Implementing Defuzzification Operators on Quantum Annealers

Amir Pourabdollah
Dept. of Computer Science
Nottingham Trent University
Nottingham, UK
amir.pourabdollah@ntu.ac.uk

Giovanni Acampora
Dept. of Physics "Ettore Pancini"
University of Naples Federico II
Naples, Italy
giovanni.acampora@unina.it

Roberto Schiattarella
Dept. of Physics "Ettore Pancini"
University of Naples Federico II
Naples, Italy
roberto.schiattarella@unina.it

***Abstract*—Due to the built-in parallelism of quantum computing, there is an unexplored potential for some complex fuzzy logic computations to take the advantage of the future quantum computers. Recently, it has been introduced a novel representation of fuzzy sets and implementations of some basic fuzzy logic operators (union, intersection, alpha-cut and maximum) based on solving a Quadratic Unconstrained Binary Optimization (QUBO) problems, on a type of quantum computers known as quantum annealers. In this paper, this work is extended by presenting an implementation of centroid defuzzification on quantum annealer machines, based on binary quadratic model (BQM) but this time using Ising model. Having the basic operations and defuzzification implemented on quantum computers, this paper paves the way towards the implementation of a whole fuzzy inference engine on enhanced devices, such as quantum annealers.**

***Index Terms*—Quantum computing, fuzzy logic, fuzzy set.**

I. Introduction

Quantum computing aims to solve intractable computational problems by leveraging quantum mechanics principles like superposition and entanglement to manipulate information in a different and potentially more efficient way than traditional electronic computers. Since Feynman [1] introduced the idea of this different paradigm of computation, several quantum algorithms have been developed in different applications domains, from finance [2] and chemistry [3] to artificial and computational intelligence [4]–[8]. In the latter field, quantum computers can play a key role in improving current implementations of fuzzy systems.

In fact, today's applications of fuzzy systems are increasingly working with large amounts of data, and there is a strong emergence of identifying innovative computational paradigms capable of efficiently managing this type of systems. However, in order to develop an efficient quantum fuzzy system, several steps still need to be taken.

In [9] the authors paved the way by introducing a quantum representation of the fuzzy sets and of the operators that operate on them, such as fuzzy union, fuzzy intersection, alpha-cut and maximum. Such a representation is based on a formulation of fuzzy logic as Binary Quadratic Model (BQM), where the aforementioned operations were formulated as Unconstrained Binary Optimization (QUBO) problems in order to be efficiently solved on particular quantum computers, known as Quantum Annealers or Adiabatic Quantum Computers.

A new step towards the development of an efficient quantum fuzzy system is taken in this research, where the well-known centroid defuzzification mechanism is reformulated as BQM, but this time by using an Ising expression of the problem. The expressions for Ising and QUBO problems do look very similar. In fact, the Ising and QUBO expressions are isomorphic [10] and both can be addressed efficiently on quantum annealers. The major difference between the two models is that an Ising problem deals with spin (-1, 1), while QUBO uses binary (0,1).

As shown in the remaining of the paper, the spin representation is particularly suitable for modelling the centroid defuzzification mechanism as an objective function that can be minimized via quantum annealing. This approach is a metaheuristic whose goal is to find the global minimum of a given objective function over a given set of candidate solutions by exploiting the massive parallelism induced by quantum superposition and entanglement. The devices that exploit quantum annealing have been shown to outperform classical computers on several instances [11], and in our opinion, the joint use of these machines with an appropriate modelling of fuzzy logic via BQM will be crucial for the development of new inference engines.

The remaining of the paper is structured as follows: in Section II an analysis of the literature about the integration of fuzzy logic and quantum computing is carried out; in order to make the paper self-contained Section III describes the basic concepts of quantum annealing and summarises the fundamental aspects of a defuzzification operator; Section IV introduces the modelling of centorid defuzzification as BQM problem; finally, Section V shows the implementation of the centroid defuzzification as BQM problem on the D-Wave quantum annealers and reports the results of the experimentation.

