A strongly polynomial-time approximation for a class of bicriteria problems

Asaf Levin[1]

February 3, 2004

Abstract

Consider the following problem: given a ground set and two minimization objectives of the same type find a subset from a given subset-class that minimizes the first objective subject to a budget constraint on the second objective. Using Megiddo's parametric method we improve an earlier weakly-polynomial time algorithm.

Keywords: parametric method, bicriteria optimization, approximation algorithms.

1 Introduction

Suppose that a ground set $\mathcal{E}$ of m elements, a bound D, a pair of non-negative integer cost functions c and d over $\mathcal{E}$, and a set SOL of *feasible subsets* (e.g., the edge sets of spanning trees of a given input graph) are given. An *element cost function* g is a non-negative function defined over $\mathcal{E}$. For $E \subseteq \mathcal{E}$, let the *g-cost of $\mathcal{E}$* denoted as $f(g, E)$ be the cost of E under the element cost function g. In this paper we consider the following type of BICRITERIA COMBINATORIAL OPTIMIZATION PROBLEM: Find a subset $E \in \mathcal{E}$ such that $f(d, E) \leq D$ and $f(c, E)$ is minimized.

Given constants a, b and element cost functions g_1 and g_2, $ag_1 + bg_2$ denotes the composite function that assigns a cost $ag_1(e) + bg_2(e)$ to each element in $\mathcal{E}$. We assume that $g_1 = c$ and $g_2 = d$ satisfy the following:

Assumption 1 $f(ag_1 + bg_2, E) = af(g_1, E) + bf(g_2, E)\ \forall E \in SOL.$

Assumption 2 $f(g_1, E) \geq 0\ \forall E \in SOL.$

[1]Faculty of Industrial Engineering and Management, Technion, Haifa 32000, Israel. levinas@tx.technion.ac.il

Assumptions 1 and 2 provide the monotonicity property: if g_1, g_2 are element cost functions such that $g_1(e) \geq g_2(e)\ \forall e \in \mathcal{E}$, then $f(g_1, E) \geq f(g_2, E)\ \forall E \in SOL$. The interesting case in which Assumptions 1 and 2 hold is when f is a weighted sum of the costs of elements in E (where the weights may depend on E). In Section 4 we discuss the use of these assumptions in the results of this paper and similar assumptions in the paper of Marathe et al. [3].

An (α, β)-approximation algorithm for a bicriteria combinatorial optimization problem is a polynomial-time algorithm that produces a solution E in which $f(d, E) \leq \beta D$ and $f(c, E)$ is at most α times the cost of any solution that satisfies the budget constraint D.

For a constant a, we let a^+ and a^- denote the symbols $a + \delta$ and $a - \delta$ for infinitesimally small $\delta > 0$.

Marathe, Ravi, Sundaram, Ravi, Rosenkrantz and Hunt [3] presented the *parametric search method*, a general method for bicriteria combinatorial optimization problems. Given an element cost function g, let A denote a ρ-approximation algorithm for the *single objective problem* defined as minimizing $f(g, E)$ subject to $E \in SOL$. We let $\mathbf{A}(g)$ denote the g-cost of the solution returned by A when applied with the cost function g. Let $\gamma > 0$ be an accuracy parameter. Let U be an upper bound on the optimal c-cost. The parametric search method (the description of [3] differs from the description below but both describe the same method) finds an approximate solution of the value λ^* which satisfies:

$$\mathbf{A}\left(c + \frac{\lambda^{*+}}{D} d\right) \leq \rho\lambda^*(1 + \gamma) \leq \mathbf{A}\left(c + \frac{\lambda^{*-}}{D} d\right). \tag{1}$$

The approximate solution is obtained using a binary search over λ that finds an interval of length ϵ that contains λ^*. In Algorithm Parametric-Search of [3], $\epsilon = 1$. Marathe et al. proved that given $\epsilon, \gamma > 0$ there is a $([1+\frac{1}{\gamma}+q_1(\epsilon)]\rho, [1+\gamma+q_2(\epsilon)]\rho)$-approximation algorithm where q_1, q_2 are non-negative functions that approach zero as their argument approaches zero. The algorithm is weakly-polynomial with worst-case time complexity of $(\log U + \log \frac{1}{\epsilon})$ times the complexity of A.

