Computers & Operations Research 64 (2015) 60–70

Contents lists available at ScienceDirect

Computers & Operations Research

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

Dimensionality reduction in multiobjective shortest path search

Francisco-Javier Pulido *, Lawrence Mandow, José-Luis Pérez-de-la-Cruz

Dpto. Lenguajes y Ciencias de la Computación, Universidad de Málaga, Campus de Excelencia Internacional Andalucia Tech, Boulevar Louis Pasteur, Málaga 35. 29071, Spain

ARTICLE INFO

Available online 10 June 2015

Keywords:
Combinatorial optimization
Multiobjective shortest path problem
Exact label-setting algorithms
Lower bounds

ABSTRACT

One-to-one multiobjective search in graphs deals with the problem of finding all Pareto-optimal solution paths between given start and goal nodes according to a number of distinct noncommensurate objectives. The problem is inherently more complex than single objective graph search. Time requirements are dominated by the facts that (a) many different non-dominated labels may need to be explored for each node; (b) each new label under consideration must be checked for dominance against various subsets of previously generated labels. This paper describes how a dimensionality reduction technique can be applied to exact label-setting algorithms, reducing the number of dominance checks and allowing for much faster multiobjective search. The technique is applied to NAMOA*, a state of the art exact label-setting multiobjective search algorithm, achieving reductions in time requirements of more than an order of magnitude over problems in random grids and realistic road maps. Tests include problems with three, four, and five objectives.

1. Introduction

Multiobjective (MO) shortest path problems arise in many fields, such as vehicle path planning [17,39,40], urban transportation networks [6,9], robot surveillance [7], satellite scheduling [11], routing in telecommunication networks [5], and route planning in different contexts [1,3,8,16].

The multiobjective shortest path problem involves finding the set of all Pareto-optimal solution paths in a graph from a given start node to a designated goal node (one-to-one problem), or to all other nodes in the graph (one-to-all problem), according to a number of distinct noncommensurate objectives. Several approaches have been proposed to solve this problem. These include enumerative approaches (label-setting and label-correcting), ranking algorithms, and two-phase algorithms (e.g. see [4,2,34,35,12,37,31,5]). For the one-to-one problem, where single start and goal nodes are designated in the graph, label-setting algorithms are generally the best option for arbitrary start–goal pairs in large graphs, e.g. navigation queries in road maps. Multiobjective label-setting algorithms are generally extensions of single objective ones, and particularly of Dijkstra's algorithm and *A** [14]. The latter is a variant of the former for one-to-one problems that uses distance estimates to the goal to improve search efficiency. If these estimates are lower bounds, then *A** is an exact algorithm, i.e. it always returns an optimal solution. The one-to-one version of Dijkstra's algorithm is a particular case of *A** with lower bounds equal to zero.[1]

Hansen [13] first extended Dijkstra's algorithm to the biobjective problem, and showed that even with two objectives the number of Pareto-optimal solution paths can grow exponentially with solution depth in the worst case. However, there are interesting classes of MO search problems where this worst-case behavior does not appear [25,28]. Martins [27] provided an exact label-setting algorithm for the general case, i.e. each label scanned by the algorithm is Pareto-optimal. Martins proposed selecting the lexicographically smallest label from the set of pending labels at each step, in order to guarantee that a Pareto-optimal label is always selected. Other authors have proposed selecting the minimum according to a linear aggregate function, which has been reported to be more efficient in practice [15,20]. Several extensions of *A** to the multiobjective case have also been proposed. These include MOA* [36], TC [38], and NAMOA* [26]. The latter has been proven to be an exact algorithm when provided with lower bound estimates, and to explore an optimal number of labels. Additionally, it has been formally shown that MOA* does not always explore

* Corresponding author. Tel.: +34 952 132863.
E-mail addresses: francis@lcc.uma.es (F.-J. Pulido), lawrence@lcc.uma.es (L. Mandow), perez@lcc.uma.es (J.-L. Pérez-de-la-Cruz).