II. Related Works

The similarity between fuzzy logic and quantum computing has motivated several researchers to start bridging the gap between these two theories. On the one hand, fuzzy logic has started to be used for simulating or modelling physical quantum systems [12]–[14], on the other hand, quantum computing concepts are currently investigated for a more efficient reformulation of fuzzy systems. In this latter area, both quantum-

inspired and real-quantum algorithms have been proposed for achieve different results. In [15] a quantum inspired genetic algorithm is proposed to initialize cluster centers in fuzzy c-means; in [16] a different example of quantum-inspired classical computation is used to improve the robustness of fuzzy controllers by modifying their inference performance based on quantum algorithms. In general, quantum-inspired algorithms are not specific to fuzzy systems, as it also exists for some other computational intelligence methods which are reviewed in [17].

As for purely quantum algorithms, these have been held back by the fact that only in recent years have usable quantum processors become available via cloud, but they are still severely limited by the high level of noise and small number of qubits that constitute them [18]. Therefore no advantage in using real-quantum algorithm in fuzzy systems has still been demonstrated practically. However, important theoretical results have been already achieved: for instance in [19] Rigatos and Tzafestas have proposed a procedure to speedup the inference process of a fuzzy switching control by means of a one-step quantum addition and subtraction that replace classical operations between large matrices.

In [20] it is presented a quantum approach to implementing a fuzzy system based on a lookup table. In particular, Grover's algorithm [21] has been used to perform the search for relationships between input and output variables of a system using a quantum computer. Although this paper represents the first attempt to use a well-known quantum algorithm to implement fuzzy systems, its use is very limited since the input-output relationships present in the lookup must be generated using classical computation.

Alongside these pioneering works directly attempting to model fuzzy systems via quantum algorithms, many researchers are following a more gradual approach to the development of quantum fuzzy systems, starting with a quantum implementation of fuzzy logic operators. In [22], Visintin et al. use quantum gates to implement t-norm and t-conorm operations so as to enable a quantum version of fuzzy union, intersection, and so on. Similarly, in [9], it is introduced a novel representation of fuzzy sets and operators based on (QUBO) problems, which are solvable efficiently by quantum annealers. However, none of the above researches focuses on the implementation of defuzzification process on quantum computers, that is one of the most computationally expensive process in a fuzzy inference engine.

III. Basic Concepts

A. BQM Problem and Quantum Annealers

Binary Quadratic Model (BQM) problems are traditionally used in computer science, with applications ranging from machine learning [23] to biology [24]. They are defined as optimization problems formulating as follows: if $\mathcal{Q}$ is an upper-diagonal matrix, which is an NxN upper-triangular matrix of real weights, and X is a vector of binary variables, a BQM problem consists in minimizing the function:

$$f(X) = \sum_{i=1}^{n}(q_i x_i) + \sum_{i=1}^{n-1}\sum_{j=i+1}^{n}(q_{ij} x_i x_j) \tag{1}$$

where q_i and q_{ij} are configurable (linear and quadratic) coefficients. BQM encompasses both Ising and QUBO problems, with the difference that in the former case the solutions are spin solutions, i.e. $x_i \in \{-1, 1\}$ with $i \in [1, \dots n]$, whereas in the latter case the solutions are binary solutions, i.e. $x_i \in \{0, 1\}$ with $i \in [1, \dots n]$. This kind of problems can be addressed efficiently by quantum annealers. In this model of computing the basic information units are the so-called *quantum bits (qubits)*. While a classical bit can take a binary value, a qubit in its superpositioned state can take both 0 and 1 with different "probabilities". Moreover, qubits can exploit the quantum mechanical phenomenon of entanglement, that happens when one qubit state depends on another one. In the quantum annealers model, a setting of qubits is specialised to find the optimum solution for minimising a binary objective function [25].

The quantum computer works out the optimum solution by means of minimising the total energy of the quantum system in an annealing process, that is why this model is also called quantum annealing. Briefly, formulating a problem in adiabatic model is finding q_i and q_{ij}, respectively associated to the superposition and entanglement biases, so that assignments of binary values $x_a, \dots, x_n$ minimises the objective function, thus represents the solutions to the problem. Then during an annealing phase, the qubits are collapsed to 0 or 1 states, so that the system naturally selects its minimum possible energy. This means that the binary states of the collapsed qubits collectively provide a solution for $f(X)$ minimisation. Similar to any quantum system, the solution is probabilistic, so that the solutions made by a number of runs (called sampling) are being averaged.

