Chapter

A Heuristically Generated Metric Approach to the Solution of Chase Problem

İhsan Ömür Bucak

Abstract

In this work, heuristic, hyper-heuristic, and metaheuristic approaches are reviewed. Distance metrics are also examined to solve the "puzzle problems by searching" in AI. A viewpoint is brought by introducing the so-called Heuristically Generated Angular Metric Approach (HAMA) through the explanation of the metrics world. Distance metrics are applied to "cat and mouse" problem where cat and mouse makes smart moves relative to each other and therefore makes more appropriate decisions. The design is built around Fuzzy logic control to determine route finding between the pursuer and prey. As the puzzle size increases, the effect of HAMA can be distinguished more clearly in terms of computation time towards a solution. Hence, mouse will gain more time in perceiving the incoming danger, thus increasing the percentage of evading the danger. 'Caught and escape percentages vs. number of cats' for three distance metrics have been created and the results evaluated comparatively. Given three termination criteria, it is never inconsistent to define two different objective functions: either the cat travels the distance to catch the mouse, or the mouse increases the percentage of escape from the cat.

Keywords: hyper-heuristics, metaheuristics, heuristically generated distance-metric, chase problem, fuzzy logic control

1. Introduction

Heuristics methods are usually employed to solve the AI search problems; however, in current approaches, heuristic algorithms are not always working. Indeed, a problem which is invalid for a heuristic approach can give successful results in an algorithmic approach. Heuristic approach, unlike algorithmic methods, does not show the exact path to reach the goal. The deficiency in question is not due to the heuristic methods but is related to the field of problem itself as well as the intuitiveness of the algorithms [1]. The features of the problems are also important in terms of heuristic programming. Solution accuracy in well-formed problems can be demonstrated by algorithmic approach. Theorem proofs can be given as an example to this kind of heuristic problems [1].

Time and memory limitation is concerned in resolving the well-formed problems by algorithmic approach on computers. For example, even though "Magic

square" or "The Eight Queens" problems, at first glance, carry some sort of algorithmic nature, examination of all the situation yields combinatorial growth. Despite the fact that there are 362,880 states under discussion for a 3×3 sized magic square problem, the state space of the 5×5 sized problem grows so large that it cannot be searched thoroughly ($1,5x10^{25}$ states). Once the problem state space grows larger, heuristic approach aims to search the solution in real-time. In light of these facts, the definition of intuitiveness according to Feldman and Feigenbaum, is given as follows [2]: "Intuitiveness once the state space of the problem becomes too large, is the usage of any rule, strategy, trick, simplification, and the other factors". Therefore, when the problem contains complexity, intuitiveness plays an important role to find the path to the solution [1].

The importance of heuristic metrics lies in guaranteeing the closest optimal solution in a short time in problems having variables with huge values. In toy problems such as an 8-puzzle, heuristic metrics which do the search, selection and optimization in a very short period of time can be seen as very critical. Optimization calculations of subatomic particles, and the use of heuristics in route finding, automation and city planning are some major applications which increase its importance. Nevertheless, the effects of investigation and analysis of solving behaviors of heuristic functions through puzzle problems can be observed easily as the puzzle size is increased. Thus, the better investigation or review possibility is provided.

This paper is organized as follows: The first section is an introduction to the research area and it describes the problem. The second section reviews heuristics, hyper-heuristics, and metaheuristics applied to solve AI problems from a broad literature perspective, including the relationships between them, while the third section introduces heuristically generated metric approach, and lays down its principles. The fourth section discusses a chasing problem between pursuers and pray. The design and development are built around fuzzy logic control to determine the paths of the pursuer and prey. The simulation results are reported and discussed in the fifth section. The last section summarizes the work with a conclusion.

2. A review of heuristics, hyper-heuristics, and metaheuristics with applications to solve AI problems

Burke et al. in their research have aimed to both seek the common goal of automating the design of heuristic metrics to solve computationally hard search problems and generalize search methodologies through the use of their introduced metric concept in solving the target problem [3]. They have classified the heuristics as "disposable" and "reusable" analogous to "on-line" and "off-line" learning, respectively. "On-line" learning heuristics learn a heuristic method while solving a given instance of a problem whereas "off-line" learning heuristics learn the method from a set of training instances to make generalization in the case of unseen examples. An example to on-line learning approach can be given as the use of reinforcement for heuristic selection, the use of hyper-heuristics as high-level search strategies and genetic algorithms in a search space of heuristics [4, 5]. Similarly, learning classifier systems, case-based reasoning, and genetic programming can be given as the examples to off-line learning [6–8]. Both heuristic approaches have their own advantages. For example, on-line learning methods or heuristics may provide a favorable structure to the search space; therefore, it can be more effective searching in the space of heuristics rather than in the space of problems directly. The other advantage may arise as an alternative solution against not having a set of

related instances to train the heuristic methods off-line in newly encountered problems. A reusable or off-line method will always have an advantage of increasing the speed of solving new instances of problems [3].

As an example to the class of on-line learning algorithms, to improve the performance of Traveling Salesman Problem (TSP), a wide range of geometric heuristics have been investigated. The prime objective in terms of searching for a useful metric in TSP is a tour or a solution quality. Tour quality is assumed "good" when a valid tour is found even though only small percentage of output tours are valid tours (≈15% for random 10-city instances in the Euclidian metric). A valid tour and minimized cost (i.e., minimum tour length) are the major ones that affect the solution quality most out of the resulting tours. Usually, the incidence of valid tours are sought to be a part of any measure of success for performance [9].

A broad range of metrics are used in machine learning to evaluate the performance of metrics. Although there are many different metrics used in machine learning, a general theory is lacking to characterize the behavior of these metrics. For example, determining which metric can or should be used in a certain application or why a particular metric is a good metric is not clear. As some metrics behave differently from some other metrics, determining characteristics of these metrics can not be made precisely. Flach in their study has aimed to build a general theory and present a formal analysis in order to characterize the behavior of machine learning metrics such that dependence on these aspects become more precise and adequate. For instance, some metrics, by nature, do not depend on the class distribution in order to determine the amount of profit incurred for correctly classified examples, or misclassification cost distribution in order to determine the amount of cost incurred. The amount of cost or profit incurred is used to define the expected yield of a model. Some metrics such as precision, information gain, weighted relative accuracy have been used to build models in machine learning. Some others such as accuracy, F-measure, or area under ROC curve have been used to evaluate models on a test set. They have also proposed the use of true and false positive rates to evaluate the performance or the quality of models. Metric has been defined in terms of the counts of these true and false positive rates in a contingency table (a.k.a confusion matrix). Contingency table must have been chosen because it allows the metric to eliminate or exclude model complexity. It is also more suited to tabulate true and false positive rates as sufficient statistics for characterizing the performance of a classifier in any target context, and for evaluating the quality of a model. The metric also considers *skew ratio* as an additional parameter to indicate a trade-off between true and false positive rates to determine the direction in which improvements are to be found. Obviously, what a metric measures is no different from what an expected skew ratio is [10].

Another research aims to discover criteria that are responsible for a good performance of rule learning heuristics, theoretically and empirically. Rule learning heuristics aims to control a trade-off between consistency and coverage and therefore aims to determine optimizing parameters. The trade-off between consistency and coverage can only be explained through rules. The objective of this work is to discover criteria that are responsible for a good performance of rule learning heuristics, theoretically and empirically, and to understand the properties of rule learning heuristics such that those properties exhibit desirable performance for a large variety of datasets [11].

The term "metaheuristics" was first coined by Glover [12] and was used to explain a higher level strategy to modify other heuristics toward solutions generated beyond the search for local optimality [13]. Furthermore, a metaheuristics can be used to obtain quality solutions to the challenging optimization problems in a reasonable amount of time through the use of randomized local search algorithms

under a certain tradeoff. Nearly, all of the metaheuristic algorithms show a tendency to work suitably for global optimization. However, metaheuristic algorithms do not guarantee to reach the optimal solutions, and do not work all the time either. There have been two main approaches to calculate the robustness of a particular solution in metaheuristic optimization. One is resampling and another one is the reuse of neighborhood solutions [14]. The first one is reliable but expensive whereas the latter one is unreliable but cheap. Mirjalili et al. have proposed a metric called confidence measure for metaheuristic optimization algorithms which aims to increase the reliability by an effective calculation of the confidence level for each solution during the optimization process. The authors have also proposed new confidence-based operations for robust metaheuristics and used these operators to design a confidence-based robust optimization algorithm which is applicable to different metaheuristics [14].

Early work was introduced on the approaches to the automatic heuristic generation during the period of late 70s and early 80s for less constrained subproblems (i.e., auxiliary problems) which were also called relaxed models [15–18]. These approaches were lacking systematic means to produce the problems and the models, and therefore they were mostly proved to be computationally expensive and inefficient [19]. Passino et al. have introduced a metric space approach to specify the "heuristic function" which is often difficult for the A^* algorithm. It was shown how to specify an admissible and monotonic heuristic function for a wide class of problem domains, which is, in general, too difficult to find one. The objective was also to reduce computational complexity or to obtain a huge computational savings in addition to the introduction of a new class of "good" heuristic functions which are admissible and monotone [20].

Rosenfeld et al. have focused on adaptive weight based heuristic approaches for dynamic and effective robot coordination to meet perceived environmental conditions such as coordination and spatial conflicts within the robot group. The authors have further stated that their robot coordination using interference metrics minimized interference and thus achieved high productivity [21].

Metric optimization problems have mostly been analyzed probabilistically as based on Euclidian instances. Nothing much has been done on the side of Non-Euclidian instances. Therefore, this has motivated Bringman et al. with a study of random metric instances for optimization problems which were obtained as based on a complete graph whose edges are assigned to random weights independently. The length of a shortest path between any two connecting nodes was specified as "distance". Then, the authors have proved structured properties of the random metric instances obtained as above. Their objective was to build good clusters. They have further used these results to analyze the approximation ratios of heuristics to match large-scale optimization problems such as TSP [22].

Akinyemi has investigated how to improve the play performance and game strategy of Ayo game through the enhancement of the knowledge of minimax search technique by using a refinement-based heuristic method [23].

Sosa-Ascencio et al. have proposed a variable solution heuristics generated by a grammar-based genetic programming framework to solve constraint satisfaction problems (CSPs) which is related to artificial intelligence. This approach is also called the newly generated grammar-based hyper-heuristic and therefore categorized under heuristic generation methodology, and is distinguished from heuristic selection methodology in that the former generates new heuristics from constituents of available heuristics whereas the latter chooses or selects available heuristics [24].