Let OPT be an optimal solution to the input bicriteria combinatorial optimization problem.

We assume that A is a ρ-approximation algorithm for the single objective problem that

uses only additions, multiplications by constants, and comparisons. Denote by $T_a(m)$ ($T_c(m)$) the maximum number of additions and multiplications (comparisons) done by A when it is applied to a ground set with m elements. We first assume that $\rho = 1$, and present a $(1 + \frac{1}{\gamma}, 1 + \gamma)$-approximation algorithm with time complexity $O(T_a(m) + T_c(m)[T_a(m) + T_c(m)])$. This approximation algorithm is based on using Megiddo's parametric method [4] to solve *exactly* the following equation in the "parametric search method":

$$\mathbf{A}\left(c + \frac{\lambda^*}{D}d\right) = \rho\lambda^*(1 + \gamma). \tag{2}$$

Next, we generalize the method to the case where each computation gives an approximate solution, and use this generalization to obtain a $([1 + \frac{1}{\gamma}]\rho, [1 + \gamma]\rho)$-approximation algorithm with time complexity $O(T_a(m) + T_c(m)[T_a(m) + T_c(m)])$. These results improve the one of [3] both in the approximation ratio and in the time complexity (if the size of the costs in the input is sufficiently large). In particular, if A is a strongly-polynomial algorithm, then our method is strongly-polynomial whereas the method in [3] is only weakly-polynomial.

We note that in many cases the time-complexity of our method can be reduced using the methods of Megiddo [5] and Cole [1]. But such uses are problem-specific.

2 A $(1 + \frac{1}{\gamma}, 1 + \gamma)$-approximation algorithm when $\rho = 1$

In this section we assume that A solves (optimally) the single objective problem (i.e., $\rho = 1$). Assume that $\gamma > 0$. We consider the following function of λ:

$$\mathbf{F}(\lambda) = \mathbf{A}\left(c + \frac{\lambda}{D}d\right) - \lambda(1 + \gamma).$$

$\mathbf{F}$ is a piecewise-linear continuous (strictly monotone decreasing) function because for each $E \in SOL$,

$$\mathbf{F_E}(\lambda) = f\left(c + \frac{\lambda}{D}d, E\right) - \lambda(1 + \gamma) = f(c, E) + \lambda\left[\frac{f(d, E)}{D} - (1 + \gamma)\right]$$

is a linear function of λ, and $\mathbf{F}$ is the minimum of $\mathbf{F_E}$ over $E \in SOL$. We use Megiddo's method [4] (see also [2] and [5]) to obtain a value λ^* such that $\mathbf{F}(\lambda^*) = 0$. I.e.,

$$\mathbf{A}\left(c + \frac{\lambda^*}{D}d\right) = \lambda^*(1 + \gamma). \tag{3}$$

We now describe the use of Megiddo's method. We execute A with the cost function $c+\frac{\lambda^*}{D}d$ without knowing λ^* in advance. In each phase we maintain an interval $I = [\lambda_{\text{low}}, \lambda_{\text{hi}}]$ such that

$$\mathbf{A}\left(c + \frac{\lambda_{\text{low}}}{D}d\right) \geq \lambda_{\text{low}}(1+\gamma) \tag{4}$$

and

$$\mathbf{A}\left(c + \frac{\lambda_{\text{hi}}}{D}d\right) \leq \lambda_{\text{hi}}(1+\gamma). \tag{5}$$

