A Pseudopolynomial Problem Formulation for Exact Knapsack Separation

E.A. Boyd

CRPC-TR91145
1991

Center for Research on Parallel Computation
Rice University
P.O. Box 1892
Houston, TX 77251-1892

A Pseudopolynomial Problem Formulation

for Exact Knapsack Separation

E. Andrew Boyd[1]

TR91-04

[1]This work was sponsored in part by NSF grant ECS-8809053.

Abstract

The NP-complete separation problem for the knapsack polyhedron $\mathcal{P}$ is formulated as a linear program with a pseudopolynomial number of variables and constraints. Due to the intrinsic difficulty of the problem this is arguably the most compact formulation achievable. It is demonstrated that the primal polyhedron associated with this formulation can be projected onto an appropriate subspace to yield $\mathcal{P}$ and that the dual polyhedron can be projected onto an appropriate subspace to yield the polar of $\mathcal{P}$. Practical consequences of the formulation are discussed.

1 Introduction

It is difficult to overemphasize the impact of cutting planes on the fields of integer programming and combinatorial optimization. Beyond leading to a wealth of elegant theoretical results, the use of cutting planes in the solution of integer programs has lead to breakthroughs that would have been unimaginable only a decade ago.

Fundamental to the application of successful cutting plane techniques is the ability to solve the *separation* problem: given a point $\hat{x} \in \mathbb{R}^n$ and a polyhedron $\mathcal{P} \in \mathbb{R}^n$, demonstrate that $\hat{x} \in \mathcal{P}$ or provide a vector d and a scalar d_0 such that $dx \leq d_0$ for all $x \in \mathcal{P}$ but $d\hat{x} > d_0$. If $\hat{x}$ is the solution to a linear programming relaxation of an integer program and $\mathcal{P}$ is a polyhedron that contains all feasible solutions to that integer program then the constraint $dx \leq d_0$ is a cutting plane. While ultimately it would be desirable to solve the separation problem with $\mathcal{P}$ equal to the convex hull of feasible integer solutions to the integer program, in most instances it is necessary to seek good separation algorithms for relaxations of this convex hull due to the intimate relation between the complexity of optimizing a linear function on a polyhedron and the separation problem [13]. Cutting planes generated by solving the separation problem for such relaxations have proven to be remarkably powerful computational tools.

One of the greatest demonstrations of the power of cutting planes generated by solving the separation problem for relaxed polyhedra was the Lanchester prize-winning paper of Crowder, Johnson, and Padberg [11]. In this paper, facets of the knapsack polyhedra defined by individual constraints of the integer programs they sought to solve were used as cutting planes. Their results were truly astonishing, as they were able to solve all but one of a collection of integer programs to optimality in under 15 minutes even though "most [of the problems] were originally considered not amenable to exact solution in economically feasible computation times" ([11], p. 828). The largest most difficult problem required only slightly less than an hour to solve. Cutting planes based on knapsack polyhedra have begun to find their way into publically available general purpose algorithms such as

IBM's OSL and Georgia Tech's GT-MIO.

The separation problem for the knapsack polyhedron is known to be NP-complete since the well-known NP-complete problem of optimizing a linear function on this polyhedron can be solved by the ellipsoid algorithm with a polynomial number of calls to an algorithm for solving the separation problem. However, the knapsack optimization problem is also well-known to be solvable in pseudopolynomial time using dynamic programming. In many instances of practical merit this allows the knapsack optimization problem to be solved in reasonable computation times, and the dynamic programming formulation for this problem can be found in almost any text on integer programming (see, for example, [19]).

Crowder, Johnson, and Padberg [11] used the ability to solve the knapsack optimization problem to develop a pseudopolynomial time algorithm for generating violated *minimal cover* inequalities for the knapsack polyhedron; specifically, their algorithm generates from among the class of minimal cover inequalities an inequality that is most violated in the L^1 norm or a proof that no such inequality exists. While it is possible for a point to satisfy all minimal cover inequalities without being contained in the knapsack polyhedron, Balas and Zemel [4] showed that all facets of the knapsack polyhedron could be generated by applying the operations of *lifting* and *complementing* to some minimal cover. Zemel [25] and Zemel and Hartvigsen [26] have presented a number of results regarding the complexity of questions related to finding, recognizing, and lifting facets of the knapsack polyhedron. One result of particular interest is that *sequential lifting* for the knapsack polyhedron can be accomplished in polynomial time. Nonetheless, it is not clear that if a violated inequality for the knapsack polyhedron exists that it can be generated by lifting a minimal cover inequality generated by Padberg's algorithm. Further, is it not clear that given a minimal cover from which a violated facet can be generated by lifting and complementing that such a facet can be generated in pseudopolynomial time (the polynomial time lifting theorem of Zemel guarantees a facet, not a violated facet).

