Finding a Small, Diverse Subset of the Pareto Solution Set in Bi-Objective Search (Extended Abstract)

Pablo Araneda[1,2], Carlos Hernández Ulloa[2,3], Nicolás Rivera[4], Jorge A. Baier[1,2,5]
[1] Pontificia Universidad Católica de Chile, Chile
[2] Centro Nacional de Inteligencia Artificial CENIA, Santiago, Chile
[3] Universidad San Sebastián, Chile
[4] Universidad de Valparaíso, Chile
[5] Instituto Milenio Fundamentos de los Datos, Chile
pharaneda@uc.cl, carlos.hernandez@uss.cl, nicolas.rivera@uv.cl, jabaier@ing.puc.cl

Introduction

In bi-objective search, we are given a graph where each arc is associated with a pair of nonnegative costs. Consequently, each path also has two costs associated with it. To compare different paths, a dominance structure is employed. A path π_1 is dominated by path π_2 if their costs are different and both costs of π_1 are smaller than or equal to the corresponding costs of π_2. This dominance relation only establishes a partial order, and thus, given a starting node s_{start} and a goal node s_{goal}, it is generally not possible to find a minimum cost path from s_{start} to s_{goal}. Instead, we find the so-called Pareto solution set, which contains all s_{start}-to-s_{goal} paths that are not dominated by another s_{start}-to-s_{goal} path.

Because bi-objective search instances do not have a unique solution, much of the research in the area focuses on the computation of Pareto solution sets. However, from a practical perspective, this approach may be unsatisfactory since Pareto sets could be quite large—containing several hundreds of solutions—and navigating through them can become overwhelming for a user. This issue motivates us to study the problem of finding a *small*, *good-quality* subset of the Pareto solution set. By small we mean that the resulting set should be small enough to allow a human user to reason about the solution set with relative ease. By good-quality we mean that the paths in the returned set correspond to different areas of the Pareto solution set.

Bi-Objective A* (BOA*) (Hernández et al. 2020) is a state-of-the-art bi-objective search algorithm on top of which we build our approaches for computing small diverse subsets of Pareto solution sets. It receives a bi-objective search instance and returns a cost-unique Pareto solution set.

This document presents two main contributions. First, we provide a simple formalization of *good-quality* subsets of a Pareto solution set. For this, we use measure of *richness* which has been employed in the study of Population Dynamics. Second, we propose Chebyshev BOA*, a variant of BOA* to compute good-quality subset approximations.

Richness in Bi-Objective Search

Following Magurran and McGill (2010), assume we are given a sample of organisms each of which belongs to a certain unique species. *Richness* is the number of species for which at least one organism has been found in the sample.

To compute the richness of a solution set we need to specify what 'species' to map each path to. To do this we divide the Pareto solution set in k numbered buckets, and then we say that a solution is of species i if it is in the bucket numbered with i. To define such the buckets we "normalize" the Pareto solution set. Assume $sols_P$ is a solution set with the two extreme solutions: the left-most solution, π_l, is the solution in $sols_p$ with minimum fist component (and maximum second component). The right-most solution, π_r, is the solution in $sols_p$ with minimum second component (and maximum first component). Let (c_1^{min}, c_2^{max}) and (c_1^{max}, c_2^{min}) be, respectively, the costs of solutions π_l and π_r. If $\mathbf{c} = (c_1, c_2)$ is the cost of a certain solution, then its normalized form is defined as $\mathbf{c}^{norm} = (c_1 \cdot (c_2^{max} - c_2^{min}), c_2 \cdot (c_1^{max} - c_1^{min}))$.

Let sub be any subset of $sols_P$. To compute richness for subset sub, we assume we are given a positive integer k, which defines k solution buckets $B_1, \ldots, B_k$. To define which solutions in sub go to which buckets, we define the vector which points from the former cost to the latter as $\mathbf{d} = \mathbf{c}^{norm}(\pi_r) - \mathbf{c}^{norm}(\pi_l)$. We define k bucket *centroids*, $\mathbf{b}_1, \ldots, \mathbf{b}_k$, where $\mathbf{b}_i = (c_1^{min} \cdot \frac{i-1}{k-1} + c_1^{max} \cdot \frac{k-i}{k-1}, c_2^{max} \cdot \frac{i-1}{k-1} + c_2^{min} \cdot \frac{k-i}{k-1})$, for $i \in \{1, \ldots, k\}$. A solution π in sub is associated with B_i if its closest centroid, with respect to the Euclidean distance from its normalized cost, is $\mathbf{b}_i$; in other words, we find the i minimizing the distance between $\mathbf{c}^{norm}(b_i)$ and $\mathbf{c}^{norm}(\pi)$.

Chebyshev BOA*

To compute good-quality subsets of the Pareto solution set, we propose to use Chebyshev BOA*, a best-first search algorithm that optimizes a rectangle-shaped function. Chebyshev BOA* receives the aspect of the rectangle is defined by its diagonal, represented with the standard $y = mx + n$ line notation. The solution it finds corresponds to the path in the Pareto solution set that intersects the smallest rectangle of the given aspect. As such it can be configured to 'target' any region of the full solution set.

