Portfolio Composition Based on High-Ranking Hedge Funds Using Dynamic Genetic Algorithms

Gustavo da Silva Carita
Instituto Superior Técnico / IT
University of Lisbon
Lisbon, Portugal
joao.carita@tecnico.ulisboa.pt

Abstract— **The work presented proposes a system that optimizes a portfolio of stocks based on the hedge funds' assets, strategies and by tracking their past performances using Dynamic strategies in Genetic Algorithms. Most works that developed systems for the stock market use technical and fundamental indicators as features, but in this work, the hedge fund public data was used to model the behaviour of these all-star teams of financial experts. Being the stock market search space time-variant, the system developed is going to exploit a set of Dynamic strategies based on state-of-the-art approaches, such as *Temporal Seeding*, *Direct Memory*, *Immigrant* manipulation and a custom *Hyper-state*. The proposed architecture used information from 12th of February of 2014 until 12th of February of 2016. The proposed system outperformed the benchmark indexes for an entire quarter, obtaining an average *Return On Investment* of 5.7% and a *Sharpe Ratio* of 3.26.**

Keywords- Hedge Funds, Portfolio, Genetic Algorithm, Dynamic Systems, Financial Markets.

I. INTRODUCTION

The Stock Exchange Market is a mechanism that allows shares, bonds and other kinds of securities to be traded between private investors and institutions. The financial markets didn't remain untouched after the artificial intelligence rise that we faced in the most recent years, becoming a common tool for predicting outcomes, and started to be vastly used by institutions, such as banks and private funds.

The time series generated every second by the financial markets show that the systems to tackle in this paper's work are populated with noise and are non-stationary, thus being hard to study and predict. This noisy, non-stationary structure presented on stocks' prices is the reason why many investors and researchers use machine learning, to discover the underlying arrangement, present in the price signals from the stock market. To date, there are several studies that develop investment strategies by balancing technical indicators, such as relative strength index, moving averages, rate of change and many others. However, not much attention has been given to other financial behaviors and macroeconomic factors. Under this context, alternative investment strategies are less studied when compared to the regular ones which mainly use technical indicators.

One alternative, sometimes neglected, source of information about financial markets are the assets held by the biggest hedge funds in the world. Information on purchase or sale orders of shares by hedge fund managers allow to sometimes predict the evolution of future prices of that asset. When the shares are undervalued there is a tendency of increase in purchases, and when they are overvalued increased buyback. Having this into account, our main goal is to use hedge fund behaviors with techniques of intelligent computing to optimize investment strategies.

This work application establishes a path to predict profitable and risk-controlled portfolios of stocks based both on assets controlled by the biggest American Hedge Funds and the past prices of the different stocks. The application developed in the context of this thesis, must have the power to do predictions, adopting soft computing procedures. The problem at hand is a resource allocation problem with competing objectives such as risk and expected return, being the objective to find an investment strategy that can pick a weighted group of stocks from many available financial assets, to minimize the risk and maximize the expected return.

The proposed system is a combinatorial model based on Dynamic Genetic Algorithms (DGA) that applies dynamic strategies that invoke a hyper-state when certain conditions are met, while taking advantage of memory to preserve good solutions and temporal seeding for a better tune of our population of portfolios. In the end, the model should return a portfolio of stocks and a valorization rule that translates into the stock weight. The data used is based on fundamental indicators, created to describe a Hedge Fund portfolio using stock sectors and industries to balance the risk exposure.

This paper is divided into five main chapters. Chapter 1 Introduction, Chapter 2 Background and State-of-the-Art, focus on the theoretical foundations of financial markets and genetic algorithms. Ultimately, the state of the art of the problem under study, where is given an overview about other methodologies. Chapter 3 Proposed System Architecture, shows the adopted methodologies to successfully conclude this thesis' work, as well as the software's architecture and the computational improvements made to boost our application. Chapter 4 System Validation, introduces a solid validation procedure to assess the solution performance and Chapter 5 Conclusions, encapsulates the work developed and concludes about the strategy used and the overall performance. Finally, are presented the future work propositions.

II. Background and State-of-the-Art

This section provides an explanation about financial markets and the underlying mechanisms involved, genetic algorithms and dynamic optimization problems. Finally, is given the state of the art research, where will be explored the works on portfolio composition using genetic algorithms and dynamic optimization problems using evolutionary strategies.

A. Financial Markets

In a market economy, the allocation of economic resources is the outcome of many private decisions. Thus, prices become the signals operating in a market economy that direct economic resources towards their best use. This paper will focus on a segment of the factor market called financial market, which is a market where financial assets are traded. Nevertheless, the existence of a financial market is not a fundamental condition for the creation and exchange of a financial asset, however in most world economies this kind of asset is always created and traded in some type of financial market.

B. Hedge Funds

The term "hedge fund" (HF) defines a group of different investment vehicles that share similar characteristics. Hedge fund term includes any type of pooled investment vehicle that is privately organized, managed by professional investment managers and not widely accessible to the public, i.e., the average investor. The main financiers of this kind of investment vehicle are institutional investors and wealthy individuals. As the hedge funds became bigger and bigger, the U.S. Securities and Exchange Commission forced these investment vehicles, with over $100 million in marketable securities, to report all their holdings quarterly by submitting the form 13F [1]. This kind of funds usually use strategies that take more risk than usual when compared with mutual funds, although their profits usually outperform benchmark indexes such as Standard & Poor's 500 (S&P 500) stock index.

C. Portfolio Composition using Genetic Algorithms

Works developed on stock portfolio composition have showed encouraging results by using Genetic Algorithms (GA) to optimize both fundamental and technical analysis of the market. The works listed below use genetic algorithms for this process, but other implementations have been suggested using other evolutionary programming algorithms, like neural networks, swarm optimization, ant and bee colonies, and many other approaches.

One of the first works involving GAs and stock portfolio composition and evaluation was made by Wagman [2], where a portfolio selection was developed using technical analysis indicators as a strategy that generates buy and sell signals for different stocks. In this approach a new genetic operator to the Standard GA was added called shuffle, this genetic operator will select two random stocks from an individual and randomly modifies their percentage shares, being shuffle a variation of the mutation process for a specific propose. The results were promising and the returns were 3% higher that the interest rate of that year. Before, Tomassini et al. (1995) developed a portfolio selection distributed genetic algorithm [3]. In their work, they implemented a genetic algorithm with multiple equal populations, called islands, that will exchange groups of individuals in a synchronous manner. The authors state that the returns of the portfolios generated by the genetic algorithm are satisfactory but no precise values were given.

