The Anglerfish Algorithm: A Derivation of Randomized Incremental Construction Technique for Solving TSP

Mei F. Pook mfpook@siswa.um.edu.my
Natural Computing Laboratory, Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, 50603, Kuala Lumpur, Malaysia

Effirul I. Ramlan effirul@um.edu.my
Natural Computing Laboratory, Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, 50603, Kuala Lumpur, Malaysia

Malaysia Genome Institute, National Institutes of Biotechnology Malaysia, Ministry of Science, Technology and Innovation (MOSTI), Jalan Bangi, 43000 Kajang, Selangor, Malaysia

Centre of Research for Computational Sciences and Informatics for Biology, Bioindustry, Environment, Agriculture, and Healthcare (CRYSTAL), University of Malaya, 50603 Kuala Lumpur, Malaysia

Abstract

Combinatorial optimization focuses on arriving at a globally optimal solution given constraints, incomplete information and limited computational resources. The combination of possible solutions are rather vast and often overwhelmed the limited computational power. Smart algorithms have been developed to address this issue. Each offers a more efficient way of traversing the search landscapes. Inadvertently, clogging the field with specialized algorithms for every new optimization problems. Critics have called for a realignment in the bio-inspired metaheuristics field. Inspired by the the Anglerfish population (found in the deep sea), we proposed an algorithm that simplified the search operation to only randomize population initialization. This relieves the need of complex operators normally imposed in the current meta-heuristics pool. The algorithm is more generic and adaptable to any optimization problems. A unique method of reproduction by the anglerfish provides a simple and elegant way to randomly generate good solutions. Benchmarking is conducted using the Traveling Salesman Problem (TSP). The progression of our experiments charts the development of the anglerfish algorithm, and the results are comparable with advanced meta-heuristic algorithms. Hence, suggesting that arbitrary exploration is practicable as an operator to solve optimization problem.

Keywords

Combinatorial optimization, Bio-inspired algorithms, Random incremental construction, Traveling salesman problem

1 Introduction

Various methods and algorithms have been proposed to solve optimization problems. As of late, bio-inspired metaheuristics are among the favorites. On the one hand, this favoritism is highly influenced by the effectiveness of the search mechanism and the availability of powerful computer to generate potential solutions in a reasonable amount of time. On the other, these solutions are impractical to be generated using deterministic approaches due to the incomplete problem definition and vast search landscape characteristics of the potential solutions.

Evidently, as the biological narratives grew, together with the advancement of computing, the pool of bio-inspired algorithms become excessive. As a consequence, knowledge creation stopped and the sophistication of the mechanisms remain hidden behind their metaphors and were never throughly discussed [1, 2, 3]. Issues related to the conflicting representation of the biological narrative which obfuscate the current knowledge and incessant competition with other algorithms further degraded the field. For instance, criticism on the harmony search algorithm [3, 4], and firefly algorithm [5] as being redundant copies of earlier bio-inspired algorithms (i.e., Evolutionary strategies and particle swarm) [2] is a common occurrence, emphasizing on the issue of redundancy populating the ever expanding pool of bio-inspired algorithms.

Despite these criticisms, the expansion of the field is rather positive. The availability of these algorithms in tackling many optimization problems is fundamental to its existence (i.e., why we need algorithms in the first place). As a result, we can wisely choose the most suitable algorithm given the specific needs of the problem. Therefore, the problem is not how many, but how good are those algorithms. More importantly, whether these algorithms add any novelty to the body of knowledge in the metaheuristics field. Ideally, every new algorithm has to be thoroughly examined. This is to prevent redundancy, since it is liable to the pseudo-novelty trap. When designing bio-inspired metaheuristics, we have millions of species in the planet and consequently, we have millions of metaphors that might overlap biologically. As suggested in [1], metaheuristics should be explicitly identified, stripped down to the their essentials, and analyzed, to reveal their mechanisms in arriving to the solutions.

Metaheuristics is a relatively new field, however, the adoption of metaheuristics in solving combinatorial optimization problems has attracted massive attention [6]. Bioinsipired algorithms that closely mimic biological systems are synonymous with this field. At the forefront of is Genetic Algorithm (GA) [7], Evolutionary Programming (EP) [8] and Evolution Strategies (ES) [9]. The underlying idea behind these algorithms is fundamentally similar. Using natural selection as a key operator, iterative improvement of the population occurs through the survival-of-the-fittest principal.

Briefly, a set of candidate solutions is randomly generated, and based on a quality function to be maximized, a fitness is measured. Using this fitness measure, selected candidates undergo recombination or mutation (i.e., at times both operators) to generate the next generation of candidate solutions, producing offsprings for the new population. Both population are re-evaluated to produce parents for the next iteration. The process is repeated until a candidate with sufficient quality is produced or a computational limit is reached. Further sophistications related to gender-biases have been introduced to improve on the natural selection process. There are gender-based selection whereby gender (female or male) value is assigned to each candidate alternatively in a population (sorted in descending fitness values) [10] and the introduction of selection pressure on the two gender population whereby only one gender of the population goes through competition in order to produce offsprings [11]. Accordingly, these inclu-

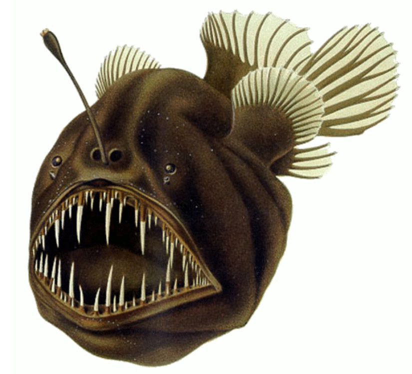

Figure 1: The Humpback anglerfish (*Melanocetus johnsonii*), a species of black sea devil (*Melanocetidae*). Adapted from August Brauer (1863ñ1917): Die Tiefsee-Fische. I. Systematischer Teil.. In C. Chun. Wissenschaftl. Ergebnisse der deutschen Tiefsee-Expedition 'Valdivia', 1898-99, 1906.

sions produced significant improvements when compared to their corresponding basic version, however, the procedural structure of each remains (i.e., iterative improvement of a randomly generated set of individuals).

