The Anglerfish algorithm: A derivation of randomized incremental construction technique for solving the traveling salesman problem

Pook, M. F., & Ramlan, E. (2019). The Anglerfish algorithm: A derivation of randomized incremental construction technique for solving the traveling salesman problem. *Evolutionary Intelligence*, 12(1), 11-20.
<https://doi.org/10.1007/s12065-018-0169-x>

[Link to publication record in Ulster University Research Portal](#)

Published in:
Evolutionary Intelligence

Publication Status:
Published (in print/issue): 01/03/2019

DOI:
[10.1007/s12065-018-0169-x](https://doi.org/10.1007/s12065-018-0169-x)

Document Version
Author Accepted version

General rights
Copyright for the publications made accessible via Ulster University's Research Portal is retained by the author(s) and / or other copyright owners and it is a condition of accessing these publications that users recognise and abide by the legal requirements associated with these rights.

Take down policy
The Research Portal is Ulster University's institutional repository that provides access to Ulster's research outputs. Every effort has been made to ensure that content in the Research Portal does not infringe any person's rights, or applicable UK laws. If you discover content in the Research Portal that you believe breaches copyright or violates any law, please contact pure-support@ulster.ac.uk.

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

1 Abstract

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

19 Keywords

20 Combinatorial optimization, Bio-inspired algorithms, Random incremental construc-
21 tion, Traveling salesman problem

22 1 Introduction

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

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

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

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

59 Briefly, a set of candidate solutions is randomly generated, and based on a qual-
 60 ity function to be maximized, a fitness is measured. Using this fitness measure, se-
 61 lected candidates undergo recombination or mutation (i.e., at times both operators) to
 62 generate the next generation of candidate solutions, producing offsprings for the new
 63 population. Both population are re-evaluated to produce parents for the next iteration.
 64 The process is repeated until a candidate with sufficient quality is produced or a com-
 65 putational limit is reached. Further sophistications related to gender-biases have been
 66 introduced to improve on the natural selection process. There are gender-based selec-
 67 tion whereby gender (female or male) value is assigned to each candidate alternatively
 68 in a population (sorted in descending fitness values) [10] and the introduction of selec-
 69 tion pressure on the two gender population whereby only one gender of the population
 70 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 (1863–1917): 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 exemplified 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 fleshy 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.

129 3 Formal Definition of the Anglerfish algorithm

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

$$C = \{1, 2, 3, \dots, N\} \quad (1)$$

$$F \subset C, (N - 8) < |F| < (N - 1) \quad (2)$$

$$m \in C \quad (3)$$

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

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

136 Females are much rarer than males. Therefore,

$$m(t) > F(t) \quad (5)$$

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

$$C = F \cup m_1, \text{ whereby } m_1 \notin F, |F| = N - 1 \quad (6)$$

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

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

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

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

141 Mature females spawn young females and young males. Spawning is skewed to-
142 wards the male offsprings.

$$Pr(m) > Pr(F) \quad (8)$$

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

$$Pr(m) = 0.8 \quad (9)$$

$$Pr(F) = 0.2$$

148 Number of spawns that can be generated at each time cycle is assigned as max-
 149 imum spawn number, sp . Let $S_{fittest}$ be the spawn group of the fittest mature fish,
 150 $C_{fittest}$. We denote s as the individual spawn as

$$s = m \text{ or } F \quad (10)$$

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

$$sp = sp - r \quad (11)$$

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

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

156 4 The Anglerfish algorithm

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

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

177 A key feature of population based algorithms is utilizing the neighborhood search
 178 to find the optimal solution. This is possible only if a neighborhood relation is defined
 179 in the search space. For instance, in Ant Colony Optimization (ACO), the neighbor-
 180 hood search are directed using pheromone as weight [17], while Particle Swarm Opti-
 181 mization (PSO) utilizes the positioning and velocity values to determines its flocking
 182 behaviour [18]. There is a need to define adaptive parameters to reflect the relation
 183 between agents. These parameters are constantly updated at each iteration, taking into
 184 consideration input from the sub-sequence or even the entire population. Adaptive pa-
 185 rameters 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)

```

1 initialization with 10 young females and 50 young males;
2 while not end of Time cycle do
3   mating;
4   fitness evaluation;
5   sort according to descending fitness;
6   maximum spawn number,  $sp = 100$ , reduction number,  $r=10$ ;
7   for each female fish,  $F$  from the top do
8     if  $sp > 0$  then
9        $f$  spawns  $sp$ ,  $Pr(m)=0.8$  and  $Pr(F)=0.2$ ;
10       $sp = sp - r$ ;
11     else
12       break;
13     end
14   end
15   Time cycle=Time cycle+1;
16 end

