Knowledge and Information Systems manuscript No. KAIS-2416R1

MOSubdue: A Pareto Dominance-based Multiobjective Subdue Algorithm For Frequent Subgraph Mining

Prakash Shelokar · Arnaud Quirin · Óscar Cordón

Received: Mar 23, 2010 / Revised: Feb 16, 2011 / Accepted: Oct 04, 2011

Abstract Graph-based data mining approaches have been mainly proposed to the task popularly known as frequent subgraph mining subject to a single user preference, like frequency, size, etc. In this work, we propose to deal with the frequent subgraph mining problem from multiobjective optimization viewpoint, where a subgraph (or solution) is defined by several user-defined preferences (or objectives), which are conflicting in nature. For example, mined subgraphs with high frequency are often of small size, and *vice-versa*. Use of such objectives in the multiobjective subgraph mining process generates Pareto-optimal subgraphs, where no subgraph is better than another subgraph in all objectives. We have applied a *Pareto-dominance approach* for evaluation and search subgraphs regarding to both proximity and diversity in multiobjective sense, which has incorporated in the framework of Subdue algorithm for subgraph mining. The method is called Multi-Objective subgraph mining by Subdue (MOSubdue), and has several advantages: i) generation of Pareto-optimal subgraphs in a single run, ii) selection of subgraph-seeds from the candidate subgraphs based on all objectives, iii) search in the multiobjective subgraphs lattice space, and iv) capability to deal with different multiobjective frequent subgraph mining tasks by customizing the tackled objectives. The good performance of MOSubdue is shown by performing multiobjective subgraph mining defined by two and three objectives on two real-life datasets.

Keywords Graph-based data mining · Frequent subgraph mining · Subdue · Gaston · Multiobjective graph-based data mining · Pareto-based multiobjective optimization · Evolutionary multiobjective optimization

1 Introduction

Graph-based data mining (GBDM) has been prevalently used in a wide range of application domains, such as computing communities [11, 31], subgraph discovery [7, 41, 48, 51], topic

Prakash Shelokar · Arnaud Quirin · Óscar Cordón
European Centre for Soft Computing, 33600-Mieres, Spain.
Dr. Óscar Cordón is also affiliated to the Department of Computer Science and Artificial Intelligence (DECSAI) and the Research Centre on Information and Communication Technologies (CITIC-UGR), University of Granada. 18071-Granada, Spain.
E-mail: {prakash.shelokar, arnaud.quirin, oscar.cordon}@softcomputing.es, ocordon@decsai.ugr.es

detection [38], attack detection [45], computing the number of triangles [46], clustering [27, 36], peta graph mining [23], etc. Recently GBDM has been recognized as one of the ten challenging problems in data mining research [50]. For the recent developments and comprehensive survey of this important and emerging topic the reader is referred to [1, 7].

GBDM approaches are characterized by representation of multi-relational data in the form of graphs. They have been extensively applied to the task popularly known as frequent subgraph mining. These approaches can be categorized into mathematical graph theory based approaches (such as, MoFa/MoSS [3], FSG [26], Gaston [32], gSpan [48], CloseGraph [49], gPrune [51]), greedy search based approaches (like Subdue [6] and GBI [29]), and kernel function based approaches [24]. All these approaches work by performing a search in the lattice of all possible subgraphs [12]. The underlying search process, which could either involve an exact exhaustive or approximate heuristic search, is usually guided by a single *objective*, which represents a unique and specific user *preference*. For example, mining subgraphs which are present in at least m graphs, or mining subgraphs which contain at least n nodes are typical choices.

The existing GBDM approaches applying such simple thresholds for frequent subgraph mining task have important limitations. For example, the number of mined subgraphs is large (respectively, few or nil) in the cases of weak (respectively, strict) thresholds [35]. Moreover, in real-life applications a user is generally interested in mining a graph-based repository using several objectives that are actually meaningful to her/him, which are often conflicting in nature [35]. For example, users prefer obtaining subgraphs with both high frequency and size values. Nevertheless, these objectives are conflicting as simpler descriptions are usually the most frequent ones and *vice versa*. In view of the reasons stated above, a GBDM methodology should not only rely on the optimization of a simple objective but also consider simultaneously additional, conflicting objectives to extract better defined concepts, which may be based on the size of the subgraph being explained, the number of retrieved subgraphs, and their diversity.

Towards dealing with the limitations of a simple single objective-based search, SkyGraph [35] has recently shown an application of skyline processing incorporating multiple objectives for subgraph mining. The skyline processing has been predominantly called as *Pareto-based optimization* in multiobjective optimization, which has been important for several applications involving multicriteria decision making [4,14]. Recently, Pareto dominance-based multiobjective optimization has also gained much importance in the data mining and machine learning communities [21, 22]. Besides, it has also been applied to other kinds of optimization problems based on graph datasets such as multiobjective graph partitioning [2]. Multiobjective optimization usually contains several conflicting objectives that require optimization, and normally there exist many (Pareto) optimal solutions to this problem, where no solution is better than another in all objectives. *Pareto dominance* is an approach to evaluate different solutions based on objective vectors [4, 5, 14]. It is illustrated in Fig. 1.1 using a familiar example in the literature. Assume we have a set of hotels $P = \{p_1, p_i, \ldots, p_{11}\}$ with information of the price and the distance from beach. The Pareto dominance says: point p_i dominates another point $p_j \in P$ if p_i is better than or equal to p_j in all objectives and is strictly better than p_j in at least one objective. With this definition, point p_i is said to be a Pareto-optimal solution if it is not dominated by any other point $p_j \in P$. Thus, Fig. 1.1 contains three points p_1, p_2, p_3 that are said to be Pareto-optimal solutions which collectively form a Pareto-optimal set. An interesting property of the Pareto-optimal set is that it is independent of how you weigh your preferences towards the price and the distance of hotels during selection. In any case, you will find your favorite hotel in the Pareto-optimal set. The Pareto dominance approach is scale invariant, it does not need a ranking function, it does

not apply any threshold and can be used as long as the length of objective vector is low (e.g., $d < 10$) [5, 35]. For high dimensional objective vectors the probability that a solution dominates another becomes very small and this may lead to a large number of Pareto-optimal solutions. Nevertheless, recent proposals have managed to deal with a significantly large number of objectives in what is called *evolutionary many-objective optimization* [20, 37].

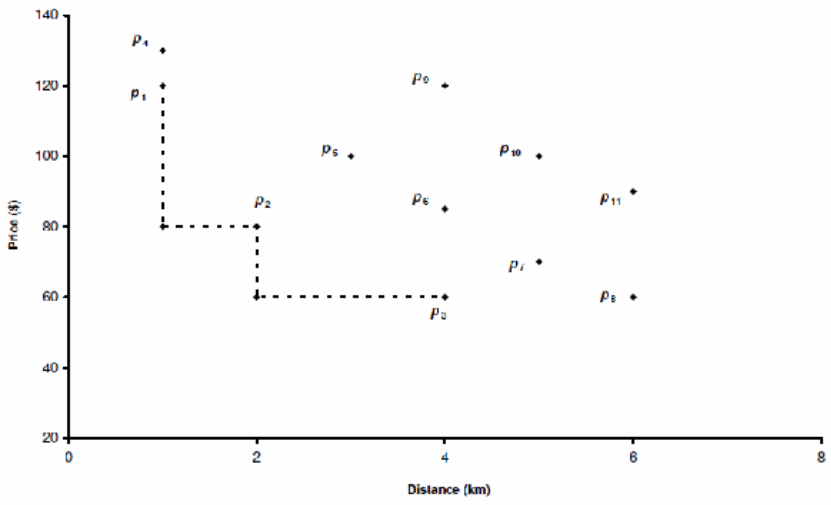

Fig. 1.1 A toy hotel example dataset and Pareto-optimal points in the set

For multiobjective subgraph mining, generating the true Pareto-optimal subgraph set can be computationally expensive and is often infeasible because the complexity of the underlying application prevents exact methods from being applicable [35]. The number of subgraphs in the input graph dataset grows exponentially in relation to the number of nodes, thus resulting in a combinatorial explosion in the subgraph lattice search space. Hence, a challenge is to provide an efficient polynomial time algorithm that can mine a good approximation to the true Pareto-optimal set of the input graph data, i.e., a set of subgraphs whose objective vectors are not too far away from the optimal objective vectors. SkyGraph [35] actually manages to generate Pareto-optimal subgraphs defined by two objectives, the edge connectivity and the order of the subgraph, by means of a polynomial time, exhaustive search algorithm. To do so, SkyGraph performs recursive graph partitioning within a very advanced and well designed framework. 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, which only uses the edge-connectivity to evaluate graph partitioning in a recursive fashion. Therefore, the Pareto dominance approach is only applied for evaluation purposes each time a new subgraph (or solution) is discovered in the recursive search in order to maintain a Pareto-optimal set of discovered subgraphs. As a consequence, it cannot be applied to other multiobjective graph mining tasks requiring the use of different objectives.

In this work, we propose the incorporation of Pareto dominance-based multiobjective search and evaluation strategies from the field of evolutionary multiobjective optimiza-

tion [5] to an existing graph mining method, Subdue [6]. This is done in order to allow this graph mining method to tackle the simultaneous optimization of several conflicting objectives representing different user preferences. The new proposal to perform Multi-Objective subgraph mining using the Subdue algorithm (thus called MOSubdue) is able to generate Pareto-optimal subgraphs regarding to several user-defined criteria on the subgraphs' characteristics.

MOSubdue applies a heuristic search, a more general framework to perform multiobjective subgraph mining. It extends Subdue's beam search in a multiobjective fashion but keeps the remaining Subdue's components (such as the subgraph-growth method) unaltered. Hence, it can work on exactly the same kinds of graph data handled by Subdue (sets of connected relational graphs with or without cycles and directed or undirected edges). The resulting multiobjective beam search is not restricted to use any specific objective but can be customized to different multiobjective GBDM tasks. To illustrate this idea, the current contribution deals with two different multiobjective frequent subgraph mining problems considering two and three objectives, respectively. First, MOSubdue generates a set of Pareto-optimal subgraphs in the case of mining subgraphs jointly maximizing two conflicting objectives, i) the order of the subgraph (the number of nodes) and ii) the support of the subgraph (the occurrence frequency in graph data). Further, to show that MOSubdue is completely general purpose, MOSubdue is extended to solve a three-objective subgraph mining task by considering one more objective (the density of subgraph) along with the latter two.

Two real-life graph-based datasets developed under Predictive Toxicology Evaluation (PTE) challenge, and scientific publication domain-based knowledge discovery (scientograms) are considered to validate our proposal. PTE data has been applied in the past as a benchmark dataset to study the performance of different proposals for frequent subgraph mining task [32, 33, 48]. Scientograms database has been recently applied to propose several automatic knowledge discovery tasks in visual science maps, like the evolution of a scientific domain over time or the extraction of the common research fronts in the world [40]. The performance of MOSubdue is benchmarked with two variants of single-objective Subdue and a multiobjective extension of the well-known Gaston method [32].

The rest of the paper is organized as follows. Section 2 discusses the related work in the area of frequent subgraph mining. Section 3 provides some basic definitions of the different objectives considered and the description of Subdue method for frequent subgraph mining. Section 4 describes the proposed MOSubdue methodology. Experimental results and comparison based on real-life datasets are provided in Section 5. Finally, Section 6 concludes the work and discusses some ideas for the future work.

2 Related Work and Contribution

Recent work in the data mining community has been focused on developing graph-based data approaches to discover subgraphs consisting of complex relationships between entities [1,7]. In this section, we briefly review some fundamental developments related to our work.

2.1 Related Work

Frequent subgraph mining has been the most studied problem in GBDM [3, 6, 18, 19, 26, 32, 48]. The goal is to apply mining techniques on graph data to discover some information that is likely to be useful for the user. For example, a typical graph mining task is to report all

subgraphs that appear in at least m graphs, where m is the minimum frequency (or support) threshold specified, or report all subgraphs that contain at least n nodes, where n is the minimum number of nodes specified by the user. Recently, gPrune [51] has been proposed as a general general framework to incorporate several such thresholds (or constraints) into the graph pattern mining process. gPrune performs constraint-based frequent graph pattern mining using the concept of pattern-inseparable-data-antimonotonicity.

All the previously mentioned developments have in common the application of some specific preference in the graph mining process, such as the number of nodes, the frequency, the density, or the edge-connectivity. In many situations, it would be more useful for the user if the algorithm could jointly consider several preferences to evaluate the mined subgraphs (i.e., multiobjective GBDM), or even better if the algorithm could perform automatic subgraph mining for the preferences defined by the user. To do so, SkyGraph [35] has incorporated skyline processing to discover important subgraphs during the mining process. The method applies the skyline processing on the mined subgraph defined by two objectives, i) the subgraph edge connectivity, and ii) the order of the mined subgraph (the number of nodes). To mine a subgraph, SkyGraph carries out successive application of a min-cut algorithm that uses only one of those two objectives, the edge connectivity. Then, the other objective, the order of the mined subgraph, is evaluated as the number of nodes it consists of. This mined subgraph is stored into an external set P of skyline (or Pareto-optimal) subgraphs if it is not dominated by another subgraph in the set P. Hence, SkyGraph's subgraph search in graph data is single-objective and not multiobjective, as a subgraph is mined according to the edge connectivity by the successive application of the min-cut algorithm. Thus, it only applies a multiobjective processing while evaluating the discovered subgraphs in order to check if they must be stored in the external set of mined subgraphs. In SkyGraph, you could use several objectives while storing the mined subgraph in the external set of mined subgraphs, but Skygraph searches for a subgraph in graph data according to only objective, the edge connectivity. Besides, the strong relation between this objective and the search method applied prevents SkyGraph from being utilized for other multiobjective frequent subgraph mining tasks.