Compared to the conventional initialize-and-then-optimize-procedure, we are proposing a random selection procedure, whereby only the initialization step occurs during each iteration. This highlights the importance of randomness as examplified in Greedy Randomized Adaptive Search Procedures (GRASP), with elements from a list created by a greedy function added randomly in constructing a solution [12]. Following this recommendation, we introduced a simple bio-inspired algorithm based on the Humpback Anglerfish. In this study, we dissected the algorithm thoroughly to explain the mechanism behind the metaphor and demonstrated its ability to solve the popular traveling salesman problem (TSP). The Anglerfish metaphor resembles the random incremental construction (RIC) function introduced in computational geometry [13]. RIC prevents similarity and pre-mature convergence with the asymptotic bound of $O(n \log n)$ in terms of complexity. The proposed algorithm is rather minimal; using only randomized iterative population as the only operator and a direct fitness evaluation between generations. The mechanism significantly improves on the execution time, thus enabling it to become a plausible candidate for unsupervised learning intended for analytic applications.

2 The Anglerfish metaphor

The deep sea is known for its treacherous environment, e.g., freezing temperature, massive water pressure weight, the absence of solar and inadequate food sources. However, there are species that have adapted and thrived in such harsh environment, including the deep sea Humpback Anglerfish (i.e., a prime example of deep sea adaptation [14]). Anglerfish is a predator fish commonly identified by a fleshly growth on the fish head called the *esca* (Refer Fig. 1), that acts as a lure and found on most adult females [15, 16] An interesting trait of the Anglerfish is sexual parasitism, prevalent among the sub-order called *Ceratiodei*, in which males are dwarfed and become permanently attached to their larger female counterpart.

The males Anglerfish have difficulty in finding food due to their size. Their survival depends entirely on finding a female partner for mating. Naturally, the males have big eyes and huge nostrils, primarily for detecting pheromone released by the females. The common jaw teeth (observed in most females) are replaced by a set of pincer-like denticles at the tips of the jaws for grasping on a female. The male latches onto the female. The male then becomes permanently dependent on the female for blood-transported nutrients, and the female becomes a self-fertilizing hermaphrodite. Multiple spawning may take place afterwards. This sexual dimorphism ensures that there is a supply of sperms when the female is ready to spawn. Multiple males, up to eight males in some species, can be fused.

Some key ideas were extracted from the metaphor in formulating the algorithm. These ideas are converted to the procedural and randomization mechanism of the algorithm.

- A population consists of both gender. Males presence are more frequent than females.
- Males will die when they could not find a mate. There is some possibility for immature female to die without any attachment from the male.
- Only mature females have the ability to spawn.
- The fittest mature female spawns the most. However, there is a fix number of spawns that can be generated at each time cycle to control the population.
- The spawns from the best mature female inherit her legacy. They have priority of luring males for mating.

The adaptation of the ideas into the Anglerfish algorithm is presented in Fig. 2. As depicted in the figure, the procedure consists of only two processes (i.e., initialization and re-initialization). Although loosely resembles the natural selection principal, the recombination process is clearly absent (i.e., which is vital in directed evolution). The algorithm simply resets and repopulates after each iteration. Sub-mechanisms such as mating and spawning are selective randomization process to control the initialization of the next population based on the fitness value as a guide.

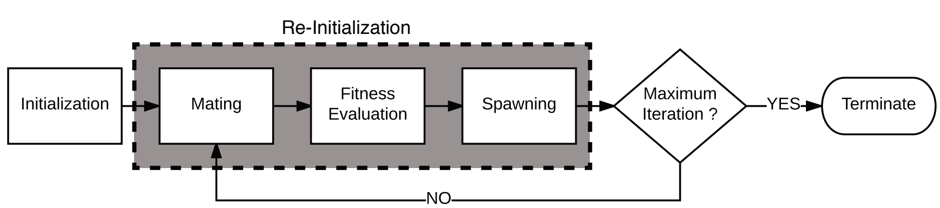

Figure 2: The Anglerfish Algorithm. The procedural step consists of intialization and re-initialisation. Initialization is a purely random process unlike re-initialization, where selective randomization occurs with embedded elitism element. The re-initialization process is comprised of mating, fitness evaluation and spawning. Algorithm is terminated once maximum epoch has been reached.

