Compact vs. Exponential-Size LP Relaxations

Robert D. Carr and Guiseppe Lancia

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company, for the United States Department of Energy under Contract DE-AC04-94AL85000.

Approved for public release; further dissemination unlimited.
DISCLAIMER

Portions of this document may be illegible in electronic image products. Images are produced from the best available original document.
Compact vs. Exponential–size LP Relaxations

Robert D. Carr and Giuseppe Lancia
Applied Mathematics Department
Sandia National Laboratories
P.O. Box 5800
Albuquerque, NM 87185-1110

Abstract

In this paper we introduce by means of examples a new technique for formulating compact (i.e. polynomial–size) LP relaxations in place of exponential–size models requiring separation algorithms. In the same vein as a celebrated theorem by Grötschel, Lovász and Schrijver, we state the equivalence of compact separation and compact optimization. Among the examples used to illustrate our technique, we introduce a new formulation for the Traveling Salesman Problem, whose relaxation we show equivalent to the subtour elimination relaxation.
1 Introduction

In the past few decades, Linear Integer Programming has emerged as perhaps the most powerful technique for solving exactly difficult combinatorial problems. The approach relies on formulating a problem as the minimization or maximization of a linear function over integer variables satisfying a set of linear constraints. The Linear Programming (LP) Relaxation, consisting in optimizing without requiring the variables to be integer, is polynomially solvable, and its value is used as a bound in an implicit enumeration (branch and bound) scheme. A formulation is as successful as the strength of its LP bound. That is, if the value of the objective function over the relaxation is close to the value over the integers, then the bound, and hence the pruning of the search space, will be effective. The choice of a tight formulation is therefore the key to an effective solution. Sometimes the tightest formulations require a very large (exponential) number of constraints or variables. A fundamental paper by Grötschel, Lovász and Schrijver ([5]) states that even if these LPs have an exponential size, this does not forbid their solvability in polynomial time, provided that separation can be done in polynomial time. Separation consists in solving the LP with only a polynomial subset of the constraints, and then checking if any of the –exponentially many– constraints that were left out is violated by the optimal solution. If this is the case, one such constraint is found and added to the current LP, which is then solved again. By using the ellipsoid method as the LP solver, Grötschel, Lovász and Schrijver showed that the above procedure converges in polynomial time.

In this work we show how oftentimes the whole process of iterating the solution of several Linear Programs and the combinatorial algorithm for separation can be replaced by the use of a suitable, single, polynomial–size (or, as we shall call it, compact) Linear Program. This idea is not only interesting from a theoretical point of view, but has also some useful practical applications. For instance, many of the commercially available integer programming solvers cannot be programmed or linked to user–developed code, but rely instead on the solution of a given input formulation by means of some general procedure (typically, a built–in branch and bound code enhanced by some general–purpose cuts). In particular, focusing on modeling the problem with the right variables and constraints, and then letting the package take care of the solution, is the approach typically used by engineers working in industry, while more sophisticated applications are developed by researchers in the academic world. If one gives up the possibility of writing a procedure which alternates separation and LP–solution, he could be forced to use inferior (weaker) formulations than others with exponentially many
constraints. By using our approach, however, the separation algorithm can be "hardcoded" into a polynomial number of new variables and constraints, leading to a "static" LP–relaxation as tight as the "dynamic" one, requiring separation on-line.

Another aspect worth mentioning is the following. Sometimes, the optimization undergoes long runs of LP solution/separation in which the new-found constraints affect only slightly the objective function. At the end, a very large number of runs may have been required, resulting in a very large running time. The contact map overlap problem (third example in this paper) is an example of such behavior. When this is the case, it is very possible that the solution of a single LP (although bigger) may be faster than that on many LPs. Furthermore, after years of refinements, state-of-the-art algorithms are today known for the typical separation problems, which are usually famous combinatorial problems such as shortest path and maximum flow. Hence, there is probably not much to be expected in practical improvements on those algorithms. However, many developments and increased interest have recently been shown in speeding up traditional as well as new, polynomial and non–polynomial, procedures for Linear Programming. Any such improvements would translate in the speed–up of the compact version of the relaxation vs. the one based on separation.

Finally, from a theoretical standpoint, it is interesting to note that, although the issue is not usually addressed, polynomial convergence of the separation method relies on the ellipsoid algorithm for solving the LPs, which is in practice never used. Our approach, however, does only require the use of any polynomial algorithm for Linear Programming, such as the now widely available interior–point methods.

