DEVELOPMENT EVOLUTIONARY ALGORITHM OF HARMONY ALGORITHM USING MATHEMATICAL MODELING BASED ON C++ FOR SOLVING QUADRATIC ASSIGNMENT PROBLEM

Lutfan Anas Zahir

Universitas Tulungagung

Keywords:
Harmony Algorithm, Construction Project Facilities, Work Trip Frequency, Optimization, Quadratic Assignment Problem.

***Correspondence Address:**
lutfananas@gmail.com

Abstract: Industry is beginning to touch cyberspace in the form of human, machine, and computer connectivity data, which can be found everywhere in our time. The Internet of Things (IoT) is the term used to describe this phenomenon (IoT). The fourth industrial revolution changed the concept of employment, job structure, and capabilities needed in highly efficient workplaces. The Industrial Revolution 4.0 is driven by digital technology. Artificial intelligence, the Internet of Things (IoT), and big data processing can process enormous amounts of data, virtually expose a situation, and offer solutions for appropriate, effective, and efficient decision-making, particularly in development and construction projects. A construction project's facility layout significantly impacts the project's efficiency and implementation costs. Project facility layout aims to find minimum operating costs through various facility layout compositions. In this study, researchers used the harmony algorithm, which was applied to determine the layout of project facilities with the function of duration, distance, and frequency constraints. The constraint function in determining facilities can be modeled with the Quadratic Assignment Problem equation so that it has the goal of minimizing results. Researchers also developed a harmony algorithm using C ++ language program code to make it easier to find optimal solutions. Running the program with Borland C ++ compiler software shows that with data of 14 facilities and 14 work trips between facilities and Iteration parameters = 2000, HM = 100, HMCR = 0.7, PAR = 0.3, and Bandwidth = 0.1, the best solution results (8.1) - (9.2) - (14.3) - (12.4) - (7.5) - (1.6) - (5.7) - (3.8) - (6.9) - (13.10) - (11.11) - (10.12) - (2.13) - (4.14). The results show that the larger the iteration, the program will provide output that is closer to the optimal solution.

INTRODUCTION

Businesses are starting to enter cyberspace with data on computer, machine, and human connectivity—info that is ubiquitous in modern society. This concept is referred to as the Internet of Things (IoT). The nature of employment, the organization of labor, and the skills needed in the workplace are all evolving due to the fourth industrial revolution. The emphasis on digital transformation from business to platform has increased the need for Human Resources (HR) specialists with expertise in areas that differ significantly from previous needs. (Oktika, 2022). The disruptive age, or Industrial Revolution 4.0, emerged quickly, eradicating all prior guiding principles and erecting a new, structured framework. Digital technology is driving the Industrial Revolution. Artificial intelligence (AI), the Internet of Things (IoT), and big data, when combined with the industry 4.0 movement, can process enormous volumes of data, virtually explain a situation, and provide solutions for precise, effective, and efficient decision-making. Considering those above, developed countries began to make changes. Japan pioneered the concept of Society 5.0, a technologically advanced, people-focused society. (Suherman, 2020).

The development of a universal science—which serves as the foundation for advancing human thought and has significant applications across a wide range of disciplines—has been greatly aided by mathematics. Mathematics is crucial for resolving issues in daily life (Kusmanto & Marliyana, 2014). Using matrix concepts, which can be used as arrays in an algorithm or as programming codes to create layout facilities using the assignment method, is one way that mathematical concepts are implemented.

The assignment problem is allocating resources (facilities) to activities on a one-to-one basis. Each activity or task will get a specific facility from allocating facility resources. Allocating facilities and activities are interrelated costs. The objective form of minimization is to determine all activities carried out at the minimum possible cost (Singh et al., 2012). The way construction sites are designed has a significant impact on how productive construction workers are. Optimization of the layout of construction facilities needs to be done when planning the layout of building facilities; it is essential to figure out how often people need to travel between them and how much open space is needed. When arranging construction facilities, two requirements must be met for the number of facilities and the amount of land needed to be equal. The first requirement is that there are more land holdings than there are facilities. The second requirement is the quantity of land necessary to accommodate the necessary construction facilities. Determining the facility's location (n) on the available land (m) with the least total worker travel distance is the primary goal of construction facility layout planning. In order to attain optimal results, worker travel frequency and trip distance are utilized as the primary variables in the quadratic assignment problem (QAP) model of layout planning for construction projects (Prayogo et al., 2018). The Quadratic Assignment Problem (QAP) is a facility placement model that aims to reduce the overall costs of completed tasks. The locations of the currently operating facilities have an associated cost. Facilities can be positioned strategically if they are close to other nearby locations. Relocation will be less expensive the closer the links are between locations and facilities. (Koopmans and Beckmann, 1957).