```

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

188 In adapting the anglerfish metaphor, the fittest fish gets to spawn the most and
 189 the best breed of spawn gets to mate first because they are more attractive. Follow-
 190 ing the metaphor, ranking is performed to determine the candidate solution (C) that
 191 can become parents for the next generation. To ensure the fittest fish has an advan-
 192 tage compared to the unfit candidate, we reduce the spawn number for the next fittest
 193 fish until a threshold is reached. The spawn limit (sp) and reduction rate (r) can be
 194 tuned to optimize the algorithm. During the spawning process, a legacy value is as-
 195 signed to all female spawns. The legacy value represents the fitness order of their par-
 196 ent. This legacy attribute enables the spawn to have priorities during mating. Finally
 197 the algorithm checks for the end of time cycle (T) and repeats the whole process if it
 198 has not reached T . Unlike most meta-heuristics, the exploration of the search land-
 199 scape is rather loose and undirected, except for the preferential treatment (priority) of
 200 the fittest candidate during mating. Further randomizations on the population are en-
 201 forced to ensure diversity is preserved (i.e., during the spawning and mating phases).
 202 This randomization mechanism would negate the elitism aspect in mating to indirectly
 203 prevents local optima.

204 The basic version of the anglerfish algorithm (i.e., no legacy option) was imple-
 205 mented first on the TSP. Pseudo-code for the basic TSP Anglerfish is listed in Algo-
 206 rithm. 1. The legacy enable version (i.e., the advanced anglerfish algorithm), is pre-
 207 sented in Algorithm. 2.

