Egyptian Informatics Journal 22 (2021) 505–513

Contents lists available at ScienceDirect

Egyptian Informatics Journal

journal homepage: www.sciencedirect.com

A comparative study of hybrid estimation distribution algorithms in solving the facility layout problem

Amalia Utamima *

Faculty of Intelligent Electrical and Informatics Technology, Institut Teknologi Sepuluh Nopember, Indonesia

ARTICLE INFO

Article history:
Received 30 June 2020
Revised 15 September 2020
Accepted 11 April 2021
Available online 30 April 2021

Keywords:
Estimation distribution algorithm
Facility layout problem
Hybrid algorithms
Safe work

ABSTRACT

The Estimation Distribution Algorithm (EDA) is an evolutionary algorithm that uses probabilistic models to create candidate solutions. Previous researchers have suggested various hybrid methods to avoid the premature convergence of EDA. This research conducts a comparative study between several variations of hybridization in EDA with regards to the descriptive statistics in the objective values.

This study also proposes a new hybrid approach, named Adapted EDA (AEDA), by adapting the structure of EDA by adding a lottery procedure, an elitism strategy, and a neighborhood search. The proposed AEDA, several hybridizations of EDA, and Genetic Algorithm (GA) plus Tabu Search (TS) are applied to the facility layout design in manufacture – Enhanced Facility Layout Problem (EFLP) – to analyze their solutions. The hybrid EDAs that are being compared are EDA plus GA (EDAGA), EDA plus Particle Swarm Optimization (EDAPSO), the combination of EDAPSO plus TS (EDAhybrid), and AEDA. The experimental results show that the AEDA can significantly improves the solution quality in solving all the EFLP instances compared to other algorithms.

1. Introduction

The Estimation Distribution Algorithm (EDA) is an evolutionary technique that reproduces new candidate solutions from the evaluation of the probabilistic distribution of the population to get a more efficient search. EDA has an issue with lack of diversity, which results in a premature convergence of its solution [1]. Previous research proposed various hybridizations of EDA with other heuristics or metaheuristics algorithms to avoid the local optima. The hybridizations of EDA were applied to solve different kinds of optimization such as supermarket location problem [2], agricultural routing planning [3], and vehicle routing problem [4]. The algorithms' results were proven to provide practical solutions to the issues given. Despite the effectiveness of the hybridizations of EDA to solve the various problems, no research compares the solution quality for different types of hybridizations of EDA. Hence this research makes a comparative study between several different hybridizations of EDA.

The single row facility layout problem (SFLP) is categorized as an NP-complete problem, and a heuristics method is needed to provide a near-optimal solution [5]. The enhanced facility layout problem (EFLP) is an extension of SFLP with the addition of installment cost, and the safety constraint [6]. The experiments in this research show that the solution to several instances of EFLP can still be improved. Therefore, the new variant of EDA is proposed to deal with the problem. This research tests the proposed algorithm to solve instances of EFLP.

The contributions of this research are the comparative study between the hybridization of EDA, the introduction of a new variant of EDA named Adapted EDA (AEDA), and the application of the algorithms to EFLP datasets. This paper is organized as follows. Previous studies are reviewed in Section 2, while the problem is described in Section 3. The methodology is followed by a description of AEDA, hybridization of EDA, and Genetic Algorithm (GA) with Tabu Search (TS) in Section 4. The experimental results and the comparative study are recorded and analyzed as well in Section 5. Finally, the discussion and conclusion of this study are drawn based on the outcome analysis in Section 6.

* Corresponding author.
E-mail address: amalia@is.its.ac.id

Peer review under responsibility of Faculty of Computers and Artificial Intelligence, Cairo University.

Production and hosting by Elsevier

https://doi.org/10.1016/j.eij.2021.04.002

2. Literature review

EDA is categorized as a stochastic optimization method that builds and samples the probabilistic models from a group of promising solutions. Metaheuristic algorithms tend to generate random candidate solutions to be processed into the new generation. In contrast, EDA uses the probabilistic models to create a group of new candidate solutions. This approach enhances the solution quality of EDA because the produced offspring are statistically built and sampled based on a probability model derived from a group of parents' best fitness. As a tool for evolutionary computation, EDA becomes a promising algorithm that can competitively solve the optimization problem [1].

EDA, first introduced by Baluja in [7], formerly contained a single probabilistic model. Current research hypothesizes that the effectiveness of EDA depends primarily on the accuracy of the probabilistic model to extract the information in the population. Therefore, the multivariate probabilistic model becomes an option to build a better probabilistic model [8]. This reasoning determined the use of two probabilistic models in EDA, the ordinal and the dependency, in this research. A detailed explanation is provided in Section 4.1.1.

As with other metaheuristic algorithms, EDA appears to be trapped in local optima when dealing with a large scale problem [9]. Therefore, EDA is hybridized with several heuristics and metaheuristics algorithms to improve the solution. Among all, there were Particle Swarm Optimization (PSO) [10], Differential Evolution (DE) [2], GA [11], and Bayesian Optimization Algorithm (BOA) [4]. The specific combination of EDA with a mutation operator, demonstrated in [12], can improve the searchability of the algorithm.

The supermarket location problem was solved with the hybridization of EDA and DE [2]. The hybrid EDA and PSO was utilized for solving the reservoir flood control operation [10]. Meanwhile, the combination of EDA and GA was used to deal with the scheduling problem [2]. The combination of EDA and BOA was used to solve vehicle routing problem [4].

The objective of SFLP is to minimize the sum of the products' flow cost based on the distance between facilities. The SFLP is found in many real-world problems, such as room arrangements, department arrangements in offices, and machine arrangements in the manufacture [13]. A constrained SFLP that considers a few facilities need to be placed in certain positions with/without allowing the arrangement of any other facility in between two ordered facilities is described in [5,14]. A permutation-based GA (pGA) was proposed by [5] to solve the constrained SFLP. Recently, the work in [14] demonstrated that a fireworks algorithm performs better than the pGA for solving the constrained SFLP.