http://dx.doi.org/10.1016/j.cor.2015.05.007
0305-0548/

[1] The distance estimate functions (used by *A**) are called *heuristic* functions in the Artificial Intelligence literature (e.g. see [10,29]). However, in this paper we adopt the Operations Research terminology, where the term "heuristic" is used to designate approximate algorithms. Notice that when distance estimates are lower bounds, the algorithms considered in this paper (*A** and NAMOA*) are in fact exact algorithms.

an optimal number of labels [30]. Tung and Chew also proposed using the ideal point of each node n as a lower bound estimate of the cost of all paths from n to the goal [38]. These ideal points can be efficiently calculated performing q one-to-all single objective searches from the goal node in a graph identical to the original one, but with all arcs reversed (where q is the number of objectives, and each search optimizes a different objective). Using these lower bounds, NAMOA* was shown to outperform the other MO extensions of A* over problems in random grids and realistic road maps [21,18] with two objectives.

However, even in this case, time requirements are still an important limiting factor in the size of problems that exact algorithms can solve in practice [19,21,23]. This paper describes a dimensionality reduction technique that speeds up the time performance of multiobjective label-setting search. Dimensionality reduction was first proposed as a space saving technique in the development of *vector frontier search* [24,25], a multiobjective exact label-setting algorithm that achieved important reductions in space requirements at the expense of increasing time requirements. This paper extends this technique, showing that it can be applied, under reasonable assumptions, to best-first label-setting algorithms (and to NAMOA* in particular). Results show improvements in time performance. Furthermore, the use of this technique preserves all the properties of NAMOA*. More precisely, it remains an exact algorithm when provided with lower bound estimates. In other words, the proposed technique reduces the time requirements of NAMOA*, extending the size of MO problems that can be solved exactly in practice.

This paper is organized as follows. Section 2 reviews important concepts in MO search. Section 3 describes the dimensionality reduction technique, and proves its correctness for MO search with lower bounds under reasonable assumptions. An experimental study is described in Section 4, and its results are discussed in Section 5. Finally, some conclusions and future work are outlined.

2. Multiobjective search

The multiobjective search problem can be stated as follows: let G be a locally finite directed graph $G=(N,A,\vec{c})$, of N nodes, and $A=\{(i_i,j_1),\ldots,(i_m,j_m)\}\subseteq N\times N$ arcs, where q positive costs $\vec{c}_{ij}=(c_{ij}^1,\ldots,c_{ij}^q)\in\mathbb{R}^{q+}$ are associated with each arc $(i,j)\in A$. Sometimes we will denote $\vec{c}_{ij}$ as $\vec{c}(i,j)$.

Let a path from node n_1 to node n_k be a sequence of nodes $(n_1,n_2,\ldots n_k)$ such that $\forall i<k\ (n_i,n_{i+1})\in A$. Let the cost vector of a path be defined as the sum of the cost vectors of its component arcs.

Definition 1 (*Adapted from [31]*). The multiobjective shortest path problem consist in finding the set of all *non-dominated* cost paths in G, with source node $s\in N$ and target node $\gamma\in N$. It can be formulated mathematically as the following network flow problem,

$$\min\quad \vec{z}(\vec{x})=\begin{cases} z_1(\vec{x})=\sum_{(i,j)\in A} c_{ij}^1 x_{ij},\\ \cdots\\ z_q(\vec{x})=\sum_{(i,j)\in A} c_{ij}^q x_{ij},\end{cases} \tag{1}$$

$$s.t.\quad \sum_{(i,j)\in A} x_{ij}-\sum_{(j,i)\in A} x_{ji}=\begin{cases}1 & \text{if } i=s,\\ 0 & \text{if } i\neq s,\gamma,\\ -1 & \text{if } i=\gamma,\end{cases} \tag{2}$$

$$x_{ij}\in\{0,1\}\quad \text{for all } (i,j)\in A. \tag{3}$$

where $\vec{x}$ is a vector of flows on the arcs, and the constraints (2) represent flow balance at the different nodes.

