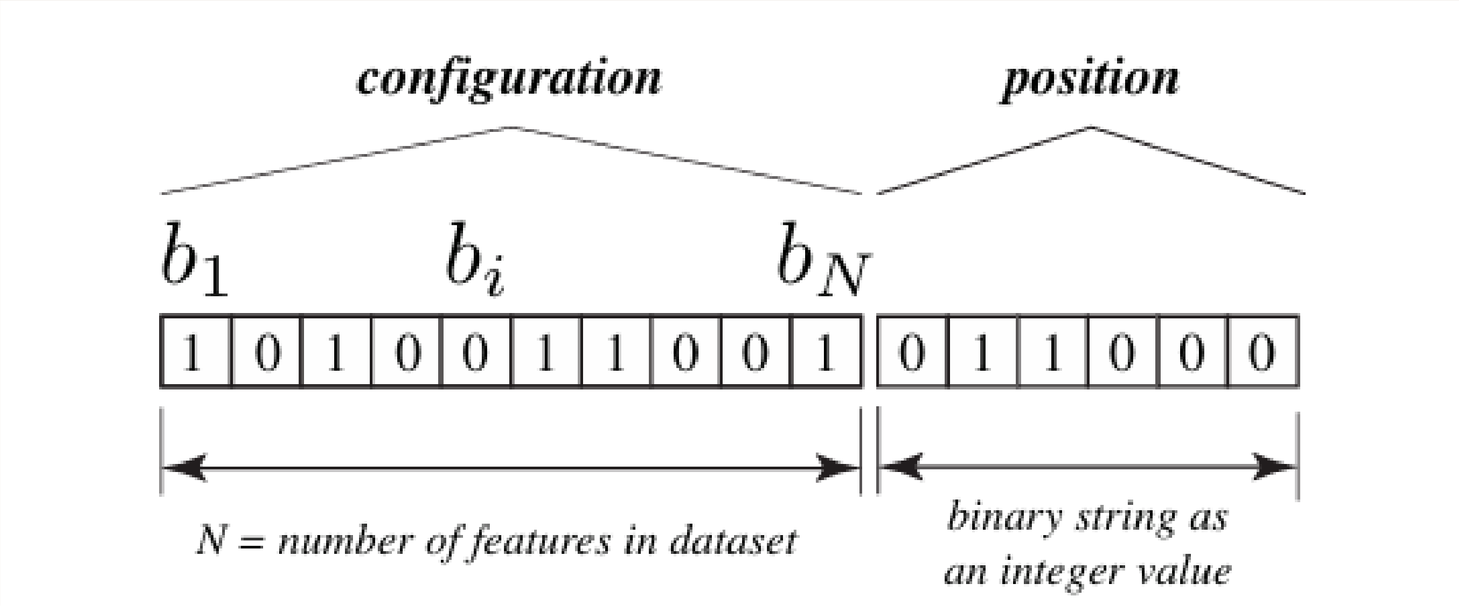
4.2

Fitness Function

4.3

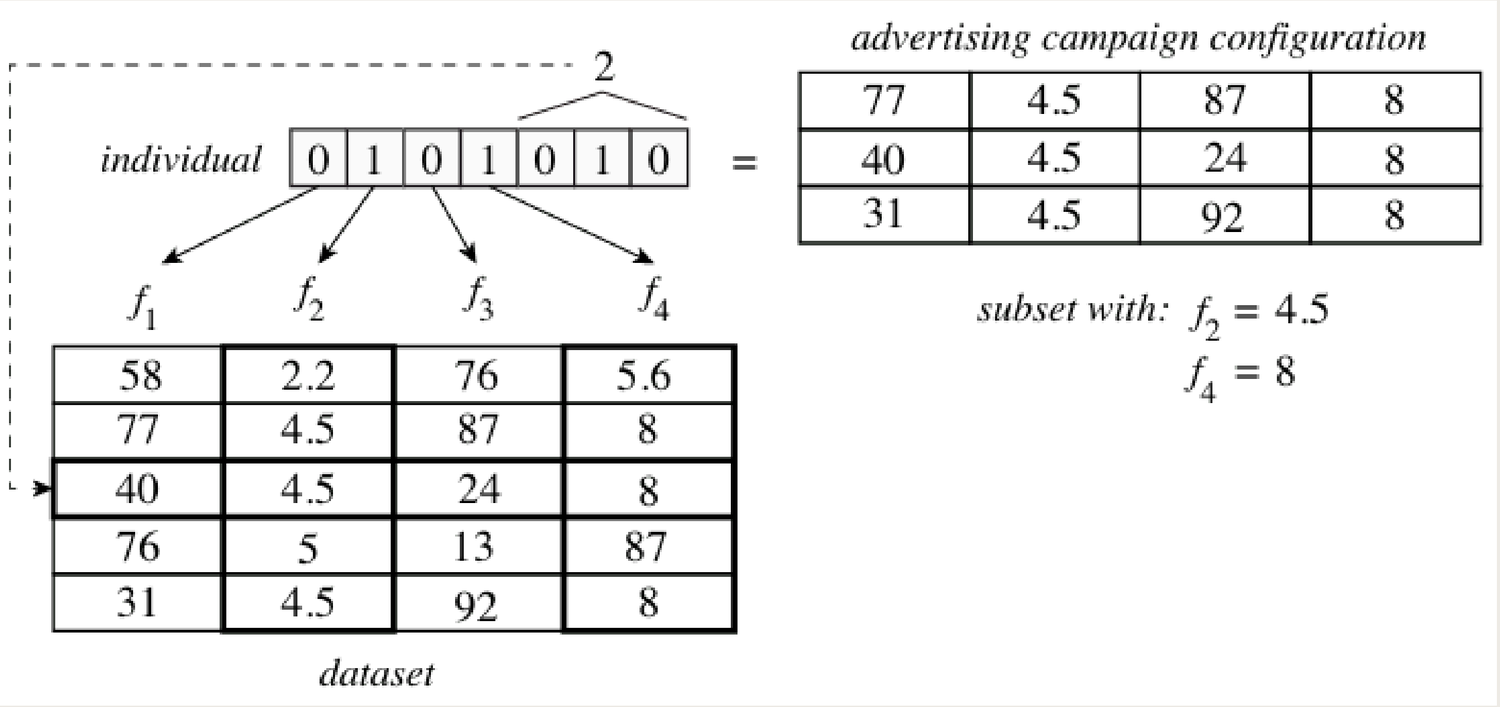
Selection, Crossover and Mutation Operators

INDIVIDUAL ENCODING



- Chromosome 1: random binary string có N bits
- Chromosome 2: là 1 đoạn mã nhị phân biểu thị vị trí của cá thể trong dữ liệu

FITNESS FUNCTION



- D': số sample trong subset
- CTR Average:

$$CTR_{average} = \frac{1}{D'} \sum_{k=1}^{D'} p_k$$

- Fitness fuction: $f(individual) = CTR_{average} \times \min(D', T)$

SELECTION, CROSSOVER AND MUTATION OPERATORS

- Selection: Linear Ranking selection (p(i) = (2 - elitism) / N + 2 (rank(i) - 1) (elitism - 1) / (N * (N - 1)))
- Crossover: Lai ghép 1 điểm 0.2
- Mutation: tỉ lệ 0.3

p_m \ p_c	**0.1**	**0.2**	**0.3**	**0.4**	**0.5**	**0.6**	**0.7**	**0.8**	**0.9**
0.1	1306.48	1293.84	1285.14	1281.03	1325.25	1297.14	1316.91	1314.43	1288.35
0.2	1343.89	1314.14	1359.57	1303.31	1316.87	1317.57	1311.78	1322.47	1277.48
0.3	1313.68	**1375.71**	1348.36	1316.86	1309.21	1311.98	1315.48	1321.02	1299.37
0.4	1313.79	1328.77	1346.24	1319.09	1314.67	1312.18	1328.36	1316.44	1289.55
0.5	1337.88	1349.90	1337.83	1338.95	1336.79	1354.18	1289.94	1316.72	1303.00
0.6	1361.53	1354.09	1317.51	1324.24	1279.99	1316.87	1289.92	1289.59	1285.07
0.7	1345.76	1325.94	1341.92	1327.09	1309.07	1332.03	1287.41	1312.90	1249.44
0.8	1312.20	1332.25	1331.23	1339.68	1320.27	1300.19	1337.20	1288.46	1261.30
0.9	1351.96	1322.49	1326.18	1304.30	1319.22	1314.88	1278.75	1266.95	1278.97