SFLP that consider material handling cost and rearrangement cost of the departments at the beginning of each period was proposed in [15]. The authors used Simulated Annealing (SA) and GA and their hybridizations to deal with the problem. The results showed that SA demonstrated better performance comparing to other algorithms [15]. Meanwhile, DE and its variant were explored by [16] to solve SFLP. GA and a neighborhood search were also proposed in the previous literature to deal with the SFLP [17,18].

To summarize, the following research gaps can be stated, as indicated earlier in the Introduction section. First, there is no research that compares the solution quality of the hybridization of EDA. Hence, more research is essential to direct the development of EDA. Second, in recent years, there have been no studies that utilize EDA to deal with SFLP and EFLP. Therefore, the comparative study in this problem is a potential area for further research.

3. Problem description

EFLP considers the installation cost and inserts the safety constraint. The installation cost is related to the fixed cost to assign a facility to a specific location [19]. Because of the variation of a facility's installment cost in every position, then besides flow matrices, EFLP also adds installation cost matrices. Meanwhile, the safety constraint, which is also called technological constraint in [20], prohibits two specified facilities from being placed directly adjacent to each other [5].

A description of the variables in the EFLP's mathematical model is given in Table 1. The EFLP's objective function, adapted from [7], is stated in Eq. (1), which minimizes the sum of both the flow's distance between every facility and the installation cost of every facility. This research generalizes the formula by adding three decision variables (Eqs. (2)–(4). The decision variables x_{ij}, y_{ik}, and a_{ij} indicate the adjacency of two facilities, the position of facilities in locations, and the safety constraint, respectively. Eq. (5) specifies the distance of two adjacent facilities and considers their length and clearance space. Eqs. (6) and (7) makes sure the flow loads and installation costs are greater than 0. Finally, Eq. (8) limits that all decision variables are a binary number.

$$Z = \min\left(\sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(x_{ij}W_{ij}d_{ij} + a_{ij}P) + \sum_{k=1}^{n}\sum_{i=1}^{n}y_{ik}C_{ik}\right) \tag{1}$$

$$x_{ij} = \begin{cases} 1, & \text{if facility } i \text{ is put next to } j \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

$$y_{ik} = \begin{cases} 1, & \text{if facility } i \text{ is assigned to location } k \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

$$a_{ij} = \begin{cases} 1, & \text{if facility } i \text{ cannot be put adjacent to } j \\ 0, & \text{otherwise} \end{cases} \tag{4}$$

s.t.:

$$d_{ij} \geqslant \frac{l_i + l_j}{2} + s_{ij}; d_{ij}, s_{ij} \geqslant 0; l_i, l_j > 0 \tag{5}$$

$$W_{ij} > 0; i, j = 1, 2, \ldots, n \tag{6}$$

$$C_{ik} > 0; i, j, k = 1, 2, \ldots, n \tag{7}$$

$$x_{ij} \in \{0, 1\}; y_{ik} \in \{0, 1\}; p_{ij} \in \{0, 1\} \tag{8}$$

4. Methods

4.1. The proposed algorithm: AEDA

This study suggests a new hybrid EDA, the Adapted Estimation Distribution Algorithm (AEDA). The structure of EDA is modified by

Table 1
The variable description in the EFLP mathematical model.

Variable	Descriptions
x_{ij}	A decision variable, equal to 1 if facility i is put adjacent to j; otherwise it equals to 0
y_{ik}	A decision variable, equal to 1 if facility i is put into location k; otherwise it equals to 0
a_{ij}	A decision variable, equals to 1 if facility i cannot be placed adjacent to facility j; otherwise it equals to 0
n	Number of facilities
k	Location of a facility ($k = 1, 2, \ldots, n$)
i,j	Facility indices ($i, j = 1, 2, \ldots, n$)
l_i	Length of facility i
W_{ij}	Flow loads between facility i and facility j
C_{ik}	Installation cost of assigning facility i to location k
d_{ij}	Centers' distance of facility i and facility j
s_{ij}	Clearance space between the two facilities
P	Penalty cost of violating safety constraint

adding a lottery part, a roulette wheel selection, an elitism strategy, and TS. These additional components in AEDA are used to maintain the self-adaptation of the evolutionary algorithm [21]. The main phases of AEDA are selection, probabilistic model calculation, sampling, and neighborhood search. The pseudocode of AEDA is detailed in Algorithm 1, while a description of the variables is given in Table 2.

At first, a group of candidate solutions is initialized randomly. The structure of these candidate solutions is constructed using the permutation-based variable. The cost of every candidate solution is then calculated. The iteration begins and the main phases of AEDA (described in the following subsections) are executed.

4.1.1. Selection and probabilistic model calculation phases

The iteration of AEDA starts with the selection phase, which utilizes truncation selection. Truncation selection, significantly improve the model accuracy [22], selects a set of candidate solutions that have better fitness compared to other solutions. The result of the selection phase (Algorithm 1, line 4) is recorded in S.

The following step in AEDA calculates the probabilistic models (Algorithm 1, lines 5–6). Two probabilistic models are used in this stage, ordinal and dependent. Eq. (9) is used to calculate the probability of assigning a facility to a location. In Eq. (9), P_ik is the probability of installing facility i at position k (where $k = 2, 3, \ldots, n; i, j = 1, 2, \ldots, n$). The ordinal probability model (φ_{ik}) represents the significance of a facility situated in a location, while the dependence probability model (ψ_{ij}) refers to the importance of a facility placed next to another facility.

$$P_{ik} = \frac{\varphi_{ik} + \psi_{ij}}{\sum \varphi_{ik} + \psi_{ij}} \tag{9}$$

4.1.2. Sampling phase

The sampling phase is shown in Algorithm 1, lines 7–21. For each candidate solution, the sampling step begins with the placing of a facility on the first location (Algorithm 1, lines 9–13). This research proposes a lottery procedure to maintain the diversity of the population. Based on the previous study from [11], one of the methods to keep the diverseness is by adding randomness in a part of EDA. This lottery procedure selects either the first facility from the previous generation (from S) or a random facility to be placed in the first location. This research experimented in running AEDA with several combinations of *LotteryRate* and found that the best setting to be 5%.