B. Defuzzification Process

Defuzzification is a process that maps a fuzzy set into a crisp value [26]. In detail, the process of defuzzification of a fuzzy set A defined on an universe X can be seen as the selection of a single element of X, based on the information conveyed in A. In [27] Runkler and Glesner developed a mathematically motivated set of 13 constraints characterizing rational defuzzification procedures. Such constraints can be summarized in four core properties as proposed in [28]:

1) A defuzzification operator always computes to one numeric value
2) The membership function determines the defuzzified value
3) The defuzzified value of two triangular-operated fuzzy sets is always contained within the bounds of individual defuzzified values
4) In the case of prohibitive information, the defuzzified value should fall in the permitted zone

Property 1 implies that defuzzification is an injective operator. Property 2 asserts that membership function is critical

in determining the defuzzified value and operations that don't affect membership functions such as the scaling or the translation of fuzzy sets, don't affect defuzzified values which do not get scaled or translated consequently. Formally, property 3 ensures that if $C = T(A, B)$ is a fuzzy set obtained as T-norm (or T-conorm) of two fuzzy sets A and B, then $\mathcal{D}(A) \leq \mathcal{D}(C) \leq \mathcal{D}(B)$ where with $\mathcal{D}$ it is intended the defuzzified values. Finally, property 4 refers to the specific situations in which particular fuzzy sets can be inferred from inference engines [29], [30] and also in such cases the defuzzification procedure has to be effective.

Over the years, many researches have studied the logic and the workings of defuzzification processes from different points of view: among the others, in [30] Yager and Filev analyzed defuzzification as invariant transformations between different uncertainty paradigms or in [31], Roychowdhury and Wang have attempted to understand the problem of defuzzification from the scope of optimal selection of an element from a fuzzy set. However, despite the different theoretical justifications underlying the defuzzification processes, well known approaches are used in current fuzzy inference engines. In general, all defuzzification operators can be formulated in discrete form (via $\sum$) as well as in continuous form (via $\int$). For simplicity, the following digression will be restrict to the discrete formulation.

Many of these operators focus on geometric support-based computation such as the *Mean of Maxima* (MOM) and the *Center of Gravity* (COG). The former [32] computes the output crisp value $\bar{x}$ as center of gravity of the area under the maxima of the fuzzy set as follows:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{|i|\mu_i = \max\{\mu_1, \ldots, \mu_n\}|} \tag{2}$$

where n is the number of samples along x and $\mu(x)$ is a discrete fuzzy set. The latter computes the defuzzified value according to (3):

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i \mu(x_i)}{\sum_{i=1}^{n} \mu(x_i)} \tag{3}$$

As shown in [28], the COG approach involves an optimization process that minimizes the membership graded weighted mean of the square of the distance. These methods are quite popular as they are computationally inexpensive and they are easy to implement within fuzzy hardware chips.

While COG is the most widely used approach, other defuzzification approaches have been introduced in literature, such as the defuzzifier based on fuzzy clustering proposed by Genther, Runkler and Glesner in [33] or the one based on neural network proposed in [34] by Halgamuge, Runkler and Glesner. There are also some level-based approaches such as alpha-cut defuzzification (ACD) that is shown to consider both static and dynamic aspects of a fuzzy set [35].

IV. Modelling the Centroid Defuzzification as BQM Problem

Centroid defuzzification for a discrete fuzzy set A with membership function $\mu_A(x)$ is defined as:

$$C = \frac{\sum_{i=1}^{n} x_i \mu_A(x_i)}{\sum_{i=1}^{n} \mu_A(x_i)} \tag{4}$$

where n is the number of samples along the x axis. To implement the centroid on a quantum annealer, an objective function must be formulated in BQM. In this model, given n binary variables $Y = \{y_1, \ldots, y_n\}$ an objective function $f(Y)$ is defined as:

$$f(Y) = \sum_{i=1}^{n} (q_i y_i) + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} (q_{ij} y_i y_j) \tag{5}$$

where q_i and q_{ij} are configurable (linear and quadratic) coefficients, respectively associated to the superposition and entanglement biases for the corresponding binary values y_i and y_j. Given a fixed set of coefficients q_i and q_{ij}, the aim is find an assignment of binary values y_i so that the objective function $f(Y)$ is minimised.