Another heuristic selection design for a grammar-based hyper-heuristic model has been applied this time to solve the two dimensional bin packing problem consisting of irregular pieces and regular objects. The objective of designing such a

model was to select heuristics to determine the piece to be packed and the object in which the piece is located [25].

Dokeroglu and Cosar have proposed a novel multistart hyper-heuristic algorithm on the grid to solve the quadratic assignment problem (QAP) which is an NP-hard combinatorial optimization problem. The QAP is defined as the problem of assigning facilities to locations where each location has varying installation costs. The goal is to find an allocation where the total cost belongs to the installation and the transportation of the required amount of materials between the facilities is minimized. The proposed algorithm has put to use hyper-heuristics to find the best solution for an optimization problem by controlling and combining the strengths of several heuristics [26].

Wu et al. have proposed an evolutionary hyper-heuristic to solve the software project scheduling problem (SPSP). In a software project, an NP-hard combinatorial optimization problem, the SPSP assigns employees to tasks where the completion time, that is, the project duration, and the cost, that is, the total amount of salaries paid are to be minimized. The objective of this work is to find most suitable search operators for the type of the problem instance considered [27].

Lozano et al., in their approach, have used a parameterized schema of metaheuristics in which each metaheuristic or hybridized basic metaheuristics is represented by the values of a set consisting of numerical parameters. Their proposed schema aids the accomplishment of a metaheuristic selection or metaheuristics combination through the selection of the parameter values within the schema. The hyper-heuristic search of metaheuristic parameters in the metaheuristic space is automated except for the initial set-up of the hyper-heuristic parameters by the user. They finally decide to use a shared-memory parameterized schema both at the hyper-heuristic and the metaheuristic level, resulting in four level parallelism to reduce the solution time with high computation cost [28].

Another work explores a generation hyper-heuristic that automatically builds a selection hyper-heuristic using a machine learning algorithm called Time-Delay Neural Network (TDNN) used to extract hidden patterns within the collected data in the form a classifier, that is, an 'apprentice' hyper-heuristics, which is then used to solve the 'unseen' problem instances. The influence of extending and enriching the information collected from the expert and fed into TDNN is explored on the behavior of the generated apprentice hyper-heuristic [29].

In another work, the performance of a hyper-heuristic in terms of the quality and size of the heuristic pool is investigated. The objective is to produce a compact subset of effective heuristics from the unnecessary large pool that can decrease the performance of adaptive approaches. A new variant of iterated local search hyper-heuristics was also proposed, which incorporates dynamic multi-armed bandits. Both the heuristic pool selection method and the hyper-heuristic variant were successfully tested on two complex optimization problems: course timetabling and vehicle routing [30].

Whether sophisticated learning mechanisms are always necessary for hyper-heuristics to perform well has also been analyzed. For a benchmark function, Lissovoi et al. have proved that the Generalized Random Gradient Hyper-heuristics can learn to adapt the neighborhood size of Randomized Local Search to optimality during the run. They have also proved that the performance of the hyper-heuristics improves as the number of low-level local search heuristics to choose from increases [31].

Genetic programming has also been used as an offline hyper-heuristic to automatically evolve probability distributions, and hence to automatically generate mutation operators in an evolutionary programming as opposed to human designed existing operators [32].

Schlünz et al., in their work, have claimed that the first application of a multiobjective hyperheuristic, which is an evolutionary-based technique incorporating multiple sub-algorithms simultaneously, is applied to the multi-objective in-core fuel management (MICFMO) optimization problem. The hyperheuristic is able to raise the level of generality at which MICFMO may be performed, and it is capable of yielding improved quality in optimization results (compared to the preferred metaheuristics) [33].

A general-purpose selection hyper-heuristic search framework designed for the grouping has been extended to pair up various heuristic/operator selection and move acceptance methods for an NP-hard combinatorial optimization grouping problem of data clustering. The performances of various selection hyper-heuristics are compared using a set of benchmark instances which vary in terms of the number of items, groups as well as number ad nature of dimensions. The empirical results show that the proposed framework is indeed sufficiently general and reusable [34].

Cyber security in the context of big data is known to be a critical problem and presents a great challenge to the research community. Sabar et al. have proposed a novel, domain-independent hyper-heuristic framework for the formulated "Support Vector Machine" (SVM) configuration process as a bi-objective optimization problem in which accuracy and model complexity are considered as two conflicting objectives. The effectiveness of the proposed framework has been evaluated on Microsoft malware big data classification and anomaly intrusion detection [35].

The next work analyzes the ability of popular selection operators used in a hyper-heuristic framework to continuously select the most appropriate optimization method over time. Van der Stockt et al. have presented the considerations and criteria to select a diverse mix of heuristics specific to dynamic optimization problems and non-dynamic optimization problems to enable the heterogeneous meta-hyper-heuristic to effectively solve dynamic optimization problems [36].

A review article identifies the characteristics necessary for the development of frameworks for optimization using metaheuristics and, from these characteristics, identifies existing gaps, especially those related to the hybridization of metaheuristics. Silva et al., in this article, have also showed that the concepts of multi-agent systems in the design of frameworks for optimization using metaheuristics facilitates and flexibilizes the development of hybrid metaheuristics and allows simultaneous exploration of different regions of the search space [37].

A Modified Choice Function (MCF), a hyper-heuristic method, is applied such that it can regulate the selection of the neighborhood search heuristics adopted by the employed and onlooker bees automatically. The proposed MCF-ABC (Artificial Bee Colony) model is a bee algorithm with multiple neighborhood search heuristics. While the employed bees and onlooker bees perform neighborhood search to exploit the promising areas of the search space, the scout bees focus on exploration of a new region in the search space. The proposed model solves the 64 Traveling Salesman Problem instances available in TSPLIB, on average, to 0.055% from the known optimum within approximately 2.7 minutes [38].

Ahmed et al. have evaluated the performance of a set of selection hyper-heuristics on the route design problem of bus networks, with the goal of minimizing the passengers' travel time, and the operator's costs. Their analysis shows the success of the sequence-based selection method combined with great deluge acceptance method, outperforming other selection hyper-heuristics in both passenger and operator objectives [39].

A hyperheuristic framework, namely hyperSPAM, composed of three search algorithms for continuous optimization problems has been proposed. The main focus is to select the search algorithms correctly such that a simple random

coordination can lead to satisfactory results. Four coordination strategies, in the fashion of hyperheuristics, have been used to coordinate the second and the third single-solution search algorithms. One of them is a simple randomized criterion while the other three are based on a success based reward mechanism [40].

Lin has proposed an effective backtracking search based hyper-heuristic (BS-HH) approach to address the Flexible job-shop scheduling problem (FJSP) with fuzzy processing time (FJSPF). A back-tracking search algorithm is introduced as the high-level strategy to manage the low-level heuristics incorporated into the BS-HH to operate on the solution domain directly. Additionally, a novel hybrid solution decoding scheme is proposed to find an optimal solution more efficiently. The author has emphasized by saying that the FJSPF which extends FJSP by allowing processing time or due date to be fuzzy variable is more close to the real-world situation [41].

A two-layered decision-making system is proposed where the first step introduces task allocation and sequencing into the system's energy management procedures for the purpose of enabling long-term autonomy of a heterogeneous of marine robots, and the next step includes constructing a validation and evaluation system for solutions and methods which will enable objective grading during the process of training a hyper-heuristic top decision-making layer [42].

Another study evaluates Multi-Objective Agent-Based Hyper-Heuristic in real-world applications, by searching solutions for four multiobjective engineering optimization problems. For this purpose, an additional multi-objective evolutionary algorithm and new quality indicators better adapted to real-world problems are used [43].

Next, the resulting sequences of low level heuristic selections and objective function values minimized through the use of a selection hyper-heuristic are used to generate a database of heuristic selections. The sequences in the database are broken down into subsequences and the mathematical concept of a logarithmic return is used to discriminate between "effective" subsequences, which tend to decrease the objective value, and "disruptive" subsequences, which tend to increase the objective value. These subsequences are then employed in a *sequenced based hyper-heuristic* and evaluated on an unseen set of benchmark problems [44].

Motivation in another work is to generate effective dynamic scheduling policies (SPs) through off-line learning and to implement the evolved SPs online for fast application. Three types of hyper-heuristic methods are proposed for coevolution of the machine assignment rules and job sequencing rules to solve the multi-objective dynamic flexible job shop scheduling problem. The results reveal that the evolved SPs can discover more useful heuristics and behave more competitive than the man-made SPs in more complex scheduling scenarios without increasing the online solution time. It also demonstrates that the evolved SPs can obtain trade offs among different objectives and have a strong generalization performance to be reused in new unobserved scheduling scenarios, which make the evolved SPs more robust when they are employed in a stochastic and dynamic scheduling environment [45].

A case study focusing on multi-objective flexible job shop scheduling problem (MO-FJSP) in an aero-engine blade manufacturing plant has been proposed. Three multi-agent-based hyper-heuristic integrated with the prior knowledge of the shop floor are proposed to evolve SPs for the online scheduling problem. This situation poses a major challenge to the current scheduling system because dynamic changes in the shop-floor require real-time responses thanks to the widely used numerous sensors, automatic robots and enhanced systems in it. Since it is not necessary to get an optimal solution in real-time scheduling, the use of heuristics to produce a satisfactory solution for solving the production scheduling problem in an acceptable

time frame is a viable option. With the change of orders, routes and other elements in the shop-floor, the previously established rules may not be able to adapt to new scheduling scenarios. Hence, the implementation of hyper-heuristics to further enhance the heuristics made by experts is necessitated. The results show that the bottleneck agent model is more favorable than the other two agent models and succeeds to make a good trade-off between the solution quality and the generalization performance among the three agent models [46].

Oyebolu et al. have presented a discrete-event simulation of continuous bioprocesses in a scheduling environment. More specifically, characteristics specific to bioprocessing and biopharmaceutical manufacturing are addressed using a simulation optimization approach. For the optimization algorithm, the authors use an evolutionary algorithm to search for optimal production control policies. As they search the space of possible heuristics or rules as opposed to possible solutions, it constitutes a hyper-heuristic approach. Dynamic SPs are investigated to make operational decisions in a multi-product manufacturing facility and react to process failure events and uncertain demand. In particular, tuning of process run times leads to improved performance as this enables better lot-sizing decisions which may allow hedging against process failure by utilizing a shorter run time [47].