The roulette wheel selection is then utilized to assign facilities into the second location until the n position is reached (Algorithm 1, lines 14–20). The probability of setting a facility to a location (calculated in Eq. (9)) is recorded in every position. Next, the cumulative probability is calculated, and a uniformly distributed random number is generated [31]. Using the roulette wheel concept, facilities are put into locations. Each time a facility is placed in a location, it is added to F and Ω will remove it from the unassigned facilities. The cost of each candidate solution is calculated, and *cBest* is updated. TS (Algorithm 1, lines 23–25) is described in the following subsection.

Table 2
Description of the variables used in the pseudocode of AEDA.

Variable	Descriptions
S	A group of selected candidate solutions
n	Number of facilities
cBest	Current best solution
gBest	Global best solution through generations so far
F	Set of assigned facilities
P	Probability of assigning facilities to locations
Ω	Set of unassigned facilities
θ	A random number
i	A selected facility by roulette wheel selection
k	The index of a facility's position

4.1.3. Neighborhood search phase

AEDA first checks whether *cBest* is better than *gBest*. Neighborhood search is performed if there is no improvement in *gBest* in the current generation [18]. If *cBest* in the current generation cannot improve on *gBest*, then a neighborhood search based on TS is run. Otherwise, AEDA will go directly to the Elitism procedure and progress to the next generation. The ability of AEDA to choose between running TS or moving on to the next iteration saves the computation time. If *gBest* is still better than *cBest*, then TS will be operated; otherwise, *gBest* is set equal to *cBest*. The TS procedure is performed for some generations (*tabuGen*).

The *swaplist*, constructed in every *tabuGen*, contains a set of the move (the swap of two facilities) and the associated costs (*tcost*). Next, TS checks if a move is taboo and if the cost of every move (*tcost*) worse than tabuSol. A penalty charge is given if a move matches these conditions. TS then updates the tabooList and tabuSol. At the end of a generation, tabuSol will contain the best solution from TS. The TS phase ends with the replacement of *gBest* with tabuSol if the better solution is found in tabuSol. Finally, the elitism procedure, which records 10% of the best solution through the generations, is performed at the end of every generation of AEDA.

Algorithm 1 AEDA

```
1: Initialize variables and set parameters
2: for each generation do
3:     Calculate cost, cBest, and gBest
4:     S ← Selection()
5:     Compute Ordinal probabilistic model (S)
6:     Compute Dependency probabilistic model (S)
7:     for each candidate solution do
8:         F ← ∅
9:         if j < lotteryRate then
10:            F(1) ← Lottery()
11:        else
12:            F(1) ← RandSe1ect (S(1))
13:        end if
14:        for k=2 until n do
15:            cumP ← Calculate cumulative probability of P
16:            θ ← U(0, 1)
17:            Roulette Wheel Selection (θ, cumP)
18:            F(k) ← i
19:            Ω ← Ω \ i
20:        end for
21:    end for
22:    Calculate cost; Update cBest
23:    if cBest > gBest then
24:        TabuSearch (cBest)
25:    end if
26:    Update gBest
27:    Elitism()
28: end for
```

4.2. The hybridization of EDA

This research re-coded several hybridizations of EDA based on explanation in previous literature. Those combinations are EDA plus GA (EDAGA), EDA plus PSO (EDAPSO), and EDAPSO plus TS (EDAhybrid).

4.2.1. EDAGA

The concept of EDAGA, firstly introduced by Chen et al. (2012) [23,24], was to combine GA into the structure of EDA. GA ensures the group of candidates remains diverse, while EDA exploits the candidate to achieve a good solution. EDAGA has been used for the scheduling problem [23], no-idle flow shop scheduling problem [24]. Algorithm 2 shows the basic pseudocode of EDAGA.

The general concept is to run EDA and GA alternately until a stopping criterion (i.e., the maximum number of generations) is met. The EDA procedure (Algorithm 2, lines 4–6), is run on every even generation. A probabilistic model is developed based on the selected chromosomes. The sampling procedure generates new chromosomes based on the probabilistic model.

The GA approach (Algorithm 2, lines 7–9) is executed when the generation is odd. The two-point crossover procedure is applied in the selected chromosomes. Meanwhile, mutations will occur randomly at several points in a chromosome. The crossover and mutation procedures are applied to the chromosomes using crossover rate of 0.7 and a mutation rate of 0.3 [6]. The process continues with the evaluation of the fitness value of every chromosome. The best solution and population updating will then follow.

4.2.2. EDAhybrid

EDAhybrid is a combination of EDA, PSO, and TS. Generally, EDA and PSO are run alternately to maintain the diverseness in the EDAhybrid, and TS concept is then added. This algorithm is used to solve the facility layout problem [6].

Algorithm 3 shows the pseudocode of EDAhybrid. The EDA procedure (Algorithm 3, lines 4–7) employs two probabilistic models to sampling (generating) new solutions. The PSO procedure (Algorithm 3, lines 8–9) calculates the velocity of each particle (candidate solution) in the swarm (population). Particle updating is then performed.