Definition 2. In multiobjective problems, cost vectors induce a partial order preference relation $\prec$ called *dominance*,

$$\forall\vec{y},\vec{y}'\in\mathbb{R}^q\quad \vec{y}\prec\vec{y}'\Leftrightarrow\forall i\quad y_i\le y_i'\wedge\vec{y}\neq\vec{y}' \tag{4}$$

Analogously, we define the preference relation $\preceq$called *dominance or equality*,

$$\forall\vec{y},\vec{y}'\in\mathbb{R}^q\quad \vec{y}\preceq\vec{y}'\Leftrightarrow\vec{y}\prec\vec{y}'\wedge\vec{y}=\vec{y}' \tag{5}$$

where y_i denotes the ith element of vector $\vec{y}$. For any two vectors, it is not always possible to rank one as better than the other according to dominance, e.g. none of (20,30), (30,20) dominates each other, but both are dominated by (10,10).

Definition 3. Given a set of vectors X, we define $nd(X)$, the set of non-dominated vectors in X as

$$nd(X)=\{\vec{x}\in X\mid\nexists\vec{y}\in X,\quad \vec{y}\prec\vec{x}\} \tag{6}$$

Definition 4. We define the lexicographic order $\prec_L$ as follows:

$$\forall\vec{y},\vec{y}'\in\mathbb{R}^q\quad \vec{y}\prec_L\vec{y}'\Leftrightarrow\exists j\,y_j<y_j'\wedge\forall i<j\,y_i=y_i'. \tag{7}$$

and the preference relation $\preceq_L$,

$$\forall\vec{y},\vec{y}'\in\mathbb{R}^q\quad \vec{y}\preceq_L\vec{y}'\Leftrightarrow\vec{y}\prec_L\vec{y}'\wedge\vec{y}=\vec{y}' \tag{8}$$

Definition 5. We also define the linear aggregate order $\prec_{lin}$ as follows:

$$\forall\vec{y},\vec{y}'\in\mathbb{R}^q\quad \vec{y}\prec_{lin}\vec{y}'\Leftrightarrow\sum_i y_i<\sum_i y_i',\ 1\le i\le q \tag{9}$$

where y_i denotes the ith component of vector $\vec{y}$.

The lexicographic and linear aggregate orders are strict total orders. The lexicographic and linear aggregate optima of a set of vectors are trivially non-dominated vectors.

A basic pseudocode for algorithm NAMOA* is shown in Table 1. More details about this algorithm can be found elsewhere [26]. NAMOA* is a best-first algorithm that builds a search graph SG, rooted at the start node s, to store all non-dominated paths found to each node. Each node in the search graph has two sets of labels: $G_{op}(n)$ denotes the set of cost vectors of paths reaching n that can be further explored; $G_{cl}(n)$ denotes the set of cost vectors that have already been expanded, and we refer to them as closed or permanent. The algorithm uses a distance estimate function $H(n)$ that returns a set of the cost estimates of the cost of all non-dominated paths from node n to the goal. When the cost estimates lower bound real costs, NAMOA* is an exact algorithm. In this work we are concerned with the case where each node has a single lower bound, which we denote as $\vec{h}(n)$. Let $\vec{g}(P_{sn})$ be the cost of some path P_{sn} reaching some node n from the start node. Then, $\vec{f}(P_{sn})=\vec{g}(P_{sn})+\vec{h}(n)$ is an estimate of the cost of an extension of such path to the goal node.

For each unexplored label $(n,g), n\in N, \vec{g}\in G_{op}(n)$, an extended label $(n,\vec{g},\vec{f})$ is calculated such that $\vec{f}=\vec{g}+\vec{h}(n)$. All such extended labels are kept in an OPEN set. At each iteration, an extended label with a non-dominated $\vec{f}$ in OPEN is selected for expansion. By abuse of language, we will sometimes refer to extended labels simply as labels. If n is the goal node, then $\vec{f}$ is stored in COSTS, the set of non-dominated solutions costs. Otherwise, the non-dominated label $(n,\vec{g},\vec{f})$ selected from OPEN is made permanent, i.e., g is moved from $G_{op}(n)$ to $G_{cl}(n)$. For each arc (n,n') in the graph with cost $\vec{c}(n,n')$, a new label is generated for