While combinatorial approaches extending from the work of Padberg, Balas, Zemel, and many other important contributors do not lead to a pseudopolynomial time separation algorithm for the knapsack polyhedron, the existence of a pseudopolynomial time separation algorithm for the knapsack polyhedron is guaranteed by the complexity equivalence of separation and optimization as described in [13]. However, while theoretically satisfying, the algorithm described in these monographs is is based on the ellipsoid algorithm and as such is not a practical algorithm. Specifically, the algorithm solves the separation problem by using the ellipsoid algorithm to optimize over the polar of the knapsack polyhedron and invoking a knapsack optimization algorithm as a separation oracle. Further, with its facets implicitly defined by the extreme points of the knapsack polyhedron, the knapsack polar can have a number of facets exponential in the number of variables of the knapsack problem. Thus, practically efficient procedures such as the simplex algorithm or interior point methods that require an explicit description of a polyhedron cannot be used in place of the ellipsoid algorithm without modification. In [9] and [10] the author demonstrated how practically efficient nonellipsoid algorithms could be used to optimize a linear function on a polyhedron closely related to the polar polyhedron, thus allowing the exact knapsack separation problem to be solved efficiently in practice. However, the results in these papers were achieved by dealing intelligently with the large number of implicit constraints and not through reformulating the problem to overcome difficulties encountered with such a large number of constraints.

In this paper we present a linear program to solve the separation problem for the knapsack polyhedron that has a pseudopolynomial number of variables and constraints in the size of the knapsack problem instance. With knapsack separation an NP-complete problem and with the best known algorithm for the knapsack optimization problem having pseudopolynomial complexity this pseudopolynomial problem formulation is the most compact that can be expected. One practical implication of this result is that polynomial time linear programming algorithms that require an explicit formulation of a linear program — which includes most interior point algorithms — can be used to solve the knapsack separation problem in pseudopolynomial time. Further important practical implications are discussed in the conclusions section.

We also demonstrate two results related to the projection of the primal and dual polyhedra of the compact problem formulation. Specifically, it is demonstrated that the knapsack polyhedron can be obtained by projecting the primal polyhedron of the formulation onto an appropriate subspace and that the polar of the knapsack polyhedron can be obtained by projecting the dual polyhedron onto an appropriate subspace. These results are in keeping with a number of recent projection results such as those found in Balas and Pulleyblank [2] [3], Ball, Liu, and Pulleyblank [5], Barahona [6] [7], Barany, Van Roy, and Wolsey [8], Eppen and Martin [12], Jeroslow and Lowe [14], Liu [15], Maculan [16], Martin [17], Martin, Rardin, and Campbell [18], Pochet and Wolsey [20], Rardin and Wolsey [22], and Yannakakis [24]. Two excellent concise surveys of recent work on projection can be found in sections of Pulleyblank [21] and Wolsey [23].

2 The Problem Reformulation

Formally stated as an integer program, the knapsack problem is the following.

$$
\begin{aligned}
& && \max \textstyle\sum_{j=1}^{n} c_j x_j \\
& (K) && \text{s.t. } \textstyle\sum_{j=1}^{n} a_j x_j \le b \\
& && x_j \in \{0,1\}
\end{aligned}
$$

In this definition b and all c_j and a_j are assumed to be positive integers. We let $\mathcal{P}$ denote the convex hull of feasible integer solutions for this problem. Throughout this paper we let $\hat{x} \ge 0$ denote the point for which a separating hyperplane from $\mathcal{P}$ is sought.

The knapsack problem is well-known to be NP-complete but is one of a relatively small class of problems for which a pseudopolynomial time algorithm exists. While this algorithm and its interpretation as a shortest path problem are well-known, we take a moment to outline it in preparation for results to follow. The algorithm is based on dynamic programming with the recursive relation given by if $g(j-1,k) > g(j-1,k-a_j)+c_j$ then

$g(j,k) = g(j-1,k)$

$pr(j,k) = (j-1,k)$

else

$g(j,k) = g(j-1,k-a_j)+c_j$

$pr(j,k) = (j-1,k-a_j)$

with $g(0,k)$ defined to be 0 so that $g(j,k)$ conceptually represents the optimal solution to the problem

$$\begin{aligned} \max \quad & \textstyle\sum_{t=1}^{j} c_t x_t \\ \text{s.t.} \quad & \textstyle\sum_{t=1}^{j} a_t x_t \leq k \\ & 0 \leq x_t \leq 1 \\ & x_t \text{ integer.} \end{aligned}$$