2.2 Our Contribution

Pareto dominance-based evaluation and search strategies are commonly used with evolutionary algorithms for solving multiobjective optimization problems in different fields of science and engineering [5, 21]. In this work, we propose incorporating Pareto dominance-based search [4, 5, 14] in graph data according to several general and user-customizable objectives (i.e. d-dimensional objective vectors) to mine interesting subgraphs. The mined set of subgraphs is a set of Pareto-optimal subgraphs P, where no subgraph is better than another subgraph and every included subgraph is better than all the remaining ones not included in that optimal set. The proposed Pareto dominance-based method performs evaluation and search for d-dimensional objective vector subgraphs in the multiobjective subgraph-search space.

For this purpose, we have used the framework of Subdue method [6] that carries out two main steps, i) subgraph-seed generation, and ii) subgraph-growth. The standard Subdue performs the subgraph-seed generation by using an evaluation method based the minimum description length (MDL) principle [42], and the subgraph-growth by adding an edge and node or only edge to the current subgraph. It applies single-objective heuristic search based on a beam search method [28] to explore the subgraph lattice. We have changed both the

evaluation and the search methods in the standard Subdue. The evaluation method is now based on the actual d-dimensional objective vector that defines a subgraph, and a multiobjective subgraph is evaluated using the Pareto dominance-based approach. A multiobjective subgraph can be seen as a record defined by d objectives in the Pareto dominance-based evaluation method. The search method in Subdue is now extended by incorporating Pareto dominance-based strategy for selecting candidate subgraphs as subgraph-seeds in the mining process for further subgraph-growth operation. This implementation is called MOSubdue.

We show that the subgraph mining process can itself handle multiple preferences (or objectives) that could be meaningful to the user. To do so, MOSubdue is applied for two different graph mining tasks. First, the generation of subgraphs defined by two conflicting objectives, i) the order of the subgraph (the number of nodes) and ii) the support of the subgraph (the occurrence frequency in graph data). Further, to show that MOSubdue can handle more than two objectives, it is extended to solve a three-objective subgraph mining task by considering one more objective (i.e., the density of the subgraph) along with the latter two.

We emphasize that MOSubdue is fundamentally different than SkyGraph [35] method. SkyGraph is a multiobjective GBDM method showing differential characteristics as it is designed for a very specific kind of multiobjective frequent subgraph mining task. As a consequence, it uses two specific objectives for its graph mining task and it cannot work with a different definition for those objectives. Skygraph does not apply a classical Pareto-based multiobjective search but an exhaustive search method based on recursive graph partitioning by only considering a single objective (i.e., the edge connectivity). The underlying search method is thus not guided by any multiobjective approach. A multiobjective evaluation is only performed each time a new subgraph is explored by the recursive search in order to finally keep the non-dominated subgraphs mined in the external set.

On the opposite, MOSubdue's subgraph mining process is based on a pure general-purpose multiobjective subgraph search. MOSubdue implements a Pareto-based multiobjective approximate heuristic search based on Subdue's subgraph-growth approach and beam search method. Hence, MOSubdue applies Pareto dominance not only to evaluate the explored subgraphs but also to actually perform multiobjective search in the subgraph lattice space. MOSubdue is able to deal with several objectives (two and three 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.

In summary, we do not claim MOSubdue is a better multiobjective frequent subgraph mining method than SkyGraph but it is proposed as a complementary approach which can deal with more general multiobjective graph mining tasks.

3 Preliminaries

In this section, we provide some basic definitions of the different preferences considered for the multiobjective subgraph mining tasks. Besides, as the work described in this paper applies the Subdue framework, a brief description of the standard Subdue method [6] is provided.

3.1 Definitions

A labeled connected graph G is denoted by a set of nodes $V(G)$ and a set of edges $E(G)$, where there is an edge e_l between every pair of nodes (v_i, v_j). Each node $v_i \in V(G)$ has a label from the node label set L_V, and each edge $e_l \in E(G)$ that connects two nodes v_i, v_j has a label from the edge label set L_E. The edge e_l can be directed or undirected. In this work, we only consider a set of relational connected graphs $D = \{G_1, G_2, \ldots, G_N\}$, where a graph G_i is said to be relational if each node it contains has an unique label. There are several applications of relational graphs, such as web community detection [11], analysis of biological networks [17], scientific publication domain analysis [40], and social networks [45], among others.

In this study, we have considered some of the commonly used preferences (or objectives) for our multiobjective subgraph mining task which are given below as:

Definition 1.(Induced subgraph p): In GBDM, a subgraph is itself a graph, and will be denoted as p. Graph p is a subgraph of graph p' if p is subgraph isomorphic with p', denoted by $p \subseteq p'$.

Definition 2.(Support of subgraph p): The frequency (or support) of subgraph p denoted by $q(p)$ in a graph database D is the cardinality of the set $\{G_i | p \subseteq G_i, i = 1, \ldots, N\}$. Given a threshold m, the subgraph p is frequent iff $q(p) \geq m$. Exhaustive frequent subgraph mining methods find all such subgraphs.

Definition 3.(Order of subgraph p): The order or size of subgraph p denoted by $s(p)$ is the number of nodes present in the subgraph p. Given a threshold n, the subgraph p is extracted iff $s(p) \geq n$. Exhaustive subgraph mining methods find all such feasible subgraphs.

Definition 4.(Density of subgraph p): The density of subgraph p denoted by $\rho(p)$ is the fraction $\frac{2.|E(p)|}{|V(p)|.(|V(p)|-1)}$. The value of $\rho(p) = 1$ assumes a complete graph.

As seen, all these objectives have been commonly applied in the frequent subgraph mining literature primarily to guide single objective-based search methods by posing some threshold in the mining process [11,17,45]. Recently, some approaches considering multiple objectives together to mine subgraphs have also been introduced [35], suggesting a need of multiobjective subgraph search in graph data during the mining process.

3.2 The Subgraph Mining Framework of the Subdue Method

Subdue [6] is a GBDM method designed for different tasks as frequent subgraph mining, hierarchical clustering, and classification model building from relational data. Subdue has been successfully applied on many real-world problems including, chemistry [6], geology [15], counter-terrorism [16], bioinformatics [25], anomaly detection [34], and scientific publication domain analysis [40], among others.

Subdue is an instance of greedy search-based approaches, which use heuristics to evaluate the subgraphs. It represents data in graph form and can support either directed or undirected edges. Input to Subdue is a single graph or a set of graphs. The framework of Subdue has two main components: i) subgraph-seed generation, and ii) subgraph-growth.

Subgraph-seed Generation: Subdue uses a beam search [28] to enumerate *beamWidth* number of subgraph-seeds according to a subgraph evaluation method based on the MDL principle [42]. It begins from subgraph-seeds consisting of all nodes with unique labels.

The MDL value of the subgraph p is given as:

$$\text{MDL}(G, p) = DL(p) + DL(G|p) \tag{3.1}$$

where $DL(p)$ is the description length of the subgraph p, and $DL(G|p)$ is the description length of the input graph G compressed by the subgraph p. The better a subgraph performs, the smaller the value of equation (3.1) will be. Notice that, to evaluate a subgraph, the MDL measure in equation (3.1) jointly considers two commonly used objectives in GBDM, the support and the size of the subgraph.

Subgraph-growth: The subgraph-seeds are extended by one node and one edge or one edge in all possible ways to generate candidate subgraphs. Candidate subgraphs are evaluated and ranked according to the MDL principle. Following the beam search principle, the best *beamWidth* number of candidate subgraphs are retained as new subgraph-seeds for further expansion. Subdue's output is a set of best (or most descriptive) subgraphs according to the evaluation method in equation (3.1). This procedure repeats until all subgraphs are considered or the user imposed computational constraints are exceeded. Notice that, Subdue is an heuristic search method, which does not perform an exhaustive search in the subgraph lattice.

Fig. 3.1 summarizes the outline of the Subdue method. Inputs are a single graph or a set of graphs G, $maxBest$ is the maximum number of best subgraphs to be reported, *beamWidth* is the length of subgraph-seeds considered for expansion, and $Limit$ is the maximum number of total subgraph-seeds to be expanded. The output comprises the best subgraphs found.

1. **Subdue** (Graph G, *BeamWidth*, *Limit*, *maxBest*)
2. Subgraph-seeds List, Q = {Node v — v has a unique label in graph}
3. Best Subgraphs List, BestP = UpdateBestList(Q) //can store *maxBest* subgraphs
4. **while** *Limit* > 0 and Q $\neq \emptyset$ **do**
5. Candidate Subgraphs List, newQ = {}
6. **for each** p $\in$ Q
7. newQ = newQ $\cup$ NewSubgraphsByExpansion(p) //subgraph-growth
8. $Limit = Limit - 1$
9. Evaluate subgraphs in newQ by MDL measure in Eq.(3.1)
   Sort newQ in ascending order of MDL measure
10. Q = the first *beamWidth* number of subgraphs in newQ
11. BestP = UpdateBestList(Q)
12. **end while**
13. **Return** BestP // the best subgraphs found

Fig. 3.1 The outline of Subdue algorithm.

4 MOSubdue Proposal

In this section, we describe the application of Pareto dominance-based evaluation and search method to enumerate multiobjective subgraph-seeds in Subdue to generate Pareto-optimal subgraphs. Before, we briefly review some basics on multiobjective optimization and methods to solve this problem commonly employed in the multicriteria decision making [4, 5, 10, 14].

4.1 Multiobjective Optimization

Single-objective optimization problems may have a unique optimal solution, while multiobjective optimization problems (MOPs) usually present a set of optimal solutions, which represent trade-offs in objective space. A decision maker then implicitly chooses an acceptable solution or some of them by selecting one or more from the set. An MOP is mathematically defined as follows [4, 5, 14]:

Definition 5 (General MOP): In general, an MOP minimizes $\mathbf{f(x)} = (f_1(\mathbf{x}), \ldots, f_d(\mathbf{x}))$ subject to $\mathbf{x} \in X$, where $\mathbf{x}$ is the solution vector and X is the solution space. An MOP solution minimizes the components of objective vector $\mathbf{f(x)}$, where solution vector $\mathbf{x}$ belongs to solution search space X.

Definition 6 (Pareto dominance): An objective vector $\mathbf{u} = (u_1, \ldots, u_d)$ is said to dominate another vector $\mathbf{v} = (v_1, \ldots, v_d)$ (denoted by $\mathbf{u} \preceq \mathbf{v}$) if $\mathbf{u}$ is less than or equal to $\mathbf{v}$ in all objectives, and is strictly less than $\mathbf{v}$ in at least one objective, i.e., $\forall i \in \{1, 2, \ldots, d\} : u_i \leq v_i \wedge \exists i \in \{1, 2, \ldots, d\} : u_j < v_j$. This definition can also be applied for maximization or any condition of objectives. For simplicity, we have considered minimization of all objectives defining solution $\mathbf{x}$. Actually, our multiobjective subgraph mining problem defined later considers maximization of all objectives.

Definition 7 (Pareto optimality): A solution $\mathbf{x} \in X$ with objective vector $\mathbf{u}$ is said to be Pareto optimal with respect to the search space X iff there is no solution $\mathbf{x'} \in X$ with objective vector $\mathbf{v}$ that dominates $\mathbf{u}$.

Definition 8 (Pareto optimal set): For a given MOP $\mathbf{f(x)}$, the Pareto optimal set $\mathcal{P}^*$ is defined as:

$$\mathcal{P}^* := \{\mathbf{x} \in X \mid \neg\exists\, \mathbf{x'} \in X\ \mathbf{f(x')} \preceq \mathbf{f(x)}\} \tag{4.1}$$

Definition 9 (Pareto front): For a given MOP $\mathbf{f(x)}$, the Pareto-optimal front $\mathcal{PF}^*$ associated with the Pareto optimal set $\mathcal{P}^*$ is defined as:

$$\mathcal{PF}^* := \{\mathbf{u} = \mathbf{f(x)} = (f_i(\mathbf{x}), \ldots, f_d(\mathbf{x}))|(\mathbf{x} \in \mathcal{P}^*)\} \tag{4.2}$$

Thus, an MOP contains several objectives that must be jointly optimized. These objectives are usually conflicting in nature, their optimization offers several optimal solutions in the objective space. To solve an MOP, the optimization algorithm should efficiently and effectively find those solutions that satisfy multiple objectives. In other words, the obtained solutions should be of good proximity and diversity to the true Pareto-optimal solution set $\mathcal{P}^*$. Proximity means that the algorithm is of excellent searching ability to obtain good solutions on or close to the true Pareto-optimal front $\mathcal{PF}^*$. Diversity means that the algorithm is capable to obtain solutions distributed uniformly to some extent for the decision-maker to find a comparatively satisfying solution close to his preference at any time.

Perhaps the most straightforward approach to solve an MOP is to combine different objectives into a single-objective scalar value function by any kind of objective aggregation scheme, and apply a single-objective optimization approach to generate Pareto-optimal solutions [4, 5, 14]. However, this formulation will generate only the specific solutions subject to the trade-off between the objectives explicitly or implicitly specified by the aggregation function. To overcome limitations of aggregating schemes, evolutionary multiobjective optimization (EMO) algorithms have successfully shown the application of Pareto dominance-based evaluation of solutions to guide the search process in the multiobjective solution search space to generate good Pareto front approximations [5, 10, 52].

