Information Sciences 237 (2013) 118–136

Contents lists available at SciVerse ScienceDirect

Information Sciences

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

A multiobjective evolutionary programming framework for graph-based data mining

Prakash Shelokar [a], Arnaud Quirin [a], Óscar Cordón [a,b,*]

[a] European Centre for Soft Computing, 33600-Mieres, Spain
[b] Department of Computer Science and Artificial Intelligence (DECSAI) and Research Centre on Information and Communication Technologies (CITIC-UGR), University of Granada, 18071 Granada, Spain

ARTICLE INFO

Article history:
Received 20 March 2012
Received in revised form 6 December 2012
Accepted 12 February 2013
Available online 4 March 2013

Keywords:
Graph-based data mining
Frequent subgraph mining
Evolutionary multiobjective optimization
Pareto optimality
Multiobjective graph-based data mining
Multiobjective evolutionary programming

ABSTRACT

Subgraph mining is the process of identifying concepts describing interesting and repetitive subgraphs within graph-based data. The exponential number of possible subgraphs makes the problem very challenging. Existing methods apply a single-objective subgraph search with the view that interesting subgraphs are those capable of not merely compressing the data, but also enhancing the interpretation of the data considerably. Usually the methods operate by posing simple constraints (or user-defined thresholds) such as returning all subgraphs whose frequency is above a specified threshold. Such search approach may often return either a large number of solutions in the case of a weakly defined objective or very few in the case of a very strictly defined objective. In this paper, we propose a framework based on multiobjective evolutionary programming to mine subgraphs by jointly maximizing two objectives, support and size of the extracted subgraphs. The proposed methodology is able to discover a nondominated set of interesting subgraphs subject to tradeoff between the two objectives, which otherwise would not be achieved by the single-objective search. Besides, it can use different specific multiobjective evolutionary programming methods. Experimental results obtained by three of the latter methods on synthetically generated as well as real-life graph-based datasets validate the utility of the proposed methodology when benchmarked against classical single-objective methods and their previous, nonevolutionary multiobjective extensions.

1. Introduction

Much of the real world data have spatial or temporal characteristics. Some examples include web data [35], transportation networks [44], scientific information analysis [48], bioinformatics [51], and Oligonucleotide probe design [59], among others. In recent years, labeled graphs have emerged as a promising abstraction to model such complex data [1,11]. In this approach, each object corresponds to a separate graph whose nodes represent the entities in the object, and edges represent the relations between the entities. Within that model, one method of formulating a data mining task is that of finding subgraphs that occur frequently over the graph-based data, called frequent subgraph mining. Frequent subgraphs are especially useful in characterizing graph sets, detecting network motifs [7], discriminating different groups of graphs [3], classifying and clustering graphs [17,24], and building graph indices [67]. A number of efficient frequent subgraph mining algorithms

* Corresponding author at: Department of Computer Science and Artificial Intelligence (DECSAI) and Research Centre on Information and Communication Technologies (CITIC-UGR), University of Granada, 18071 Granada, Spain.
E-mail address: oscar.cordon@softcomputing.es (Ó. Cordón).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.02.014

List of symbols

G, g	graph dataset and a graph $g \in G$
S	subgraph in the dataset G
$MDL(.)$	the minimum description length measure in Eq. (1)
$DL(.)$	description length of (sub) graph in Eq. (1)
$\mathbf{f}$	vector of objective functions in Eq. (2)
$\mathbf{y}$	vector of objective values in Eq. (2)
d	size of vector of objective values in Eq. (2)
$\mathbf{x}$	vector of decision variables in Eq. (2)
o	size of vector of decision variables vector in Eq. (2)
$\mathbb{X}$	solution search space
$\mathbb{Y}$	objective space
$\mathcal{P}$	the Pareto optimal set defined in Eq. (4)
$\mathcal{PF}$	the Pareto optimal front defined in Eq. (5)
PF	an approximated Pareto front produced by an algorithm
P	population of subgraph individuals
Q	population of child subgraph individuals
R	temporary population of $P \cup Q$
PA	external Pareto archive of nondominated subgraph individuals
N	number of subgraphs in the population P
$\#\mathcal{N}(.)$	function to compute the support of a subgraph in Eq. (6)
$\#\mathcal{V}(.)$	function to compute the size of a subgraph in Eq. (7)
fitness	fitness of an individual in the population
r	rank of an individual in the population
$sh(.)$	sharing function in Eq. (9)
σ_{share}	the maximum distance allowed, a parameter, in Eq. (9)
a	positive scaling factor, a parameter, in Eq. (9)
c_{ij}	the normalized distance between two individuals i and j in Eq. (10)
nc	niche count in Eq. (11)
cd	crowding distance in Eq. (14)
$f^l_{\min}$, $f^l_{\max}$	the minimum and maximum values of the lth objective in Eqs. (10) and (14)
f^l	value of lth objective in Eqs. (10) and (14)
$\lvert . \rvert$	size of population/graph dataset/nondominated set

are available in the Graph-Based Data Mining (GBDM) community, including Subdue [10,13], Gaston [42], the apriori family of methods [27], FSG[34], gSpan [65], MoFa/MoSS [3], JoinPath [61], CloseGraph [66], FFSM [25], Spin [26], CLOSECUT and SPLAT [68], gPrune [69], etc. Recently, evolutionary programming [18,19] has also been applied for frequent subgraph mining [2,39]. The proposal was basically an extension of Subdue and showed an improved performance over it. The performance improvement was a consequence of the use of global search instead of the beam search [37] with no backtracking as applied by the standard Subdue method in the subgraph search space.

Frequent subgraph mining is generally guided by a single objective, i.e., applying some threshold, such as mine subgraphs whose *frequency* (or *support*) is above a specified threshold, or mine subgraphs whose *size* is greater than some user-defined threshold. Other objectives may be defined as well towards obtaining the mined subgraphs [1,11]. This approach has some important limitations [44]. In some cases where the constraints are weak, the number of mined subgraphs is large, posing difficulties in selecting the most important ones. In other cases where the constraints are too strict, we risk an empty output or very few subgraphs.

In addition, any successful methodology should consider additional criteria to extract better defined concepts based on the complexity of the subgraph being explained, the number of retrieved subgraphs, and their diversity [12,50,51,53,73]. These are conflicting criteria that can be approached as an optimization problem, close in spirit to the minimum description length methods [49], which are based on the aggregation of the various objectives into a global measure of subgraph quality. The basic challenge with this approach is the potential bias caused by weighting the objectives [52,53], which always derives from the convergence to solutions corresponding to single or limited regions of the search space. This problem is noteworthy because typical data mining approaches, particularly in computational biology, tend to emphasize consensus or most frequent subgraphs [72]. These subgraphs often conceal rather than reveal novel and useful knowledge about the problem, retrieving only already known or irrelevant information that discourages the use of computational methods [38,40]. Consequently, there is a need for new methods that can provide even less frequent but more descriptive substructures that reflect problem descriptions from different angles [74]. This emphasizes the application of multiobjective subgraph mining in GBDM.

Multiobjective subgraph mining has been termed as *multiobjective GBDM* [51,55,58] or *skyline processing* [44]. To tackle this problem, one method is of formulating a single-objective optimization problem in which the objective is an aggregation function of all the individual objectives, as classically done in multiobjective optimization literature [6,9,15,60]. However, this will discover only the specific subgraphs subject to the tradeoff between the objectives explicitly or implicitly specified by the aggregation function, as seen in [58] for a Subdue variant based on a weighted combination of support and size.

Apart from that basic approach, there are a few instances of multiobjective GBDM methods in the literature. Papadopoulos et al.'s SkyGraph [44] actually manages to generate Pareto-optimal subgraphs defined by two specific objectives, the edge connectivity and the size of the subgraph, by means of a very advanced and well designed polynomial time, exact algorithm. However, the drawback of SkyGraph is that it is problem-specific, i.e., it can only be applied to the latter concrete multiobjective frequent subgraph mining task. This specificity allows it to use a single-objective (and not multiobjective) underlying search method. Romero-Zaliz et al. [51] introduced the EMO-CC methodology (Evolutionary Multiobjective Optimization-based Conceptual Clustering) for the Gene Ontology domain. In EMO-CC, the chromosome representation used is a tree-like subgraph, which is evaluated using two objectives, support and size. However, it cannot handle large graphs having a more general graph representation, where a node may have several parents. Finally, MOSubdue (Multi-Objective Subdue), a Pareto dominance-based multiobjective subgraph mining algorithm, was developed by the authors in [55,58]. It is an adaptation of Subdue, as MOSubdue selects subgraphs for the underlying beam search by means of NSGA-II's [16] nondominated sorting method using two objectives, support and size. Nevertheless, MOSubdue has an important limitation as Subdue's beam search performs a state space search [43], which does not allow backtracking in the multiobjective subgraph solution space. The algorithm is limited in exploring the different search paths due to the choice of the beam width, and thus it may end up providing sub-optimal results.

In this paper, we present a pure Evolutionary Multiobjective Optimization (EMO) (and, more specifically, Multiobjective Evolutionary Propgramming (MOEP)-based) framework to perform global search in the multiobjective subgraph solution space. Thus, it allows the user to obtain a good approximation to the whole set of Pareto-optimal subgraphs at a reasonable computational effort. In the proposed MOEP framework, an individual in the population is a subgraph in the input data. The input dataset is a set of connected relational graphs with or without cycles and directed or undirected edges. Each individual in the population is evaluated using two objectives, support and size. At any generation, parent individuals give rise to child individuals through mutation, and subsequently the next generation is selected from the collection of parent and child individuals.

We empirically demonstrate the feasibility of evolutionary search for the multiobjective GBDM task on 10 different datasets. They include three synthetically generated and seven real-life datasets concerning the analysis of web sites [22] and scientograms (visual representations of scientific domains) [48], respectively. Three different MOEP approaches, namely MOEP-based on Nondominated Sorting (MOEP-NS) [63], MOEP with Summation of Objectives (MOEP-SO) [46], and MOEP with nondominance (MOEP-ND) [41] have been implemented as specific methods within our proposed framework. To compare their performance, we have also applied two single-objective methods, Subdue [10,13] and EP-Subdue [2], as well as a multiobjective one, MOSubdue [58], for the multiobjective GBDM task. Their performance is compared using different multiobjective metrics commonly employed in the EMO field [9,70,71].

The paper is organized as follows. Section 2 presents some preliminaries related to single- and multiobjective GBDM using Subdue and EP-Subdue. Section 3 outlines the proposed MOEP framework for multiobjective GBDM. The experimental analysis is given in Section 4, and finally some concluding remarks and proposals for future work are provided in Section 5.

2. Preliminaries

2.1. Graph-based data mining

Fig. 1 represents a subgraph lattice for a graph dataset $G = \{g_1, g_2, g_3, g_4\}$. The subgraph lattice in Fig. 1 models the search space for frequent subgraph mining in the dataset G, where the top of the lattice, i.e., at level 0, is the empty subgraph labeled *. The first level includes all possible single node subgraphs. The second level of the lattice represents subgraphs with one edge, and so on. At the bottom of the lattice graphs in G are shown. Frequent subgraph mining can be formulated as finding the embedding subgraphs in the lattice. For example, at level 1 of the lattice, a single node subgraph Q can be embedded in two single edge subgraphs P–Q and Q–R at level 2. In Fig. 1, the subgraph P–Q at level 2 is a parent of the subgraph P–Q–R at level 3 of the lattice as the subgraph P–Q–R is different from the subgraph P–Q by exactly one edge. Thus, the subgraph P–Q–R is a child of the subgraph P–Q. All the subgraphs of $g_i \in G$ are present in the lattice and every subgraph occurs only once in it.

