A Note on the Polynomial Solvability of the Resource Allocation Problem over Polymatroids

Arie Tamir

Raymond and Beverly Sackler
Faculty of Exact Sciences
School of Mathematical Sciences
Tel Aviv University
Tel Aviv 69978, Israel

March 1993

Abstract

It is shown that the resource allocation problem over a polymatroid with a quadratic objective can be solved in strongly polynomial time.

Given the set $E = \{1, \ldots, n\}$ let $r : 2^E \to R$ be a real-valued set function defined on the power set of E with $r(\phi) = 0$. We say that r is *submodular* if the submodular inequality

$$r(X) + r(Y) \geq r(X \cap Y) + r(X \cup Y) \tag{1}$$

holds for every pair of subsets X and Y of E. The set function r is *monotone* if $r(X) \leq r(Y)$ for every pair of subsets X and Y satisfying $X \subseteq Y$. Let x be a vector in R^E. For every subset S of E the notation $x(S) = \sum_{i \in S} x_i$ is adopted. We use e_i to denote the i-th unit vector in R^E.

Given a submodular function r, the polymatroid defined by r is the polyhedron

$$F = \{x \in R_+^E |\ x(S) \leq r(S) \quad \text{for all} \quad S \subseteq E\} . \tag{2}$$

For $j = 1, \ldots, n$, let $f_j(z)$ be a real-valued, continuous and concave function defined on the real line. Consider the following resource allocation problem:

$$\begin{aligned} &\text{Max} \sum_{j=1}^{n} f_j(x_j) \\ &\text{s.t. } x = (x_1, \ldots, x_n) \in F . \end{aligned} \tag{3}$$

We assume that the set function r defining F is submodular. If r is integer-valued define the *discrete equivalent* of the above resource allocation problem by requiring feasible vectors to have integer components. The above problem is discussed in [2,5,6,8,9,10]. Polynomial time algorithms to find an ϵ-accurate solution to the continuous problem and an optimal solution to its discrete version are given in [8]. The polynomial time bounds are expressed in terms of the input length and the time spent by an oracle which checks the feasibility of an increment in a component of a given feasible solution. Specifically, let G denote the number of operations needed for such a check. Then the bound in [8] for solving the discrete version is $O(n(\log n + G)\log(r(E)/n))$. (If $r(E) \leq n$, $\log(r(E)/n)$ should be replaced by 1.) The algorithms in [8] are based on efficient implementations of the greedy algorithm [2,10]. We note that checking the feasibility of a given increment can be performed by minimizing a submodular function. Suppose that x is a feasible solution, i.e., x is in F, and consider incrementing the i-th component of x. Then such an increment is feasible if and only if the minimum value of the following optimization problem is positive:

$$\begin{aligned} &\operatorname{Min}\{r(S) - x(S)\} \\ &\text{s.t. } S \text{ is a subset of } E \text{ containing } i \ . \end{aligned} \tag{4}$$

Since the objective in (4) is a submodular function defined over the lattice of all subsets of E containing i, the above minimization can be performed in strongly polynomial time using the ellipsoidal approach, provided that r is given by an appropriate oracle and is rational-valued, [7].

As is apparent from the above bounds, the number of calls to the oracle in the greedy algorithm in [8], depends on the value $r(E)$. Thus, this number of calls is not strongly polynomial. This might be a disadvantage if we want to use the greedy algorithm to prove that certain instances of (3) are solvable in strongly polynomial time.

We will show in this note that the decomposition algorithm in [4,5,6] can be implemented in polynomial time where the number of calls to an oracle is independent of the function r, and is only $O(n^2)$. We will then use this result to conclude that the special case of (3) where each function f_j is quadratic, is solvable in strongly polynomial time.

The following is the description of the decomposition algorithm in [6] which solves the resource allocation problem and its discrete variation. To simplify the presentation it is assumed, as in [6], that the function r is both submodular and monotone, and that each function f_j is concave and strictly increasing. (See the remarks at the end for a discussion of these assumptions.)

Decomposition Algorithm.

Step 1. Find a solution y of the single constraint problem

$$\begin{gathered} \operatorname{Max} \sum_{j=1}^{n} f_j(x_j) \\ \text{s.t. } x(E) \le r(E) \\ x \in R_+^E \ . \end{gathered}$$

Step 2. Find a maximal vector v in F satisfying $v \le y$.

Step 3. Find E_1, the subset of E, defined by

$$E_1 = \{i \in E | \text{ there exists no positive real } a \text{ such that } v + ae_i \text{ is in } F\} .$$

Let $E_2 = E \backslash E_1$.

Step 4. If $E_1 = E$ then stop: y is an optimal solution, otherwise continue with Step 5.

Step 5. Find a solution x^1 to the following problem

$$\begin{aligned} &\text{Max} \sum_{j \in E_1} f_j(x_j) \\ &\text{s.t. } x(S) \leq r(S) \text{ for all } S \subseteq E_1 , \\ &x \in R_+^{E_1} . \end{aligned}$$

Step 6. Find a solution x^2 to the following problem

$$\begin{aligned} &\text{Max} \sum_{j \in E_2} f_j(x_j) \\ &\text{s.t. } x(S) \leq r(S \cup E_1) - r(E_1) \text{ for all } S \subseteq E_2 , \\ &x \in R_+^{E_2} . \end{aligned}$$

Step 7. Define the vector $x \in F$ by setting

$$x_i = \begin{cases} x_i^1 & \text{if } i \in E_1 \\ x_i^2 & \text{if } i \in E_2 , \end{cases}$$

for $i = 1, \ldots, n$.

x is an optimal solution.