Leng et al. have investigated the optimization of a variant of the location-routing problem (LRP), namely the regional low-carbon LRP (RLCLRP), considering simultaneous pickup and delivery, hard time windows, and a heterogeneous fleet. In order to solve this problem, the authors construct a biobjective model for the RLCLRP with minium total cost consisting depot, vehicle rental, fuel consumption, carbon emission costs, and vehicle waiting time. They also further propose a novel hyper-heuristic method to tackle the biobjective model. The proposed method applies a quantum-based approach as a high-level selection strategy and the great deluge, late acceptance, and environmental selection as the acceptance criteria [48].

Finally, Lissovoi have aimed to extend the understanding of the behavior and performance of hyper-heuristcis to multimodal optimization problems. In order to evaluate their capability at escaping local optima, they consider elitist, which only accepts moves that improve the current solution, and the non-elitist, which accepts any new solution independent of its quality, selection operators that have been used in hyper-heuristics in the literature [49].

3. Heuristically generated angular metric approach (HAMA)

Three heuristic properties are important. The first one is a property of dominance as a distinctive factor. The objective here is to increase the discriminativeness, because the efficiency of the heuristic entirely depends on it. The second one is the property of consistency which is based on the triangle inequality. The sum of the two sides of the three that make up a triangle must always be greater than or equal to the third side; their difference must also be less than or equal to the third one. For example, suppose we are on the node 20 at any puzzle problem. The functional value of the sum of the twentieth node and its successor must be larger than the one arising from the successor node so that the heuristic function should not repeat in itself. The third property is admissibility. In this property, any distance determined by a heuristic function is always expected to result in smaller than the actual distance. In other words, heuristic will never yield an overestimation, thereby making it an admissible one.

In light of these three properties above, it is evident that the metrics turn into a straight line when it starts from a semi-circle. The fact that it is a semi-circle is consistent with explaining the principle brought about by triangle inequality. It is

possible to further state that a heuristic acquisition is proportional to the arc length of the circle-segment in geometrical representation of HAMA [50].

Although the triangle has different line-segments (i.e., sides) such as r_1 and r_2 in the triangle (see **Figure 1**), the heuristic acquisition can be explained by setting up a proportional relation with the arc length seen by the angle of α of an isosceles triangle which is formed by selecting the smaller of r_1 and r_2 as the equal sides of the triangle and the radius of the circle as shown in **Figure 1**. Moreover, it should be noted the appearance of the angle of α as a choice which is not dependent on the lengths of the sides given by r_1 and r_2.

The arc length of the circle-segment shown in **Figure 1** is equal to $2\pi r\alpha/360^{\circ}$ where α resides in the range of $0^{\circ} \leq \alpha \leq 180^{\circ}$.

We will focus specifically on the Manhattan, the Euclidean, and the Chebyshev as a common distance metrics/measures. These distance metrics have advantages and pitfalls depending on when and how they will be used. One may emerge as a better alternative to the other. For example, k-NN, a technique often used for supervised learning, often uses the Euclidean metric. But it would not work if our data is high dimensional or consists of geospatial information.

The Euclidean distance metric: We can easily calculate the distance from the Cartesian coordinates of the points according to the Pythagorean theorem:

$$D_E(x,y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{1}$$

Two major disadvantages of the Euclidean distance metric are the lack of scale-invariance property and the dimensionality increase of the data used. Although it is a common distance metric, the Euclidean distance metric becomes the less useful under these situations. In particular, due to the *curse of dimensionality*, high dimensional space does not behave intuitively as desired in 2- or 3-dimensional space. The more the dimension increases, the closer will be the average distance and the maximum distance between randomly placed points. Therefore, the Euclidean distance metric works well in the case of low-dimensional data where it is important to measure the magnitude of the vectors.

The Manhattan distance metric: It refers to the distance between two vectors that can only move at right angles to each other. Diagonal movement is not taken into account when calculating the distance:

$$D_M(x,y) = \sum_{i=1}^{k} |x_i - y_i| \tag{2}$$

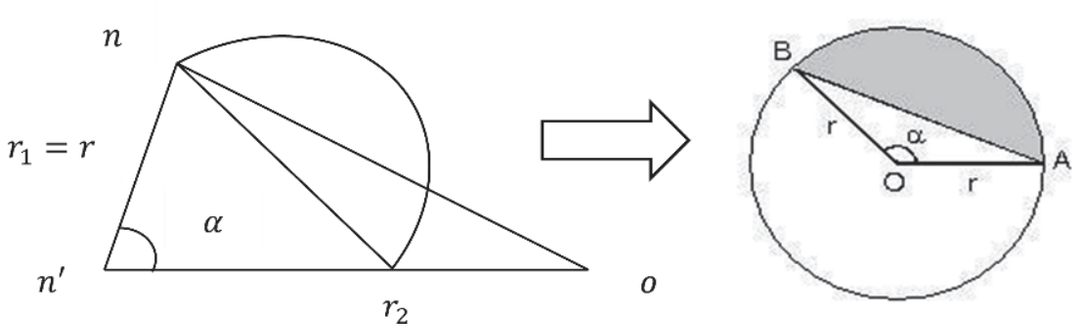

Figure 1.
Side-selection to determine the circle-segment proportional to heuristic acquisition (shaded area).

It is a less intuitive metric than the Euclidean metric, especially when used with high-dimensional data. If the dataset to be used has discrete and/or binary attributes, the Manhattan stands out as a metric that works quite well since it takes into account the realistic paths that would be taken within those attribute values. For example, while the Euclidean metric could create a straight line between two vectors, it is highly unlikely that in reality this is actually possible!

The Chebyshev distance metric: The Chebyshev distance determines the largest difference between two vectors along any coordinate dimension. With a more rigorous definition, it represents the maximum amplitude difference of the two vectors in terms of coordinates. In other words, it is simply the maximum distance through one axis:

$$D_C(x,y) = \max_i \left(\left|x_i - y_i\right|\right). \tag{3}$$

It can also be a useful metric in games that allow unrestricted 8-way movement. In warehouse logistics, it is preferred because the time an overhead crane takes to move an object is similar to the Chebshev distance. Except in such very special cases, it is unlikely to be used as an all-purpose measure of distance, such as the Euclidean or the Cosine similarity.

All the angles in the range of $0^\circ \leq \alpha \leq 180^\circ$ are within the limits of the *heuristic metric approach*, *HAMA*. In this range, an unlimited number of metrics can be achieved including the Manhattan, the Euclidean, and the Chebyshev distance metrics covered [50]. For example, if α is selected 120 degrees, then the heuristic distance metric between the two vectors under the assumption of equal segments, that is, equal lengths of adjacent nodes n' and n and n' and 0, represented as $\Delta x = \Delta y$ (that is, $|x_1 - x_2| = |y_1 - y_2|$, respectively, since our approach accepts equal sides as circle radius) can be calculated by using the Cosine theorem as follows:

$$\sqrt{(\Delta x)^2 + (\Delta y)^2 - 2\Delta x \Delta y cos 120^\circ} = \sqrt{3}\Delta x. \tag{4}$$

If α is selected 60 degrees, then the new heuristic distance metric can be calculated as,

$$\sqrt{(\Delta x)^2 + (\Delta y)^2 - 2\Delta x \Delta y cos 60^\circ} = \Delta x. \tag{5}$$

On the other hand, the Manhattan distance metric will calculate the result as follows:

$$|x_1 - x_2| + |y_1 - y_2| = 2\Delta x. \tag{6}$$

The same result can also be found with the Cosine theorem by taking the angle 180 degrees:

$$\sqrt{(\Delta x)^2 + (\Delta y)^2 - 2\Delta x \Delta y cos 180^\circ} = 2\Delta x. \tag{7}$$

Similarly, the Cosine theorem with an angle of 90 degrees and the Euclidean metric will give the same result of $\sqrt{2}\Delta x$.

Another equal result is obtained as Δx between Chebshev and the Cosine theorem with an angle of 60 degrees. This result is misleading. In order for HAMA to produce the correct results is that the lengths between each pair of nodes, that is, the lengths of the segments, have to be given differently (i.e., $r_1 \neq r_2$) from each

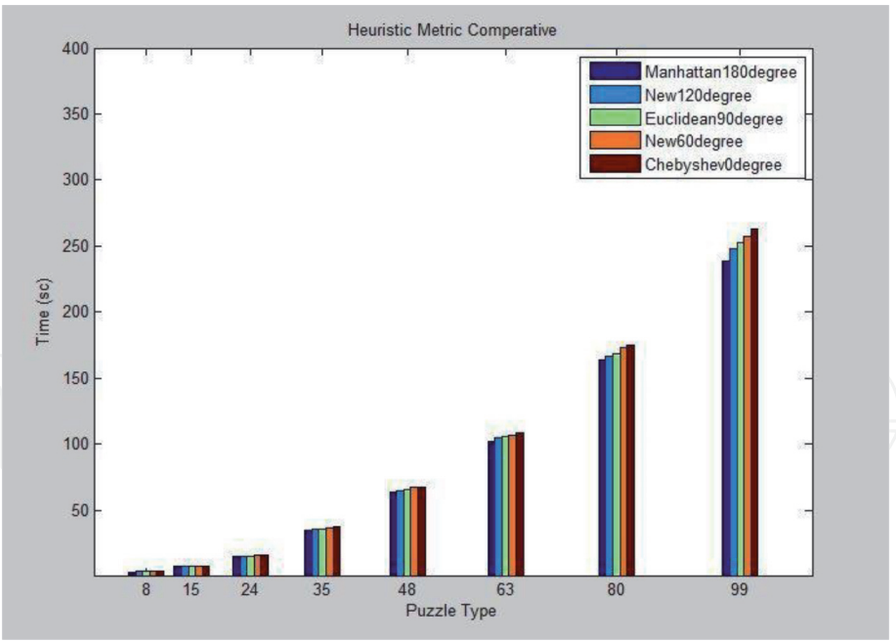

Figure 2.
A comparison of different heuristic metrics.

other in the original problem. In fact, in general, the Cosine theorem is used to solve for missing side or angle in non-right triangle and a non-matching pair/side. As we defined earlier, the Chebyshev distance metric is obtained from the maximum amplitude difference among elements of the two vectors for a given dimension. However, this amplitude-based distance is sensitive to spikes or abnormal peaks in data. As we already know, the angular distance metric belongs to the family of cosine distances derived from the Cosine similarity metric. As long as the angle between the vectors under consideration is maintained, the major advantage of the angular distance metric is its low sensitivity to any changes in vector norms, thus providing the desired distance that is not dependent on the amplitude [51].

The computer used in the simulations features an Intel 4-core i5-3230 processor with a 2.6 GHz CPU clock rate and 8 GB RAM. Run times toward a solution for each one of the metrics in 8-puzzle, 15-puzzle, 25-puzzle, 35-puzzle, 48-puzzle, 63-puzzle, 80-puzzle, and 99-puzzle problems have been compared and the comparisons have been presented in **Figure 2**.