All the GBDM algorithms have in common that they search a subgraph lattice of the latter kind representing all possible subgraphs. They are interested in finding subgraphs whose support is above a user specified threshold. Thus, the goal is to discover all subgraphs that are frequent in this dataset. For instance, corresponding to a frequency threshold of three and an order threshold of two, the algorithm will output the circled subgraph (at level 2) shown in Fig. 1 as it can be embedded thrice in the running example.

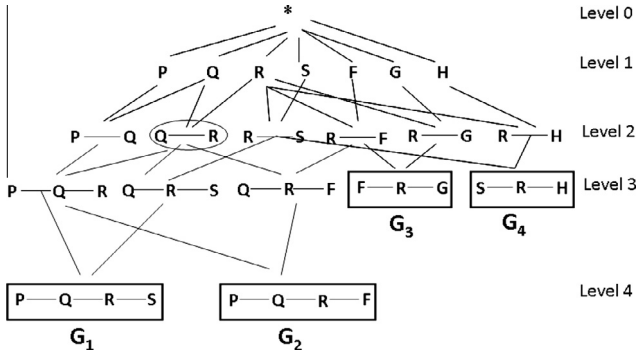

Fig. 1. A subgraph lattice of the graph dataset $G = \{g_1, \ldots, g_4\}$ (modification of a figure presented in Jiang et al. [30]).

Searching this lattice can be done depth- or breadth-first. Most of the graph mining algorithms traverse the lattice in a depth-first way as it needs less memory than breadth-first search. Exploring the hypothesis space involves two highly complex steps: *candidate generation*, where child subgraphs are created from parent subgraphs, and *support computation*, where the support of candidate subgraphs in the input data is obtained. There are two popular ways of creating child subgraphs: extending parent subgraph by an edge (and a node if no cycle is closed) or merging parent subgraphs that share a common core. Edge extension needs access to the input data and only creates existing candidates. This strategy has been most widely used. The other possibility to generate child subgraphs is by merging two parent subgraphs that have common core. However, this merging approach solely works on the subgraphs and may generate infeasible candidates which may not constitute to be part of the input data.

Support calculation of a candidate subgraph involves several computationally intensive subgraph and graph isomorphism tests. There are two possibilities to reduce the number of tests. One possibility is to use a so-called appearance list that records the graphs as candidate subgraphs occur in. Another possibility is to use a so-called embedding lists. An embedding is a subgraph isomorphism of a subgraph in the lattice to a graph in the input data. In general the use of embedding lists needs much more memory.

Recently evolutionary algorithms have been proposed to work on graph-represented data for dealing with tasks such as community detection [45], pattern matching [33], graph isomorphism [64], etc. They have shown encouraging results. However, the specialized solution encoding schemes and fitness functions employed to solve particular tasks do not make them readily adaptable for frequent graph mining problem, where the computation of the objective, support of a subgraph, is crucial.

2.2. The Subdue algorithm

Subdue [10,13] is a classical method in GBDM that performs different tasks such as subgraph discovery, conceptual clustering, and classification. The search process of Subdue is guided by the minimum description length (MDL) principle [49], which assumes the best subgraph is the one that minimizes the description length of the input graph when compressed by the subgraph [13]. This compression is calculated as:

$$\mathrm{MDL}(G,S) = (DL(S) + DL(G|S))/DL(G) \tag{1}$$

where $DL(S)$ and $DL(G)$ are the description lengths of the subgraph S and the complete graph G, and $DL(G|S)$ is the description length of the input graph G compressed by the subgraph S. The description length of a graph is calculated here as the number of bits needed to encode an adjacency matrix representation of the graph [10,13]. The Subdue algorithm attempts to maximize the value of the subgraph S, which is the multiplicative inverse of the compression in Eq. (1). Notice that, to evaluate a subgraph, the compression measure in Eq. (1) jointly considers two commonly used objectives in GBDM, the support and the size of the subgraph.

Subdue performs a level-wise search in the subgraph lattice as shown in Fig. 1. Subdue starts with a parent list of single node subgraphs corresponding to unique label nodes in the input graph. It creates child subgraphs from each subgraph in the parent list by expanding each of its instances in all possible ways. The expansion is done by adding an edge (and a node if no cycle is closed). These subgraphs are stored in a child list in ascending order of MDL values. In order to avoid an exponential explosion, Subdue applies a beam search [37], limiting the number of subgraphs in the child list by the choice of a threshold parameter called *beam-width*. For the next level of expansion, the child list becomes the parent list. The algorithm iteratively applies this process until reaching a user specified limit on the number of parent subgraphs to be extended or exhaustion of the search space. The output is a list of the best subgraphs found.

The drawback of Subdue is that it uses a computationally constrained beam search that discards some of the less promising subgraphs at an early stage of the exploration. This thereby terminates the possibility of expanding these less promising subgraphs later on in order to search other promising subgraph solution space regions. Thus, beam search is a kind of search

in the state space of subgraphs lattice not allowing backtracking. Therefore, it may often end up providing sub-optimal results.

2.3. Evolutionary programming

Evolutionary programming (EP) is a special type of evolutionary algorithm (EA) that seeks to optimize a fitness function. To this end, it simulates the evolution process by repeatedly executing the following loop [18,19]: (i) Initially, a population P consisting of N individuals, each representing a candidate solution, is generated at random; $|P|$ specifies the population size per generation. (ii) In each generation, a population Q of offspring individuals is created; To generate a single offspring, a mutation operator is utilized on each parent individual from P. (iii) The offsprings are evaluated and a temporary population $R = P \cup Q$ is created. Among the individuals in R, the selection operator chooses the best N candidates, which then form the population P of the next generation. (iv) The whole procedure is repeated until a stopping criterion is met. We refer [19] for a general introduction to EP.

2.4. Evolutionary programming-based Subdue

One way to overcome the drawback of Subdue is the use of a global search to explore different layers of the subgraph lattice using a population of subgraphs. One straight away improvement to the beam search of Subdue was maintaining the parent list like a population in EP [2,39]. Unlike Subdue, this EP-Subdue method utilizes a population P of subgraphs in order to explore different regions of the subgraph lattice in each generation.

At any generation, a child subgraph population Q is created using mutation on each parent subgraph from P. The mutation applies Subdue's original subgraph expansion procedure. During mutation of an individual, the subgraph S that it holds is extracted. Each of its instances is expanded in all possible ways to create new child subgraphs. Among the generated child subgraphs from the subgraph S, a child subgraph is randomly selected that becomes the mutated individual in the population Q. The fitness of this new individual is to be maximized and evaluated as the multiplicative inverse of the compression in Eq. (1). The global best individual is updated using the individuals in Q. To yield the next generation, the population is created by copying the global best individual, and selecting the remaining individuals from $R = P \cup Q$ based on the fitness proportionate selection. The method continues exploring the search space for a maximum number of given generations. A single best subgraph as pointed by the global best individual is then returned.

The performance of EP-Subdue was tested on several WWW datasets [22] and compared against that of the standard Subdue method. The results indicated that the EP-Subdue approach had often outperformed the Subdue method. The reason is that EP is a global search that can simultaneously explore different regions in the search space of the subgraph lattice.

This EP-Subdue [2,39] implementation has one drawback. Mutation creates a new individual by random selection of a child subgraph among all the possible child subgraphs generated from the individual. Generating all the possible child subgraphs for choosing just a single child subgraph is a needless waste of computational resources. Instead, it would be faster and much more efficient if a feasible child subgraph is randomly created from an individual. With this view, we have modified the mutation operator in EP-Subdue as follows. To mutate the subgraph S, first the instances of subgraph S in the input dataset are expanded by adding either an edge and a node or an edge only. This gives rise to a list of new instances. From this list an instance is picked up randomly that becomes the definition of child subgraph, and only the instances that match the definition of this child subgraph are retained. Thus, this new mutation operator has discarded all other possible child subgraphs that could otherwise be produced by the original mutation operator of EP-Subdue to generate a single child subgraph.

We have applied this new mutation in the EP-Subdue method and tested its performance on several WWW datasets[1] [22]. We have used five datasets which were also employed by EP-Subdue in [2]. Both EP-Subdue variants were executed over an equal run time as given in Table 1.

The results represent the average graph compression values over 10 executions for both methods, together with the standard deviations in parentheses. The lower value corresponds to higher compression of the input graph according to Eq. (1). As can be seen, the new variant incorporating the modified mutation operator clearly outperforms the original approach on all the input datasets. Therefore, we will consider this new mutation operation in our proposals for multiobjective GBDM which will be detailed in Section 3.

2.5. Multiobjective optimization in data mining and multiobjective graph-based data mining

Several studies have shown that multiobjective learning approaches are more powerful compared to learning algorithms with a scalar objective function in addressing various topics of machine learning. A nonexhaustive list of examples includes classification, clustering, feature selection, improvement of generalization ability, knowledge extraction, system identification, and ensemble generation [4,5,14,23,29,36]. The concept of Pareto optimality [6] has been recently applied to machine learning and data mining, particularly inspired by the successful developments in EMO [8,31,32].

[1] http://ailab.wsu.edu/subdue/datasets/webdata.tar.gz.

Table 1
Comparison of results between modified and original EP-Subdue methods.

Dataset	#Nodes	#Edges	Run time (s)	EP-Subdue	
				Modified	Original
W1	8	30	1	0.7928(0.01)	0.7969(0.00)
W2	31	43	1	0.6060(0.01)	0.8157(0.00)
W3	19	159	50	0.7086(0.03)	0.7383(0.04)
W4	22	314	50	0.5237(0.00)	0.5964(0.02)
W5	94	106	100	0.9176(0.02)	0.9257(0.02)

A general multiobjective optimization (MOO) problem can be described as a vector function $\mathbf{f}$ that maps a tuple of o parameters (decision variables) to a tuple of d objectives [6,21]. Assume, without loss of generality, a maximization problem:

$$\begin{aligned} \max . \ \mathbf{y} &= \mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_d(\mathbf{x})) \\ \text{subject to } \mathbf{x} &= (x_1, x_2, \ldots, x_o) \in \mathbb{X} \\ \mathbf{y} &= (y_1, y_2, \ldots, y_d) \in \mathbb{Y} \end{aligned} \tag{2}$$

where $\mathbf{x}$ is called the decision vector, $\mathbb{X}$ is the parameter space, $\mathbf{y}$ is the objective vector, and $\mathbb{Y}$ is the objective space. To compare any two solutions, we apply the well known concept of Pareto dominance: consider two solutions $\mathbf{x_1}$ and $\mathbf{x_2}$ with vector-valued objective functions $\mathbf{y_1}$ and $\mathbf{y_2}$ respectively. An objective vector $\mathbf{y_1}$ is said to weakly dominate another objective vector $\mathbf{y_2}$ ($\mathbf{y_1} \succeq \mathbf{y_2}$) if no component of $\mathbf{y_1}$ is smaller than the corresponding component of $\mathbf{y_2}$ and at least one component is greater. Accordingly, we can say that a solution $\mathbf{x}_1$ is better than another solution $\mathbf{x_2}$, i.e., $\mathbf{x}_1$ weakly dominates $\mathbf{x_2}$ as $\mathbf{f}(\mathbf{x}_1) \succeq \mathbf{f}(\mathbf{x_2})$. Mathematically, the concept of Pareto dominance is defined as follows:

$$\begin{aligned} &\forall i \in \{1, 2, \ldots, d\} : f_i(\mathbf{x_1}) \geqslant f_i(\mathbf{x_2}) \wedge \\ &\exists j \in \{1, 2, \ldots, d\} : f_j(\mathbf{x_1}) > f_j(\mathbf{x_2}) \end{aligned} \tag{3}$$

A solution $\mathbf{x} \in \mathbb{X}$ with objective vector $\mathbf{y}$ is said to be Pareto optimal with respect to the search space $\mathbb{X}$ iff there is no solution $\mathbf{x}' \in \mathbb{X}$ with objective vector $\mathbf{y}'$ that dominates $\mathbf{y}$. For a MOO problem in Eq. (2), the Pareto optimal set $\mathcal{P}$ is defined as:

$$\mathcal{P} := \{\mathbf{x} \in \mathbb{X} | \neg \exists \mathbf{x}' \in \mathbb{X} \mathbf{f}(\mathbf{x}) \preceq \mathbf{f}(\mathbf{x}')\} \tag{4}$$

and the Pareto-optimal front $\mathcal{PF}$ associated with the Pareto optimal set $\mathcal{P}$ is defined as:

$$\mathcal{PF} := \{\mathbf{f}(\mathbf{x}) = (f_i(\mathbf{x}), \ldots, f_d(\mathbf{x})) | (\mathbf{x} \in \mathcal{P})\} \tag{5}$$

Hence, there is not usually a single optimal solution to solve a typical MOO problem but a set of optimal solutions when all the objectives are jointly considered. These optimal solutions are known as nondominated, efficient, or Pareto optimal, and constitute the so-called nondominated, efficient, or Pareto-optimal solution set. Their set of objective vectors is called the nondominated, efficient or Pareto front (*PF*). With the aid of EMO, it may be possible to obtain an approximation to the true Pareto front of nondominated solutions for MOO problems [8,31,32].