The Chebyshev transformation (e.g., Miettinen 1998) assumes we are given a bi-objective search instance with cost function $\mathbf{c} = (c_1, c_2)$, and two real numbers, m and n, and

state	algorithm	t (ms)	S
New York City (NY)	BOA*	495	7.40
	Chebyshev BOA*	205	7.18
San Francisco Bay (BAY)	BOA*	262	6.92
	Chebyshev BOA*	121	6.78
Colorado (COL)	BOA*	1582	7.14
	Chebyshev BOA*	771	7.00
Florida (FLA)	BOA*	10800	7.54
	Chebyshev BOA*	4006	7.42

Table 1: Average time (t; in milliseconds) and richness (S) for each algorithm on each set of problems of a State. For Chebyshev BOA* we set the k parameter (number of solutions to be found) to 8.

defines the following transformation for the cost function of a given path π, $c_{m,n}(\pi) = \max\{c_1(\pi), (c_2(\pi) - n)/m\}$.

Intuitively $c_{m,n}(\pi)$ projects the point $\mathbf{c}(\pi)$ horizontally onto the line $y = mx + n$, and returns the maximum value between the X coordinate of such a projection and the original X coordinate value, $c_1(\pi)$. As a consequence, all points that lie in the perimeter of a rectangle whose diagonal is the line $y = mx + n$ are mapped to the same cost. To break ties on **f**, we use $f_1(\pi) + f_2(\pi)$, which guarantees that the soulution found is not dominated by another path.

Optimality of the Chebyshev transformation Searching for an optimal Chebyshev path will lead us to find a solution in the Pareto solution set. This is because, lines with equal costs are rectangles whose diagonals are a specific line. Larger rectangles are associated with a larger Chebyshev cost, and a Pareto solution set is found when the smallest rectangle 'hits' the Pareto solution set. Unlike the linear transformation, this approach allows to hit solutions that are in a concave area of the Pareto solution set.

Finding $k-2$ solutions in the Pareto solution set We already have two solutions from $sols_P$ since they result as a by-product of computing the heuristics to guide the search. We define how we run $k-2$ searches that attempt to find $k-2$ solutions between π_l and π_r. To this end, we configure m, n so that m is perpendicular to the line that passes through the normalized costs π_r, π_l, while n is such as the line passes through the bucked *centroid*. In this way, we attempt to find $k-2$ solutions which are equally separated among themselves and from the extremes π_l and π_r. Note that each bucket can be searched in parallel.

Empirical Evaluation

The objective of our evaluation was to evaluate the richness obtained by Chevyshev BOA* comparing against BOA*, which computes the full Pareto solution set. We also wanted to see if there was an advantage Chevyshev BOA* versus BOA* in terms of runtime. Since independent searches of Chevyshev BOA* can be parallelized, we report the maximum runtime among all searches. We evaluated the algorithms on four maps (see Table 1) of the 9th DIMACS Implementation Challenge: Shortest Path[1]. For average results, we used 50 random instances of each map. We implement our algorithm on top of the publicly available implementation of BOA*. All experiments were run on a 2.00GHz Intel(R) Xeon(R) CPU Linux computer with 12GB of RAM.

[1]http://users.diag.uniroma1.it/challenge9/download.shtml

Algorithm 1: Chebyshev BOA*

Input: $m, n \in \mathbb{R}$, Search instance $(S, E, c, s_{start}, s_{goal})$
Returns: A path in $sols_p$

for each $s \in S$ **do**
  $g_1^{min}(s) \leftarrow \infty$
  $g_2^{min}(s) \leftarrow \infty$
$x \leftarrow$ new node with $s(x) = s_{start}$
$\mathbf{g}(x) \leftarrow (0, 0)$
$parent(x) \leftarrow$ null
$\mathbf{f}(x) \leftarrow \mathbf{h}(s_{start})$
Initialize priority queue *Open*, where the priority of element x is given by $(c_{m,n}(\mathbf{f}(x)), f_1(x) + f_2(x))$ and lexicographical order is used to compare two priorities
Insert x to $Open$
while $Open \neq \emptyset$ **do**
  Remove a node x from $Open$ with lowest priority
  if $g_2(x) \geq g_2^{min}(s(x)) \wedge g_1(x) \geq g_1^{min}(s(x))$ **then**
    continue
  if $g_1(x) < g_1^{min}(s(x))$ **then** $g_1^{min}(s(x)) \leftarrow g_1(x)$
  if $g_2(x) < g_2^{min}(s(x))$ **then** $g_2^{min}(s(x)) \leftarrow g_2(x)$
  if $s(x) = s_{goal}$ **then return** x
  for each $t \in Succ(s(x))$ **do**
    $y \leftarrow$ new node with $s(y) = t$
    $\mathbf{g}(y) \leftarrow \mathbf{g}(x) + \mathbf{c}(s(x), t)$
    $parent(y) \leftarrow x$
    $\mathbf{f}(y) \leftarrow \mathbf{g}(y) + \mathbf{h}(t)$
    if $g_2(y) \geq g_2^{min}(t) \wedge g_1(y) \geq g_1^{min}(t)$ **then**
      continue
    Add y to $Open$
return $sols$

Conclusion

Chebyshev BOA*, a best-first algorithm we designed to find solutions lying on a specific region of the Pareto solution set, obtains small subsets of the solution set in about half of the time required by BOA* with a comparable solution richness.

Acknowledgements

The research was supported by National Center for Artificial Intelligence CENIA FB210017, Basal ANID.

References

Hernández, C.; Yeoh, W.; Baier, J.; Zhang, H.; Suazo, L.; and Koenig, S. 2020. A simple and fast bi-objective search algorithm. In *Proceedings of the 30th International Conference on Automated Planning and Scheduling (ICAPS)*, 143–151.

Magurran, A. E.; and McGill, B. J. 2010. *Biological diversity: frontiers in measurement and assessment*. OUP Oxford.

Miettinen, K. 1998. *Nonlinear multiobjective optimization*, volume 12 of *International Series in Operations Research & Management Science*. Springer New York, NY.