EDAhybrid continues with the fitness value calculation of every particle. Particle best (*pBest*), that is, the current best particle found in the swarm and *gBest* (the best particle found so far through generation) are then updated. TS (Algorithm 3, lines 12) will exploits *gBest* to find whether an improvement can be made. If the solution from TS is better than *gBest*, it will replace *gBest*. The elitism procedure is applied at the end of every iteration.

4.2.3. EDAPSO

EDAPSO algorithm is a combination of EDA plus PSO. Algorithm 4 shows the pseudocode of EDAPSO. The hybridization of EDA and PSO also can be found in the reservoir flood control operation [10].

The concept of this EDAPSO is embedding PSO inside the EDA to preserve the diversity of the algorithm. The difference between EDAPSO and EDAhybrid is the existence of a local search and elitism. EDAPSO does not use a local search and elitism in its iteration. The EDA part is run every even generation (Algorithm 4, lines 4–6), while PSO is executed every odd generation (Algorithm 4, lines 7–8). Then, *pBest* and *gBest* will be updated and the new particles is sent to the next generation.

Algorithm 2 EDAGA

```
1: Initialize variables and set parameters
2: for each generation g do
3:     Compute fitness in population
4:     if g is even then
5:         Select P; Calculate a probabilistic model(P)
6:         Sampling()
7:     else
8:         Select P; Crossover(P)
9:         Mutation()
10:    end if
11:    Update fitness, population and gbest
12: end for
```

Algorithm 3 EDAhybrid

```
1: Initialize variables and set parameters
2: for each generation g do
3:     Calculate fitness, pBest, and gBest
4:     if g is even then
5:         Select S; Calculate ordinal probabilistic model(S)
6:         Calculate dependence probabilistic model(S)
7:         Sampling()
8:     else
9:         Calculate velocity(); Update population()
10:    end if
11:    Update fitness, pbest
12:    gbest ← TabuSearch (pBest)
13:    Elitism()
14: end for
```

Algorithm 4 EDAPSO

```
1: Initialize variables and set parameters
2: for each generation g do
3:     Compute fitness, pBest, and gBest
4:     if g is even then
5:         Select S; Calculate probabilistic model(S)
6:         Sampling()
7:     else
8:         Calculate velocity(); Update swarm()
9:     end If
10:    Update pbest and gbest
11: end for
```

4.3. Genetic algorithm with Tabu search

GA, as well as its hybridization with other heuristic algorithms, is often found in the literature [25]. A hybrid GA with a deconstruction and reconstruction solution for a facility layout problem was demonstrated in [26]. A modified rotation operator in GA was used for static facility layout problems [27]. Among others, a hybridization of GA and TS was proven as a robust algorithm to solve combinatoric problems such as job shop scheduling [28] and vehicle routing [29]. However, in the SFLP area, the combination of GA and TS as a method is hardly found. Therefore, this research establishes the GA plus TS (GATS) to be benchmarked against AEDA. The concept of GATS is to add TS as a local search. Algorithm 5 presents the GATS procedure.

As shown in Algorithm 5, the generation starts with the fitness calculation of every chromosome in the population. The parents are then selected via roulette wheel selection. The crossover (using two-point operator) and mutation procedures are applied to the chromosomes. After that, the replacement step will renew the chromosomes with their offspring. TS part of GATS is executed if there is no improvement in the *gbest*. The elitism procedure is performed at the end of every generation.

Algorithm 5 GATS

```
Initialize variables and set parameters
for each generation do
    Calculate fitness in population
    P ← RouletteWheelSelection()
    Crossover(P)
    Mutation()
    Replacement(); Update fitness, cbest
    gbest ← TabuSearch (cBest)
    Elitism()
end for
```

5. Experimental results

This study utilizes Matlab in a PC with an Intel i5 processor and 12 GB RAM to develop all algorithms. The parameters of these algorithms are similar in that the bigger the problem size, the more iterations needed. This section lists the results of the implementation of all algorithms along with an analysis of the results.

5.1. Testing

In the testing phase, all algorithms are coded to solve 15 benchmarked datasets in SFLP. Every algorithm is run 10 times. Table 3 shows the minimum value generated by our proposed algorithm (AEDA), along with those of EDAGA, EDAhybrid, EDAPSO, and GATS. The bold value indicates the minimum value in every row. The different name of the problem code means the different specification of the problem. The last row is the gap (represent the difference) of the achieved solutions of each algorithm compare to the optimum solution (from literature [17,30]) found so far.

Both EDAGA and EDAPSO achieve optimum solutions in 9 out of the 15 problems. However, EDAGA's solutions has the biggest gap compare other algorithms. Meanwhile, GATS achieves the minimum solutions in 13 out of 15 instances with the smaller gap than EDAGA and EDAPSO. AEDA achieves the optimum solution in all 15 problems listed in Table 3 that make a 0% gap and the smallest (best objective value) compare to other algorithms. In 13 of the problems (LW5 to P30), AEDA achieves the same optimum solution as EDAhybrid, while in A30, AEDA outperforms EDAhybrid.

5.2. Solving EFLP

The EFLP data consists of seven instances varying from 5 to 30 facilities. Three of them (the problems with 5, 11, and 20 facilities) are taken from previous literature [6]. This research generated the other four instances, those are the problems with 8, 15, 17, and 30 facilities. This research compares the minimum results, maximum value, mean, standard deviation, and the running time of AEDA, EDAGA, EDAhybrid, EDAPSO, and GATS in 100 replications. Table 4 compares the minimum value and maximum result of the algorithms in solving EFLP, while Table 5 lists the mean and the standard deviation.

The results in Table 4 shows that the AEDA and GATS successfully achieve the minimum (optimum) solution in all problem instances, while the EDAhybrid gets the minimum solution from problem five until Problem 20. AEDA and EDAhybrid achieve the best maximum value on 5 and 4 problems, respectively. However, GATS only gets the best maximum value in 2 problems. Meanwhile, both EDAGA and EDAPSO only can get the minimum solution in 3 problems.