The facility layout arrangement on a construction project significantly affects the project's efficiency and the cost of implementation. The project facility layout aims to find the minimum operational cost through various facility layout compositions. A good layout can produce the most efficient operations and reduce project costs (Prayogo et al., 2018b). A good layout can be realized with effective and efficient resource management. A good site layout must also consider the function of each existing layout so that the existing infrastructure can work optimally (Irawan & Supani, 2016). The arrangement is to utilize the area to place machines or other production support facilities to increase production output, reduce waiting time (delay), and reduce the process of moving materials (material handling). If the material handling system is sound, the production process will run well, too (Rajak, 2018).

The rapid development of technology provides many options for a construction company to determine the methods and tools of Metaheuristic search algorithms that can be modeled or adapted to various problems. Metaheuristic algorithms combine rules and randomness from natural phenomena to find the global optimum result using trial and error methods (Cheng et al., 2016). Harmony algorithm as a meta-heuristics algorithm is an optimization algorithm that can be applied in various cases. The Harmony Search algorithm is part of Heuristic search. This approach bases its search on musical harmonization. There is a common thread that bridges between music and all kinds of optimization problems. An experience will contain a harmonious memory matrix. The optimal solution is obtained by determining an evaluation function known as the New Solution Vector (NSV). The New Decision Variable finds the desired result, leading to a minimization or maximization function (Risan et al., 2021).

Zoom Woo Geem presented the Harmony Search (HS) algorithm in 2001. The Harmony Search (HS) algorithm's basic idea is to imitate the method used by music experts to improve musical harmony. There are multiple options when a music expert adjusts the piece's harmony. Three options are available: playing a well-known harmony from memory, playing a well-known harmony with minor modifications, or inventing a brand-new harmony. Pitch modification, the use of Harmony Memory (HM), and the random generation process are the three options that Geem et al. (2001) developed for the quantitative optimization process. Using Harmony Memory is the first option (HM).

Using harmonic modulation (HM) is essential because it guarantees that harmonic components will be considered when developing the new solution vector. In order to effectively use HM, harmonic modulation (HM) is essential because it guarantees that harmonic components will be considered when developing the new solution vector. In order to effectively use HM. In order to use Harmony Search effectively, a parameter is adopted by the algorithm $_{accept}$ ϵ [0,1], called Harmony Memory Considering (or Accepting) Rate (HMCR). If this value is too low, then only a few elite harmonies are selected, and it can also cause the convergence process to be too slow. If the value is too large, it will cause the notes in the HM to be used a lot and not have time to explore other notes, making it difficult to achieve a good solution. Therefore, γ_{accept}=0.7~0.95 (HMCR) is usually used. The second option is pitching adjustment, which has several parameters such as bandwidth adjustment ($_{range}$) and pitch adjustment value, namely Pitch Adjusting Rate (PAR).

Generates slightly different values in the HS algorithm. The following is the pitch adjustment formulation:

$$x_{new} = x_{old} + (b_{range})\,\varepsilon$$

With:

x_{new} = new pitch after pitch adjustment

x_{old} = tones stored on HM

$b_{range} = bandwidth$

ε = a random number from the uniform distribution with the interval [-1,1]

A low pitch adjustment (PAR) value with a narrow bandwidth can lead to slow convergence due to limited exploration of an ample search space. On the other hand, a high pitch adjustment value with a wide bandwidth may cause the solutions to be too spread out around the optimal solution as in a random search. Therefore, PAR = 0.1 ~ 0.5 is usually used.

The third option is randomization or generating real numbers between intervals [0,1]. This process is used to increase the diversity of solution vector values. Randomization can encourage the system to search for solutions to achieve the global optimum (Yang, 2009).