The predecessor array $pr(j,k)$ implicitly defines the x_j values by

$$x_j = 0 \text{ if } pr(j,k) = (j-1,k)$$

$$x_j = 1 \text{ if } pr(j,k) = (j-1,k-a_j).$$

The value $g(n,b)$ is thus the optimal solution value of the knapsack problem. In general, each stage j of the dynamic program can have as many as b states k and thus this dynamic programming formulation is only pseudopolynomial in the size of the problem it seeks to solve. Specifically, the complexity of the algorithm is $O(nb)$, however in practice the actual number of calculations can be far smaller.

The recursive relation can be viewed as defining a graph $G = (V,E)$ which we will find need to reference. Let each vertex $\{(j,k) : j = 0,\ldots,n;\ k = 0,\ldots,b\}$ of this graph correspond to one possible stage/state pair in the dynamic program, and let the directed edge set correspond to the pairs defined by the recursive relation; specifically, let the edge set consist of the edges (j,k)–$(j-1,k)$ and (j,k)–$(j-1,k-a_j)$ for $j = 1,\ldots,n$ and $k = 0,\ldots,b$, disallowing edges in this definition that are not contained in a path emanating from vertex (n,b) or that are incident to nonexistent vertices.

With G defined in this way there is a natural one-to-one correspondence between directed paths in G and feasible solutions to the knapsack problem (K). Conceptually, the optimal solution to the knapsack problem can be found by determining a longest path in this graph from (n, b) to any vertex $(0, k)$ where the edge weights are 0 for edges (j, k)–$(j-1, k)$ and c_j for edges (j, k)–$(j-1, k-a_j)$.

At the heart of the linear program to be described is a close formal relation to the following linear program which seeks to express $\hat{x}$ as a convex combination of the N extreme points x^p of the knapsack polyhedron $\mathcal{P}$.

$$
\begin{array}{llcccccl}
 & \min & & \sum_{j=1}^{n} s_j^1 & + & \sum_{j=1}^{n} s_j^2 & & \\
(C) & \text{s.t.} & \sum_{p=1}^{M} x_j^p y_p & + \; s_j^1 & & - \; s_j^2 & = \; \hat{x}_j & j = 1, \ldots, n \\
 & & \sum_{p=1}^{M} y_p & & & & = \; 1 & \\
 & & y_p \geq 0 & s^1 \geq 0 & & s^2 \geq 0 & &
\end{array}
$$

The fact that (C) solves the knapsack separation problem can be seen by looking at its dual.

$$
\begin{array}{lll}
 & \max & \mu\hat{x} + \sigma \\
(D) & \text{s.t.} & \mu x^p + \sigma \leq 0 \quad p = 1, \ldots, M \\
 & & -1 \leq \mu_j \leq 1 \quad j = 1, \ldots, n
\end{array}
$$

If $\hat{x} \notin \mathcal{P}$ then the optimal value of (C) and therefore (D) is strictly positive in which case $\mu\hat{x} > -\sigma$ while $\mu x^p \leq -\sigma$ for all $p = 1, \ldots, M$ and thus for $\mathcal{P}$. The difficulty with solving (C), of course, is that it will in general have an exponential number of columns. We proceed to develop a linear program that solves (C) that has a pseudopolynomial number of rows and columns.

To the graph G defined above we add an additional "sink" vertex $(-1, 0)$ and edges $(0, k)$–$(-1, 0)$ for all relevant k to G and call the augmented graph $\overline{G} = (\overline{V}, \overline{E})$. Treating vertex (n, b) as a "source" vertex we seek to send one unit of flow from the source to the sink while attempting to force the flow through the "1" edges (j, k)–$(j-1, k-a_j)$ in each layer j of the graph to $\hat{x}_j$. Let $\overline{E}^1(j) = \{(j, k)\text{–}(j-1, k-a_j)\}$, $\overline{E}^1 = \bigcup_{j=1}^{n} \overline{E}^1(j)$, and $\overline{E}^0 = \overline{E}^1 \backslash \overline{E}^1$. Further, let u_e denote the flow through edge $e \in \overline{E}$ and let $Nu = \hat{b} = (-1, 0, \ldots, 0, 1)$ denote the network constraints restricting the net flow into the source to -1, the net flow into the sink to 1, and the net flow into each remaining vertex to 0. The linear program we seek to solve is the following.