4. An application of heuristically generated distance-metric to a chase problem

4.1 Problem statement

In Cat chasing mouse (CCM) problem, the actions and reactions of the pursuer and prey are designed to be as realistic as possible to the real-world. The prey (i.e., mouse) is smart enough how to avoid the pursuer's (i.e., cat's) maneuvers. The model allows multiple cats to chase a single mouse. Cats know how to take the appropriate angles to block off the mouse [52].

For example, the mouse's decision-making procedure works like this: the strategy is developed in such a way that the mouse's future route is to distance itself from the nearest cat where it sees the greatest threat. After determining which cat is closest, the mouse runs in the direction that the cat is running. If the cat is away from the mouse, it is not always necessary to do so away from the cat.

Cat's artificial intelligence, on the other hand, is built on fuzzy logic control (FLC), mainly to increase the cat's ability to catch the mouse [53]. The model checks if the cat is close enough to the mouse. If close, the cat uses the FLC to set its course. If it is far, it sets its course for a point in front of the mouse using the current position of the mouse and the angle it is facing. If the cat is away from the mouse but the mouse is partially moving towards the cat, then the route is again determined by the FLC. This model also allows the cats close to the mouse to move in a group, while the cats far away from the mouse block off the mouse in case of an attempt to escape. Using the random cat locations has also been a good choice for the model to better represent a real-world situation. Finally, the Max-Min rule is preferred over Kosko's Max-Product rule as the inference method in this model, claiming that it leads to a more successful FLC [52].

At this point, x_{cat} and y_{cat} represent the coordinates of cat's position whereas x_{mouse} and y_{mouse} represent the coordinates of mouse's position.

Let us define the mouse's speed traveling eastwards as a fixed variable v, and the cat's speed traveling at a pursuit direction as a fixed variable w, and it is always $w > v$. From **Figure 3**, we can write from AOB right-angled triangle,

$$\tan\left(azi(t) + ang(t)\right) = \frac{x_{mouse} - x_{cat}}{y_{mouse} - y_{cat}}. \tag{8}$$

Applying the inverse tangent (functional) operator to each side of Eq. (8) yields,

$$ang(t) = atan\left(\frac{x_{mouse} - x_{cat}}{y_{mouse} - y_{cat}}\right) - azi(t), \tag{9}$$

where $ang(t)$ is a dependent dynamic variable.

4.2 Fuzzy logic control (FLC) and design procedures for cat-mouse problem

4.2.1 Mathematical model

First of all, we will set up a mathematical model of the "plant" and determine the state variables and control input variables from this model as follows:

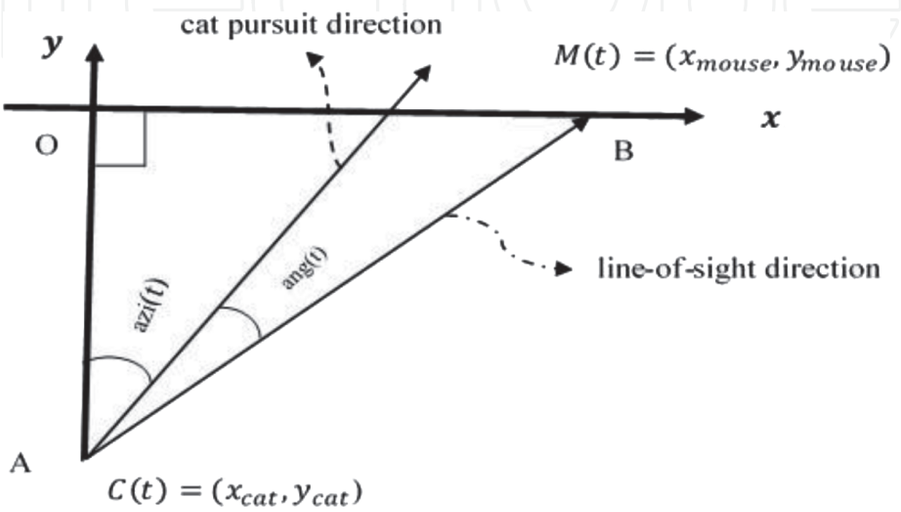

Figure 3.
Geometric representation of the problem.

$$\begin{aligned} x_{mouse}(t+1) &= x_{mouse}(t+1) + v.1; \\ y_{mouse}(t+1) &= y_{mouse}(t); \\ x_{cat}(t+1) &= x_{cat}(t) + \text{w.}\sin\left(azi(t+1)\right); \\ y_{cat}(t+1) &= y_{cat}(t) + \text{w.}\cos\left(azi(t+1)\right); \\ azi(t+1) &= azi(t) + dz(t).1; \end{aligned} \tag{10}$$

where state variables are mouse's and cat's positions, that is, $M(t) = \left(x_{mouse}(t), y_{mouse}(t)\right)$ and $C(t) = \left(x_{cat}(t), y_{cat}(t)\right)$, respectively, and azimuth angle $azi(t)$. Control input variable is $dz(t)$. We will keep these state variables as crisp variables. On the other hand, the control input $dz(t)$ and dependent dynamic variable $ang(t)$ which is derived from the state variables will be characterized as fuzzy variables that will be fuzzified to apply the fuzzy logic.

Next, we will specify the control objective. Our control objective is to minimize $ang(t+1)$ by applying appropriate control input $dz(t)$ to update $azi(t+1)$ as $ang(t)$ is given. Then, we can express our control law as part of the control effort as follows:

$$dz(t+1) = K.ang(t), \tag{11}$$

which is simply 'proportional control' P. In this control law, how to choose the most appropriate value of the proportional constant K is the main question we seek the answer to. Since this system is observed as nonlinear by nature, conventional PID design technique could be difficult to apply. Therefore, extensive simulation and trial-and-error would be required.

4.2.2 Fuzzy logic control (FLC) architecture

Fuzzy Logic Control (*a.k.a*. Fuzzy Linguistic Control) is a knowledge based control strategy that can be used when either a sufficiently accurate and yet not unreasonably complex model of the plant is unavailable, or when a (single) precise measure of performance is not meaningful or practical.

FLC design is based on empirically acquired knowledge regarding the operation of the process. This knowledge, transformed or changed into linguistic, or rule-based form, is the core of the FLC system. FLC architecture differs from a conventional controller in that the selected control algorithm and mathematical model blocks in the conventional controller will be completely replaced by Fuzzifier (Encoder), Defuzzifier (Decoder), Knowledge, and Inference Engine as shown in **Figure 4**.

In this architecture, the dynamic filter computes all the system dynamics: x (state variables) consists of selected elements of $e = r - y$, de/dt, and $\int ed\tau$. The rule base (knowledge base) provides nonlinear transformation without any built-in dynamics.

4.2.3 Fuzzification of the input variable 'ang' to fuzzy logic controller

Fuzzification is the process of decomposing a range of each fuzzy variable into one or more fuzzy sets via membership functions. Simply, it converts crisp sets to a fuzzy set. To achieve the fuzzy decomposition firstly, the range of each fuzzy variable will be specified, and then a set linguistic variables will be devised. Linguistic variables are variables whose values are words in natural language.

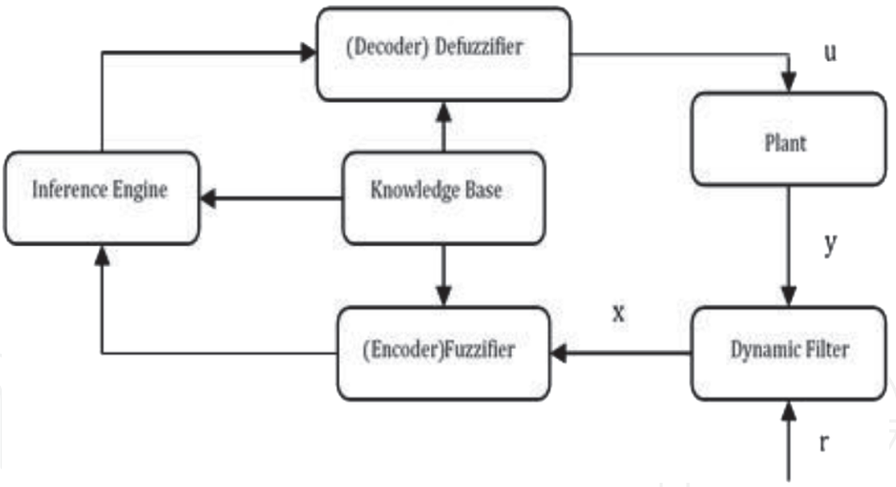

Figure 4.
FLC architecture.

Fuzzification is no different from finding an estimate of an input value. That is, it returns an *activation vector* as a fuzzy function output, representing linearly interpolated values of a given input vector '$ang(t)$'. In the meantime, briefly the *activation vector* is a row vector of the same size as the number of columns of a fuzzy set matrix M. If '$ang(t)$' falls outside the range of the support vector, an error is reported. If '$ang(t)$' falls between two elements of the support vector, a linear interpolation is performed as follows:

Let us assume that we have two known points x_1, y_1 and x_2, y_2. Our objective is to estimate y value for some x value that is between x_1 and x_2. We call this y value an "interpolated" value. There exists two simple methods for choosing y. The first one is to see whether x is closer to x_1 or to x_2. If x is closer to x_1, then we use y_1 as the estimate, otherwise we use y_2. This is called 'nearest neighbor' interpolation. The second one is to draw a straight line between x_1, y_1 and x_2, y_2. We look to see the y value on the line for our chosen x. This is called "linear interpolation."

Straight-line equation between x_1, y_1 and x_2, y_2 in **Figure 5** is given as:

$$y = y_1 + (x - x_1)\frac{y_2 - y_1}{x_2 - x_1} \tag{12}$$

where y is the estimate of the chosen value x, and represents the output of the 'fuzzify' function 'a'.

Firstly, between $ang(t)$ and 'support for $ang(t)$', $sang$, which we will define later, the following relationship exists: $ang(t) = sang(ne)$. At the output, the following relationship will be observed: $a = M(:, ne)^T$, where 'ne' is the initials of 'number of elements'.