In this study, a multiobjective GBDM problem is formulated as that of finding an optimal set of subgraphs corresponding to the maximization of two objectives, support and size, in the search space of the subgraphs [51,58]. A subgraph is evaluated considering two objective functions: (i) the support (the occurrence frequency of the subgraph S in the input set of graphs G), and (ii) the size or order (the number of nodes in the subgraph S). These objectives can be calculated as:

$$support(G, S) = \#\mathcal{N}(g_i | S \subseteq g_i, i = 1, \ldots, |G|) \tag{6}$$

$$size(S) = \#\mathcal{V}(S) \tag{7}$$

where $\#\mathcal{N}(.)$ is the number of graphs in G which contains the subgraph S and $\#\mathcal{V}(.)$ return the number of nodes of the subgraph S. The calculation of support of S requires subgraph isomorphism test for determining how many graphs in G contain a subgraph that is isomorphic to S. It is an NP-complete task [65]. Both objectives are clearly in conflict. More frequent (or higher support) subgraphs represent sounder insights but often of small size, while large size subgraphs represent more concise (and thus more difficult to uncover) descriptions but of small support values. Therefore, there may exist several optimal objective vectors representing tradeoff between the objectives. A user will be interested in an algorithm that will allow her/him to uncover cohesive subgraphs comprising even a moderate number of concepts (and not only the ordinary ones, as usual) which describe the underlying phenomena from different angles, revealing novel information that otherwise would be concealed by the usual uninformative mined concepts. In the following section, we briefly describe one such algorithm for multiobjective GBDM.

2.6. The multiobjective Subdue algorithm

MOSubdue [58] is an extension of the single-objective Subdue method that utilizes a Pareto dominance-based approach for selecting child subgraphs for the next level of expansion. It employs NSGA-II's [16] nondominated sorting procedure for evaluating the fitness of child subgraphs.

Based on the ways of selecting child subgraphs for the beam search, two different MOSubdue methods were applied. In the first approach (MOSubdue-I), a *beam-width* number of child subgraphs are selected according to their fitness values. MOSubdue-I performs a deterministic search as the selection is performed on the child list ordered according to the fitness of child subgraphs. In the other approach (MOSubdue-II), a *beam-width* number of diversified child subgraphs from a nondominated front are selected according to NSGA-II's crowding-distance measure computed for each subgraph in that front. When two subgraphs share the same fitness value, the one located in the less crowded-region is preferred. The effect of diversified child subgraphs selection is that MOSubdue-II may perform a stochastic search in the multiobjective search space of the subgraphs.

As seen in [58], the use of a Pareto-based search strategy and a diversified subgraph selection can benefit the MOSubdue algorithm in achieving a good nondominated set approximation. However, in MOSubdue-II the crowding distance-based selection of child subgraphs at any expansion level is biased towards one objective, support, as the subgraphs are nearly indifferent in the other objective, size. The reason is that Subdue's constructive search generates nearly equal size child subgraphs by adding either an edge and a node or an edge only to the parent subgraphs. Moreover, at the early expansion stage, the algorithm usually generates a large number of child subgraphs belonging to the first nondominated front. This indicates that the diversified child subgraphs selection will not help the MOSubdue-II method to achieve a well distributed nondominated set of subgraphs [58]. This suggests an implementation of EMO-based GBDM could be more useful. This is due to the fact that at any generation, it can explore different layers of the lattice in the multiobjective subgraph search space. The next section is devoted to introduce an algorithm of that kind.

3. Multiobjective evolutionary programming for subgraph mining

3.1. Graph and subgraph representation

The input dataset G to the MOEP-based GBDM methodology consists of feature-values that map to nodes and relationships between them that map to edges. For example, Fig. 2 shows the input G corresponding to the Shapes geometrical domain [10]. The objects in the figure (e.g., C, T, S) become labeled nodes in the graph and the relationships (e.g. on (T, S), on shape triangle) become labeled edges in the graph G.

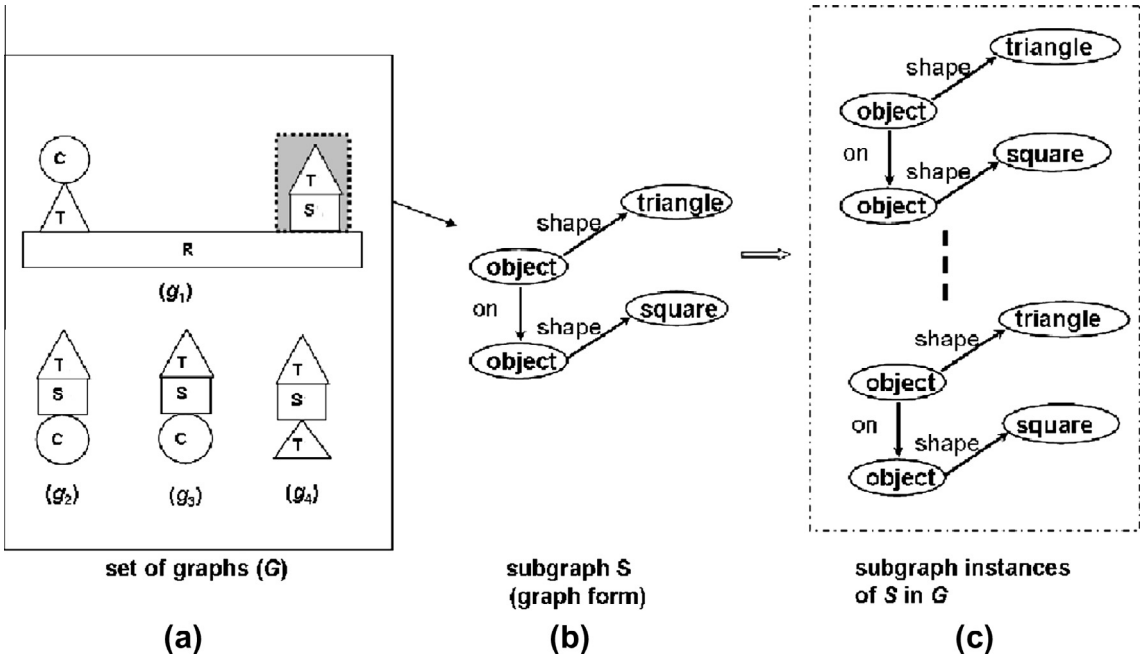

Fig. 2. An example of a subgraph and its instances in geometrical domain: (a) an input set of graphs G; (b) a subgraph S being the graphical representation corresponding to the object in gray shade in G; (c) a list of instance subgraphs in G that match to the subgraph S. The subgraph S has four instances in G.

3.2. Individual representation

An individual in the population P is represented as a possible subgraph S in the dataset G. P is a set of N subgraphs $S_1, S_i, \ldots, S_N \in \mathbb{X}$, where S_i is a connected subgraph within the graphical representation for all those subgraphs in G that match to the subgraph S. This graphical representation serves as a solution in the MOEP-based GBDM framework. It is clarified with an example given in Fig. 2. Consider subgraph S is a solution under the subgraph mining task within the input G. One of the four instances of the subgraph S is highlighted in gray area in the figure. The instance shown in the gray area in G is a subgraph having a set of nodes and edges from G. This subgraph can become the instance of subgraph S only if there is the graph theoretic match as one-to-one mapping of all nodes of the two subgraphs (i.e. the subgraph S in graphical representation and the subgraph shown in the gray area) such that all the adjacency relationships are preserved. A list of the four subgraph instances of the solution S in G that match to the subgraph S is also shown in Fig. 2c. In the proposed MOEP framework, an individual (or a solution) in the population P is always composed of a subgraph S with its associated instances in the input dataset G.

3.3. Population initialization

Usually, an EP initializes the population P by random generation of individuals in the solution search space. For the current application, we can form the initial population of possible subgraphs in G from the output of any GBDM method, such as the Subdue algorithm or any other. This population can contain subgraphs from different layers of lattice. In this work, a simple procedure has been applied to initialize the population. Referring to Fig. 1, initially, all single node (i.e. one-size) subgraphs are created from unique label nodes in G. The next layer of the lattice is then explored by extending all the instances of these subgraphs by drawing feasible edge and a node from the graph data. This will produce candidate subgraphs of the same size (i.e., of order three) but may have different values for the other objective, support. An initial population contains subgraphs randomly selected from these candidate subgraphs. More sophisticated procedures can also be applied to generate the initial population that may well represent different search space solutions.

3.4. Candidate subgraphs generation

The current application of our MOEP approach for GBDM relies only on the use of a mutation operator for candidate generation. Mutation yields a maximum of N new candidate subgraphs as each parent subgraph S in P is used to create a new child subgraph. All the instances in G of the parent subgraph S are extended by an edge (and a node if no cycle is closed) in order to create child instances. A child instance is then randomly selected that becomes a child subgraph in graphical representation. All of the child instances that match this child subgraph become its new instances. This child subgraph should have at least two instances in G to qualify as a child subgraph of the parent subgraph S. Otherwise a new child instance will be randomly selected to form a child subgraph.

A child subgraph generation is illustrated in Fig. 3 by considering a parent subgraph S as shown in Fig. 2 with its associated instances. In Fig. 3a, the four instances of the subgraph S are extended by an edge and a node which has produced four child instances. Three different child subgraphs can be assembled from these child instances, as can be seen in Fig. 3b. Out of three child subgraphs, one with a support value of two will be chosen. The mutation applies extending parent subgraph by an edge, which always create existing (or feasible) candidates in the input G. This is the most commonly used candidate generation method employed by GBDM techniques [10,65].

3.5. Objective functions calculation

An individual S in the population P is evaluated using the two objective functions given by Eqs. (6) and (7). To compute the support objective, the present MOEP framework applies a kind of appearance list that records the graphs in which a candidate subgraph occurs [10,13]. An individual S in P is always associated with its subgraph instances in the input G. It suffices to check all the subgraph instances in the appearance list instead of testing the whole input data for calculating the support. The second objective, size (or order) of the subgraph S, is equal to the number of nodes present in S.