On portfolio composition for index tracking, Kyong et al. (2005) focus their work on using a genetic algorithm to create a portfolio for index fund management. Index fund portfolios aim to match the performance of a chosen benchmark index like S&P 500 or Financial Times Stock Exchange 100 (FTSE 100) [4]. In this strategy, the selection of stocks used fundamental variables like standard error of the portfolio beta, average trading amount and market capitalization. The authors show that the GA optimized strategy has better results when compared to the standard version. Aranha & Iba (2007) went further and developed a portfolio selection system using a GA that considers the modelling cost and uses a seeding and multi-objective strategy [5], which are considered Dynamic Approaches. The returns obtained were 3.12% compared with the 1.7% obtained by the NASDAQ index in the same period. The returns on the NIKKEI for the seeding GA were like the benchmark index, thus concluding that the seeding genetic algorithm has difficulty when leading with these extreme movements. Gorgulho et al. (2009) proposed a GA that will optimize an investment strategy based on technical indicators by optimizing the weights of each one of them in the decision for an efficient portfolio composition [6]. The GA proposed follows the standard GA with a new feature that will mimic an artificial immune system with the objective of handling the infeasible individuals on the population that might be generated. This immune system will store the infeasible individuals inside a separated population and they will suffer the same effects of evolution as the main population. The goal here is to store relevant information that might be contained inside the infeasible individuals. This strategy was compared with other strategies like Buy & Hold (B&H) and "Random Walk", showing far better results, with an average return of 3.46% compared with 0.49% from B&H and 0.9% from the random approach. Gorgulho et al. (2011) proposed a system that develop a technical analysis strategy for stock picking and portfolio composition using a GA optimization [7]. The strategy created was compared with other approaches such as B&H and "Random Walk", as well as the benchmark index. The GA system applied used a selection system that first choose the best individuals for selection and then applied a Roulette-Wheel procedure to select the breeders. The crossover system used was a standard one-point crossover method. The mutation method applied would randomly generate a value for each gene selected inside the chromosome, although Gorgulho used an elitist approach by keeping the best individual of the population away from the mutation process. Finally, the fitness function used was the Return on Investment, being the goal of the GA to maximize this function. The results were positive, being the "Random Walk" and B&H outperformed by the GA strategy using technical indicators.

In other hand we have approaches who try to optimize a multi-objective portfolio composition, taking into account not only *Return on Investment* (ROI) but risk. Simões et al. (2010) proposed a system that using Gas that will formulate a technical indicators strategy, then compares the results with a B&H, Moving Averages Crossover and Moving Average Derivative strategies [8]. In this work, a standard GA was applied, the crossover operator used was a one-point crossover and the selection procedure was a roulette-wheel selection. The fitness function used was the Sharpe ratio, being this approach multi-objective, i.e., minimizing the risk and maximizing the return. The strategy applied by Simões was a combination of Simple Moving Averages Crossover and Moving Averages Derivative, being this strategy optimized by the GA. The results were promising, presenting an average return on investment of 9.0%, thus beating all other strategies.

Finally, some approaches use both technical and fundamental indicators. Silva et al. (2015) approach leverages both indicators and a group of real world constraints to make the strategy closer to the real financial world [9]. The implementation proposed by Silva used a multi-objective goal, i.e., minimizing the risk and maximizing the return. Three chromosomes were developed to simulate different investment models with different real-world constraints like transaction costs, long only investments and quantity constraint for each asset inside a portfolio. This approach shown promising results and with solutions that outperformed the benchmark index S&P 500. Michalak et al. (2012) strategy uses a Multi-Objective optimization to find portfolios that outperform investment funds while showing an identical behavior [10]. The crossover used was a standard single-point crossover. The mutation procedure selects a random individual and then adds a random value with uniform probability to a gene inside that chromosome. The select procedure is based on the tournament-selection approach with slight changes from the NSGA-II algorithm proposed by Deb et al. (2000) [11]. Another two genetic operators were added, called Division Mutation and Local Search operator. This approach composes portfolios of stocks and currencies. The strategy proposed is Multi-Objective, using as objectives the investment return, value-at-risk (measure of risk) and mean squared error (measure of difference between generated portfolios and investment funds actual portfolios). The proposed algorithm could find portfolios similar to investment funds' portfolios but with higher returns and lesser risk.

D. Dynamic Optimization Problems using Genetic Algorithms

Dynamic Optimization Problems have become an important research field in recent years and many works have been developed to optimize the already existing solutions. Many approaches have been proposed based on abstractions of mechanisms used by actual living beings. One typical approach for DOP using EP makes use of memory systems to track the moving optimum inside a specific search space. Memory approaches are divided into two sub-groups, implicit and explicit. Implicit approaches include examples as Multiploidy and Dominance and Dualism Mechanisms. Inspired by diploid and multiploid organisms, Multiploidy and Dominance schemes were proposed to improve the performance of solving systems for DOPs, at first was proposed a system that used a triallelic dominance scheme to extended a simple haploid GA into a diploid GA by Goldberg & Smith (1987) [12]. After, was proposed by Ng & Wong (1995) [13] a diploid scheme with four possible combinations per gene for GA implementations. Another approach proposed three years later added a dominance scheme with additive diploid mechanism by Lewis et al. (1998)

