Applying a glowworm swarm algorithm for optimising the assembly sequence of a car engine pump valve

1st Fawaz Alharbi
School of Engineering
University of Portsmouth
Portsmouth, United Kingdom
Fawaz.alharbi@port.ac.uk

2nd Qian Wang
School of Engineering
University of Portsmouth
Portsmouth, United Kingdom
Qian.wang@port.ac.uk

Abstract When assembling a product, assembly sequence is subject to the inherent nature of assembly components as well as the number of assembly components that may also increase complexity of assembly process. Additionally, assembly efficiency relies on reduction of assembly time that is partially affected by determination of an optimal assembly sequence of a product. A study through a literature review shows that the glowworm swarm optimisation algorithm (GSOA) can be possibly useful for solving assembly sequence optimisation problems. This paper presents an investigation into a GSOA used for solving an assembly sequence problem of a car engine pump valve. The study demonstrates that the GSOA can provide a quick solution in obtaining an optimal or near-optimal assembly sequence of the assembled product that is manufactured in a small-sized company.

Keywords Assembly Sequence Optimisation, Glowworm Swarm Optimisation Algorithm, Product Design

I. Introduction

If a product has more than one component, it must be assembled. Thus, assembly sequence planning (ASP) is one of major tasks in product design that needs to be decided at the early stage; reduction of assembly time is the key for improving assembly efficiency therefore lowering assembly cost. One of the issues in ASP is that an expansion in the number of components may lead to the assembly process more unpredictable Chang et al. [1]. A matrix approach for analysing the information derived from a CAD model to obtain the assembly sequence for a two-stroke engine aiming to reduce both assembly time and cost Ou et al. [2]. Huang and Zhou, He et al., Marinakia and Marinakis, Yang et al., and Yu and Yang developed the glowworm swarm optimisation algorithm (GSOA) aiming to solve engineering optimisation problems, GSOA was derived from the courtship behaviour of the insect called glowworms who alter the sufficiency of their light discharge of bioluminescence gleam for various purposes [3-8]. The GSOA is useful for a simultaneous search of multiple optimal values usually based on different objective functions. To accomplish this objective, a swarm must have a capacity to be part into disjoint gatherings. Amid one program run, the GSOA is fit for deciding the numerous ideal arrangements in parallel. To begin with, the algorithm includes a random deployment of a population in a specified size n glowworms in an inquiry space at the beginning and each conveys a luminescence containing an amount of luciferin as physical entity. Area of a glowworm is dictated by a target work figuring the quality of luciferin, i.e., the force of luciferin is related with the target capacity of a glowworm's area. A more noteworthy luciferin force suggests a superior area related with a target work esteem. Each individual glowworm refreshes its luciferin level in light of the target work estimation of its current position. It was accounted for that the GSOA was viably utilized for streamlining of multi-work remote sensors solving a number of analytical problems [8, 9].

II. The Glowworm Swarm Optimisation Algorithm

Fig.1illustrates the mechanism of the GSOA. In this paper, a glowworm represents a component and a swarm of components is a population that is firstly spread randomly in a search space. Like the natural world, each component also acts as if it is a luminescent glowworm emitting a light whose intensity is proportional to the linked luciferin interacting with other glowworms within a defined neighbourhood. The neighbourhood area is classified as a local-decision domain that has a variable neighbourhood range r_d^i bounded by a radial luciferin sensor range *rs* ($0 < r_d^i \leq rs$).

In this work, assuming that component *i* considers another component *j* of its neighbour, if *j* is within the neighbourhood range of *i* and the luciferin level (in this instance, it denotes to the gap in dimensions between two mating components, i.e., mating component *i* with component j or parts based on the time taken to assemble) of *j* is higher than that of *i*. Each component is attracted by a suitable dimension of another glowworn in the neighbourhood. Components in a GSOA depend only on information accessible in their neighbourhood to make possible decisions. Consequently, each component selects a probabilistic neighbour that has a higher appropriate dimension and moves toward it. These movements, enable a swarm of components to be divided into separate subgroups that lead toward and meet with a multiple optima of a given multimodal function, whereby the functional integrity of the components are not compromised.