3 Formal Definition of the Anglerfish algorithm

Let N as real numbers, we define mature female, C as a set of N elements, young female F as a subset of C, and male, m an element from C.

$$C = \{1, 2, 3, ..., N\} \tag{1}$$

$$F \subset C, (N-8) < |F| < (N-1) \tag{2}$$

$$m \in C \tag{3}$$

The number 8 is chosen because up to 8 males can be attached to a female as indicated in the Anglerfish ecosystem [14]. At time cycle t=0, initialization happens with u young females and v males. There is no restriction on u and v, the only condition is that v must be a larger number than u.

$$A(0) = \{F_1, F_2, F_3, ..., F_u, m_1, m_2, m_3, ...m_v\} \tag{4}$$

Females are much rarer than males. Therefore,

$$m(t) > F(t) \tag{5}$$

Mating occurs when the male has any elements absence in young female. They merge to become a mature female. This continues until all 7 cases are merged.

$$C = F \cup m_1,\ whereby\ m_1 \notin F, |F| = N-1 \tag{6}$$

$$C = F \cup m_1 \cup m_2,\ whereby\ m_1, m_2 \notin F, |F| = N-2, m_1 \neq m_2$$

$$C = F \cup m_1 \cup m_2 \cup m_3,\ whereby\ m_1, m_2, m_3 \notin F, |F| = N-3, m_1 \neq m_2 \neq m_3$$

Males die when they could not find a mate. There is probability of a young female to remain immature due to lack of males. Eventually, she will die as well.

$$A(t) = \{C_1, C_2, C_3, ..., C_{c(t)}\} \tag{7}$$

Mature females spawn young females and young males. Spawning is skewed towards the male offsprings.

$$Pr(m) > Pr(F) \tag{8}$$

We fixed the probability of a young male to spawn at 0.8 after intial trial runs. This value can be optimized depending on a given task. By increasing the bias towards male offspring, we will effectively preserved the diversity of the population. Reversely, the bias skewed towards female offspring generation limits the randomization mechanism, influencing the exploration capability of the algorithm.

$$Pr(m) = 0.8 \tag{9}$$

$$Pr(F) = 0.2$$

Number of spawns that can be generated at each time cycle is assigned as maximum spawn number, sp. Let $S_{fittest}$ be the spawn group of the fittest mature fish, $C_{fittest}$. We denote s as the individual spawn as

$$s = m \text{ } or \text{ } F \tag{10}$$

$$S_{fittest} = \{s_1, s_2, s_3, ..., s_{sp}\}$$

$$sp = sp - r \tag{11}$$

$$S_{next\ fittest} = \{s_1, s_2, s_3, ..., s_{sp}\}$$

We denote r as the number to be reduced from sp. Each subsequent fittest fish will spawn a smaller group of sp (gradually). This iteration will continue until sp =0. The three dynamic parameters that can be refine for optimization are sp, r and maximum time cycle T as the termination criterion. All three variables effect the performance of the algorithm depending on the optimization problem at hand.

4 The Anglerfish algorithm

Similar to the existing population based optimization algorithms, the algorithms starts with the initialization phase. During initialization, only young females and young males are created as opposed to the complete candidate solution, which in our case is the mature female. In essence, representation of the sub-problems or sub-components of the solution, similar to the procedural steps of the randomized incremental construction (RIC) technique proposed in [13]. RIC utilizes random sampling to split problems into subproblems, and then incrementally assembles the solution. These younglings are representation of sub-problems and accordingly, the incremental approach is imitated through the merging process of males with immature female.

The next phase is mating. Unlike the recombination operator found in evolutionary optimization algorithm, the mating process is a form of selective randomization applied to form the candidate solutions similar to the incremental approach in RIC. However, cifferent from RIC, the incremental steps in the anglerfish algorithm are arbitrary for each young female (F) with a maximum incremental step (sets at 8 times following the metaphore). The anglerfish combines a single female (F) with up to eight male (m). This produces a richer pool of candidates irregardless of the fitness value. In Anglerfish, mating is a part of the re-initialization process, and it is directly responsible in creating the candidate solutions instead of the recombination process to produce offsprings as commonly observed in evolutionary algorithms. Randomness is further promoted during mating to allow for a creation of diverse candidate solutions.

A key feature of population based algorithms is utilizing the neighborhood search to find the optimal solution. This is possible only if a neighborhood relation is defined in the search space. For instance, in Ant Colony Optimization (ACO), the neighborhood search are directed using pheromone as weight [17], while Particle Swarm Optimization (PSO) utilizes the positioning and velocity values to determines its flocking behaviour [18]. There is a need to define adaptive parameters to reflect the relation between agents. These parameters are constantly updated at each iteration, taking into consideration input from the sub-sequence or even the entire population. Adaptive parameters between population are discarded and do not contribute to the optimization

Algorithm 1: The basic Anglerfish (TSP) algorithm

```
Data: TSP instance
Result: find the fittest solution (fish)
initialization with 10 young females and 50 young males;
while not end of Time cycle do
    mating;
    fitness evaluation;
    sort according to descending fitness;
    maximum spawn number, sp = 100, reduction number, r=10;
    for each female fish,F from the top do
        if sp > 0 then
            f spawns sp, Pr(m)=0.8 and Pr(F)=0.2;
            sp = sp - r;
        else
            break;
        end
    end
    Time cycle=Time cycle+1;
end
```

process. Compared to common population based algorithms, Anglerfish has the ability to stumble upon quality solution at any steps even in less preferrable setting.