TABLE 1. STATE-OF-THE-ART ON PORTFOLIO COMPOSITION USING GA

WORK	DATE	HEURISTIC	STRATEGY	FITNESS FUNCTION	BENCHMARKS	FINANCIAL APPLICATION	ASSETS/PERIOD	RETURNS
[15]	2003	GA	Standard Genetic Algorithm w/ shuffle operator.	6 technical analysis indicators combined, being the risk and return the fitness parameters.	Interest rate of 4.5%	Portfolio Composition Stock Market	DJIA: 1979-1980	7.5%
[16]	1995	MODistGA	Multi-population w/ synchronous immigration; Multi-objective	Lambda Trade-off Function	-	Portfolio Composition	-	-
[17]	2005	GA	Standard Genetic Algorithm	Standard Error of Portfolio Beta; Average Trading Amount; Market Capitalization	KOSPI 200 Conventional Algorithm	Portfolio Composition for Index Funds	KOSPI 200: 1999-2001	-
[18]	2007	MOGA	Multi-Objective; Seeding	Euclidian Distance; Sharpe Ratio	NASDAQ NIKKEI Standard GA	Portfolio Composition Stock Market	NIKKEI: 1998-2006 NASDAQ: 2000-2006	NIKKEI: 1.49% NASDAQ: 3.14%
[19]	2009	GA	Artificial Immune System	Return on Investment	B&H Random Walk	Portfolio Composition Stock Market	S&P 500: 2005-2007	Avg. 3.46%
[20]	2010	MOGA	Multi-Objective	Sharpe Ratio	B&H Random Walk	Portfolio Composition Stock Market	S&P 500, FTSE 100, DAX 30, NIKKEI 225: 1993-2009	Avg. 9.0%
[21]	2011	GA	Artificial Immune System	Return on Investment	B&H Random Walk	Portfolio Composition Stock Market	DJIA: 2003-2009	Best. 62.95%
[22]	2015	MOGA	Strength Pareto Evolutionary Algorithm; Multi-Objective	Return on Investment; Variance of returns in a portfolio	S&P 500	Portfolio Composition Stock Market	S&P 500: 2010-2014	Best. 50.24%
[23]	2012	MOGA	Non-dominated Sorting Genetic Algorithm; Multi-Objective; Division Mutation; Local Search	Return on Investment; Value-at-Risk; Mean Square Error	18 Polish Investment Funds	Portfolio Composition for Investment Funds	Warsaw Stock Exchange & FOREX (CHF, CZK, EUR, GBP, HUF, USD)	Positive

[14]. In figure 15 can be seen an example of a dominance and diploid scheme, the dominance schemes presented come from the works of Lewis and Ng & Wong. Another implicit memory approach is dualism, where every individual has two chromosomes (diploid), being the chromosomes the inverse of each other. Several approaches have been developed, for example, the primal-dualism genetic algorithm [15] and the dual population-based incremental learning [16]. Explicit memory approaches include direct and associative memory. The direct memory approach was proposed by Branke (1999) [17]. The associative approach named population-based incremental learning using an associative memory scheme was developed by Yang & Yao (2008) [18].

Diversity approaches for DOPs using EP includes approaches like random immigrants, memory-based immigrants [31] and finally hybrid immigrants approach [19]. Multi-Population approaches for DOPs include two main approaches, self-organizing scouts [20] and shifting balance [21].

A distinct strategy proposed is adaptive, and examples of this mechanism's implementations are hyper-mutation, hyper-selection and hyper-learning. In hyper-mutation, the mutation rate will be increased to galvanize the population diversity [22]. In hyper-selection, the selection pressure will be increased, so the population convergence will increase to specific optimum [23]. Finally, the adaptive hyper-learning strategy will raise the learning capability for the population-based incremental learning [24]. There are other implementations available to this approach, as well as combinations of multiple implementations.

Ultimately, there are the predictive approaches for DOPs solving. One of the first works developed was a Kalman-Extended GA [25] which uses a Kalman filter over the fitness values of each individual of the population. These Kalman filter will discover when is needed to create an individual or when is needed to re-classify an individual. Next, there is the Van Hemert et al. (2001) [26] strategy, which used a learning module to estimate the future state of the environment. This is implemented by using two populations, one that performs a search at the current environment and another that, by using the best individuals from the past, will predict the future optimum. Other approach by Bosman [27] combined EP with machine learning algorithms and statistical techniques to predict the future optimums for a specific environment. At last, there are the works of Simões & Costa [28] that made use of linear regression strategies and Markov chains to predict changes in environments.

III. Proposed System Architecture

The proposed system has the objective of generating a possible set of solutions based on hedge fund information and past stock prices. The goal here is to build an infrastructure capable of digesting both types of data into a working prediction model for the stock market portfolio management by using a DGA with a custom backbone infrastructure.

A. Architectural Overview

To achieve a better level of implementation a layer abstraction was deployed. In more general terms, the end-to-end description of this system starts in the *Mining Layer*, where different internet sources are compressed into useable data and promptly stored at the designated file system inside the *Database Layer*. Some data about the hedge fund's performance needs to be calculated, being the *Simulation Layer* the responsible for this computation, returning all the data back