In this contribution, we apply the concept of general MOP to define the multiobjective subgraph mining problem, and apply a Pareto dominance-based scheme for evaluation of subgraphs to guide the mining process in the multiobjective subgraph lattice search space. In the following sections, we provide the problem statement for multiobjective subgraph mining as well as the methodology for evaluation of subgraphs and to guide the mining process. For this purpose, we use the terms subgraph and solution interchangeably.

4.2 Multiobjective Subgraph Mining Problem Statement

Multiobjective subgraph mining is based on the idea of multiobjective optimization, where a solution $\mathbf{x}$ is defined as a subgraph p, a set of nodes and edges, the solution space X is referred as the subgraph search space, i.e., the subgraph lattice. A subgraph is defined by several d user-defined objectives on the subgraph's characteristics, such as the frequency q, the order s, etc., which are usually conflicting. For example, subgraphs with high frequency are usually of small order (or size) and *vice-versa*. Formally, given a set of graphs G, our goal is to mine the Pareto-optimal subgraph set representing all the induced connected subgraphs of G defined by several user-defined objectives. For this purpose, we have formulated two multiobjective subgraph mining tasks as:

- *Given a set of graphs G, mine all Pareto-optimal subgraphs of G which are maximal with respect to the support (or frequency) and the number of nodes (or order).*
- *Given a set of graphs G, mine all Pareto-optimal subgraphs of G which are maximal with respect to the support, order, and density.*

Theoretically, the subgraph mining algorithm has to search the entire subgraph lattice that represents all possible subgraphs to determine if a mined subgraph is Pareto-optimal [35]. However, the number of possible subgraphs in the subgraph lattice grows exponentially in relation to the number of nodes. This makes finding Pareto-optimal subgraphs computationally expensive and often infeasible when dealing with large graph databases. Moreover, the complexity of the underlying application prevents exact methods from being applicable. In this scenario, we need to rely on the GBDM methods proposed in the literature that perform approximate heuristic search in the subgraph lattice to generate good approximations to the true Pareto-optimal subgraph set in reasonable computational time.

Subdue [6] is an instance of approximate heuristic search in the subgraph lattice for frequent subgraph mining. In this work, we apply the framework of Subdue to solve the above formulated multiobjective subgraph mining problem. As said, one way to solve this problem is by aggregation of objectives (see Section 4.1). However, a subgraph defined by the aggregation of objectives, e.g., the support and the order, in a single-objective scalar function would result in a similar behavior where only the specific subgraphs showing the specified trade-off between the two objectives would be mined. This is the classical drawback of aggregation schemes as found in the multiobjective optimization area [4, 5, 14]. With that problem in mind, we propose the use of Pareto dominance-based evaluation and search methods in the framework of Subdue algorithm to mine subgraphs defined by several user-defined preferences (or objectives).

4.3 Pareto Dominance-based Subgraph Evaluation and Selection Method

Pareto dominance definition (6) can be used: i) to estimate the quality (or fitness or rank) of a solution using the objective vector, and ii) to establish preference between solutions

for selection. Methods based on the concept of Pareto dominance are very popular in EMO area [5, 10, 13, 52]. One such method is proposed in [10], which is called Pareto dominance-based ranking. To describe this solution ranking procedure, we revisit the classical hotel selection example shown in Fig. 1.1. We seek hotels with low price and short distance to beach. Applying the Pareto dominance definition (6) on the entire set P of size 11, we find three points $p_1, p_2,$ and p_3 that are optimal and comprise the front F_1 as shown in Fig. 4.1. Suppose, the hotels belonging to this front have been fully occupied. In this scenario, the visitor needs to draw another front considering the remaining hotels (i.e., by temporarily discarding points $p_1, p_2,$ and p_3 in the set P). Application of the Pareto dominance definition (6) on the temporarily pruned set P provides the second front F_2 to choose from five points ($p_4, p_5, p_6, p_7,$ and p_8). To make further choice, the last front F_3 which contains three points $p_9, p_{10},$ and p_{11} is obtained by temporarily discarding all points belonging to the fronts F_1 and F_2. In this way, we have sorted the set P into different fronts using the Pareto dominance definition (6). It can be noticed that the first front F_1 is better than any other front in the set P. This is because it was obtained on the entire set P. Any subsequent front was obtained on the temporarily pruned set P. Thus, we can actually rank points in the set P based on the front number to which they belong. Hence, points belonging to the front F_1 share rank 1, points in the front F_2 have rank 2, and so on. As points holding the best rank 1 are from the front F_1, this ranking method assumes rank minimization. In this way, in Fig. 4.1 the Pareto dominance-based ranking has performed two functions: i) evaluation of points using objective vectors, i.e., estimation of rank for each point in the set P based on the front number it belongs, and ii) preference based selection, i.e., minimization of rank is assumed and thus points with the rank 1 are the best, points with rank 2 are the second-best, and so on. MOSubdue applies this procedure in the multiobjective beam search to enable generation of *beamWidth* number of subgraph-seeds from the multiobjective candidate subgraphs.

The pseudo-code of MOSubdue is given in Fig. 4.2. In this figure, in line 9, the list newQ contains the candidate subgraphs defined by d number of user-defined objectives. The multiobjective beam search sorts the list newQ into different fronts using the Pareto dominance definition (6) (like as illustrated in Fig.4.1). It assigns rank to each candidate subgraph in the list newQ equal to the front number it belongs. To generate *beamWidth* number of subgraph-seeds from the list newQ, assuming the rank minimization, the candidate subgraphs are sorted in the ascending order of rank. The topmost *beamWidth* number of candidate subgraphs in the sorted list newQ are selected as subgraph-seeds for further expansion (line 10). We call this approach as MOSubdue-I method.

In MOSubdue-I, generation of the subgraph-seed list Q of length *beamWidth* from the list newQ sorted into different fronts can be seen as follows. First we choose the front F_1. If the size of front F_1 is smaller than *beamWidth* then all the candidate subgraphs belonging to this front are selected. Next we choose the front F_2, and so on until the total of number candidate subgraphs in the fronts $F_1, F_2, \ldots, F_l$ is not greater than/equal to *beamWidth*. F_l is the last front that can be accommodated to form the list Q. We cut the front F_l simply at a point where the addition of the sizes of fronts $F_1, F_2, \ldots, F_l$ is equal to *beamWidth*.

The worst case complexity of this Pareto dominance-based ranking is $O(dK^2)$, where d is the number of objectives to define a subgraph, and K is the length of candidate subgraph list newQ. In the worst case, the list newQ is sorted into K fronts with one subgraph per front [10]. However, in practice, the actual computational time complexity is low as we terminate fronts generation as soon as we find enough fronts to obtain *beamWidth* number of subgraph-seeds.

When reviewing the latter selection procedure, it should be noticed that all the candidate subgraphs in the front F_l share same rank l, and hence have equal probability to become

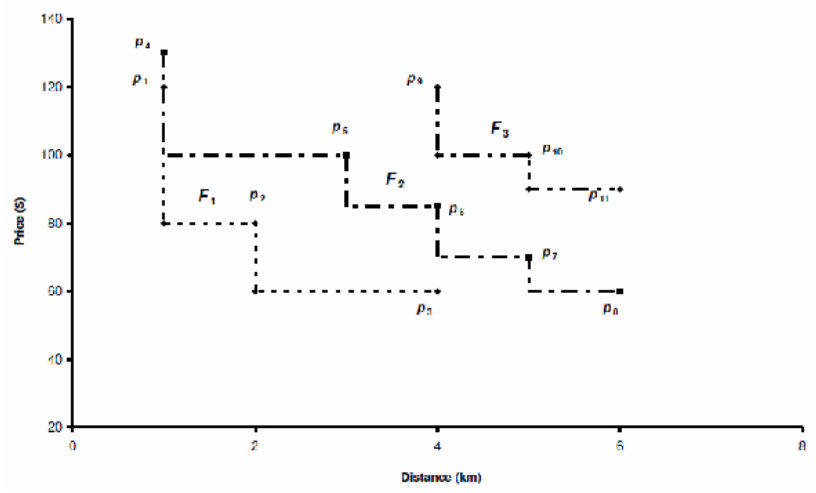

Fig. 4.1 All possible Pareto fronts generated in the toy hotel example dataset. The front F_1 is generated considering all 11 points. The front F_2 is produced by temporarily discarding all points belonging to the front F_1. Similarly, the front F_3 is obtained by discarding all points belonging to the previously generated fronts, i.e., both the fronts F_1 and F_2.

1. **MOSubdue** (Graph G, *BeamWidth*, *Limit*, *maxParetoSubs*)
2. Subgraph-seeds List, Q = {Node v — v has a unique label in graph}
3. Pareto Subgraphs List, ParetoList = UpdateParetoSubList(Q) //can store max. *maxParetoSubs* subgraphs
4. **while** *Limit* > 0 and Q $\neq \emptyset$ **do**
5. Candidate Subgraphs List, newQ = {}
6. **for each** p $\in$ Q
7. newQ = newQ $\cup$ NewSubgraphsByExpansion(p) //subgraph-growth
8. $Limit = Limit - 1$
9. Apply Pareto dominance-based evaluation and selection on the list newQ
   Sort the list newQ into different fronts using objective vectors
   Assign rank to each candidate subgraph equal to front number it belongs
10. Q = *beamWidth* number of subgraphs in newQ according to the minimum rank and uniformly distributed subgraph selection
11. ParetoList = UpdateParetoList(Q)
12. **end while**
13. **Return** ParetoList // the Pareto-optimal subgraphs found

Fig. 4.2 The Pseudo-Code of MOSubdue.

subgraph-seeds in the list Q. So, it will be appropriate to perform uniformly distributed selection on the front F_l. We have done so using the objective vectors of the candidate subgraphs in the front F_l. We apply this modification as MOSubdue-II method [44]. The overall procedure for evaluation and selection applied by the designed multiobjective beam search in the list newQ is depicted in Fig. 4.3.

Application of uniformly distributed selection in the front F_l of solutions defined by d objectives is given as follows. To measure how a solution p_i is spread over the front F_l, we

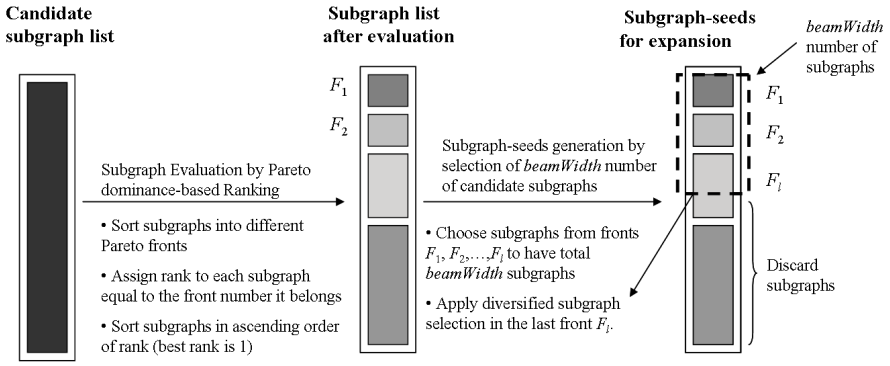

Fig. 4.3 Multiobjective beam search applied by MOSubude. At any expansion stage, the subgraph-growth operation has generated a d-objective vector candidate subgraph list newQ. The beam search applies Pareto dominance-based ranking to sort list newQ into different fronts, say F_1, F_2, ..., and assigns rank to each candidate subgraph equal to the front number it belongs. It assumes the best candidate subgraph has minimum rank and sorts list newQ according to ascending value of the rank. To create the subgraph-seed list Q of size *beamWidth*, it selects the candidate subgraphs with minimum rank such that $|F_1| + |F_2| + \ldots + |F_l| \geq beamWidth$. From the last accommodated front F_l, the most diversified candidate subgraphs are selected.

calculate α_i of p_i as the average distance of two solutions on either side of p_i along each of d objectives given as:

step 1 Sort L solutions in the front F_l in the ascending order of each f_j objective. α_{ij} has assigned infinite value for solutions with the smallest and largest values of objective f_j (i.e., $\alpha_{1j} = \alpha_{Lj} = \infty$). For the remaining solutions it is calculated as:

$$\alpha_{ij} = \frac{f_j(p_{i+1}) - f_j(p_{i-1})}{f_j(p_{max}) - f_j(p_{min})}, i = 2, \ldots, L-1 \tag{4.3}$$

step 2 Repeat step 1 with each objective $f_j, j = 1, \ldots, d$, and find the distribution value α_i of solution p_i as:

$$\alpha_i = \sum_{j=1}^{d} \alpha_{ij} \tag{4.4}$$

This diversified selection method has a computational complexity of $O(dLlogL)$, where L is the size of Pareto front F_l. This type of distributed selection has been applied in the nondominated sorting genetic algorithm-II (NSGA-II) in the EMO area [10].

We exemplify the computation of α_i on points in the front F_2 in the hotel selection example as shown in Fig. 4.4. There are five points $p_4, p_5, \ldots, p_8$ in the front F_2. To calculate distribution α_i for each point p_i in F_2, we first sort this front in ascending order of price. Thereafter, points with the smallest and the largest price values have assigned an infinite value (i.e., $\alpha_1 = \infty$ for points p_8 and p_4, respectively). For all other intermediate points, α_1 is equal to the absolute normalized difference in the price values of two adjacent points. For example, for point p_5 $\alpha_1 = 45/70 = (130 - 85)/(130 - 60)$ is obtained as the normalized

difference in the price values of points p_4 and p_6. Similarly, we apply these two steps considering the other objective, the distance from beach. Sorting according to ascending order of this objective values assigns $\alpha_2 = \infty$ for points p_4 and p_8. α_2 for the remaining points is computed as the absolute normalized difference in the values of the distance from beach using two nearest neighbors with each one from either side. α_2 for one of the intermedidate points p_5 is given as $\alpha_2 = 3/5 = (4-1)/(6-1)$. Thus, the distribution value α_i for p_5 in the front F_2 is the sum of α_1 and α_2 values. It can be observed that the procedure is straight forward to apply for any number of objectives, as we have used it for a three-objective subgraph mining problem in this study.