3.6. Implementing multiobjective evolutionary programming methods for multiobjective subgraph mining

In our earlier studies [56,57] with preliminary results, we have shown different applications of EP for multiobjective GBDM. This study involves proposing a general framework based on EP for GBDM. To this end, in the following, we consider three different MOEP methods, namely MOEP-ND, MOEP-NS, and MOEP-SO. MOEP-ND is newly proposed in this study while the other two were previously described in [56,57].

MOEP-ND: This EP algorithm was recently proposed for multiobjective optimization in power generation expansion planning [41]. The algorithm employs the nondominance to evaluate individuals' fitness and applies niching for diversified selection of individuals. MOEP-ND utilizes an external archive PA to store nondominated solutions. Here, an application of MOEP-ND for subgraph discovery can be given in the following steps as:

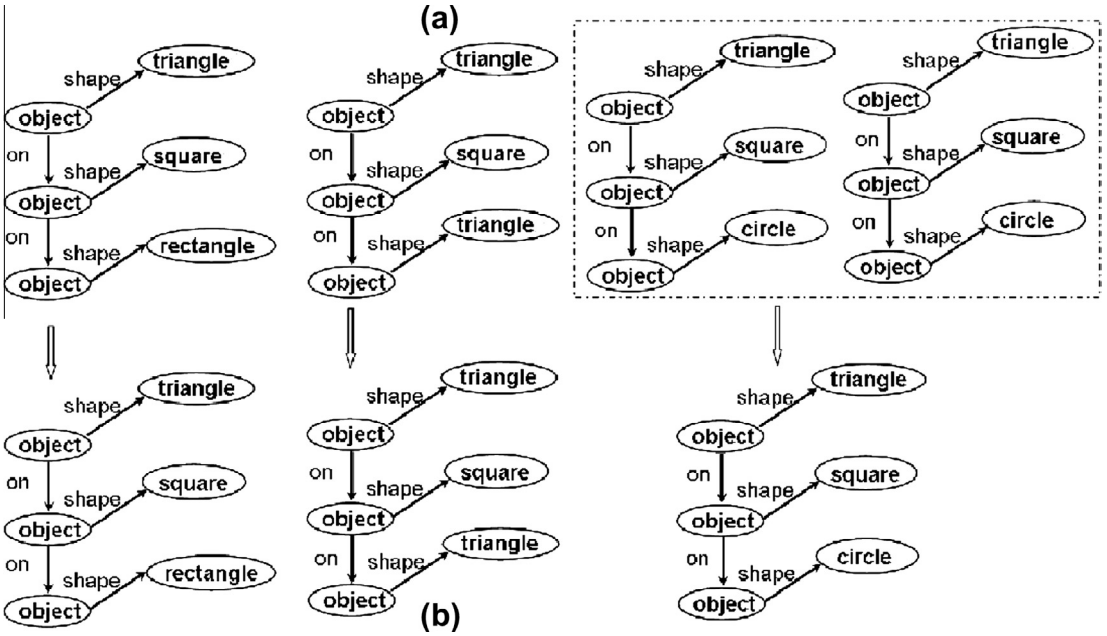

Fig. 3. An example of the mutation operation that creates a candidate subgraph. (a) Four possible child instances from the expansion of instances in G corresponding to the subgraph S shown in Fig. 2. (b) Three different child subgraphs in graphical representation corresponding to the four child instances in (a).

Step 1: Initialize the population P of N individual subgraphs, and an external population $PA = \emptyset$.
Step 2: Update PA using nondominated solutions in P.
Step 3: Create a population Q of child subgraphs from parent subgraphs in P by mutation.
Step 4: Update PA using nondominated solutions in Q.
Step 5: Combine the two populations to produce a temporary population $R = P \cup Q$.
Step 6: Assign fitness to individuals in the current population R using Pareto dominance [41]. Note that dominance of an individual is evaluated only with respect to the current population and its nondominance is determined by comparing against the remaining ones. A rank $1 + r_i$ is assigned to an individual i reflecting its domination by r individuals. As a result, all the nondominated individuals are assigned a rank of one. In R, the nondominated individuals carry the lowest (best) rank and the dominated individuals are with the highest (worst) rank. A fitness value based on this rank is then assigned to individuals

$$fitness_i = 1/(1 + r_i) \tag{8}$$

where r_i is the domination count (rank) of the subgraph individual S_i in the population R.

Step 7: Select N best individuals (as a new parent population P) from R according to the fitness. This selection also uses the idea of niching [20] to preserve the diversity of the new population P along the approximate Pareto front. The niching updates the fitness of individuals in R based on their proximities in the objective space. To this end, the sharing function value for two individuals i and j with the same rank is computed as follows:

$$sh(c_{ij}) = \begin{cases} 1 - \left(\frac{c_{ij}}{\sigma_{\text{share}}}\right)^a & \text{if } c_{ij} \leqslant \sigma_{\text{share}} \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

where σ_{share} is the maximum distance allowed between i and j individuals to become a member of a niche, and a is a positive scaling factor typically less than one. c_{ij} is the normalized distance between individuals i and j with the same rank

$$c_{ij} = \left[\sum_{l=1}^{d} \frac{\left(f_i^l - f_j^l\right)^2}{f_{\max}^l - f_{\min}^l}\right]^{1/2} \tag{10}$$

where $f_{\min}^l$ and $f_{\max}^l$ are the minimum and maximum objective function values of the lth objective, in the current iteration, respectively. f_i^l and f_j^l are objective function values of i and j individuals. The niche count of individual i is calculated by summing the sharing function

$$nc_i = \sum_{j=1}^{|R|} sh(c_{ij}) \tag{11}$$

finally, the fitness for individual i is adjusted by the niche count

$$fitness_i = fitness_i / nc_i \tag{12}$$

and the best N individuals from R are selected based on this updated fitness values.

Step 8: Go to step 2, if termination criterion is not satisfied.
Step 9: Report the nondominated subgraphs in PA.

MOEP-NS: This algorithm for subgraph discovery was proposed in [57] and can be described in the following steps:

Step 1: Initialize the population P of N subgraphs.
Step 2: Create a population Q of child subgraphs using mutation on each parent subgraph in P.
Step 3: Combine the two populations $R = P \cup Q$.
Step 4: Assign fitness to subgraphs in R using the nondominated sorting procedure [16] as follows. Individuals in R receive a rank value based upon their degree of nondominance. The nondominance among the individuals is determined by comparing all the subgraphs to each other. A rank one individuals are obtained when the nondominance is evaluated on the current population R. To obtain rank two individuals, discard temporarily all the ranked (i.e., rank one) individuals in R and then evaluate the nondominance on this reduced population R. Thus, to obtain rank r individuals, first discard temporarily all the ranked (i.e. rank 1, 2, ..., $r-1$) individuals from the population R and then evaluate the nondominance on the reduced population R. As a result, all nondominated subgraphs are assigned a rank one, while the dominated ones are penalized by assigning rank $1 + r$ according to the region of the tradeoff surface. In the population R, individuals that are nondominant are assigned with the lowest (best) rank and individuals that are highly dominated are assigned highest (worst) rank. A minimization of fitness function is then assumed

$$fitness_i = r_i \tag{13}$$

where r_i is the rank of the subgraph individual S_i in the population R.
Step 5: Create a new population P by selecting among individuals in R according to minimization of fitness function. To select N subgraphs, first choose the set of rank one subgraphs in R. If the size of this set is smaller than N then choose the set of rank two subgraphs, and so on. Consider that the set of rank r is the last one that can be accommodated. In general, the total number of subgraphs belonging to rank 1 to r will be greater than N. To choose exactly N subgraphs, a crowding-distance assignment [16] is applied in the set of rank r subgraphs. The crowding-distance cd_i associated with the subgraph S_i is the absolute normalized distance between two subgraphs S_j and S_k on either side of the subgraph S_i with the same rank r

$$cd_i = \sum_{l=1}^{d} \frac{f_j^l - f_k^l}{f_{\max}^l - f_{\min}^l} \tag{14}$$

where $f_{\max}^l$ and $f_{\min}^l$ are the maximum and minimum objective function values of the lth objective, respectively. f_j^l and f_k^l are the objective function values of the subgraphs S_j and S_k, respectively. Note that, the subgraphs with the minimum and maximum objective function values are assigned an infinite crowding-distance. For all other subgraphs, the crowding-distance is calculated using Eq. (14). The subgraph set with rank r is sorted in the descending order of the crowding-distance values and is then pruned at a point to have exactly N subgraphs in the population P.
Step 6: Go to Step 2, if termination criterion is not satisfied.
Step 7: Report the nondominated subgraphs in P.

MOEP-SO: This MOEP algorithm is different from MOEP-NS in two ways [56]: (a) creating the new population from R, and (b) use of an external population PA for storing nondominated solutions. Here, an application of MOEP-SO for subgraph discovery can be described in the following steps as:

Step 1: Initialize the population P of N subgraphs, and an external archive $PA = \emptyset$.
Step 2: Update archive PA using nondominated solutions in P.
Step 3: Create a population Q of child subgraphs from parent subgraphs in P by mutation.
Step 4: Update archive PA using nondominated subgraphs in Q.
Step 5: Combine the two populations $R = P \cup Q$.
Step 6: Create a new population from R using summation of objectives for diversified selection [46] as follows. For each objective function find its range from the maximum and minimum objective values, and evenly divide the range of the objective space into some 100 bins (or grids). For each individual a fitness (or rank) is assigned equal to the summation of grid numbers that holds each of the objective values of the individual. The diversified selection constitutes two sets of individuals, preferential set and backup set. To obtain the preferential set corresponding

to each objective function scan some 80 or 90 percent of the objective space divided into 100 bins. For each scanned bin an individual with the lowest fitness value is chosen. The backup set is obtained from the individuals that are not part of the preferential set. The new population is obtained from the preferential set. If the size of the preferential set is greater than the population size, the new population P is created by random selection of individuals from the preferential set. If the size of the preferential set is smaller than N then the remaining individuals are selected from the backup set according to the lowest fitness values.

Step 7: Go to Step 2, if termination criterion is not satisfied.

Step 8: Report the nondominated subgraphs in *PA*.

4. Experimental results

The performance of the three MOEP algorithms for the proposed multiobjective subgraph mining task (as defined in Section 2.5) is analyzed by means of various unary and binary metrics [9,70,71], and visual representations of the obtained PF approximations. For comparison purposes, we also apply single-objective Subdue [10,13] and EP-Subdue [2] using three different objective functions to produce aggregated PFs on several graph datasets (see Section 4.2). Besides, the two variants of MOSubdue [58] are applied for a broader performance study. All the considered methods have been implemented in C, and all experiments have been performed on an Intel Core Quad at 2.66 GHz, with 4 GB RAM, running CentOS 5.5.

4.1. Graph-represented datasets used

A total of 10 datasets have been employed. Table 2 summarizes a few characteristics of these datasets, such as the number of nodes, the number of edges, etc. The datasets are of different sizes. Besides, these datasets consist of varying degrees of nodes and unique labels. For example, the smallest size dataset *www2* consists of only four graphs while the biggest size dataset *random2* consists of 200 graphs. The number of nodes ranges from a small number of 500 in the *shapes* dataset to a large one of 19,253 in the *scientograms73* dataset. The number of unique nodes ranges from 7 in the *random1* dataset to 1156 in the *www2* dataset. The first two datasets, *random1* and *random2*, were synthetically generated using the random graph generator available at Subdue's website.[2] A brief description of the remaining datasets is provided as follows.

Shapes is a synthetically constructed geometrical example of structural data in the classical Subdue's study domain [10]. Fig. 2 shows an example of a graphical representation of the input shapes data. This dataset consists of 100 different graphs with a total of 500 nodes, 400 edges, and 8 unique labels. The true Pareto set (which is known for this simple domain as it has been computed in an exhaustive way) contains 12 different nondominated subgraphs out of which 7 are distinct in the objective vector space.