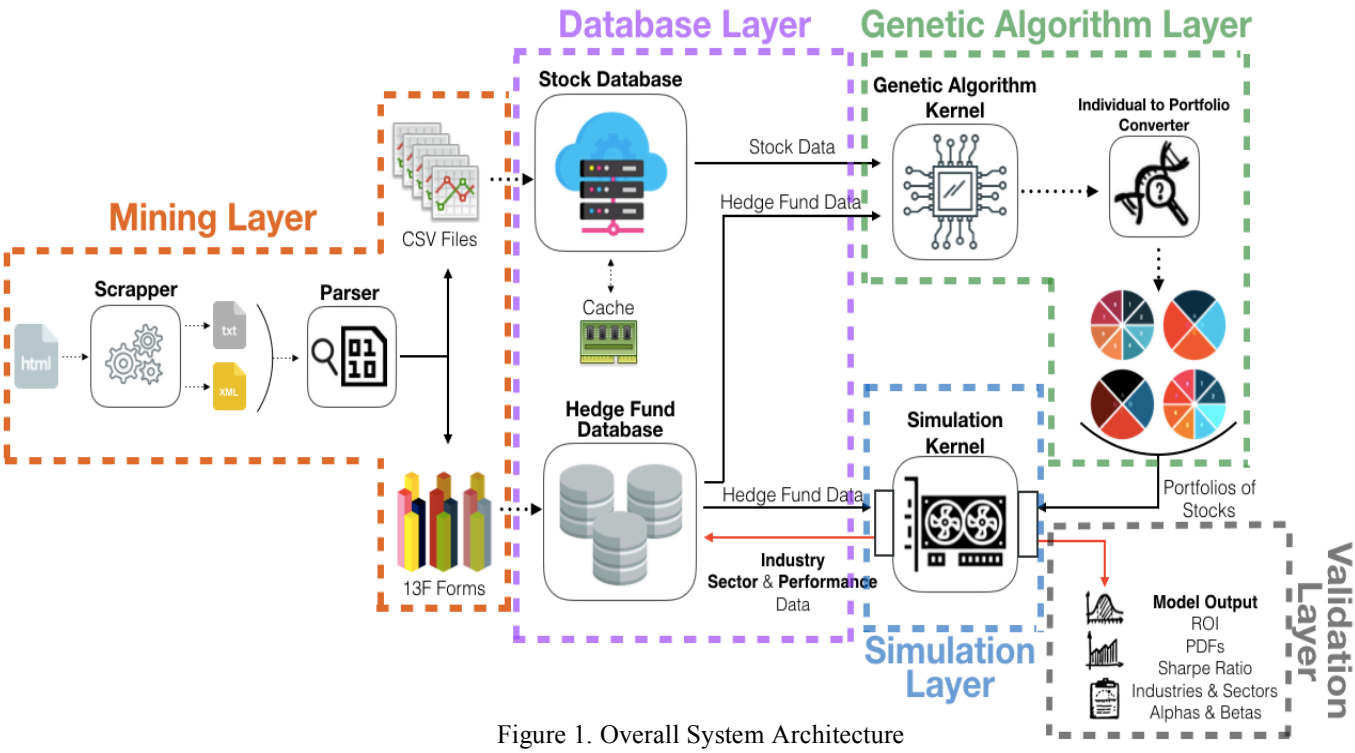

Figure 1. Overall System Architecture

to the *Database Layer* for storage. After achieving all the pre-processing necessary, our data is sent accordingly for the desired optimization into the *Genetic Algorithm Layer.* At this stage, the genetic kernel will receive both stock and hedge fund data to start the optimization process, where a population of portfolios of stocks is optimized, based by a set of rules, in an incremental manner for several pre-defined iterations until the best performing portion is proposed as solution for the problem at hand. The proposed solutions are tested, once again at the *Simulation Layer*, and all the outputted results sent into the *Validation Layer*, where the solutions are evaluated and the overall system performance is addressed by a set of benchmarks. Figure 1 shows the overall architecture plan with well-defined layers.

B. Mining Layer

The Hedge Fund data is composed by the 13F formularies of each hedge fund considered. Each year has four 13F formularies and each one of these formularies represents a specific quarter of a specific hedge fund, in sum each 13F formulary represents a hedge fund portfolio of stocks. The information gathered starts at the fourth quarter of 1998 and ends at the first quarter of 2016. For the Hedge Fund data acquisition, this system uses scripting to access the Edgar Platform [29] made available by the *U.S. Securities and Exchange Commission*. Each 13F is composed by a set of stocks with the title of class, Committee on Uniform Securities Identification Procedures number. Value in US Dollars and number of common stock shares owned.

The stock data is composed by a set of files, each one of them represents a specific stock price daily since December of 1998 until May of 2016. All the data acquired was retrieved from the Yahoo Finance Platform by using a custom library made for Python called Yahoo-finance. The stock data format is quite simple in comparison, showing each daily price, open, high and adjusted close with the corresponding value in *US Dollars.*

C. Database Layer

The *Database Layer* is divided into two separate categories, the *Stock Database* and the *Hedge Fund* Database. Since some extra calculations are needed for the finding of the desired solutions, a set of preprocessing activities need to happen.

1) Stock Database

This database is responsible for handling all the stock price data. Considering that our system uses 4000 stocks, during our computation at the Optimization Layer, the GA sometimes needs other information related to the stock prices, such as a stock return on investment and standard deviation for each day for a given interval of time, and Stock's overall return on investment for a given interval of time.

2) Hedge Fund Database

This database is responsible for handling all the Hedge Fund data that is relevant for our study. Taking into consideration that our system uses information about 29 different Hedge Funds with multiple 13F formularies each one of them. During the optimization process that happens at the GA Kernel, other relevant information about hedge fund's assets is required. The following information is preprocessed, stock popularity among hedge funds, stock sector popularity among hedge funds, stock industry popularity among hedge funds, hedge fund popularity associated to a specific stock, stock exposure variation associated to a specific stock and two consecutive 13F formularies.

D. Genetic Algorithm Layer

The *Genetic Algorithm Layer* is responsible for the generation of a set of possible solutions based on hedge fund information and past stock prices that are returned by the *Database Layer*. For the Genetic Algorithm Kernel implementation, a Python library called *DEAP [30]* was used, this library enables concurrency, enhances our algorithm robustness and allows us to use out-of-box implementations of different GA strategies. Before explaining all the constituent parts of the developed Genetic Algorithm, it is important to list all the strategies implemented in this work.

a) Chromosome Structure

The chromosome can be considered the basic unit of our GA, because it represents a possible solution inside our population and the function parameters that we want to optimize. Taking this into consideration, becomes empirical that our chromosome structure needs to be well-defined, to suitably describe our optimization problem. Each individual has its own chromosome, and each chromosome is represented by an array of floats (genes), being each float between 0 and 1. When the Genetic Algorithm starts its optimization process, each individual is initialized by giving random values to each gene. The chromosome representation can be seen in the figure 2.

Figure 2. Chromosome Representation

The chromosome structure is inspired in a regular portfolio, thus having a set of stocks and for each stock a specific weight. The above chromosome follows this paradigm and is divided into two sub-sets, the first set represents which stocks to pick for our portfolio (red box in figure 2) and another is responsible for the *Valuation Rule* (green boxes in figure 2) that defines the weight of each individual stock based on hedge fund information. In the end, both problems are optimized by a defined evaluation function and a set of stocks and the corresponding weights of each one of them is returned.

Before explaining how the *Valuation Rule* maps the weights for each stock, it is necessary to understand each one of the five members that compose this rule. The *Valuation Rule* uses as input five different parameters, such as, *Stock Popularity* (StP), *Sector Popularity* (SeP), *Industry Popularity* (IP), *Hedge Fund Popularity* (HFP) and *Stock Exposure*

Variation (SEV). The *Valuation Rule* parameters are explained bellow:

- **Stock Popularity** – Average weight of a specific stock in all the HF's portfolios where exists.

$$\mathrm{StP} = \frac{1}{N}\sum_{i=0}^{N} HF_{i_{weight}}(stock) \quad (1)$$

- **Sector Popularity** – Average weight of a specific stock sector in all the HF's portfolios.

$$\mathrm{SeP} = \frac{1}{N}\sum_{i=0}^{N} HF_{i_{sector}}(stock) \quad (2)$$

- **Industry Popularity** – Average weight of a specific stock industry in all the HF's portfolios.