Our aim in this paper is to map the problem of minimising $f(Y)$ to the problem of finding the centroid of the fuzzy set A. Therefore, for a given fuzzy set A, coefficient sets q_i and q_{ij} must be found in a way that minimising $f(Y)$ yields a set of binary values y_i that can collectively locate the centroid of A. An intuitive approach to locate the centroid by a set of binary values is as follows:

For a given discrete membership function $\mu_A(x_i)$, let's assume that the centroid is located at the kth position between 0 and n. By definition, the centroid divides the set based on its centre of gravity. This means that the sum of the membership values on the left side of k must be equal (or the closest, due to the quantisation error) to the sum of the values on its right side. This is therefore equivalent to minimising an objective function defined as:

$$f_c(k) = \left(\sum_{i=1}^{k} \mu_A(x_i) - \sum_{i=k+1}^{n} \mu_A(x_i) \right)^2 \tag{6}$$

To convert the above problem to BQM, we notice that BQM has two implementations on quantum annealers, one with binary values $(0, 1)$ called QUBO (Binary unconstrained binary optimisation) and the other one with values $(1, -1)$ called Ising. The Ising implementation has a simpler match to mapping the centroid problem to the BQM problem, since one can simply associate $y_i = 1$ to the centroid's left-side and $y_i = -1$ to its right-side (or vice-versa) and can target minimising $f_1(Y)$ defined as:

$$f_1(X, Y) = \big(\sum_{i=1}^{n} y_i \mu_A(x_i) \big)^2; Y = \{1, ..., 1, -1, ..., -1\} \tag{7}$$

For a given fixed set of A, let's rename the values $\mu_A(x_i)$ as simply μ_i. In this case, the objective function for a given set is a function of binary values y_i, defined as:

$$f_1(Y) = (\sum_{i=1}^{n} y_i \mu_i)^2 \quad ; \quad Y = \{1, ..., 1, -1, ..., -1\} \tag{8}$$

The constraint of having a single switch-over point between 1 and -1 in Y is an extra limitation that makes f not readily mappable to BQM, since the function can take smaller values if Y has more than a single switch-over point. We suggest to add to f_1 another function f_2 (as a penalty function) that adds positive values to it unless the sequence of y_i's has a single switch-over point. To realise this function, we notice that sum the square of differences between each two consecutive y_i's in Y becomes 4 if and only if there is a single-switch-over point in Y. Therefore we define:

$$f_2(Y) = (y_1 - y_2)^2 + (y_2 - y_3)^2 + ... + (y_{n-1} - y_n)^2 - 4 \tag{9}$$

For more that one switch-over points, $f_2(Y)$ takes positive values. In an extreme case, any added penalty value that is more than zero should be big enough to rule out any small value of $f_1(Y)$ from being detected as minimum by the annealer. We notice that:

$$\begin{aligned} &max\big(f_1(Y)\big) = (n-2)^2 \\ &f_2(Y) \in \{0, 4, 8, ..., (n-1)^2 - 4\} \end{aligned} \tag{10}$$

To make sure the penalty value is big enough, we consider a factor k so that the value of $f_2(Y)$ in the case of one switch-over point is equal or greater than the maximum of $f_1(Y)$:

$$\begin{aligned} &kf_2(Y) \geq f_1(Y); \quad if \quad f_2(Y) \geq 4 \\ &4k \geq (n-2)^2 \quad or \quad k \geq \frac{n}{4}(n-4) + 1 \end{aligned} \tag{11}$$

Therefore, the new objective function is defined as:

$$f(Y) = f_1(Y) + kf_2(Y); \quad k \geq \frac{n}{4}(n-4) + 1 \tag{12}$$

There is still an outstanding issue, which is for when there is no switch-over point in Y, i.e., when $Y = \{1, 1, ..., 1\}$ or $Y = \{-1, -1, ..., -1\}$. For these two cases, $f_2(Y) = -4$ which gives $f(Y)$ a lower value than what it would be for any single or multiple switch-over forms of Y, thus misleading the annealer. To avoid these two cases, we notice that the outcome of a quantum annealer is a list of Y arrangements sorted by their energy levels. If one simply ignores the first two low energy levels and take the third lowest one, the two problematic cases will be avoided. This policy has no effect on the objective function formulation, but is something to be done programmatically after the annealing process.