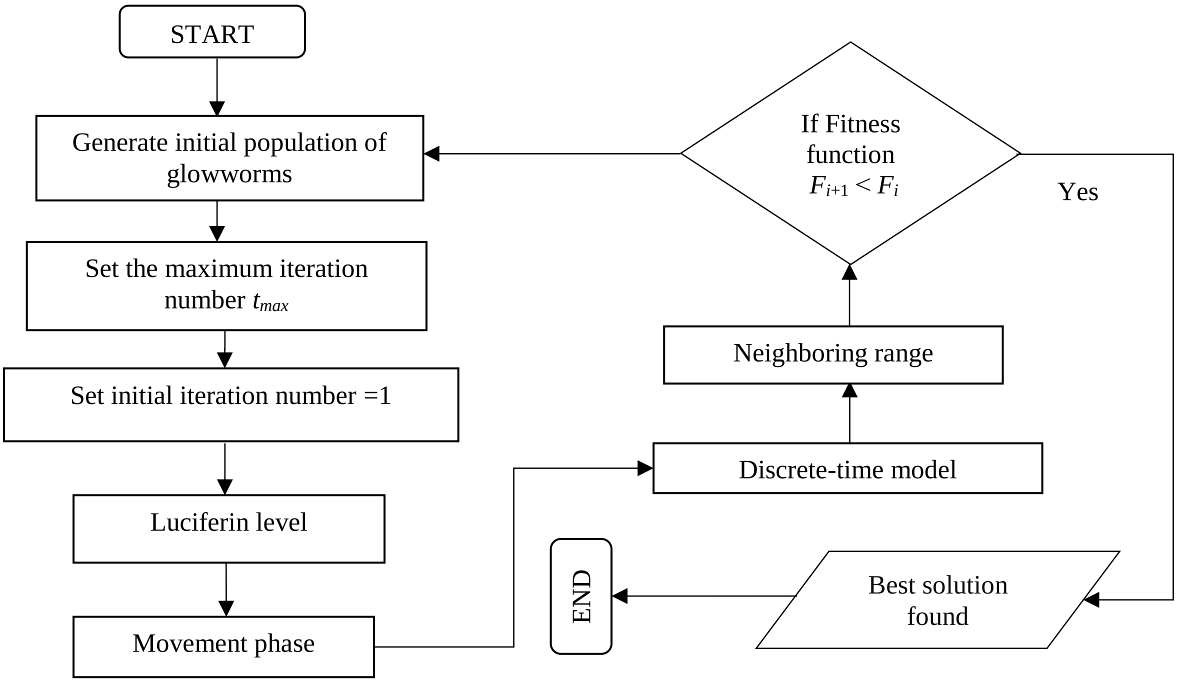
START
Generate initial population of glowworms
Set the maximum iteration number t_{max}
Set initial iteration number =1
Luciferin level
Movement phase
Discrete-time model
Neighboring range
If Fitness function $F_{i+1} < F_i$
Yes
Best solution found
END

Fig. 1. Mechanisme of the glowworm swarm optimisation algorithm

The following variables are used:

ℓ_0 quantity of luciferin

n random population of n glowworms ($1 \leq n \leq 14$ in this study)

r_d^i neighbourhood range

r_s radial sensor range

γ luciferin enhancement constant

ρ luciferin decay constant

A. *The Luciferin level*

At the beginning of the initial iteration, all the glowworms start with the equal value of luciferin ℓ_0 and the luciferin update is depending on the function value at a glowworm location. Through the luciferin-update phase; each glowworm adds its previous luciferin level, i.e., a luciferin quantity proportional to the fitness of its current location based on the objective function. Therefore, the objective function value for a glowworm at iteration *t* is calculated using the luciferin update rule as follows:

$$\ell_i(t) = (1-\rho)\ell_i(t) + \gamma J(\chi_i(t)) \tag{1}$$

Where $\ell_i(t)$ represents the luciferin level of glowworm *i* at time *t*; *J* (*xi* (*t*)) indicates the objective function value of glowworm *i* at time *t*; *xi* denotes the luciferin's location of a glowworm *i*; ρ denotes the luciferin decay constant ($0 < \rho < 1$), and γ is enhancement constant of the luciferin