$$\mathrm{IP} = \frac{1}{N}\sum_{i=0}^{N} HF_{i_{industry}}(stock) \quad (3)$$

- **Hedge Fund Popularity -** Each HF has a Popularity parameter between 0 and 1 and its related to each HF performance. The best HF has a value of 1 and the worst a value of 0, all other HFs use values inside this interval according to their performance rank. The HF Popularity parameter is the average value of all HF popularities for the HFs where the stock exists.

$$\mathrm{HFP} = \frac{1}{N}\sum_{i=0}^{N} HF_{i_{performance}} \quad (4)$$

- **Stock Exposure Variation -** Average variation of a specific stock weight between two consecutive quarters for the HF portfolios where that stock exists.

$$SEV = \frac{1}{N}\sum_{i=0}^{N} \Delta HF_{i_{weight}}(stock, Quarter_t, Quarter_{t-1}) \quad (5)$$

The chromosome structure implemented tries to find a specific set of stocks and weights for the *Valuation Rule* that maximizes a fitness function. In this context, the chromosome presented is the genotype of the individual and the final portfolio with every weight defined is the phenotype for this individual.

The above statement raises the following question: *"How do we get a phenotype from a genotype like our chromosome?"*. Firstly, the stocks picked are quite easy to convert, each one of them is a value from 0 to 1 that is mapped to an integer from 0 to 4000, each integer represents a stock ID that is transformed into a ticker. For example, a *Stock gene* has a value of 0.1 represents the ID 432, then converted to the corresponding ticker, for instance, *GOOGL*. The final gene, called *Number of Stocks*, limits the maximum number of stocks in our portfolio and can vary from 0 to 1, then mapped into an interval from 8 to 12. So, in summary, the stock genes first need to be converted to their corresponding ticker, and the number of stocks is defined by the last gene, called *Number of Stocks*.
Secondly, the stocks weights must be defined, and each weight is based in the *Valuation Rule* optimized during our GA. The Valuation Rule can be defined by:

$$\mathrm{VR(stock)} = \mathrm{w_{StP}}\times\mathrm{StP(stock)} + \mathrm{w_{SeP}}\times\mathrm{SeP(stock)} + \mathrm{w_{IP}}\times \mathrm{IP(stock)} + \mathrm{w_{HFP}}\times\mathrm{HFP(stock)} + \mathrm{w_{SEV}}\times\mathrm{SEV(stock)} \quad (6)$$

In equation 6, $\mathrm{w_{StP}}$, $\mathrm{w_{SeP}}$, $\mathrm{w_{IP}}$, $\mathrm{w_{HFP}}$ and $\mathrm{w_{SEV}}$ are defined from 0 to 1, all these parameters are optimized during our GA and come directly from our possible solution chromosome. The functions StP, SeP, IP, HFP and SEV receive a specific stock and return a value from 0 to 1 based on the Hedge Fund data. These functions are calculated during the pre-processing and are stored inside the *Hedge Fund Database*. So, the *Valuation Rule* will return a value from 0 to 1 for each stock considered, then all the *Valuation Rule* values obtained are normalized and finally each stock weight is derived. The stocks weights calculated must respect the following conditions:

$$\begin{cases} 0 < w_i < 1, \ \ i \in 0,1,\dots,N \\ \sum_0^N w_i = 1 \end{cases} \quad (7)$$

Where N is the number of stocks considered, and is defined by the last gene of the chromosome. After the two conversion processes above described, a phenotype solution emerges from a different genotype representation in the form of a regular portfolio that is composed by a set of stocks $\{stock_0, \dots, \ stock_N\}$ and a set of stock weights $\{w_0, \dots, \ w_N\}$, both ready to be studied in matters of performance.

b) Fitness Function

The fitness function implemented is called *Sharpe Ratio* and is defined by the following equation:

$$\mathrm{S_r} = \frac{\overline{r_p} - r_f}{\sigma_p} \quad (6)$$

Where, $\overline{r_p}$ is the mean portfolio return, r_f is the return of a risk-free investment and σ_p is the standard-deviation of between all stocks contained in the portfolio. The risk-free investment considered for this work were the U.S.A. treasury bonds yield, that is widely considered a safe asset with a return on investment value of 0.01. The *Sharpe Ratio* evaluates both risk and return, being the main goal to maximize the return on investment and minimize the risk associated. This ratio is used to specify how much excess of return you are receiving for the higher volatility endured for holding riskier stocks. The *Sharpe Ratio* works as a multi-objective fitness function, where the return on investment must be maximized and the risk minimized, thus generating portfolios with positive returns and stocks with low volatility, i.e., semi-steady growth. The *Sharpe Ratio* can return any real number, but values higher than one are usually considered good, a value between 0 and 1 is considered satisfactory and all the negative values are considered unsatisfactory, because it means the portfolio is losing money.

c) Genetic Algorithm Operators

As explained before, every Genetic Algorithm implementation needs to support three different operators, such as *Selection, Crossover* and *Mutation*. The three operators used are explained bellow.

- **Selection Operator -** Two distinct GA selection operators were used, the Tournament Selection and the Elitist Selection. The Tournament Selection will randomly group every individual into groups of 3, inside each sub-group the individuals will compete against each other in terms of fitness and the fittest of each sub-set survives, thus passing its traits to the next generation. The Elitist Selection will select a defined portion of the fittest individuals from a population and always passes them to the next generation.
- **Crossover Operator -** The crossover method used was the Single-point crossover. This method selects a point at a random location at both parents, then each parent contributes with one part of the chromosome to generate an offspring.