The performance of all algorithms to solve EFLP is also shown in Table 5. The percentage error in the last row in Table 5 represents the gap between the mean value in 100 runs with the lowest minimum value of all algorithms. AEDA achieves the smallest mean in 6 problems, while EDAhybrid gets the smallest mean in 1 problem. AEDA's standard deviations are the smallest in 4 problems, while EDAhybrid's standard deviations are the smallest in 3 problems. Meanwhile, none of EDAGA, EDAPSO, and GATS can get the smallest mean and standard deviation. In terms of percentage error, AEDA obtains the first place (lowest value with 1.12%) compare to other algorithms, while EDAhybrid and GATS come in second and third place, respectively. EDAPSO's percentage error is almost three times higher than AEDA, while that in EDAGA is more than five times higher than AEDA.

Fig. 1 shows the boxplots of all algorithms' solutions for small-size problem (8), medium-size problem (15), and large-size problem (30). The boxplot of each algorithm in Problem 8 is almost the same, while the boxplots of Problems 15 and 30 show that AEDA, EDAhybrid, and GATS are better than EDAPSO and EDAGA. In order to test whether the means of each algorithm are significantly different, the ANOVA test is then applied. Since both of the boxplots of Problems 15 and 30 are almost similar, the ANOVA tests are performed to Problems 8 and 30. Tables 6 and 7 show the ANOVA tables of Problems 8 and 30, respectively. The p-values in Tables 6 and 7 indicate that ANOVA rejects the null hypothesis that all algorithms means are equal. The post hoc test using a multiple

Table 3
The comparison of the minimum results of all algorithms (in SFLP).

Case	Minimum (Objective) Value				
	EDAGA	EDAhybrid	EDAPSO	GATS	**AEDA**
S4	**638.0**	**638.0**	**638.0**	**638.0**	**638.0**
LW5	**151.0**	**151.0**	**151.0**	**151.0**	**151.0**
N6	**2.0**	**2.0**	**2.0**	**2.0**	**2.0**
S8	**801.0**	**801.0**	**801.0**	**801.0**	**801.0**
S8H	**2324.5**	**2324.5**	**2324.5**	**2324.5**	**2324.5**
S9	**2469.5**	**2469.5**	**2469.5**	**2469.5**	**2469.5**
S10	**2781.5**	**2781.5**	**2781.5**	**2781.5**	**2781.5**
S11	**6933.5**	**6933.5**	**6933.5**	**6933.5**	**6933.5**
LW11	**6933.5**	**6933.5**	**6933.5**	**6933.5**	**6933.5**
N12	23.550	**23.365**	23.550	**23.365**	**23.365**
P15	6335.0	**6305.0**	6320.0	**6305.0**	**6305.0**
P20	15849.0	**15549.0**	15797.0	**15549.0**	**15549.0**
P25	4699.0	**4618.0**	4668.0	**4618.0**	**4618.0**
P30	46985.0	**44965.0**	45577.0	44985.0	**44965.0**
A30	47444.0	45454.0	46002.0	45544.0	**45449.0**
Gap	0.922%	0.001%	0.419%	0.017%	**0.000%**

Table 4
Comparison of the minimum and maximum value for all algorithms (in EFLP).

Problem	EDAGA		EDAhybrid		EDAPSO		GATS		AEDA	
	Min	Max	Min	Max	Min	Max	Min	Max	Min	Max
5	**410.39**	**425.39**	**410.39**	**425.39**	**410.39**	440.45	**410.39**	**425.39**	**410.39**	**425.39**
8	**823.12**	856.07	**823.12**	835.12	**823.12**	873.14	**823.12**	**836.03**	**823.12**	**836.03**
11	**9398.10**	9902.40	**9398.10**	9801.85	**9398.10**	9916.35	**9398.10**	9846.50	**9398.10**	**9793.85**
15	5805.00	6457.00	**5700.00**	6375.00	5708.00	6388.00	**5700.00**	6443.00	**5700.00**	**6317.00**
17	8294.00	9197.00	**8074.00**	**8550.00**	8092.00	9029.00	**8074.00**	8573.00	**8074.00**	8597.00
20	61401.9	65937.2	**60334.6**	**61783.9**	60639.5	65257.0	**60334.6**	62122.7	**60334.6**	**61783.9**
30	24970.70	29307.70	23694.20	**25836.70**	24138.10	28852.40	**23606.10**	26406.30	**23606.10**	25995.40

Table 5
The comparison of the mean and standard deviation of all algorithms (in EFLP).

Problem	EDAGA		EDAhybrid		EDAPSO		GATS		**AEDA**	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
5	412.94	5.66	**412.79**	**5.53**	414.57	8.21	413.69	6.24	414.29	6.61
8	830.82	5.87	829.14	**4.60**	833.10	8.99	828.86	5.09	**828.772**	4.97
11	9512.58	93.87	9480.26	119.75	9519.65	131.21	9517.30	149.88	**9458.354**	**105.93**
15	6123.52	131.65	5790.49	151.51	5772.14	134.61	5784.16	152.53	**5771.78**	**122.72**
17	8755.08	184.73	8201.46	132.13	8394.21	167.02	8230.53	149.42	**8200.65**	**120.26**
20	63867.88	933.66	60816.19	387.29	61983.28	857.62	60840.50	396.28	**60739.43**	**336.85**
30	27282.48	841.87	24261.73	**607.09**	26105.37	1268.55	24140.78	633.50	**24092.755**	623.39
Percentage error	5.69%		1.28%		3.12%		1.33%		**1.12%**	