- Select random 20k samples, set 9 different values for pc and pm in the interval [0.1,0.9], iteration: 30, population size: 500, T: 20, elitism: 0.05

GENETIC ALGORITHM (PAPER + CTR MODEL)

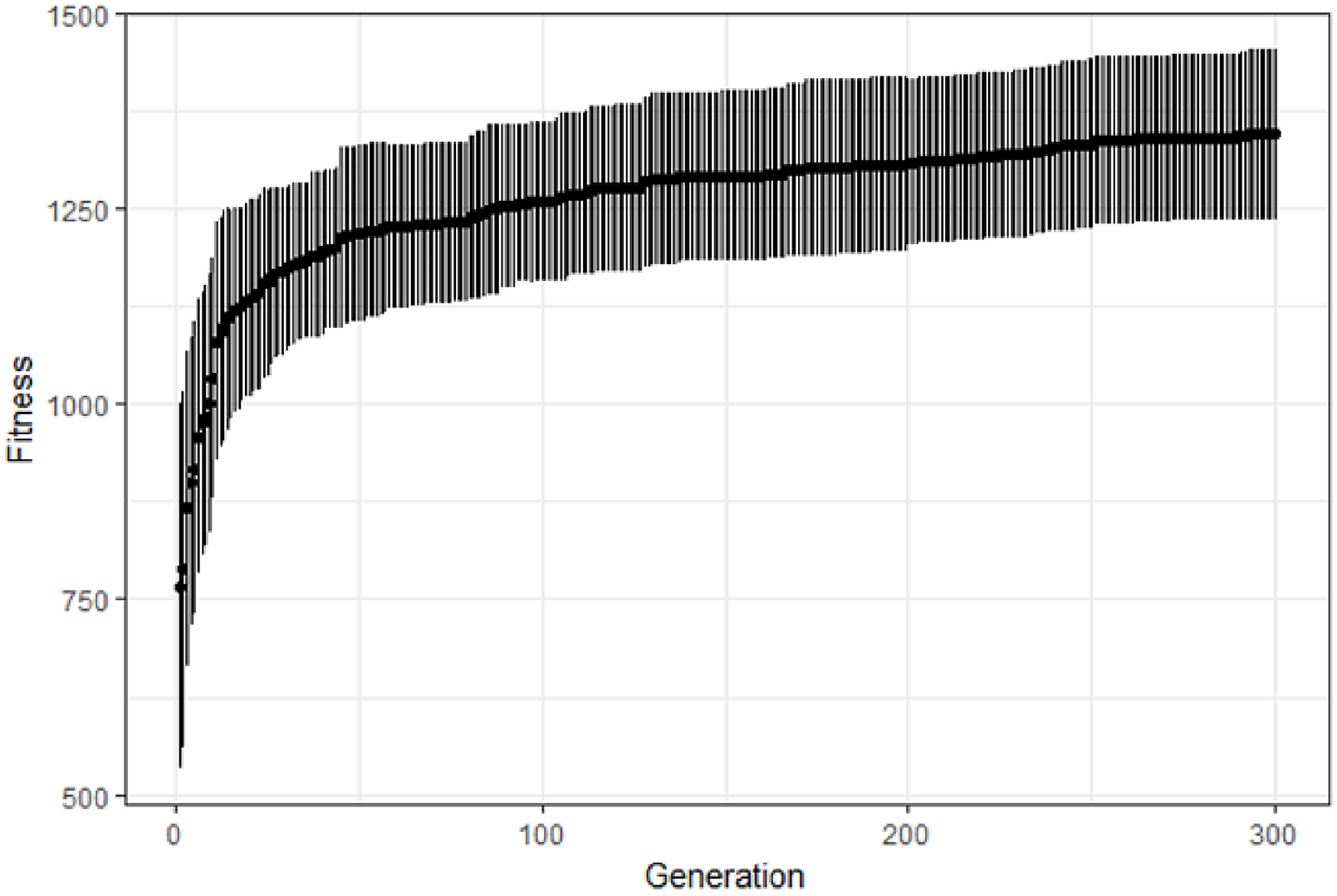
Algorithm 3 Proposed online advertising campaigns optimization.