The following are the steps of the Harmony Search algorithm according to Gholipur and Shahbazi (2011) and Saka (2009):

a. Initialization of Harmony Search Algorithm Parameters

Harmony Memory Size (HMS), Harmony Memory Considering (or Accepting) Rate (HMCR), Pitch Adjusting Rate (PAR), bandwidth, and stopping criteria are the parameters that make up the Harmony Search algorithm.Inisialisasi *Harmony Memory* (HM)

$$\begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_{N-1}^1 & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_{N-1}^2 & x_N^2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ x_1^{HMS-1} & x_2^{HMS-1} & \cdots & x_{N-1}^{HMS-1} & x_N^{HMS-1} \\ x_1^{HMS} & x_2^{HMS} & \cdots & x_{N-1}^{HMS} & x_N^{HMS} \end{bmatrix}_{\text{HMS x N}}$$

Figure 1. Matriks Harmony

b. Generating New Solution Vectors

Generate new solution vectors $x_1', x_2', \dots, x_n'$. The new solution vector $x' = x_1', x_2', \dots, x_n'$ is generated by selecting a value stored in HM or from all possible values. Generate a new solution vector. If r_1 is more significant than HMCR, the new solution value is randomly generated and uniformly distributed between 0 and 1. This possibility is determined by randomly generating a uniformly distributed number between 0 and 1 (r_1). If r_1 is less than or equal to Harmony Memory Considering Rate, then one value in the i-th column of Harmony Memory is chosen as the new solution value. If r_1 is more significant than HMCR, a new solution value is randomly generated and uniformly distributed between 0 and 1. The process can be expressed in the equation below.

$$x_i' = \begin{cases} x_i' \in \{ x_1', x_2', \dots, x_n' \}, r_1 \le HMCR \\ x_i' \in rand\ [0,1], \quad r_1 > HMCR \end{cases}$$

The solution value that has been obtained needs to be adjusted pitch. This adjustment is done by adding or subtracting 1 from the current solution value. This adjustment process is influenced by the Pitch Adjustment Rate (PAR) parameter; if the solution value is outside the PAR parameter, the solution value does not change. The process can be expressed in the equation below.

$$x_i'' = \begin{cases} x_i' \pm 1, & r_2 \le PAR \\ x_i' \quad , & r_2 > PAR \end{cases}$$

In previous research, the harmony algorithm has been widely applied to solve several discussions, including Project Scheduling Optimization Application with the

Harmony Search Method (Widana, 2020), Optimization of Steel Portal Structure Design with the Metaheuristic Method - Harmony Algorithm (Yusliani & Rizki, 2016). Previous research shows that the Harmony Algorithm can provide problem optimization, and the solution obtained can be used as the best decision-making.

Based on the description previously described, it is exciting to apply the Harmony Algorithm as a means of Optimizing the Layout of Construction Project Facilities. The solutions obtained are expected to be minimal and better than previous algorithms.

RESEARCH METHODS

The main objective of construction facility layout planning is to determine the position of facilities (n) on the available land (m) with the minimum total worker travel distance. Construction project facility layout planning is modeled as a Quadratic assignment problem (QAP) using travel distance and worker travel frequency as the main variables in obtaining optimum results.

The data used in this study is secondary data; the data source comes from observations by previous researchers by Jonathan et al. (2004). This case has 14 facilities that must be organized in 14 locations. These facilities include: (1) Main Gate (MG); (2) Site Gate (SG); (3) Guard Post (PJ); (4) Office (K); (5) Workers' Toilet (TP1); (6) Wiremash Place (TW); (7) Tower Crane (TC); (8) Workers' Toilet 2 (TP2); (9) Power Source (SL); (10) First Aid Post (PP); (11) Warehouse (G); (12) Worker Barracks (BP); (13) Reinforcement Fabrication Site (TFP); (14) Column Formwork Site (TBK).

The steps used in writing this thesis are as follows:

1. Review literature quadratic assignment problem and harmony search.
2. Implement a hybrid harmony search process and taboo search with the following steps:
   a. Initialization of harmony data and parameters and taboo *search*, among others: harmony memory search (HMS), *harmony memory considering* rate (HMCR), *pitch adjustment rate* (PAR), *bandwidth*, and *swap permutation maximum*
   b. Generate as many initial solution vectors as HMS
   c. Calculates the destination (*fitness*) function of each solution vector to obtain the initial solution vector (VSA)
   d. Generating a new solution vector (VSB) through 3 stages, among others:

1) If the generation of real interval numbers [0,1] randomly r_1 greater than HMCR will be generated by a new decision variable (VKB) by *random* means [0,1]

.