Next, let's convert $f_1(Y)$, $f_2(Y)$ and $f(Y)$ to BQM forms, so that the coefficient values q_i and q_{ij} are determined.

$$\begin{aligned} f_1(Y) =& \Big(\sum_{i=1}^{n} y_i \mu_i\Big)^2 \\ &= y_1^2 \mu_1^2 + ... + y_n^2 \mu_n^2 \\ &+ 2y_1 y_2 \mu_1 \mu_2 + ... + 2y_{n-1} y_n \mu_{n-1} \mu_n \\ &= \sum_{i=1}^{n} \mu_i^2 + 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mu_i \mu_j y_i y_j \end{aligned} \tag{13}$$

In the above, we notice that $y_i^2 = 1$ for all i.

$$\begin{aligned} kf_2(Y) =& k\big((y_1 - y_2)^2 + ... + (y_{n-1} - y_n)^2 - 4\big) \\ &= k(y_1^2 + 2y_2^2 + ... + 2y_{n-1}^2 + y_n^2) \\ &- k(2y_1 y_2 + 2y_2 y_3 - ... + 2y_{n-1} y_n) - 4k \\ &= k(2n-2) - 2k \sum_{i=1}^{n-1} y_i y_{i+1} - 4k \\ &= 2kn - 6k - 2k \sum_{i=1}^{n-1} y_i y_{i+1} \end{aligned} \tag{14}$$

$$\begin{aligned} f(Y) =& f_1(Y) + kf_2(Y) \\ &= \sum_{i=1}^{n} \mu_i^2 + 2 \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mu_i \mu_j y_i y_j \\ &+ 2kn - 6k - 2k \sum_{i=1}^{n-1} y_i y_{i+1} \\ &= \sum_{i=1}^{n} \mu_i^2 + 2kn - 6k \qquad \text{(constant term)} \\ &+ 2\big(\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mu_i \mu_j y_i y_j - k \sum_{i=1}^{n-1} y_i y_{i+1}\big) \end{aligned} \tag{15}$$

For minimising $f(Y)$ in (15), we ignore the constant terms and factors, thus the aim is to minimise the following term:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \mu_i \mu_j y_i y_j - k \sum_{i=1}^{n-1} y_i y_{i+1} \tag{16}$$

To determine the BQM coefficient, (5) and (16) can be compared, therefore the linear and quadratic coefficients of (5) can be determined to be:

$$\begin{aligned} &q_i = 0 \\ &q_{ij} = \mu_i \mu_j - \begin{cases} k & \text{if } j = i+1 \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{17}$$

k can be any value equal or greater than $\frac{n}{4}(n-4) + 1$.

The linear and quadratic coefficients determined in (17) can be calculated for any given fuzzy set, and be used to set up a quantum annealer. Once the annealer reaches its collapsed states, the collapsed qubit values in the third lowest energy level can be picked up. The location of the single switch-over between 1 and -1 in (or vice-versa) in the collapsed bit array is equivalent to the location of the centroid.

V. Implementig BMQ-based Centroid Defuzzification on Quantum Annealers

Here a sample implementation of a centroid defuzzification operation is demonstrated. It used (15) for building the objective function, then it is minimised using quantum annealing and the resulted binary sequences are finally shown. The implementation is based on D-Wave System[1]. D-Wave currently provides access to its D-2000 series of quantum computers through D-Wave Leap-2, a cloud-based quantum programming platform. D-Wave also provides some Python libraries for programming using web-based and desktop IDE that connect to the same platform. The details of these libraries is out of the scope of this paper, and can be found in D-Wave Ocean Software Documentation[2].

In our experimentation a discrete fuzzy set A composed of 10 samples along the x axis for $x \in \{1, 2, ..., 10\}$ is considered as:

$$A = \{0.3/1, 0.3/2, 0.5/3, 0.7/4, 0.9/5, 0.9/6, 0.3/7, 0.2/8, 0.1/9, 0.1/10\} \quad (18)$$

The Python listing for this program is shown in Listing 1.