Points	Price ($)	Distance (km)	α_{price} (α_1)	$\alpha_{distance}$ (α_2)	$\alpha = \alpha_1 + \alpha_2$
p_4	130	1	∞	∞	∞
p_5	100	3	45/70	3/5	87/70
p_6	85	4	30/70	2/5	29/35
p_7	70	5	25/70	2/5	53/70
p_8	60	6	∞	∞	∞

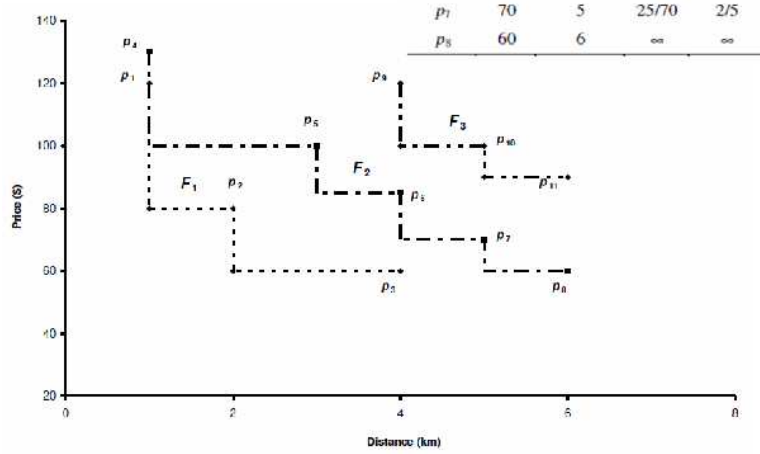

Fig. 4.4 Diversification calculation for points in the hotel example. The procedure is described using the points belonging to the front F_2. For each point p_4 to p_8 corresponding the price ($) and the distance (km) values are given. α_1 and α_2 represent the distribution values for points according to the price and the distance values. First sort the points in the ascending order of magnitude for each objective, and points at the extreme ends have assigned an infinite diversification value. Thus, points p_4 and p_8 have been assigned $\alpha = \infty$ for carrying the smallest and largest values with the two objectives. For all other remaining points α values are calculated as ($\alpha = \alpha_1 + \alpha_2$), the summation of the absolute normalized difference between the two adjacent points in both the objectives.

Let us consider the complexity of one iteration of the entire MOSubdue algorithm. The basic operations and their worst-case complexities are as follows:

1. subgraph-growth operation by Subdue is $O((\sum_{i=1}^{Limit} i*((v-1)-(i-1)))*(v(Limit-1))*gm)$ [8]
2. subgraph-seeds generation
   (a) Pareto dominance-based ranking is $O(dK^2)$
   (b) diversified subgraph selection is $O(dLlogL)$

The run time for Subdue's subgraph-growth process is $O((\sum_{i=1}^{Limit} i*((V-1)-(i-1)))*(V(Limit-1))*gm)$, calculated considering the total number of subgraphs to be

expanded $Limit$, the number of instances of each subgraph in the input graph G, the number of partial mappings considered during graph matching gm, and the number of nodes $V(G)$. The introduced Pareto-dominance-based subgraph-seeds generation adds the overall complexity of $O(dK^2)$ in MOSubdue. If the subgraph evaluation is performed carefully, the complete candidate subgraph list newQ does not need to be sorted into different Pareto fronts. As soon as the Pareto dominance-based ranking procedure has found enough number of fronts in the list newQ to have *beamWidth* number of candidate subgraphs, there is no reason to continue the ranking computation.

4.4 Comparison Between the Operation of Subdue and MOSubdue

In this section, we provide an illustrative example to show the different operation that Subdue and MOSubdue apply for solving the two objective subgraph mining problem defined in Section 4.2. The single-objective beam search in the Subdue method in Fig. 3.1 uses the MDL measure in equation (3.1) to evaluate subgraphs. Equation (3.1) is a combination of two objectives, the support and the size (#nodes + #edges) of the subgraph. The multiobjective beam search in the MOSubdue-I method (i.e., without the application of the diversified selection procedure) uses Pareto dominance-based ranking to evaluate subgraphs defined by two objectives, the support and the order of the subgraphs[1].

We apply identical parameter settings for both methods, i.e., *beamWidth* = 5, *Limit* = ∞, and *maxBest* = *maxParetoSubs* = 100. Both methods are applied on the *shapes domain*, a synthetically generated dataset frequently used in the study of Subdue method [6]. Fig. 4.5 shows an example of a graphical representation of the input shapes data. The objects in the figure (e.g., C1, T1, S1) become labeled nodes in the graph and the relationships (e.g., on (T1,S1), shape (T1,triangle)) become labeled edges in the graph. The considered dataset consists of 100 different graphs with a total of 500 nodes, 400 edges, and 6 unique node labels. The Pareto-optimal set $\mathcal{P}^*$ (which is known for this simple domain as it has been computed in an exhaustive way) contains 12 different subgraphs out of which 7 are distinct in the objective vector space. The graphical representation of one of the subgraphs discovered by MOSubdue-I from this dataset is also shown in Fig. 4.5.

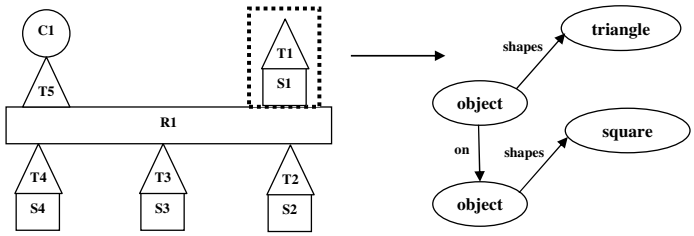

Fig. 4.5 An example of a subgraph in the shapes domain.

Both methods start from subgraph-seeds consisting of all nodes with unique labels. At any expansion stage, both methods apply the same subgraph-growth operation on the subgraph-seeds to generate the candidate subgraphs. However, they consider a different

[1]Notice that, any other formulation for these two objectives can be considered. Anyway, they constitute support and size objectives as in the MDL measure.

subgraph selection procedure on the candidate subgraphs to generate *beamWidth* number of subgraph-seeds for the next stage of expansion. Fig. 4.6 reports the support and the order values of *beamWidth* = 5 subgraph-seeds generated by the Subdue and MOSubdue methods at each expansion stage. At the beginning, i.e., at the first expansion stage in Fig. 4.6, both methods start from the same subgraph-seeds as shown in Fig. 4.6(a), consisting of all nodes with unique labels in the shapes dataset. For the second expansion stage, one out of 5 subgraph-seeds generated by Subdue is different from that generated by MOSubdue-I as highlighted in gray in Fig. 4.6(b). For the third expansion stage, Subdue has generated the subgraph-seed list which contains two solutions that are not present in the subgraph-seed list created by MOSubdue-I (see, Fig. 4.6(c)). Both methods have performed 6 generations. Finally, as shown in Fig. 4.6(h), the Pareto-optimal set of subgraphs reported by Subdue (resulting from the use of Pareto dominance definition (6) on the final output list of single-objective Subdue) has two solutions dominated by that of MOSubdue-I, showing the better performance of the latter.

5 Experimental Study

The performance evaluation study has been conducted in our experiments on two real-world datasets, which are summarized in Table 5.1 and are briefly described as follows:

Table 5.1 Description of different datasets used.

Datasets	#Graphs	#Nodes	#Edges	#Unique Labels
chemical	340	9189	9317	66
scientograms	73	19253	19709	296

Chemical Compound Data is a dataset which was available under the Predictive Toxicology Evaluation (PTE) challenge [2]. The dataset contains 340 chemical compounds, 24 different atoms, 66 atom types, and 4 types of bonds. The dataset consists of 27 nodes and 28 edges per graph on average. The largest one contains 214 edges and 214 nodes. So, the discovered subgraphs are much like trees, though they do contain some cycles. The type of atoms and bonds form the labels to the nodes and edges in the dataset. The PTE dataset was earlier used in [32, 33, 48].

Scientograms Database [40] is a database built following De Moya-Anegón et al.'s methodology [30, 47] to design visual science maps (scientograms) for huge scientific publications collections. The nodes of the graphs correspond to Elsevier SCOPUS-SJR[3] co-citation categories. Each category agglutinates the journals that were categorized under that name, and likewise the documents that were published in those journals. A co-citation measure is used to compute the relational similarity between two categories, thus defining a relation matrix with an associated graph. Only the salient relationships between categories are kept, capturing the essential underlying intellectual structure of the studied scientific domain, using the Pathfinder algorithm [9, 39] to prune the graphs. The rough considered data

[2] http://www.comlab.ox.ac.uk/activities/machinelearning/PTE/

[3] http://www.scopus.com

Subdue and MOSubdue-I	
support	order
100	1
43	1
43	1
43	1
41	1
34	1

(a)

Subdue		MOSubdue-I	
support	order	support	order
76	2	76	2
43	2	43	2
43	2	43	2
43	2	43	2
41	2	43	2

(b)

Subdue		MOSubdue-I	
support	order	support	order
55	3	55	3
31	3	30	3
30	3	28	3
29	3	28	3
28	3	27	3

(c)

Subdue		MOSubdue-I	
support	order	support	order
19	4	19	4
17	4	16	4
16	4	16	4
15	4	16	4
13	4	15	4

(d)

Subdue		MOSubdue-I	
support	order	support	order
6	5	9	5
6	5	9	5
6	5	7	5
5	5	6	5
5	5	6	5

(e)

Subdue		MOSubdue-I	
support	order	support	order
3	6	4	6
3	6	4	6
3	6	3	6
3	6	3	6
2	6	3	6

(f)

Subdue and MOSubdue-I	
support	order
2	7
2	7

(g)

Subdue		MOSubdue-I	
support	order	support	order
2	7	2	7
3	6	4	6
6	5	9	5
19	4	19	4
55	3	55	3
76	2	76	2
100	1	100	1

(h)

Fig. 4.6 Comparison between the operation of Subdue and MOSubdue methods on the shapes dataset. The subgraph-seed list of length *beamWidth=5* generated by each algorithm at the end of each expansion stage is reported. Both methods terminated after 6 generations shown in b to g. Solutions, if any, not present in either generated list are highlighted in gray. (a) Both methods began with subgraph-seed list consisting of all nodes with unique labels; (b) After the first expansion stage, Subdue has generated a subgraph-seed list which contains one solution with a different value of support to that of generated by MOSubdue-I, and so on; (h) Finally, both methods have reported their Pareto-optimal set of subgraphs, which indicates the two solutions in gray produced by Subdue are dominated by those of MOSubdue-I.

have been extracted from the Scimago Journal & Country Rank portal[4] and comprise a set

[4]http://www.scimagojr.com/

of 36 millions of documents indexed by Elsevier SCOPUS-SJR from 1996 to 2008 over 73 countries [47]. This database has been extensively analyzed in [40] to propose an automatic approach allowing the identification and the comparison of scientific structures within scientograms. To do so, the Subdue algorithm has been applied for three different scientogram analysis tasks regarding the evolution of a scientific domain over time, the extraction of the common research fronts in the world, and the comparison of scientific domains between different countries. In the current study, the scientogram dataset contains visual science maps generated for 73 countries for the year 2005. The dataset contains 73 graphs with 264 nodes and 270 edges per graph on average, and consists of 294 unique node labels. As the dataset does not contain any cycles, the mined subgraphs are like trees. However, the large size and the presence of several unique node labels make this dataset a challenging one for the defined multiobjective subgraph mining tasks.

The two variants of MOSubdue have been implemented in C, and all experiments have been performed on an Intel Core Quad at 2.66GHz, with 4GB RAM, running CentOS 5.5. Additionally, for the comparison study, Subdue and Gaston methods have been adapted for solving the defined multiobjective subgraph mining tasks. These methods were originally proposed for single-objective frequent subgraph mining and their source code in C is available through URLs[5,6]. Their adaptation is briefly described as follows:

Subdue-I: This applies three different independent subgraph evaluation methods, viz. the MDL, size, and set cover, originally supported by the Subdue algorithm [6] for frequent subgraph mining. Subdue is executed on the input graph dataset with the three evaluation methods independently. The outputs from three evaluation methods are merged and repeated subgraphs are removed, and later domination checks using the Pareto dominance definition (6) are performed to produce Pareto-optimal subgraph set as generated by the Subdue-I method.

Subdue-II: This basic multiobjective extension of the Subdue algorithm applies a modified subgraph evaluation method based on a single-objective function combining multiple objectives in a weighted additive fashion [4, 5, 14] (as said, in our case, d= 2 and 3). Let $\lambda = (\lambda_1, \ldots, \lambda_d)^T$ be a weight vector, i.e. $\lambda_i \geq 0$ for all $i = 1, \ldots, d$ and $\sum_{i=1}^{d} \lambda_i = 1$. Then, the subgraph p is evaluated using the following scalar objective function as:

$$\begin{aligned} &\text{maximize} \quad z(p|\lambda) = \sum_{i=1}^{d} \lambda_i f_i(p) \\ &\text{subject to} \quad p \in P \end{aligned} \tag{5.1}$$

where we use $z(p|\lambda)$ to emphasize that λ is a coefficient vector in this objective function. Of course, the considered objectives are normalized. To generate a set of different optimal subgraphs, one can use different weight vectors in the above scalar objective function, and perform repeated runs of Subdue-II.