- **Mutation Operator -** The population of individuals has a chance to be mutated in every generation of our GA, if an individual is chosen for mutation a sub-set of genes will be randomly picked using a uniform distribution since every gene has equal probability of being mutated. If one of those picked genes is a stock gene, then a uniform distribution is used to pick a value from 0 to 1, if the picked gene is a *Valuation Rule* gene then a Gaussian distribution is called with the gene own value as mean and standard deviation of 0.2.

d) Dynamic Enhancement

Being the task at hand composed by a DOP, to enhance our GA performance some Dynamic Genetic Algorithm's strategies were used to explore memory, diversity and adaption in our GA. In this work five different strategies were applied and are the following:

- **Random and Elitist Immigrants -** This proactive method enhances the diversity of our population every generation by doing a global search of the search space with the generation of random individuals. The difference is that Elitist Immigrants are individuals based on the best solutions of the past generation plus mutation, thus using some past information.
- **Hyper-Mutation -** This dynamic procedure exploits the mutation probability by increasing the rate of mutation for a period to introduce more diversity in the overall population. This strategy enhances our GA adaptability by improving the population diversity.
- **Hyper-Selection -** This method exploits the tournament size of the Tournament Selection to create a higher selection pressure. This method is activated after the Hyper-Mutation state for a single generation, because by increasing our selection pressure we want a rise in the overall quality of the potential solutions.
- **Direct Memory -** The Direct Memory approach uses an auxiliary memory to store individuals based on their fitness for later use. In this strategy, the best individuals of each generation are selected and if some individual, already inside the memory, has a lower fitness, then its replaced by a new one from the present population. The best individuals among all the generations are injected in the next temporal iteration of the Temporal Seeding.

e) Temporal Seeding

The Temporal Seeding approach implemented for this thesis is based on the work developed by Aranha & Iba [5]. This technique splits the time-series dataset into even parts, then each part is optimized by a different GA that uses the best individuals, from the previous time-series chunk, as immigrants into the new initial population. This procedure is repeated for every time-series spilt made, in order. In our problem context, a year dataset is divided into four different quarters, each quarter is optimized by a different GA that uses the best solutions of the past quarter (*Seeding*) on the initial population.

f) Overall Kernel Architecture

Now that all the parts of the *Genetic Algorithm Layer* were explained, the overall architecture of his GA Kernel must be explained. The structure of the GA Kernel can be seen in figure 3 and receives a year as input, this year is then divided into 4 equal parts, each one of them represents a specific quarter. Each quarter is optimized by a GA that receives the *Seeding Immigrants* of the previous quarter as input and outputs the top 10 individuals to be used as input in the next quarter. The first quarter starts with no *Seeding Immigrants*, being the population fully random.

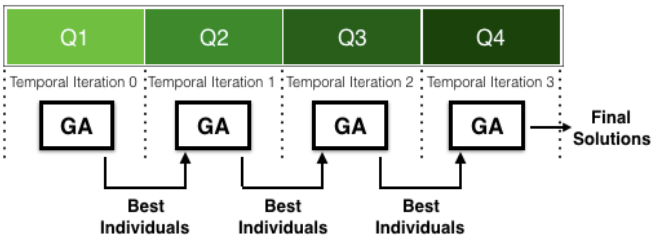

Figure 3. GA Kernel Architecture

E. Simulation and Validation Layer Implementation

The Simulation Layer is responsible for the Hedge Fund Simulator and the Model Portfolio's Simulator. Both simulators were implemented as standard Python Class and for all calculations needed the Numpy library was used [31]. The Validation Layer was implemented as a standard Python Class, using Numpy library [31] for calculations and Matplotlib library [54] for all the plotting needed for the analysis of the model performance.

IV. System Validation

For a valid system validation, a case study was created to address the performance of the proposed architecture.

A. Case Study - Description

To successfully address the quality of our approaches, they must be evaluated by mutual evaluation metrics. The performance metrics used in this system, to compare the quality of distinct approaches, were the Sharpe Ratio and Return on Investment.

a) Evaluation Metrics

The Sharpe Ratio is a performance measure used to address a portfolio profitability and associated risk. In simple terms, the Sharpe Ratio measures a portfolio's risk-adjusted returns, i.e. for each unit of risk taken how much return was accomplished. The Sharpe Ratio is defined by the equation 6. The Return On Investment (ROI) measures the efficiency of an asset by evaluating his return relative to the asset acquisition cost. The ROI expression is defined by the following equation:

$$ROI = \frac{r_{asset} - c_{asset}}{c_{asset}} \quad (7)$$

b) Benchmarks

The proposed system results will be compared with two benchmark indexes. The first index will be the S&P 500, a benchmark that tracks the 500 biggest companies in the United States of America.

The second benchmark index is going to be the overall return of all hedge funds' portfolios combined. The Whale Index follows all the S&P 500 stocks owned by leading hedge fund managers and compares it with S&P 500 Index. Citing the Whale Index creators: *"The Whale Index tracks the highest-conviction stocks held by leading hedge fund managers. It is a long-only index that tracks the 100 most commonly held stocks disclosed on the quarterly 13F regulatory filings of consistently successful managers. The index is equal-weighted and rebalanced on a quarterly basis 46 days after the end of each quarter."* [32]. Thus, becomes obvious that Whale Index would be a great benchmark for a model in the work context, but since it's not publicly available an approximation for this index must be found. So, the main goal of the Whale Index estimation is to create an approximate index that effectively reflects the overall Hedge Fund performance. Since the Whale Index is not publicly available an approximated version was needed. The original Whale Index is calculated using only the 100 most common stocks and using different HFs each quarter. Our proposed approximation is the average ROI for all the 29 HFs considered.

TABLE 2. DGA TRAIN/TEST CONFIGURATION

EXPERIMENT	TRAINING PERIOD	TEST PERIOD
I	12/02/2014-12/02/2015	12/02/2014-12/02/2015
II	12/02/2014-12/02/2015	12/05/2015-12/08/2015
III	12/08/2014-12/08/2015	12/08/2015–12/11/2015
IV	12/11/2014–12/11/2015	12/11/2015–12/02/2016

c) Model Validation Setup

The following experiment is structured into four different pairs of train and test periods; the train periods are 12 months while the test periods are only 3 months. The train and test pairs are generated by a sliding window. The four experiments follow different pairs of train and test created according to the figure 35 and each experiment configuration can be seen in the following table 2.

TABLE 3. DGA CONFIGURATION

PARAMETERS	METHOD	HYPERPARAMETERS
SELECTION	Tournament	**Tournament size:** 3
MUTATION	Gaussian + Uniform	**Mutation Rate:** 20% **Gaussian:** • Mean: Gene Value; • Std. Deviation: 0.2
CROSSOVER	Single Point	**Cut-Point:** Randomly selected. **Crossover Probability:** 40%
DIRECT MEMORY	-	**Top 10** individuals.
ELITISM IMMIGRANTS	-	**Top 15%** individuals.
RANDOM IMMIGRANTS	-	**35%** of the next generation individuals.
HYPER-MUTATION	Gaussian + Uniform	**Mutation Rate:** 40% **Gaussian:** • Mean: Gene Value; • Std. Deviation: 0.2
HYPER-SELECTION	Tournament	**Tournament size:** 6
TEMPORAL SEEDING	-	**Number of Temporal Splits:** 4

B. Case Study – Parameter Setup

Each experiment is composed by a training set with 4000 stocks available, doing 100 executions which returned the top 10 portfolios. Each execution is composed by a population of 100 possible solutions, optimized during 10 generations. After all the executions, the 1000 possible portfolios will be analyzed in terms of ROI and *Sharpe Ratio* to validate the proposed system performance. The DGA used in our Kernel for the four experiments follows the configurations present in the table 3.