In adapting the anglerfish metaphor, the fittest fish gets to spawn the most and the best breed of spawn gets to mate first because they are more attractive. Following the metaphor, ranking is performed to determine the candidate solution (C) that can become parents for the next generation. To ensure the fittest fish has an advantage compared to the unfit candidate, we reduce the spawn number for the next fittest fish until a threshold is reached. The spawn limit (sp) and reduction rate (r) can be tuned to optimize the algorithm. During the spawning process, a legacy value is assigned to all female spawns. The legacy value represents the fitness order of their parent. This legacy attribute enables the spawn to have priorities during mating. Finally the algorithm checks for the end of time cycle (T) and repeats the whole process if it has not reached T. Unlike most meta-heuristics, the exploration of the search landscape is rather loose and undirected, except for the preferential treatment (priority) of the fittest candidate during mating. Further randomizations on the population are enforced to ensure diversity is preserved (i.e., during the spawning and mating phases). This randomization mechanism would negate the elitism aspect in mating to indirectly prevents local optima.

The basic version of the anglerfish algorithm (i.e., no legacy option) was implemented first on the TSP. Pseudo-code for the basic TSP Anglerfish is listed in Algorithm. 1. The legacy enable version (i.e., the advanced anglerfish algorithm), is presented in Algorithm. 2.

5 Results and Discussions

An instance of the Traveling Salesman Problem (TSP) (from the TSPLIB [19]) was selected for benchmarking (the *ulysses16*). This instance has 16 cities with their respective coordinates. For the Anglerfish TSP algorithm, young males, (m) are representing a

Algorithm 2: The legacy Anglerfish (TSP)

```
Data: TSP instance
Result: find the fittest solution (fish)
initialization with 10 young females and 50 young males;
assign similar legacy to all females;
while not end of Time cycle do
    sort according to descending legacy;
    for each female fish,f from the top legacy do
        mating;
    end
    remove all young males and young females;
    fitness evaluation;
    sort according to descending fitness;
    assign descending legacy to all fishes,fittest fish has best legacy;
    maximum spawn number sp=100, and reduction number r=10;
    for each female fish,f from the top fitness do
        if sp > 0 then
            F spawns sp, Pr(male)=0.8 and Pr(female)=0.2;
            assign F's legacy to all spawns;
            sp = sp - r;
        else
            break;
        end
    end
    Time cycle=Time cycle+1;
end
```

single city and young females, (F) are representing any 8 to 15 arbitrary ordered cities. The range of between 8 to 15 is selected based on the metaphor of having a minimum of eight males partner (that will latch to the female fish). Mating is permitted only if the city is not yet available in the females. A new city is added at any random points once mating is initiated. A female is deemed mature once all 16 cities are connected.

Fitness evaluation is performed to all mature females in the population. The fitness value is determined by calculating the route of all 16 cities, in which the fittest represents the shortest path. Re-population is performed afterwards. Priority of spawning is assigned to the fittest mature female. During spawning, young males is randomly assigned a city number of the 16 cities (with the likelihood sets to 0.8 as default). Young females inherit the route from their ancestor minus a single point (i.e., imitating a single base mutation operator common in evolutionary algorithms). These younglings are then allowed to latch to new males.

For the simulation, 10 young females and 50 young males are initialized. Five sets of simulations were conducted. The five sets differ by the time cycle T (25 time cycles, 50 time cycles, 75 time cycles, 100 time cycles and 125 time cycles). These time cycles act as a termination point of the algorithm. These cycles were selected based on pretrial runs while developing the algorithm. Each set of simulation consists of 30 runs and the optimal solution is identified at 6859, as quoted from the online TSPLIB [1]. Both

[1] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html

Anglerfish TSP algorithms (with and without the legacy attribute) were tested.

5.1 The Anglerfish TSP without the legacy attribute

Benchmarking is conducted on the basic version of the Anglerfish TSP algorithm. We are excluding the legacy attribute to evaluate the performance of the exploration mechanism. The population simply resets after the first initialization without ranking and assignment of the legacy attribute. Table 1 depicts the best and mean results from the

No. of Iteration	Best Result	Mean Result	Std. Dev.	Std. Error
25	7002	7536.13	248.2	45.3
50	6875	7130.80	146.7	26.8
75	6859	7027.46	106.8	19.5
100	6859	6988.96	106.4	19.4
125	6859	6961.56	86.9	15.9

Table 1: Results for the Anglerfish TSP without the legacy attribute (or Basic Anglerfish TSP). Optimal solution of 6859 were generated from 75, 100 and 125 cycles.

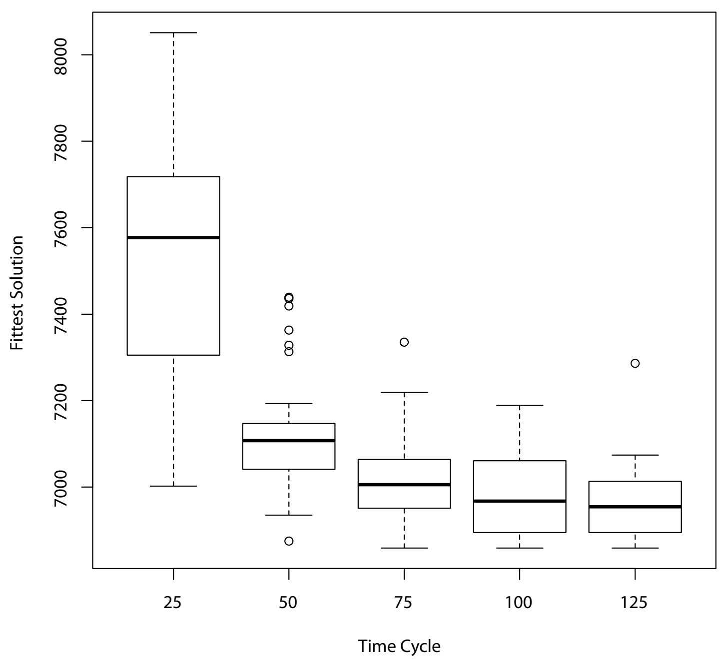

Figure 3: Results distribution for the candidates in Table 1. Aside from cycle 25 and 50, the remaining cycles (75, 100 and 125) showed consistent means that hover approximately within 7000.