The validity of the decomposition algorithm for solving (3) under the above assumptions is demonstrated in [6]. We now analyze its complexity. It is obvious that the algorithm iterates at most $2n + 1$ times. Let T_1 denote the number of operations needed to solve a single constraint problem as in Step 1, and let T_2 denote the number of operations required to perform one pass through Steps 2 and 3. Thus, the complexity of the decomposition algorithm is $O(n(T_1 + T_2))$. To estimate T_2 we let G denote the number of operations needed to minimize a submodular function over a lattice family. We show that $T_2 = O(nG)$. Starting with Step 2 we need to find a maximal vector v in F such that $v \leq y$ for a given vector y.

Define v^0 to be the zero vector in R^E. For $i = 1, \ldots, n$, let

$$a_i = \text{Minimum}\{r(S) - v^{i-1}(S)\}$$
$$\text{s.t. } S \subseteq E \text{ and } i \in S \;.$$

Set

$$b_i = \text{Minimum}[a_i, y_i]$$

and

$$v^i = v^{i-1} + b_i e_i \;.$$

It is easy to see that the vector v^n is a maximal vector in F satisfying $v^n \leq y$. As noted above the effort to find a_i is G. Therefore one pass through Step 2 of the decomposition algorithm takes $O(nG)$ time. Turning to Step 3, we note that an index i is in E_1 if and only if the following minimum is positive,

$$\text{Minimum}\{r(S) - v(S)\}$$
$$\text{s.t. } S \subseteq E \text{ and } i \in S \;.$$

Therefore, a single pass through Step 3 also consumes $O(nG)$ time. We now conclude that the decomposition algorithm takes $O(nT_1 + n^2G)$ time to solve (3) or its discrete variant. It is known that for the discrete case $T_1 = O\big(n \log(r(E)/n)\big)$ [3].

To demonstrate the usefulness of the decomposition algorithm consider the quadratic case in which each function f_j is quadratic. The existence of a strongly polynomial algorithm to solve this case is mentioned in [9] as an open problem. It is known that for the quadratic case $T_1 = O(n)$ for both the discrete and the continuous cases [1,10]. Since the bound G is strongly polynomial in general [7], it follows that the quadratic case can be solved in strongly polynomial time, $O(n^2 + n^2G) = O(n^2G)$.

Remarks. We have assumed above that the set function r is monotone and that the concave functions f_j, $j = 1, \ldots, n$, are strictly increasing. It is already noted in [6] that the latter assumption can be removed without affecting the complexity bound. If r is not monotone we can easily monotonize it by again applying the minimization in [7]. For example, in the single constraint problem of Step 1 we have to replace $r(E_1)$ by $r_{\mathrm{mon}}(E_1)$ where the latter is defined by

$$r_{\mathrm{mon}}(E_1) = \mathrm{Minimum}\{r(S) | \ E_1 \subseteq S \subseteq E\} \ .$$

Computing the above minimum amounts to minimizing a submodular function over the lattice of subsets containing E_1. We can easily verify that this additional effort for monotonization also does not affect the complexity bound.

References

[1] P. Brucker, "An $O(n)$ algorithm for quadratic knapsack problems", *Operations Research Letters* **3** (1984) 163-166.

[2] A. Federgruen and H. Groenevelt, "The greedy procedure for resource allocation problems: Necessary and sufficient conditions for optimality", *Operations Research* **34** (1986) 909-918.

[3] G.N. Frederickson and D.B. Johnson, "The complexity of selection and ranking in $X + Y$ and matrices with sorted columns", *Journal on Computer and System Science* (1982) 197-208.

[4] S. Fujishige, "Lexicographically optimal base of a polymatroid with respect to a weight vector", *Mathematics of Operations Research* **5** (1980) 186-196.

[5] S. Fujishige, Submodular Functions and Optimization, *Ann. Discrete Mathematics* **47**, North-Holland, New York 1991.

[6] H. Groenevelt, "Two algorithms for maximizing a separable concave function over a polymatroid feasible region", *European J. Operational Research* **54** (1991) 227-236.

[7] M. Grotschel, L. Lovasz and A. Schrijver, Geometric Algorithms and Combinatorial Optimization, Springer-Verlag, Berlin, 1988.

[8] D.S. Hochbaum, "Optimal algorithms for the allocation problem and its extensions", to appear in *Mathematics of Operations Research.*

[9] D.S. Hochbaum and S.P. Hong, "About strongly polynomial time algorithms for quadratic optimization over submodular constraints", Technical report, Department of IEOR, University of California, Berkeley, August 1992.

[10] T. Ibaraki and N. Katoh, Resource Allocation Problems: Algorithmic Approaches, MIT Press, Cambridge 1988.