Require: dataset, threshold T, prediction CTR model, GA parameters
Ensure: best campaign configuration

▷ Set the parameters
$|population| \leftarrow 500$, $max\_iterations \leftarrow 300$, $p_c \leftarrow 0.2$, $p_m \leftarrow 0.3$ and elitism $\leftarrow 5\%$

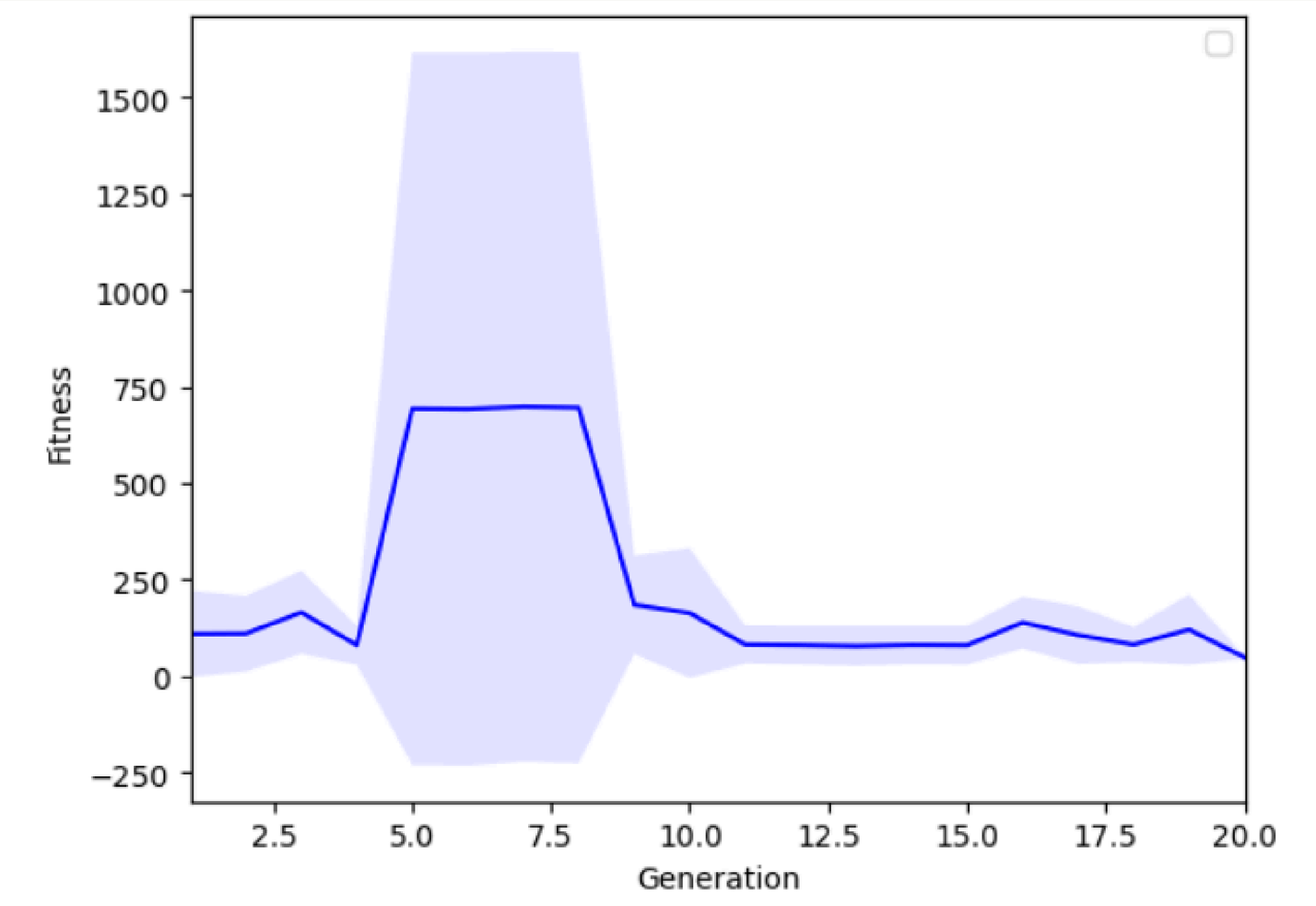
▷ GA begins
Generate an initial population
$individuals \leftarrow$ Generate allowed individuals ▷ see Figure 3
for i $\leftarrow 1 : max\_iterations$ **do**
    $individuals \leftarrow$ Verify or modify individuals to properly allow them
    for all $individuals$ **do**
        Decode the individual, and set $chromosome_1$ and $chromosome_2$
        Select the columns of dataset where values in $chromosome_1$ are "1"
        Select those samples in dataset where values are equal to those of $chromosome_2$
        $subset \leftarrow$ Generate a subset using columns and samples ▷ see Figure 4
        if rows in $subset > T$ **then**
            $subset \leftarrow RandomSelection(subset, T)$ ▷ randomly select T samples
        end if
        $CTR_{average} \leftarrow$ Compute the average CTR of $subset$ using (4)
        $D' \leftarrow size(subset)$
        $f(individual) \leftarrow CTR_{average} \times \min(D', T)$
    end for
    Select $individuals$ using linear-rank selection and 5% elitism
    Crossover pairs of $individuals$ with probability p_c using the single-point operator
    Mutate genes of $individuals$ with probability p_m using uniform random operator
    $individuals \leftarrow$ new individuals
end for
Return best individual