B. *The Movement phase*

Through the movement phase, the probabilty of the location of a glowworm moves in the direction of a neighbour that has a luciferin value higher than its own value. The glowworm tends to get more attraction as its luciferin level increases. This is obtained from the fact that glowworms are attracted to neighbours that glow brighter. The probability p of glowworm *i* that moves towards *j* at time *t* is given below:

$$P_{ij}(t) = \frac{\ell_j(t)-\ell_i(t)}{\sum_{k\epsilon N_i(t)} \ell_k(t)-\ell_i(t)} \tag{2}$$

Where, $j \in Ni$ (*t*) and *Ni* (*t*) = {*j* : *dij* (*t*) < r_d^i (*t*); ℓ_i (*t*) < ℓ_j (*t*)}, which is a set of neighbour of glowworm *i* at time *t*, dij (t) denotes the Euclidean space, usually from glowworms *i* and *j* at time *t*, and r_d^i (*t*) represents the variable neighbourhood difference related to glowworms *i* and at time *t*. Let glowworm *i* select a glowworm $j \in Ni$ (*t*) with *Pij* (*t*) given by Eq. 2.

The discrete-time of the glowworm movements can be represented as:

$$x_i(t) = x_i(t) + \left(\frac{x_j(t)-x_i(t)}{\|x_j(t)-x_i(t)\|}\right) \tag{3}$$

Where, *xi* (*t*) denotes glowworm i location at time *t*, ‖ . ‖ represents the norm operator of Euclidean.

C. *The neighbourhood range*

There is a combination between glowworm *i* and *j* within a neighbourhood range. The term r_d^i of glowworm *i* is a dynamic radial range at initial iteration, providing $0 < r_d^i \leq rs$. The number of peaks taken is a function in a range of the radial sensor. When the glowworms depend only on local information to decide their movements, it is predictable that the number of peaks captured would be a function of the radial sensor range. Actually, if the sensor range of every glowworm covers the whole search space, all the glowworms move to the global optimum and the local optima are neglected. Since assuming a priori information about the objective function (e.g., number of peaks and inter-peak distances) is not obtainable, it is hard to fix the neighborhood range at a value that works well for different function landscapes. For example, a selected neighborhood range rd would work comparatively better on objective functions where the minimum inter-peak distance is more than rd rather than on those where it is less than *rd* [10]. Consequently, the GSOA applies an adaptive neighbourhood range to detect the presence of multiple peaks in a multimodal function landscape. Thus, the neighbourhood range can be updated as:

$$r_d^i(t) = min\{r_s, max\{0, r_d^i(t) + \beta(n_t - |N_i(t)|)\}\} \quad (4)$$

Besides that, there are constant values including parameters one used in this study:

$\rho = 0.4, \gamma = 0.6, \beta = 0.08$ and $\ell_0 = 5$

III. Case study

Table 1 displays the components used for assembly of a car engine pump valve as a case study of this work. Table 2 displays the liaisons between two possible assembly components. The binary numbers 0 and 1 indicate the impossibility and possibility, individually. Table 3 shows the average time taken for assembly between two possible components, which is calculated in seconds.

TABLE I. Assembly components of the car engine pump valve

Number of Component	***Component Names***
1	Arm
2	Body
3	Bolt
4	Bolt-Shaft
5	Key
6	Nut-Shaft
7	Nut3
8	Plate
9	Retainer
10	Shaft
11	Sleeve1
12	Sleeve2
13	Washer-shaft
14	Washer3

TABLE II. The priority matrix in liaisons between two possible assembly components of the car engine pump valve