$$
(E)\quad
\begin{array}{llcccccl}
\min & & & \sum_{j=1}^{n} t_j^1 & + & \sum_{j=1}^{n} t_j^2 & & \\
\text{s.t.} & \sum_{e\in \overline{E}^1(j)} u_e & + & t_j^1 & - & t_j^2 & = \hat{x}_j & j=1,\ldots,n \\
 & Nu = \hat{b} & & & & & & \\
 & u \geq 0 & & t^1 \geq 0 & & t^2 \geq 0 & &
\end{array}
$$

By the construction of $\overline{G}$ there are a maximum of nb network constraints and $2nb$ variables u_e, so that the problem formulation (E) is pseudopolynomial in the size of the original knapsack separation problem. We will also find need for the dual of (E).

$$
(F)\quad
\begin{array}{llcccccl}
\max & \sum_{j=1}^{n} \hat{x}_j \mu_j & - & \omega_{n,b} & + & \omega_{-1,0} & & \\
\text{s.t.} & & - & \omega_{j,k} & + & \omega_{j-1,k} & \leq 0 & (j,k)\text{-}(j-1,k) \in \overline{E}^0 \\
 & \mu_j & - & \omega_{j,k} & + & \omega_{j-1,k-a_j} & \leq 0 & (j,k)\text{-}(j-1,k-a_j) \in \overline{E}^1 \\
 & -1 \leq \mu_j \leq 1 & & & & & &
\end{array}
$$

For notational convenience let $F(\mathrm{X})$ denote the set of feasible points for linear program (X). Further, let $E(x^p)$ denote the path in G corresponding to the extreme point x^p of $\mathcal{P}$. Formally, $E(x^p)$ is the set of n edges $\{(j, b - \sum_{i=j+1}^{n} a_i x_i^p)\text{-}(j-1, b - \sum_{i=j}^{n} a_i x_i^p)\}$. Letting $E(x^p)$ end at vertex $(0,k)$ we let $\overline{E}(x^p) = E(x^p) \cup \{(0,k)\text{-}(-1,0)\}$.

To understand the relationship between the problems (C) and (E) we define a function $\gamma : F(\mathrm{C}) \rightarrow F(\mathrm{E})$. Conceptually, γ is most easily interpreted as assigning a flow of y_p to the path $\overline{E}(x^p)$. Letting $\delta_p(e) = 1$ if $e \in \overline{E}(x^p)$ and 0 otherwise, the edge flows defined by the function γ are given by

$$u_e = \sum_{p=1}^{M} y_p \delta_p(e)$$

with the values t_j^1 and t_j^2 uniquely defined to satisfy $\sum_{e\in E^1(j)} + t_j^1 - t_j^2 = \hat{x}_j$, with $t_j^1 \geq 0$, $t_j^2 \geq 0$, and either $t_j^1 = 0$ or $t_j^2 = 0$.

Theorem 1 *The function γ maps $F(C)$ onto $F(E)$, and a vector (y,s) is optimal for (C) if and only if $\gamma(y,s)$ is optimal for (E). Further, there exists a pseudopolynomial time algorithm for calculating $(y,s) \in \gamma^{-1}(u,t)$ for any $(u,t) \in F(E)$.*

Proof . We first show that γ maps $F(\mathrm{C})$ into $F(\mathrm{E})$. Using the path flow interpretation of γ it is easy to see that the net flow through each vertex is 0 for all vertices in $\overline{G}$ other than (n,b) and $(-1,0)$, and since $\sum_{p=1}^{M} y_p = 1$ it follows that the net flow out of (n,b) and into $(-1,0)$ is 1. Thus, u as defined by γ satisfies the network flow constraints in (E). Further, note that

$$\begin{aligned} \sum_{e\in\overline{E}^1(j)} u_e = & \sum_{e\in\overline{E}^1(j)} \sum_{p=1}^{M} y_p \delta_p(e) \\ = & \sum_{p=1}^{M} y_p \sum_{e\in\overline{E}^1(j)} \delta_p(e) \\ = & \sum_{p=1}^{M} y_p x_j^p . \end{aligned} \tag{1}$$

Thus, setting $t^1 = s^1$ and $t^2 = s^2$ it is clear that γ maps $F(\mathrm{C})$ into $F(\mathrm{E})$ and that γ preserves objective function value.