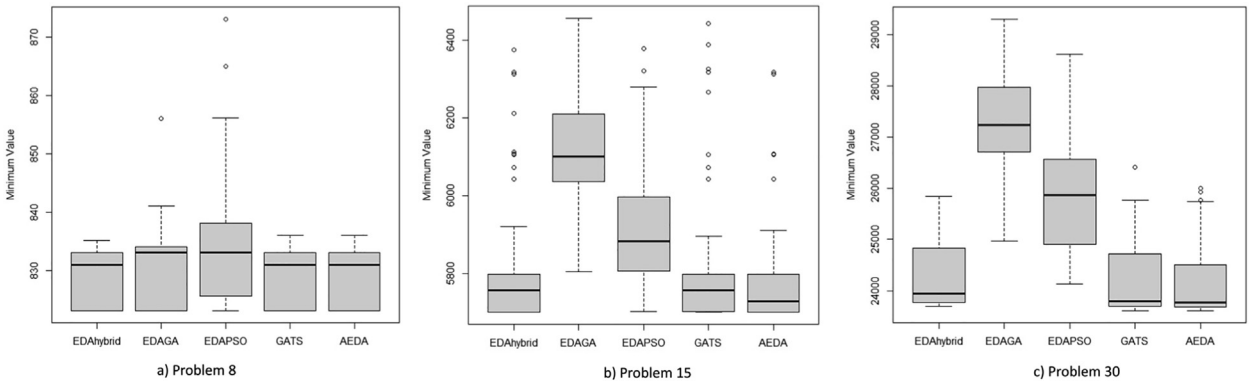

Fig. 1. The boxplot of all algorithms' solutions in (a) Problem 8, (b) Problem 15, and (c) Problem 30.

Table 6
ANOVA result of Problem 8.

Source	SS	df	MS	F	p-value
Columns	1374.1	4	343.522	9.18	$3.69x10^{-7}$
Error	18526.9	495	37.428		
Total	19901	499			

Table 7
ANOVA result of Problem 30.

Source	SS	df	MS	F	p-value
Columns	$8.38x10^{8}$	4	$209.6x10^{6}$	301.38	$3.9x10^{-131}$
Error	$3.44x10^{8}$	495	$0.7x10^{6}$		
Total	$1.18x10^{8}$	499			

comparison test (Dunnett test) is then used to determine which algorithms means are different from others. In the Dunnett test, two group means are significantly different if their intervals are disjoint; they are not significantly different if their intervals overlap.

Fig. 2 shows the Dunnet tests of (a) Problem 8 and (b) Problem 30. Fig. 2 (a) describes that in Problem 8, EDAhybrid, GATS, and AEDA have means significantly different from EDAPSO; no algorithms have means significantly different from EDAGA; and the means of EDAhybrid, EDAGA, GATS, and AEDA are not significantly

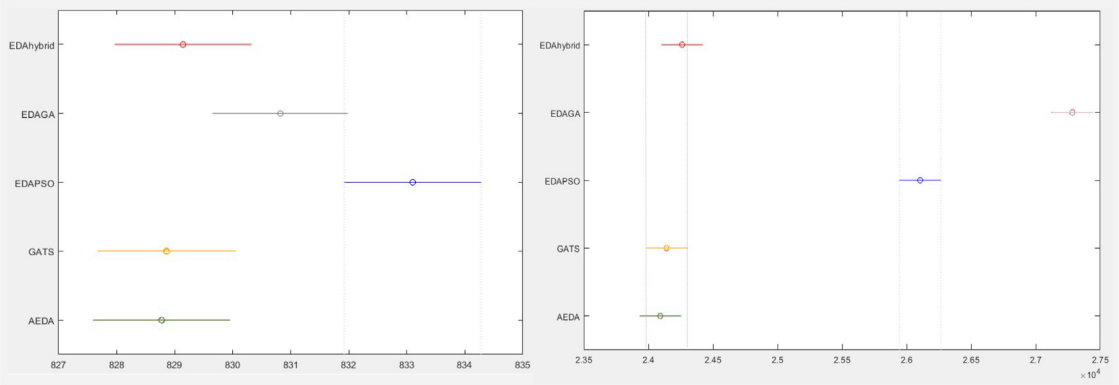

Fig. 2. The Dunnet test of (a) Problem 8 and (b) Problem 30.

Table 8
The comparison of the mean and standard deviation of the all algorithms' running time.

Problem	EDAhybrid		EDAGA		EDAPSO		GATS		**AEDA**	
	Mean	Std	Mean	Std	Mean	Std	Mean	Std	Mean	Std
5	0.0213	0.0048	0.0096	**0.0003**	0.0110	0.0006	0.0456	0.0012	**0.0094**	0.0051
8	0.0548	0.0012	0.0451	0.0008	0.0428	**0.0005**	0.0566	0.0011	**0.0378**	0.0009
11	0.2232	0.0093	0.1745	0.0090	0.1624	**0.0052**	0.1670	0.0102	**0.1220**	0.0072
15	0.5961	0.0263	0.5055	0.0160	0.3852	0.0129	0.4975	0.0289	**0.3840**	**0.0108**
17	1.0940	0.0270	0.6548	0.0200	**0.6187**	**0.0130**	0.7936	0.0260	0.6224	0.0170
20	2.1758	0.0483	1.1190	0.0311	**1.0495**	**0.0207**	1.5079	0.0307	1.2510	0.0389
30	14.2640	0.2630	4.6006	0.1317	**4.4065**	0.1663	10.2325	0.3598	8.2888	**0.1533**

