



Blockstake.ai

Latest Updated!

## Optimization of Cryptocurrency Algorithmic Trading Strategies Using the Decomposition Approach, with virtual direction and Mathematical analysis of all the bots in the BOT TRADE package.

**Abstract:** A cryptocurrency is a non-centralized form of money that facilitates financial transactions using cryptographic processes. It can be thought of as a virtual currency or a payment mechanism for sending and receiving money online. Cryptocurrencies have gained wide market acceptance and rapid development during the past few years. Due to the volatile nature of the crypto-market, cryptocurrency trading involves a high level of risk. In this paper, a new normalized decomposition-based, multi-objective particle swarm optimization (N-MOPSO/D) algorithm is presented for cryptocurrency algorithmic trading. The aim of this algorithm is to help traders find the best Litecoin trading strategies that improve their outcomes. The proposed algorithm is used to manage the trade-offs among three objectives: the return on investment, the Sortino ratio, and the number of trades. A hybrid weight assignment mechanism has also been proposed. It was compared against the trading rules with their standard parameters, MOPSO/D, using normalized weighted Tchebycheff scalarization, and MOEA/D. The proposed algorithm could outperform the counterpart algorithms for benchmark and real-world problems. Results showed that the proposed algorithm is very promising and stable under different market conditions. It could maintain the best returns and risk during both training and testing with a moderate number of trades.

---

Copyright © 2024. Blockchain.com Inc. NMLS ID# 2024031 All rights reserved..

portfolio or the optimal collection of assets from all those under consideration [11]. Portfolio optimization has been abundantly presented in the literature [11–15]. Trading strategy optimization means the optimization of the parameters of the trading rules used to examine historical trends in order to infer future behavior. With the help of one or more trading rules, the traders can determine the best entry and exit points, i.e., long and short positions. Karahan et al. [16] studied the incorporation of reinforcement learning as well as a collective decision optimization algorithm for Bitcoin (BTC) trading. The aim of their study is to maximize profits. Leung et al. [17] proposed an intelligent system for generating cryptocurrency trading signals that are developed using the sentiments exhibited in market tweets.

Despite its importance and widespread use, trading rule optimization is not frequently found in the literature. One of the main objectives of this research is to test the ability of optimized algorithmic trading strategies to hold on during different market conditions.

Cryptocurrency trading is a Multi-Objective Optimization (MOO) problem where there is more than one objective that needs to be optimized. Amongst these objectives are the return, the trading risks, the number of trades, the ratio between positive negative trades, the Maximum Draw-Down (MDD), which quantifies the maximum drop in an investment or asset's value within a certain period, etc. [18].

The MOO problem can be described as shown in [19,20]:

$$\text{Maximize } F(x) = (f_1(x), f_2(x), \dots, f_N(x)) \text{ subject to } x \in \mathcal{S} \quad (1)$$

where  $N$  is the number of objectives,  $\mathcal{S}$  is the variable space or decision space, and  $F$ :

$\mathcal{S} \rightarrow \mathbb{R}^N$  is the objective space. The achievable objective set can be characterized as  $\{F(x) \mid x \in \mathcal{S}\}$ . Due to the conflicts among the objectives, the solution for such problems is not a single point, but rather a set of non-dominated points, i.e., the Pareto Set (PS) [21,22].

A solution is considered a non-dominated one when the advancement of one objective leads to the weakening of at least one of the other objectives. For any solution  $A$  to be a Pareto Optimal (PO), it should not be dominated by another solution. It can be said that solution  $A$  dominates another solution  $B$  in the case that  $A$  is not worse than  $B$  for any of the objectives under study, and  $A$  is better than  $B$  for at least one objective [21]. The Pareto Front (PF) is the set that is constructed by mapping the non-dominated points into the objective space, whereas the PS contains the PO points in the decision space.

The MOO algorithms can be categorized as algorithms based on Pareto-dominance and algorithms based on decomposition [20].

Multi-Objectives Evolutionary Algorithms using Decomposition (MOEA/D) is a promising algorithm used to resolve both multi- and many-objective optimization problems (i.e., MOO problems with  $N \geq 3$ ). Zhang and Li [21] first introduced MOEA/D to overcome the problems with dominance-based MOEAs, where they sometimes fail to handle many objective problems without a performance reduction. MOEA/D has demonstrated its simplicity and superiority in solving complex problems. It separates the complex MOO problem into a set of scalar Sub-Problems (SPs) with the help of a Scalarization Function (SF), also called an aggregation function. A set of well-generated weight vectors, as well as a finely selected SF, are the main factors that influence the performance of such an algorithm [21,22].

Due to the good performance shown by MOEA/D, different researchers have investigated some improvements to the original algorithm. The research studies for MOEA/D can be classified into four different categories: weight-generation strategies, the adaptation of one or more SFs, the implementation of different variants of the original algorithm to solve the challenges of more complex problems, and the application of MOEA/D algorithms to real-world problems.

Some new Pareto adaptively weighted generation aspects have been presented, such as  $\mu\lambda$ -MOEA/D [23], AWD-MOEA/D [24], MOEA/D-AWG [25], and MOEA/D-URAW [26].

New decomposition mechanisms were also found in the literature, either by using new SFs, as shown in [27–29], or by using a collection of different SFs [30,31].

The decomposition method has been expanded to a larger number of EAs, such as MOEA/DD-CMA [32] and MOEA/D-ACO [33]. The employment of a variety of novel operators to manage the diversity-convergence tradeoff has also been found in the literature, such as Differential-Evolution (DE) [34], a Two-Phase with Niching mechanism MOEA/ D-TPN [35], and Hierarchical Decomposition (HD) [36].

MOEA/D was also applied to different application areas, such as network routing [37], portfolio optimization [38,39], image segmentation [40], and aerospace applications [41].

This study aims to evaluate the ability of the decomposition-based algorithms to enhance the performance of the cryptocurrency trading rules to help traders and investors improve their outcomes and compare them with the original or standard trading rules over a hardly predictable market condition, i.e., the COVID-19 pandemic.

In this paper, a decomposition-based Particle-Swarm-Optimization (MOPSO/D) algorithm, using a new linearly normalized Augmented-Weighted-Tchebycheff (AWTCH) scalarization method, has been proposed and applied to cryptocurrency trading strategy optimization. The AWTCH [42] is an updated version of the WTCH proposed by Zhang [21]. Normalization here is employed because the objective functions have extensively different ranges. The algorithm is used to optimize the controlling parameters of a set of four trading strategies named (Linear Weighted-Moving-Average, Bollinger-Bands, Stochastic Relative-Strength-Index, and Smoothed Rate-of-Change). In addition, a new hybrid weight generation strategy combining both systematic and random weight generation has also been proposed. The hybrid weight strategy is proposed to resolve the shortcomings of the systematic weight distribution proposed in [21] (as it fails to handle problems with complex PFs [20]) in a simple yet efficient methodology. The algorithm is trained and tested for the daily closing prices of Litecoin over two different periods. Litecoin was selected for two reasons. The first is that it can validate more transactions per unit time as compared to other cryptocurrencies. The second reason is that Litecoin has a low unit price, which makes it the best choice for novice traders.

The basic contributions of this research are:

- This research is the first to address using decomposition-based optimization strategies for cryptocurrency algorithmic trading.
- The proposed algorithm aims to find the best parameter set for each of the trading strategies in order to enhance the accuracy of the generated trading signals and avoid false signals.
- A new MOPSO/D has been proposed for the application at hand with two basic contributions, i.e., a newly proposed normalized aggregation mechanism and a new hybrid weight distribution mechanism.
- Furthermore, the algorithm is verified on some benchmark problems and compared against some state-of-the-art algorithms, such as the original MOEA/D and Strength Pareto Evolutionary Algorithm (SPEA2).

The following sections of this paper are organized as follows: Cryptocurrency algorithmic trading principles are shown in Section 2. The original MOEA/D algorithm is clarified in Section 3. The details of the proposed algorithm are presented in Section 4. The proposed hybrid weight generation strategy is presented in Section 5. In Section 6, the empirical results are shown, whereas the conclusions are viewed in Section 7.

## 2. Cryptocurrency Algorithmic Trading

Cryptocurrency investment involves a higher level of risk than other markets [43]. This risk arises from its volatile nature, where there is a wide range of fluctuations within limited time intervals [43]. The algorithmic trading of cryptocurrencies plays a crucial role in the world of digital currencies, providing numerous benefits and contributing to the market's overall development and efficiency.

The algorithmic trading strategy used in this paper is accomplished with the help of four recommended technical indicators (TIs): Linear Weighted-Moving-Average (L-WMA), Bollinger-Bands (BB), Stochastic Relative-Strength-Index (St-RSI), and Smoothed Rate-of-Change (S-RoC). These indicators are mathematically based calculations that are used to generate trading signals (i.e., buy and sell signals) with the help of a set of controlling parameters [44].

These indicators are important for traders for many reasons:

- They are frequently used by traders through trading platforms due to their simplicity and understandability [44,45].
- They can help traders identify not only the upcoming trends but they can also reflect the strength of these trends [44,46].
- They can be used either individually or with other trading strategies as confirmation tools for trading decisions. For example, most of the portfolio optimization models make their trading decisions based on one or more selected TIs [11,47,48].
- They are applicable for different types of markets, i.e., trending and sideways markets, such that the indicators are classified into overlay indicators and window oscillators suitable for trading and trending markets in sequence. The overlay indicators are lagging indicators (they are called lagging because the signals are generated based on price changes, such as Moving-Average (MA) indicators). The window oscillators are leading indicators (leading because the signals are generated prior to price changes, such as the RSI indicator) [44,46].

Each indicator has a set of standard parameters that were originally developed and recommended by its creator [49]. With different market conditions, there is no guarantee that these standard parameters could provide good profits or an acceptable level of risk [40]. Our aim here is to find the optimal indicators' controlling parameters that simultaneously optimize the set of selected objective functions.

The L-WMA [41] is a Moving-Average (MA) indicator. MA indicators are used to find an updated price average, which helps to smooth and strip out the market's sharp fluctuations. The L-WMA uses a previously decided weight value, which gradually decreases from the latest data point to the oldest one [50,51]. The greater the weighting factor, the more recent the pricing or data. The calculation for a L-WMA for an  $n$ -days timespan is as follows:

$$L_{WMA} = \frac{(Price_1 \times n) + (Price_2 \times n - 1) + \dots + (Price_{n-1} \times 2) + (Price_n)}{\frac{n \times (n+1)}{2}} \quad (2)$$

Trading based on MAs is performed according to the crosses against the original price chart or against a longer-term MA. Crossing above the longer MA implies a long position (buy), while crossing below it implies a short position (sell) [51,52]. The standard  $n$ -values for double L-WMA are 20 and 50 days for short and long MAs, respectively.

Figure 1 shows the effect of the chosen parameters for the L-WMA indicator on the final trading decisions. The two line charts represent the two L-WMAs, i.e., 20 and 50 days, while the bar chart is the daily price chart, where the color of the bar represents the rise or fall of the current price as compared to the previous one, i.e., red in case of a fall and green in case of a raise. As shown, there are many areas of false signals (shown in ovals). In these areas, the prices of buying are higher than those of selling, i.e., loss trade. This is due to the inability of the indicator's parameters to reflect market changes. The smaller the number of days for the MA, the more sensitive the indicator is to market changes, but with more chances for false or wrong signals, i.e., high risk [44,53]. On the other hand, a larger number of days reduces the number of false signals at the expense of the sensitivity of the indicator, which in turn could generate many delayed signals and miss a large amount of profit [44]. This fact is common for all TIs. That is why the selection of the indicators' parameters is a crucial decision in financial trading.



**Figure 1.** The crossovers between 20- and 50-days L-WMA indicator for LTC/USD trading. (The red line is for the short MA and the blue line is for the long MA, whereas the ovals show the areas of false signals) (using [Tradingview.com](https://tradingview.com), accessed on 8 November 2023).

The BB [44] is a trading mechanism that employs two volatility bands: lower and upper bands, such that these bands are calculated as positive and negative Standard Deviations (STD) from a middle band. The middle band is typically calculated as the price Simple MA (SMA) (here, the Exponentially-MA (EXMA) is used instead in order to improve the accuracy). Trading using the BB is performed through the crossovers between the prices with the two volatility bands, i.e., the upper and lower bands [52,54].