The fact that γ maps $F(\mathrm{C})$ onto $F(\mathrm{E})$ (and that γ is many-to-one) is demonstrated by the following procedure for generating a solution $(y,s) \in \gamma^{-1}(u,t)$. Given any $(u,t) \in F(\mathrm{E})$ it is easily demonstrated that it can be decomposed into a collection of flows y_p on at most $|\overline{E}|$ paths $\overline{E}(x^p)$ in time polynomial in $|\overline{V}|$ and $|\overline{E}|$ and therefore in time polynomial in n and b (see, for example, [1] pp. 236-238). Due to the specific form of the network flow constraints in (E) such a decomposition has the property that $\sum_{p=1}^{M} y_p = 1$. Further, as a path decomposition of edge flows in $\overline{G}$, y and u are related by (1) so that letting $s^1 = t^1$ and $s^2 = t^2$ the constructed solution $(y,s) \in F(\mathrm{C})$. It is clear that $\gamma(y,s) = (u,t)$, and since objective function value is preserved under γ and γ^{-1} the proof is complete. □

The previous theorem demonstrates how the problem formulation (E) solves the knapsack separation problem. Since $\hat{x} \in \mathcal{P}$ if and only if the optimal value to (C) is 0 and since the mapping γ provides a correspondence between solutions to (C) and (E) which preserves objective function value it follows that $\hat{x} \in \mathcal{P}$ if and only if the optimal value to (E) is 0. If $\hat{x} \in \mathcal{P}$ the values y_p constructed as outlined in the above theorem give a representation of $\hat{x}$ as a convex combination of a pseudopolynomial number of extreme points of $\mathcal{P}$. If $\hat{x} \notin \mathcal{P}$ an optimal solution to (D) can be constructed in turn and, as mentioned earlier, defines a plane separating $\hat{x}$ from $\mathcal{P}$. Thus, an optimal solution to (E) represents an optimal solution to the knapsack separation problem.

3 Projected Polyhedra

In the previous section it was outlined how an optimal solution to the knapsack separation problem could be constructed in pseudopolynomial time from an optimal solution to (E). In point of fact, an optimal solution to (D) can be constructed immediately from an optimal solution to (F). However, this follows from more general results relating the set of feasible solutions to (F) and (D). Specifically, we proceed to show that the knapsack polyhedron $\mathcal{P}$ can be obtained by projecting the set of feasible solutions to (E) onto an appropriate subspace and that the polyhedron $F(\mathrm{D})$ of valid inequalities for $\mathcal{P}$ can be obtained by projecting the set of feasible solutions to (F) onto an appropriate subspace. The primal/dual pair of linear programs (E) and (F) — both with a pseudopolynomial dimension and number of facets — therefore play the role of an intermediary between the knapsack polyhedron $\mathcal{P}$ and its polar $F(\mathrm{D})$, both of which have polynomial dimension but a potentially exponential number of facets.

The results to be presented can be developed in the context of projecting the polyhedra $F(\mathrm{E})$ and $F(\mathrm{F})$ onto appropriate non-coordinate subspaces. However, the development is conceptually simpler if the formulations of $F(\mathrm{E})$ and $F(\mathrm{F})$ are first augmented with a small number of variables and constraints. For this reason we introduce the following problem (E′) and its corresponding polyhedron of feasible solutions $F(\mathrm{E}')$.

$$
(E')\qquad
\begin{array}{llllllll}
\min & & & \sum_{j=1}^{n} t_j^1 & + & \sum_{j=1}^{n} t_j^2 & & \\
\text{s.t.} & \sum_{e\in E^1(j)} u_e & + & t_j^1 & - & t_j^2 & = \hat{x}_j & j=1,\dots,n \\
 & Nu = \hat{b} & & & & & & \\
 & \sum_{e\in E^1(j)} u_e & - & x_j & & & = 0 & j=1,\dots,n \\
 & u \geq 0 & & t^1 \geq 0 & & t^2 \geq 0 & &
\end{array}
$$

Note that there is a natural one-to-one correspondence between solutions $(u,t)\in F(\mathrm{E})$ and $(u,t,x)\in F(\mathrm{E}')$ determined by including or excluding the variables x.