2) It will be calculated if the generation of a random real interval [0,1] r_1 is smaller than HMCR.

   $P_1 = \text{int}(1 + (\text{HMS-1})r_1)$\

   $P_2 = \text{HM}(P_1,i)$

   $i = 1, .. , N$

   with:

   P_1 = The value to be used for site selection on the matrix HM

   P_2 = new decision variable (VKB) values are taken from the matrix HM

   int = Rounding process

   i = Solution vector index

   After obtaining a new decision variable, the actual interval number [0,1] is generated randomly r_2; if the value of r_2 is more significant than PAR, then the new decision variable I chose is P_2.

   $\text{VKB}(i) = P_2$

3) If the random generation of the real interval [0,1] is less r_1 than HMCR and the random generation of the real interval [0,1] is less than PAR, it will be calculated. r_2

   $P_3 = P_2 + bandwith * \varepsilon$

   $\text{VKB}(i) = P_3$

e. Calculate the value of the destination function (*fitness*) of the new solution vector (VSB)

f. Check whether the new solution vector is better than the worst solution vector stored in *Harmony Memory* with the following conditions:

   1) If it is better, then the vector of the solution replaces the vector of the worst solution stored on the HM matrix (HM *update)*

   2) If it is not better, the vector must be *updated* on HM.

g. Check whether the termination criteria have been met with the following conditions:

   1) If so, then the iteration process stops

2) If not, then the process is repeated, starting from the step of generating a new solution vector (VSB)

3. Create a *Harmony Search* program to complete Construction Project Facility Layout Optimization.
4. Implement the program in case examples

RESULTS AND DISCUSSION

Startup Project Layout

The case of the research data by this data source has 14 facilities that must be arranged in 14 locations. These facilities include: (1) Main Gate (MG); (2) Site Gate (SG); (3) Guard Post (PJ); (4) Office (K); (5) Workers' Toilet (TP1); (6) Wiremash Place (TW); (7) Tower Crane (TC); (8) Workers' Toilet 2 (TP2); (9) Power Source (SL); (10) First Aid Post (PP); (11) Warehouse (G); (12) Worker Barracks (BP); (13) Reinforcement Fabrication Site (TFP); (14) Column Formwork Site (TBK).

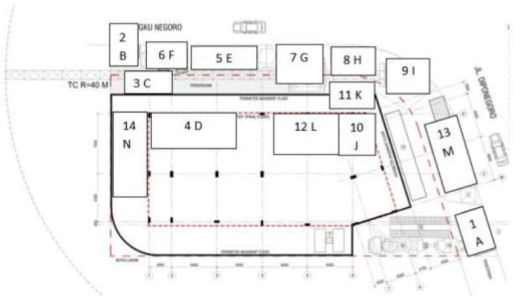

Figure 2. Startup Project Layout

Mileage Data between Locations

Table 1. Mileage between Locations

Locations	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**0**	**1**	**2**	**3**	**4**
1	0	5	0	3	8	7	5	7	0	3	1	7	0	1
2	5	0	7	4	5	7	3	3	1	5	0	6	7	5
3	0	7	0	7	2	4	0	0	3	7	1	8	5	3
4	3	4	7	0	9	9	2	3	6	0	5	1	2	5
5	8	5	2	9	0	2	4	4	2	3	5	4	4	8
6	7	7	4	9	2	0	3	8	6	5	9	8	5	2
7	5	3	0	2	4	3	0	2	0	0	5	0	2	8
8	7	3	0	3	4	8	2	0	3	9	5	3	0	8
9	0	1	3	6	2	6	0	3	0	2	5	5	1	2

10	8	.5	7	0	3	5	.0	9	.2	0	1	9	6	6
11	.1	.2	4	.5	.5	.9	6	5	5	1	0	6	4	6
12	.7	6	8	.1	.4	.8	.0	.3	.5	9	6	0	.5	7
13	0	.7	5	2	4	5	.2	.0	1	6	4	.5	0	1
14	1	.5	8	6	.8	.2	8	8	2	6	6	7	1	0