Next, let '$ang(t)$' be our x. In this case, if we assume that $y_1 = M(:, ne)^T$, then, $y_2 = M(:, ne + 1)^T$. In response to these two, we can write the followings as a result, respectively: $x_1 = sang(ne)$ and $x_2 = sang(ne + 1)$. If we place all of these into the straight-line equation in Eq. (12), then we can write:

$$a = M(:, ne)^T + (ang(t) - sang(ne))\frac{M(:, ne + 1)^T - M(:, ne)^T}{sang(ne + 1) - sang(ne)}. \tag{13}$$

Here, if a suitable definition is made, such as $alpha \triangleq \frac{(ang(t) - sang(ne))}{sang(ne+1) - sang(ne)}$, which simplifies the above expression, then we get:

$$a = M(:, ne)^T + alpha\left(M(:, ne+1)^T - M(:, ne)^T\right). \tag{14}$$

By rearranging above, we can finally write the following:

$$a = alpha * M(:, ne+1)^T + (1 - alpha) * M(:, ne)^T. \tag{15}$$

This appears as the most suitable form of equation for coding the chosen programming language.

Fuzzification uses a discrete support. The universe-of-discourse (UoD) (i.e., the range of all possible values applicable to the chosen variable) of support are sampled at uniform (or non-uniform) intervals. For our CCM problem, both '$ang(t)$' and '$dz(t)$' will share the same set of linguistic variables: LN, SN, ZO, SP, LP. The dynamic ranges of the supports (i.e., UoD) of these two fuzzy variables however, are different: $ang(t) = -180^0$ *to* 180^0 and $dz(t) = -30^0$ *to* 30^0. These choices may be due to physical constraints and other prior knowledge. In this problem, the range difference is determined by the proportional control (law) constant K which were set to 1/6 [53]. Therefore, $dz = K.ang$, and $sdz = K.sang$. It is important that the supports of adjacent linguistic variables overlap so that more than one fuzzy rules may be fired as shown in **Figure 6**.

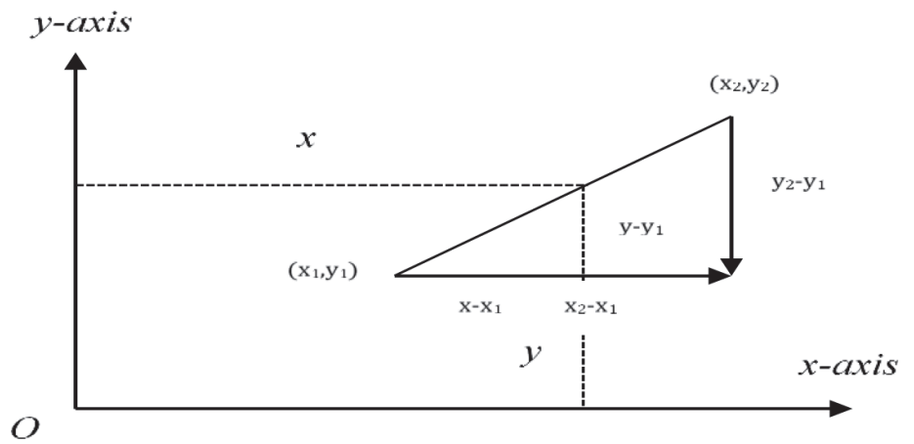

Figure 5.
Linear interpolation by drawing a straight-line between two points.

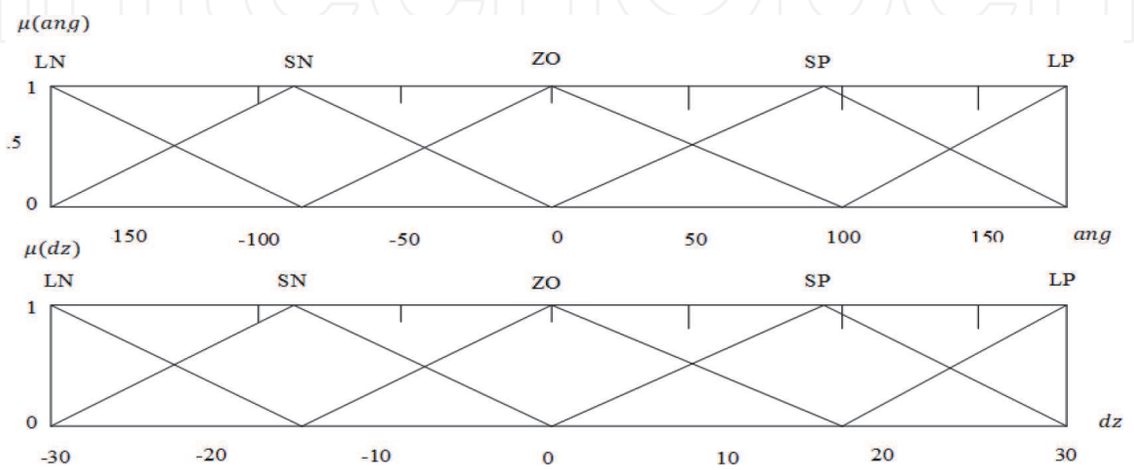

Figure 6.
Fuzzy quantization of the state variables into a set of linguistic variables. Output: fuzzy inputs. Inputs: Membership functions vs. crisp inputs.

Fuzzification is essentially a fuzzy quantization of the state variables. In this respect, the state variables '*ang*' and '*dz*' may be quantified into a set of linguistic variables, with two parameters, polarity and size, as follows: NL-Negative, Large; NS-Negative, Small; ZO-Zero; PS-Positive, Small; PL-Positive, Large. Fuzzification process converts a crisp sensor reading (value of state variable) $x = x_0$ into the grade values of each of these linguistic variables. In particular, we have,

$$[\mu_{NL}(x_0), \mu_{NS}(x_0), \mu_{ZO}(x_0), \mu_{PS}(x_0), \mu_{PL}(x_0)].$$

Fuzzy sets other than LN- and LP- consist of arrays of elements on the identical sides of an isosceles triangle, moving in the same direction at uniform intervals (i.e., quantized values of linguistic variables) and ultimately forming the row of the M fuzzy matrix. On the other hand, LN- and LP-fuzzy sets are the arrays whose elements consist of the points advancing at uniform intervals on only one of the identical sides of a triangle as follows:

$$M = \begin{bmatrix} 1 & 0.67 & 0.33 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0.33 & 0.67 & 1 & 0.67 & 0.33 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.33 & 0.67 & 1 & 0.67 & 0.33 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.33 & 0.67 & 1 & 0.67 & 0.33 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.33 & 0.67 & 1 \end{bmatrix}$$

Furthermore, from **Figure 6**, we can see that the ZO and SP fuzzy sets are the shifted versions of the SN fuzzy set, which is formed by moving from the left zero to the right zero of the triangle in **Figure 7**. After drawing attention to these important points, we can now construct the M fuzzy matrix as follows:where the first row is the LN fuzzy set, the second row is the SN fuzzy set, and similarly the third row is the ZO fuzzy set, the fourth row is the SP fuzzy set, and finally the fifth row is the LP fuzzy set.

4.2.4 Representing fuzzy logic control rules

Rules can be represented conveniently as a matrix if there are two input fuzzy variables. To represent rules, a matrix is defined where each row is a rule. Each row contains $d(1) + d(2)$ elements. Here, $d(1)$ represents the number of fuzzy sets defined on the input variable and $d(2)$ represents the number of fuzzy sets defined on the output variable. The first $d(1)$ elements specify which fuzzy set is used as the antecedent part. The next $d(2)$ elements specify which fuzzy sets in the output control variable are used. A simple rule base for the CCM problem that has only one input fuzzy variable '$ang(t)$' and one control (output) variable '$dz(t+1)$'.

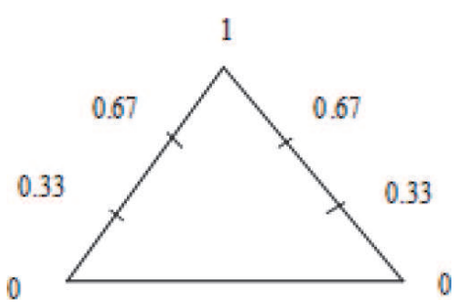

Figure 7.
Fuzzy sets arising as arrays of elements at uniform intervals.

Rule matrix is defined as follows:

$$rule = \left[\begin{array}{cccccc|cccccc} \multicolumn{6}{c|}{\overbrace{}^{}} & & & & & & \\ LN & SN & ZO & SP & LP & \&? & LN & SN & ZO & SP & LP & weight \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right].$$

In the rule matrix, there are five rules, one from each row, respectively:

$$\text{If } ang(t) \text{ is LN/SN/ZO/SP/LP, then } dz(t+1) \text{ is LN/SN/ZO/SP/LP.}$$

$\&? = 1$ means that if there is only one input fuzzy variable (i.e., this case) or the second fuzzy variable is to be ignored for that rule. The last column indicates the relative weighting of that rule.

4.2.5 Inference engine

A fuzzy inference engine is a mechanism to calculate an output from given inputs using fuzzy logic. When input variables are fuzzified, each rule in the rule base will try to determine its degree of activation using '*min-max*' or '*correlation-max*' method. For rules which have a non-zero activation value, the output fuzzy variables will be combined (fuzzy union) yielding a resultant fuzzy set.

Now, let us show how to create a rule matrix where each row corresponds to a rule, then its rows and columns can be specified by '*nrule*', and $(d(1) + d(2) + d(3) + 1)$, respectively.

Each row consists of five parts. The first part with $1xd(1)$ dimensional input variable '*input*1' and the second part with $1xd(2)$ dimensional input variable '*input*2' are allocated for antecedent variables. The third part, which is a single column, will be equal to '1' if '*input*2' is to be ignored for that rule, and '0' otherwise. The fourth part with $1xd(3)$ dimensional output variable, *output*, is fuzzy representation of consequent variable. Here, $d = [d(1)\ d(2)\ d(3)]$ is a 1×3 dimensional vector that specifies the number of fuzzy sets (adjectives or linguistic variables) defined on each UoD. In this d vector, $d(2)$ gives the number of fuzzy sets of the second input variable, if there are 6 input arguments for inference function, otherwise (i.e., the case of 5 input arguments only for the function), $d(2)$ gives the number of fuzzy sets of the output, or else $d(3)$ is the number of fuzzy sets defined on the output variable.

- First, we consider the first $d(1)$ columns of the matrix giving the first input set.

- Then, an *nrulex*1 column vector, of which all elements are '1', premultiplied by a $1xd(1)$ row vector 'a_1', is multiplied by the corresponding elements of the first $d(1)$ columns of the rule matrix to eventually produce a matrix of size $nrulexd(1)$. Here, 'a_1' represents the activation of fuzzy (antecedent) variables for input-1 and has the same dimension as $d(1)$.