Theorem 2 *The projection $F_p(E')$ of $F(E')$ onto the n-dimensional subspace corresponding to the variables x is $\mathcal{P}$.*

Proof. We begin by showing that $\mathcal{P} \subseteq F_p(\mathrm{E}')$. If $\overline{x} \in \mathcal{P}$ it can be expressed as $\overline{x} = \sum_{p=1}^{M} x^p y_p$ for some vector $y \geq 0$ satisfying $\sum_{p=1}^{M} y_p = 1$. Clearly, s can be chosen appropriately so that $(y, s) \in F(\mathrm{C})$. By (1) the corresponding solution $(u, t) \in F(\mathrm{E})$ defined by the mapping γ of Theorem 1 satisfies

$$\sum_{e \in E^1(j)} u_e = \sum_{p=1}^{M} x_j^p y_p = \overline{x}_j.$$

Thus, $(u, t, \overline{x}) \in F(\mathrm{E}')$ is the solution corresponding to $(u, t) \in F(\mathrm{E})$, and this point projects onto $\overline{x} \in F_p(\mathrm{E}')$. Thus, any $\overline{x} \in \mathcal{P}$ is the projection of some point in $F(\mathrm{E}')$, or $\mathcal{P} \subseteq F_p(\mathrm{E}')$.

Similarly, consider any $(u, t, \overline{x}) \in F(\mathrm{E}')$. Let $(y, s) \in \gamma^{-1}(u, t)$ for γ as defined in Theorem 1. By (1) it follows that

$$\sum_{p=1}^{M} x_j^p y_p = \sum_{e \in E^1(j)} u_e = \overline{x}_j$$

and since $(y, s) \in F(\mathrm{C})$ it follows that $y \geq 0$ and $\sum_{p=1}^{M} y_p = 1$. Thus, $\overline{x} \in \mathcal{P}$ implying $F_p(\mathrm{E}') \subseteq \mathcal{P}$ and therefore $F_p(\mathrm{E}') = \mathcal{P}$. $\square$

In order to develop a dual projection result it is useful to introduce the following problem (F′) which is nothing more than the problem (F) with one additional constraint.

$$
(F') \quad
\begin{array}{llcccccll}
\max & \sum_{j=1}^{n} \hat{x}_j \mu_j & - & \omega_{n,b} & + & \omega_{-1,0} & & & \\
\text{s.t.} & & - & \omega_{j,k} & + & \omega_{j-1,k} & \leq & 0 & (j,k)\text{–}(j-1,k) \in \overline{E}^0 \\
 & \mu_j & - & \omega_{j,k} & + & \omega_{j-1,k-a_j} & \leq & 0 & (j,k)\text{–}(j-1,k-a_j) \in \overline{E}^1 \\
 & -\sigma & - & \omega_{n,b} & + & \omega_{-1,0} & = & 0 & \\
 & -1 \leq \mu_j \leq 1 & & & & & & &
\end{array}
$$

Note that there is a natural one-to-one correspondence between solutions $(\mu, \omega) \in F(\mathrm{F})$ and $(\mu, \omega, \sigma) \in F(\mathrm{F}')$ determined by including or excluding σ.

Theorem 3 *The projection $F_p(F')$ of $F(F')$ onto the $n+1$-dimensional subspace corresponding to the variables μ and σ is $F(D)$.*

Proof . We begin by showing that $F_p(\mathrm{F}') \subseteq F(\mathrm{D})$. Consider any path $\overline{E}(x^p)$ in $\overline{G}$. Adding the constraints in (F′) corresponding to vertices visited by the path the resultant constraint is

$$\textstyle\sum_{j:x_j^p=1}\mu_j + \omega_{n,b} - \omega_{-1,0} \leq 0.$$

As this is true for any path $\overline{E}(x^p)$ and recognizing that $\sum_{j:x_j^p=1}\mu_j = \mu x^p$ and $\omega_{n,b} - \omega_{-1,0} = \sigma$, it follows that (μ, σ) is feasible for (D) and thus $F_p(\mathrm{F}') \subseteq F(\mathrm{D})$.