The volatility of the bands is directly proportional to the STD, such that the bands expand with increasing volatility and contract otherwise [54]. The parameters that affect the BB indicator's performance are the middle-band timespan  $n$ , the STD lookback timespan  $s$ , and the STD multiplier  $mul$ . The standard parameter values for the BB indicator are (20, 20, 2) for  $n$ ,  $s$ , and  $mul$  consequently. The three bands are calculated as shown below, where the  $EXMA(n)$  is calculated the same as WMA with a higher weight on the latest data point, i.e., the current day, and equal weights on the rest of the days [44,53].

$$BB_{middle} = EXMA(n) \tag{3}$$

$$BB_{upper} = Middle_{Band} + STD(s) \times mul \quad (4)$$

$$BB_{lower} = Middle_{Band} - STD(s) \times mul \quad (5)$$

St-RSI [44] is a combined indicator that applies the Stochastic-Oscillator indicator to the RSI value, where RSI is another indicator that compares the change in the current price values to the prices of the recent past. The output is a value in the range between 0 and 100 that is used to determine the over-sold and over-bought layers. Upward and downward crossovers with these layers provide the buy and sell signals, consequently [52]. The value of St-RSI [44] is calculated as follows:

$$St_{RSI} = \frac{RSI_{current} - RSI_{\min(t)}}{RSI_{\max(t)} - RSI_{\min(t)}} \quad (6)$$

$$RSI_{current} = 100 - \frac{100}{\frac{Avg_{gain}(n)}{1 + Avg_{loss}(n)}}} \quad (7)$$

Such that  $n$  is the RSI timespan,  $t$  is the St-RSI timespan, and  $Avg_{gain}(n)$  and  $Avg_{loss}(n)$  are the average gains and losses during the past  $n$ -days.  $RSI_{current}$  is the current RSI value whereas  $RSI_{\max(t)}$  and  $RSI_{\min(t)}$  are the maximum and the minimum values of the RSI during the past  $t$  days. The standard St-RSI parameter values are (14, 14, 30, 70), such that the first two values are the values of both  $n$  and  $t$ , whereas the last two values are the values of the over-sold and over-bought levels in sequence.

The S-RoC [44] is also a combined indicator that applies the RoC to the EXMA value instead of the price data value. The output is a value in the range  $\pm 100$ . The power of this indicator is that it measures the trends' strength by comparing the current EXMA value to the value of the EXMA calculated for the previous  $n$  days. The S-RoC is calculated as shown:

$$S_{RoC} = \frac{EXMA_{current} - EXMA_{n-days\ ago}}{EXMA_{n-days\ ago}} \times 100 \quad (8)$$

The trading signals are generated through crossovers above and below a center line (typically a zero line) [52]. In this research, crossovers with over-bought and over-sold levels are used instead to generate trading signals. In this case, two additional parameters need to be optimized i.e., overbought and oversold levels. The standard parameter values for S-RoC are 14 days for the RoC calculation and 20 days for the EXMA.

These indicators have been selected in order to cover the two types of indicators, such that the first two (WMA and BBs) are classified as overlay indicators, whereas the last two (St-RSI and S-RoC) are classified as window oscillators [44].

During the optimization process, three objective functions have been considered: the Return on Investment (ROI), Sortino-Ratio (SOR), and the number of trades (TR). The ROI is the ratio between the net gain of the investment and the investment costs [55].

$$ROI = \frac{Netgain}{Investment\ costs} \times 100 \quad (9)$$

SOR is a statistical measurement used to evaluate the riskiness of an investment. It is one of the powerful tools used to evaluate the performance of the trading strategy, where high SOR values could reveal the stability of the selected strategy. It is calculated as the investment's return relative to its downside volatility [56].

$$SOR = \frac{r - r_f}{\sigma_d} \quad (10)$$

where  $r$  is the aggregated return value obtained by applying the trading strategy,  $r_f$  is the risk-free rate, and  $\sigma_d$  is the downward standard deviation of returns.

The third objective function used is a calculation of the number of trades. The aim here is to minimize the number of trades, which in turn minimizes the trading costs.

The selected strategies are applied to the daily closing prices of Litecoin (LTC), which is one of the top three cryptocurrency trading volumes [18]. Two different time intervals were used. The first interval, starting from 3 January 2017 to 3 January 2019, was selected for training, while the second interval, starting from 3 January 2019 to 3 January 2021, was selected for testing, with 1460 data points divided equally between training and testing. These intervals were selected to test the robustness of the proposed methodology during the COVID-19 pandemic, as the crypto-market was highly influenced during that period [57].

### 3. Preliminaries of MOEA/D

In MOEA/D, the MOO problem is separated into a set of simultaneous single-objective Sub-Problems (SPs) with the help of an aggregation mechanism (i.e., a Scalarization Function (SF)). Several different SFs have been found in the literature. For example, Zhang [21] has evaluated three different SFs: Penalty-Boundary-Intersection (PBI), Weighted-Sum (WS), and Weighted-Tchebycheff (WTCH). Some other functions were also found in [28,30].

The Weighted Tchebycheff (WTCH), or weighted minimum-maximum, is one of the most suggested SFs as it is suitable for both convex and nonconvex problems [58]. As represented by Zhang [21], the PF of the MOO problem, shown in Equation (1), can be approximated by simplifying the problem into a set of SPs, such that the objective function of the  $j^{th}$  SP can be shown as follows:

$$Minimize_{g^{WTCH}(x, r^*)} = \max_{1 \leq i \leq N} \{w_i^j | f_i(x) - r_i^* | \} \tag{11}$$

$$* = (r_1^*, r_2^*, \dots, r_N^*) \quad T \quad * = \max\{f_i(x) | x \in \mathcal{S}\} \text{ for}$$

where  $r_i$  is the reference point, i.e.,  $r_i$

$i = 1 \rightarrow N$  objectives. As previously mentioned,  $\mathcal{S}$  is the decision space. For each SP,  $N$

there exists a different weight vector  $w = \{w_1, w_2, \dots, w_N\}$ , s.t.  $w_i \geq 0$ , and  $\sum_{i=1}^N w_i = 1$ .

In [21],

Zhang proposed a systematic weight distribution architecture, such that each

$\{w_1^1, w_1^2, \dots, w_1^H\}$  where  $H$  is a regulating integer parameter greater than 0,  $w \in [0, H]$

than 0. The number of SPs or weight vectors  $m = C_{H+N-1}^{N-1}$  (such that  $C$  denotes the mathematical combinations).

A neighborhood mechanism is employed such that each SP is optimized in accordance with its neighbors.

The neighborhood of the  $j^{th}$  SP contains the set of SPs that have weight vectors  $T$  distance away from  $j$ , where  $T$  is the neighborhood size. The basic steps of the original MOEA/D can be found in [21].

Due to the ability of the WTCH to solve various kinds of problems, different variants have been proposed to improve its performance [28,59]. The Augmented-WeightedTchebycheff (AWTCH) is one of the variants of the WTCH with a controlling parameter ( $\rho$ ) that is provided in order to improve the quality of the generated non-dominated solutions and to avoid the weak optimal ones, such that  $\rho$  is a very small value in the range 0.001 to 0.1 [42]. The AWTCH is calculated as follows:

$$g^{AWTCH}(x, r^*) = \max_{1 \leq i \leq N} \{w_i^j | f_i(x) - r_i^* | \} + \rho \sum_{i=1}^N |f_i(x) - r_i^*| \tag{12} \quad Minimize$$

Some MOO problems have objectives that have extremely different ranges, which in turn affect the overall performance. So, different normalization processes have been found in the literature [60,61]. Linear normalization is a straightforward method that is simple and efficient. The result of this normalization process is that all the objectives will have the same range of values from 0 to 1. The linearly normalized form of the WTCH scalarization method (N-WTCH) is shown as follows:

$$\text{Minimize } g^{\text{N-WTCH}}(x | w^j, r^*) = \max_{1 \leq i \leq N} \left\{ w_i \left| \frac{f_i(x) - r_i^{\text{nad}}}{r_i^* - r_i^{\text{nad}}} \right| \right\} \quad (13)$$

Such that  $r^*$  is the same as before, and  $r_i^{\text{nad}}$  is the minimum value of the objective space  $r_i^{\text{nad}} = \min\{f_i(x) | x \in S\}$ . As both  $r^*$  and  $r_i^{\text{nad}}$  are not previously known, they are adaptively changed during each iteration [62].

#### 4. The Proposed Algorithm

In this paper, a new decomposition-based Particle Swarm Optimization algorithm is proposed. The Normalized MOPSO/D (NMOPSO/D) is based on a linear normalization process to the AWTCH scalarization approach. The normalized form of the AWTCH (N-AWTCH) is calculated as:

$$\text{N-AWTCH}(x | w_j, r^*) = \frac{\sum_{i=1}^N w_i \left| \frac{f_i(x) - r_i^{\text{nad}}}{r_i^* - r_i^{\text{nad}}} \right|}{\max_{1 \leq i \leq N} \left\{ w_i \left| \frac{f_i(x) - r_i^{\text{nad}}}{r_i^* - r_i^{\text{nad}}} \right| \right\}} \quad (14)$$

Minimize  $g$

The proposed algorithm (Algorithm 1) is used to optimize the parameters of a set of four algorithmic trading paradigms over three objective functions. The optimization process is performed using the MOPSO/D algorithm, where the algorithm first initializes the particles' positions, velocities, and personal and global best positions. Each particle represents a separate SP, such that  $P^i(t)$  is the position of particle  $i$  at time  $t$ , while  $v^i(t)$  is the velocity of particle  $i$  at time  $t$ . For the application at hand, the particles' positions are the indicators' parameters. Each particle is assigned a weight vector that is used for fitness aggregation purposes. The Euclidean distance between each pair of particles, or SPs, is calculated in order to determine the particles that lie in the same neighborhood.

The evolution of the algorithm is obtained through a certain number of iterations. In order to find an optimum in the search space, the PSO formulation takes into account how the particles interact and move as a swarm. Over time, the particles cluster into an optimal location in the search space by using both exploration and exploitation. While the particles try to enhance or gain from the known promising locations through exploitation, the particles explore new areas of feasible space. Exploration and exploitation can be achieved and controlled during the velocity update through the inertia weight  $\omega$  as well as the personal and global best constant accelerators ( $c_1$  and  $c_2$ ). The inertia component is used for exploration principles as it is used to control the influence of the previous particle's velocity, i.e., search direction.

The personal and global best accelerators are used for exploitation purposes, such that  $c_1$  is the accelerator value that limits the step towards the particle's personal best position  $P_{pbest}^i$ , whereas  $c_2$  is used to limit the step toward the global best position  $P_{gbest}^i$ . The  $P_{pbest}^i$  is the best position that the particle achieved through the past iterations (the one that provided the best-aggregated fitness).

The  $P_{gbest}^i$  is the best obtainable position among the neighborhood of particle  $i$  through the past iterations. To improve the exploration facility of the proposed algorithm and avoid premature convergence, a uniform mutation operator is applied to the particles' positions and then the

related data structures, i.e.,  $P_{Pbest}^i$ ,  $P_{Gbest}^i$ , and reference point are updated. In uniform mutation, the value to be mutated is exchanged with a new uniformly randomly generated value within the predetermined range of each variable [63]. As the mutation improves the diversity of the particles through the search process, it sometimes negatively affects their convergence. To improve the exploitation of the algorithm and ensure convergence, a repair strategy is proposed that re-evaluates the particle's performance after mutation and ensures that it is not decayed; if so, the effect of the mutation is reversed, and the particle is returned to its previous position. An unbounded External Archive (EA) is used to hold the final non-dominated solutions found through the search process. Although using an unbounded archive increases computational complexity, it is very useful in keeping all the non-dominated solutions and not losing good solutions, which in turn improves diversity.

As shown in Figure 2, the algorithmic trader starts with randomly generated indicators' parameters (particles' positions) as well as a set of weight vectors. Applying the generated parameters to the crypto market produces a set of buy-sell signals. Such that the signals are generated based on the nature and the rules of every indicator (as shown in Section 2). These signals in turn are used to evaluate the objective functions at hand i.e., ROI, SOR, and trades.

**Algorithm 1:** NMOPSO/D algorithm for cryptocurrency trading

**Inputs:**

- Cryptocurrency historical data.
- $m$ : The swarm size (number of SPs).
- $N$ : The number of objectives at hand (3 objectives i.e., the ROI, SOR, and #trades).
- $T$ : The neighborhood size.
- A stopping condition (the stopping condition here is the number of iterations).

**Output:** External Archive (EA).

**Steps:**

**Repeat for each indicator:**

1. Initialization:

- Randomly generate a swarm of particles  $x^1 \rightarrow x^m$  at random such that  $m$  is the swarm size where,  $x^i$  is the current solution to the  $i$ th sub problem SP.
- For each particle, initiate the current position  $P^i$ , the personal-best-position  $P_{Pbest}^i$ , and the global-best-position  $P_{Gbest}^i$ . Such that  $P_{Pbest}^i$  and  $P_{Gbest}^i$  are initially set with the value of  $P^i$ .
- For each particle, initialize the velocity  $v^i = 0$ .
- Initialize the reference point  $r$ .
- Initialize the set of weight vectors  $W$  as shown by Algorithm 2.
- Calculate the Euclidean distances for each pair of weight vectors.
- Define  $B(i) = \{i_1, \dots, i_T\}$ , such that  $B(i)$  is the set of neighbors for each SP where  $w^{i_1}, \dots, w^{i_T}$  are the  $T$  closest weight vector to  $w^i$ .