Since $\mathbf{F}$ is continuous, $\mathbf{A}(c+\frac{\lambda}{D}d)$ is continuous, and therefore, in each phase there is a $\lambda^* \in I$ such that $\mathbf{A}(c+\frac{\lambda^*}{D}d) = \lambda^*(1+\gamma)$. Initially, we set $\lambda_{\text{low}} = 0$ that satisfies (4) and $\lambda_{\text{hi}} = \infty$ that satisfies (5) because for a solution whose d-cost is at most D, the coefficient of λ_{hi} on the left-hand side is less than 1, whereas its coefficient on the right-hand side is $1+\gamma$, therefore, property (5) will be satisfied for a sufficiently large value of the parameter λ. While executing A, we need to compare pairs of linear functions of λ according to their relative order for $\lambda = \lambda^*$. To do so, we first compute the break-point λ_{br} of the two linear functions. If there is no such break-point, then the comparison is independent of λ (the two functions differ by a constant), and we can resolve it in constant time. Next, we check if $\lambda_{\text{br}} \in I$. If not, then the comparison is independent of the value of $\lambda \in I$, and we can resolve it in constant time. Assume that $\lambda_{\text{br}} \in I$. We execute A with cost function $c+\frac{\lambda_{\text{br}}}{D}d$. If $\mathbf{A}(c+\frac{\lambda_{\text{br}}}{D}d) \geq \lambda_{\text{br}}(1+\gamma)$, then we set $\lambda_{\text{low}} = \lambda_{\text{br}}$, otherwise, we set $\lambda_{\text{hi}} = \lambda_{\text{br}}$. Now the comparison is independent of λ for $\lambda \in I$, and therefore, we can resolve it in constant time.

We return the solution HEU that is found by A when it is applied to the cost function $c+\frac{\lambda^*}{D}d$. The time complexity of our algorithm is $O(T_a(m) + T_c(m)[T_a(m) + T_c(m)])$.

The proof of the next theorem is similar to the proof of Claim 6.2 in [3], and it follows by the proof of Theorem 4 below.

Theorem 3 *The algorithm described above is a* $(1+\frac{1}{\gamma}, 1+\gamma)$*-approximation algorithm.*

3 $\rho > 1$

Given $\lambda \geq 0$, denote by OPT_λ the cost of an optimal solution to the single objective problem with the element cost function $c+\frac{\lambda}{D}d$. In this section we assume that A is a deterministic

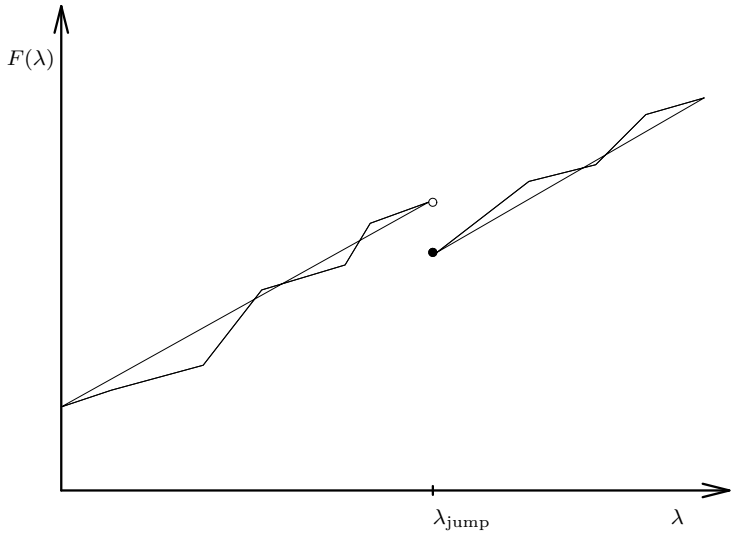

Figure 1: An illustration of $F(\lambda)$

ρ-approximation algorithm (i.e., for each λ, $\mathbf{A}(c + \frac{\lambda}{D}d) \leq \rho OPT_\lambda$).

Denote by λ^* a value of λ that satisfies $\mathbf{A}(c + \frac{\lambda}{D}d) = \rho\lambda(1+\gamma)$. However, A is an approximation algorithm and not an exact algorithm, therefore there may be jumps in the cost of the solution returned by A as a function of λ. Therefore, we cannot use the method described in the previous section to find λ^*. Instead we find a value $\hat{\lambda}$ of λ such that either

$$\mathbf{A}\left(c + \frac{\hat{\lambda}}{D}d\right) = \rho\hat{\lambda}(1+\gamma) \tag{6}$$

or

$$\mathbf{A}\left(c + \frac{\hat{\lambda}^+}{D}d\right) \leq \rho\hat{\lambda}(1+\gamma) \leq \mathbf{A}\left(c + \frac{\hat{\lambda}^-}{D}d\right). \tag{7}$$