- Next, we apply the '*max*' operation to the $nrulexd(1)$ matrix after it is transposed, find the maximum values in each '*nrule*' column, and place these values in a $1xnrule$ row vector.

- Finally, the *nrule* dimensional row vector is converted back to an *nrule* dimensional column vector with a transpose operation and is represented by A_1.

Case 1. Having only input-1 or $d(1)$:

- *This nrulex1 column vector A_1 is multiplied by the corresponding elements of the same size column vector called 'weight' which is positioned in the last column of the rule matrix and represents the weighting of each rule.*
- *Each element of this nrule-by-1 product is placed on the main diagonal of an nrule dimensional square matrix which is also a diagonal matrix.*
- *Later, this diagonal matrix is multiplied by the output fuzzy variable set.*

Case 2. Having the second input, input-2 or $d(2)$, as well:

- *First, we consider the next $d(2)$ columns of the matrix giving the second input set.*
- *Then, an nrule-by-1 column vector, of which all elements are '1', premultiplied by a 1x$d(2)$ row vector 'a_2', is multiplied by the corresponding elements of the next $d(2)$ columns to eventually produce a matrix of size nrulex$d(2)$. Here, 'a_2' represents the activation of fuzzy (antecedent) variables for input-2 and has the same dimension as $d(2)$.*
- *Next, we apply the 'max' operation to lastly obtained nrule-by-$d(2)$ matrix after it is transposed, and as a result, we get an nrulex1 column vector, A_2.*
- *A 'min' operator is then applied on a 2xnrule transposed augmented nrulex2 matrix of A_1 and A_2. The result is a 1xnrule row vector consisting of the minimum elements in each column of the transposed augmented matrix of $[A_1 : A_2]^T$.*
- *Afterwards, this 1xnrule row vector converted to nrulex1 column vector via a transpose operator is multiplied by the corresponding elements of the column vector 'weight' which forms the last column of the rule matrix. The resulting product is also an nrulex1 column vector.*
- *In addition, each element of this column vector is placed on the main diagonal of an nrule dimensional square matrix which is also a diagonal matrix.*
- *Besides, this diagonal matrix is multiplied by a submatrix consisting of columns that make up the output variable of the rule matrix. Let us call this nrule-by-dout matrix 'Tmp' which gives the activation of each rule. Here, 'dout' represents the number of the fuzzy sets defined on the output variable.*
- *Now, with the 'max' operator to be applied on this nrulexdout matrix 'Tmp', we will obtain a row vector whose elements consist of the maximum values in each column of the aforementioned matrix. This dout dimensional row vector called 'act' will represent the activation of each output fuzzy set.*
- *B is an output fuzzy variable matrix whose size is defined by dout-by-nofz, where dout is equal to $d(2)$ if there is a single input, and $d(3)$ in case of two inputs, and nofz is equal to the length of a row vector giving the coordinates of UoD, or it is equal to the number of output fuzzy sets which is a subset of UoD. In other words, each row*

in B is a fuzzy set defined on that output variable and each column is the element of the discrete support it corresponds to.

- *If Kosko's max product rule is applied, then the output is calculated with the following formula where '*' implies that the product rule is already applied:*

$$out = max\,(diag(act) * B). \tag{16}$$

- *If the max-min rule will be applied instead of the product rule, the calculation will then be as follows:*
  - *Firstly, the 'act' vector converted into a doutx*1 *column vector is multiplied by a* 1x*nofz row vector composed of 1 s equal to the number of the output fuzzy sets.*
  - *The objective here is to generate a sparse matrix of the same size as the B matrix to be compared via the 'min' operator.*
  - *According to the max-min rule, firstly, by applying the 'min' operator between this generated sparse matrix and the B matrix, an array of the same size as the sparse matrix and the B matrix is created, in which the elements of both matrices in the same position are compared and the smaller one is placed.*
  - *Finally, by applying the 'max' operator to the resulting matrix, a row vector containing the maximum element from each column will be returned as the output 'out'.*
  - *The equation of the max-min rule applied step by step above is as follows:*

$$out = max\left(min\left(act^T * ones(1, nofz), B\right)\right). \tag{17}$$

4.2.6 Defuzzification

Fuzziness helps us to evaluate the rules, but the final output of a fuzzy system has to be a crisp number. Defuzzification is the process of combining the successful fuzzy output sets produced by the inference mechanism. The input for the defuzzification process is the aggregate output (i.e., unified outputs of all rules) fuzzy set '*out*'.

Most popular defuzzification method is the *centroid technique*. It involves calculating the point where a vertical line would slice the aggregate set into two equal masses. It calculates the centroid of output of the fuzzy set '*out*' defined on the discrete support, '*support*'. Here '*out*' and '*support*' should have the same size. The *centroid method* equation is:

$$y = \frac{\int \mu_B(z).z\,dz}{\int \mu_B(z)dz} \tag{18}$$

Centroid defuzzification method finds a point representing the center of gravity (or area) of the fuzzy set B on the interval ab. In Eq. (18), z represents a crisp sensor reading or value of the state variable, and $\mu_B(z)$ is the fuzzy quantity that is the graded value of the particular linguistic variable upon the process of conversion (i.e., fuzzification process) from the crisp or precise quantity. Simply, we can adapt Eq. (18) for coding convenience with our values '*out*' and '*support*' as follows:

$$y = sum(out. * support)/sum(out) \tag{19}$$

5. Numerical experiments and simulations

In this work, we have devised a heuristic distance metric approach that employs the Cosine theorem, which uses the angle between the line-of-sight direction from cat to mouse and the cat's current pursuit direction only at non-right angles, rather than the distance metric which calculates the distance of current cat to mouse based on the Euclidean distance [52]. As here, if the metric that calculates the current distance from cat to mouse is not always taken in Euclidean, but built around our heuristic metric approach that allows an unlimited number of metrics to be achieved, whereby the distinction between the computation times of each metric can be clearly seen in **Figure 2** as the puzzle size increases, mouse will gain more time in perceiving the incoming danger, thus increasing the percentage of evading it, and will escape.

We have also compared our heuristic approach with the metric distance between two adjacent nodes, d_m, that do not take into account the angle between the nodes, defined as [54],

$$d_m = \frac{l_i + l_j}{2}, \{l_i, l_j\} \in R^+, \tag{20}$$

where l_i, l_j represent the metric lengths of the segments i and j, respectively. Here, the metric distance defines geodesics (i.e., the shortest paths) as those paths with minimal sum of metric length.

Next, we chose randomly generated cat locations as they better reflect the real-world in each run, and compared the results by running a total of 100 trials for each of the selected cases that accepted first 4 cats and then 8 cats and finally 12 cats against a single mouse. Thus, 'caught and escape percentages vs. number of cats' findings for three metric distances (i.e., the metric calculated only based on the EUCLIDEAN distance, the metric calculated over the GEODESICS definition, and the metric devised using the angle-dependent Cosine theorem as a result of our angular metric approach, that is HAMA) have been searched for and the results evaluated comparatively. **Table 1** below confirms that our approach, which we call HAMA, performs best in terms of evasion and escape performance that we consider:

As can be seen from **Figure 8**, the best caught and escape performance is exhibited by HAMA. That is, when compared to the other two metrics, the caught performance is minimum and the escape performance is maximum. Another noteworthy finding is that as the number of predators increases, caught and escape

Distance metrics used	Number of trials	Number of cats	Caught percentage (%)	Evasion and escape percentage (%)
HAMA	100	4	14	86
GEODESICS			18	82
EUCLIDEAN			21	79
HAMA	100	8	30	70
GEODESICS			39	61
EUCLIDEAN			41	59
HAMA	100	12	47	53
GEODESICS			52	48
EUCLIDEAN			61	39

Table 1.
Comparative metric performances based on numerical experiments.

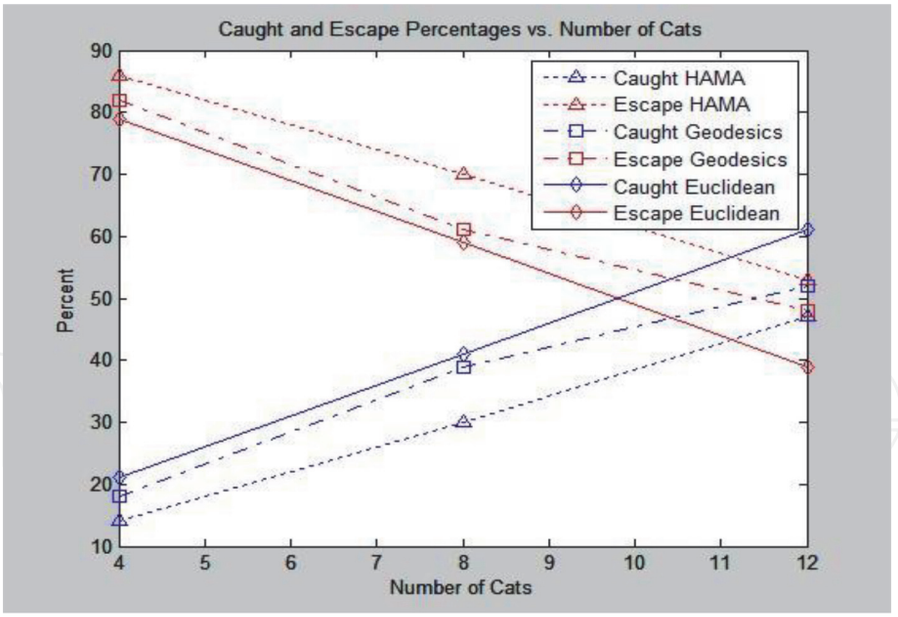

Figure 8.
The percentages of caught and escape performances of the distance metrics considered.

performances in other metrics replace. However, in this sense, compared to the increasing number of cats, HAMA still demonstrates the ability to at least maintain its initially set objective—that is, the prey can constantly escape from the pack of predators, and avert their attacks as much as possible.