2. Update:

For  $i = 1 \rightarrow m$ , do

2.1 Calculate the aggregation function:

$$g_{N-AWTCH}(x | w_j, r^*) = \max_{1 \leq i \leq N} \left\{ w^j \left| \frac{f(x) - r^*}{r_{inad}^{inad}} \right| \right\} + \rho \sum_{i \in B(i)} |f_i(x) - r^*|$$

2.2 According to the  $g_{N-AWTCH}$ , update  $P_{Pbest}^i$  and  $P_{Gbest}^i$ .

2.3 Update the reference point  $r$ .

2.4 Update the velocity as:

$$v_i^{(t+1)} = \omega v_i^{(t)} + c_1 r_1 (P_{Pbest}^i - P^i(t)) + c_2 r_2 (P_{Gbest}^i - P^i(t))$$

Such that:  $\omega$  is the inertia weight,  $c_1$  and  $c_2$  are two predetermined constant accelerators,  $r_1$ , and  $r_2$  are two random variables in the range  $[0, 1]$ .

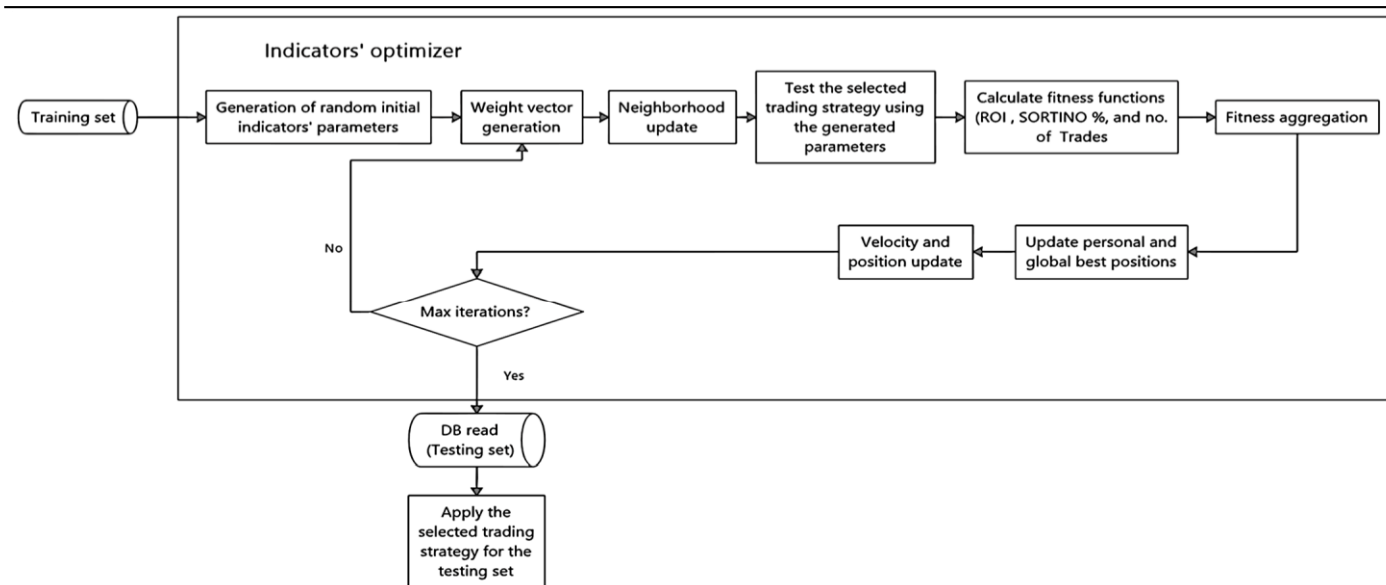
2.5 Update the current position as:

$$P^i(t+1) = P^i(t) + v^i(t+1)$$

2.6 Apply mutation operator.



- 2.7 Repair.
- 2.8 Update the EA.
- 2.9 Update weight vectors  $W$  (Algorithm 2).
- 2.10 Update neighborhood  $B(i)$ .
- 3. If the stopping criteria is met, then stop. Else go back to step 2.
- 4. Return EA.



**Figure 2.** The proposed cryptocurrency algorithmic trader model.

A fitness aggregation mechanism is then applied to aggregate the objectives. Based on the aggregated value for each particle, the personal and global best positions are updated, as well as the particles’ positions and velocities as shown in Algorithm 1. The same process is repeated with the updated positions. The algorithm stops when it reaches the maximum number of iterations. Finally, it returns the non-dominated solutions found during the search process that are then applied to the testing set.

*The Hybrid Weight Generation Mechanism*

As previously mentioned, the weight assignment mechanism is one of the main factors that affect the MOEA/D search process. Identical or poorly distributed weight vectors lead to poor solutions that are unable to cover the entire PF [21,64]. The weight vector generation approaches can be classified into systematic and random weight generation.

In systematic weight generation, the weights are constructed in repeated patterns to ensure evenly distributed vectors over the PF. The systematic distribution works very well in problems with regular or continuous PF; however, its performance reduces for problems with complex or scattered PFs [65]. The other problem with systematic distribution is that it sometimes provides similar or very close vectors, which in turn leads to redundant solutions [12].

On the other hand, the random weight creation can provide a more thorough exploration of the search space since it generates vectors that are not necessarily equally distributed across the PF. This in turn provides some unique and diverse solutions since it creates vectors that are dissimilar to one another. The problem with random weight generation is that there is no guarantee that the generated vectors could represent the entire PF [12]. In real-world problems, the search complexity increases as the PF is scattered and hardly covered in most cases. In such a scenario, a Pareto solution set to a challenging subproblem cannot be located through a simple search. To handle this problem, a new hybrid weight assignment mechanism has been proposed.

The proposed strategy merges both the systematic and random weight assignment approaches in order to utilize the advantages of both of them in a single algorithm. The proposed methodology allows to hit different random areas in the search space while maintaining the systematic distribution mechanism.

As shown by Algorithm 2, the total number of subproblems is divided into two partitions. In the first partition, the weights are systematically generated once at equal distances across the PF, while in the second partition, they are randomly generated during each iteration.

**Algorithm 2:** Hybrid weight generation algorithm

**Inputs:**

- $H$ : A regulating integer parameter greater than zero.
- $N$ : The number of objectives.
- $iter$ : The number of iterations.
- $m$ : The total number of subproblems.

**Output: The set of weight vectors  $W$ . Steps:**

1. Calculate the number of weight vectors  $m^1 = C_{H+N-1}^{N-1}$  (such that,  $C$  denotes the mathematical combinations and  $m_1 < m$ ).
  2. Let  $S$ , be a set of values in the range  $[0, 1]$  with an increment of  $\frac{1}{H}$ , such that  $S = \{0, \frac{1}{H}, \frac{2}{H}, \dots, \frac{H}{H}\}$ .
  3. For  $j = 1 \rightarrow m_1$ , do
    - For  $i = 1 \rightarrow N$ , do
      - Generate a new non repeated weight vector  $w_i$ , such that  $w_i \in S$  and  $\sum_{i=1}^{N_j} w_i = 1$ .
      - Append  $w_i$  to  $W_S$ , such that  $W_S$  is the set of the systematically generated weight vectors.
  4. For  $i = 1 \rightarrow iter$ , do For  $j = m_1 + 1 \rightarrow m$ , do For  $i = 1 \rightarrow N$ , do
    - Generate a weight vector  $w_i$ , with  $w_i$  randomly selected from  $[0, 1]$  and  $\sum_{i=1}^{N_j} w_i = 1$ .
    - Append  $w_i$  to  $W_R$ , such that  $W_R$  is the set of the uniformly randomly generated weight vectors.
- $W \leftarrow W_S \cup W_R$ .
5. Return  $W$ .

This process affects the neighborhood of each particle in the search space. Instead of having the same neighboring solutions during the search process, the neighbors are dynamically changed during each iteration, which in turn adds some more experience to the particles. The random assignment used here helps explore different areas that could hardly be reached using the systematic distribution alone, which aims at enhancing both convergence and diversity. In our experiments, the number of weight vectors that are systematically generated is set to  $\approx 80\%$  of the total number of SPs, while the rest of the weight vectors (20%) are randomly generated.

## 5. Results

The optimized algorithmic trading strategy is applied to Litecoin (LTC). The algorithm is trained first on the training data set, and then the set of the obtained optimized parameters is applied to the testing set. As mentioned before, the training and testing sets were selected in order to evaluate the performance of the proposed strategy during the COVID-19 pandemic. The cryptocurrency historical data used in this research is from CoinMarketCap and Yahoo Finance. The algorithmic trader is publicly available at <https://github.com/SherinOmran/CryptoMarket.git> (accessed on 23 September 2023).

### 5.1. Evaluation Metrics

The evaluation metric is an indicator or a measure of the quality of the generated solutions. There are various kinds of metrics to evaluate the MOO algorithms, each of which is used to evaluate different criteria [66]. Among the most important evaluation criteria are convergence, diversity, and statistical measurements. Different metrics have been found that evaluate either one or more criteria simultaneously. Three different indicators were used in this study: the Generational Distance (GD), the Hypervolume (HV), and the Average Fitness Value (AFV).

The GD is a convergence indicator that is used to measure the distance between the obtained non-dominated solutions (the obtained, also called the approximated PF) and the true PF [66,67]. Since the true PF is not known in real-world problems, a reference set that is obtained from the collection of the final non-dominated solution set among the approximated PFs found by all the considered search algorithms can be used instead. As a measure of how close the obtained solution set is to the true PF or the reference set, the lowest GD value is the best, and vice versa.

The HV indicator is a measure of both diversity and convergence. The HV quantifies the N-dimensional volume of the objective space area bounded by the approximated PF ( $y$ ) and a dominated reference point ( $r$ ) [66]. Higher values of the HV indicate a larger dominated area by the obtained approximated PF, so a higher HV is always preferred.

The AFV of each objective is calculated as the average of the fitness of the nondominated solutions found during  $R$  independent runs. Such that the total number of solutions are the number of non-dominated solutions found during each run multiplied by  $R$ . However, the number of non-dominated solutions is not necessarily the same during each run. The results are available at [https://drive.google.com/drive/folders/1V-bBCcs2L9\\_XoOrXasHa4aowTf0oj0C8a](https://drive.google.com/drive/folders/1V-bBCcs2L9_XoOrXasHa4aowTf0oj0C8a) (accessed on 23 September 2023).

### 5.2. Benchmark Problems

The proposed algorithm is first tested on three benchmark minimization problems with different PF shapes, i.e., ZDT2 [21], ZDT3 [21], and Viennet [68]. Such that, ZDT2 and ZDT3 are bi-objective problems, whereas the Viennet problem is a tri-objective problem.

To evaluate the performance of the hybrid assignment mechanism (MOPSO/D-Hyb), the proposed algorithm is compared against both Pareto-dominance-based algorithms, i.e., SPEA2 [69] and decomposition-based algorithms, i.e., the original MOEA/D and MOPSO/D with systematic weight distribution (MOPSO/D-Sys). For fair comparison and to evaluate the effect of the hybrid strategy clearly without any external improvements (specifically the normalization effect), all the three decomposition-based algorithms are implemented using the non-normalized WTC aggregation function. In this case, all the algorithm steps are the same as mentioned, except step 2.1 in Algorithm 1, which will be replaced with Equation (11).

The parameter settings for the bi-objectives benchmark problems are as follows: The number of SPs for each of the algorithms is set to 250, the number of iterations is 100, the neighborhood size ( $T$ ) is 20, a uniform mutation with a mutation rate of 0.25 (in this case, only one randomly selected decision variable is changed with each SP according to the mutation probability). The inertia weight  $w$  is 0.98, and both  $c_1$  and  $c_2$  are set to 2. For the original MOEA/D, a blend crossover with a crossover rate of 1 is used, whereas the other parameters are the same. The number of SPs for the Viennet problem is set to 435 with 150 iterations while the rest of the parameters are kept the same as the other problems.

The results of the benchmark problems are evaluated based on the GD and HV obtained by each algorithm on each benchmark problem, such that the reported results are the best and average values over 20 independent runs.

Table 1 shows the best and average GD and HV as well as their standard deviations (STD) obtained by each algorithm for ZDT2, ZDT3, and the Viennet benchmark problems in sequence. The best result in each case is highlighted in bold.