We now show how to compute $\hat{\lambda}$. Since A is deterministic and it is composed of a finite number of additions and comparisons, the function $F(\lambda) = \mathbf{A}(c + \frac{\lambda}{D}d)$ is a piecewise-linear function of λ (see Figure 1 for an illustration of F where F has a single jump-point at λ_{jump}). The break-points and the jump-points of $F(\lambda)$ are points where the solution returned by A changes. Since the cost of each solution $E \in SOL$ according to the element cost function $c + \frac{\lambda}{D}d$ is a non-decreasing function of λ, OPT_λ is a non-decreasing function (of λ). Therefore, since A is a ρ-approximation, for a jump-point λ_j of F,

$$F(\lambda_j^-) \leq \rho OPT_{\lambda_j^-} \leq \rho OPT_{\lambda_j} \leq \rho OPT_{\lambda_j^+}. \tag{8}$$

We extend the parametric method of [4] in the following way: we execute A with cost function $c + \frac{\hat{\lambda}}{D}d$ without knowing $\hat{\lambda}$ in advance. In each phase we maintain an interval $I = [\lambda_{\text{low}}, \lambda_{\text{hi}}]$ such that

$$\mathbf{A}\left(c + \frac{\lambda_{\text{low}}}{D}d\right) \geq \lambda_{\text{low}}\rho(1+\gamma) \tag{9}$$

and

$$\mathbf{A}\left(c + \frac{\lambda_{\text{hi}}}{D}d\right) \leq \lambda_{\text{hi}}\rho(1+\gamma). \tag{10}$$

In each phase there exists a value $\hat{\lambda} \in I$ such that either $\mathbf{A}(c+\frac{\hat{\lambda}}{D}d) = \hat{\lambda}\rho(1+\gamma)$ or $\mathbf{A}(c+\frac{\hat{\lambda}^+}{D}d) \leq \hat{\lambda}\rho(1+\gamma) \leq \mathbf{A}(c+\frac{\hat{\lambda}^-}{D}d)$. Similar to the $\rho = 1$ case, initially we set $\lambda_{\text{low}} = 0$ that satisfies (9) and $\lambda_{\text{hi}} = \infty$ that satisfies (10). While executing A, we need to compare pairs of linear functions of λ according to their relative order for $\lambda = \hat{\lambda}$. To do so, we first compute the break-point λ_{br} of the two linear functions. If there is no such break-point or $\lambda_{\text{br}} \notin I$, then the comparison is independent of the value of $\lambda \in I$ and we can resolve it in constant time. Assume that $\lambda_{\text{br}} \in I$. We execute A in λ_{br}^-, λ_{br} and λ_{br}^+. If λ_{br} is a jump-point (i.e., $F(\lambda_{\text{br}}^-)$, $F(\lambda_{\text{br}})$ and $F(\lambda_{\text{br}}^+)$ are not all equal), then we check if $F(\lambda_{\text{br}}^+) \leq \lambda_{\text{br}}\rho(1+\gamma) \leq F(\lambda_{\text{br}}^-)$. If this holds, then $\hat{\lambda} = \lambda_{\text{br}}$. In any other case (also if λ_{br} is not a jump-point) the following holds: Either $F(\lambda_{\text{br}}^-), F(\lambda_{\text{br}}^+) \geq \lambda_{\text{br}}\rho(1+\gamma)$ or $F(\lambda_{\text{br}}^-), F(\lambda_{\text{br}}^+) \leq \lambda_{\text{br}}\rho(1+\gamma)$. In the first case we set $\lambda_{\text{low}} = \lambda_{\text{br}}^+$, and in the second case we set $\lambda_{\text{hi}} = \lambda_{\text{br}}^-$. Now the comparison is independent of λ for $\lambda \in I$, and therefore, we can resolve it in constant time.

The time complexity of our algorithm is $O(T_a(m) + T_c(m)[T_a(m) + T_c(m)])$ using the same analysis as in [4].

If (6) is satisfied, we return the solution that is found by A when it is applied to the element cost function $c + \frac{\hat{\lambda}}{D}d$. If (7) is satisfied, we return the solution that is found by A when it is applied to the element cost function $c + \frac{\hat{\lambda}^+}{D}d$. In both cases let HEU be the returned solution.