2 Optimization and Separation

Let

\[ \text{(IP)} \quad \min \{cx \mid Ax \geq b, x \in \mathbb{Z}_+^n \} \]

be a formulation with an exponential number of constraints for an integer programming problem. Let \( I \) be the index set for the rows of \( A \). In a very general way, we can usually partition the rows of \( A \) into two subsets: \( I_p \), a family with a polynomial number of constraints, and \( I_e \), a family with an exponential number of constraints. For instance, in the Traveling Salesman Problem, \( I_p \) would consist of the degree constraints, while \( I_e \) indexes the
subtour-elimination constraints. We can then rewrite the problem as

$$\begin{align*}
(\text{IP}) \quad & \min_{x \in \mathbb{Z}_+^n} c^T x \\
\text{subject to} \quad & a_i x \geq b_i \quad i \in I_p \\
& a_i x \geq b_i \quad i \in I_e.
\end{align*}$$

Let us call $P_x$ the polyhedron defined by (3), (4) and $x \geq 0$. An inequality $ax \geq b$ is valid for $P_x$ if it is satisfied by each $x \in P_x$. In [5], Grötschel, Lovász and Schrijver show that (LP), the LP-relaxation of (IP), is solvable in polynomial time provided that the following separation problem is solvable in polynomial time:

**Separation:** Given a vector $x^* \in \mathbb{R}_+^n$, with $a_i x^* \geq b_i$ for $i \in I_p$, determine if $x^* \in P_x$ or find a violated valid inequality.

The proof in [5] relies on the ellipsoid method for Linear Programming. Based on this technique, several works have appeared over the years using tight LP-relaxations with an exponential number of constraints. Furthermore, by taking the dual of an LP-relaxation, the same technique can be applied for solving models with an exponential number of variables, leading to an approach known as branch-and-price. A separation problem is almost invariably an optimization problem, whose optimal value is used to determine if $x^* \in P_x$, and whose optimal solution identifies a violated inequality. Separation problems are typically solved by well-known polynomial combinatorial algorithms, such as shortest path, maximum flow, minimum cost flow, and matching, bipartite and non. With the notable exception of the nonbipartite matching, we noticed that the vast majority of these separation algorithms can also be phrased and solved as Linear Programs. Our main idea is to replace the exponentially many constraints in (4) with a polynomial family $I'_p$ of new constraints, in the variables $x$ and $y$ ($y$ being a new set of polynomially many variables), which essentially represent an LP formulation for the separation problem. That is, we suitably lift the problem (LP) in the $(x,y)$ space, so that $x^*$, the projection on the $x$-space of an optimal solution $(x^*, y^*)$, does not violate any of the constraints originally in (4). We therefore obtain the following compact formulation:

$$\begin{align*}
(\text{CLP}) \quad & \min_{x \in \mathbb{R}_+^n} cx \\
\text{subject to} \quad & a_i x \geq b_i \quad i \in I_p
\end{align*}$$
This formulation has now a polynomial number of constraints and can be solved in polynomial time. We call $P_{xy}$ the polyhedron, in the $(x, y)$ space, defined by (6), (7) and $x \geq 0$.

We now come to define the concepts of compact separation and compact optimization, and prove that they are equivalent. Given the formulation (LP), with an exponential number of constraints defining the polyhedron $P_x$, compact optimization is the existence of a polynomial–size Linear Program, i.e. (CLP), defining a polyhedron $P_{xy}$ whose projection on the $x$–space is equal to $P_x$. Compact separation is, loosely speaking, the possibility of stating a separation algorithm for (LP) as a polynomial–size Linear Program. More precisely, we say that (LP) has a compact separation if there exist a matrix $D$ of size $N' \times (n + 1 + N'')$ and a vector $d$ of size $N' \times 1$, such that

1. $N'$ and $N''$ are polynomial in $n$.
2. Let a vector $x^* \in R^n$, with $a_i x^* \geq b_i$ for $i \in I_p$, be given. Then the Linear Program (SEP), in the variables $(a \in R^n, b \in R, w \in R^{N''})$, has an optimal value smaller than 0 if and only if $x^*$ violates some constraints in $I_e$. In this case, $a' x \geq b'$ is a violated valid inequality, where $(a', b', w')$ is the optimal solution of (SEP).

\begin{equation}
\begin{align*}
\text{(SEP)} & \quad \min x^* a - b \\
\text{subject to} & \quad D \cdot \begin{pmatrix} a \\ b \\ w \end{pmatrix} \geq d.
\end{align*}
\end{equation}

We have the following

**Theorem 1** For an exponential–size formulation (LP), compact optimization is possible if and only if compact separation is possible.