As seen from the table, the original MOEA/D always had the least performance in terms of all the evaluation metrics at hand. Although SPEA2 showed the best obtainable GD in the Viennet problem, it could not efficiently cover the whole PF as shown by the associated HV values. In ZDT3 problem, it showed a good approximation to the true PF as it provided the second-best average values for both GD and HV, whereas it provided the lowest performance in the ZDT2 problem for

all metrics. It can be seen that the MOPSO/D-Sys could provide good performance in the continuous optimization problem, i.e., ZDT2, as it provided the least average GD and could efficiently cover the whole PF. However, the STD of the GDs of the provided solutions is not the best. For the other benchmarks, the performance of MOPSO/D-Sys reduces in terms of both GD and HV, and the performance is not always stable, as can be seen through the STD of both metrics. On the other hand, the hybrid weight generation MOPSO/ D-Hyb provided the best performance in problems with scattered PFs, i.e., ZDT3, and Viennet. MOPSO/D-Hyb has the best coverage of the PF, having the highest HVs, and the closest solution set to the true PF as it has the least GD. For the continuous problem, i.e., ZDT2, it could always cover the whole PF, giving equal best and average HV with MOPSO/D-Sys. Although the hybrid distribution strategy could not find the best average GD, it could find the best obtainable GD. The hybrid distribution strategy provided the least STD over all the test instances, which emphasizes its stability during different runs.

**Table 1.** The GDs and HVs obtained by each algorithm for each benchmark problem.

		GD			HV		
		Best	Average	STD	Best	Average	STD
ZDT2		<b><math>1.2152 \times 10^{-7}</math></b>	$1.6424 \times 10^{-7}$	<b><math>2.2852 \times 10^{-8}</math></b>	<b><math>4.4890 \times 10^{-1}</math></b>	<b><math>4.4890 \times 10^{-1}</math></b>	<b><math>5.6953 \times 10^{-17}</math></b>
	SPEA2	$2.0350 \times 10^{-4}$	$2.5175 \times 10^{-4}$	$3.2318 \times 10^{-5}$	$4.4380 \times 10^{-1}$	$4.4234 \times 10^{-1}$	$1.0559 \times 10^{-3}$
	MOEA/D	$5.7890 \times 10^{-7}$	$8.0228 \times 10^{-7}$	$9.6221 \times 10^{-8}$	$4.4620 \times 10^{-1}$	$4.4480 \times 10^{-1}$	$6.9510 \times 10^{-4}$
	MOPSO/D-Sys	$1.2353 \times 10^{-7}$	<b><math>1.5280 \times 10^{-7}</math></b>	$3.5567 \times 10^{-8}$	<b><math>4.4890 \times 10^{-1}</math></b>	<b><math>4.4890 \times 10^{-1}</math></b>	<b><math>5.6953 \times 10^{-17}</math></b>
	MOPSO/D-Hyb						
ZDT3	SPEA2	$8.7146 \times 10^{-5}$	$1.4222 \times 10^{-4}$	$7.8160 \times 10^{-5}$	$6.0030 \times 10^{-1}$	$5.9765 \times 10^{-1}$	$4.6254 \times 10^{-1}$
	MOEA/D	$2.9423 \times 10^{-5}$	$1.1886 \times 10^{-4}$	$9.9334 \times 10^{-5}$	$5.9890 \times 10^{-1}$	$5.9646 \times 10^{-1}$	$1.2709 \times 10^{-3}$
	MOPSO/D-Sys	$7.2900 \times 10^{-6}$	$4.5964 \times 10^{-4}$	$1.5465 \times 10^{-3}$	$6.0090 \times 10^{-1}$	$5.9687 \times 10^{-1}$	$1.1177 \times 10^{-2}$
	MOPSO/D-Hyb						
	Viennet	SPEA2	<b><math>1.6463 \times 10^{-4}</math></b>	$1.9837 \times 10^{-4}$	$1.8045 \times 10^{-5}$	$6.2180 \times 10^{-1}$	$6.1925 \times 10^{-1}$
	MOEA/D	$2.688 \times 10^{-4}$	$7.7789 \times 10^{-4}$	$2.864 \times 10^4$	$6.218 \times 10^{-1}$	$6.177 \times 10^{-1}$	$2.520 \times 10^{-3}$
	MOPSO/D-Sys	$2.138 \times 10^{-4}$	$2.3292 \times 10^{-4}$	$1.577 \times 10^{-5}$	$6.246 \times 10^{-1}$	$6.227 \times 10^{-1}$	$9.080 \times 10^{-4}$
	MOPSO/D-Hyb	$1.745 \times 10^{-4}$	<b><math>1.9829 \times 10^{-4}</math></b>	<b><math>1.458 \times 10^{-5}</math></b>	<b><math>6.251 \times 10^{-1}</math></b>	<b><math>6.233 \times 10^{-1}</math></b>	<b><math>8.113 \times 10^{-4}</math></b>
		<b><math>6.9152 \times 10^{-6}</math></b>	<b><math>2.8271 \times 10^{-5}</math></b>	<b><math>2.5840 \times 10^{-5}</math></b>	<b><math>6.0160 \times 10^{-1}</math></b>	<b><math>6.0020 \times 10^{-1}</math></b>	<b><math>9.9233 \times 10^{-4}</math></b>

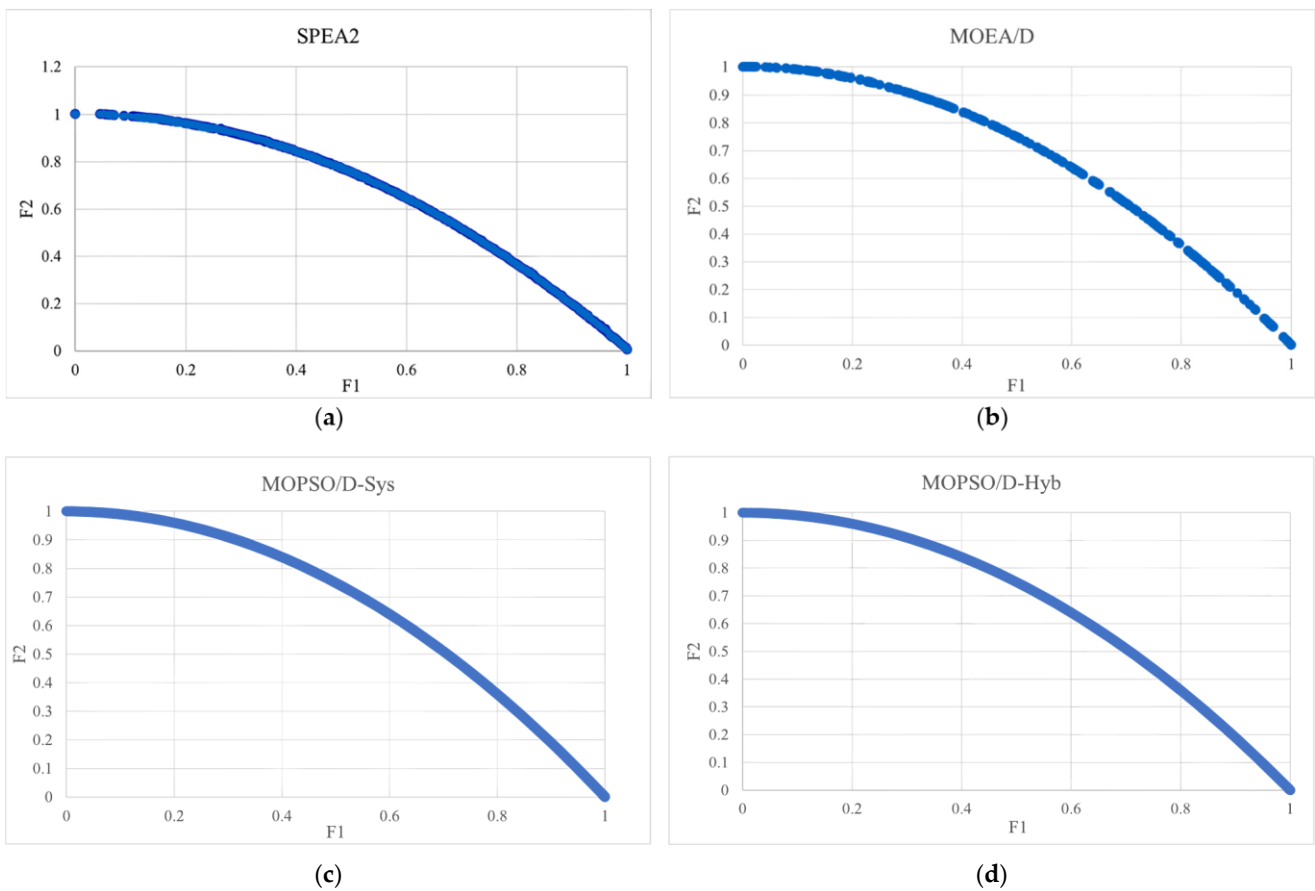
For more clarity, the PF solutions with the median GD for each algorithm over the different problems are presented in the following figures, such that Figure 3 shows the PFs obtained by the algorithms under study for the ZDT2 problem. Figure 4 presents the PFs for the ZDT3 problem, whereas Figure 5 shows the obtained PFs for the Viennet optimization problem. It can be seen from the figures that the hybrid weight generation could maintain the best distribution of solutions along the entire PFs for all the test instances. MOPSO/ D-Sys showed the best distribution of solutions for only ZDT2. The MOEA/D algorithm always provided the least performance and could not efficiently cover the entire PFs for all the test instances. For SPEA2, it could efficiently cover the PF only in the ZDT3 problem.

### 5.3. The Algorithmic Trader Experimental Results

As seen above, the hybrid weight distribution methodology could provide the best results for the benchmark problems. For the algorithmic trader optimizer problem, four algorithms are used in comparison. The original MOEA/D is tested against the MOPSO/D using the normalized WTCH scalarization (N-WTCH). The other two are MOPSO/D using the proposed normalized AWTC

presented in Equation (14); one uses the systematic weight distribution denoted as (N-AWTCH-Sys), whereas the other uses the hybrid weight distribution denoted as (N-AWTCH-Hyb).

The parameter settings are as follows: The swarm size  $m = 351$ , where the organizing parameter  $H = 25$ . For the N-AWTCH-Hyb, the swarm is divided into two partitions; the first partition used for systematic weight distribution is chosen with  $H = 22$ , which generates 276 SPs, while the other partition size is 75 to keep the same total number of SPs (351 as the other algorithms). The maximum number of iterations is 200. The augmentation parameter  $\rho$  is set to 0.05. A small neighborhood size  $T = 20$  is selected to maintain the solutions' diversity. For PSO parameters, the inertia parameter  $\omega$  is 0.98, whereas the constant accelerators  $c_1$  and  $c_2$  are set to 2 and the mutation rate is 0.15. For MOEA/D, the crossover rate is 1 and the mutation rate is 0.15. The indicators' parameters ranges are as follows: The number of days for any of the indicators ranges from 3 to 200 days. The St-RSI over-bought level  $\in [60, 90]$ , whereas the over-sold level  $\in [10, 40]$ . The S-RoC over-bought and over-sold levels range from 0 to 20% above and below the center line.



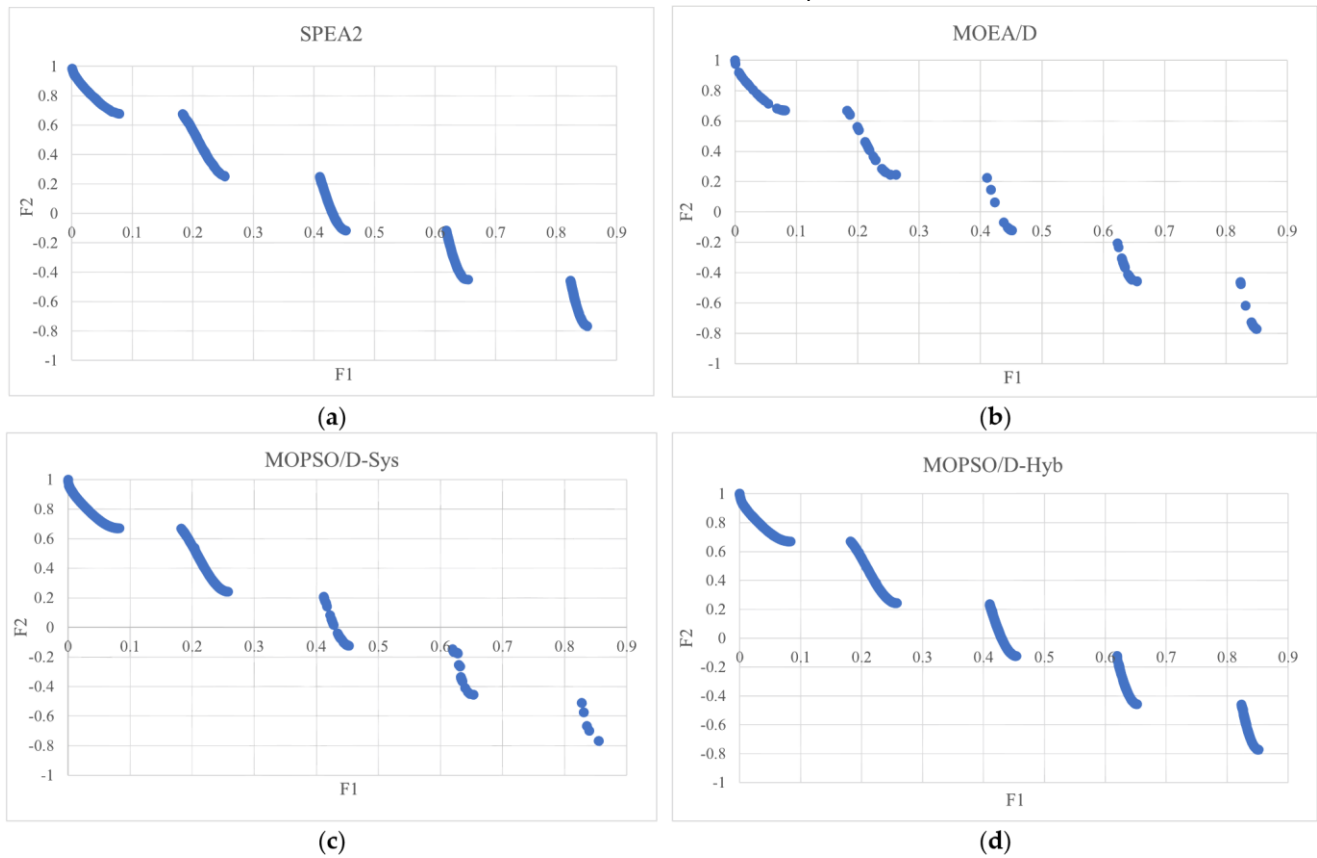
**Figure 3.** The set of Pareto optimal solutions found by SPEA2 (a), MOEA/D (b), MOPSO/D-Sys (c), and MOPSO/D-Hyb (d) for the ZDT2 problem.

