Discrete Optimization ■ (■■■■) ■■■–■■■

Contents lists available at ScienceDirect

Discrete Optimization

www.elsevier.com/locate/disopt

A hybrid approach for biobjective optimization

Thomas Stidsen[a,*], Kim Allan Andersen[b]

[a] *DTU-Management, Technical University of Denmark, Denmark*
[b] *Department of Economics and Business Economics, Aarhus University, Denmark*

ARTICLE INFO

Article history:
Received 29 March 2017
Received in revised form 6 October 2017
Accepted 5 February 2018
Available online xxxx

Keywords:
Biobjective optimization
Mixed integer programming
Traveling salesman problem
Branch-and-cut algorithm

ABSTRACT

A large number of the real world planning problems which are today solved using Operations Research methods are actually multiobjective planning problems, but most of them are solved using singleobjective methods. The reason for converting, i.e. simplifying, multiobjective problems to singleobjective problems is that no standard multiobjective solvers exist and specialized algorithms need to be programmed from scratch.

In this article we will present a hybrid approach, which operates both in decision space and in objective space. The approach enables massive efficient parallelization and can be used to a wide variety of biobjective Mixed Integer Programming models. We test the approach on the biobjective extension of the classic traveling salesman problem, on the standard datasets, and determine the full set of nondominated points. This has only been done once before (Florios and Mavrotas, 2014), and in our approach we do it in a fraction of the time.

1. Introduction

Most real-world problems which are optimized using Operations Research (OR) methods are actually multiobjective. Only very seldom do the OR practitioners consider their optimization problem as multiobjective optimization problems, i.e. in most cases the objectives are simply summed, possibly with weights thus emphasizing the most important objectives. While this approach may work in practice, it is problematic since it ignores a multitude of other (Pareto) optimal solutions. While there has been research going on in multiobjective optimization for decades, an increased interest in exact solution of multiobjective optimization problems has flourished in the last decade (i.e. since 2006): [1–14]. This research has however not yet lead to general practical solvers of multiobjective Mixed Integer Programming (MOMIP) models.

Generally speaking there have been two different types of approaches: Criteria Space Search (CSS) algorithms and Decision Space Search (DSS) algorithms. We will briefly describe the most important CSS algorithms and DSS algorithms below.

* Corresponding author.
E-mail addresses: thst@dtu.dk (T. Stidsen), kia@econ.au.dk (K.A. Andersen).

https://doi.org/10.1016/j.disopt.2018.02.001
1572-5286/

1.1. Criteria space search algorithms

CSS algorithms are iterative algorithms which utilize a series of singleobjective optimizations, where extra constraints are added to the MOMIP models criteria space. The two "classic" CSS algorithms are the ϵ-constraint method [15] and the two-phase method [16]. The ϵ-constraint method starts by finding one of the two lexicographic points. This point is then ruled out by a constraint on one of the criteria space variables and another (lexicographic) optimization leads to the next point of the Pareto front. The two-phase method starts with the two lexicographic points. In the first phase, the nondominated extreme (and possibly some non-extreme) points of the Pareto front are found by solving a series of singleobjective MIP's. In the second phase, a series of "triangles" in criteria space are iteratively searched for solutions, again using singleobjective MIP's or problem-dependent enumeration methods.

While the CSS algorithms have been known for decades modern versions have been constructed which seems very promising, e.g. [3,6].

1.2. Decision space search algorithms

DSS algorithms are (basically) branch-and-bound algorithms, tweaked to work with more than one objective. This requires a re-definition of the fathoming rules, new types of branching and new versions of pre-processing, of probing etc. The cutting plane generation is however **not** changed! Building an efficient branch-and-cut algorithm from scratch is a monumental task, which only very few researchers could possible do and the developed algorithm would most likely still be seriously inferior to standard solvers like CPLEX or Gurobi, if applied to singleobjective MIP models. This approach has been attempted a few times in [17,18]. Recently, there have however been approaches which utilize the so-called call-back procedures in modern solvers such as CPLEX or Gurobi [10,13]. These approaches overcome **some** of the issues.

1.3. Which approach is the best? CSS or DSS?

The question of which approach is the best, i.e. can solve the largest MOMIP models fastest is in our opinion open. More research is required and both approaches have their advantages and disadvantages: The CSS algorithms can utilize the fast development of new solver versions, which promise continuous improvement. Their main disadvantage is that they solve a large set of almost alike MIP models, and they will most likely process the same intermediate solutions, i.e. nodes in their branch-and-bound tree, many times. The DSS algorithms on the other hand suffer from reduced strength in the fathoming process and the extra requirement of integer branching, see Section 3.2. On the other hand, they will ideally only see a node once.

In our opinion the question is not CSS or DSS, but **CSS and DSS!** We believe that the paradigms should be mixed to improve performance, thus creating hybrid methods. Several papers have already been proposed that combine both approaches, of which we just name a few. For instance, [19] embeds dynamic programming into a two phases method, using bound sets to discard triangles to be searched. The method has been applied to the biobjective knapsack problem with promising results. [20] proposes a biobjective local branching method for mixed binary biobjective programming problems with promising results, although the CPU times are rather high, and [21] presents an algorithm that can determine all efficient points for the biobjective integer minimum cost flow problem, using a two-phases method combined with a ranking algorithm in the second phase. In our previous paper [10] we have also demonstrated the advantage of combining CSS and DSS, where a special branching method, called Pareto branching, is introduced. This allows branching in the criteria space, thus creating a hybrid approach. In our opinion this is a viable approach: To "enrich" DSS algorithms, with features from CSS algorithms, to speed up performance.

Please cite this article in press as: T. Stidsen, K.A. Andersen, A hybrid approach for biobjective optimization, Discrete Optimization (2018), https://doi.org/10.1016/j.disopt.2018.02.001.