Listing 1. Python program for centroid defuzzification

```
import dimod
from dwave.system import DWaveSampler,
  Embeddingomposite

mu = [0.3, 0.3, 0.5, 0.7, 0.9, 0.9,
  0.3, 0.2, 0.1, 0.1]
n = len(mu)
k = n*(n-4)/4+1
linear = {}
quadratic = {}
for i in range(n):
  linear.update(('y'+str(i):0})
for i in range(n):
  for j in range(i+1, n):
    qij=mu[i]*mu[j]
    if (j==i+1):
      qij=qij-k
    quadratic.update({('y'+str(i),'y'+str(j)):qij})

bqm = dimod.BinaryQuadraticModel(
  linear, quadratic, 0, 'SPIN')

sampler=EmbeddingComposite(DWaveSampler())
print (sampler.sample(bqm, num reads=1000))
```

The results of executing the program of Listing 1 on D-Wave quantum computer are shown in Listing 2, which lists the 12 best solutions found by the annealer sorted by increasing energy. The energy of each solution is reported in the last column of the listing. As highlighted in Section IV theoretically, the two solutions of lower energy (rows 0 and 1) correspond to the cases in which there is no switch-over point in Y and therefore they must be ignored. The third Y reported in solution 2 contains, as expected, a single switch-over point between y_3 and y_4 therefore the COG location is found between the 3rd and the 4th positions. As the y-index starts from 0, this matches to the calculated COG from (3) being 4.81.

[1]https://docs.dwavesys.com/docs/latest/index.html

[2]https://docs.ocean.dwavesys.com/en/stable/

Listing 2. The results of running Listing 1 on D-Wave quantum computer. In yellow, it is highlighted the best correct solution found.

	y0	y1	y2	y3	y4	y5	y6	y7	y8	y9	energy
0	+1	+1	+1	+1	+1	+1	+1	+1	+1	+1	-136.1
1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-136.1
2	+1	+1	+1	+1	-1	-1	-1	-1	-1	-1	-113.1
3	-1	-1	-1	-1	+1	+1	+1	+1	+1	+1	-113.1
4	+1	+1	+1	+1	+1	-1	-1	-1	-1	-1	-112.74
5	-1	-1	-1	-1	-1	+1	+1	+1	+1	+1	-112.74
6	+1	+1	+1	-1	-1	-1	-1	-1	-1	-1	-111.14
7	-1	-1	-1	+1	+1	+1	+1	+1	+1	+1	-111.14
8	-1	-1	-1	-1	-1	-1	+1	+1	+1	+1	-109.14
9	+1	+1	+1	+1	+1	+1	-1	-1	-1	-1	-109.14
10	+1	+1	-1	-1	-1	-1	-1	-1	-1	-1	-108.54
11	-1	-1	+1	+1	+1	+1	+1	+1	+1	+1	-108.54

Moreover, it is important to highlight two aspects of the achieved results: firstly, for each energy level reached by the annealer there are two solutions, representing the same location of the centroid; secondly, all the acceptable solutions reported on top of Listing 2 contain no more than a single switch-over point, proving that the formulation of the penalty function proposed in Section IV is correct.

VI. Conclusion

Defuzzification is the final step of a fuzzy inference engine, through which the fuzzy output set of the system is mapped into the numeric output value. In this paper, we reformulated the centroid defuzzification mechanism as an Ising problem, a particular BQM problem that can be solved efficiently by means of a quantum annealer. Such a device is a particular type of quantum computer that exploits the annealing phenomenon to find, in an efficient way, optimal solutions of problems formulated as BQM. This work takes another step towards modelling fuzzy inferential systems on quantum computers.

Considering that today's applications of fuzzy systems are increasingly working with large amounts of data or large sets of rules, there is a strong emergence of identifying innovative computational paradigms capable of efficiently managing this type of systems.

It is important to highlight that the main goal of this paper is not to demonstrate an advantage in using quantum computers to perform centroid computation, which even classically is particularly efficient. Rather than that, the goal achieved by this research is the proof of the quantum annealers feasibility in performing defuzzification operations.

In the future, this approach will be used to implement other defuzzification methods such as those presented in [36]. Furthermore, a comparison of the proposed approach will be carried out with a formulation of the centroid calculation as a BQM problem solved by using the quantum circuit computation model. In particular, with such a model of quantum computing some quantum algorithms present in literature for BQM problems optimization can be exploited, such as those proposed in [37], [38], which respectively leverage approaches based on the Grover's algorithm [21] and quantum genetic algorithms.