MOGaston: Gaston [32] is a quick start algorithm for frequent subgraph mining, as it applies efficient ways to uniquely enumerate paths and trees. The algorithm first generates paths, then trees, and finally general graphs in order to efficiently search through the subgraph lattice. It stores all embeddings to generate only new subgraphs that actually appear in the database and to achieve fast isomorphism testing. In the last phase, the algorithm deals with general graphs by defining a global order on cycle-closing edges to minimize the

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

[6] Gaston: http://www.liacs.nl/~snijssen/gaston/

need for graph isomorphism tests. Only in handling general graphs, Gaston faces the NP-completeness of the subgraph isomorphism problem. Gaston can calculate the frequency of a subgraph either with isomorphism tests or embedding lists. In total the theoretical time complexity of Gaston is $O(|E_c(G)|^c \log |E(G)| + mc \log\ c)$, where E_c is the number of edges of the connected graph G that occurs in a cycle, c is the number of edges that should be removed to obtain a tree, and m is the number of automorphisms of the spanning tree. If c and m are small, this computation is polynomial in the size of the graph to be normalized [32].

Input to Gaston is a set of graphs and a value for minimum frequency (or support) m to retrieve subgraphs, and the output is a list of all the mined subgraphs with frequency greater than or equal to m. To obtain a Pareto-optimal set from the output list, a simple modification is done in the output of Gaston as: i) compute the additional objectives, the order and the density of the mined subgraph, and ii) check dominance of the mined subgraph with the subgraphs in an external Pareto set archive P. If this mined subgraph is not dominated by any subgraph in the set P then it is included in P, which is updated eventually to remove dominated subgraph, if any, it could contain. Notice that, as Gaston is an exhaustive search method, when the lowest values for the thresholds are considered, the multiobjective extension designed is able to obtain the true Pareto-optimal set of subgraphs for the tackled frequent subgraph mining task. Nevertheless, that would require an enormous and many times unaffordable computation time for large graph databases due to the exponential size of the subgraph search space. In our study, the graph datasets are large and complex and thus it has been practically infeasible to let MOGaston run till exhuastion to carry out an exhaustive search for mining subgraphs with $m \geq 2$. In fact, for the scientogram dataset, MOGaston spent more than ten hours in mining subgraphs with only $m \geq 8$. Thus, for the purpose of performance comparison study, we have decided to fix the execution time for MOGaston based on the time corresponding to the best result obtained by any of the Subdue-based methods on each dataset.

5.1 Parameter Setting

Subdue-I and II and MOSubdue-I and II methods have been run with three different values of *beamWidth* = 5, 10, and 20. Each of these methods has been run till subgraph-seeds can not be grown further to generate candidate subgraphs, i.e., till exhaustion of the explored subgraph search space. A maximum number of Pareto subgraphs to be reported was set to $maxBest = maxParetoSubs$ = 100. A single execution of Subdue-I and MOSubdue-I has been carried out on the input graph datasets as a consequence of being deterministic methods while MOSubdue-II has been run ten times with ten different seeds.

Subdue-II has applied different weight vectors in the case of two and three objective problems. For the two-objective problem, the weight of the first objective function, the support, is varied from 0 to 1 in the step of 0.1, which has resulted into 11 weight vectors. The algorithm has been run for each of the eleven weight vectors. For the three-objective problem, we have used 13 different weight vectors given in Table 5.2, and Subdue-II has been run with each of them.

Finally, simulations have been performed with MOGaston using three different run times on each dataset. The duration for the first run was set corresponding to the computational time associated with the best result produced by any of the Subdue-based methods, while the duration for the other two runs were set equal to two and five times the duration for the first run.

Table 5.2 Different weight vectors to transform the three-objective problem into a single value scalar function in Subdue-II method

	Weights for objectivesWeights for objectives		
Sr.No.	SupportSupport	OrderOrder	DensityDensity
1	1.00	0.00	0.00
2	0.90	0.10	0.00
3	0.80	0.20	0.00
4	0.70	0.20	0.10
5	0.50	0.30	0.20
6	0.40	0.30	0.30
7	0.33	0.33	0.33
8	0.30	0.30	0.40
9	0.20	0.30	0.50
10	0.10	0.20	0.70
11	0.00	0.20	0.80
12	0.00	0.10	0.90
13	0.00	0.00	1.00

5.2 Performance Evaluation

To evaluate the performance of the proposed Pareto dominance-based multiobjective subgraph search approach, we compare the Pareto-optimal set of subgraphs produced by each of the applied methods. A classical way to do so in EMO studies [5, 53, 54] is to check the closeness of the Pareto-optimal set P produced by the algorithm with respect to the true Pareto-optimal set $\mathcal{P}^*$ on the input dataset G. Thus, the set P produced by the algorithm is an *approximation* to the set $\mathcal{P}^*$. The true set $\mathcal{P}^*$ contains all subgraphs according to definition (7) of Pareto optimality (see Section 4.1) from the multiobjective subgraph search space, i.e. the subgraph lattice of the input graph dataset, and it may be obtained by employing an exhaustive search on small size datasets. However, it is practically infeasible to run an exhaustive search on large sets of real-world graphs, which is the case in our experimental study. To overcome this problem, we have generated a pseudo Pareto-optimal set obtained from the aggregation of the set P produced by the different methods in all runs performed. Here after, we consider this pseudo Pareto-optimal set is equivalent to the set $\mathcal{P}^*$ in the performance analysis, unless otherwise specified.

Hypervolume ratio (*HVR*) is a commonly used and powerful measure in EMO studies [5, 53, 54] to compute the proximity of Pareto-optimal front PF obtained from the objective vectors of solutions in the set P to the Pareto-optimal front $\mathcal{PF}^*$ of the set $\mathcal{P}^*$. It is measured in the objective space of solutions. For a two-objective problem, the hypervolume is the summation of the area covered by each member in the front PF with respect to the objective space axis. The use of *HVR*-metric is very extended in the EMO area as it measures both diversity and closeness of the approximation to the set $\mathcal{P}^*$. It is calculated as the ratio of the hypervolume for the front PF to that for the front $\mathcal{PF}^*$. A value of 1 for the *HVR*-metric indicates the front PF of the solution set P obtained by the algorithm duplicates the front $\mathcal{PF}^*$ of the solution set $\mathcal{P}^*$ on the input dataset G. Thus, a high value of *HVR*-metric indicates a good approximation to the set $\mathcal{P}^*$ has been produced by the algorithm.

We have computed the *HVR*-metric corresponding to Pareto-optimal subgraph set P obtained by the different methods for the two and three objective problems. For the two-objective problem, the methods have produced Pareto-optimal solutions defined by the support and the order of the subgraphs. Meanwhile, for the three-objective problem, the methods have generated Pareto-optimal solutions defined by the support, the order, and the density of the subgraphs. In the following subsections, we analyze the performance of the different methods for both problems.

5.3 Analysis of Results for the Two Objective Subgraph Mining Task

Tables 5.3 and 5.4 report the *HVR*-metric values for the approximations produced by the different Subdue methods on both datasets for the two-objective subgraph mining task. The values in these tables associated with MOSubdue-II and Subdue-II represent the mean and standard deviation values corresponding to the 10 and 11 different runs performed, respectively. Table 5.5 provides the *HVR*-metric values for the obtained approximations corresponding to the different runs of MOGaston on both datasets. The values in the brackets in this table represent the run time in seconds for each execution of MOGaston. Table 5.6 shows the HVR-metric values corresponding to the best approximation produced by the different methods on each dataset. Tables 5.7 and 5.8 provide the run time analysis for the different Subdue methods to produce their approximations.

Table 5.3 *The HVR*-metric values for Pareto-optimal sets obtained by the different Subdue methods on the chemical dataset for the two-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	0.9729 (-)	0.9392 (-)	0.9243 (-)
Subdue-II	0.8120 (0.2042)	0.3009 (0.3129)	0.1054 (0.0560)
MOSubdue-I	0.9537 (-)	**0.9898 (-)**	0.9675 (-)
MOSubdue-II	0.9522 (0.0000)	0.9662 (0.0012)	0.9652 (0.0036)

Table 5.4 The *HVR*-metric values for Pareto-optimal sets obtained by the different Subdue methods on the scientogram dataset for the two-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	0.8545 (-)	0.7990 (-)	0.8090 (-)
Subdue-II	0.1606 (0.0242)	0.1052 (0.0242)	0.1017 (0.0265)
MOSubdue-I	0.8206 (-)	0.8491 (-)	0.6520 (-)
MOSubdue-II	0.8606 (0.0000)	**0.8968 (0.0000)**	0.6735 (0.0000)

Table 5.5 The *HVR*-metric values for Pareto-optimal sets obtained by MOGaston using three different run times on both datasets for the two-objective subgraph mining task. The numbers in the brackets represent the run times in seconds.

DatasetDataset	Run 1Run 1	Run 2Run 2	Run 3Run 3
Chemical	0.0583 [50]	0.0583 [100]	0.0612 [250]
Scientogram	0.0746 [485]	0.0762 [970]	0.0762 [2425]

Table 5.6 The *HVR*-metric values corresponding to the best result produced by the different methods on both datasets for the two-objective subgraph mining task.

Datasets	Subdue-I	Subdue-II	MOSubdue-I	MOSubdue-II	MOGaston
Chemical	0.9729	0.8120	**0.9898**	0.9662	0.0612
Scientogram	0.8545	0.1606	0.8491	**0.8968**	0.0762

Table 5.7 Run time in seconds for the different Subdue methods on the chemical dataset for the two-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	40.71 (-)	79.2 (-)	165.53 (-)
Subdue-II	10.46 (3.27)	15.69 (4.87)	29.29 (10.48)
MOSubdue-I	20.73 (-)	**49.47 (-)**	92.71 (-)
MOSubdue-II	19.93 (0.24)	40.64 (0.49)	87.76 (7.73)

Table 5.8 Run time in seconds for the different Subdue methods on the scientogram dataset for the two-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	661.72 (-)	1289.62 (-)	5674.45 (-)
Subdue-II	47.20 (15.01)	99.38 (32.27)	2876.70 (953.40)
MOSubdue-I	134.36 (-)	684.77 (-)	199.14 (-)
MOSubdue-II	181.76 (1.32)	**484.65 (4.14)**	217.25 (1.52)

From Tables 5.3 and 5.4, it can be seen how both single-objective Subdue variants, Subdue-I and II, have shown performance decrease with increase in *beamWidth*. Increase in *beamWidth* means more subgraph-seeds availaible for expansion. This will generate many repeated solutions (high redundancy) at the early stage of search in the subgraph lattice. Both Subdue-I and II have applied single-objective beam search in the subgraph lattice. Subdue-I uses the MDL-measure in equation (3.1) that constitutes the combination of the support and the size of the mined subgraphs, and Subdue-II applies a scalar function based on weighted addition of the support and the order of the mined subgraphs. This suggests single-objective beam search using these measures is unable to handle the selection pressure under the high

redundancy. As against, both variants of MOSubdue have shown improvement in the performance for increase in *beamWidth* = 10. This shows that the multiobjective beam search in MOSubdue can handle the selection pressure better than the single-objective beam search in Subdue-I and II. However, they have shown a decrease in the performance for further increase in *beamWidth*. This is because the multiobjective beam search considering these (the support and the order) objectives are somewhat unable to handle the selection pressure under such high redundancy. That configuration could be more beneficial for the case of having some additional objectives, as we will see in the following Section.

The analysis of the *HVR*-metric values reported in Table 5.3 corresponding to the chemical dataset reveals that Subdue-I and II have produced their best approximation corresponding to *beamWidth* = 5 for which the estimated *HVR*-metric value is 0.9729 and 0.8120, respectively. The best *HVR*-metric values produced by MOSubdue-I and II are equal to 0.9898 and 0.9662, respectively, corresponding to *beamWidth* = 10. The *HVR*-metric values on the scientogram dataset reported in Table 5.4 show that the best value of *HVR*-metric obtained by Subdue-I and II is 0.8545 and 0.1606, respectively, corresponding to *beamWidth* = 5. MOSubdue-I and II have produced their best value of *HVR*-metric equal to 0.8491 and 0.8968, respectively, corresponding to *beamWidth* = 10.

Finally, we compare the performance of the different Subdue methods on each dataset based on the *HVR*-metric values. From Table 5.6, on the chemical dataset MOSubdue-I has obtained the best value of *HVR*-metric equal to 0.9898. The second-best value of the *HVR*-metric equal to 0.9729 was obtained by Subdue-I. It was followed by MOSubdue-II and Subdue-II with *HVR*-metric values of 0.9662 and 0.8120, respectively. On the scientogram dataset (see 5.6), the best approximation was obtained by MOSubdue-II with a *HVR*-metric value of 0.8968. Subdue-I has produced the second-best approximation with a *HVR*-metric value of 0.8545. This performance was followed by MOSubdue-I and Subdue-II with *HVR*-metric values of 0.8491 and 0.1606, respectively.

Table 5.5 shows the *HVR*-metric values corresponding to three different runs of MOGaston on both datasets. On the chemical dataset, three different runs of the algorithm have been carried out with run times of 50, 100 and 250 seconds. Meanwhile, on the scientogram dataset, three different runs correspond to computational times of 485, 970 and 2425 seconds. The run time on the chemical dataset was estimated based on the execution time of 49.47 seconds required by MOSubdue-I in Table 5.7 to obtain the best value of *HVR*-metric equal to 0.9898 (see Tables 5.3 and 5.6). The run time on the scientogram dataset was estimated from the run time of 484.65 seconds in Table 5.8 required by MOSubdue-II to produce the best approximation with the *HVR*-metric value equal to 0.8968 (see Tables 5.4 and 5.6). The *HVR*-metric values in Table 5.5 for the approximation produced by the different runs of MOGaston show that the performance of the algorithm has improved for the higher run times, although the quality of the results is not very significant.