208 5 Results and Discussions

209 An instance of the Traveling Salesman Problem (TSP) (from the TSPLIB [19]) was se-
 210 lected for benchmarking (the *ulysses16*). This instance has 16 cities with their respective
 211 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)

```

1 initialization with 10 young females and 50 young males;
2 assign similar legacy to all females;
3 while not end of Time cycle do
4   sort according to descending legacy;
5   for each female fish,  $f$  from the top legacy do
6     | mating;
7   end
8   remove all young males and young females;
9   fitness evaluation;
10  sort according to descending fitness;
11  assign descending legacy to all fishes, fittest fish has best legacy;
12  maximum spawn number  $sp=100$ , and reduction number  $r=10$ ;
13  for each female fish,  $f$  from the top fitness do
14    | if  $sp > 0$  then
15    |   F spawns  $sp$ ,  $Pr(\text{male})=0.8$  and  $Pr(\text{female})=0.2$ ;
16    |   assign F's legacy to all spawns;
17    |    $sp = sp - r$ ;
18    | else
19    |   | break;
20    | end
21  end
22  Time cycle=Time cycle+1;
23 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 pre-trial 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 TSPLIB¹. Both

¹<http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/STSP.html>

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

232 5.1 The Anglerfish TSP without the legacy attribute

233 Benchmarking is conducted on the basic version of the Anglerfish TSP algorithm. We
 234 are excluding the legacy attribute to evaluate the performance of the exploration mech-
 235 anism. 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.

236 30 runs. The distribution of the solutions is presented in Fig. 3. Runs were conducted
 237 with the spawn number (sp) sets to 100, this value is deducted with $r = 10$ from the
 238 previous run spawn number for subsequent runs.
 239

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 ± 20 points between the three cycles.

275 tional “selection” phase employed in most bio-inspired algorithms. The Anglerfish
 276 maintain all individuals, however the legacy attribute allows mating to be prioritized,
 277 thus allowing more suitable males to latch first with more attractive females. The luring
 278 process remains random. Unlike conventional “selection” and “recombination” strate-
 279 gies that forced fittest individuals to become parent, enabling better offspring genera-
 280 tion.

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

288 In principal, both the spawn number sp and reduction number r are able to affect
 289 the diversity of the candidate solution, thus allowing better results to be generated
 290 using smaller time cycle T . Although the optimization of sp and r values can reduce
 291 the time cycle T , the actual processing time might not differ by much, because the re-
 292 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 separate 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 observed 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 parameters 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

<i>sp</i>	<i>r</i>	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
<i>oliver30</i>	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 online². 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
<i>berlin52</i> (Best)	7542	7542	7542	7542	7542	7542

Table 6: Results for *berlin52* TSP benchmarking. Optimal values for the common meta-heuristics were extracted from Ouaraab et al. [20, Table 2] for Basic and Improved Discrete Cuckoo Search (DCS), and Ouaraab 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.

- 388 [5] Xin-She Yang. Firefly algorithms for multimodal optimization. In *International Symposium*
389 *on Stochastic Algorithms*, pages 169–178. Springer, 2009.
- 390 [6] Kenneth Sörensen, Marc Sevaux, and Fred Glover. A history of metaheuristics. In *Handbook*
391 *of Heuristics*. Springer, 2016.
- 392 [7] JH Holland. Adaptation in natural and artificial systems. an introductory analysis with ap-
393 plication to biology, control, and artificial intelligence. *Ann Arbor, MI: University of Michigan*
394 *Press*, 1975.
- 395 [8] LJ Fogel, AJ Owens, and MJ Walsh. *Artificial intelligence through simulated evolution*. John
396 Wiley, 1966.
- 397 [9] Manfred Eigen. *Ingo Rechenberg Evolutionsstrategie Optimierung technischer Systeme nach*
398 *Prinzipien der biologischen Evolution*. mit einem Nachwort von Manfred Eigen, Friedrich
399 Frommann Verlag, Struttgart-Bad Cannstatt, 1973.
- 400 [10] Jalel Rejeb and M AbuElhajj. New gender genetic algorithm for solving graph partitioning
401 problems. In *Circuits and Systems, 2000. Proceedings of the 43rd IEEE Midwest Symposium on*,
402 volume 1, pages 444–446. IEEE, 2000.
- 403 [11] Carlos Ansótegui, Meinolf Sellmann, and Kevin Tierney. A gender-based genetic algorithm
404 for the automatic configuration of algorithms. In Ian P. Gent, editor, *Principles and Practice of*
405 *Constraint Programming - CP 2009*, pages 142–157, Berlin, Heidelberg, 2009. Springer Berlin
406 Heidelberg. ISBN 978-3-642-04244-7.
- 407 [12] TA Feo and MGC Resende. Greedy randomized adaptive search procedures. *Journal of*
408 *global optimization*, 6(2):109–133, 1995.
- 409 [13] Kenneth L Clarkson and Peter W Shor. Applications of random sampling in computational
410 geometry, ii. *Discrete & Computational Geometry*, 4(1):387–421, 1989.
- 411 [14] Theodore W. Pietsch. *Oceanic Anglerfishes: Extraordinary Diversity in the Deep Sea*. University
412 of California Press, 2009. ISBN 0520255429.
- 413 [15] Theodore W Pietsch. Dimorphism, parasitism, and sex revisited: modes of reproduction
414 among deep-sea ceratioid anglerfishes (teleostei: Lophiiformes). *Ichthyological Research*, 52
415 (3):207–236, 2005.
- 416 [16] Masaki Miya, Theodore W Pietsch, James W Orr, Rachel J Arnold, Takashi P Satoh, An-
417 drew M Shedlock, Hsuan-Ching Ho, Mitsuomi Shimazaki, Mamoru Yabe, and Mutsumi
418 Nishida. Evolutionary history of anglerfishes (teleostei: Lophiiformes): a mitogenomic per-
419 spective. *BMC Evolutionary Biology*, 10(1):1, 2010.
- 420 [17] Marco Dorigo and Luca Maria Gambardella. Ant colonies for the travelling salesman prob-
421 lem. *BioSystems*, 43(2):73–81, 1997.
- 422 [18] RC Eberhart and James Kennedy. A new optimizer using particle swarm theory. In *Proceed-*
423 *ings of the sixth international symposium on micro machine and human science*, volume 1, pages
424 39–43. New York, NY, 1995.
- 425 [19] Gerhard Reinelt. Tspplib—a traveling salesman problem library. *ORSA journal on computing*,
426 3(4):376–384, 1991.
- 427 [20] Aziz Ouaraab, Belaïd Ahiod, and Xin-She Yang. Discrete cuckoo search algorithm for the
428 travelling salesman problem. *Neural Computing and Applications*, 24(7-8):1659–1669, 2014.
- 429 [21] Jose B Escario, Juan F Jimenez, and Jose M Giron-Sierra. Ant colony extended: experiments
430 on the travelling salesman problem. *Expert Systems with Applications*, 42(1):390–410, 2015.