Expert Systems with Applications 42 (2015) 8039–8051

Contents lists available at ScienceDirect

Expert Systems with Applications

journal homepage: www.elsevier.com/locate/eswa

Good versus bad knowledge: Ontology guided evolutionary algorithms

Hayden Wimmer [a,*], Roy Rada [b]

[a] Georgia Southern University, Department of Information Technology, P.O. Box 8150, Statesboro, GA 30460, United States
[b] University of Maryland Baltimore County, Department of Information Systems, 1000 Hilltop Circle, Baltimore, MD 21250, United States

ARTICLE INFO

Article history:
Available online 28 May 2015

Keywords:
Evolutionary algorithms
Knowledge guided
Genetic algorithm
Ontology
Decision trees

ABSTRACT

Good knowledge would be expected to help a knowledge-based algorithm more than bad knowledge. In this research, the precise effect of good versus bad knowledge on evolutionary algorithms is explored. The testable hypothesis of this paper is that good knowledge will have a significant effect on the evolutionary mutation process, whereas bad knowledge will have no significant effect. A knowledge-guided evolutionary algorithm is developed where ontologies, representing knowledge, are applied to the mutation process. Bad knowledge is represented as a randomly generated ontology, while good knowledge is represented by ontologies constructed with domain knowledge and following a formal ontology development process. Decision trees are evolved to solve a classification problem. Fitness is classification accuracy. The experiment is replicated over 2 data-sets from different domains with one being time-series, financial data and the other being wine data. As hypothesized, poorly constructed, or bad knowledge, has no effect while good knowledge is shown to have a significant effect. Bad knowledge, being random in character in these experiments, has understandably no impact on an already random mutation process. However, employing knowledge to guide the mutation process significantly constrains the traversal of the search space. Employing knowledge in an evolutionary algorithm has the potential to increase the efficiency and accuracy of evolutionary algorithms.

1. Introduction

Employing codified domain knowledge to guide the evolutionary mutation process of an evolutionary algorithm (EA) has the potential to increase the efficiency of the EA. The knowledge guides the EA by mutating genes within an organism to genes which are semantically similar. A genetic algorithm (GA) is an EA inspired by natural evolution and capable of locating an optimal solution in a complex landscape (Zarifia, Ghalehjogh, & Baradaran-nia, 2015). GAs have been applied to complex optimization problems, such as logistics scheduling (Chang, Wu, Lee, & Shen, 2014). Combining a GA with another algorithm to improve performance creates a hybrid GA, with one example being a GA for parameter optimization of a support vector machine (Chou, Cheng, Wu, & Pham, 2014). Directed GAs seek to direct the mutation operation to increase the efficiency and performance of a GA (Kuo & Lin, 2013). Employing knowledge to guide the mutation process is another option to improve performance of a GA. This research explores the effect of good versus bad knowledge guiding the evolutionary process of a GA.

Ontologies are a shared conceptualization of a domain (Gruber, 1993). Ontologies are used for communication between humans, between human and machine, or between machines, as well as for computational inference and knowledge reuse (Gruninger & Lee, 2002). Good, or well-constructed, knowledge is constructed following formal ontology development processes coupled with domain knowledge. Bad, or poorly-constructed knowledge is defined here as an ontology constructed in a random fashion. Knowledge has been used to improve genetic algorithms by optimizing the feature subset selection for input to a GA (Wendt, Cortés, & Margalef, 2010; Yang & Honavar, 1998). Case-based reasoning has been paired with GAs to find an optimal solution (Huang, Huang, & Chen, 2007). Heuristics and GAs have been paired by using GAs to extract heuristics from data (Gordini, 2014) as well as exploiting heuristics to guide the mutation process of a GA (Johns, Keedwell, & Savic, 2014; Wimmer & Rada, 2013). The aforementioned research indicates the potential for knowledge guided evolutionary algorithms.

Evolutionary algorithms, specifically genetic algorithms, are stochastic in nature and therefore unpredictable and uncontrolled. Controlling the mutation of an EA via constraining the search operation may lead to improved performance via improved fitness or reaching an optimum in fewer generations. Knowledge may be

* Corresponding author.
E-mail addresses: hwimmer@georgiasouthern.edu (H. Wimmer), rada@umbc.edu (R. Rada).

http://dx.doi.org/10.1016/j.eswa.2015.04.064
0957-4174/

exploited to constrain the search; however, first it is necessary to determine if knowledge can have an effect when guiding the search and mutation operation. A critical step is to explore good versus bad knowledge constraining the EA. In order to further explore the role of knowledge in EAs, specifically GAs, this work seeks to determine the effect of good versus bad knowledge on an EA, specifically a genetic algorithm. Genetic algorithms are stochastic in nature and therefore unpredictable and uncontrolled. The first hypothesis is formally stated as

Hypothesis 1. When employing poorly or randomly constructed knowledge to influence the genetic mutation process there will be no difference between the change in fitness between knowledge-guided and random mutation.