As mentioned before, the algorithms are evaluated based on the GD, the HV, and the AFVs found by each algorithm over five independent runs.

Table 2 shows the best and average HV and GD values obtained by optimizing the four trading strategies, i.e., L-WMA, St-RSI, S-ROC, and BB, using the four counterpart algorithms. The best results in each case are highlighted in bold. The results show clearly that the proposed N-AWTCH could outperform both the original MOEA/D and the NWTCH in terms of the distribution of solutions and the distance to the true PF. Adding the hybrid distribution to the N-AWTCH could further improve the accuracy of the obtained results for most of the test instances.

Figures 6 and 7 show the approximated PFs obtained by optimizing the L-WMA and the St-RSI indicators in sequence. The two indicators, i.e., L-WMA and St-RSI, are selected as samples to figure out the shape of the true PF for each case and the distribution of the results. The figures show a comparison between the approximated PF solutions obtained by each of the algorithms

under study and the true PF, such that the approximated PFs shown are the final solutions obtained from the run with the least GD. As seen from the figures, the shape of the true PFs is different for each test instance; however, they are all scattered.

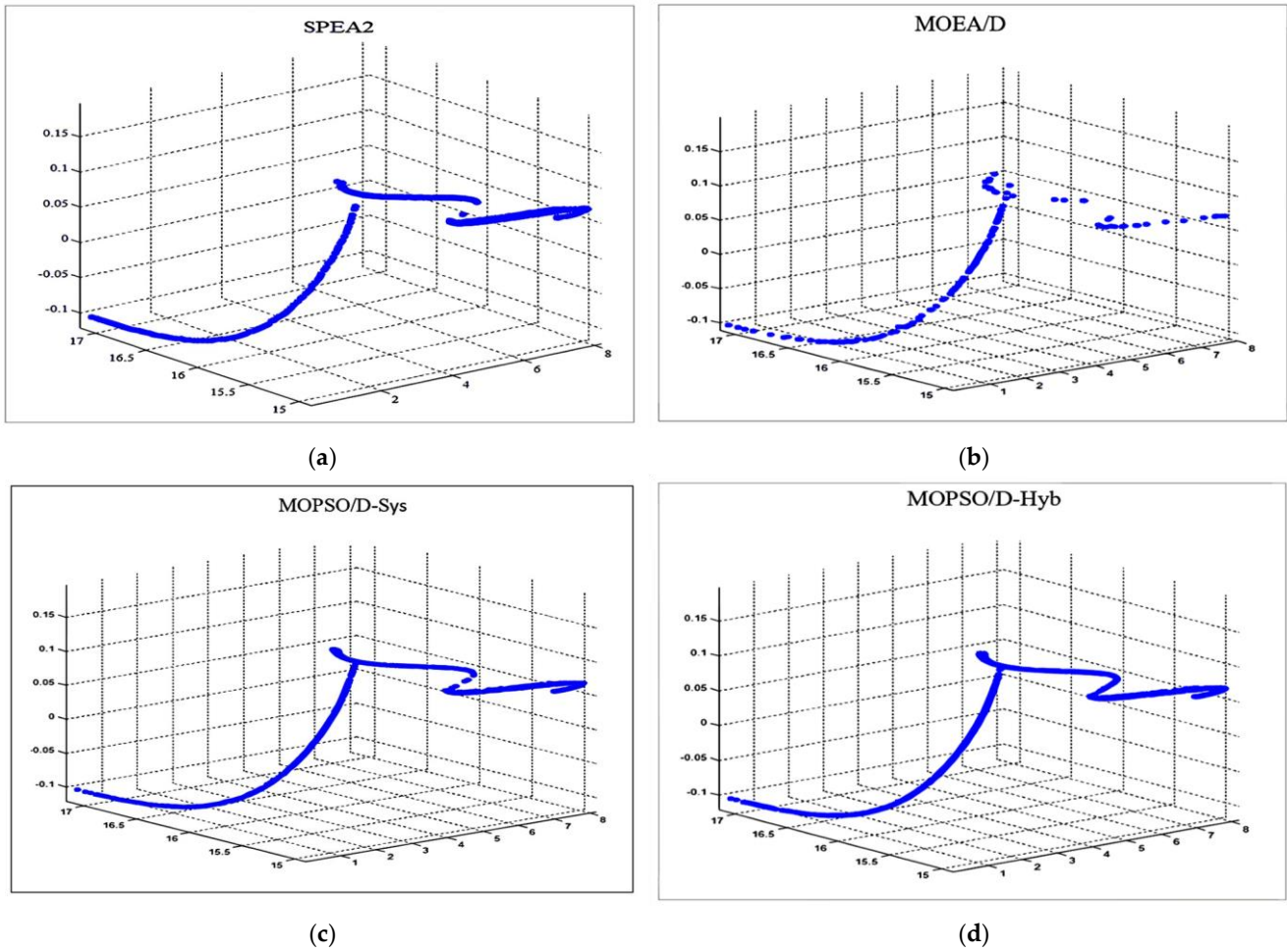


**Figure 4.** The set of Pareto optimal solutions found by SPEA2 (a), MOEA/D (b), MOPSO/D–Sys (c), and MOPSO/D–Hyb (d) for the ZDT3 problem.

To evaluate the results obtained by each algorithm in accordance with the indicators using their standard parameters, the AFVs for the final set of non-dominated solutions obtained by each algorithm are calculated. The non-dominated solutions are a set of vectors, where each vector has three components, one for each objective, i.e., ROI, SOR, and number of trades. The AFV is calculated by averaging the objectives of the non-dominated PF solutions over a set of five independent runs for each test instance in training.

During testing, the solutions obtained by each algorithm are re-evaluated, and the performance is averaged in the same way as in the training period. As is the case of all MOO problems, there are always tradeoffs between the objectives, so in order to compare the overall performance of each of the algorithms with the indicators’ standard parameters, each of the competitors is given a rank associated with each objective. The best value is given a rank of 1, and the second best is given a rank of 2, etc., while the worst value is assigned a rank of 5. The comparison includes the four counterpart algorithms and the indicator’s performance using their standard parameters.

Tables 3 and 4 show a comparison of the AFVs by each algorithm for each trading strategy, i.e., TI, during the training and testing periods in sequence. The results reported as standard are the values of the objectives obtained by trading using the standard parameters of each indicator.



**Figure 5.** The set of Pareto optimal solutions found by SPEA2 (a), MOEA/D (b), MOPSO/D-Sys (c), and MOPSO/D-Hyb (d) for the Viennet problem.

**Table 2.** The best and average GD and HV values obtained by each algorithm for the set of optimized indicators.

		GD		HV	
		Best	Average	Best	Average
L-WMA	MOEA/D	$1.35 \times 10^{-4}$	$9.00 \times 10^{-4}$	$4.76 \times 10^{-1}$	$4.73 \times 10^{-1}$
	N-WTCH	$1.20 \times 10^{-3}$	$6.96 \times 10^{-2}$	$4.45 \times 10^{-1}$	$4.11 \times 10^{-1}$
	N-AWTCH-Sys	$1.50 \times 10^{-3}$	$1.93 \times 10^{-2}$	$4.76 \times 10^{-1}$	$4.26 \times 10^{-1}$
	N-AWTCH-Hyb	<b><math>9.07 \times 10^{-5}</math></b>	<b><math>2.38 \times 10^{-4}</math></b>	<b><math>4.82 \times 10^{-1}</math></b>	<b><math>4.78 \times 10^{-1}</math></b>
St_RSI	MOEA/D	$0.344 \times 10^1$	$0.768 \times 10^1$	$3.75 \times 10^{-1}$	$3.30 \times 10^{-1}$
	N-WTCH	$1.11 \times 10^{-1}$	$1.51 \times 10^{-1}$	$4.75 \times 10^{-1}$	$4.68 \times 10^{-1}$
	N-AWTCH-Sys	$8.33 \times 10^{-2}$	$1.38 \times 10^{-1}$	$4.76 \times 10^{-1}$	$4.68 \times 10^{-1}$
	N-AWTCH-Hyb	<b><math>6.17 \times 10^{-2}</math></b>	<b><math>1.07 \times 10^{-1}</math></b>	<b><math>4.88 \times 10^{-1}</math></b>	<b><math>4.70 \times 10^{-1}</math></b>
S-RoC	MOEA/D	$2.84 \times 10^{-2}$	$2.34 \times 10^{-1}$	$5.04 \times 10^{-1}$	$4.21 \times 10^{-1}$
	N-WTCH	$1.56 \times 10^{-2}$	$7.61 \times 10^{-2}$	$6.97 \times 10^{-1}$	$6.84 \times 10^{-1}$
	N-AWTCH-Sys	$5.50 \times 10^{-3}$	$1.12 \times 10^{-2}$	<b><math>7.20 \times 10^{-1}</math></b>	<b><math>7.01 \times 10^{-1}</math></b>
	N-AWTCH-Hyb	<b><math>5.30 \times 10^{-3}</math></b>	<b><math>9.38 \times 10^{-3}</math></b>	$7.02 \times 10^{-1}$	$6.96 \times 10^{-1}$

**Table 2. Cont.**

GD

HV

	Best	Average	Best	Average
BB		$92 \times 10^{-3}$	$6.877 \times 10^{-1}$	$6.875 \times 10^{-1}$
		$77 \times 10^{-3}$	$6.903 \times 10^{-1}$	$6.897 \times 10^{-1}$
		$80 \times 10^{-4}$	$6.897 \times 10^{-1}$	$6.845 \times 10^{-1}$
		$25 \times 10^{-3}$	$6.903 \times 10^{-1}$	$6.900 \times 10^{-1}$

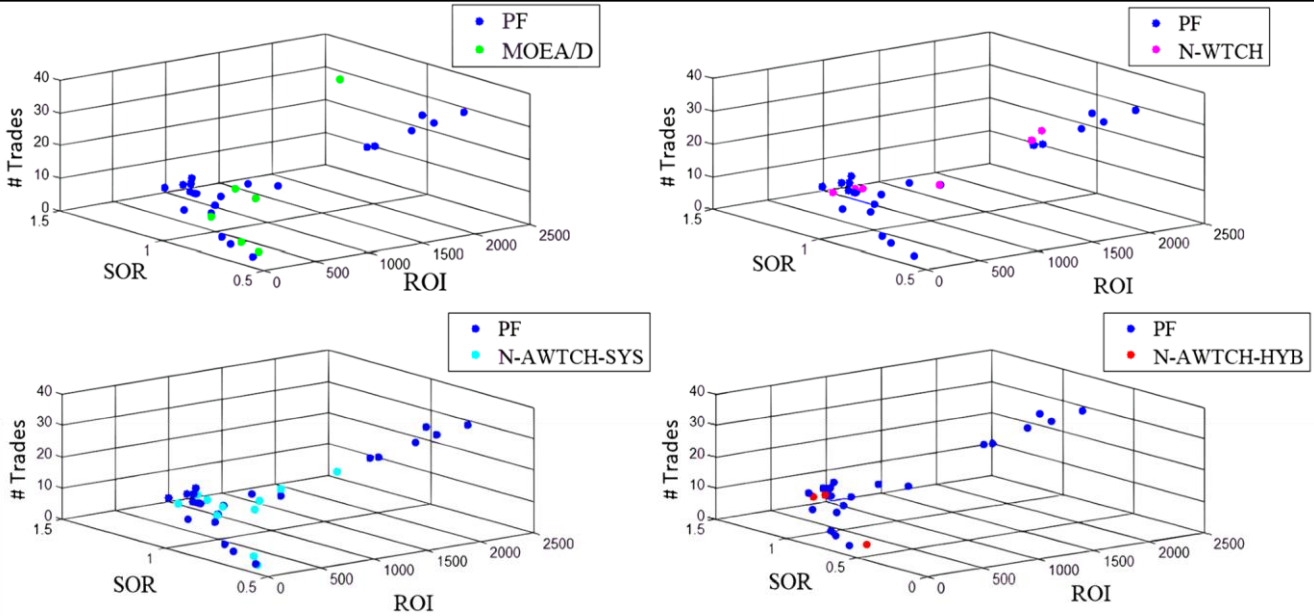


Figure 6. The approximated PF versus the true PF for L-WMA indicator optimization of each algorithm for LTC trading.