**Proof** (if:) Assume compact separation is possible. Since the optimal value of (SEP) is equal to the optimal value of its dual, we formulate the dual and force its value to be nonnegative (inequality (13)). This would correspond to have all inequalities in $I_e$ satisfied. We therefore obtain the following constraints, which are the family $I'_p$ that, added to $I_p$, yields a compact relaxation of (IP). Note how $x$ was replaced for $x^*$.
where \( D = (D_a|D_b|D_w). \)

(only if.) Assume now that (LP) has a compact optimization like (CLP).
We can rewrite the constraints of (CLP) as

\[
(B_x|B_y) \cdot \begin{pmatrix} x \\ y \end{pmatrix} \geq g
\]

for some matrix \( B = (B_x|B_y) \) and vector \( g \). Let \( P_x \) be the polytope defined by (LP), which is also the projection of the polytope defined by (14) on the \( x \)-space. We are now going to describe a Linear Program that solves the separation problem. Let \( x^* \in R^*_n \) be given. \( x^* \in P_x \) iff \( (x^*, y^*) \) is feasible for (14) for some \( y^* \), i.e. the following Linear Program has a nonnegative optimum:

\[
\begin{align*}
\max \lambda \\
\text{subject to} \\
B_x x^* + B_y y - g \geq \lambda \mathbf{1}
\end{align*}
\]

where \( \mathbf{1} \) is a vector of value 1 in each component. We can rewrite the constraints (16) as

\[
(1 \mid -B_y) \cdot \begin{pmatrix} \lambda \\ y \end{pmatrix} \leq B_x x^* - g.
\]

By taking the dual of (15), (17), and introducing the new variables \( a \in R_n \) and \( b \in R \), we have that \( x^* \in P_x \) iff the following LP has nonnegative optimal value:

\[
\begin{align*}
\min_{w \geq 0} & \quad ax^* - b \\
\text{subject to} \\
w1 - wB_y &= (1 \mid 0 \ldots 0) \\
a &= wB_x \\
b &= wg.
\end{align*}
\]
If this LP has negative optimum, then the optimal solution \((a', b', w')\) identifies a valid inequality for \(P_x\) violated by \(x^*\). To see this, note that, since \(w' \geq 0\) and, from (19), \(w'1 = 1\), then \(w'(B_x x + B_y y - g) \geq 0\) is a valid inequality for (CLP). But, from (19), \(w' B_y = 0\) and hence the inequality \(w'B_x x \geq w'g\), i.e. \(a' x \geq b'\), is valid for \(P_x\); furthermore, it is violated by \(x^*\). Hence, (18)–(19) constitute a compact LP formulation of the separation algorithm, and the theorem is proved.

Note that, although this is not required for the validity of our theorem, it can be proved by projection theory (see Nemhauser and Wolsey [8]) that the inequality found by the compact separation algorithm is a maximally violated one.

3 Examples

3.1 The Minimum Routing Cost Tree Problem

The Minimum Routing Cost Tree (MRCT), [6, 4, 9], is a network–design problem which can be stated as follows: Given an undirected graph \(G = (V, E)\) with nonnegative lengths on the edges, find a spanning tree such that the sum over all pairs of vertices of the length of the path connecting them in the tree is minimized. Fischetti, Lancia and Serafini in [4] describe an integer programming formulation with an exponential number of variables, one for each possible path between two nodes. The model is then solved by Branch–and–Price, and the pricing subroutine is a shortest path algorithm for nonnegative lengths. We start illustrating our technique with this example for two reasons. First, the MRCT shows how also a primal formulation, with an exponential number of variables, can be modeled as a compact optimization problem. The answer, of course, is to turn the exponentially many variables into exponentially many constraints, by means of the dual problem. To the dual, we then apply our technique. Secondly, we are going to use the MRCT problem to derive a new IP formulation of the TSP, as described in the next section.