Theorem 4 *The algorithm described above is a* $([1+\frac{1}{\gamma}]\rho, [1+\gamma]\rho)$*-approximation algorithm.*

Proof:

$$\hat{\lambda}(1+\gamma) \quad \leq \quad \frac{\mathbf{A}(c + \frac{\hat{\lambda}^-}{D}d)}{\rho} \tag{11}$$

$$
\begin{aligned}
&\leq OPT_{\hat{\lambda}} \\
&\leq f(c, OPT) + \frac{\hat{\lambda}}{D} f(d, OPT) \\
&\leq f(c, OPT) + \hat{\lambda},
\end{aligned}
$$

where the first inequality holds by (6) and (7), the second inequality holds because A is a ρ-approximation and by (8), the third inequality holds because OPT is a feasible solution to the single objective problem, and the last inequality holds because $f(d, OPT) \leq D$. Therefore,

$$
\hat{\lambda} \leq \frac{f(c, OPT)}{\gamma}. \tag{12}
$$

We conclude that

$$
\begin{aligned}
f(c, HEU) &\leq f\left(c + \frac{\hat{\lambda}}{D} d, HEU\right) \\
&\leq \rho OPT_{\hat{\lambda}} \\
&\leq \rho[f(c, OPT) + \hat{\lambda}] \\
&\leq f(c, OPT)\rho\left(1 + \frac{1}{\gamma}\right),
\end{aligned}
$$

where the first inequality holds because $d \geq 0$, the second inequality holds because A is a ρ-approximation algorithm and OPT_{λ} is continuous (and therefore, $OPT_{\hat{\lambda}} = OPT_{\hat{\lambda}^+}$), the third inequality holds by (11), and the last inequality holds by (12).

Finally, if (6) holds, then let $\tilde{\lambda} = \hat{\lambda}$, and if (7) holds, then let $\tilde{\lambda} = \hat{\lambda}^+$. Then,

$$
\begin{aligned}
f(d, HEU) &\leq \frac{D}{\hat{\lambda}} \mathbf{A}\left(c + \frac{\tilde{\lambda}}{D} d\right) \\
&\leq \frac{D}{\hat{\lambda}} \rho \hat{\lambda}(1 + \gamma) = D\rho(1 + \gamma),
\end{aligned}
$$

where the first inequality holds because $c \geq 0$, and the second inequality holds by (6) and (7). ■

4 Discussion

We conclude the paper with a discussion on the use of Assumptions 1 and 2 in the results of this paper, and the use of some assumptions on f in [3]. Assumption 2 is a natural assumption. The more restricting assumption is Assumption 1.

The following is an example for a function f that does not satisfy Assumption 1 where the parametric approaches of this paper and [3] fail: Let $\mathcal{E}= \{e_1, e_2, e_3\}$ and $SOL = \{S_1, S_2\}$, where $S_1 = \{e_1, e_2\}$ and $S_2 = \{e_2, e_3\}$. c and d are defined as $c(e_1) = 1$, $c(e_2) = c(e_3) = 0$, $d(e_1) = 0$, $d(e_2) = d(e_3) = 1$. The budget D is 1. let M be a large number, and let f is defined as:

$$f(g, S_1) = \begin{cases} 0 & \text{if } g(e_1) \neq 0 \\ M & \text{otherwise} \end{cases}$$

and $f(g, S_2) = 1$. Then, S_2 is a feasible solution with a unit c-cost, whereas S_1 does not satisfy the budget constraint. However, for all values of λ the optimal solution with element cost function $c + \lambda d$ is S_1 with zero cost, but $f(d, S_1) = M >> 1$. Therefore, our method and the method of [3] do not find an approximated solution for this problem.

References

[1] R. Cole, "Slowing down sorting networks to obtain faster sorting algorithms," *Journal of the ACM*, **34**, 200-208, 1987.

[2] D. Gusfield, "Parametric combinatorial computing and a problem of program module distribution," *Journal of the ACM*, **30**, 551-563, 1983.

[3] M. V. Marathe, R. Ravi, R. Sundaram, S. S. Ravi, D. J. Rosenkrantz and H. B. Hunt III, "Bicriteria network design problems," *Journal of Algorithms*, **28**, 142-171, 1998.

[4] N. Megiddo, "Combinatorial optimization with rational objective functions," *Mathematics of Operations Research*, **4**, 414-424, 1979.

[5] N. Megiddo, "Applying parallel computation algorithms in the design of serial algorithms," *Journal of the ACM*, **30**, 852-865, 1983.