Work Trip Frequency Between Facilities

Table 2. Work Trip Frequency between Facilities

Facilities	IG	G	J	K	P1	W	C	P2	L	P	G	P	FT	FB
MG	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**
SG	**0**	**0**	**1**	**1**	**1**	**30**	**1**	**1**	**1**	**3**	**.5**	**2**	**2**	**0**
PJ	**0**	**1**	**0**	**1**	**0**	**0**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**0**
K	**0**	**1**	**1**	**0**	**3**	**1**	**1**	**1**	**1**	**2**	**2**	**3**	**2**	**2**
TP1	**0**	**1**	**0**	**3**	**0**	**0**	**1**	**0**	**0**	**2**	**0**	**4**	**0**	**0**
TW	**0**	**30**	**0**	**1**	**0**	**0**	**0**	**1**	**0**	**4**	**2**	**4**	**4**	**0**
TC	**0**	**1**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**1**	**0**	**1**	**0**	**0**
TP2	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**0**
SL	**0**	**1**	**1**	**1**	**0**	**0**	**1**	**1**	**0**	**0**	**0**	**1**	**0**	**0**
PP	**0**	**3**	**1**	**2**	**2**	**4**	**1**	**2**	**0**	**0**	**3**	**3**	**2**	**2**
G	**0**	**.5**	**1**	**2**	**0**	**2**	**0**	**2**	**3**	**3**	**0**	**2**	**.5**	**2**
BP	**0**	**2**	**1**	**3**	**4**	**4**	**1**	**2**	**3**	**3**	**2**	**0**	**2**	**2**
TFT	**0**	**2**	**1**	**2**	**0**	**4**	**0**	**2**	**2**	**2**	**.5**	**2**	**0**	**0**
TFB	**0**	**0**	**0**	**2**	**0**	**0**	**0**	**2**	**2**	**2**	**2**	**2**	**0**	**0**

Data Processing Process with C ++ Programming Language

The layout of construction project facilities is optimized with harmony algorithms as a search method to obtain optimal solutions. The results of the solution will be used as a sequence in determining the function of the goal, namely to minimize the cost of work. The process of encoding or pseudocode will be explained as follows:

The harmony algorithm process begins with the initialization of data and parameters. The following process is to input the location data matrix and work frequency between facilities with data calling and continue running the program according to the harmony algorithm to optimize the construction project facility layout. The pseudocode Initialization of Parameter listing like this below :

# Initialization of Parameter

```
void Parameter()
 do{cout<<"    Masukan HMS\t\t: ";
 do{cout<<"    Masukan HMCR\t: ";
 do{cout<<"    Masukan PAR\t\t: ";
 do{cout<<"    Masukan Bandwith\t: ";
```

The parameter input process is a decisive step in determining the probability of running the harmony algorithm. The parameter is used as the decision determination rate so that the solution vector is obtained optimally quickly. The next step is to *generate the* initial solution vector. The input process *generates* the initial population through calling data stored through *external* data. The pseudocode-generating vector Solution listing is like below :

#Generate Vektor Initial Solution

```
void
    HMTS1.open("1. Data Fasilitas 14x14.txt");
    HMTS2.open("2. Data Lokasi 14x14.txt");
```

After the initial solution vector is formed, according to the harmony algorithm steps, it is necessary to calculate the fitness value first using the Quadratic Assignment Problem (QAP) model by the research problem as a solution in determining the layout of facilities with two constraint functions. The pseudocode for calculating fitness listing is like this:

#Calculating *Fitness*

```
float ft=0;
 for(i=0;i<nfasilitas;i++)
  { for(j=0;j<nfasilitas;j++)
    { for(k=0;k<nfasilitas;k++)
      { for(l=0;l<nfasilitas;l++)
        { ft+=fasilitas[i][k]*lokasi[j][l]*penempatan[i][j]*penempatan[k][l]; }}}}
  return ft; }
```

The results of fitness calculations will be stored through memory harmony. The next step is to generate a New Decision Vector (VKB) through e-condition criteria and continue to calculate the fitness value of the new VKB. The algorithm will compare whether the fitness score is better than the previous fitness. If yes, update memory; if not, check whether the maximum iteration has been met. The VKB generation process is as follows:

#Generation of New Decision Variables Condition 1

```
float index[ap][aq],ft[ap];
for(i=0;i<HMS;i++)
{
for(j=0;j<nfasilitas;j++)
{
 if(jln==1)
{index[i][j]=indexharmonymemory[i][j];ft[i]=ftharmonymemory[i];}
else if(jln==2)
{index[i][j]=indexVKB[i][j];ft[i]=ftVKB;}
else if(jln==3)
{index[i][j]=indexharmonyupdate[i][j];ft[i]=ftharmonyupdate[i];}}
```

#Generation of New Decision Variables Conditions 2 and 3

```
float HarmonySearch()
{ i=0;
do
```

# Generation of New Decision Variables Condition 2

```
{
r1[i]=R*random(10000);
if(r1[i]<HMCR)
{
d1[i]=1+(HMS-1)*r1[i];
D1[i]=cek(d1[i]);
D2[i]=harmonymemory[D1[i]][i];
r2[i]=R*random(10000);
```

# Generation of New Decision Variables Condition 3

```
if(r2[i]<PAR)
{
epsilon[i]=pow(-1,random(10))*R*random(10000);
VKB[i]=D2[i]+bandwith*epsilon[i];
}
else
{VKB[i]=D2[i];}
}
else
{VKB[i]=R*random(10000);}
i++;
}
```

After the New Decision Variable is obtained, the next process is to calculate the fitness value from VKB and update Memory Harmony if there is a better fitness value. The process of Updating Memory Harmony is as follows:

#Updating Harmony Memory

```
if(i==ftmax)
{ftharmonyupdate[ftmax]=ftVKB; harmonyupdate[ftmax][j]=VKB[j];
 indexharmonyupdate[ftmax][j]=indexVKB[0][j];}
 else
{ftharmonyupdate[i]=ftharmonymemory[i];
 harmonyupdate[i][j]=harmonymemory[i][j];
 indexharmonyupdate[i][j]=indexharmonymemory[i][j]; }
```

The process of updating harmony generates a New Solution Vector that has been stored in harmony memory. Next, check whether the maximum iteration has been met. If the maximum iteration is complete, the best Solution Vector is obtained. The process of checking the iteration of harmony memory is as follows:

#Maximum Iteration Check

```
if(iterasi==1)
{ iterasi++; }
```

```
while(iterasi<=maxiterasi);
getch();
```

If the maximum iteration has been met, determine the New Solution Vector selected from harmony memory. Selection is based on the criteria of the smallest fitness value by the function of the goal, namely minimization/optimization. The output of VSB determination is used as the most optimal solution of the Harmony algorithm process. The process of maintaining the best solution with harmonious memory is as follows:

#Vector Determination of New Solutions

```
void FungsiTujuanBest()
{
  float best;
  int   index best;
  best=Minimum(harmony update,HMS);
  for(i=0;i<HMS;i++)
{
  if(ftharmonyupdate[i]==best)
  indexbest=i;
}
  printf("\n    X%d   ",indexbest+1);
  for(j=0;j<=nfasilitas;j++)
{if(j==nfasilitas-1)
    {printf("(%.f,%d)  ",indexharmonyupdate[indexbest][j]+1,j+1);}
  else if(j==nfasilitas)
    {printf(">>  Fungsi Tujuan = %.f  ",ftharmonyupdate[indexbest]);}
  else
    {printf("(%.f,%d) - ",indexharmonyupdate[indexbest][j]+1,j+1);}
}
  cout<<endl;}
```

The harmony algorithm for the construction project facility layout optimization solution has been completed. The compiler program in this study uses Borland C ++ Version 5.02 and C ++ language. The results of the running program with parameters showed the results of the optimal Harmony Memory (HM)= 10, Harmony Memory Considering Rate (HMCR)= 0.7, Pitch Adjusting Rate (PAR)= 0.3, bandwidth = 0.1 dan Max Iterasi=1000 *fitness* solution of 4222 with the best solution obtained the results were (8.1) - (9.2) - (14.3) - (12.4) - (7.5) - (1.6) - (5.7) - (3.8) - (6.9) - (13.10) - (11.11) - (10.12) - (2.13) - (4.14).