To see that $F(\mathrm{D}) \subseteq F_p(\mathrm{F}')$, consider any point $(\overline{\mu}, \overline{\sigma}) \in F(\mathrm{D})$. We proceed by constructing a vector ω such that $(\overline{\mu}, \omega, \overline{\sigma}) \in F(\mathrm{F}')$. Let $\overline{\mu}$ serve as the objective function for the knapsack problem (K) and consider the result from applying the dynamic programming algorithm to solve (K). By virtue of the dynamic programming recursion, $g(j,k) \geq g(j-1,k)$ for every edge (j,k)–$(j-1,k) \in E^0$ and $g(j,k) \geq g(j-1,k-a_j) + \overline{\mu}_j$ for every edge (j,k)–$(j-1,k-a_j) \in E^1$. Thus, setting $\omega_{j,k} = g(j,k)$ for all vertices $(j,k) \in V$, setting $\omega_{-1,0}$ equal to any non-positive number, and setting $\sigma = -\omega_{n,b} + \omega_{-1,0}$ it is easy to see that that $(\overline{\mu}, \omega, \overline{\sigma})$ is feasible for (F′). It remains only to show that $\omega_{-1,0}$ can be chosen appropriately so that $\overline{\sigma} = -\omega_{n,b} + \omega_{-1,0}$. To this end, note that since $(\overline{\mu}, \overline{\sigma})$ is feasible for (D) by assumption, $\overline{\mu} x^p + \overline{\sigma} \leq 0$ for $p = 1, \ldots, M$, and since $g(n,b) = \max\{\overline{\mu} x^p : p = 1, \ldots, M\}$ it follows that there exists a $\theta \geq 0$ such that $g(n,b) + \theta + \overline{\sigma} = 0$. Setting $\omega_{-1,0} = -\theta$ we have $-\omega_{n,b} + \omega_{-1,0} = -g(n,b) - \theta = \overline{\sigma}$. Thus, $(\overline{\mu}, \overline{\sigma})$ is the projection of $(\overline{\mu}, \omega, \overline{\sigma})$ as constructed, implying $F(\mathrm{D}) \subseteq F_p(\mathrm{F}')$ and therefore that $F(\mathrm{D}) = F_p(\mathrm{F}')$. □

An immediate implication of Theorem 3 is that an optimal solution to the knapsack separation problem can be read directly from an optimal solution to (F) (or (E)), namely, $\mu x \leq -\omega_{n,b} + \omega_{-1,0}$.

4 Conclusions

A linear programming formulation of the knapsack separation problem has been presented which contains a pseudopolynomial number of variables and constraints. By relating the primal and dual polyhedra of this formulation to the knapsack polyhedron and its polar it was demonstrated how an optimal solution to the knapsack separation problem could be constructed immediately from the solution to this formulation.

The potential practical significance of this result is that it provides a way to solve the knapsack separation problem by solving a linear program that in many instances of practical merit may be quite small. Very often the knapsack problems which arise as the constraints of larger integer programs have a small number of variables and this in turn translates into a small number of variables in the problem formulation (E). Further, it is often possible to further restrict the number of variables over which to search for a separating hyperplane using observations about the point $\hat{x}$ being separated. A detailed discussion of restrictions on the search space is contained in Boyd [10].

Another important advantage of the formulation (E) is that it is fundamentally a shortest path problem with a small number of highly structured side constraints. It is therefore possible to devise fast, combinatorial heuristic procedures for constructing feasible solutions that are near optimality. An efficient primal linear programming code, available either as a package or designed specifically to solve problem (E), could then be used to solve (E) to optimality from this nearly optimal starting solution. Alternatively, it is easy to see that for a fixed μ a feasible solution for (F) can be found by the dynamic programming programming algorithm outlined for solving the knapsack optimization problem. Thus, if a hyperplane $\mu x + \sigma \leq 0$ that nearly separates $\mathcal{P}$ and $\hat{x}$ is known, perhaps from solving (F) at an earlier iteration with a similar $\hat{x}$ as would be the case in integer programming applications, a feasible and nearly optimal solution to (F) can be found using a combinatorial procedure and (F) solved to optimality from this nearly optimal solution using a linear programming algorithm. This second approach has the advantage that the linear program need not be solved to optimality to generate a cutting plane. It is only necessary that a feasible solution with a positive objective function value be found.

Finally, while a pseudopolynomial time algorithm for the knapsack separation problem exists by virtue of the existence of linear programming algorithms that are polynomial in the size of the problem instance, the existence of a pseudopolynomial time *combinatorial* algorithm for this problem remains an open question. As just mentioned, apart from complicating side constraints in the case of (E) and additional variables in the case of (F) these problems have fundamentally combinatorial interpretations. Whether these interpretations can be effectively exploited deserves further attention.

References

[1] Ahuja, R. K., T. L. Magnanti, and J. B. Orlin. 1989. Network Flows. In *Optimization*, G. L. Nemhauser *et. al.* eds., *Handbooks in Operations Research and Management Science* 1, 529-572. North Holland, New York.

[2] Balas, E. and W. R. Pulleyblank. 1983. The Perfectly Matchable Subgraph Polytope of a Bipartite Graph. *Networks* 13, 486-516.