The program runs the simulation until the mouse is caught, escapes the 600×600 square, or it iterates 500 times. When any of these three termination criteria is met, the solution to problem has been achieved, and thus the program will terminate its execution. Given these three termination criteria, it can never be inconsistent to define two different objective functions: they can either be formulated in such a way that, with the advantage of the heuristic metric approach, the cat travels the distance to catch the mouse in the shortest time, or the mouse easily increases the percentage of escape from the cat. The latter essentially describes the performance we would like to see and for which we have shared the results in the light of these expectations in **Table 1** above. **Figures 9** and **10** below give the

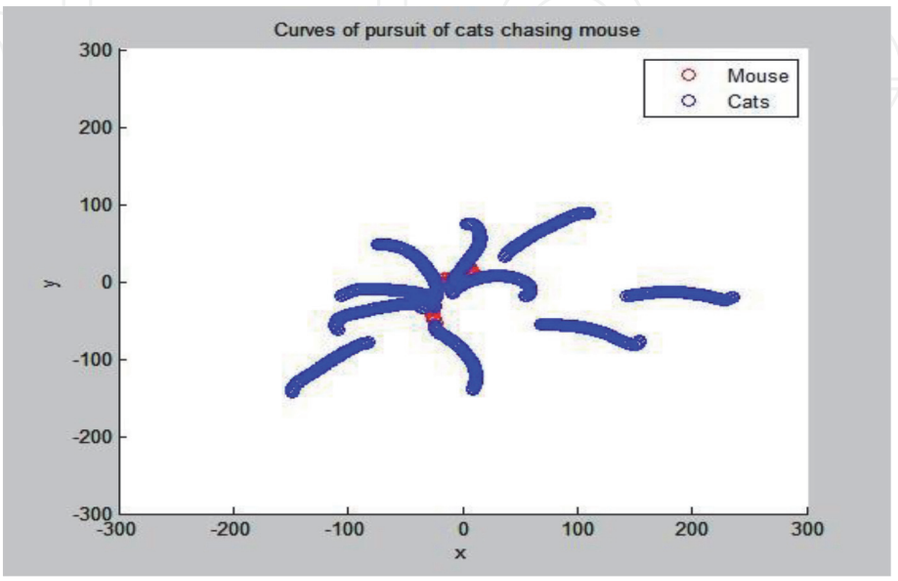

Figure 9.
Program output depicting the caught *of a single mouse for 10 randomly initialized cat states.*

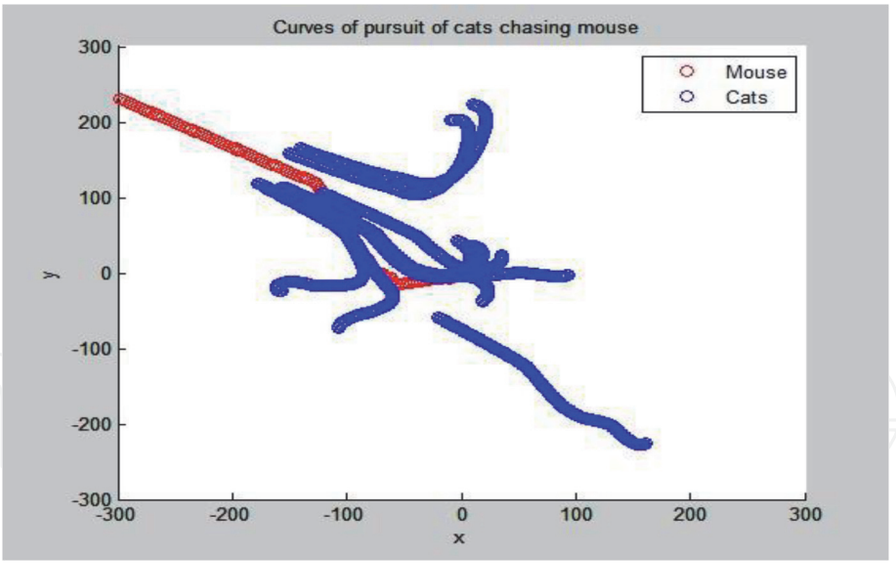

Figure 10.
Program output depicting the escape *of a single mouse for 10 randomly initialized cat states.*

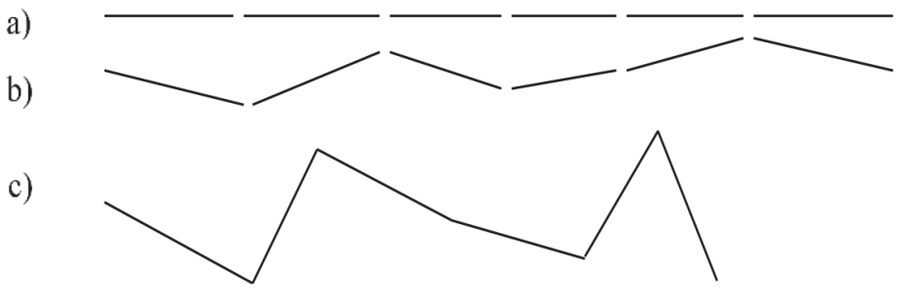

Figure 11.
Three routes of pursuit between the predator and prey, with the same metric length but with increasingly higher angular lengths [54]: a) route as simple as possible with almost zero angular variation, i.e., the shortest route, b) more complex route, but still far from random, c) the most complex route, far from any regularity, composed by angles of all amplitudes.

program output of the caught and escape problem for the case of 10 cats against a single mouse for randomly generated cat locations.

In the meantime, by emphasizing the points where we differ from different perspectives given in the literature, we find it useful to clarify the definition of angular metric that we have discussed in this work. For example, **Figure 11** shows the routes with the same metric length but with increasingly higher angular lengths [54]. The angular distance we mean in our work differs from the angular distance meant there in that: In **Figure 11(a)**, the angular distance between perfectly aligned axial segments is zero [54], whereas in our approach, while determining the heuristic metric that allows the predator to take the distance it needs to cover in order to catch its prey in the shortest time or that allows the prey to easily increase the percentage of escape from the predator, we have developed an approach that uses the angle as an effective parameter, and hence called the angular distance metric, as we explained above.

6. Conclusion

The main motivation behind our heuristic metric approach was that, consistent with the triangle inequality principle, the heuristic acqusition that started with a

semi-circle and turned into a straight-line was ultimately proportional to the length of the arc seen by the angle of α of the circle-segment. The use of HAMA became even more important as the angle emerged as the most determining factor of the problem, as we explained in the relevant section above.

In prey-predator-like pursuit-evasion problems, the angles between the direction of pursuit and the direction of the line-of-sight exhibit a broad perspective corresponding to acute, right, and wide angles, depending on how the prey and the predator are positioned relative to each other for their own purposes. In the heuristic angular distance metric approach, we therefore chose to use a formula that takes into account angles rather than vector lengths.

As a distance metric application, we believe that "cat and mouse" is a good example of what is known as chase problem and the resulting trajectories are called curves of pursuit. Cat chasing a mouse problem is a very common yet interesting problem in kinematics. The trajectory, travel time and relative approach velocity of a pursuer tracking a prey along a simple curve of pursuit are deduced using basic principles of two-dimensional kinematics. Problems of this general sort are of interest to the military community and to video game designers. Here, in the context of pursuit and evasion, a design problem that allows an artificial intelligence-like process where cat and mouse makes smart moves relative to each other and therefore makes more appropriate decisions, is discussed. The design is built around Fuzzy logic control to determine route finding between the predator and prey. As long as the angle between the vectors under consideration is maintained, the major advantage of the angular distance metric is its low sensitivity to any changes in vector norms, thus providing the desired distance that is not dependent on the amplitude.

In the numerical experiments, we chose randomly generated cat locations as they better reflect the real-world in each run, and compared the results by running a total of 100 trials for each of the selected cases that accepted first 4 cats and then 8 cats and finally 12 cats against a single mouse. Thus, 'caught and escape percentages vs. number of cats' findings for three metric distances (i.e., the metric calculated only based on the Euclidean distance, the metric calculated over the geodesics definition, and the metric devised using the angle-dependent Cosine theorem as a result of our angular metric approach) have been serached for and the results evaluated comparatively. From the comparison of these three, it is clear that our approach gives better results than the others.

As for future work, the larger the graphic size representing the route of the prey and the predator and specified by the same width and height in pixels, or the higher the number of iterations, we will end up facing high computational cost of determining the minimum routes. To prevent this situation, in the near future, we are planning to consider using a subgraph around each node containing the nodes that are reachable from the origin node within the restricted distance. It may be seen as the maximum pursuer distance from the node under calculation. In this way, we will always be able to calculate the metric geodesics or the shortest paths between each pair of nodes. For example, we would use the following set of distances (in pixels): P = {600 (current situation), 1000, 1400, 2000, 4000, 8000}.

Conflict of interest

The author declares that there is no conflict of interest regarding the publication of this paper.

Author details

İhsan Ömür Bucak
Iğdır University, Iğdır, Turkey

*Address all correspondence to: iomur.bucak@igdir.edu.tr

References

[1] Nabiyev VV. Yapay Zekâ: İnsan-Bilgisayar Etkileşimi. Ankara: Seçkin Publishing House; 2010. p. 776

[2] Feigenbaum EA, Feldman J, editors. Computers and Thought. New York: McGraw-Hill Inc.; 1963. p. 548

[3] Burke EK, Gendreau M, Hyde M, Kendall G, Ochoa G, Ozcan E, et al. Hyper-heuristics: A survey of the state of the art. Journal of the Operational Research Society. 2013;**64**(12): 1695-1724. DOI: 10.1057/jors.2013.71

[4] Fang HL, Ross P, Corne D. A promising hybrid GA/heuristic approach for open-shop scheduling problems. In: Cohn A, editor. Eleventh European Conference on Artificial Intelligence. Chichester, UK: John Wiley & Sons; 1994. pp. 590-594

[5] Hart E, Ross P, Nelson JAD. Solving a real-world problem using an evolving heuristically driven schedule builder. Evolutionary Computing. 1998; **6**(1):61-80. DOI: 10.1162/evco.1998.6.1.61

[6] Marin-Blazquez JG, Schulenburg J. A hyper-heuristic framework with XCS: Learning to create novel problem-solving algorithms constructed from simpler algorithmic ingredients. In: IWLCS, Vol. 4399 of Lecture Notes in Computer Science. Berlin, Heidelberg: Springer; 2005. pp. 193-218. DOI: 10.1007/978-3-540-71231-2_14

[7] Burke EK, Petrovic S, Qu R. Case based heuristic selection for timetabling problems. Journal of Scheduling. 2006; **9**(2):115-132. DOI: 10.1007/s10951-006-6775-y

[8] Ross P, Marin-Belazquez JG. Constructive hyper-heuristic in class timetabling. In: IEEE Congress on Evolutionary Computation; 2–4 September 2005. pp. 1493-1500

[9] Kahng AB. Traveling salesman heuristics and embedding dimension in the Hopfield model. In: Proceedings of the IEEE/INNS International Joint Conference on Neural Networks. Washington, DC; 1989. pp. 513-520. DOI:10.1109/IJCNN.1989.118627

[10] Flach P. The geometry of ROC space: Understanding machine learning metrics through ROC isometrics. In: Proceedings of the 20th International Conference on Machine Learning (ICML-2003). 2003. pp. 194-201