Comp.	***Sleeve (11)***	***Plate (8)***	***Nut-Shaft (6)***	***Bolt-Shaft (4)***	***Washer-Shaft (13)***	***Shaft (10)***	***Body (2)***	***Arm (1)***	***Sleeve (12)***	***Retainer (9)***	***Bolt (3)***	***Key (5)***	***Washer (14)***	***Nut (7)***
Sleeve (11)	0	0	0	0	0	0	0	0	0	0	0	0	0	0
Plate (8)	1	0	0	0	0	0	0	0	0	0	0	0	0	0
Nut-Shaft (6)	1	1	0	0	0	1	0	0	0	0	0	0	0	0
Bolt-Shaft (4)	1	1	0	0	0	1	0	0	0	0	0	0	0	0
Washer-Shaft (13)	1	1	0	0	0	1	0	0	0	0	0	0	0	0
Shaft (10)	1	1	0	1	0	0	0	0	0	0	0	0	0	0
Body (2)	1	1	1	1	1	1	0	0	0	0	0	0	0	0
Arm (1)	1	1	1	1	1	1	1	0	0	0	0	0	0	0
Sleeve (12)	1	1	1	1	1	1	1	1	0	0	0	0	0	0
Retainer (9)	1	1	1	1	1	1	1	1	0	0	0	0	0	0
Bolt (3)	1	1	1	1	1	1	1	1	0	0	0	0	0	0
Key (5)	1	1	1	1	1	1	1	1	1	1	1	0	0	0
Washer (14)	1	1	1	1	1	1	1	1	1	1	1	0	0	0
Nut (7)	1	1	1	1	1	1	1	1	1	1	1	0	0	0

TABLE III. Average assembly time (second) between two possible components of the car engine pump valve

Comp.	***Sleeve (11)***	***Plate (8)***	***Nut-Shaft (6)***	***Bolt-Shaft (4)***	***Washer-Shaft (13)***	***Shaft (10)***	***Body (2)***	***Arm (1)***	***Sleeve (12)***	***Retainer (9)***	***Bolt (3)***	***Key (5)***	***Washer (14)***	***Nut (7)***
Sleeve (11)	0	2	2	1	1	3	4	2	3	1	4	5	5	4
Plate (8)	2	0	5	2	2	6	6	3	10	3	2	2	5	2
Nut-Shaft (6)	3	3	0	2	2	3	3	1	3	4	5	3	4	5
Bolt-Shaft (4)	2	5	5	0	11	15	4	4	4	4	4	5	8	2
Washer-Shaft (13)	4	4	10	10	0	7	13	2	5	6	5	4	6	3
Shaft (10)	3	5	2	7	7	0	2	13	7	8	6	6	4	5
Body (2)	4	8	1	3	3	4	0	3	18	7	7	7	6	8
Arm (1)	6	7	2	8	8	5	6	0	6	6	4	3	5	6
Sleeve (12)	8	6	4	5	5	8	7	17	0	52	2	4	7	4
Retainer (9)	9	8	6	2	2	7	18	7	4	0	42	2	8	7
Bolt (3)	7	6	8	8	8	13	6	5	3	3	0	5	5	5
Key (5)	4	14	18	7	7	4	4	3	6	4	5	0	4	4
Washer (14)	2	6	6	2	9	6	2	4	1	5	6	4	0	1
Nut (7)	4	3	5	4	4	5	3	4	8	7	3	4	1	0

IV. Results

Fig.2 shows the results of assembly time gained from 5 generations using the GSOA method. Each result shows the assembly time in response to each assembly sequence of the car engine valve. As shown in Fig.2, generation 1 has the highest assembly time which is 520 seconds. The assembly time decreases in generation 2 to be 510 seconds. The result from generation 3 to generation 5 remains the same towards a steady assembly time of 500 seconds for all the assembly sequences.

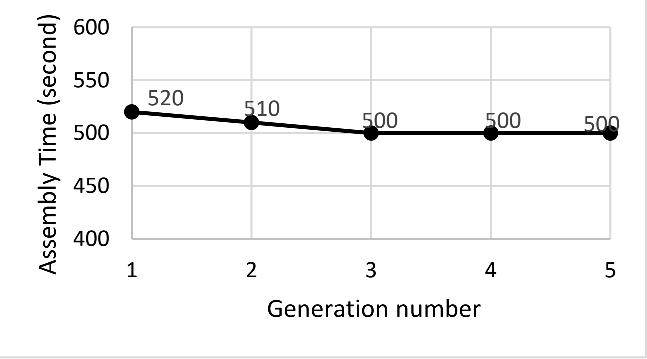

Fig. 2. Assembly time obtained using the GSOA in response to the glowworms number

Fig.3 shows the comparison in assembly time between the calculated numerical result and the computerised result, which relate to the generation number from 1 to 5. The assembly time gained from generation 1 is 520 seconds from the computerised result, which is higher than 504 seconds gained from the numerical result. In generation 2, the gap in assembly time gained from both results is comparatively smaller. In generation 3, the assembly time has the same value of 500 seconds obtained from both the numerical result and the computerised result. Afterward, the assembly time from the computerised result is 500 seconds, compared to 497 seconds obtained from the calculated numerical result. Therefore, the minimal assembly time can be seen in generations 4 and 5.