Finally, the proposed approach will be used combined with that proposed in [9], to develop a quantum framework capable of firing fuzzy rules in efficient way by exploiting quantum phenomena as superposition and entanglement, in a context where the quantum implementation of fuzzy inference engines will represent an added value both from the theoretical and application point of view.

REFERENCES

[1] R. P. Feynman, "Simulating physics with computers," *International Journal of Theoretical Physics*, vol. 21, no. 6, pp. 467–488, 1982. [Online]. Available: https://doi.org/10.1007/BF02650179

[2] R. Orus, S. Mugel, and E. Lizaso, "Quantum computing for finance: Overview and prospects," *Reviews in Physics*, vol. 4, p. 100028, 2019.

[3] B. P. Lanyon, J. D. Whitfield, G. G. Gillett, M. E. Goggin, M. P. Almeida, I. Kassal, J. D. Biamonte, M. Mohseni, B. J. Powell, M. Barbieri *et al.*, "Towards quantum chemistry on a quantum computer," *Nature chemistry*, vol. 2, no. 2, pp. 106–111, 2010.

[4] G. Acampora, "Quantum machine intelligence," *Quantum Machine Intelligence*, vol. 1, no. 1, pp. 1–3, 2019. [Online]. Available: https://doi.org/10.1007/s42484-019-00006-5

[5] G. Acampora, R. Schiattarella, and A. Vitiello, "Quantum genetic selection: using a quantum computer to select individuals in genetic algorithms," in *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, 2021, pp. 219–220.

[6] L. Alchieri, D. Badalotti, P. Bonardi, and S. Bianco, "An introduction to quantum machine learning: from quantum logic to quantum deep learning," *Quantum Machine Intelligence*, vol. 3, no. 2, p. 28, 2021. [Online]. Available: https://doi.org/10.1007/s42484-021-00056-8

[7] "Quantum computing methods for supervised learning," *Quantum Machine Intelligence*, vol. 3, no. 2, p. 23, 2021. [Online]. Available: https://doi.org/10.1007/s42484-021-00050-0

[8] M. Henderson, S. Shakya, S. Pradhan, and T. Cook, "Quanvolutional neural networks: powering image recognition with quantum circuits," *Quantum Machine Intelligence*, vol. 2, no. 1, p. 2, 2020. [Online]. Available: https://doi.org/10.1007/s42484-020-00012-y

[9] A. Pourabdollah, G. Acampora, and R. Schiattarella, "Fuzzy logic on quantum annealers," *IEEE Transactions on Fuzzy Systems*, 2021.

[10] A. Lucas, "Ising formulations of many np problems," *Frontiers in Physics*, vol. 2, p. 5, 2014.

[11] H. Ushijima-Mwesigwa, C. F. Negre, and S. M. Mniszewski, "Graph partitioning using quantum annealing on the d-wave system," in *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*, 2017, pp. 22–29.

[12] R. Leporini, C. Bertini, and F. C. Fabiani, "Fuzzy representation of finite-valued quantum gates," *Soft Computing*, vol. 24, no. 14, pp. 10 305–10 313, 2020.

[13] D. Aerts, T. Durt, and B. Van Bogaert, "A physical example of quantum fuzzy sets and the classical limit," *Tatra Mt. Math. Publ*, vol. 1, pp. 5–15, 1993.

[14] G. Acampora, F. Di Martino, R. Schiattarella, and A. Vitiello, "Measuring distance between quantum states by fuzzy similarity operators," in *2021 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2021, pp. 1–6.

[15] F. Di Martino and S. Sessa, "A novel quantum inspired genetic algorithm to initialize cluster centers in fuzzy c-means," *Expert Systems with Applications*, vol. 191, p. 116340, 2022.

[16] L. Litvintseva, I. Ul'yanov, S. Ul'yanov, and S. Ul'yanov, "Quantum fuzzy inference for knowledge base design in robust intelligent controllers," *Journal of Computer and Systems Sciences International*, vol. 46, no. 6, pp. 908–961, 2007.