Let \(|V| = n\), \(|E| = m\), and \(V = \{1, \ldots, n\}\). The length of an edge \(e = \{i, j\}\) will be denoted as \(d_e\). A pair of vertices is an edge of the complete graph \(K_n = (V, \Pi)\). For a spanning tree \(T\) and a pair \(\{i, j\} \in \Pi\) of vertices, \(d(i, j, T)\) is the length of the unique path connecting \(i\) and \(j\) in \(T\). The routing cost of \(T\) is defined as \(rc(T) := \sum_{\{i,j\} \in \Pi} d(i, j, T)\). We want to determine a spanning tree of minimum routing cost.

For each pair \(h = \{i, j\} \in \Pi\), we denote by \(\mathcal{P}^h\) the set of simple paths in
$G$ between $i$ and $j$. The set of all paths in $G$ will be denoted by $\mathcal{P}$. Since an edge is also a path (of cardinality 1), we have $E \subseteq \mathcal{P}$. For each path $P \in \mathcal{P}$, we let $d_P := \sum_{e \in P} d_e$.

The MRCT problem is formulated as an integer program with decision variables $x_P$, for $P \in \mathcal{P}$, used to select a path between each pair of vertices. The constraints are such that, in a feasible solution, the set $\{e \in E \mid x_e = 1\}$ defines a tree. The LP–relaxation is the following:

\[ \text{(MRCT)} \quad \min_{x_P \geq 0} \sum_{P \in \mathcal{P}} d_P x_P \]
subject to

\[ \sum_{P \in \mathcal{P}^h} x_P \geq 1 \quad h \in \Pi \]
\[ \sum_{P \in \mathcal{P}^h : P \ni e} x_P \leq x_e \quad e \in E, h \in \Pi - \{e\} \]
\[ \sum_{e \in E} x_e = n - 1. \]

The model has an exponential number of variables. In order to use our technique, we first consider its dual. We define the following dual variables, associated to the constraints (23), (24), and (25) respectively: $u_h$, for $h \in \Pi$; $v_{eh}$, for $e \in E, h \in \Pi - \{e\}$; and $w$. The dual of (MRCT) is the following:

\[ \text{(DMRCT)} \quad \min_{u_h \geq 0, v_{eh} \geq 0} \sum_{h \in \Pi} (u_h - \sum_{P \in \mathcal{P}^h - \{h\}} v_{eh}) + (n-1) w \]
subject to

\[ u_e + \sum_{h \in \Pi - \{e\}} v_{eh} + w \leq d_e \quad e \in E \]
\[ u_h - \sum_{e \in P} v_{eh} \leq d_P \quad h \in \Pi, P \in \mathcal{P}^h - \{h\}. \]

A variable $x_P$, $|P| \geq 2$ with negative reduced cost in (MRCT) corresponds to a violated dual constraint (28). For a fixed $h = \{i, j\}$, define $d'_e = v_{eh} + d_e$ for all $e \in E$. Since constraints (28) can be rewritten as $u_h \leq d'_P$ for all $h \in \Pi, P \in \mathcal{P}^h - \{h\}$, the pricing algorithm in [4], consists in finding the shortest $i$–$j$ path in $E - \{h\}$, with respect to the costs $d'$, and checking if it is shorter than $u_h$.
The shortest path algorithm for nonnegative lengths can be modeled as a Linear Program as follows. For each pair \( h = \{i, j\}, \ i < j \), we introduce a variable \( y_u^h \) for each \( u \in V \), representing the length of the shortest \( i-u \) path given \( d' \) lengths. We have then the following constraints:

\[
\begin{align*}
y_i^h &= 0, \quad h = \{i, j\} \in \Pi, \quad i < j \\
y_j^h &\geq y_u^h, \quad h = \{i, j\} \in \Pi, \quad i < j \\
y_v^h &\leq y_u^h + v_{\{u,v\}} + d_{\{u,v\}}, \quad h \in \Pi, \quad \{u, v\} \in E.
\end{align*}
\]

By substituting constraints (29), (30) and (31) for constraints (28), we obtain a compact formulation of (DMRCT).

3.2 The Traveling Salesman Problem

The (symmetric) traveling salesman problem is perhaps the most famous combinatorial problems in the literature. As before, we are given an undirected graph \( G = (V, E) \) with nonnegative lengths \( d_e \) on the edges. A Hamilton tour is a cycle visiting each node in \( V \) exactly once. We want to determine the Hamilton tour of smallest length.