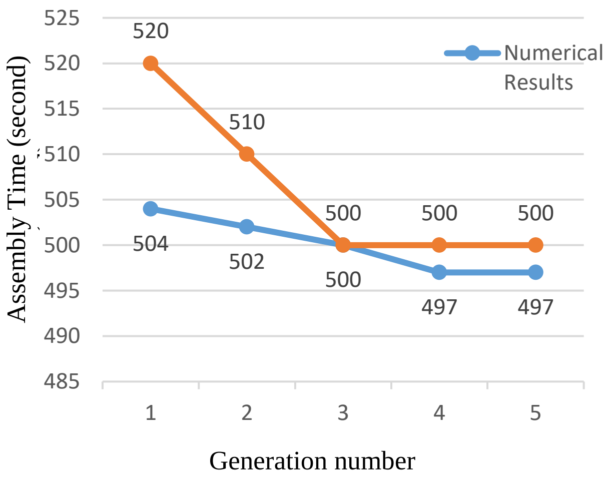

Fig. 3. Comparison in assembly time between the numerical result and computerised result of GSOA in response to generation number

V. Conclusions

This paper presented a study into feasibility and applicability using the GSOA approach for resolving the assembly sequence optimisation problem of a car engine pump valve. The GSOA was developed using Java for determining an optimal assembly sequence with the minimal assembly time. The generated result shows that the GSOA can be a useful decision-making tool in obtaining an optimal or near-optimal assembly sequence for product designers, although the assembly time can vary if the assembly is performance by human workers as the human performance is also a major human factor that may affect the assembly time. Thus, the computer modelling simulation approach may need to be applied.

References

[1] C.C. Chang, H.E. Tseng, L.P. Meng, Artificial immune systems for assembly sequence planning exploration, Eng. App. of Artif. Intell, 22, pp. 1218-1232, 2009.

[2] L.M. Ou, X. Xu, Relationship matrix based automatic assembly sequence generation from a CAD model, Comp.-Aided Design, 45, pp. 1053-1067, 2013.

[3] K. Huang and Y. Q. Zhou, A Novel Chaos Glowworm Swarm Optimization Algorithm for Optimization Functions, Proc. of Springer, Bio-Inspired Computing and Applications, Volume 6840 of the series Lecture Notes in Computer Science pp. 426-434, 2012.

[4] L. He, X. Tong and Q. Wang, Glowworm Swarm Optimization Algorithm with Improved Movement Pattern, Proc. of the IEEE 6th International Conference on Intelligenet Networks and Intelligent Systems, 2013.

[5] L. He, X. Tong and Q. Wang, Glowworm Swarm Optimization Algorithm Based on Hierarchical Multi-subgroups, Journal of Information & Computational Science 10: 4, pp.1245-1251, 2013.

[6] M. Marinaki andY. Marinakis, A Glowworm Swarm Optimization Algorithm for the Vehicle Routing Problem with Stochastic Demands, Proc. of ELSEVIER Expert Systems With Applications 46, 145–163, 2016.

[7] Y. Yang, Y. Zhou and Q. Gong, Hybrid Artificial Glowworm Swarm Optimization Algorithm for Solving System of Nonlinear Equations, Journal of Computational Information Systems 6:10, pp. 3431-3438, 2010.

[8] Z. Yu and X. Yang, Full Glowworm Swarm Optimization Algorithm for Whole-Set Orders Scheduling in Single Machine, The ScientificWorld Journal, Volume, Article ID 652061, pp. 1-6, 2013.

[9] D. Pengzhen, T. Zhenmin and S. Yan, A Quantum Glowworm Swarm Optimization Algorithm based on Chaotic Sequence, International Journal of Control and Automation Vol.7, No.9, pp.165-178, 2014.

[10] K. N. Krishnanand and D. Ghose, Glowworm swarm optimization for simultaneous capture of multiple local optima of multimodal functions, Swarm Intelligence, vol. 3, No. 2, pp. 87–124, 2009.