30 runs. The distribution of the solutions is presented in Fig. 3. Runs were conducted with the spawn number (sp) sets to 100, this value is deducted with $r = 10$ from the previous run spawn number for subsequent runs.

The mean results consistently improved in correlation to the number of iterations. The dispersion of the solution and the standard error were reduced. In the absence of any directed evolution mechanism to converge the population, randomization takes central role, thus corresponding directly to the improvement of the exploration with an increase of iteration number. The algorithm produces better solution as more individuals are initialized. This is also reflected in the result of the best solution, where the 25 iterations run was only able to produce 7002 (after 30 trials), an outlier to the 6859 optimal solution generated from 75, 100 and 125 iterations. The value 6859 is the optimal solution for this instance.

Since the Anglerfish algorithm preserved the population diversity, the population is not directed to converge to only sets of optimal individuals. As illustrated in Fig. 3, the mean results are relatively within the optimal solutions, with presence of a few outliers. We observed an improvement in the density of the population corresponding to the increase of the iterations. These occurred despite the absence of mechanism to converge the population following the underlying principal of RIC - as designed.

5.2 The Anglerfish TSP with the legacy attribute

The legacy attribute adaptation of the Anglerfish metaphor loosely mimics the elitist mechanism commonly found during the selection process in popular evolutionary algorithms. This attribute is introduced to all females. Based on the metaphor, the fittest mature female will have the highest legacy value and this attribute is inherited by subsequent generations (from the female spawns). Priority is given to the young females based on the attribute value. With the introduction of this attribute, young females with good legacy will be more attractive to the young males, thus allowing her to latch to her mates first. Ranking and legacy attribute assignment are embedded into the basic Anglerfish TSP.

No. of Iteration	Best Result	Mean Result	Std. Dev.	Std. Error
25	6976	7254.6	219.1	40.0
50	6870	7005.3	121.0	22.1
75	6859	6920.0	42.1	7.7
100	6859	6900.0	40.1	7.3
125	6859	6892.7	31.1	5.7

Table 2: Results for the legacy Anglerfish (TSP). Optimal results were generated for cycle 75, 100 and 125; as observed in Table 1. However, cycles 25 and 50 produced better optimal values. The mean and standard deviation improved with the introduction of the legacy attribute.

Immediate improvement for the best and mean values can be observed with the legacy attribute (Refer Tab. 2). Both iteration 25 and 50 produced better optimal values as compared to previous runs. Variants within the population are smaller for all runs with better dispersion, as indicated in Fig. 4). The effect of the legacy attribute is further highlighted with the significant reduction of the standard deviation values of all population. This indicates that each population has better fitted Anglerfish females as seeds during the randomization process as compared to the complete purely arbitrary order of the basic version.

It is important to note that the improvement for the individual solutions was achieved by facilitating better seeds for randomization. In contract with the conven-

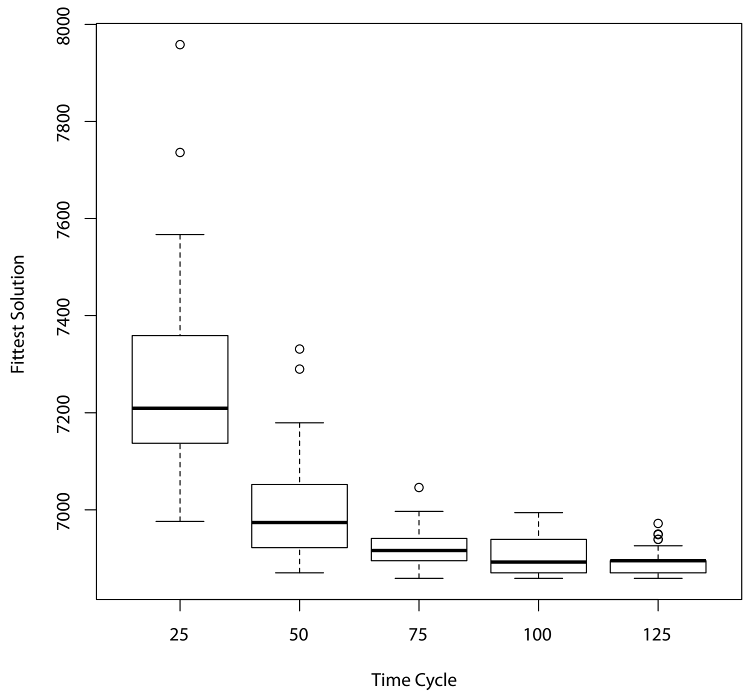

Figure 4: Result Distribution for the legacy Anglerfish (TSP). Population dispersion improved in all cycles especially for cycles 75, 100 and 125. The same cycles that performed in the basic version, however the mean improved to approximately $\pm$ 20 points between the three cycles.

tional "selection" phase employed in most bio-inspired algorithms. The Anglerfish maintain all individuals, however the legacy attribute allows mating to be prioritized, thus allowing more suitable males to latch first with more attractive females. The luring process remains random. Unlike conventional "selection" and "recombination" strategies that forced fittest individuals to become parent, enabling better offspring generation.