3.2.1 The subtour-elimination relaxation

The standard IP formulation for the STSP problem is based on binary variables \( x_{\{i,j\}} \) for each edge \( \{i, j\} \in E \). For a subset of vertices \( S \subseteq V \), let \( \delta(S) \) denote the set of edges with one endpoint in \( S \) and one out of \( S \). The \( O(n) \) degree constraints (33) force each node to be entered and exited exactly once by a feasible solution. However, without the \( O(2^n) \) subtour-elimination constraints (34), an integer solution could consist of a set of disjoint cycles.

\[
\begin{align*}
(TSP - SE) & \quad \min_{0 \leq x \leq 1} \sum_{\{i,j\} \in E} d_{\{i,j\}} \ x_{\{i,j\}} \\
& \quad \text{subject to} \\
& \quad \sum_{\{i,j\} \in \delta((i))} x_{\{i,j\}} = 2, \quad i \in V \quad (33) \\
& \quad \sum_{\{i,j\} \in \delta(S)} x_{\{i,j\}} \geq 2, \quad \emptyset \neq S \subset V. \quad (34)
\end{align*}
\]

The standard way of separating inequalities (34) is via the solution of \( O(n) \) maximum flow problems (see, e.g., Cook et al. [3]). Namely, for a
fractional solution $x^*$, direct each edge $\{i, j\}$ in the two possible ways creating the arcs $(i, j)$ and $(j, i)$, each with capacity $x^*_{[i,j]}$. Let $A$ be the set of arcs created this way. Since in (34) it is enough to consider only sets $S \neq V$ such that $1 \in S$, there are no violated constraints (34) if and only if the maximum flows from 1 to each other vertex in $V$ has value at least 2. It is well known that the maximum flow problem can be modeled as a Linear Program. To this end, define variables $y_{i,j}^u$ to represent the flow from node 1 to node $u$, traveling on the arc $(i, j)$, for all $u \in V - \{1\}$ and $(i, j) \in A$. We have then the following constraints (35), (36), (37) which can be substituted for (34) in (TSP-SE) to obtain a compact formulation of the Symmetric Traveling Salesman Problem equivalent to the subtour–elimination relaxation.

\[
\begin{align*}
y_{i,j}^u + y_{j,i}^u & \leq x^*_{[i,j]} \quad u \in V - \{1\}, \{i, j\} \in E \\
\sum_{(i, j) \in A} y_{i,j}^u - \sum_{(j, i) \in A} y_{j,i}^u & = 0 \quad u \in V - \{1\}, \ i \in V - \{1, u\} \\
\sum_{(1, j) \in A} y_{1,j}^u - \sum_{(j, 1) \in A} y_{j,1}^u & \geq 2 \quad u \in V - \{1\}.
\end{align*}
\]

3.2.2 A new IP formulation

As for the MRCT problem, we define a variable $x_P$ for each simple path $P$ in $G$. The following constraints force the set $\{e \in E | x_e = 1\}$ to be a hamilton tour, and $x_P = 1$ iff $P$ is a path contained in this cycle.

\[
(TSP-PF) \quad \min \sum_{e \in E} d_e x_e
\]

subject to

\[
\begin{align*}
\sum_{P \in P^h} x_P & \geq 2 \quad h \in \Pi \\
\sum_{P \in P^h : P \ni e} x_P & \leq x_e \quad e \in E, h \in \Pi - \{e\} \\
\sum_{e \in E} x_e & = n \\
x_e & \in \{0, 1\}, \quad x_P \geq 0 \quad e \in E, P \in \mathcal{P} - E.
\end{align*}
\]

We call this formulation for the STSP the path formulation. If we consider the LP–relaxation of the path formulation, it can be solved, similarly to the MRCT problem in [4], by using the shortest path algorithm as the
pricing subroutine. By taking its dual, we can obtain, as in section 3.1, a compact formulation of the LP relaxation for this new model of the STSP. The strength of this model, however, turns out to be exactly the same as that of the subtour-elimination relaxation.