The analysis of run times reported in Tables 5.5 and 5.7 on the chemical dataset shows that Subdue-I and II have generated their best approximation with run time of 40.71 and 10.46 seconds, respectively (see Table 5.7). The run time required by MOSubdue-I and II to provide their best approximation was 49.47 and 40.64 seconds, respectively (see Table 5.7). From Table 5.5, MOGaston has obtained the best result corresponding to run time of 250 seconds. Compared to the Subdue methods, MOGaston has taken the highest run time. The result it has produced is the worst one to that of generated by any of the Subdue methods as reported in Table 5.6. The run time analysis among Subdue methods show that MOSubdue-I has taken the highest computational time of 49.47 seconds as compared to that taken by other Subdue methods to produce their best approximation, but MOSubdue-I has reported the best approximation with the *HVR*-metric value of 0.9898 as given in Table 5.6. Subdue-

II has taken the least run time of 10.46 to generate its best performance which is in fact the worst one among the Subdue methods, but it is better than that generated by MOGaston as shown in Table 5.6.

On the scientogram dataset, the run time analysis is based on the values reported in Tables 5.5 and 5.8. Subdue-I and II have reported their best approximation requiring a run time of 661.72 and 47.20 seconds, respectively, as given in Table 5.8. MOSubdue-I and II have generated their best performance for a computational time of 684.77 and 484.65 seconds, respectively (see Table 5.8). MOGaston required a run time of 970 seconds to produce the best *HVR*-metric value as reported in Table 5.5, but it happened to be the worst one among the Subdue methods, as can be seen from Table 5.6. Among Subdue methods, MOSubdue-II has generated the best performance with the *HVR*-metric value of 0.8968 (see Table 5.6), but with less computational time of 484.65 when compared to that of required by Subdue-I and MOSubdue-I. Again, Subdue-II has taken the least computational time to produce its best performance which is the worst one to that of the remaining Subdue methods, but it is better than that of MOGaston as can be seen from Table 5.6.

Overall comparison is based on the *HVR*-metric values reported in Table 5.6. We can say that MOSubdue-I has achieved the best performance on the chemical dataset with a value of 0.9892, and on the scientogram dataset, MOSubdue-II has produced the best approximation with a value of 0.8968. Figs. 5.1 and 5.2 show the graphical representation of the best approximation based on the *HVR*-metric value produced by each of the methods applied in this study. The graphical representation produced for Subdue-I, MOSubdue-I, and MOGaston that is corresponding to the approximation generated by the single-run of the algorithm. The plotted approximation corresponding to MOSubdue-II and Subdue-II is obtained as aggregation of the output of 10 and 11 different runs carried by the algorithms, respectively. On the chemical dataset, Subdue methods have been able to generate most of the solutions present in the true front, $\mathcal{PF}^*$. For order values > 20, all Subdue methods but MOSubdue-I have found some difficulty in producing the corresponding solutions. There are four solutions in $\mathcal{PF}^*$ with order values > 20. MOSubdue-I has managed to produce four solutions with order values > 20 which are close to the front $\mathcal{PF}^*$. As against, Subdue-I, Subdue-II, and MOSubdue-II could only find one solution each with order value higher than 20. In general, MOGaston could generate very few solutions as compared to those obtained by the Subdue methods. On the scientogram dataset, MOSubdue-II has shown the best spread of solutions with respect to that in the front $\mathcal{PF}^*$. There is a good spread of solutions generated by Subdue-I and MOSubdue-I, but they find it somewhat difficult to generate any solution with order value higher than 60. Subdue-II has produced the least number of solutions that can be comparable to that in the front $\mathcal{PF}^*$. MOGaston is not able to generate any solution present in the front $\mathcal{PF}^*$ as can be seen from Fig. 5.2.

5.4 Analysis of Results for the Three-objective Subgraph Mining Task

Tables 5.9 to 5.12 report the HVR-metric values for the approximation produced by the different methods on both datasets for the three- objective subgraph mining problem. Tables 5.9 and 5.10 represent the mean and standard deviation values corresponding to MOSubdue-II and Subdue-II using the 10 and 13 runs, respectively. Table 5.11 shows the HVR-metric values corresponding MOGaston on both datasets. In this table, the values in the brackets represent again computational time corresponding to each run of MOGaston. Tables 5.13 and 5.14 report the computational time for the different Subdue methods on each dataset.

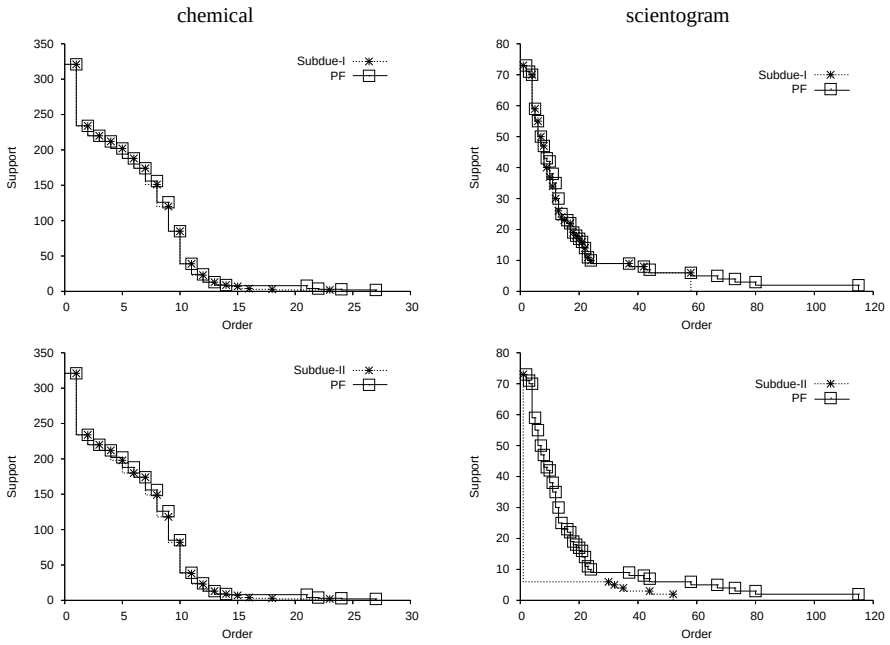

Fig. 5.1 A graphical representation of solutions corresponding to the best approximations produced by the different methods with respect to the *HVR*-metric for two objective subgraph mining task. The pseudo Pareto-optimal front is also shown as a reference.

Table 5.9 The *HVR*-metric values for Pareto-optimal sets obtained by the different Subdue methods on the chemical dataset for the three-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	0.9708 (-)	0.9609 (-)	0.9635 (-)
Subdue-II	0.6975 (0.3696)	0.2001 (0.2575)	0.0148 (0.0181)
MOSubdue-I	0.9715 (-)	0.9822 (-)	0.9879 (-)
MOSubdue-II	0.9759 (0.0052)	0.978 (0.0022)	**0.9892 (0.0009)**

When comparing the performance among Subdue methods on both datasets using the results in Tables 5.9 and 5.10, it can be seen how Subdue-I has shown a performance decrease with the increase in *beamWidth*. This is in line with the behavior on the two-objective subgraph mining task (see Tables 5.3 and 5.4). Subdue-II has shown a very significant decrease in the performance on the chemical dataset for the increase in *beamWidth*, but it has shown some small improvement in the performance on the scientogram dataset when *beamWidth* was increased to 10. On the other hand, both variants of MOSubdue have shown a better handling of the selection pressure (by using three objectives, the support, the order and the density of the mined subgraph) under the redundancy created by increasing *beamWidth*. On the chemical dataset, MOSubdue variants have shown the best performance corresponding

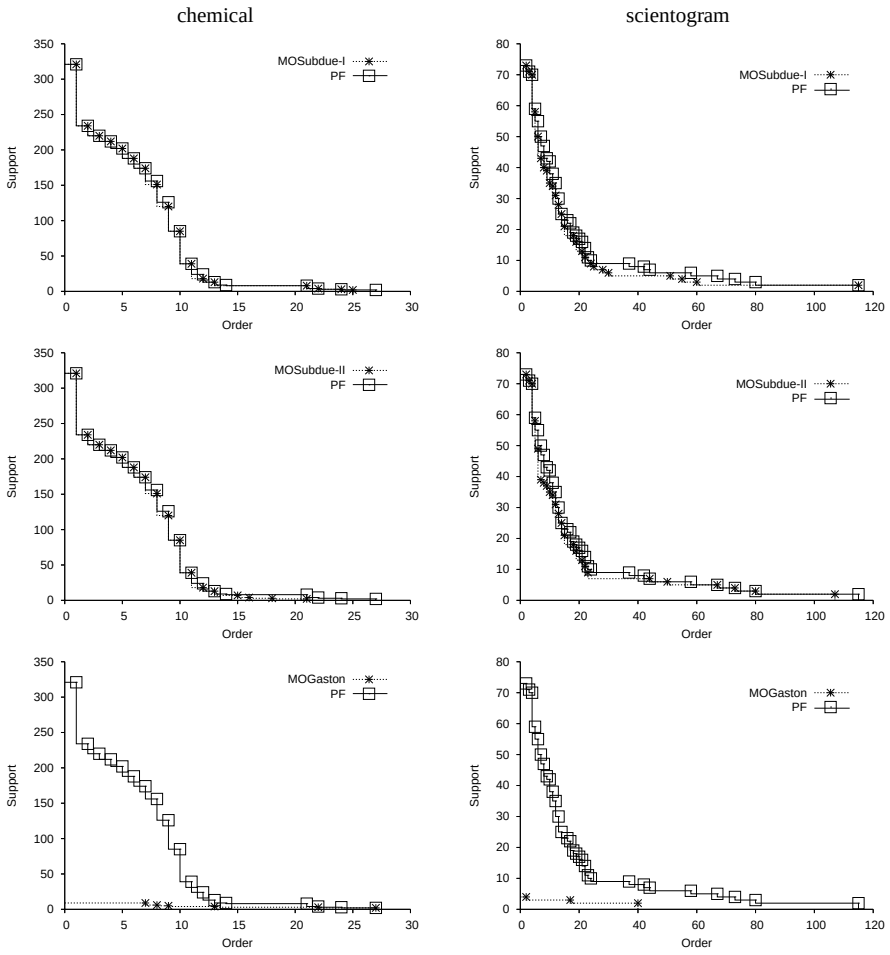

Fig. 5.2 A graphical representation of solutions corresponding to the best approximations produced by the different methods with respect to the *HVR*-metric for two objective subgraph mining task. The pseudo Pareto-optimal front is also shown as a reference.

Table 5.10 The *HVR*-metric values for Pareto-optimal sets obtained by the different Subdue methods on the scientogram dataset for the three-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	0.7334 (-)	0.5315 (-)	0.4815 (-)
Subdue-II	0.0368 (0.0010)	0.0620 (0.1002)	0.0341 (0.0756)
MOSubdue-I	0.9482 (-)	0.9486 (-)	0.9209 (-)
MOSubdue-II	0.9508 (0.0036)	**0.9540 (0.0028)**	0.9356 (0.0171)

Table 5.11 *The HVR*-metric values for Pareto-optimal sets obtained by MOGaston using three different run times on both datasets for the three-objective subgraph mining task. The numbers in the brackets represent the run times in seconds.

DatasetDataset	Run 1Run 1	Run 2Run 2	Run 3Run 3
Chemical	0.0463 [90]	0.0463 [180]	0.5062 [450]
Scientogram	0.0615 [587]	0.0617 [1174]	0.0617 [2935]

Table 5.12 *The HVR*-metric values corresponding to the best result produced by the different methods on both datasets for the three-objective subgraph mining task.

Datasets	Subdue-I	Subdue-II	MOSubdue-I	MOSubdue-II	MOGaston
Chemical	0.9708	0.6975	0.9879	**0.9892**	0.5062
Scientogram	0.7334	0.0620	0.9486	**0.9540**	0.0617

Table 5.13 Run time in seconds for the different Subdue methods on the chemical dataset for three-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	40.71 (-)	79.20 (-)	165.53 (-)
Subdue-II	10.36 (5.14)	15.78 (7.79)	27.17 (13.37)
MOSubdue-I	19.80 (-)	48 (-)	89.38 (-)
MOSubdue-II	19.13 (1.82)	42.46 (3.73)	**90.28 (7.69)**

Table 5.14 Run time in seconds for the different Subdue methods on the scientogram dataset for three-objective subgraph mining task. The numbers in the parentheses represent the standard deviation.

	beamWidthbeamWidth		
MethodsMethods	55	1010	2020
Subdue-I	661.72 (-)	1289.62 (-)	5674.45 (-)
Subdue-II	40.60 (22.05)	85.10 (47.27)	2438.18 (1388.57)
MOSubdue-I	132.55 (-)	681.10 (-)	197.50 (-)
MOSubdue-II	262.29 (48.17)	**587.28 (88.42)**	587.98 (603.86)

to *beamWidth* = 20. On the scientogram dataset, they have improved their performance corresponding to *beamWidth* = 10, but they have shown a little drop in the performance when *beamWidth* was further increased to 20. However, this performance drop was minimal as compared to that suffered by Subdue-I and II methods.