Mean processing time for all runs with the legacy attributes are marginally higher than the basic Anglerfish algorithm. Correspondingly, increasing the time cycle (T) directly affect the processing time as depicted in Fig, 5. Increasing the time cycle allows for more candidate solutions to be generated and promote a more thorough exploration. Depending on the computational power available, increasing the cycle time, might not be the best option. Similar exploration capability can be achieved through the utilization of the spawn number (sp) and reduction number (r).

In principal, both the spawn number sp and reduction number r are able to affect the diversity of the candidate solution, thus allowing better results to be generated using smaller time cycle T. Although the optimization of sp and r values can reduce the time cycle T, the actual processing time might not differ by much, because the re-initialization process that involves both mating and spawning will take longer time to

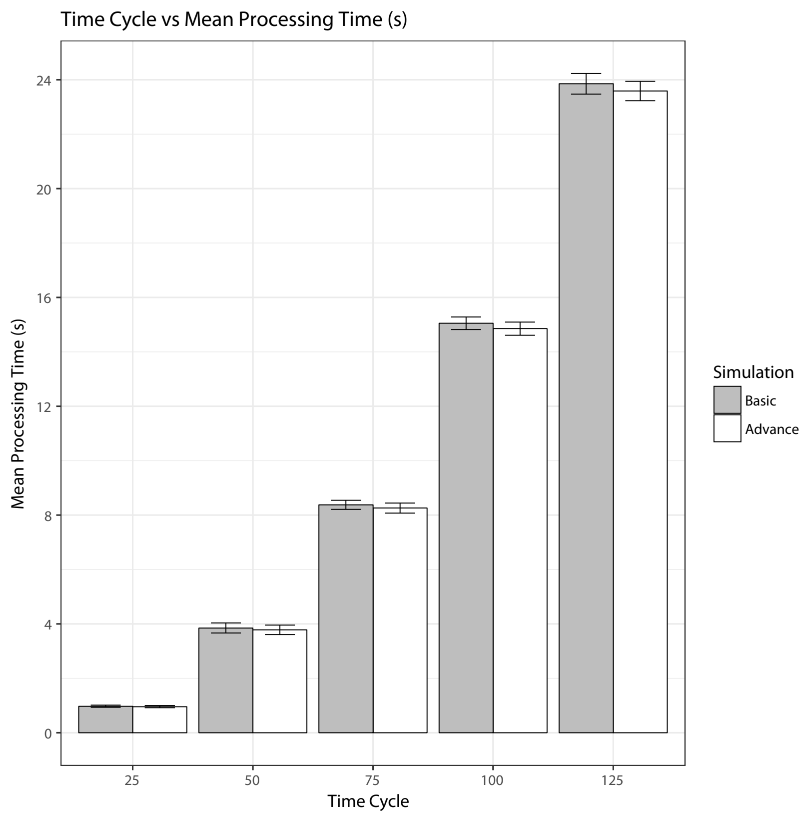

Figure 5: The mean processing time between the basic and advanced Anglerfish algorithms for each cycles. All runs were conducted on an Intel Core i7-4790 3.6GHz Quadcore machines with 8GB RAM.

complete. Three seperate runs were conducted to investigate the influence of both sp and r in determining the solutions by assigning sp = 500 and r = 50 for the first run, sp = 700 and r = 50 for the second and sp = 1000 and r = 100 for the third.

Compared to the legacy run (Tab. 2, the effect of tuning both sp and r resulted with better candidate solutions. As observed in Tab. 3, the increase of sp to 500 allows the optimal solution to be generated in only 50 iterations. The previous best solution using the legacy mode was stuck at 6870 and not the optimal solution of 6859. Furthermore, the variance between candidate solution is significantly better with 6894.4 as the mean average. This is further indicated by the smaller standard deviation value (i.e., 34.99 as compared to 219.1). Similar results can be obversed for the sp = 700 and sp = 1000. Evidently, further analysis is required to determine the impact of both sp and r parameters for the proposed algorithm. Tuning both paramaters does influence the exploration capability of the algorithm, and could potentially reduced the no. of iteration (time cycle). From our limited observation, the trade-off between iterations and re-initialization in terms of actual computational time is not as significant. Considering the abundance of parallel computing resources available currently. However, fine tuning of the sp, r

sp	r	No. of Iteration	Best Result	Mean Result	Standard Deviation
500	50	50	6859	6894.4	34.99
		100	6859	6881.7	20.87
700	50	50	6859	6883.6	31.05
		100	6859	6885.8	24.25
1000	100	50	6859	6886.0	26.39
		100	6859	6885.4	27.06

Table 3: Results for the legacy Anglerfish (TSP) with sp=500 and r=50, sp=700 and r=50, and sp=1000 and r=100. Optimal results are obtained in time cycle 50 as compared with previous legacy runs depicted in Tab. 2. Both time cycles recorded better dispersion.