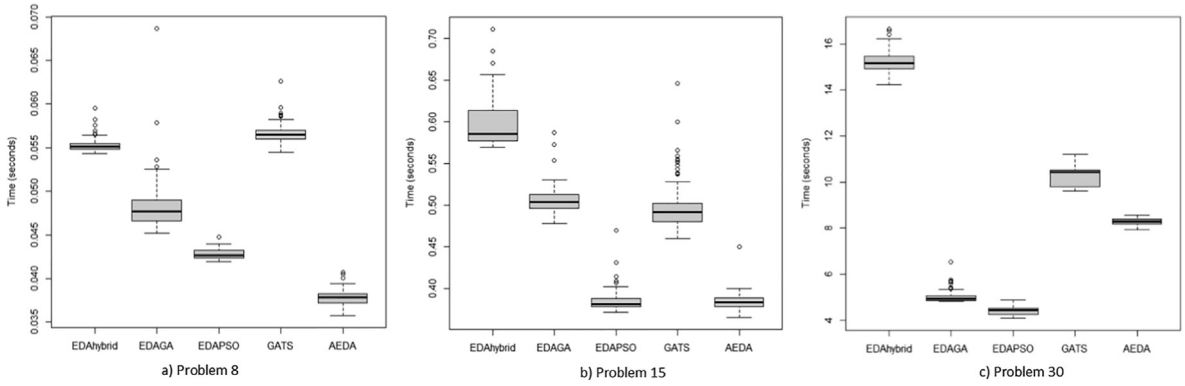

Fig. 3. The boxplot of all algorithms' running time in (a) Problem 8, (b) Problem 15, and (c) Problem 30.

different. On the other hand, Fig. 2 (b) shows that in Problem 30, EDAhybrid, GATS, and AEDA have means significantly different from EDAGA and EDAPSO; the means of EDAhybrid, GATS, and AEDA are not significantly different; and the means of EDAGA and EDAPSO are significantly different.

Table 8 provides the mean and standard deviation of the running time (in seconds) of all the algorithms. AEDA achieves the smallest mean of running time on 4 out of the 7 problems, while EDAPSO gets the smallest mean of running time in the rest of 3 problems. The standard deviation of EDAPSO's running time is the smallest in 4 instances, while those in AEDA and EDAGA are the smallest in 2 problems and 1 problem, respectively. The GATS's mean and standard deviation on its running time is slower compare to AEDA and EDAPSO. Meanwhile, EDAhybrid's running time gets the longest mean and standard deviation compare to others.

Fig. 3 illustrates the boxplots of all algorithms' running time in small-size problem (8), medium-size problem (15), and large-size problem (30). The boxplots in Fig. 3 (a) Problem 8 and (c) Problem 30 show that the box area of every algorithm is not intersected with each other; therefore, it is likely that their means are significantly different. Hence, the ANOVA test is applied only to Problem 15 and the p-value is $2.93x10^{-301}$. The Dunnett test of the running time in Problem 15 is shown in Fig. 4. The result describes that the means of running time of AEDA and EDAPSO are not significantly different, while those on other pairs are significantly different.

Fig. 5 shows the graphical illustration of all the algorithms' convergence when solving Problem 15 of EFLP. As illustrated in Fig. 5, the AEDA able to get the optimal solution in less iteration compared to other algorithms. GATS converges faster than AEDA; however, the objective value is worse than AEDA and EDAhybrid.

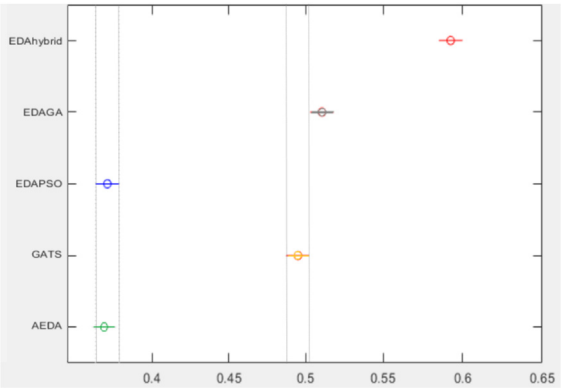

Fig. 4. The Dunnet test of Problem 15's running time.

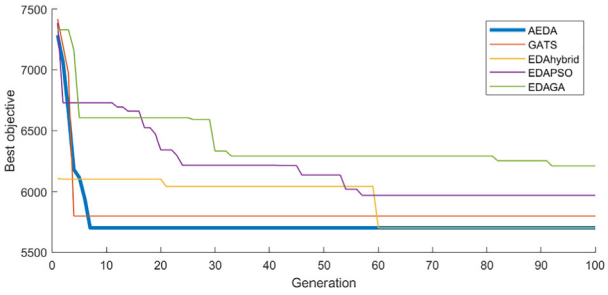

Fig. 5. The convergence comparison of all the algorithms (in Problem 15).

Although EDAhybrid first solution is better than different algorithms, it needs more generation to converge compare to AEDA and GATS. Meanwhile, EDAPSO and EDAGA need more iteration to get better solutions.

6. Conclusion

This research presents a comparative study of several hybridizations of EDA in solving the facility layout problem. This study also proposes a new algorithm called AEDA. The experimental results for AEDA, EDAGA, EDAPSO, EDAhybrid, and GATS are applied to instances of EFLP. All algorithms perform well in the first 3 problems, while AEDA, EDAhybrid, and GATS manage their good performances in all problems. Generally, based on the comparison of minimum value, maximum value, mean, and standard deviation of the solutions, the best algorithm is AEDA, followed by EDAhybrid and GATS, and finally EDAPSO and EDAGA. The ANOVA test is performed to some instances of the results and followed by post hoc (Dunnett) test. The Dunnett test show that the means of AEDA, EDAhybrid, and GATS are not significantly different. Meanwhile, the comparison of mean and standard deviation in all algorithms' running time show that AEDA and EDAPSO are faster than other algorithms.

The comparative study of the hybrid algorithms in this research is expected to inspire future research to determine which methods are likely to be chosen. The data and the results provided can become a benchmark for future research in EFLP. Future research also can focus on improving the performance of the algorithms or applying different cases.