GENETIC ALGORITHM (PAPER + CTR MODEL)



- Num_trials: 50
- num_individuals: 500
- max_iterations: 300
- elitism: 0.05
- pc: 0.2
- pm: 0.3

GENETIC ALGORITHM (PAPER + CTR MODEL)



- Num_trials: 3
- num_individuals: 64
- max_iterations: 20
- elitism: 0.05
- pc: 0.2
- pm: 0.3

GENETIC ALGORITHM (PAPER + CTR MODEL)

No.	Fitness	No. Exp.	Iteration	No. Features	Average CTR	Real Size	Features Configuration
1	1448.52	35	97	10	0.7243	2474	4, 5, 6, 7, 11, 12, 16, 19, 20, 23
2	1448.52	11	289	11	0.7243	2474	4, 5, 6, 7, 8, 10, 11, 12, 16, 20, 23
3	1448.40	10	289	7	0.7242	2474	6, 7, 10, 13, 16, 19, 23
4	1448.16	25	275	9	0.7241	2474	6, 8, 10, 11, 13, 16, 19, 20, 23
5	1447.86	12	61	11	0.7239	2447	4, 5, 6, 7, 10, 11, 13, 16, 17, 19, 23
6	1447.85	18	296	8	0.7239	2447	6, 8, 9, 10, 12, 17, 19, 23
7	1447.81	41	204	10	0.7239	2447	3, 7, 8, 9, 10, 12, 16, 17, 19, 23
8	1447.81	47	204	10	0.7239	2447	3, 7, 8, 9, 10, 12, 16, 17, 19, 23
9	1447.74	5	226	10	0.7239	2474	5, 7, 8, 9, 10, 11, 12, 13, 19, 23
10	1447.65	24	282	9	0.7238	2474	7, 8, 9, 10, 11, 12, 13, 20, 23
11	1447.57	21	199	9	0.7238	2474	3, 7, 8, 10, 11, 12, 13, 19, 23
12	1447.52	1	258	9	0.7238	2474	4, 6, 7, 9, 10, 13, 16, 19, 23
13	1447.51	4	138	9	0.7238	2474	4, 5, 7, 8, 12, 13, 19, 20, 23
14	1447.42	20	258	6	0.7237	2474	6, 7, 10, 13, 20, 23
15	1447.38	9	264	12	0.7237	2447	4, 5, 6, 7, 8, 10, 11, 13, 17, 19, 20, 23
16	1447.26	23	129	11	0.7236	2447	4, 5, 7, 9, 12, 13, 16, 17, 19, 20, 23
17	1446.92	29	295	10	0.7235	2447	5, 6, 7, 10, 11, 12, 13, 17, 20, 23
18	1445.92	30	293	12	0.7230	2447	3, 5, 7, 8, 10, 11, 12, 16, 17, 19, 20, 23
19	1440.29	2	248	11	0.7201	2059	4, 5, 7, 8, 9, 10, 13, 16, 19, 20, 22
20	1440.18	17	157	10	0.7201	2044	4, 7, 8, 9, 10, 11, 12, 17, 20, 22

- Num_trials: 50
- num_individuals: 500
- max_iterations: 300
- elitism: 0.05
- pc: 0.2
- pm: 0.3