TSP Instance	ACS	GA	EP	SA	AG	Anglerfish
oliver30	420	421	420	424	420	420

Table 4: Results for the *oliver30* TSP benchmarking. The optimal values for Ant Colony System (ACS), Genetic Algorithm (GA), Evolutionary Programming (EP), Simulated Annealing (SA), hybrid algorithm of Simulated Annealing and Genetic Algorithm (AG) are extracted directly from Table 3 in [17]. These values are the best optimal values recorded during the simulation. The optimal value for the Anglerfish algorithm (Anglerfish) was generated from the simulation, detailed in Tab. 5. Only ACS, EP, AG and the Anglerfish managed to arrive at the optimal value.

and time cycle T is necessary to influence the optimal outcome, and requires a more detailed investigation.

5.3 Benchmarking with other Algorithms

The performance of the Anglerfish TSP algorithm are then tested against well-known metaheuristics. Benchmarking is conducted using *oliver30* [17]. For replication purpose, *oliver30* is selected because this instance has an optimal value and published results for the common algorithms. Benchmarking is conducted only for these results as rerunning the experiment is difficult due to the lack of available codes, and biases that might be introduced during recoding of these algorithms.

The coordinates of *oliver30* is available online[2]. The optimal solution of *oliver30* is 420. For this experiment, the Anglerfish TSP algorithm is configured with 30 young females and 150 young males, maximum males that can attach to a female remains at 8, with the sp value sets at 700, subsequent next best fish deduction sets to $r = 50$ from the previous spawn number, and population control of 10,000 fishes. These values are configured after pretrial runs. Adjustments were made according to the number of instances involved (i.e., from 16 to 30 cities).

Benchmarking is performed only on the optimal solution based on the data available from [17]. Table 4 summarized the optimal value generated from Ant Colony System (ACS), Genetic Algorithm (GA), Evolutionary Programming (EP), Simulated Annealing (SA), hybrid of SA and GA (AG) and the proposed Anglerfish algorithm. As mentioned above, the optimal solution for *oliver30* is 420, and only ACS, EP, AG

[2]http://stevedower.id.au/blog/research/oliver-30/

No. of Iteration	Best Result	Mean Result	Std. Dev.	Std. Error
400	420	452	22.5	4.1

Table 5: Results for the *oliver30* runs from the legacy Anglerfish (TSP) algorithm after 400 cycles. The number of iterations was increased to 400 to accommodate for the number of cities involved. The number of individuals allowed after each cycle are kept at 10000.

and Anglerfish managed to produce the optimal value.

Details of the runs are listed in Table 5. Since the termination criterion is solely based on number of iterations, we have conducted trial runs to gauge the maturity of the population. Similar to previous observation, the additional nodes evidently increases the number of iterations. The number of iterations was set to 400 cycles based on the trial runs conducted prior to the simulation. After 400 cycles, the population has the optimal value of 420, with relatively better dispersion of fishes (mean of 452 $\pm$ 4.1) when compared to the optimal solution. Standard deviation of the population is relatively low at 22.5, consistent with previous our findings with legacy attribute assignment.

Benchmarking is then expanded to 52 cities (*berlin52*) to evaluate on the scalability of the proposed algorithm. As indicated in TSPLIB, the optimum solution for the *berlin52* is 7542. The same configurations as described for the *oliver30* version were applied. Summary of the results is listed in Table 6. The Anglerfish TSP algorithm was able to generate the optimal value of 7542 after 4000 iterations. Since the number of cities tripled as compared to the previous benchmark, we have to extend the run cycles accordingly. For this experiment, we ran between 600 to 4000 iterations with varying outcomes (Refer Table 7). The optimal values fluctuate inconsistently between runs,

TSP Instance	Basic DCS	Improved DCS	DPSO	ACS	ACE	Anglerfish
berlin52 (Best)	7542	7542	7542	7542	7542	7542

Table 6: Results for *berlin52* TSP benchmarking. Optimal values for the common metaheuristics were extracted from Ouaarab et al. [20, Table 2] for Basic and Improved Discrete Cuckoo Search (DCS), and Ouaarab et al. [20, Table 5] for Discrete Particle Swarm Optimization (DPSO), from Escario et al. [21, Table 7] for Ant Colony System (ACS) and Ant Colony Extended (ACE).

indicating no substantial pattern for the termination criterion (i.e., of better optimal values as the cycle increases). However, the mean values in the population are consistent. In essence, this shows the effectiveness of the randomization procedure, and at the same time highlights the importance of the stopping criterion (a common problem in combinatorial optimization algorithms). A further comparison against the common optimization strategy is omitted since performance analytics of these algorithms are missing from the references and recoding the codes would introduce unnecessary programming biases.

In both cases (*oliver30* and *berlin52*), the proposed TSP Anglerfish algorithm managed to arrive to the optimal results. Considering the minimal computational time involved for both runs, and the plausible adaptation to parallel runs, we believe that the

No. of Iteration	**Best Result**	**Mean Result**	**Std. Dev.**	**Std. Error**
600	7775	8558.5	375.7	68.6
1000	7922	8525.4	402.9	73.6
1500	7854	8559.4	362.9	66.3
2000	8142	8637.3	346.4	63.3
3000	7764	8387.8	396.1	72.3
4000	7542	8447.9	391.8	71.5

Table 7: Results for the *berlin52* runs from the legacy Anglerfish (TSP) algorithm. Since there is no reference point and the size of the cities involved, multiple runs were executed using between 600 to 4000 cycles as termination points. The optimal solution was generated after 4000 runs. As mentioned previously, the number of individuals were controlled at 10000.

proposed algorithm would be able to generate unconventional solutions as compared to the gradual improvement strategy employed by most optimization algorithms. Although there is no rule of thumb, a large time cycle would be adequate for the algorithm to stumble on the optimal values. This is suitable as the algorithm is computational inexpensive to run (i.e., and can be executed in parallel environment).