Acknowledgements

The authors gratefully acknowledge financial support from the Institut Teknologi Sepuluh Nopember for this work, under project scheme of the Publication Writing and IPR Incentive Program (PPHKI).

References

[1] Gao S, de Silva CW. Estimation distribution algorithms on constrained optimization problems. Appl Math Comput 2018;339:323–45.
[2] Zhou B-H, Tan F. A self-adaptive estimation of distribution algorithm with differential evolution strategy for supermarket location problem. Neural Comput Appl 2020;32(10):5791–804.
[3] Utamima A, Reiners T, Ansaripoor AH. Evolutionary estimation of distribution algorithm for agricultural routing planning in field logistic. Proc Comput Sci 2019;161:560–7. doi: https://doi.org/10.1016/j.procs.2019.11.156.
[4] Pérez-Rodríguez R, Hernández-Aguirre A. A hybrid estimation of distribution algorithm for the vehicle routing problem with time windows. Comput Ind Eng 2019;130:75–96.
[5] Kalita Z, Datta D. A constrained single-row facility layout problem. Int J Adv Manuf Tech 2018;98(5–8):2173–84.
[6] Ou-Yang C, Utamima A. Hybrid estimation of distribution algorithm for solving single row facility layout problem. Comput Ind Eng 2013;66(1):95–103.
[7] S. Baluja, Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning, Tech. rep., Carnegie-Mellon Univ Pittsburgh Pa Dept Of Computer Science (1994).
[8] Martins JP, Delbem AC. Pairwise independence and its impact on estimation of distribution algorithms. Swarm Evol Comput 2016;27:80–96.
[9] Cravo GL, Amaral AR. A GRASP algorithm for solving large-scale single row facility layout problems. Comput Oper Res 2019;106:49–61.
[10] Luo J, Qi Y, Xie J, Zhang X. A hybrid multi-objective PSO–EDA algorithm for reservoir flood control operation. Appl Soft Comput 2015;34:526–38.
[11] Chen S-H, Chen M-C, Liou Y-C. Artificial chromosomes with genetic Algorithm 2 (ACGA2) for single machine scheduling problems with sequence-dependent setup times. Appl Soft Comput 2014;17:167–75.
[12] Handa H. The effectiveness of mutation operation in the case of estimation of distribution algorithms. Biosystems 2007;87(2–3):243–51.
[13] Scalia G, Micale R, Enea M. Facility layout problem: Bibliometric and benchmarking analysis. Int J Ind Eng Comput 2019;10(4):453–72.
[14] Liu S, Zhang Z, Guan C, Zhu L, Zhang M, Guo P. An improved fireworks algorithm for the constrained single-row facility layout problem. Int J Prod Res 2020:1–19.
[15] Şahin R, Niroomand S, Durmaz ED, Molla-Alizadeh-Zavardehi S. Mathematical formulation and hybrid meta-heuristic solution approaches for dynamic single row facility layout problem. Ann Oper Res 2020:1–24.
[16] Krömer P, Platoš J, Snášel V. Solving the single row facility layout problem by differential evolution, in. In: Proceedings of the 2020 genetic and evolutionary computation conference. p. 210–8.
[17] Datta D, Amaral AR, Figueira JR. Single row facility layout problem using a permutation-based genetic algorithm. Eur J Oper Res 2011;213(2):388–94.
[18] Ripon KSN, Glette K, Khan KN, Hovin M, Torresen J. Adaptive variable neighborhood search for solving multi-objective facility layout problems with unequal area facilities. Swarm Evol Comput 2013;8:1–12.
[19] Sule DR. Manufacturing facilities: location, planning, and design. CRC Press; 2008.
[20] Heragu SS. Facilities design. 4th ed. CRC Press; 2018.
[21] Kramer O. Evolutionary self-adaptation: A survey of operators and strategy parameters. Evol Intel 2010;3(2):51–65.
[22] Utamima A, Ou-Yang C. Solving single row facility layout problem using extended artificial chromosome genetic algorithm. Journal of Technology 2012;27(4):189–94.
[23] Chen Y-M, Chen M-C, Chang P-C, Chen S-H. Extended artificial chromosomes genetic algorithm for permutation flowshop scheduling problems. Comput Ind Eng 2012;62(2):536–45.
[24] Shen J-N, Wang L, Wang S-Y. A bi-population EDA for solving the no-idle permutation flow-shop scheduling problem with the total tardiness criterion. Knowl-Based Syst 2015;74:167–75.
[25] Briskorn D, Dienstknecht M. Survey of quantitative methods in construction. Comput Oper Res 2018;92:194–207.
[26] Paes FG, Pessoa AA, Vidal T. A hybrid genetic algorithm with decomposition phases for the unequal area facility layout problem. Eur J Oper Res 2017;256(3):742–56.
[27] Hasda RK, Bhattacharjya RK, Bennis F. Modified genetic algorithms for solving facility layout problems. Int J Interact Des Manuf 2017;11(3):713–25.
[28] Tamssaouet K, Dauzère-Pérès S, Yugma C. Metaheuristics for the job-shop scheduling problem with machine availability constraints. Comput Ind Eng 2018;125:1–8.

[29] Utamima A, Pradina KR, Dini NS, Studiawan H. Distribution route optimization of gallon water using genetic algorithm and tabu search. Proc Comput Sci 2015;72:503–10.
[30] Guan J, Lin G. Hybridizing variable neighborhood search with ant colony optimization for solving the single row facility layout problem. Eur J Oper Res 2016;248(3):899–909.
[31] Utamima Amalia, Reiners Torsten, Ansaripoor AH. Automation in Agriculture: A Case Study of Route Planning Using an Evolutionary Lovebird Algorithm. ACM International Conference Proceeding Series 2020:13–7. doi: https://doi.org/10.1145/3384613.3384621.