[17] A. Manju and M. J. Nigam, "Applications of quantum inspired computational intelligence: a survey," *Artificial Intelligence Review*, vol. 42, no. 1, pp. 79–156, 2014.

[18] J. Preskill, "Quantum computing in the nisq era and beyond," *Quantum*, vol. 2, p. 79, 2018.

[19] G. G. Rigatos and S. G. Tzafestas, "Parallelization of a fuzzy control algorithm using quantum computation," *IEEE Transactions on Fuzzy Systems*, vol. 10, no. 4, pp. 451–460, 2002.

[20] G. Acampora, F. Luongo, and A. Vitiello, "Quantum implementation of fuzzy systems through grover's algorithm," in *2018 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2018, pp. 1–8.

[21] L. K. Grover, "A fast quantum mechanical algorithm for database search," 1996.

[22] L. Visintin, A. Maron, R. Reiser, and V. Kreinovich, "Aggregation operations from quantum computing," in *2013 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2013, pp. 1–8.

[23] P. Date, D. Arthur, and L. Pusey-Nazzaro, "Qubo formulations for training machine learning models," *Scientific Reports*, vol. 11, no. 1, pp. 1–10, 2021.

[24] D. M. Fox, K. M. Branson, and R. C. Walker, "mrna codon optimization with quantum computers," *PloS one*, vol. 16, no. 10, p. e0259101, 2021.

[25] T. Albash and D. A. Lidar, "Adiabatic quantum computation," *Reviews of Modern Physics*, vol. 90, no. 1, p. 015002, 2018.

[26] W. Van Leekwijck and E. E. Kerre, "Defuzzification: criteria and classification," *Fuzzy sets and systems*, vol. 108, no. 2, pp. 159–178, 1999.

[27] T. Runkler and M. Glesner, "A set of axioms for defuzzification strategies towards a theory of rational defuzzification operators," in *[Proceedings 1993] Second IEEE International Conference on Fuzzy Systems*. IEEE, 1993, pp. 1161–1166.

[28] S. Roychowdhury and W. Pedrycz, "A survey of defuzzification strategies," *International Journal of intelligent systems*, vol. 16, no. 6, pp. 679–695, 2001.

[29] N. Pfluger, J. Yen, and R. Langari, "A defuzzification strategy for a fuzzy logic controller employing prohibitive information in command formulation," in *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*. IEEE, 1992, pp. 717–723.

[30] R. R. Yager and D. P. Filev, "Essentials of fuzzy modeling and control, john and wiley and sons," *Inc., NJ*, 1994.

[31] S. Roychowdhury and B.-H. Wang, "Cooperative neighbors in defuzzification," *Fuzzy Sets and Systems*, vol. 78, no. 1, pp. 37–49, 1996.

[32] D. P. Filev and R. R. Yager, "A generalized defuzzification method via bad distributions," *International Journal of Intelligent Systems*, vol. 6, no. 7, pp. 687–697, 1991.

[33] H. Genther, T. A. Runkler, and M. Glesner, "Defuzzification based on fuzzy clustering," in *Proceedings of 1994 IEEE 3rd International Fuzzy Systems Conference*. IEEE, 1994, pp. 1645a–1648.

[34] S. K. Halgamuge, T. A. Runkler, and M. Glesner, "On the neural defuzzification methods," in *Proceedings of IEEE 5th International Fuzzy Systems*, vol. 1. IEEE, 1996, pp. 463–469.

[35] A. Pourabdollah, J. M. Mendel, and R. I. John, "Alpha-cut representation used for defuzzification in rule-based systems," *Fuzzy Sets and Systems*, vol. 399, pp. 110–132, 2020.

[36] T. Runkler, "Selection of appropriate defuzzification methods using application specific properties," *IEEE Transactions on Fuzzy Systems*, vol. 5, no. 1, pp. 72–79, 1997.

[37] A. Gilliam, S. Woerner, and C. Gonciulea, "Grover adaptive search for constrained polynomial binary optimization," *Quantum*, vol. 5, p. 428, 2021.

[38] A. Malossini, E. Blanzieri, and T. Calarco, "Quantum genetic optimization," *IEEE Transactions on Evolutionary Computation*, vol. 12, no. 2, pp. 231–241, 2008.