[3] Balas, E. and W. R. Pulleyblank. 1987. The Perfectly Matchable Subgraph Polytope of an Arbitrary Graph. Research Report 87470-OR, Institute for Operations Research, Bonn University.

[4] Balas, E. and E. Zemel. 1984. Lifting and Complementing Yields all the Facets of Positive Zero-One Programming Polytopes. In *Mathematical Programming*, R. W. Cottle *et. al.* eds. Elsevier Science Publishers, North Holland.

[5] Ball, M. O., W. G. Liu, and W. R. Pulleyblank. 1987. Two Terminal Steiner Tree Polyhedra. Research Report CORR 87-33, Department of Combinatorics and Optimization, University of Waterloo.

[6] Barahona, F. 1988. On Cuts and Matchings in Planar Graphs. Research Report 88503-OR, Institute for Operations Research, Bonn University.

[7] Barahona, F. 1988. Reducing Matching to Polynomial Size Linear Programming. Research Report CORR 88-51, Department of Combinatorics and Optimization, University of Waterloo.

[8] Barany, I., T. J. Van Roy, and L. A. Wolsey. 1984. Strong Formulations for Multi-item Capacitated Lot-sizing. *Management Science* 30 1255-1271.

[9] Boyd, E. A. 1990. The Lagrangian and Other Primal Cutting Planes for Linear Integer Programming Problems. Technical Report TR90-3, Department of Mathematical Sciences, Rice University.

[10] Boyd, E. A. 1990. Fenchel Cutting Planes for Linear Integer Programming Problems. Technical Report TR90-20, Department of Mathematical Sciences, Rice University.

[11] Crowder, H., E. L. Johnson, and M. W. Padberg. 1983. Solving Large-scale Zero-one Linear Programming Problems. *Operations Research* **31**, 803-834.

[12] Eppen, G. D. and R. K. Martin. 1985. Solving Multi-item Capacitated Lot-sizing Problems Using Variable Redefinition. Working paper, Graduate School of Business, University of Chicago.

[13] Grötschel, M., L. Lovász, and A. Schrijver. 1988. *Geometric Algorithms and Combinatorial Optimization*. Springer-Verlag, New York.

[14] Jeroslow, R. G. and J. K. Lowe. 1984. *Mathematical Programming Study* **22** 167-184.

[15] Liu, W. G. 1988. Extended Formulations and Polyhedral Projection. Ph.D. Thesis, Department of Combinatorics and Optimization, University of Waterloo.

[16] Maculan, N. 1987. The Steiner Problem in Graphs. *Annals of Discrete Mathematics* **33** 185-192.

[17] Martin, R. K. 1987. Using Separation Algorithms to Generate Mixed Integer Model Reformulations. Research Report, Graduate School of Business, University of Chicago.

[18] Martin, R. K., R. L. Rardin, and B. A. Campbell. 1987. Polyhedral Characterization of Discrete Dynamic Programming. Research Report CC-87-24, Institute for Interdisciplinary Engineering Studies, Purdue University.

[19] Nemhauser, G. L. and L. A. Wolsey. 1988. *Integer and Combinatorial Optimization*. Wiley and Sons, New York.

[20] Pochet, Y. and L. A. Wolsey. 1988. Lot-sizing Models with Backlogging: Strong Formulations and Cutting Planes. *Mathematical Programming* **40** 317-335.

[21] Pulleyblank, W. R. 1989. Polyhedral Combinatorics. In *Optimization*, G. L. Nemhauser *et. al.* eds., *Handbooks in Operations Research and Management Science* 1, 529-572. North Holland, New York.

[22] Rardin, R. L. and L. A. Wolsey. 1990. Valid Inequalities and Projecting the Multicommodity Extended Formulation for Uncapacitated Fixed Charge Network Flow Problems. Research Report CC-90-2, Institute for Interdisciplinary Engineering Studies, Purdue University.

[23] Wolsey, L. A. 1989. Strong Formulations for Mixed Integer Programming: A Survey. *Mathematical Programming B* 45 173-191.

[24] Yannakakis, M. 1988. Expressing Combinatorial Optimization Problems by Linear Programs. Research Report, AT&T Bell Laboratories.

[25] Zemel, E. Easily Computable Facets of the Knapsack Polytope. 1987. Working Paper 713, Department of Managerial Economics and Decision Sciences, Northwestern University.

[26] Zemel, E. and D. Hartvigsen. On the Complexity of Lifted Inequalities for the Knapsack Problem. 1987. Working Paper 740, Department of Managerial Economics and Decision Sciences, Northwestern University.