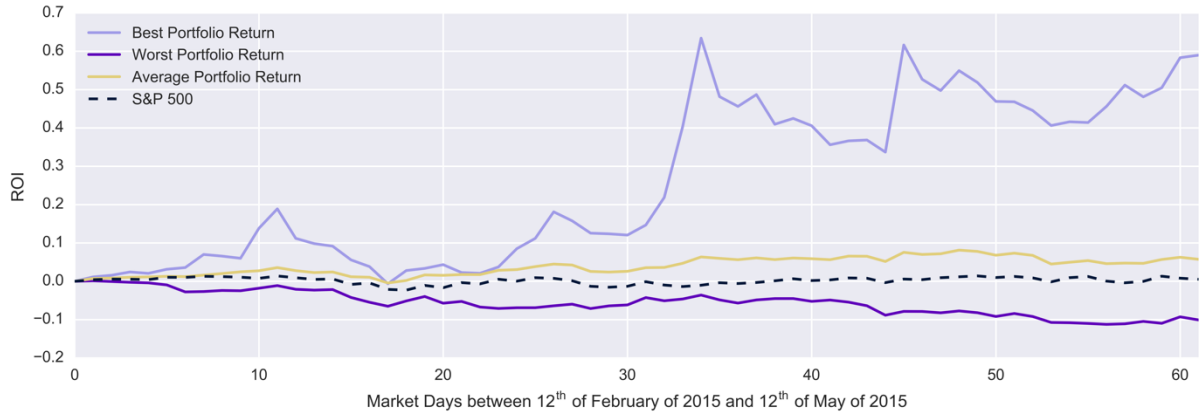

Figure 4. ROI progression fpr S&P500 and Average, Best and Worst portfolios for Experiment I

C. Case Study –Performance Analysis

The system performance was tested during the years of 2015 and 2016. The tests of performance made to the system were in matter of ROI and *Sharpe Ratio* and all the tests were compared to our benchmarks, the S&P 500 index and the *Whale Index Approximation* (WIA). The table 4 summarizes the results obtained across the experiments and the S&P 500 and WIA values for the given test periods. The year selected shows a negative trend, having a negative ROI for both benchmarks for the year tested. The year period defined was divided into four different quarters. During the four experiments, only one quarter had a positive ROI, against the other 3 quarters with an overall negative ROI for our proposed system. The experiment II, III and IV all had negative returns, with -4.9%, -5.9% and -14.5% respectively, and the experiment I presented a good overall return of 5.7%. Regarding the year tested, the proposed system had an overall ROI of -19.6% against -10.9% from the SP500 and against -18.67% from the WIA. In terms of yearly ROI against the SP500 and WIA, the model was surpassed by both benchmarks, but the period is considered adverse, with at least 3 negative quarters in matters of SP500. In the other hand, the proposed system tries to beat the two benchmark indexes by using, at maximum, 12 stocks, against 500 stocks for the SP500 portfolio and 4000 stocks for the WIA portfolio, thus being both much less volatile in comparison and exposed to much less risk by taking into consideration that the assets are much sparser. In the other hand, the proposed system only uses long positions, being exposed to a down trended market without any short activities to manage the overall market fall. Nevertheless, during a positive quarter for the American market, represented by the experiment I, the proposed system had a positive ROI, beating both benchmarks and shows that the proposed solution is well behaved during bullish markets (uptrend), see figure 4. During bearish markets, and since the proposed system has no short selling mechanisms, the overall performance is worse than both indexes during mainly down trended markets. The proposed solution showed an average quarterly ROI of -0.612% and a *Sharpe Ratio* of -0.564, the best quarter had an average ROI during the test of 16.5% and the worst quarter had an average ROI of -14.5%.

The portfolio composition system proposed by Iba & Aranha [6] showed that a Dynamic approach offers better results when compared to the simples Genetic Algorithm, the same was studied in this work context and the same conclusions were reached, that a dynamic approach exceeds the simple one by using non-stationary strattegies.

In matters of return on investment and risk, the comparison between this solution and the state-of-art offers no real value in matters of benchmark, since the time periods and assets are completely different between each one of the works, thus resulting in much different liquidity ratios and risk.

V. CONSLUSION

The main objective of the work developed is to optimize a portfolio composition problem using evolutionary computation to achieve this optimization process. The optimization is made using Dynamic Genetic Algorithms with hedge fund and stock data to better understand the underlying strategies performed by the different hedge funds. The proposed system showed a good performance at specific types of market, thus showing that the Dynamic Genetic Algorithm might be a good baseline to a portfolio composition problem. The test period is composed by 4000 stocks from 12th February of 2015 to 10th of May of 2016. The overall test period was bearish (downtrend) so the proposed system had some problems in some down trended quarters.

The proposed system was tested during a year, by creating four different test periods. The periods obtained showed an overall negative trend which was reflected in our system performance, since the model developed only has access to long market positions, thus being unable to counter balance the overall downtrend with some short positions. Regarding the four quarters used in test, only one had a positive overall S&P500 ROI, being this quarter outperformed by our proposed architecture. All the other quarters were bearish, being both indexes able to outperform the proposed system.

After scrutinizing the results obtained during the different case studies performed, the proposed system showed that can outperform both the S&P 500 and the WIA in both profit and risk if the overall period has a bullish trend (uptrend), then our algorithm is able to exploit the market and achieve better results than our benchmarks. During bearish markets, and since the proposed system has no short selling mechanisms, the overall performance is worse than both indexes during mainly down trended markets. The proposed solution showed an average quarterly ROI of -0.612% and a *Sharpe Ratio* of -0.564, the best quarter had an average ROI during the test of 16.5% and the worst quarter had an average ROI of -14.5%.

The dynamic approaches applied improved the overall performance of the Static Genetic Algorithm by being more efficient at exploiting features like adaptability, memory and diversity, being able to react to sudden changes in the market by activating different states that adjust the Genetic Algorithm parameters to more suitable ones.