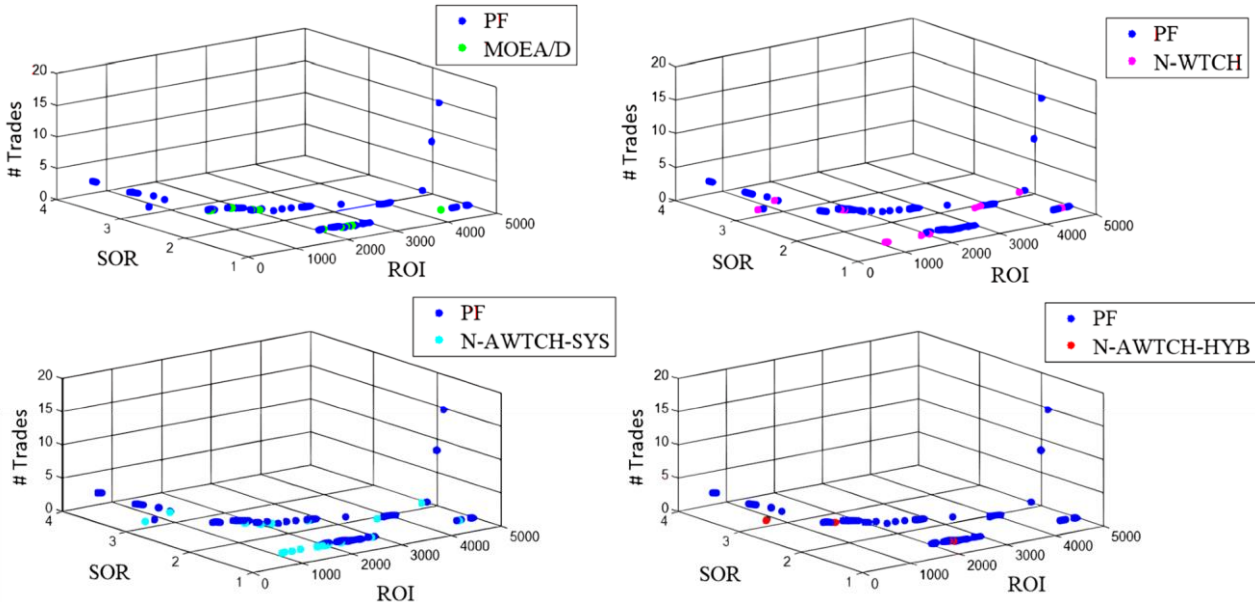


Figure 7. The approximated PF versus the true PF for St-RSI indicator optimization of each algorithm for LTC trading.



**Table 3.** The AFVs obtained for each indicator using the algorithms under study during the training period for LTC.

Standard		MOEA/D			N-WTCH			N-AWTCH-Sys			N-AWTCH-Hyb			
ROI%	SOR	#Trades	ROI%	SOR	#Trades	ROI%	SOR	#Trades	ROI%	SOR	#Trades	ROI%	SOR	#Trades
	10.00	1676.62	1.43	<b>2.59</b>	2101.66	1.42	3.01	2088.16	1.34	2.69	2178.14	<b>1.52</b>	3.33	
	5	5	2	1	4	3	3	3	4	2	2	1	4	
	61.00	306.34	0.56	35.95	791.89	<b>0.96</b>	12.90	693.04	0.93	<b>11.72</b>	<b>834.94</b>	0.95	13.32	
	5	4	4	4	2	1	2	3	3	1	1	2	3	
	4.00	178.67	<b>25.23</b>	<b>2.48</b>	411.19	24.26	7.11	<b>416.39</b>	18.90	7.18	403.17	20.01	6.80	
	2	4	1	1	2	2	4	1	4	5	3	3	3	
	<b>2.000</b>	306.62	<b>17.68</b>	2.553	625.95	12.12	2.59	651.408	11.94	2.583	<b>652.253</b>	11.92	2.583	
	1	4	1	2	3	2	5	2	5	3	1	3	3	
			33			33			36			29		

**Table 4.** The AFVs obtained for each indicator using the algorithms under study during the testing period for LTC.

Standard		MOEA/D			N-WTCH			N-AWTCH-Sys			N-AWTCH-Hyb					
ROI%	SOR	#Trades	ROI%	SOR	#Trades	ROI%	SOR	#Trades	ROI%	SOR	#Trades	ROI%	SOR	#Trades		
L-WMA	AFV Rank	<b>266.81</b>	<b>0.78</b>	10.00	70.81	0.62	4.83	91.95	0.68	19.83	92.54	0.67	4.75	118.63	0.69	5.29
		1	1	4	5	5	2	4	3	5	3	4	1	2	2	3
St-RSI	AFV Rank	-85.02	-0.32	58.00	43.01	0.09	40.51	23.43	0.30	19.83	25.42	<b>0.31</b>	<b>18.05</b>	29.5736	<b>0.31</b>	20.51
		5	5	5	1	4	4	4	3	2	3	1	1	2	1	3
S-RoC	AFV Rank	-30.39	-0.07	5.00	69.74	2.02	0.91	144.63	2.10	5.02	148.74	2.21	5.09	<b>150.85</b>	<b>2.26</b>	<b>4.82</b>
		5	5	2	4	4	4	3	3	3	2	2	5	1	1	1
BB	AFV Rank	107.87	<b>3.00</b>	<b>1.00</b>	106.48	2.07	3.45	115.28	2.43	2.59	117.24	2.44	2.60	<b>118.83</b>	2.46	2.50
		4	1	1	5	5	5	3	4	3	2	3	4	1	2	2
	Sum of ranks		39			48			40			31			21	

For example, as shown in Table 3, applying the L-WMA indicator using the crossovers between the 20–50 days L-WMAs to generate the trading signals, i.e., buy and sell, gives a set of 10 trades (each buy-sell pair is considered a single trade) with an overall ROI of 2677.46% and a SOR of 0.97, such that SOR is the measure of risk. By applying the same parameters to the L-WMA indicator during testing (Table 4), the overall ROI is 266.81% with a SOR of 0.78 over 10 trades.

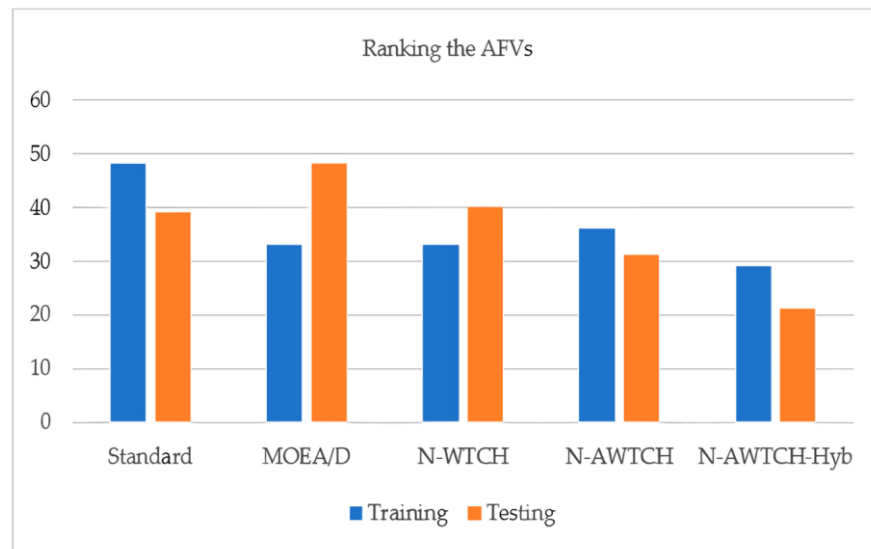
The best AFV obtained in each case is highlighted in bold and given a rank of 1. As mentioned before, the optimization problem at hand has three objectives. Two of them are to be maximized, i.e., ROI and SOR, whereas the third objective is to be minimized, i.e., number of trades. So that the highest ROI values, SOR, and the minimum number of trades are highlighted in each case.

As seen from Table 4, the average number of trades is less than one in some cases. That is because some of the obtained parameters did not generate trading signals during testing. In such a case, values less than one cannot be considered the best values. To handle this limitation, the algorithm that provided an average number of trades less than one is assigned the same rank as the ROI.

To evaluate the results presented in the last tables, the ranks obtained by each algorithm are summed, and the algorithm with the least summation of ranks is preferred. The summation of the ranks obtained by each algorithm during both training and testing is clarified in Figure 8.

For example, by going back to Table 3, the sum of ranks for MOEA/D during training is 33. The ranks assigned to the MOEA/D algorithm for L-WMA (5, 2, 1) for ROI, SOR, and number of trades in sequence, the ranks for St-RSI are (4, 4, 4), the ranks assigned to S-RoC are (4, 1, 1), and

the ranks for BB are (4, 1, 2). The summation of the ranks obtained by each case gives an overall rank of 33, and so on for the rest of the competitive algorithms and the standard trading strategies.



**Figure 8.** The summation of the ranks for each algorithm and the indicators<sup>0</sup> standard parameters during both training and testing.

It is obvious from the figure that N-AWTCH-Hyb has the best ranking during both training and testing, followed by both MOEA/D and N-WTCH during training, and NAWTCH-Sys during testing. As seen from the results, the proposed algorithm provided the best trading strategies as compared to the other counterparts.

## 6. Discussion

Finding the best trading strategies is a great challenge for traders, as they always aim to find the most profitable strategy with the minimum level of risk. The crypto market is one of the most challenging markets due to the sudden daily fluctuations in prices. This research aims at finding the optimal parameters for four different trading strategies (also called TIs), i.e., L-WMA, St-RSI, S-RoC, and BB. The proposed algorithm is applied for one of the highest volume cryptocurrencies, i.e., LTC, using decomposition-based strategies, taking into consideration three conflicting objectives, i.e., ROI, SOR, and number of trades.

The first two objectives are to be maximized whereas the last one is to be minimized.

As seen from the figures, the true PF shapes for our test instances are complex and scattered, with no definite shape. In this research, a MOPSO/D algorithm is proposed with a hybrid weight generation strategy that merges both systematic and random weight generations in order to handle the complexity of the PFs for the problem at hand.

In general, weight vector generation is one of the main factors that affect the performance of decomposition-based algorithms, as these weights are used to determine the neighborhood of each SP, which is important for recombination purposes and the generation of new solutions. Zhang [21] suggested that the recombination be performed within the same neighborhood;

however, some argued that this is not efficient for more complex problems and suggested recombination with solutions from different neighborhoods instead [69].

With the proposed hybrid weight generation strategy, the recombination process (represented by position update) is performed with the help of the information gained from the same neighborhood; however, some particles (specifically, the particles with random weight generation) move through different neighborhoods over the iterations. This in turn improves the experience of the particles over time and ensures both diversity and convergence.

To test the performance of the proposed algorithm before applying it to our real-world problem, it was tested on three different benchmark problems with different PF shapes, i.e., ZDT2, ZDT3, and Viennet, such that ZDT2 has a concave continuous PF, whereas ZDT3 and Viennet have more complex and discontinuous PFs. It was proved that the proposed MOPSO/D-Hyb could overcome the performance of both the original MOEA/D and MOPSO/D using the systematic weight generation (MOPSO/D-Sys) in terms of all the evaluation metrics for the test instances at hand except for the average GD for ZDT2 problem, which was slightly worse than MOPSO/D-Sys. This means that the proposed algorithm could efficiently cover the true PF for complex problems, with discontinuity covering all the scattered partitions.

As the objectives of our real-world optimization problem have a very wide and extensively different range of variables, there was a need for an objective normalization process.

In this research, a new linearly normalized AWTCH (N-AWTCH) scalarization method has been proposed. The proposed algorithm was implemented in both systematic (N-AWTCH-Sys) and hybrid (N-AWTCH-Hyb) weight distribution strategies and is compared against the MOPSO/D using normalized WTH (denoted as N-WTCH) and the original MOEA/D.

As seen from Table 2, the proposed normalization method N-AWTCH-Sys could obtain better solutions in terms of GD and HV as compared to the original MOEA/D and N-WTCH, where it could outperform them in three out of the four test instances in terms of the average GD and two test instances in terms of the average HV.