WWW is a dataset available online at Subdue's website.[1] The data were extracted from real World Wide Web pages and were transformed to labeled graphs using a web robot [22]. In this work, we have considered the ProfStu graph dataset which was generated using professor and student web sites. The information these graphs contain is hyperlink structure and page's content. ProfStu dataset consists of 47 graphs, which are quite complex containing several unique node labels. We consider this real-world data as a challenging way to illustrate our MOEP framework for GBDM. Instead of dealing with each of the huge graphs individually as done in [22], we created two different datasets, namely *www1* and *www2*, where *www1* is composed of five graphs numbered 6, 19, 25, 43, and 45; and *www2* consists of four graphs numbered 15, 27, 38, and 46. For these datasets, the true Pareto sets are not known due to large complexity and real-world nature. For comparison purposes, we have used a pseudo-optimal Pareto set, which is obtained as a fusion of all the nondominated sets achieved by each of the applied algorithms during the simulations.

Scientograms [48] is a database built following De Moya-Anegón et al.'s methodology [62] to design visual science maps (scientograms) for huge scientific publications collections. The rough considered data have been extracted from the Scimago Journal & Country Rank portal[3] and comprise a set of 36 millions documents indexed in Elsevier Scopus from 1996 to 2008 over 73 countries [62]. The nodes of the graphs correspond to Elsevier SCOPUS-SJR[4] co-citation categories. Only the salient relationships between categories are kept, capturing the essential underlying intellectual structure of the studied scientific domain, using the Pathfinder social networks algorithm [47] to prune the graphs. Recently, this database has been extensively analyzed in [48] to propose an automatic Subdue-based approach for the identification and the comparison of scientific structures within scientograms. In our experimental study, we have used four datasets compiled for the countries United States (*US*), United Kingdom (*UK*), Japan, and Germany over the period of 1996–2005, and with the *scientograms73* dataset combining the scientograms of 73 countries for the publication year 2005. The optimal Pareto sets are not known for these datasets, so we rely on the pseudo-optimal Pareto sets.

[2] http://ailab.wsu.edu/subdue/datasets/subgen.tar.gz.
[3] http://www.scimagojr.com/.
[4] http://www.scopus.com.

Table 2
Description of different graph-represented datasets used.

Dataset	#Graphs	#Nodes	#Edges	#Unique labels	MOEP run time (s)
random1	100	2954	3009	7	100
random2	200	5876	6015	7	145
shapes	100	500	400	8	1
www1	5	832	885	511	450
www2	4	2178	2539	1156	625
US	10	2762	2769	294	2000
UK	10	2732	2748	292	1300
Japan	10	2635	2680	278	3100
Germany	10	2676	2702	284	900
scientograms73	73	19,253	19,709	296	265

4.2. Nondominated subgraphs extraction using single-objective Subdue and EP-Subdue methods

On each dataset, both Subdue and EP-Subdue independently apply a single-objective search per objective function MDL (Eq. (1)), support (Eq. (6)), and size (Eq. (7)). The output of the algorithm was set to return a maximum of 100 best subgraphs. The three different outputs of the algorithm were aggregated and a nondominated set of subgraphs using nondominance (Eq. (3)) was produced as the final output of the algorithm. The nondominated set was restricted to contain a maximum of 100 subgraphs.

An implementation of the Subdue algorithm is publicly available at Subdue's website.[5] In this work, we have only modified the subgraph evaluation function of the Subdue algorithm to accommodate the support objective. The algorithm parameter *beam-width* was set equal to 5 after a preliminary experimentation. Subdue performs a deterministic search and thus it is executed once on each dataset.

We have implemented a single-objective EP-Subdue method [2,39] as described in Section 2.4 which considers two additional objectives the support and the size. EP-Subdue was run with the following parameters: population = 100, output best subgraphs = 100 per objective function. EP-Subdue performs a stochastic search and hence executed 10 times independently on each dataset.

4.3. Parameter settings and multiobjective metrics

MOSubdue: We have applied both MOSubdue variants. MOSubdue-I is a deterministic approach and thus it is applied once. While, MOSubdue-II is a stochastic search and thus it has been run 10 times independently on each dataset. Both the variants of MOSubdue were applied with the following parameter settings: *beam-width* = 5, size of output list = 100.

MOEP Methods: For MOEP methods the population was set to 100. The output of MOEP-NS was the nondominated subgraphs in the population when the termination criterion was met. On the contrary, MOEP-SO and MOEP-ND have reported the nondominated subgraphs stored in the external Pareto archive (*PA*). The size of *PA* was set to 100. The three MOEP methods have been run 10 times independently on each dataset. In MOEP-ND, the niche count computation needs setting two parameters σ_{share} and a in Eq. (11). After preliminary experimentation, these parameters were set to $\sigma_{share} = 0.005$ and $a = 1$.

To have a fair comparison between the EP methods and MOSubdue-II, the three MOEP variants and EP-Subdue have used a fixed run time as given in Table 2. This run time was determined from the average run time of 10 different executions of MOSubdue-II for each dataset.

Note that on all the datasets used, none of the methods in any of their executions could produce a number of nondominated subgraphs that surpass the maximum archive size limit of 100. Nevertheless, the single-objective Subdue and EP-Subdue, and MOSubdue methods used the crowding distance measure [16] to prune the nondominated set. However, when datasets have a large number of nondominated and repetitive subgraphs, a larger limit on the archive size could be used.

Due to the nature of MO problems, multiple performance indexes (classically called multiobjective metrics) should be used for comparing the performance of the different algorithms [9,70,71]. In our experimental analysis, a unary and a binary performance metric are used. As regards the former type, we use the hypervolume ratio (*HVR*) to compare the different Pareto set approximations taking the pseudo-optimal PF as a reference. For a Pareto set approximation, the *HVR*-metric value is better when it tends to one. For pair-wise evaluation of MO algorithms based on dominance, we have considered the binary *C* metric [70] that compares the Pareto set approximations in pairs. The *C*-metric is also better when it tends to one. Further, considering the stochastic nature of EAs, we apply a statistical test proposed in [54] about the probability that algorithm *A* dominates the algorithm *B* based on the binary $I_\epsilon(A, B)$-metric [71]. The three considered metrics are described in the Appendix.

[5] http://ailab.wsu.edu/subdue/software/subdue-5.2.1.zip.

4.4. Experimental analysis

Table 3 presents the mean and standard deviation of the *HVR*-metric values of the PF approximations achieved by the different algorithms for each dataset.

Fig. 4 shows the assessment of different algorithms in pairs using the *C*-metric. For an ordered algorithm pair (*A*, *B*), there is a sample of 10 *C*-metric values according to the 10 runs performed. Each value is computed on the basis of nondominated sets achieved by *A* and *B* with the same initial population. Note that, in case of deterministic algorithms (Subdue and MOSubdue-I) a single run was performed on a dataset. Here, box-plots are used to visualize the distribution of these samples. Besides, notice that, the two worst performing algorithms (the two single-objective methods, Subdue and EP-Subdue) are not included in the *C*-metric box-plots collected in Fig. 4. This is due to the fact that their poor results made the scale become so large just for the two, thus making very difficult to distinguish visually the differences among the remaining better performing algorithms. Figs. 5 and 6 show the plots of the PF approximations produced by the different algorithms on the *www2* and *scientograms73* datasets. The plots corresponding to Subdue-I and MOSubdue-I are the approximations generated by the

Table 3
HVR-metric values for the nondominated subgraph sets found by different methods. The numbers in the parentheses represent the standard deviation.

Dataset	Subdue	EP-Subdue	MOSubdue-I	MOSubdue-II	MOEP-NS	MOEP-SO	MOEP-ND
random1	0.9552(–)	0.8724(0.02)	0.9536(–)	0.9623(0.00)	**0.9721(0.01)**	0.9708(0.01)	0.9614(0.01)
random2	0.9663(–)	0.8626(0.02)	0.9747(–)	**0.9795(0.01)**	0.9163(0.02)	0.9723(0.01)	0.8675(0.03)
shapes	**1.0000(–)**	0.9920(0.00)	**1.0000(–)**	0.9954(0.01)	**1.0000(0.00)**	**1.0000(0.00)**	**1.0000(0.00)**
www1	0.7788(–)	0.8467(0.06)	0.8391(–)	0.9899(0.03)	0.9663(0.01)	0.9969(0.01)	**1.0000(0.00)**
www2	0.7567(–)	0.5939(0.07)	0.7432(–)	**0.9162(0.03)**	0.5824(0.03)	0.8790(0.06)	0.8054(0.10)
US	0.6114(–)	0.5796(0.15)	0.9673(–)	0.9013(0.11)	0.9740(0.03)	**0.9791(0.03)**	0.9741(0.03)
UK	0.6318(–)	0.4123(0.10)	0.7302(–)	0.7635(0.18)	**0.9326(0.04)**	0.9316(0.05)	0.9217(0.03)
Japan	0.6429(–)	0.6030(0.09)	0.8021(–)	0.9850(0.03)	0.9857(0.01)	**0.9878(0.01)**	0.9857(0.01)
Germany	0.7406(–)	0.5204(0.11)	0.8083(–)	0.8920(0.06)	**0.9501(0.04)**	0.9452(0.03)	0.9329(0.03)
scientograms73	0.8096(–)	0.5067(0.06)	0.7775(–)	**0.8336(0.01)**	0.8072(0.05)	0.8026(0.07)	0.7940(0.04)
Average	0.7893(0.14)	0.6770(0.20)	0.8596(0.10)	0.9219(0.08)	0.9087(0.13)	**0.9465(0.06)**	0.9336(0.08)

Bold represent the best results obtained on the dataset.

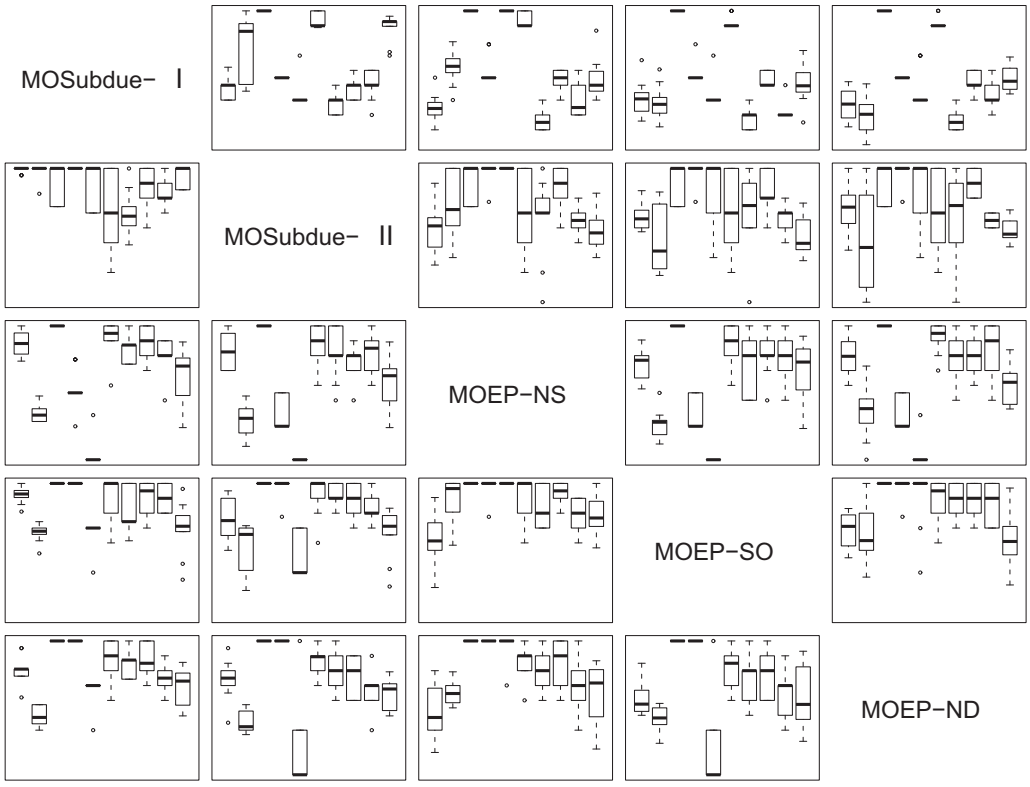

Fig. 4. Box-plots based on the *C*-metric computed for the best methods considered. Each rectangle contains 10 box-plots representing the distribution of the *C*-metric values for a certain ordered pair of algorithms; the leftmost box-plot relates to *random1* dataset, the rightmost to *scientograms73* dataset.