REFERENCES

[1] (2015, Dec.) Form 13-Reports Filed by Institutional Investment Managers, U.S. Securities and Exchange Commission [Online].

[2] L. Wagman, “Stock Portfolio Evaluation: An Application of Genetic-Programming-Based Technical Analysis”, in Genetic Algorithms and Genetic Programming at Stanford, 2003, pp. 213-220.

[3] A. Loraschi, A. Tettamanzi, M. Tomassini, P. Verda, “Distributed Genetic Algorithms with an Application to Portfolio Selection”, in Artificial Neural Nets and Genetic Algorithms, Springer, 1995, pp. 384-387.

[4] J.O. Kyong, Y.K. Tae, M. Sungky, “Using Genetic Algorithm to Support Portfolio Optimization for Index Fund Management”, in Expert Systems with Applications, vol. 28, Elsevier, 2005, pp. 371-379.

[5] C. Aranha, H. Iba, “Modelling Cost into a Genetic Algorithm-based Portfolio Optimization System by Seeding and Objective Sharing”, in IEEE Congress on Evolutionary Computation, IEEE, 2007, pp. 196-203.

[6] A. Gorgulho, R. Neves, N. Horta, “Using GAs to Balance Technical Indicators on Stock Picking for Financial Portfolio Composition”, in Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference, ACM, 2009, pp. 2041-2046.
[7] A. Gorgulho, R. Neves, N. Horta, “Applying a GA Kernel on optimizing technical analysis rules for stock picking and portfolio composition”, in Expert Systems with Applications, vol. 38, Elsevier, 2011, pp. 14072-14085.
[8] A. Simões, R. Neves, N. Horta, “Evolutionary Computing Applied to Stock Market using Technical Indicators”, 2010.
[9] A. Silva, R. Neves, N. Horta, “A Hybrid Approach to Portfolio Composition based on Fundamental and Technical Indicators”, in Expert Systems with Applications, vol. 42, Elsevier, 2015, pp. 2036-2048.
[10] K. Michalak, P. Filipiak, P. Lipinski, “Evolutionary Approach to Multi-Objective Optimization of Portfolios that Reflect the Behaviour of Investment Funds”, in Artificial Intelligence: Methodology, Systems and Applications, vol. 7557, Springer, 2012, pp. 202-211.
[11] K. Deb, S. Agrawal, A. Pratap, T. Meyarivan, “A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II”, in: PPSN VI, LNCS, vol. 1917, Springer, 2000, pp. 849–858.
[12] D.E. Goldberg, R.E. Smith, “Nonstationary Function Optimization using Genetic Algorithms with Dominance and Diploidy”, in Proceedings of Parallel Problem Solving from Nature Conference, 1987, pp. 145-154.
[13] K.P. Ng, K.C. Wong, “A new diploid scheme and dominance change mechanism for non-stationary function optimization”, in ICGA 6, 1995, pp. 159–166.
[14] E.H.J. Lewis, G. Ritchie, “A comparison of dominance mechanisms and simple mutation on non-stationary problems”, in PPSN V, 1998, pp. 139–148.
[15] S. Yang, “Non-stationary problem optimization using the primal-dual genetic algorithm”, in CEC’03, 2003, pp. 2246–2253.
[16] S. Yang, X. Yao, “Experimental study on population-based incremental learning algorithms for dynamic optimization problems”, in Soft Computing, vol. 9, Springer, 2005, pp. 815–834.
[17] S. Yang, X. Yao, “Population-based incremental learning with associative memory for dynamic environments”, in IEEE Transactions on Evolutionary Computation, vol. 12, IEEE, 2008, pp. 542–561.
[18] R. Tinos, S. Yang, “A self-organizing random immigrants’ genetic algorithm for dynamic optimization problems”, in Genetic Programming and Evolvable Machines, vol. 8, Springer, 2007, pp. 255-286.
[19] J. Branke, T. Kaußler, C. Smidt, H. Schmeck, “A multi-population approach to dynamic optimization problems”, in Evolutionary Design and Manufacture, Springer London, 2000, pp. 299-307.
[20] F. Oppacher, M. Wineberg, “The Shifting balance genetic algorithm: Improving the GA in a dynamic environment”, in GECCO’99, vol. 1, 1999, pp. 504–510.
[21] H.G. Cobb, J.J. Grefenstette, “Genetic algorithms for tracking changing environments”, in Proc. ICGA, 1993, pp. 523–530
[22] S. Yang, R. Tinos, “Hyper-selection in dynamic environments”, in CEC’08, 2008, pp. 3185–3192
[23] S. Yang, H. Richter, “Hyper-learning for population-based incremental learning in dynamic environments”, in CEC’09, 2009, pp. 682–689
[24] P. D. Stroud, “Kalman-extended Genetic Algorithm for Search in Nonstationary Environments with Noisy Fitness Evaluations”, in IEEE Transactions on Evolutionary Computation, vol. 5, 2001, pp. 66–77.
[25] J. van Hemert, C. Van Hoyweghen, E. Lukshandl and K. Verbeeck, “A Futurist Approach to Dynamic Environments”, in GECCO EvoDOP Workshop, 2001, pp. 35–38.
[26] P. A. N. Bosman, H. La Poutre, “Learning and anticipation in online dynamic optimization with evolutionary algorithms: the stochastic case”, in Proceedings of the 9th annual conference on Genetic and evolutionary computation, ACM, 2007, pp. 1165-1172.
[27] A. Simões, E. Costa, “Using Linear Regression to Predict Changes in Evolutionary Algorithms dealing with Dynamic Environments”, in Technical Report TR 2007/005, CISUC, 2007.
[28] A. Simões, E. Costa, “Improving prediction in evolutionary algorithms for dynamic environments”, in GECCO’09, 2009, pp. 875–882.
[29] (2017, Feb.) United States Securities and Exchange Commission - Edgar Platform [Online]. https://www.sec.gov/edgar/searchedgar/companysearch.html
[30] (2017, Jan.) Distributed Evolutionary Algorithms in Python [Online]. https://github.com/DEAP/deap
[31] (2017, Jan.) Numpy Library for Python [Online]. http://www.numpy.org/
[32] (2017, Jan.) Whale Index Explanation [Online]. https://whalewisdom.com/whaleindex