CONCLUSIONS AND RECOMMENDATIONS

The Harmony algorithm can be used to solve the problem of optimality of the layout of construction project facilities with algorithmic steps by construction project problems based on mathematical models called quadratic problems or Quadratic Assignment Problems (QAP). The QAP model can be used as a solution to achieve the optimization process using C ++ language programs.

The program to solve the problem of optimality of the layout of construction project facilities is based on the QAP mathematical model, meaning that the function of the program objective is to minimize each existing constraint. The program is made according to the criteria of genetic algorithm procedures made with the Borland C++ program version 5.02 with the C / C ++ programming language.

Program implementation for case examples using data from 14 facilities and 14 locations obtained the best objective function value of 4222 with the best solution (8.1) - (9.2) - (14.3) - (12.4) - (7.5) - (1.6) - (5.7) - (3.8) - (6.9) - (13.10) - (11.11) - (10.12) - (2.13) - (4.14). The parameters used include: Data = 14x14, Iteration = 1000, HM = 50, HMCR = 0.7, PAR = 0.3, and Bandwidth = 0.1.

For subsequent research, the Harmony algorithm can be combined with several other algorithms, such as *simulated annealing*, *ant colony optimization*, and others, to achieve a more optimal solution faster.

REFERENCES

Herianjo, I., & Budi, R. (2013). *Pemrograman Borland C++ Builder 6 Revisi Ketiga*. *January 2013*, 10.

Irawan, A. C., & Supani. (2016). Site Layout Optimization in One East Surabaya Building Project. *Jurnal Teknik Its*.

Jonathan, V., Sugiarto, A. K., Tanojo, E., & Prayogo, D. (2004). Optimasi Construction Site Layout Menggunakan Metode Metaheuristic Algorithm Pada Proyek Great Hotel Diponegoro. *The NCCU Journal of Sociology*, *33*(January 2003), 4–6.

K. S. &. C. Maufrais (2020), Introduction to Programming using Python, Boston.

Rizki, T. D. (2016). *Rancang Bangun Sistem Pengecekan Ambiguitas Kalimat Berbahasa Indonesia Menggunakan Harmony Search Algorithm*. *2*(1), 173–176.

Oktika, M. (2022). Digitalisasi era industri 4.0 berperan penting di dalam pendidikan. *Seminar Nasional NBM Art*, *46*.

Herianjo, I., & Budi, R. (2013). *Pemrograman Borland C++ Builder 6 Revisi Ketiga*. *January 2013*, 10.

Irawan, A. C., & Supani. (2016). Site Layout Optimization in One East Surabaya Building Project. *Jurnal Teknik Its*.

No, V., & Rizki, T. D. (2016). *Rancang Bangun Sistem Pengecekan Ambiguitas Kalimat Berbahasa Indonesia Menggunakan Harmony Search Algorithm*. *2*(1), 173–176.

Oktika, M. (2022). Digitalisasi era industri 4.0 berperan penting di dalam pendidikan. *Seminar Nasional NBM Art*, *46*.

Prayogo, D., Eric, S., Sutanto, J. C., Suryo, H. E., Studi, P., Sipil, T., & Petra, U. K. (2018). *Menggunakan Algoritma Metaheurisitik*. 1–8.

Prayogo, D., Eric, S., Sutanto, J. C., Suryo, H. E., Studi, P., Sipil, T., & Petra, U. K. (2018). *Optimasi Tata Letak Fasilitas Proyek Konstruksi Dengan Menggunakan Algoritma Metaheurisitik*. *B01*(1), 1–8.

R, R. R., Tedy Rismawan, & Rahmi Hidayati. (2021). Penerapan Algoritma. *JUPITER*, *13*(2), 179–187.

Rajak, S. (2018). *Optimasi Tata Letak Fasilitas Produksi Menggunakan Algoritma Genetika*.

Suherman. (2020). *Industry 4.0 VS Society 5.0* (Retnani et al. (ed.)). CV. Pena Persada.

Widana, Y. (2020). *Aplikasi Optimalisasi Penjadwalan Proyek dengan Metode Harmony Search (Studi Kasus : AKSDAI)*. *1*(3), 88–91.