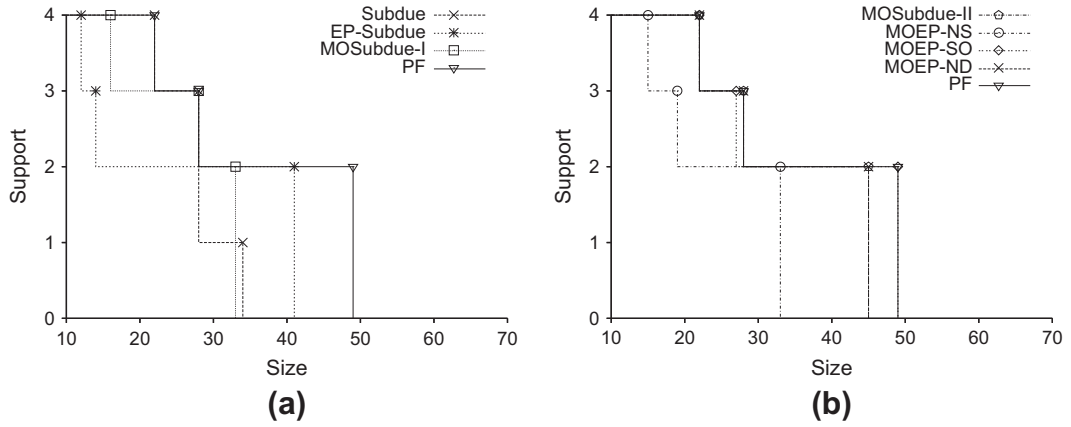

Fig. 5. The aggregated PF approximations produced by the different algorithms on *www2*. The pseudo-optimal PF is also shown as a reference.

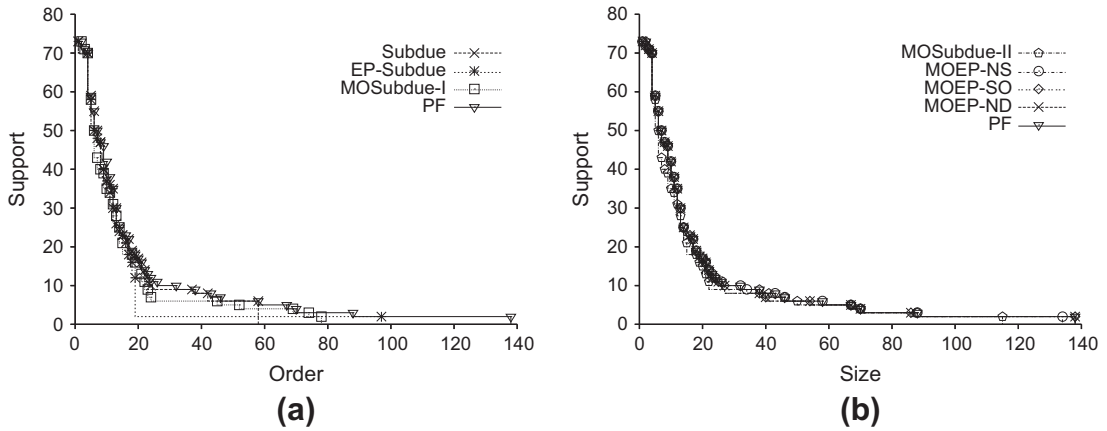

Fig. 6. The aggregated PF approximations produced by the different algorithms on *scientograms73*. The pseudo optimal PF is also shown as a reference.

single run of the algorithm, while those corresponding to the remaining methods are the aggregation of the results of 10 different runs. The fused outputs of stochastic methods are only used for graphical representation. For the sake of a better visual representation, the PF plots are grouped into two figures per dataset. They are grouped according to the average *HVR*-metric values on the 10 datasets, considering a figure for the worst performing and another for the best performing algorithms.

We can make some remarks in view of the obtained results. First, the *HVR*-metric values in Table 3 indicate that the single-objective algorithms (Subdue and EP-Subdue) have produced the worst approximations to the reference PFs on all the datasets as compared to those achieved by the multiobjective algorithms, MOSubdue and MOEP-based methods. The only exception is the small size *shapes* dataset. This conclusion can be further evident from the average values of *HVR*-metric over all the datasets, which are much lower for Subdue (0.7893) and EP-Subdue (0.6770) as compared to that of any multiobjective search method. This suggests the incorporation of multiobjective search strategy enables the algorithm to explore more efficiently the multiobjective subgraph solution search space.

Comparing the results between MOSubdue methods, Table 3 shows that MOSubdue-II has clearly been the best. MOSubdue-II has produced better results than MOSubdue-I on eight datasets. The exceptions are only the *shapes* and *US* datasets. The average of the *HVR*-metric values for MOSubdue-II is 0.9219, which is significantly higher than that of 0.8596 for MOSubdue-I. Fig. 4 also shows that the PF approximations by MOSubdue-II have achieved more coverage over those by MOSubdue-I. Further, the graphical representations of the PFs obtained by MOSubdue-I and MOSubdue-II as shown in Figs. 5 and 6 indicate that MOSubdue-II was able to produce wider or similar spread of solutions over the reference PF when compared to that obtained by MOSubdue-I. The better performance of MOSubdue-II was mainly due to the diversified subgraph solution selection during the beam search.

From Table 3, three MOEP methods have produced the best quality PFs on seven datasets. *www2* is the only dataset where MOEP-NS shows a clearly disappointing performance. The average *HVR* value corresponding to MOEP-SO is the highest (0.9465). Further, the binary *C*-metric in Fig. 4 indicates that the PF approximations produced by MOEP-SO have more

coverage to that produced by the other two MOEP methods. The aggregated PF plots of all the MOEP methods in Fig. 5 indicate that the differences are more significant in the aggregated PFs on the *www2* dataset, where the bad performance of MOEP-NS is clearly identified. The same stands for *scientograms73* in Fig. 6. Hence, the obtained differences are mainly due to the dispersion of the results in the 10 different runs.

Based on the results of the different multiobjective performance metrics, we can conclude that MOEP-SO is the best performer, followed by MOEP-ND, MOSubdue-II and MOEP-NS, respectively. Even so, as some of the executions of MOEP-SO were dominated by MOSubdue-II and *vice versa*, we want to assess the actual improvement MOEP-SO has obtained over MOSubdue-II. To do so, we perform a statistical analysis of multiple executions of MOEP-SO and MOSubdue-II. For this purpose, we apply a statistical test on the results of the $I_\epsilon(A, B)$-metric pair-wise comparison to study the extent to which, on average, the algorithm A is better than the algorithm B. Table 4 reports the obtained p-values corresponding to a pair-wise comparison between the three best performing methods, MOEP-SO, MOEP-ND, and MOSubdue-II, on all the datasets.

According to p-values < 0.05 (i.e., with 95% of confidence), MOEP-SO has significantly dominated MOSubdue-II and MOEP-ND on four datasets. As against, it has been dominated with significant difference by MOSubdue-II on a single dataset (*www2*) and by MOEP-ND on one dataset (*www1*). Comparison between MOSubdue-II and MOEP-ND shows that both algorithms have dominated each other on three and two datasets, respectively. This clearly says that MOEP-SO is the best algorithm in our experimental study, while MOSubdue-II and MOEP-ND have shown a similar performance.

The explanation for the latter good performance lies in the fact that the proposed MOEP framework is a global search approach. At any generation, the MOEP-based GBDM method maintains a population of solutions belonging to different layers of the subgraph lattice. This helps exploring more efficiently the multiobjective subgraph lattice space and thus producing more diverse solutions. On the contrary, MOSubdue performs a beam search with no backtracking in the subgraph lattice space. At any generation, the beam search explores a single layer of the subgraph lattice, and for the next generation it comprises the solutions mostly belonging to that layer. Thus, the depth exploration applied by the beam search, although complemented by width search, may miss out exploring other regions of the multiobjective subgraph lattice space due to the fact that it discards nonpromising solutions at early stages of the search and it never backtracks.

Nevertheless, the evolutionary search has performed somewhat poorly on the individual runs on *www2*. While MOEP-SO and MOEP-ND are generally the best performing methods for all the remaining datasets, they are clearly outperformed by MOSubdue-II on *www2*. On this dataset even the single-objective Subdue and EP-Subdue, and the deterministic MOSubdue-I have achieved better values of the *HVR*-metric than that obtained by MOEP-NS (Table 3). We performed an in depth analysis of the subgraph generation process of MOEP on this dataset in order to understand a possible reason behind the inferior performance. The *www2* dataset of four graphs contain not only several unique node labels (1156) but also many repeated ones. At the beginning of the search process, a parent subgraph has several repetitive instances in a graph of the dataset. As against, the definition of the objective support assumes just one occurrence (and no repetition) in any graph of the dataset. The different repetitive instances of the subgraph bring huge redundancy in the mutation operation for a child generation. The current definition of the support objective fails to take into account this redundancy during subgraph selection. Thus, a new definition of the support objective is needed to apply MOEP-SO efficiently and effectively in such scenarios. This problem of redundancy is somewhat also observed on synthetic datasets *random1* and *random2*. Note that, in the real-world scientogram datasets, a parent subgraph has no repetitive instances in a graph of the dataset and thus there is no such additional redundancy in the mutation operation for a child generation. As a result, on the scientogram datasets, the MOEP methods have produced consistently superior performance.

In this work, MOEP-based GBDM has been applied on sets of connected relational graphs with or without cycles and directed or undirected edges. In the multiobjective GBDM problem formulation, the support of a subgraph is evaluated on a set of graphs and the maximum support of the subgraph is estimated to be equal to the number of graphs in the dataset. Even so, the proposed MOEP framework can easily be applied on a single huge graph to find the nondominated subgraphs by just changing the support definition (without requiring any change in the solution methodology). This is because an individual in the MOEP population is a subgraph with all its instances in the graph data of a single huge graph or a set of graphs.

Table 4
p-values for different pairs of the three best algorithms.

	(A, B)	(B, A)	(A, C)	(C, A)	(B, C)	(C, B)
random1	1.0000	1.0000	0.0840	0.9362	0.0008	0.9994
random2	1.0000	1.0000	0.0840	0.9362	0.0168	0.9873
shapes	7.969e−06	1.0000	1.0000	1.0000	1.0000	7.969e−06
www1	0.5290	0.5290	1.0000	7.969e−06	1.0000	7.969e−06
www2	0.9998	0.0002	0.0002	0.9999	5.1704e−05	1.0000
US	0.0071	0.9943	0.1290	0.8867	0.6764	0.3541
UK	0.0032	0.9975	0.0029	0.9977	0.8985	0.1164
Japan	0.0558	0.9525	0.0015	0.9988	0.0725	0.9383
Germany	0.0029	0.9978	0.0074	0.9944	0.8643	0.1841
scientograms73	1.0000	1.0000	0.0840	0.9362	1.0000	1.0000

A = MOEP-SO, B = MOSubdue-II, and C = MOEP-NS algorithms.