Comparison of the *HVR*-metric values reported in Table 5.9 corresponding to the chemical dataset shows that the best approximation obtained by Subdue-I and II has *HVR*-metric values of 0.9708 and 0.6975, respectively, corresponding to *beamWidth* = 5. Both MOSubdue variants have generated their best approximation corresponding to *beamWidth* = 20 that has produced *HVR*-metric values of 0.9879 and 0.9892, respectively. On the sci-

entogram dataset, Subdue-I and II have generated their best performance corresponding to *beamWidth* = 5 and 10 with the *HVR*-metric value equal to 0.7334 and 0.0620, respectively. When we carry out a detailed analysis of the *HVR*-metric values produced by the different Subdue methods on the chemical dataset, it reveals that MOSubdue-II has produced the best performance with the *HVR*-metric equal to 0.9892 corresponding to *beamWidth* = 20. MOSubdue-I has generated the second-best performance with the *HVR*-metric value of 0.9879 with respect to *beamWidth* = 20. The third-best result has shown by Subdue-I and the last one was Subdue-II with *HVR*-metric values of 0.9708 and 0.6975, respectively, corresponding to *beamWidth* = 5. On the scientogram dataset, the *HVR*-metric values reported in Table 5.10 show that MOSubdue-II has again secured the best performance with a *HVR*-metric value of 0.9540 corresponding to *beamWidth* = 10. The second-best performance has been produced by MOSubdue-I with the *HVR*-metric value of 0.9486 corresponding to *beamWidth* = 10. This has followed by Subdue-I and Subdue-II with *HVR*-metric values of 0.7334 and 0.0620 corresponding, to *beamWidth* = 5 and 10, respectively.

Table 5.11 shows the *HVR*-metric values associated to three different runs of MOGaston on both datasets. On the chemical dataset, MOGaston has carried out three runs with run times of 90, 180, and 450 seconds. On the scientogram dataset, the algorithm has performed three runs with execution times of 587, 1174, and 2935 seconds. The run time on the chemical dataset was based on the run time of 90.28 seconds corresponding to the best *HVR*-metric value (0.9892, see Table 5.12) produced by MOSubdue-II (see Table 5.13). The run time on the scientogram dataset was estimated from the time 587.28 seconds as reported in Table 5.14 corresponding to MOSubdue-II to generate the best approximation (0.9540, see Table 5.12). The *HVR*-metric values in Table 5.11 indicate that MOGaston will need a significantly larger amount of run time to produce good approximations.

The run time analysis of different methods is given in Tables 5.11, 5.13, and 5.14. The analysis on the chemical dataset reveals that Subdue-I and II have produced their best approximation with run times of 40.71 and 10.36 seconds, respectively (see Table 5.13). The run times for MOSubdue-I and II to obtain their best performance were 89.38 and 90.28 seconds, respectively (see Table 5.13). MOGaston has produced its best performance for a given run time of 450 seconds. In comparison to the run time required by each of the Subdue methods to produce their best approximation, MOGaston has taken much more time and produced a significantly worst approximation as can be seen from Table 5.12. The comparison of run times for the different Subdue methods shows that MOSubdue-II has taken the highest run time of 90.28 seconds to produce its best performance that happens to be the best approximation than obtained by any applied methods (see Table 5.12). The least run time of 10.36 was required by Subdue-II to obtain its best approximation (0.6975), which is the worst one among the Subdue methods, but it is better than that of MOGaston (0.5062, see Table 5.12).

On the scientogram dataset, the run time analysis based on the values reported in Tables 5.11 and 5.14 show that Subdue-I and II have achieved their best performance requiring run times of 661.72 and 85.10 seconds, respectively (see Table 5.14). MOSubdue-I and II have obtained their best performance for run times of 681.10 and 587.28 seconds, respectively (see Table 5.14). The best performance of MOGaston has been that corresponding to run time of 1174 seconds as reported in Table 5.11. Once again, MOGaston has taken the highest run time and produced the worst performance as can be seen from Table 5.12. The run time analysis among the Subdue methods reveals that MOSubdue-II has produced the best result with *HVR*-metric value of 0.9540 as shown in Table 5.12 using a run time of 587.28 as reported in Table 5.14. This run time is less than that required by Subdue-I and MOSubdue-II to produce their best performance. Subdue-II has taken the least computa-

tional time (85.10 seconds, see Table 5.14) to produce its its best results (0.0620), which is the worst one among the Subdue methods, but it is still slightly better than that of MOGaston (0.0617, see Table 5.12).

The final comparison is based on the *HVR*-metric values reported in Table 5.12 corresponding to the best approximation produced by each method. On both datasets, MOSubdue-II has produced the best approximation with *HVR*-metric values of 0.9892 and 0.9540, respectively. On both datasets, the results show that tackling a more complex problem with three objectives has enabled the multiobjective beam search to handle the selection pressure far better as compared to that by single-objective beam search under the huge redundancy in the candidate subgraph list. Figs. 5.3 and 5.4 show the three-dimensionalMieres plotting of solutions corresponding to the best approximations produced by Subdue-I, MOSubdue-I, and MOGaston, as well as to the aggregated set of solutions obtained from the different runs of MOSubdue-II and Subdue-II. The performance advantage of both MOSubdue variants with respect to the remaining algorithms can be clearly observed.

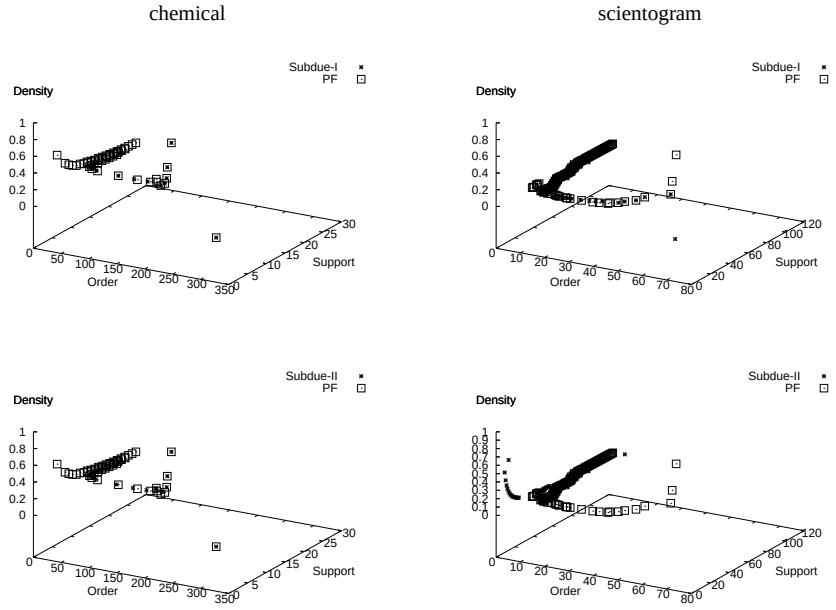

Fig. 5.3 A graphical representation of solutions corresponding to the best approximations produced by the different methods with respect to the *HVR*-metric for the three- objective subgraph mining task. The pseudo Pareto-optimal front is also shown as a reference.

We summarize that the proposed multiobjective beam search methods, i.e., both MO-Subdue variants, have outperformed the single-objective beam search (i.e., Subdue-I and II) and the exhaustive search, i.e., MOGaston, on both datasets. In particular, MOSubdue-II that has applies diversified selection has shown better performance than MOSubdue-I. This indicates that an application of multiobjective beam search for subgraph-seeds generation has indeed guided the Subdue algorithm to find Pareto-optimal subgraphs in the subgraph lattice space in a proper way. Besides, it should be noticed that the Pareto dominance-based

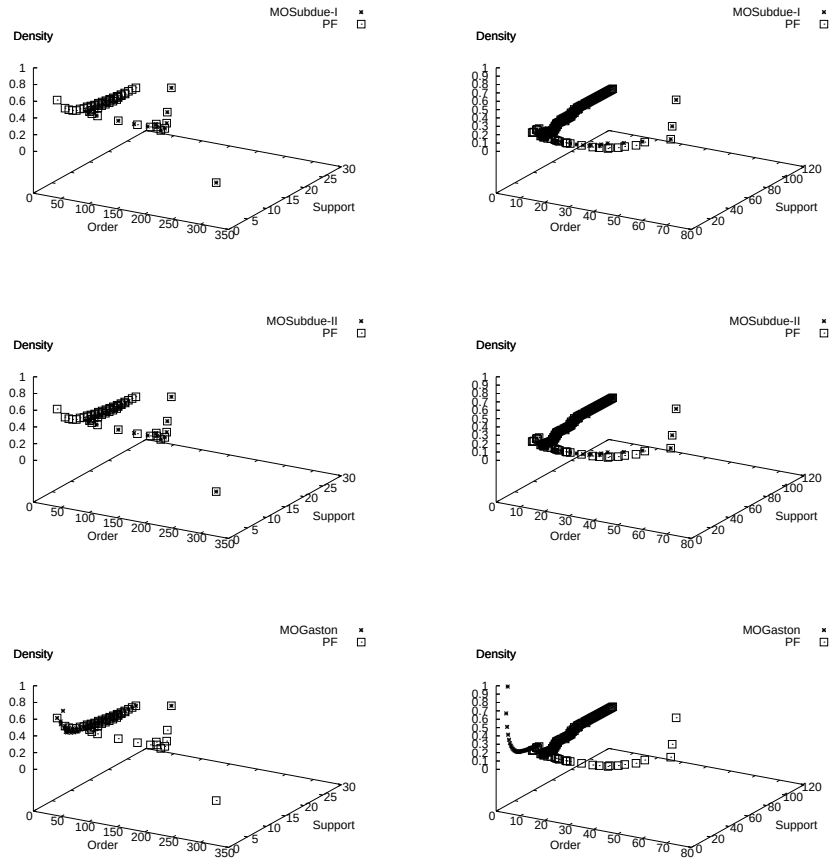

Fig. 5.4 A graphical representation of solutions corresponding to the best approximations produced by the different methods with respect to the *HVR*-metric for the three- objective subgraph mining task. The pseudo Pareto-optimal front is also shown as a reference.

computation introduced in the Subdue algorithm only adds a small complexity of $O(dK^2)$ to the theoretical complexity of standard Subdue [6].

6 Concluding Remarks and Future Work

We have proposed the use of a Pareto dominance-based search strategy for multiobjective subgraph mining in relational graph databases. The approach has been customized using the Subdue algorithm and has been called as MOSubude (Multi-Objective subgraph mining by Subdue). Two different MOSubdue variants based on the selection of subgraph-seeds for further expansion were proposed.

The performance of MOSubdue has been analyzed using two real-world datasets, and taking the original single-objective Subdue, a weighted preference-based Subdue, and a multiobjective extension of the exhaustive Gaston algorithm as baselines. From the obtained results, we have found that MOSubdue is able to discover a set of Pareto-optimal subgraphs in a single run. The performance has been tested by computing the proximity of the generated Pareto-optimal set to the pseudo Pareto-optimal set produced by combining every result obtained by any considered algorithm. For this purpose, we have employed the hypervolume measure commonly applied in EMO studies. It is evident from the experiments developed that our proposal is clearly able to perform multiobjective subgraph search in the subgraph lattice space, and that it is able to generate approximations to the Pareto-optimal front far better than obtained by the considered baseline methods. Both MOSubdue approaches were able to obtain the Pareto-optimal set of subgraphs that show good diversity and closeness to the Pareto-optimal front of the input graph dataset. In particular, for the two-objective subgraph mining problem, MOSubdue-I showed the best performance on the chemical dataset, while MOSubdue-II did so on the scientogram dataset. For three-objective subgraph mining problem, MOSubdue-II was the best performer on both datasets.

Several ideas for future developments arise from this work. On the one hand, at the start of the search process, both variants of the MOSubdue algorithm initially have a large number of subgraphs belonging to the first Pareto front, and the size of that front decreases as the search progresses. This is due to the initialization of the subgraph-seeds with the single node subgraphs and the application of a constructive search Subdue applies. MOSubdue implements the multiobjective beam search with a fixed and constant *beamWidth* value that discards some of the less promising subgraphs at the early stage of expansion, thereby terminating the possibility of expanding these less promising subgraphs later in order to search other promising subgraph search space regions. Thus, beam search is a kind of heuristic search in the state space of subgraphs lattice not allowing backtracking and hence may often end up performing local search with suboptimal results. One possible solution for this undesired behavior could be to use an adaptive *beamWidth*. We would initially keep a high value of *beamWidth* and decrease it using some adaptation scheme as the search progresses. This will allow exploring more subgraphs at the beginning of the search process, when the first Pareto front is more populated, in order to discriminate the selection procedure in a more aggressive way. On the other hand, the same problem could also be solved by considering an implementation of pure EMO method to directly perform multiobjective subgraph search in the subgraph lattice. This is due to the fact that an EMO algorithm would maintain a population of subgraph-seeds and perform the subgraph-growth at different tree levels to generate Pareto-optimal subgraphs. We aim to design such an EMO-based GBDM method.

MOSubdue applies a diversification-based subgraph selection procedure that computes the diversity of solutions in the objective space. However, it does not take into consideration the diversity of solutions in the solution space. Application of structural diversity of subgraphs will help to generate subgraph-seeds that are different from each other in the solution space and thus will enable to explore the different regions of the multiobjective subgraph search space. One such solution selection procedure that computes the diversity of subgraph using the objective space as well as the solution space has been recently applied in [43]. As a future study, this procedure can be applied for the diversified subgraph selection in MOSubdue.

The Pareto dominance-based evaluation and search can be applied with algorithms which work on a set of subgraphs to generate new subgraphs. Besides, this approach may be utilized in subgraph mining algorithms, such as those in [19, 26] whose subgraph-growth method performs merging subgraph-seeds to generate candidate subgraphs. In such subgraph-

growth methods, the selection of d-objective vector subgraph-seeds can be done using the Pareto dominance-based approach to generate candidate subgraphs for further expansion. We will consider the latter as future extensions of our methodology.