[11] Janssen F, Fürnkranz J. On the quest for optimal rule learning heuristics. Machine Learning. 2010;**78**(3):343-379. DOI: 10.1007/s10994-009-5162-2

[12] Glover F. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research. 1986;**13**:533-549. DOI: 10.1016/0305-0548(86)90048-1

[13] Glover F, Laguna M. Tabu Search. Boston: Kluwer; 1997. pp. 41-59. DOI: 10.1007/978-1-4419-1665-5_2

[14] Mirjalili S, Lewis A, Mostaghim S. Confidence measure: A novel metric for robust meta-heuristic optimisation algorithms. Information Sciences. 2015; **17**:114-142. DOI: 10.1016/j.ins.2015.04.010

[15] Gaschnig J. A problem similarity approach to devising heuristics: First results. In: IJCAI. 1979. pp. 301-307. DOI:10.1016/B978-0-934613-03-3.50007-6

[16] Guida G, Somalvica M. Semantics in problem representation. Information Processing Letters. 1976;**5**(5):141-145. DOI: 10.1016/0020-0190(76)90060-0

[17] Guida G, Somalvica M. A method for computing heuristics in problem solving. Information Sciences. 1979;**19**:

251-259. DOI: 10.1016/0020-0255(79)90024-0

[18] Pearl J. On the discovery and generation of certain heuristics. AI Magazine. 1983;**4**(1):23-33. DOI: 10.1609/aimag.v4i1.385

[19] Valtorta M. A result on the computational complexity of heuristic estimates for the A^* algorithm. Information Sciences. 1984;**34**:47-59. DOI: 10.1016/0020-0255(84)90009-4

[20] Passino KM, Antsaklis PJ. Metric space approach to the specification of the heuristic function for the A* algorithm. IEEE Transactions on Systems, Man, and Cybernetics. 1994; **24**(1):159-166. DOI: 10.1109/21.259697

[21] Rosenfeld A, Kaminka G, Kraus S. Adaptive robot coordination using interference metrics. In: Proceedings of the 16th Eureopean Conference on Artificial Intelligence (ECAI'2004); Valencia, Spain; 2004. pp. 910-916

[22] Bringmann K, Engels KC, Manthey B, Rao BVR. Random shortest paths: Non-Euclidian instances for metric optimization problems. Algorithmica. 2015;**73**(1):42-62. DOI: 10.1007/s00453-014-9901-9

[23] Akinyemi IO. A refinement-based heuristic method for decision making in the context of Ayo game [thesis]. Covenant University; 2012

[24] Sosa-Ascencio A, Ochoa G, Terashima-Marin H, Conant-Pablos SE. Grammar-based generation of variable-selection heuristics for constraint satisfaction problems. Genetic Programming and Evolvable Machines. 2016;**17**(2):119-144. DOI: 10.1007/s10710-015-9249-1

[25] Sosa-Ascencio A, Terashima-Marin H, Ortiz-Bayliss JC, Conant-Pablos SE. Grammar-based selection hyper-heuristics for solving irregular bin packing problems. In: Proceedings of the 2016 Genetic and Evolutionary Computation Conference Companion (GECCO'16); Denver, Colorado, USA; 2016. pp. 111-112

[26] Dokeroglu T, Cosar A. A novel multi-start hyper-heuristic algorithm on the grid for the quadratic assignment problem. Engineering Applications of Artificial Intelligence. 2016;**52**:10-25. DOI: 10.1016/j.engappai.2016.02.004

[27] Wu X, Consoli P, Minku L, Ochoa G, Yao X. An evolutionary hyper-heuristic for the software project scheduling problem. In: Handl J, Hart E, Lewis P, López-Ibáñez M, Ochoa G, Paechter B, editors. Parallel Problem Solving from Nature—PPSN XIV. Cham: Springer; 2016. pp. 37-47. DOI: 10.1007/978-3-319-45823-6_4

[28] Lozano JMC, Gimenez D, Garcia LP. Optimizing metaheuristics and hyperheuristics through multi-level parallelism on a many core system. In: IEEE International Parallel and Distributed Processing Symposium Workshops. 2016. pp. 786-795

[29] Tyasnurita R, Özcan E, John R. Learning heuristic selection using a time delay neural network for open vehicle routing. In: IEEE Congress on Evolutionary Computation (CEC). 2017. pp. 1474-1481. DOI:10.1109/CEC.2017.7969477

[30] Soria-Alcaraz JA, Ochoa G, Sotelo-Figeroa MA, Burke EK. A methodology for determining an effective subset of heuristics in selection hyper-heuristics. European Journal of Operational Research. 2017;**260**:972-983. DOI: 10.1016/j.ejor.2017.01.042

[31] Lissovoi A, Oliveto PS, Warwicker JA. Simple hyper-heuristics control the neighbourhood size of randomized local search optimally for leadingones. Evolutionary Computation. 2020;**28**(3):437-461. DOI: 10.1162/evco_a_00258

[32] Hong L, Drake JH, Woodward JR, Özcan E. A hyper-heuristic approach to automated generation of mutation operators for evolutionary programming. Applied Soft Computing. 2018;**62**:162-175. DOI: 10.1016/j.asoc.2017.10.002

[33] Schlünz EB, Bokov PM, van Vuuren JH. Multiobjective in-core nuclear fuel management optimization by means of a hyperheuristic. Swarm and Evolutionary Computation. 2018; **42**:58-76. DOI: 10.1016/j.swevo.2018.02.019

[34] Elhag A, Özcan E. Data clustering using grouping hyper-heuristics. In: Liefooghe A, López-Ibáñez M, editors. Evolutionary Computation in Combinatorial Optimization. Cham: Springer; 2018. pp. 101-115. DOI: 10.1007/978-3-319-77449-7_7

[35] Sabar NR, Yi X, Song A. A bi-objective hyper-heuristic support vector machines for big data cyber-security. IEEE Access. 2018;**6**:10421-10431. DOI: 10.1109/ACCESS.2018.2801792

[36] van der Stockt SAG, Engelbrecht AP. Analysis of selection hyper-heuristics for population-based meta-heuristics in real-valued dynamic optimization. Swarm and Evolutionary Computation. 2018;**43**:127-146. DOI: 10.1016/j.swevo.2018.03.012

[37] Silva MAL, de Souza SR, Souza MJF, de França Filho MF. Hybrid metaheuristics and multiagent systems for solving optimization problems: A review of frameworks and a comparative analysis. Applied Soft Computing. 2018;**71**:433-459. DOI: 10.1016/j.asoc.2018.06.050

[38] Choong SS, Wong LP, Lim CP. An artificial bee colony algorithm with a modified choice function for the traveling salesman problem. Swarm and Evolutionary Computation. 2019;**44**: 622-635. DOI: 10.1016/j.swevo.2018.08.004

[39] Ahmed L, Mumford C, Kheiri A. Solving urban transit route design problem using selection hyper-heuristics. European Journal of Operational Research. 2019;**274**:545-559. DOI: 10.1016/j.ejor.2018.10.022

[40] Caraffini F, Neri F, Epitropakis M. HyperSPAM: A study on hyper-heuristic coordination strategies in the continuous domain. Information Sciences. 2019;**477**:186-202. DOI: 10.1016/j.ins.2018.10.033

[41] Lin J. Backtracking search based hyper-heuristic for the flexible job-shop scheduling problem with fuzzy processing time. Engineering Applications of Artificial Intelligence. 2019;**77**:186-196. DOI: 10.1016/j.engappai.2018.10.008

[42] Babic A, Miskovic N, Vukic Z. Heuristics pool for hyper-heuristic selection during task allocation in a heterogeneous swarm of marine robots. IFAC-PapersOnLine. 2018;**51**(29): 412-417

[43] de Carvalho VR, Sichman JS. Solving real-world multi-objective engineering optimization problems with an election-based hyper-heuristic. In: International Workshop on Optimisation in Multi-agent Systems (OptMAS'18). 2018. pp. 1-15

[44] Yates WB, Keedwell EC. An analysis of heuristic subsequences for offline hyper-heuristic learning. Journal of Heuristics. 2019;**25**:399-430. DOI: 10.1007/s10732-018-09404-7

[45] Zhou Y, Yang JJ, Zheng LY. Hyper-heuristic coevolution of machine assignment and job sequencing rules for multi-objective dynamic flexible job shop scheduling. IEEE Access. 2019;**7**: 68-86. DOI: 10.1109/ACCESS.2018.2883802

[46] Zhou Y, Yang JJ, Zheng LY. Multi-agent based hyper-heuristics for multi-

objective flexible job shop scheduling: A case study in an aero-engine blade manufacturing plant. IEEE Access. 2019;**7**:21147-21176. DOI: 10.1109/ACCESS.2019.2897603

[47] Oyebolu FB, Allmendinger R, Farid SS, Banke J. Dynamic scheduling of multi-product continuous biopharmaceutical facilities: A hyper-heuristic framework. Computers and Chemical Engineering. 2019;**125**:71-88. DOI: 10.1016/J.COMPCHEMENG.2019.03.002

[48] Leng L, Zhao Y, Wang Z, Zhang J, Wang W, Zhang C. A novel hyper-heuristic for the biobjective regional low-carbon location-routing problem with multiple constraints. Sustainability. 2019; **11**(6):1596. DOI: 10.3390/su11061596

[49] Lissovoi A, Oliveto PS, Warwicker JA. On the time complexity of algorithm selection hyper-heuristics for multimodal optimization. In: Proceedings of the AAAI Conference on Artificial Intelligence. 2019. pp. 2322-2329

[50] Bucak IO, Tatlilioglu M. Different viewpoint for puzzle problems as artificial intelligence toy problems: A heuristic approach. In: Book of Abstracts of the 4th International Eurasian Conference on Mathematical Sciences and Applications (IECMSA-2015); Athens, Greece; 2015. p. 273

[51] Chanwimalueng T, Mandic DP. Cosine similarity entropy: Self-correlation-based complexity analysis of dynamical systems. Entropy. 2017;**19**: 652. DOI: 10.3390/e19120652

[52] Olson E. ECE539 Semester Project: Expanded Dog Chasing Cat Problem; 2003

[53] Hu YH. Intro. ANN & Fuzzy Systems, Lectures 33, 34 and 35: Fuzzy Logic Control I, II, and III; 2001

[54] Serra M, Hillier B. Angular and metric distance in road network analysis: A nationwide correlation study. Computers, Environment and Urban Systems. 2019;**74**:194-207. DOI: 10.1016/j.compenvurbsys.2018.11.003c