As compared to the hybrid weight distribution strategy, the N-AWTCH-Hyb could further improve the performance of the N-AWTCH-Sys in terms of both GD and HV obtaining the best results in three out of the four test instances as compared to the other counterparts.

As noticed from the figures Figures 6 and 7, the number of true PF points is limited that is due to many reasons. Among the factors that affect the number of true PF points is the relationship between the objectives, such that the objectives at hand are highly skewed. The second reason is the effect of the market changes during the period under study (i.e., the COVID-19 pandemic), such that a large number of parameter combinations are found to be non-sensitive to the market changes, generating either non-profitable trades or no trades. Moreover, some parameter combinations are found to generate the same trading signals.

To examine the performance of the proposed algorithm in terms of the efficiency of the generated trading strategies, the AFVs of the algorithms under study as well as the indicators using their standard parameters are calculated and ranked, such that the lowest rank is the best and the highest is the worst. It was found that the proposed algorithm (N-AWTCH-Hyb) obtains the best summation of ranks during both the training and testing periods.

By examining the ranking of the proposed algorithm in terms of each objective individually, we can find that it has the best ranking in terms of both ROI and SOR during both training and testing periods. In terms of the number of trades, it was found that the proposed algorithm sometimes provides a high number of trades during training; however, these trades are profitable. During testing, it could also provide the best ranking in terms of the number of trades.

## 7. Conclusions

The crypto market is extremely risky and volatile, which makes it hard to predict. An optimized algorithmic trading approach using four independent technical indicators (i.e., L-WMA, St-RSI, S-RoC, and BB) has been proposed, taking into consideration the trade-offs between different objectives. A Normalized MOPSO/D has been used to optimize the algorithmic trading

over three different objectives: ROI, SOR, and number of trades. The normalization mechanism is applied for the Augmented Weighted-Tchebycheff (A-WTCH) scalarization function (N-AWTCH). As real-world optimization, problems always have complex PFs that are hardly covered with the regular weight generation strategy of the original decomposition approach, a hybrid weight distribution strategy is proposed that combines both systematic and random weight generations. In this context, a single grid of weight vectors is generated and partitioned such that the first partition is assigned constant and equally distant weight vectors that are generated once, whereas the other partition is randomly generated during each iteration. To evaluate the performance of the hybrid weight distribution strategy, it was first applied to some benchmark problems with different PF shapes and tested against the original MOEA/D and MOPSO/D with systematic weight distribution. The proposed algorithm was found to optimize the PF finding the best Generational Distance (GD) and Hyper Volume (HV) for problems with irregular PF shapes (i.e., noncontinuous and degenerated), which emphasizes the ability of the proposed weight generation strategy to handle complex PFs.

The proposed algorithm combining the N-AWTCH aggregation function as well as the hybrid weight distribution (N-AWTCH-Hyb) is applied to the real-world problem at hand and is compared against the N-AWTCH using systematic weight distribution (N-AWTCHSys), the normalized version of the WTCH scalarization (N-WTCH), the original MOEA/D, as well as the indicators with their standard parameters. Three different evaluation criteria have been taken into account: the GD, the HV, and the Average Fitness Values (AFV). The proposed approaches were tested on Litecoin, across training and testing sets. The

COVID-19 outbreak has been taken into consideration such that the training period is the pre-pandemic period, whereas the testing period is considered the pandemic period.

Results showed that the proposed N-AWTCH-Hyb outperformed the other counterpart algorithms in 75% of the test instances in terms of both convergence and diversity indicators (i.e., GD and HV). In terms of the AFVs, the proposed strategy provided the best ranking as compared to all the competitive algorithms as well as the indicators with their standard parameters. Although there were extreme market changes during both training and testing due to the outbreak effect, the optimized trading strategy using the proposed algorithm revealed its stability over the other counterparts during both training and testing periods (which is the main challenge).

**Author Contributions:** Conceptualization, A.A.A.Y. and W.H.E.-B.; data curation, S.M.O.; formal analysis, S.M.O. and A.A.A.Y.; investigation, W.H.E.-B.; methodology, S.M.O.; supervision, A.A.A.Y.; validation, W.H.E.-B.; writing—original draft, S.M.O.; writing—review and editing, A.A.A.Y. and W.H.E.-B. All authors have read and agreed to the published version of the manuscript.

**Funding:** This research received no external funding.

**Informed Consent Statement:** Informed consent is not required for this study.

**Data Availability Statement:** The data used in this study is publicly available on CoinMarketCap.

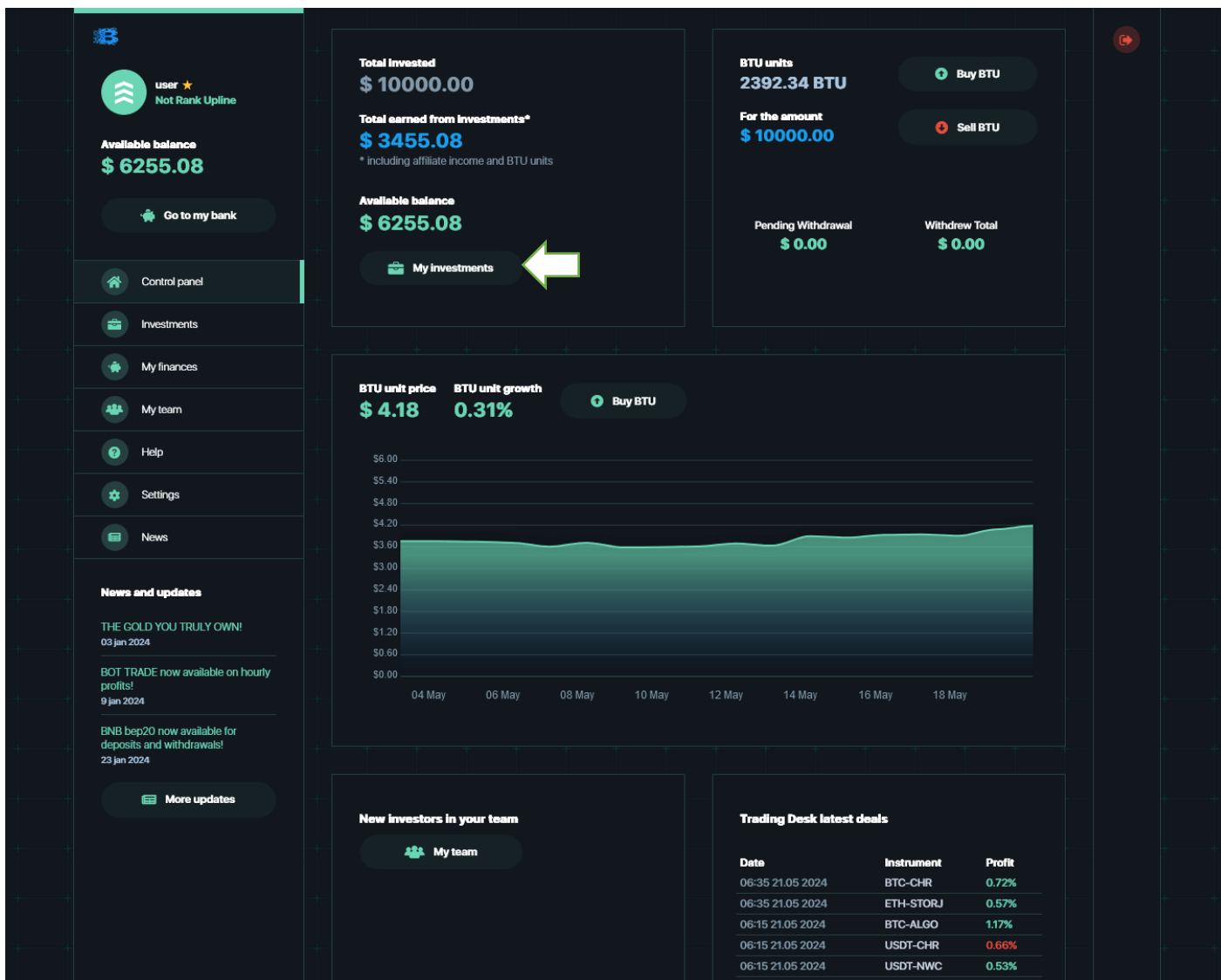
**Conflicts of Interest:** All authors declare that they have no conflict of interest.

## Virtual directions of how to subscribe to a “BOT TRADE” package in Blockstake.ai

Blockchain.com inc, built these algorithms with perfect precision to avoid high frequency trading, so as to maintain regular profits.

Below are graphic directions of how to subscribe to “Bot trade” package. There are 4 bots’ stages under this package and each bot has a minimum and maximum deposit amount with an hourly profit percentage attached to it.

**NOTE:** these percentages are not stable and can change without notifying the subscribers.



This is a virtual representation of what a private account looks like. “complete analysis”

Clicking on “my investments” will take you to the investment page where all the packages are fully analyzed.

The dashboard features a dark theme with a sidebar on the left containing navigation options: Control panel, Investments, My finances, My team, Help, Settings, and News. A 'user' profile is shown with a 'Not Rank Upline' status and an available balance of \$6165.08. A 'Go to my bank' button is also present.

The main content area is titled 'My active investments' and includes a 'Create a new investment' button. A large graphic displays interest rates: 0.9% per day on business days and 0.8% per day on weekends.

Summary statistics on the right show: Total Invested \$10000.00, Total earned from Investments\* \$3365.08, and Available balance \$6165.08. A 'My Investments' button is located below.

The 'My active investments' section is divided into three categories:

- simple earn**: A table with columns 'Plan', 'Deposit Amount', and 'Daily Profit (%)'. It lists a '10 days project' with a deposit range of \$10.00 - \$2000.00 and a daily profit of 0.98. An 'Activate Now' button is provided.
- Project staking (30 days)**: A table with columns 'Plan', 'Deposit Amount', and 'Daily Profit (%)'. It lists three plans: Plan 1 (\$2000.01 - \$30000.00, 0.69%), Plan 2 (\$30000.01 - \$60000.00, 1.05%), and Plan 3 + 35% compounding (\$60000.01 and more, 1.45%). An 'Activate Now' button is provided.
- Defi staking (flexible)**: A table with columns 'Plan', 'Deposit Amount', and 'Daily Profit (%)'. It lists 'Plan 1' with a deposit of '\$10000.00 and more' and a daily profit of 0.90. Below this is a table for 'Your deposits' with columns 'Date', 'Amount', 'PS', and 'Release'. A deposit on 'Apr-13-2024 01:07:48 AM' for \$10000.00 is shown with a 'Release' button. A red arrow points to this button.

Below the Defi staking section are two more investment options:

- Bot trade (30 days)**: A table with columns 'Plan', 'Deposit Amount', and 'Hourly Profit (%)'. It lists three bot plans: Bot 1 (\$10000.00 - \$49999.99, 0.06%), Bot 2 (\$50000.00 - \$99999.99, 0.09%), and Bot 3 + 35% compounding (\$100000.00 and more, 0.15%). An 'Activate Now' button is provided.
- Bot trade (flexible)**: A table with columns 'Plan', 'Deposit Amount', and 'Hourly Profit (%)'. It lists a plan for 'Bot (max) + 50% interest compounding' with a deposit of '\$250000.00 and more' and an hourly profit of 0.16%. An 'Activate Now' button is provided.

At the bottom, there is a 'Save to my balance' section with a table for 'Plan', 'Deposit Amount', and 'Profit (%)'.

Callouts: A blue arrow points to the 'Create a new investment' button in the sidebar. A red arrow points to the 'Release' button in the Defi staking section. Two white arrows point to the 'Bot trade (30 days)' and 'Bot trade (flexible)' sections.

Here's the full analysis of all packages. "BOT TRADE" and "BOT TRADE flexible" are at the bottom but the most productive of all.

The hourly profit percentages are not stable. They can change without notification.

**Bot 3** comes with **35%** interest compounding and a minimum deposit of \$100,000.00 on **0.15%** profit per hour according to the current ROI. **Bot trade flexible** comes with **50%** interest compounding and a minimum deposit of \$250,000.00 on **0.16%** profit per hour according to the current ROI. These compounding rates increases the total profit at the end of 30 days

**BTU** profits will also be added up as extra bonuses at the end of the 30days.

### Mathematical analysis:

**BOT 1;** with minimum deposit of \$10,000.00 and maximum of \$49,999.99, if the least minimum (\$10,000.00) is invested for 30days on 0.06% per hour (current % which can change without notice).

i.e:

$$\$10,000 \times 0.06\% \div 100 = \$6. \quad \$6 \times 24\text{hrs} = \$144. \quad \$144 \times 30\text{days} = \$4,320.00$$

User will be earning \$6 every hour, \$144 every day, and \$4,320 at the end of his trade. There will also be bonuses according to the market performance of his BTU.