GENETIC ALGORITHM (PAPER + CTR MODEL)

]:

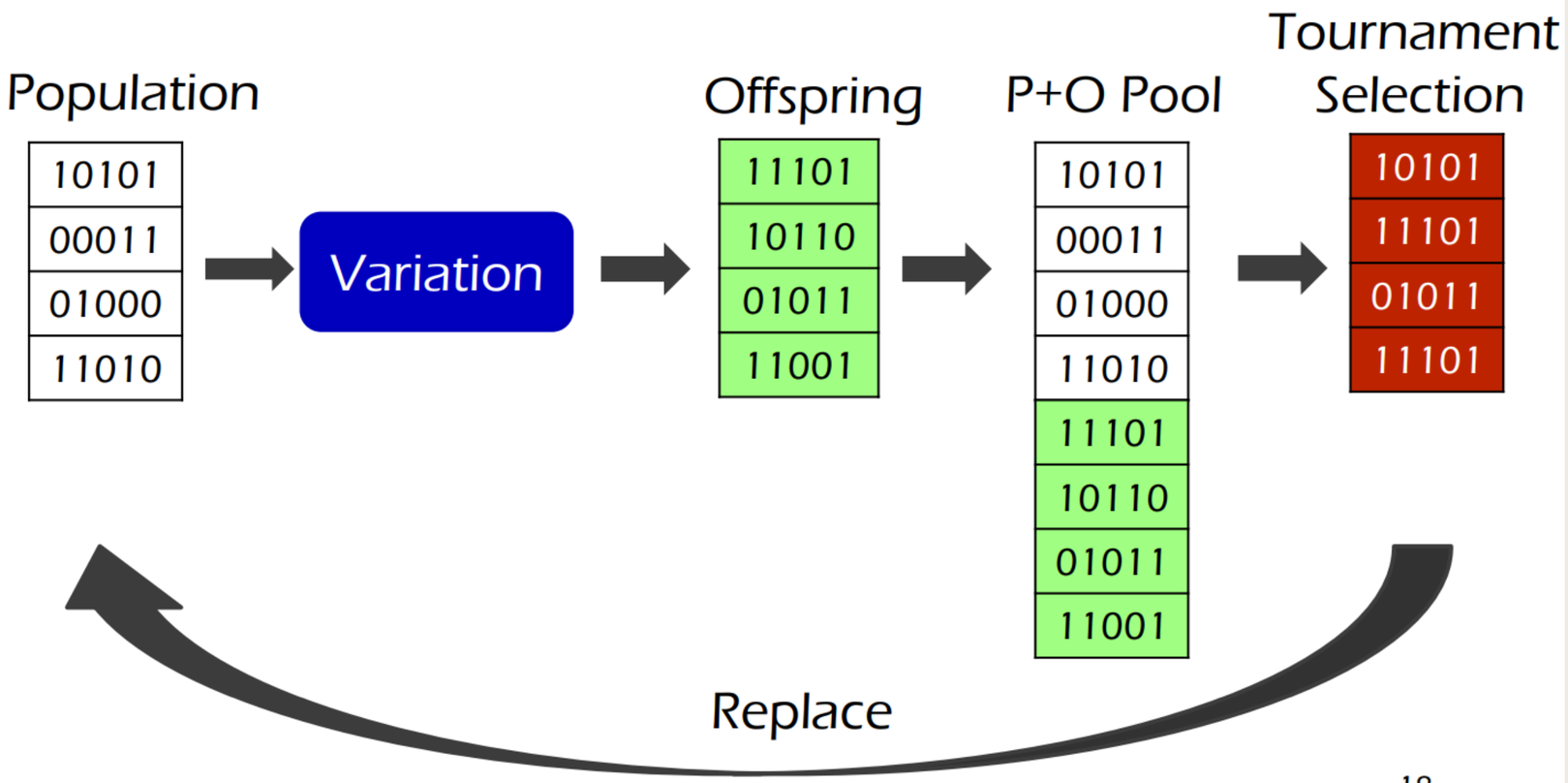
	No.	Fitness	No.Exp.	Iteration	No.Features	Average CTR	Features Configuration
4	1	2000.0	0	5	9	1.000000	[4, 5, 9, 11, 12, 15, 16, 22, 24]
5	2	2000.0	0	6	9	1.000000	[4, 5, 9, 11, 12, 15, 16, 22, 24]
6	3	2000.0	0	7	9	1.000000	[4, 5, 9, 11, 12, 15, 16, 22, 24]
7	4	2000.0	0	8	9	1.000000	[4, 5, 9, 11, 12, 15, 16, 22, 24]
9	5	400.0	0	10	9	0.997506	[11, 12, 14, 17, 18, 19, 20, 23, 24]
28	6	356.0	1	9	10	0.775599	[4, 6, 7, 10, 12, 15, 18, 21, 22, 24]
42	7	303.0	2	3	9	0.322684	[3, 7, 8, 9, 12, 15, 19, 21, 24]
0	8	267.0	0	1	7	0.133500	[4, 6, 7, 14, 19, 21, 22]
1	9	249.0	0	2	7	0.124500	[4, 6, 7, 14, 19, 21, 22]
58	10	248.0	2	19	7	0.379205	[4, 5, 7, 8, 18, 20, 22]
35	11	214.0	1	16	4	0.107000	[16, 18, 23, 24]
36	12	211.0	1	17	4	0.105500	[16, 18, 23, 24]
8	13	151.0	0	9	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
10	14	151.0	0	11	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
2	15	151.0	0	3	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
13	16	151.0	0	14	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
15	17	151.0	0	16	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
14	18	151.0	0	15	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
3	19	151.0	0	4	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]
12	20	151.0	0	13	11	0.838889	[4, 6, 7, 8, 12, 13, 15, 16, 17, 18, 20]