Furthermore,

Hypothesis 2 states. When employing well-constructed knowledge to influence the genetic mutation process there will be a difference in the change in fitness between the knowledge-guided and random mutation.

To test the hypotheses, the knowledge-guided GA is compared with a GA that mutates an organism, represented as a decision tree, at random. First, bad knowledge guiding the mutation is compared with a random mutation and second, good knowledge guiding the mutation is compared with a random mutation. The experiment is performed on 2 datasets from separate domains and each experiment repeated 5 times for 1000 generations. One of the aforementioned datasets is time series financial data; therefore, the experiment is repeated 5 times over a 4 year period for a total of 20 trials of 1000 generations.

The remainder of this paper is structured as follows: Section 2 provides a background of the framework used to develop the knowledge guided evolutionary algorithm and subsequent experiments. Section 3 gives an overview of the framework including the KGGA, datasets and ontologies, Section 4 describes the experiments and results, and Section 5 provides a discussion. Finally, Section 6 provides concluding remarks and future directions.

2. Background

The following section provides a background of the framework utilized by the knowledge guided evolutionary algorithm. Decision trees are reviewed including common decision tree algorithms and an example. Decision trees are followed by a discussion on evolutionary algorithms. The section concludes with the concept of the knowledge guided evolutionary algorithm.

2.1. Decision trees

Techniques, such as decision trees, have been widely studied and applied to a plethora of issues in computing. Decision trees are directed graphs that are employed to aid in the decision making process and used to classify data (Apté & Weiss, 1997). Decision trees are based on nodes and edges. A decision tree may be defined by its nodes and edges such as decision tree D has nodes in the set $\{N_1, N_2, \ldots, N_i\}$ and edges may be defined as two nodes in the tree such that edge $E = \{N_x, N_y\}$. There is a root node which may be defined as the initial or top node. It is also defined as a node with no parent nodes. Child nodes may be referred to as inner nodes. Additionally, a terminating node, which is referred to as a leaf node, is a node which has no additional child nodes. Decision trees divide, or split, on a node. This is referred to as branching. Trees also have depth which may be defined as the number of edges to reach the root node. The root node is considered a level or depth of 0. Decision trees create unique paths to the various terminating – or leaf – nodes. Each of these paths represents a potential decision. These unique paths may be extracted as a rule set with each representing a separate rule based on specific conditions.

Machine learning is an artificial intelligence technique in which computer algorithms train themselves based on input data in order to make predictions (Mitchell, 1997). More formally, machine learning may be defined as "A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E" (Mitchell, 1997). There are four machine learning algorithms for decision trees which are commonly utilized and implemented in data mining software packages. Classification and regression tree – or CART – is an umbrella term applied to techniques which predict the class of an instance (classification) or a numerical attribute such as price (regression) (Loh, 2011; Olshen & Stone, 1984; Ripley, 2005). More specific decision tree algorithms are Chi-Squared Automated Interaction Detector (CHAID), ID3, and C4.5. CHAID employs statistical significance testing to determine nodes and splits within the decision tree (Ozgulbas & Koyuncugil, 2009). Typically, Bonferroni testing is utilized, although the algorithm may be customized using other statistical tests based on the application. ID3 computes the maximum information gain or least amount of entropy. First, a data set's attributes are checked to determine which has the maximum information gain. The root node becomes the attribute with the highest information gain and then the remaining attributes are passed again to the ID3 algorithm recursively until all instances have been classified or all attributes utilized. The algorithm may be customized in many ways that are application specific such as defining a maximum depth (Quinlan, 1986). C4.5 is an extension of ID3 which provides support for non-discretized values (Quinlan, 1993). C4.5 has since been extended to C5.0 which, unlike ID3 and C4.5, is a proprietary algorithm (Quinlan, 2012).

Decision trees may be extracted into heuristics which can be used for decision support. The weather dataset is a standard dataset available from the UCI Machine Learning Repository (Bache & Lichman, 2013) and applied to decision tree learning such as random forest generation (Livingston, 2005). The dataset has 5 attributes and 14 records which are used to predict whether to play a game based on the weather. Fig. 1 presents the weather dataset in ARFF format. Fig. 2 shows a C4.5 decision tree created

```
@relation weather

@attribute outlook {sunny, overcast, rainy}
@attribute temperature real
@attribute humidity real
@attribute windy {TRUE, FALSE}
@attribute play {yes, no}

@data
sunny,85,85,FALSE,no
sunny,80,90,TRUE,no
overcast,83,86,FALSE,yes
rainy,70,96,FALSE,yes
rainy,68,80,FALSE,yes
rainy,65,70,TRUE,no
overcast,64,65,TRUE,yes
sunny,72,95,FALSE,no
sunny,69,70,FALSE,yes
rainy,75,80,FALSE,yes
sunny,75,70,TRUE,yes
overcast,72,90,TRUE,yes
overcast,81,75,FALSE,yes
rainy,71,91,TRUE,no
```

Fig. 1. Weather dataset from the UCI machine learning repository in ARFF format.