The existing subgraph mining approaches operate by using simple user-defined thresholds on the mined subgraphs. However, not all the subgraphs uncovered by this approach are important, especially when mined at high support threshold resulting into many uninteresting small size subgraphs. Facilitating the use of such simple constraints make it difficult for the user to discover any tradeoff between the support and the size of mined subgraphs to call them interesting patterns. As against, a new multiobjective subgraph mining approach based on EP is proposed here to support the user specified preferences (or objectives) regarding the characteristics of the mined subgraphs. The extracted subgraphs subject to the tradeoff between the objectives facilitate the user to select the important patterns.

Finally, we have also applied gSpan[6] [65], an existing exact subgraph mining algorithm, on the datasets given in Table 2. The gSpan algorithm was very fast to completely mine the first three datasets (*random1*, *random2*, and *shapes*) in Table 2 containing small size graphs with the average size of 25 nodes and 6 unique node labels. Nevertheless, the algorithm failed to perform a complete search on the remaining seven datasets even after spending over two days on each dataset. For example, gSpan's aborted run with 10% of frequency threshold has mined over 5 millon subgraphs on *scientograms73* dataset of 73 graphs and average graph size of 264 nodes. The biggest mined subgraph was of size 28 and support of seven, which is dominated by the solutions obtained all the MOEP and MOSubdue methods. This experimentation clearly highlights that MOEP is useful for domains where complete subgraph search is not feasible, and in cases where the number of possible maximal subgraphs is relatively large.

5. Conclusions and future work

We have successfully shown the application of MOEP for multiobjective subgraph mining. To our knowledge, this is the first inclusive study of evolutionary search for this data mining task. The need of global search in the multiobjective subgraph lattice space and the obtained experimental results highlight the suitability of the proposed framework for producing good PF approximations. Three different MOEP methods have been implemented. Comparing to single-objective Subdue and EP-Subdue, and MOSubdue methods on the 10 different datasets considered, the three MOEP methods have shown a good performance and two of them have achieved a substantial performance gain. This is because the use of a population allows us to explore different regions of the multiobjective subgraph lattice in a single run. Overall, the MOEP-SO algorithm has been the best performer in the developed experiments.

Different lines for future works arise from this contribution. The present MOEP approach has shown very good performance on different datasets overall, but has faired marginally inferior in the case of synthetic datasets and somewhat poorly in the case of *www2* dataset. To this end, a new definition of the support objective is required to handle the subgraph selection pressure in the presence of subgraphs with repetitive instances in a graph of the input dataset.

Further, in a future study, we consider building a more general EMO framework by developing a crossover operator for subgraph generation. For child creation, the crossover operation involves information sharing by parent individuals which helps the co-operative evolution of the population. We will explore an approach that joins two size-k frequent subgraphs when they have the same size-$(k-1)$ subgraph. Here, graph *size* means the number of nodes in a graph. This will generate two candidate subgraphs with the common size-$(k-1)$ subgraph and the additional two nodes from the two size-k patterns. A similar approach has been applied in apriori-based algorithms [28,34] with edge (or node)-based candidate generation strategy. However, this approach has considerable overhead when two size-k frequent subgraphs are joined to generate size-$(k+1)$ subgraph candidates [28,34]. Thus, obtaining a good design for a subgraph crossover operator is an open challenge.

In addition, the proposed MOEP framework is based on a pure general purpose multiobjective subgraph search. Thus, it is able to deal with several objectives (two in this contribution) which could be generically customized by the user to deal with different GBDM tasks as long as they can be formulated in a simple way [58].

Acknowledgments

This work has been partially supported by the Spanish Ministry of Science and Innovation (MICINN) under Project TIN2009-07727, including EDRF fundings. The first author acknowledges the partial support received from MICINN under the Juan de la Cierva Programme JCI-2010-07626.

Appendix A

In the case of multiobjective optimization, comparing the results obtained by two algorithms is substantially more complex than in single-objective optimization. To do so, we should compare two sets of solutions where some solutions in either set may be dominated by solutions in the other set, while others may be incomparable. Several unary and binary performance indicators are proposed in the EMO literature [9,15,70,71] in order to compare the outcomes of multiobjective optimizers in a quantitative and qualitative manner. It is widely recommended to consider the use of both unary and binary indexes.

[6] http://www.cs.ucsb.edu/xyan/software/gSpan.htm.

Hypervolume is a unary index that measures the volume enclosed by the PF approximation X' with respect to a reference point. For a two-dimensional objective vector, hypervolume is the summation of the area covered by each member of the Pareto front. In the case of a maximization problem, as ours, we define a reference point as (0, 0). In our experiments, we have used the hypervolume ratio (*HVR*) [9]. *HVR* measures both diversity and closeness of X' to the true Pareto set and it is calculated as:

$$HVR = \frac{H_1}{H_2} \tag{15}$$

where H_1 and H_2 are the volume of the Pareto set approximation X', and of the true Pareto set (or the pseudo-optimal Pareto set in case the latter is not known), respectively. A pseudo-optimal Pareto set, i.e. an estimation of the true Pareto set, is obtained by fusing all the Pareto sets generated for the problem by any algorithm in any run. A value of *HVR* equal to one represents that the PF approximation and the true PF are equal.

The unary indexes allow us to determine the absolute, individual quality of the approximation, but they cannot be used for direct comparison purposes. Alternatively, binary indexes have been proposed for comparing the Pareto set approximations obtained by different multiobjective algorithms. In this work, we have used the following binary metric.

The set coverage (or *C*-metric) compares a pair of nondominated sets by computing the fraction of each set that is covered by the other [70]:

$$C(X', X'') = \frac{|\{\forall S'' \in X''; \exists S' \in X' : S' \succeq S''\}|\leftarrow}{|X''|} \tag{16}$$

where $S' \succeq S''$ indicates that the subgraph S' dominates or cover the subgraph S'' in a maximization problem. A value of $C(X', X'') = 1$ means that all the subgraphs in X'' are dominated or covered by the subgraphs in X'.

The statistical test in [54] is applied to test if there is significant difference in the PF approximations of two algorithms when compared in pairs. For this purpose, all the executions of both algorithms are compared using the binary $I_\epsilon(A, B)$-metric [71]. Let the probability $p_A(B)$ is 1 if ($I_\epsilon(A, B) \geqslant 1$ and $I_\epsilon(B, A) < 1$) otherwise 0. For 10 different executions of the algorithms A and B, the full set of comparisons produce two vectors Γ_A and Γ_B:

$$\begin{aligned} \mathbf{p}_{\mathbf{A}j}(B) &= \sum_{i=1}^{10} p_A(B_i), \quad j = 1, \ldots, 10 \\ \mathbf{p}_{\mathbf{B}j}(A) &= \sum_{i=1}^{10} p_B(A_i), \quad j = 1, \ldots, 10 \end{aligned} \tag{17}$$

$$\begin{aligned} \Gamma_A &= \mathbf{p}_{\mathbf{A}1}(B), \quad \mathbf{p}_{\mathbf{A}2}(B), \ldots, \mathbf{p}_{\mathbf{A}10}(B) \\ \Gamma_B &= \mathbf{p}_{\mathbf{B}1}(A), \quad \mathbf{p}_{\mathbf{B}2}(A), \ldots, \mathbf{p}_{\mathbf{B}10}(A) \end{aligned} \tag{18}$$

Γ_A and Γ_B can be seen as samples of a random variable and can be used to get the fraction of times the outputs of one algorithm dominates the other. With this, we can estimate the outputs of two algorithms are significantly different, i.e., the expectation of the algorithm A is greater than the algorithm B ($E(\Gamma_A) > E(\Gamma_B)$), and *vice versa* against the assumption that the outputs of A and B are the same, i.e., ($E(\Gamma_A) = E(\Gamma_B)$). Γ_A and Γ_B values have nonGaussian distribution therefore we apply a Wilcoxon test setting a null hypothesis ($E(\Gamma_A) = E(\Gamma_B)$) and an alternate hypothesis ($E(\Gamma_A) > E(\Gamma_B)$).

References

[1] C. Aggarwal, H. Wang (Eds.), Managing and Mining Graph Data, Springer, 2010.
[2] S. Bandyopadhyay, U. Maulik, D.J. Cook, L.B. Holder, Y. Ajmerwala, Enhancing structure discovery for data mining in graphical databases using evolutionary programming, in: Int. Conf. Florida Artificial Intelligence Research Society (FLAIRS), 2002, pp. 232–236.
[3] C. Borgelt, M. Berthold, Mining molecular fragments: finding relevant substructures of molecules, in: Proc. IEEE Conf. Data Mining (ICDM'02), 2002, pp. 51–58.
[4] C. Carmona, P. González, M. del Jesus, F. Herrera, NMEEF-SD: non-dominated multi-objective evolutionary algorithm for extracting fuzzy rules in subgroup discovery, IEEE Trans. Fuzzy Systems 18 (2010) 958–970.
[5] A. Chandra, X. Yao, Evolving hybrid ensembles of learning machines for better generalisation, Neurocomputing 69 (2006) 686–700.
[6] V. Chankong, Y.Y. Haimes, Multiobjective Decision Making Theory and Methodology, North-Holland, Amsterdam, 1983.
[7] M. Coatney, S. Parthasarathy, MotifMiner: efficient discovery of common substructures in biochemical molecules, Knowledge & Information Systems 7 (2005) 202–223.
[8] C. Coello, S. Dehuri, S. Ghosh, Swarm Intelligence for Multi-objective Problems in Data Mining, Springer, 2009.
[9] C.A. Coello, G.B. Lamont, D.A.V. Veldhuizen, Evolutionary Algorithms for Solving Multi-objective Problems, Springer, Berlin, 2007.
[10] D. Cook, L. Holder, Graph-based data mining, IEEE Intelligent Systems 15 (2000) 32–41.
[11] D. Cook, L. Holder (Eds.), Mining Graph Data, Wiley, London, 2007.
[12] D. Cook, L. Holder, S. Su, R. Maglothin, I. Jonyer, Structural mining of molecular biology data, IEEE Engineering in Medicine & Biology Magazine 20 (2001) 67–74.
[13] D.J. Cook, L.B. Holder, Substructure discovery using minimum description length and background knowledge, Journal of Artificial Intelligence Research 1 (1994) 231–255.
[14] Ó. Cordón, E. Herrera-Viedma, M. Luque, Improving the learning of Boolean queries by means of a multiobjective IQBE evolutionary algorithm, Information Processing & Management 42 (2006) 615–632.
[15] K. Deb, Multi-objective Optimization using Evolutionary Algorithms, Wiley, Chichester, UK, 2001.