- Num_trials: 3
- num_individuals: 64
- max_iterations: 20
- elitism: 0.05
- pc: 0.2
- pm: 0.3

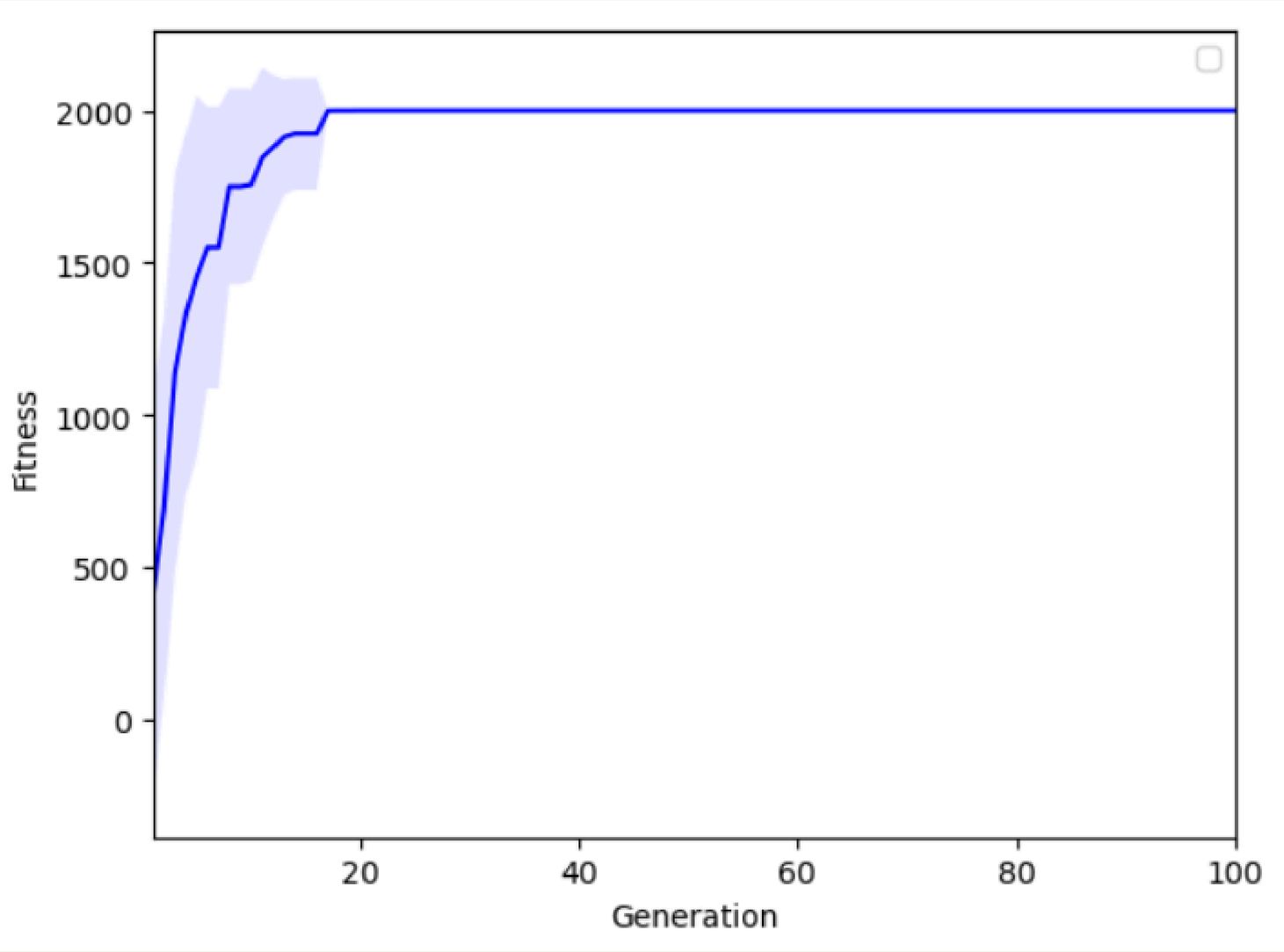
GENETIC ALGORITHM (POPOP + DNN MODEL)