6 Conclusion

Extensive computational power is now available in the form of multi-core processors, where instruction can be executed in parallel. Therefore, the need of complicated algorithms to speed up computational is no longer necessary. To leverage on such technology, we need to be able to run simple instructions concurrently for multiple times. The proposed Anglerfish algorithm fits this description. The algorithm traverses the search landscape using random sampling without any complicated procedural routines. Issues such as the termination criterion and the efficacy of the algorithm remained, however, the proposed algorithm can become a blueprint towards realigning the bio-inspired metaheuristics field in producing simple and elegant solution, leveraging on the current computational platform for future autonomous optimization.

7 Additional Information

The Anglerfish TSP algorithm is available for downloads at `https://github.com/meifoong/AnglerfishAlgorithm`

References

[1] Kenneth Sörensen. Metaheuristics—the metaphor exposed. *International Transactions in Operational Research*, 22(1):3–18, 2015.

[2] Michael A Lones. Metaheuristics in nature-inspired algorithms. In *Proceedings of the Companion Publication of the 2014 Annual Conference on Genetic and Evolutionary Computation*, pages 1419–1422. ACM, 2014.

[3] Dennis Weyland. A critical analysis of the harmony search algorithm—how not to solve sudoku. *Operations Research Perspectives*, 2:97–105, 2015.

[4] Zong Woo Geem, Joong Hoon Kim, and GV Loganathan. A new heuristic optimization algorithm: harmony search. *Simulation*, 76(2):60–68, 2001.

[5] Xin-She Yang. Firefly algorithms for multimodal optimization. In *International Symposium on Stochastic Algorithms*, pages 169–178. Springer, 2009.

[6] Kenneth Sörensen, Marc Sevaux, and Fred Glover. A history of metaheuristics. In *Handbook of Heuristics*. Springer, 2016.

[7] JH Holland. Adaptation in natural and artificial systems. an introductory analysis with application to biology, control, and artificial intelligence. *Ann Arbor, MI: University of Michigan Press*, 1975.

[8] LJ Fogel, AJ Owens, and MJ Walsh. *Artificial intelligence through simulated evolution*. John Wiley, 1966.

[9] Manfred Eigen. *Ingo Rechenberg Evolutionsstrategie Optimierung technischer Systeme nach Prinzipien der biologishen Evolution*. mit einem Nachwort von Manfred Eigen, Friedrich Frommann Verlag, Struttgart-Bad Cannstatt, 1973.

[10] Jalel Rejeb and M AbuElhaij. New gender genetic algorithm for solving graph partitioning problems. In *Circuits and Systems, 2000. Proceedings of the 43rd IEEE Midwest Symposium on*, volume 1, pages 444–446. IEEE, 2000.

[11] Carlos Ansótegui, Meinolf Sellmann, and Kevin Tierney. A gender-based genetic algorithm for the automatic configuration of algorithms. In Ian P. Gent, editor, *Principles and Practice of Constraint Programming - CP 2009*, pages 142–157, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-04244-7.

[12] TA Feo and MGC Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133, 1995.

[13] Kenneth L Clarkson and Peter W Shor. Applications of random sampling in computational geometry, ii. *Discrete & Computational Geometry*, 4(1):387–421, 1989.

[14] Theodore W. Pietsch. *Oceanic Anglerfishes: Extraordinary Diversity in the Deep Sea*. University of California Press, 2009. ISBN 0520255429.

[15] Theodore W Pietsch. Dimorphism, parasitism, and sex revisited: modes of reproduction among deep-sea ceratioid anglerfishes (teleostei: Lophiiformes). *Ichthyological Research*, 52 (3):207–236, 2005.

[16] Masaki Miya, Theodore W Pietsch, James W Orr, Rachel J Arnold, Takashi P Satoh, Andrew M Shedlock, Hsuan-Ching Ho, Mitsuomi Shimazaki, Mamoru Yabe, and Mutsumi Nishida. Evolutionary history of anglerfishes (teleostei: Lophiiformes): a mitogenomic perspective. *BMC Evolutionary Biology*, 10(1):1, 2010.

[17] Marco Dorigo and Luca Maria Gambardella. Ant colonies for the travelling salesman problem. *BioSystems*, 43(2):73–81, 1997.

[18] RC Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Proceedings of the sixth international symposium on micro machine and human science*, volume 1, pages 39–43. New York, NY, 1995.

[19] Gerhard Reinelt. Tsplib—a traveling salesman problem library. *ORSA journal on computing*, 3(4):376–384, 1991.

[20] Aziz Ouaarab, Belaïd Ahiod, and Xin-She Yang. Discrete cuckoo search algorithm for the travelling salesman problem. *Neural Computing and Applications*, 24(7-8):1659–1669, 2014.

[21] Jose B Escario, Juan F Jimenez, and Jose M Giron-Sierra. Ant colony extended: experiments on the travelling salesman problem. *Expert Systems with Applications*, 42(1):390–410, 2015.