[16] K. Deb, A. Pratap, S. Agarwal, T. Meyarivan, A fast and elitist multiobjective genetic algorithm: NSGA-II, IEEE Transactions on Evolutionary Computation 6 (2002) 182–197.
[17] M. Deshpande, M. Kuramochi, N. Wale, G. Karypis, Frequent substructure-based approaches for classifying chemical compounds, IEEE Transactions on Knowledge & Data Engineering 17 (2005) 1036–1050.
[18] H. Dong, J. He, H. Huang, W. Hou, Evolutionary programming using a mixed mutation strategy, Information Sciences 177 (2007) 312–327.
[19] D. Fogel, System Identification through Simulated Evolution: A Machine Learning Approach to Modeling, Ginn Press, 1991.
[20] C. Fonseca, P. Fleming, Genetic algorithms for multiobjective optimization: formulation, discussion and generalization, in: Proc. 5th Int. Conf. Genetic Algorithms (ICGA93), 1993, pp. 416–423.
[21] T. Gal, T. Stewart, T. Hanne (Eds.), Multicriteria Decision Making: Advances in MCDM Models, Algorithms, Theory and Applications, Kluwer Academic, Dordrecht, 1999.
[22] J. Gonzalez, Empirical and Theoretical Analysis of Relational Concept Learning Using a Graph Based Representation, Ph.D. thesis, Department of Computer Science & Engineering, University of Texas at Arlington, USA, 2001.
[23] J. Handl, J. Knowles, An evolutionary approach to multiobjective clustering, IEEE Transactions on Evolutionary Computation 11 (2007) 56–76.
[24] H. Hu, X. Yan, Y. Huang, J. Han, X. Zhou, Mining coherent dense subgraphs across massive biological networks for functional discovery, Bioinformatics 21 (2005) i213–i221.
[25] J. Huan, W. Wang, J. Prins, Efficient mining of frequent subgraphs in the presence of isomorphism, in: Proc. IEEE Conf. Data Mining (ICDM'03), 2003, pp. 549–552.
[26] J. Huan, W. Wang, J. Prins, J. Yang, Spin: mining maximal frequent subgraphs from graph databases, in: Proc. ACM/SIGKDD Int. Conf. Knowledge Discovery & Data Mining (KDD'04), 2004, pp. 581–586.
[27] A. Inokuchi, T. Washio, H. Motoda, An apriori-based algorithm for mining frequent substructures from graph data, in: Proc. Euro Conf. Principles of Data Mining & Knowledge Discovery (PKDD'00), 2000, pp. 13–23.
[28] A. Inokuchi, T. Washio, K. Nishimura, H. Motoda, A fast algorithm for mining frequent connected subgraphs, Technical Report, IBM Research, Tokyo Research Laboratory, Tokyo, Japan, 2002.
[29] H. Ishibuchi, T. Yamamoto, Fuzzy rule selection by multi-objective genetic local search algorithms and rule evaluation measures in data mining, Fuzzy Set Systems 141 (2004) 59–88.
[30] C. Jiang, F. Coenen, M. Zito, A survey of frequent subgraph mining algorithms, Knowledge Engineering Review 28 (2013) 75–105.
[31] Y. Jin (Ed.), Multi-Objective Machine Learning, Springer-Verlag, New York, 2006.
[32] Y. Jin, B. Sendhoff, Pareto-based multi-objective machine learning: an overview and case studies, IEEE Transactions on Systems Man Cybernetics – C 38 (2008) 397–415.
[33] K. Khoo, P. Suganthan, Structural pattern recognition using genetic algorithms with specialized operators, IEEE Transactions on Systems Man Cybernetics – B 33 (2003) 156–165.
[34] M. Kuramochi, G. Karypis, An efficient algorithm for discovering frequent subgraphs, IEEE Transactions on Knowledge & Data Engineering 16 (2004) 1038–1051.
[35] A.J. Lee, Y.A. Chen, W.C. Ip, Mining frequent trajectory patterns in spatialtemporal databases, Information Sciences 179 (2009) 2218–2231.
[36] A. López-Herrera, E. Herrera-Viedma, F. Herrera, A study of the use of multi-objective evolutionary algorithms to learn Boolean queries: a comparative study, Journal of American Society Information Science & Technology 60 (2009) 1192–1207.
[37] B.T. Lowerre, The HARPY speech recognition system, Ph.D. thesis, Carnegie Mellon University, Pittsburgh, 1976.
[38] A. Martinez-Antonio, J. Collado-Vides, Identifying global regulators in transcriptional regulatory networks in bacteria, Current Opinion in Microbiology 6 (2003) 482–489.
[39] U. Maulik, Hierarchical pattern discovery in graphs, IEEE Transactions on Systems Man Cybernetics – C 38 (2008) 867–872.
[40] L. McCue, W. Thompson, C. Carmack, M.P. Ryan, J.S. Liu, V. Derbyshire, C.E. Lawrence, Phylogenetic footprinting of transcription factor binding sites in proteobacterial genomes, Nucleic Acids Research 29 (2001) 774–782.
[41] J.C. Meza, M. Yildirim, S.A. Masud, Multiobjective evolutionary programming algorithm and its applications to power generation expansion planning, IEEE Transactions on Systems Man Cybernetics – A 39 (2009) 1086–1096.
[42] S. Nijssen, J. Kok, A quickstart in frequent structure mining can make a difference, in: Proc. ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining (KDD'04), 2004, pp. 647–652.
[43] N. Nilsson, Artificial Intelligence: A New Synthesis, Morgan Kaufmann, San Francisco, 1998.
[44] A. Papadopoulos, A. Lyritsis, Y. Manolopoulos, SkyGraph: an algorithm for important subgraph discovery in relational graphs, Data Mining and Knowledge Discovery 17 (2008) 57–76.
[45] C. Pizzuti, Community detection in social networks with genetic algorithms, in: Proc. Genetic & Evolutionary Computation Conf. (GECCO'08), ACM, New York, NY, USA, 2008, pp. 1137–1138.
[46] B. Qu, P. Suganthan, Multi-objective evolutionary algorithms based on the summation of normalized objectives and diversified selection, Information Sciences 180 (2010) 3170–3181.
[47] A. Quirin, Ó. Cordón, V.P. Guerrero-Bote, B. Vargas-Quesada, F. de Moya-Anegón, A quick MST-based algorithm to obtain Pathfinder networks, Journal of American Society Information Science & Technology 59 (2008) 1912–1924.
[48] A. Quirin, Ó. Cordón, B. Vargas-Quesada, F. de Moya-Anegón, Graph-based data mining: a new tool for the analysis and comparison of scientific domains represented as scientograms, Journal of Informetrics 4 (2010) 291–312.
[49] J. Rissanen, Stochastic Complexity in Statistical Inquiry Theory, World Scientific Publishing Co., Inc., River Edge, 1989.
[50] R. Romero-Zaliz, I. Zwir, E. Ruspini, Generalized analysis of promoters (GAP): a method for DNA sequence description, in: C.A. Coello, G.B. Lamont (Eds.), Applications of Multi-Objective Evolutionary Algorithms, vol. 1, World Scientific Company, 2004, pp. 427–450.
[51] R.C. Romero-Zaliz, C. Rubio-Escudero, J.P. Cobb, F. Herrera, Ó. Cordón, I. Zwir, A multiobjective evolutionary conceptual clustering methodology for gene annotation within structural databases: a case of study on the gene ontology database, IEEE Transactions on Evolutionary Computation 12 (2008) 679–701.
[52] E. Ruspini, I. Zwir, Automated qualitative description of measurements, in: Proc. IEEE Conf. Instrumentation & Measurement Technology (IMTC'99), 1999, pp. 1086–1091.
[53] E. Ruspini, I. Zwir, Automated generation of qualitative representations of complex object by hybrid soft-computing methods, in: S. Pal, A. Pal (Eds.), Pattern Recognition: From Classical to Modern Approaches, World Scientific Company, 2001, pp. 453–474.
[54] L. Sánchez, J. Villar, Obtaining transparent models of chaotic systems with multi-objective simulated annealing algorithms, Information Sciences 178 (2008) 952–970.
[55] P. Shelokar, A. Quirin, Ó. Cordón, A multiobjective variant of the Subdue graph mining algorithm based on the NSGA-II selection mechanism, in: Proc. IEEE Conf. Evolutionary Computation (CEC'10), 2010, pp. 463–470.
[56] P. Shelokar, A. Quirin, Ó. Cordón, MOEP-SO: a multiobjective evolutionary programming algorithm for graph mining, in: Int. Conf. Intelligent System Design and Application (ISDA'11), 2011a, pp. 219–224.
[57] P. Shelokar, A. Quirin, Ó. Cordón, Subgraph mining in graph-based data using multiobjective evolutionary programming, in: Proc. IEEE Conf. Evolutionary Computation (CEC'11), 2011b, pp. 1730–1737.
[58] P. Shelokar, A. Quirin, Ó. Cordón, MOSubdue: a Pareto dominance-based multiobjective Subdue algorithm for frequent subgraph mining, Knowledge & Information Systems, in press. doi:http://dx.doi.org/10.1007/s10115-011-0452-y.
[59] S.Y. Shin, I.H. Lee, Y.M. Cho, K.A. Yang, B.T. Zhang, EvoOligo: oligonucleotide probe design with multiobjective evolutionary algorithms, IEEE Transactions on Systems Man Cybernetics – B 39 (2009) 1606–1616.

[60] K.C. Tan, E.F. Khor, T.H. Lee, Multiobjective Evolutionary Algorithms and Applications, Springer-Verlag, London, 2005.
[61] N. Vanetik, E. Gudes, S. Shimoy, Computing frequent graph patterns from semistructured data, in: Proc. IEEE Conf. Data Mining (ICDM'02), 2002, pp. 458–465.
[62] B. Vargas-Quesada, F. de Moya-Anegón, Visualizing the Structure of Science, Springer-Verlag, New York, Secaucus, 2007.
[63] P. Venkatesh, K. Lee, Multi-objective evolutionary programming for economic emission dispatch problem, in: Proc. IEEE PES General Meeting, 2008, pp. 1–8.
[64] Y.K. Wang, K.C. Fan, J.T. Horng, Genetic-based search for error-correcting graph isomorphism, IEEE Transactions on Systems Man Cybernetics – B 27 (1997) 588–597.
[65] X. Yan, J. Han, gSpan: graph-based substructure pattern mining, in: Proc. IEEE Conf. Data Mining (ICDM'02), 2002, pp. 721–724.
[66] X. Yan, J. Han, CloseGraph: mining closed frequent graph patterns, in: Proc. ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining (KDD'03), 2003, pp. 286–295.
[67] X. Yan, P. Yu, J. Han, Graph indexing: a frequent structure-based approach, in: Proc. ACM/SIGMOD Int. Conf. Management Data (SIGMOD'04), ACM, New York, NY, USA, 2004, pp. 335–346.
[68] X. Yan, X. Zhou, J. Han, Mining closed relational graphs with connectivity constraints, in: Proc. ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining (KDD'05), 2005, pp. 324–333.
[69] F. Zhu, X. Yan, J. Han, P. Yu, gPrune: a constraint pushing framework for graph pattern mining, in: Proc. Pacific–Asia Conf. Knowledge Discovery & Data Mining (PAKDD'07), 2007, pp. 388–400.
[70] E. Zitzler, L. Thiele, K. Deb, Comparison of multiobjective evolutionary algorithms: empirical results, IEEE Transactions on Evolutionary Computation 8 (2000) 173–195.
[71] E. Zitzler, L. Thiele, M. Laumanns, C. Fonseca, V. da Fonseca, Performance assessment of multiobjective optimizers: an analysis and review, IEEE Transactions on Evolutionary Computation 7 (2003) 117–132.
[72] I. Zwir, H. Huang, E. Groisman, Analysis of differentially-regulated genes within a regulatory network by GPS genome navigation, Bioinformatics 21 (2005) 4073–4083.
[73] I. Zwir, R. Romero-Zaliz, E. Ruspini, Automated biological sequence description by genetic multiobjective generalized clustering, in: F. Valafar (Ed.), Techniques in Bioinformatics and Medical Informatics, vol. 980, Annals of the New York Academy of Sciences, 2002, pp. 65–82.
[74] I. Zwir, D. Shin, D.A. Kato, K. Nishino, T. Kunihiko, F. Solomon, J. Hare, H. Huang, E. Groisman, Dissecting the PhoP regulatory network of *Escherichia coli* and *Salmonella enterica*, PNAS 102 (2005) 2862–2867.