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

Genetic Algorithm (POPOP)



GENETIC ALGORITHM (POPOP + DNN MODEL)



- Num_trials: 10
- num_individuals: 256
- max_iterations: 100
- pc: 0.2
- pm: 0.3

GENETIC ALGORITHM (POPOP + DNN MODEL)

	No.	Fitness	No.Exp.	Iteration	No.Features	Average CTR	Features Configuration
143	1	2000.0	1	44	5	1.0	[9, 15, 16, 19, 22]
363	2	2000.0	3	64	8	1.0	[6, 8, 9, 10, 11, 16, 21, 22]
884	3	2000.0	8	85	11	1.0	[4, 7, 9, 12, 15, 16, 20, 21, 22, 23, 24]
411	4	2000.0	4	12	8	1.0	[7, 9, 11, 14, 15, 18, 21, 22]
395	5	2000.0	3	96	9	1.0	[5, 6, 8, 9, 10, 11, 12, 15, 18]
131	6	2000.0	1	32	10	1.0	[5, 9, 10, 12, 15, 18, 19, 20, 23, 24]
933	7	2000.0	9	34	10	1.0	[4, 5, 6, 8, 9, 12, 14, 16, 18, 23]
887	8	2000.0	8	88	11	1.0	[4, 5, 7, 8, 9, 10, 11, 12, 15, 16, 21]
921	9	2000.0	9	22	8	1.0	[4, 6, 9, 10, 11, 16, 17, 18]
611	10	2000.0	6	12	6	1.0	[8, 9, 10, 11, 14, 19]
79	11	2000.0	0	80	6	1.0	[7, 8, 9, 12, 15, 18]
620	12	2000.0	6	21	10	1.0	[4, 5, 9, 10, 11, 15, 18, 19, 21, 22]
367	13	2000.0	3	68	9	1.0	[4, 6, 8, 9, 11, 18, 19, 21, 22]
626	14	2000.0	6	27	10	1.0	[4, 7, 8, 10, 15, 18, 20, 21, 22, 24]
944	15	2000.0	9	45	6	1.0	[7, 10, 11, 14, 18, 19]
822	16	2000.0	8	23	7	1.0	[7, 9, 14, 15, 16, 20, 23]
250	17	2000.0	2	51	8	1.0	[6, 8, 9, 12, 15, 16, 23, 24]
815	18	2000.0	8	16	10	1.0	[6, 7, 11, 12, 15, 17, 18, 21, 22, 24]
718	19	2000.0	7	19	10	1.0	[5, 6, 9, 11, 12, 15, 18, 19, 22, 23]
696	20	2000.0	6	97	8	1.0	[9, 11, 15, 17, 18, 19, 21, 24]

- Num_trials: 10
- num_individuals: 256
- max_iterations: 100
- pc: 0.2
- pm: 0.3

Thank you for Listening