Acknowledgements 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.

References

1. C. Aggarwal and H. Wang, editors. *Managing and Mining Graph Data Series*. Springer, 2010.
2. R. Baños, C. Gil, M.G. Montoya, and J. Ortega. A new Pareto-based algorithm for multi-objective graph partitioning. In C. Aykanat, T. Dayar, and I. Körpeoglu, editors, *Computer and Information Sciences - ISCIS 2004*, volume 3280 of *Lecture Notes in Computer Science*, pages 779–788. Springer Berlin / Heidelberg, 2004.
3. C. Borgelt and M.R. Berthold. Mining molecular fragments : Finding relevant substructures of molecules. In *Proc IEEE Int Conf Data Min (ICDM'02)*, pages 51–58, 2002.
4. V. Chankong and Y. Y. Haimes. *Multiobjective Decision Making Theory and Methodology*. North-Holland, Amsterdam, 1983.
5. C. A. Coello, G. B. Lamont, and D. A. Van Veldhuizen. *Evolutionary Algorithms for Solving Multi-objective Problems*. Springer, Berlin, 2007.
6. D.J. Cook and L.B. Holder. Graph-based data mining. *IEEE Intell Syst*, 15:32–41, 2000.
7. D.J. Cook and L.B. Holder, editors. *Mining Graph Data*. Wiley, London, 2007.
8. D.J. Cook, L.B. Holder, and S. Djoko. Scalable discovery of informative structural concepts using domain knowledge. *IEEE Expert: Intelligent Systems and Their Applications*, 11:59–68, 1996.
9. D.W. Dearholt and R.W. Schvaneveldt. Properties of Pathfinder networks. In R. Schvaneveldt, editor, *Pathfinder Associative Networks: Studies in Knowledge Organization*, pages 1–30. Ablex Publishing Corporation, 1990.
10. K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Trans Evol Comput*, 6:182–197, 2002.
11. T. Falkowski, A. Barth, and M. Spiliopoulou. Dengraph: A density-based community detection algorithm. In *IEEE/WIC/ACM Int Conf Web Intelligence*, pages 112–115, Los Alamitos, CA, USA, 2007. IEEE Computer Society.
12. I. Fischer and T. Meinl. Graph based molecular data mining - An overview. In W. Thissen, P. Wieringa, M. Pantic, and M. Ludema, editors, *IEEE Int Conf Syst Man Cy*, volume 76, pages 4578–4582, 2004.
13. C.M. Fonseca and P.J. Fleming. Genetic algorithms for multiobjective optimization: Formulation, discussion and generalization. In *Proc 5th Int Conf Genetic Algorithms (ICGA93)*, pages 416–423, 1993.
14. T. Gal, T.J. Stewart, and T. Hanne, editors. *Multicriteria Decision Making: Advances in MCDM Models, Algorithms, Theory and Applications*. Kluwer Academic, Dordrecht, 1999.
15. J.A. Gonzalez, L.B. Holder, and D.J. Cook. Structural knowledge discovery used to analyze earthquake activity. In *Proc 13th Ann Florida Art Intell Res Symp (FLAIRS)*, pages 86–90, 2000.
16. L. B. Holder and D. J. Cook. Graph-based data mining. In J. Wang, editor, *Encyclopedia of Data Warehousing and Mining, Vol. II*, pages 943–949. Information Science Reference, Hershey, 2005.
17. H. Hu, X. Yan, Y. Huang, J. Han, and X.J. Zhou. Mining coherent dense subgraphs acrossmassive biological networks for functional discovery. *Bioinformatics*, 21(1):i213–i221, 2005.
18. J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *Proc IEEE Int Conf Data Min (ICDM'03)*, pages 549–552, 2003.
19. A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *Proc 4th Euro Conf Prin Data Min Knowl Disc (PKDD'00)*, pages 13–23, 2000.
20. H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. In *Proc. IEEE Congr Evol Comput*, pages 2424–2431, 2008.
21. Y. Jin, editor. *Multi-Objective Machine Learning*. Springer-Verlag, New York, 2006.
22. Y. Jin and B. Sendhoff. Pareto-based multi-objective machine learning: An overview and case studies. *IEEE T Syst Man Cy C*, 38:397–415, 2008.
23. U. Kang, C. Tsourakakis, and C. Faloutsos. Pegasus: Mining peta-scale graphs. *Know Inf Syst*, 27:303–325, 2011.
24. R.I. Kondor and J.D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *Proc. 19th Int Conf Machine Learning, (ICML'02)*, pages 315–322, 2002.

25. J. Kukluk, L.B. Holder, and D.J. Cook. Learning node replacement graph grammars in metabolic pathways. In *Proc Int Conf Bioinformat & Comput Biol (BIOCOMP-07)*, pages 44–50, 2007.
26. M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Trans Knowl Data Eng*, 16:1038–1051, 2004.
27. B. Long, Z. Zhang, and P. Yu. A general framework for relation graph clustering. *Knowl Inf Syst*, 24(3):393–413, 2010.
28. B. T. Lowerre. *The HARPY speech recognition system*. PhD thesis, Carnegie Mellon University, Pittsburgh, 1976.
29. T. Matsuda, T. Horiuchi, H. Motoda, and T. Washio. Extension of graph-based induction for general graph structured data. In T. Terano, H. Liu, and A.L.P. Chen, editors, *Proc. 4th Pacific-Asia Conf Know Dis Data Mining (PAKDD'00)*, volume 1805 of *Lecture Notes in Computer Science*, pages 420–431. Springer Berlin / Heidelberg, 2000.
30. F. De Moya-Anegón, B. Vargas-Quesada, V. Herrero-Solana, Z. Chinchilla-Rodríguez, E. Corera-Álvarez, and F. J. Munoz-Fernández. A new technique for building maps of large scientific domains based on the cocitation of classes and categories. *Scientometrics*, 61(1):129–145, 2004.
31. A. Narasimhamurthy, D. Greene, N. Hurley, and P. Cunningham. Partitioning large networks without breaking communities. *Know Inf Syst*, 25:345–369, 2010.
32. S. Nijssen and J.N. Kok. A quickstart in frequent structure mining can make a difference. In *Proc 10th ACM SIGKDD Int Conf Knowl Disc & Data Min (KDD'04)*, pages 647–652, 2004.
33. S. Nijssen and J.N. Kok. Frequent subgraphs: Runtimes don't say everything. In *Proc 4th Int Conf Mining Learn Graphs (MLG'06)*, pages 173–180, 2006.
34. C. Noble and D. Cook. Graph-based anomaly detection. In *Proc 9th ACM SIGKDD Int Conf Knowl Disc Data Mining*, pages 631–636, 2003.
35. A.N. Papadopoulos, A. Lyritsis, and Y. Manolopoulos. SkyGraph: An algorithm for important subgraph discovery in relational graphs. *Data Min Knowl Disc*, 17:57–76, 2008.
36. W. Peng and T. Li. Temporal relation co-clustering on directional social network and author-topic evolution. *Knowl Inf Syst*, 26(3):467–486, 2011.
37. RC Purshouse and PJ Fleming. On the evolutionary optimisation of many conflicting objectives. *IEEE Trans Evol Comput*, 11(6):770–784, 2007.
38. T. Qian, J. Srivastava, Z. Peng, and P. Sheu. Simultaneously finding fundamental articles and new topics using a community tracking method. In T. Thanaruk, K. Boonserm, C. Nick, and H. Tu-Bao, editors, *Advances in Knowledge Discovery and Data Mining*, volume 5476 of *Lecture Notes in Computer Science*, pages 796–803. Springer Berlin / Heidelberg, 2009.
39. A. Quirin, Ó. Cordón, V. P. Guerrero-Bote, B. Vargas-Quesada, and F. De Moya-Anegón. A quick MST-based algorithm to obtain Pathfinder networks. *J Am Soc Inf Sci Technol*, 59(12):1912–1924, 2008.
40. A. Quirin, Ó. Cordón, B. Vargas-Quesada, and F. Moya-Anegon. Graph-based data mining: A new tool for the analysis and comparison of scientific domains represented as scientograms. *J Informetr*, 4(3):291–312, 2010.
41. S. Ranu and A.K. Singh. Graphsig: A scalable approach to mining significant subgraphs in large graph databases. In *Proc 25th Int Conf Data Engg (ICDE'09)*, pages 844–855. IEEE, 2009.
42. J. Rissanen. *Stochastic Complexity in Statistical Inquiry Theory*. World Scientific Publishing Co., Inc., River Edge, 1989.
43. R. C. Romero-Zaliz, C. Rubio-Escudero, J.P. Cobb, F. Herrera, , and I. Zwir. A multiobjective evolutionary conceptual clustering methodology for gene annotation within structural databases: A case of study on the gene ontology database. *IEEE Trans Evol Comput*, 12(6):679–701, 2008.
44. P. Shelokar, A. Quirin, and Ó. Cordón. A multiobjective variant of the subdue graph mining algorithm based on the NSGA-II selection mechanism. In *Proc IEEE Congr Evol Comput (CEC'10)*, pages 463–470, 2010.
45. N. Shrivastava, A. Majumder, and R. Rastogi. Mining (Social) Network Graphs to Detect Random Link Attacks. In *IEEE 24th Int Conf Data Engg (ICDE'08)*, pages 486–495, 2008.
46. C. Tsourakakis. Counting triangles in real-world networks using projections. *Knowl Inf Syst*, 26(3):501–520, 2011.
47. B. Vargas-Quesada and F. De Moya-Anegón. *Visualizing the Structure of Science*. Springer-Verlag New York, Secaucus, 2007.
48. X. Yan and J. Han. gSpan: Graph-based substructure pattern mining. In *Proc IEEE Int Conf Data Min (ICDM'02)*, pages 721–724, 2002.
49. X. Yan and J. Han. CloseGraph: Mining closed frequent graph patterns. In *Proc 9th ACM SIGKDD Int Conf Knowl Disc & Data Min (KDD'03)*, pages 286–295, 2003.
50. Q. Yang and X. Wu. 10 challenging problems in data mining research. *Int J Inf Tech Decis*, 5:597–604, 2006.

51. F. Zhu, X. Yan, J. Han, and P.S. Yu. gPrune: A constraint pushing framework for graph pattern mining. In *Proc PAKDD Conf*, pages 388–400, 2007.
52. E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: A comparative case study and the strength Pareto approach. *IEEE Trans Evol Comput*, 3:257–271, 1999.
53. E. Zitzler, L. Thiele, and K. Deb. Comparison of multiobjective evolutionary algorithms: Empirical results. *IEEE Trans Evol Comput*, 8:173–195, 2000.
54. E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. da Fonseca. Performance assessment of multiobjective optimizers: An analysis and review. *IEEE Trans Evol Comput*, 7:117–132, 2003.

Authors Biographies

Prakash Shelokar is currently a Post-Doctoral Researcher at the European Centre for Soft Computing, Mieres, Spain. He earned his PhD(2009) at the University of Pune, India. He received the *Juan de La Cierrva* grant for young researcher from the Spanish Ministry of Science and Innovation (MICINN) (2010-2013). Starting with natured-inspired techniques for optimization, his interests moved to machine learning ranging from rule-based classification, clustering to feature selection, then to multiobjective graph-based data mining. He has served program and technical review committees of major international conferences ICSI2010, ISDA2009, ISDA2006. He has published around 27 peer-reviewed scientific papers. His publications have received over 375 citations, carrying an h-index of 6 as on October 2011. Some of his papers have featured in Top25-Hottest Articles published by Computers & Chemical Engineering (2004), Applied Mathematics & Computation for 3 years. Wiley interscience in its Operations Research Volume (June 2005) has featured his paper under hot articles in International Journal of Quality & Reliability Engineering journal.

Arnaud Quirin is a Post-Doctoral Researcher since the founding of the European Centre for Soft Computing, Mieres, Spain, in 2006. He received his M.S. degree (2002) and his Ph.D. (2005) in Computer Science from the University Louis Pasteur of Strasbourg, France, where he has been a teaching/researcher assistant until September 2006.

Dr. Quirin has published more than 30 peer-reviewed scientific publications, including 3 book chapters and 8 JCR-SCI-indexed journal papers, and he is a reviewer for 6 international journals. He was involved in several national projects, private contracts and one European project, related to the application of evolutionary algorithms to image classification.

His current main research interests are in the fields of evolutionary algorithms, multiobjective graph-based mining, fuzzy-based multiclassifiers, and genetic-fuzzy systems.

Óscar Cordón is Professor with the University of Granada (UGR), Spain. Founder of its Virtual Learning Center (2001-2005) and founding researcher of the European Centre for Soft Computing, Spain (2006-2011). IEEE Senior Member with many representative positions at IEEE Computational Intelligence Society (CIS) and EUSFLAT. UGR Young Researcher Award (2004), IEEE CIS Outstanding Early Career Award (2011, the first such award conferred), and IFSA Award for Outstanding Applications of Fuzzy Technology (2011). Advisor of the 2011 EUSFLAT Best Ph.D. Thesis Awardee, Dr. Ibañez.

He has published 280 scientific publications including a research book and 64 JCR-SCI-indexed journal papers, with 1640 citations (October 2011, h index 23, 1% of most-cited researchers in the world). He advised 13 Ph.D. dissertations, participated in 30 research projects/contracts (coordinating 15), and coedited 8 journal special issues. He has an approved international patent and is Editor of 10 international journals (IEEE TFS Outstanding Associate Editor, 2008).