**Note:** These percentages are influenced by market sentiments. User may earn more or less.

**BOT 2;** with minimum deposit of \$50,000.00 and maximum of \$99,999.99, if the least minimum (\$50,000.00) is invested for 30days on 0.09% per hour (current % which can change without notice).

i.e;

$$\$50,000 \times 0.09\% \div 100 = \$45. \quad \$45 \times 24\text{hrs} = \$1,080. \quad \$1,080 \times 30\text{days} = \$32,400.00$$

User will be earning \$45 every hour, \$1,080 every day, and \$32,400 at the end of his trade. There will also be bonuses according to the market performance of his BTU.

**Note:** These percentages are influenced by market sentiments. User may earn more or less.

### Bots are selected automatically according to the amount invested.

**BOT 3;** This is the second most interesting package in this blockstake.ai algorithm. It comes with 35% interest compounding (compounding % is also influenced by market sentiments & can also increase or decrease without notifying User). Just like BOT 1 and BOT 2, the calculation is similar but with little beneficial change. There's also no limit to what you can invest here.

If the least minimum (\$100,000.00) is traded on 0.15% profit per hour with 35% compounding,

i.e

$$\$100,000 \times 0.15\% \div 100 = \$150. \quad \$150 \times 35\% \div 100 = \$52.5. \quad \$150 - \$52.5 = \$97.5$$

Due to the 35% compounding rate, User should earn \$150 per hour but 35% of the hourly profit, which is \$52.5 will be added back to the total invested to make it \$100,052.50. the remaining \$97.50 will be dropped into the available balance as the hourly profit. The next hour profit will be a bit higher than the previous profit because of the change in Total invested.

i.e

$$\$100,052.50 \times 0.15\% \div 100 = \$150.07875 \approx \$150.10.$$

User will earn approximately \$150.10 but same 35% compounding rate will return to the total invested. The remaining \$97.57 will drop into the available balance. Same thing goes on until the end of the trade (30days).

If there's no compounding rate attached to this plan, user is supposed to earn a total of \$108,000 (more or less according to market sentiments). But the added compounding rate makes it even much higher.

Just like the other packages, BTU profits are also added up after trading as bonuses.

**BOT TRADE (flexible);** This is the best package in this blockstake algorithm. It comes with 50% interest compounding (compounding % is also influenced by market sentiments & can also increase or decrease without notifying User). Just like BOT 3, the calculation is similar but with little beneficial changes. There's also no limit to what you can invest here.

If the least minimum **(\$250,000.00)** is traded on **0.16%** profit per hour with **50%** compounding, i.e

$$\$250,000 \times 0.16\% \div 100 = \$400. \quad \$400 \times 50\% \div 100 = \$200.$$

Due to the 50% compounding rate, User should earn \$400 per hour, but 50% of the hourly profit, which is \$200 will be added back to the total invested to make it \$250,200.00. the remaining \$200 will be dropped into the available balance as the hourly profit. The next hour profit will be a bit higher than the previous profit because of the change in Total invested. Same thing goes on for 30 days just like it is explained on Bot 3. But there are greater gains here. After trading, 64.99% of all the tokens your trade generated for blockstake will be added back into your account as extra bonuse.

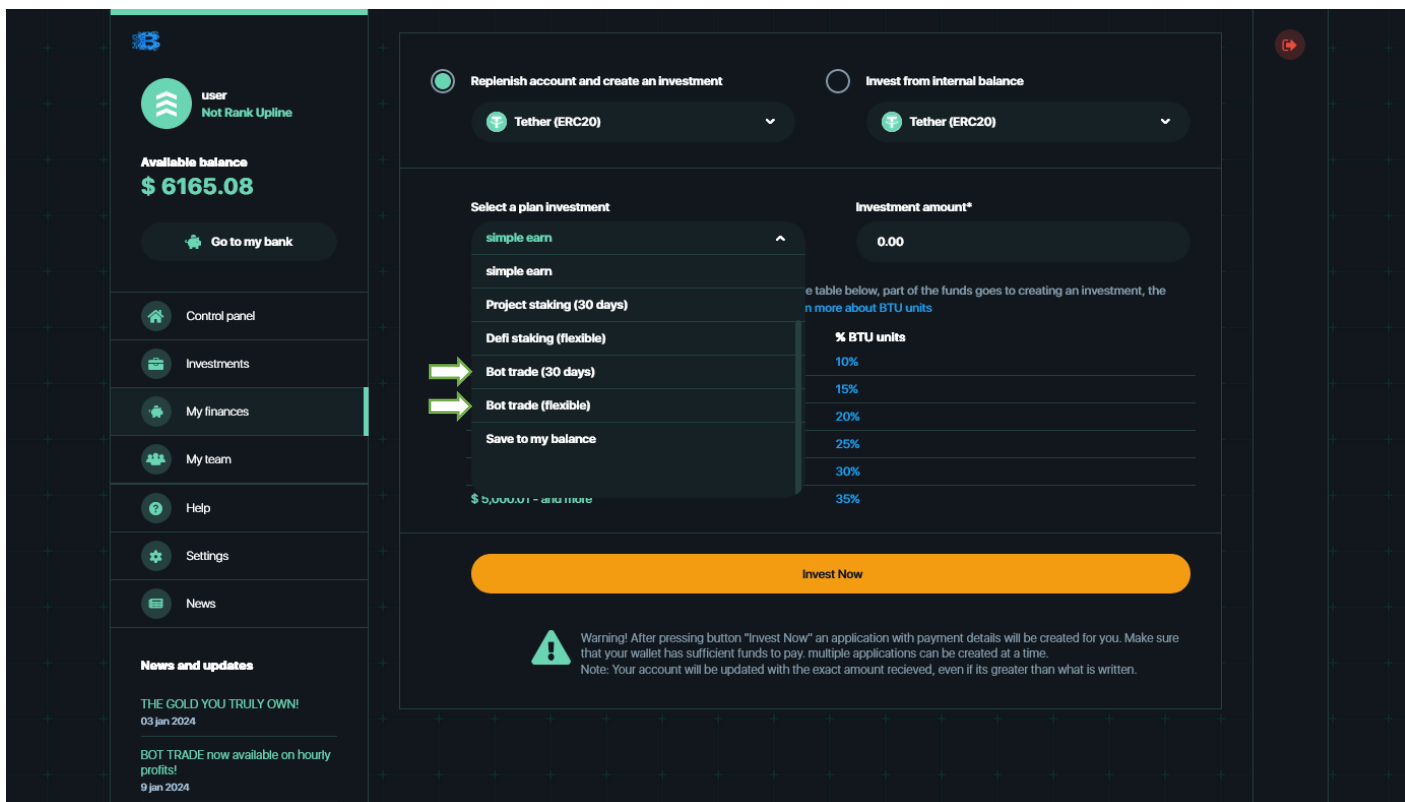
**NOTE: Principal return is guaranteed after trade.**

**There's a "release" button (red arrow), whenever user wants to end the trade before 30 days. But it comes with a cut from the total invested. Some amount will be deducted from the principal.**

As little as these % are, they are set like that to avoid high frequency trading (HFT) which may lead to principal loss. Currency trading is the biggest market in the world and in this market, volatility is equal to opportunity, which is also equal to profits or loses. Blockstake bots are created with outmost precision.

Blockchain.com inc, takes 100% responsibility of any loss of fund in any blockstake trading account at the course of trading. We also earn as your account earns. Therefore, with almost 0% attempt on HFT, we prefer a little but guaranteed profit to a huge but risky profit.





Clicking on “create a new investment” on the investment page will bring you to this deposit page.

On this deposit page, user should select BOT TRADE (30days) or BOT TRADE flexible as plan investment. Enter the amount you choose to invest, (BOT 1-3 will be determined by the deposited amount as listed on the investment page) then click on “INVEST NOW” which will take you to where you will find the specific wallet address of the coin you selected. There are 8 approved coins at the moment as listed under the “replenish account and create an investment”.

Once your deposit is confirmed in the given address, it will reflect on your blockstake “total invested” and your trade will start counting immediately.

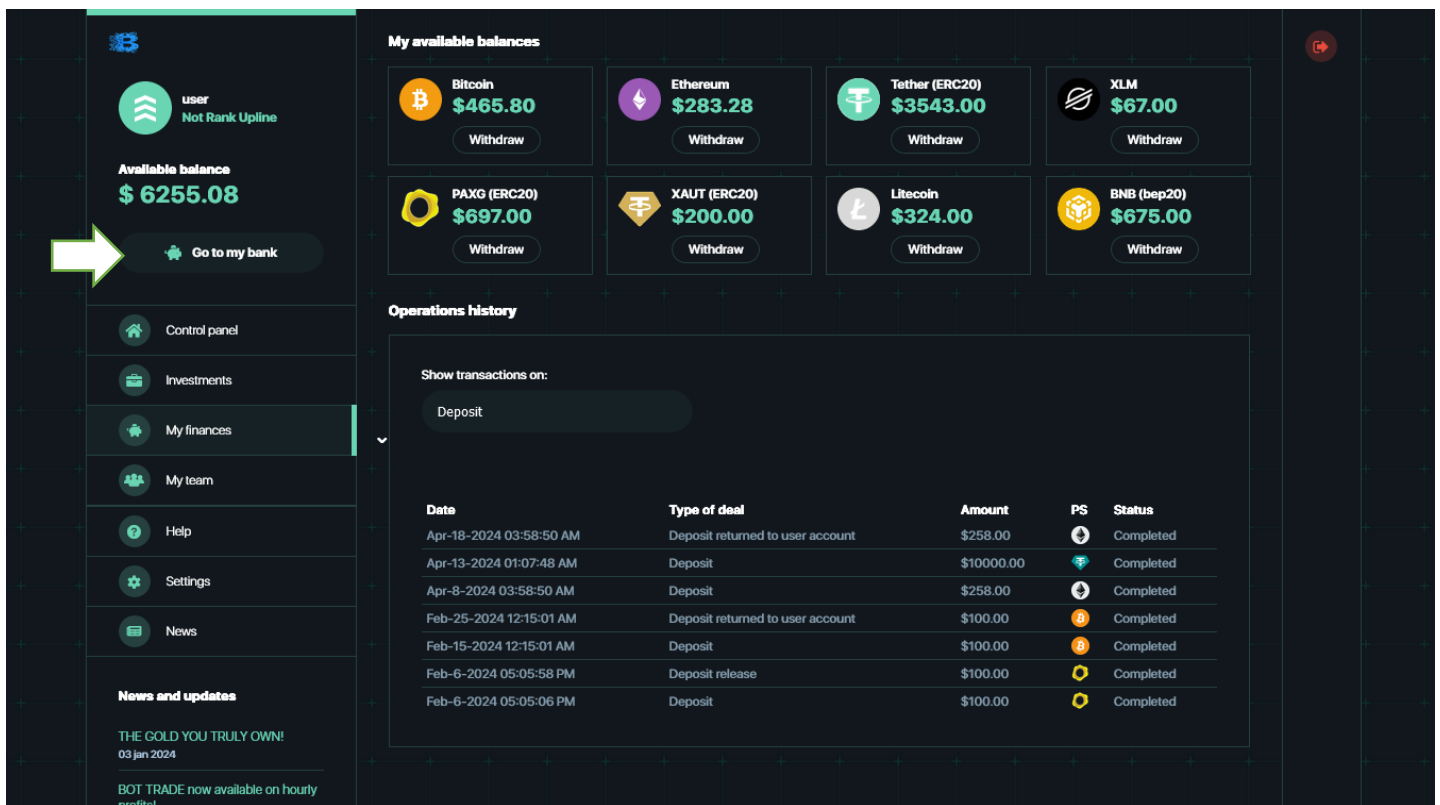
**Note; follow the instruction on the address page carefully. Click on “PROCESS” after copying the address to save it.**

After clicking on “PROCESS”, you will be returned to this page with a notification at the top of the page that your plan is saved, waiting for your deposit to be confirmed.

User can invest in multiple plans at the same time. You can reinvest from available balance by clicking on “INVEST FROM INTERNAL BALANCE” and selecting the specific coin you have funds in. The balance on each coin will be displayed beside it.

**THIRD-PARTY coin sellers may be introduced later, for investors without the blockchain exchange wallet or any other wallet that wants to buy directly into their trading account. For easy deposit with card or bank transfer.**

User will have to paste the copied address in the provided space by the Third-party API and carefully complete the KYC as demanded by the API.



### Withdrawal.

By clicking on **GO TO MY BANK**, it brings you to this page that will enable you to withdraw from your blockstake account into your private wallet.

The available amount on each coin will be displayed below it. Click on the withdraw button below to get to the proper withdraw page. Follow the instructions properly and the withdrawn amount will be successfully sent to your private account.

**NOTE:** Big transfers may require further identity verifications to make sure that the funds are truly withdrawn by the real account owner.

DO NOT SHARE YOUR LOGIN INFORMATIONS WITH ANY UNTRUSTED THIRD-PARTY. Blockchain.com inc, has a strong privacy policy. Your data remains intact no matter the circumstance.