**Theorem 2** The LP-relaxation of the path formulation and subtour elimination have the same value.

**Proof** Call $z_S$ the value of the (LP-relaxation of) the subtour elimination and $z_P$ the value of the path formulation. By using the Flow Decomposition Theorem (Ahuja et al., [1]), stating that any $u$-$v$ flow is the sum of a set of $u$-$v$ paths, a solution of (TSP-SE) can be mapped into a solution of (TSP-PF) of the same value, and conversely. Let $x_e$, $e \in E$, be a solution feasible for (TSP-SE). Since $x_e$ is feasible for all constraints (34), the maximum flow from each $u \in V$ to each $v \in V$ in the network with capacities $x_e$ is at least 2, and, by the flow decomposition theorem, there are paths $P$, with values that we denote by $x_p$, adding up to at least 2. Furthermore, by adding (33) for all nodes we get (41). So, $(x_e, x_P)$ is feasible for (TSP-PF) and hence $z_P \leq z_S$. Similarly, given a solution $(x_e, x_P)$ to (TSP-PF), it is not difficult to see that its projection $x_e$, $e \in E$, is feasible for (TSP-SE), and hence $z_S \leq z_P$.  

By some preliminary testing, it appears that the compact formulation for (TSP-SE) is solved faster (by using CPLEX) than that for (TSP-PF). However, (TSP-PF) may lead to a stronger bound, by replacing constraints (39) with

$$\sum_{p \in \Pi_h} x_p = 2 \quad h \in \Pi. \quad (43)$$

The problem in doing so, is that the pricing algorithm would become the shortest path in a graph with positive and negative-length edges. We have not been able to prove that there would not be negative-length cycles, and so the relaxation with $= 2$ in place of $\geq 2$ may in fact be NP-complete to optimize. Note that changing $\geq 2$ with $= 2$ in (TSP-SE) would not lead to a valid formulation of the STSP.

### 3.3 Non–crossing Matchings

Given two ordered sets $V_1 = \{v_1, \ldots, v_{n_1}\}$ with $v_1 < \ldots < v_{n_1}$ and $V_2 = \{u_1, \ldots, u_{n_2}\}$ with $u_1 < \ldots < u_{n_2}$, a non–crossing matching of $V_1$ and $V_2$ is a matching in the complete bipartite graph $B = (V_1 \cup V_2, V_1 \times V_2)$ such that, if
\(v_i\) is matched with \(u_j\) and \(v_k\) is matched with \(u_h\), then either \(k < i\) and \(h < j\) or \(k > i\) and \(h > j\). For simplicity, we can assume \(V_1 = V_2 = \{1, \ldots, n\}\) and introduce variables \(x_{ij}\) such that \(x_{ij} = 1\) iff \(v_i\) is matched with \(u_j\). Then, the non-crossing constraints are of the form

\[
x_{ij} + x_{hk} \leq 1 \quad 1 \leq i \leq h \leq n, \ 1 \leq k \leq j \leq n, \ \{i, j\} \neq \{h, k\}. \tag{44}
\]

If we draw \(B\) in the customary way, with \(V_1\) on the left and \(V_2\) on the right, a noncrossing matching can be seen as a set of non-crossing line segments between \(V_1\) and \(V_2\). Denote a generic such line by \([i, j]\). If \(G'\) the graph in which each node corresponds to a line of \(B\) and two nodes are connected if the lines do not cross, a set \(Q\) of lines which are all mutually crossing is a clique in \(G'\). Hence, we can strengthen inequalities (44) to (45), called clique inequalities:

\[
\sum_{[i,j] \in Q} x_{ij} \leq 1 \quad \text{for all } Q \text{ cliques in } G'. \tag{45}
\]

Non-crossing matchings appear as part of the model for many combinatorial optimization problems. In particular, in the field of Molecular Computational Biology, many problems require to align some genetic data, like DNA or protein sequences, or 3-dimensional folds of proteins. In an alignment, given two objects (e.g., DNA fragments) comprised of linear arrangements of simpler units (e.g., nucleotides), we seek a mapping of some units of one object into the other, which respects the linear orderings. Lenhof, Reinert, and Vingron [7] give an integer programming formulation for RNA multiple sequence alignment, which contains non-crossing matchings as part of the model. Carr, Lancia and Istrail [2], use non-crossing matching to model some of the variables for an IP formulation of the Contact Map Overlap problem. A contact map is a representation of the 3D folding of a protein as a graph, in which each amino acid corresponds to a vertex, and there is an edge between two amino acids if their distance is smaller than a given threshold when the protein is folded. Given two contact maps, the contact map overlap problem consists in finding a non-crossing matching between the amino acids which identifies the largest common subgraph in the two maps.

Carr, Lancia and Istrail in [2] prove that \(G'\) is a perfect graph, and give an \(O(n^2)\) algorithm for separating the clique inequalities (45). The details of the algorithm are somewhat complex and can be found in [2], while here we will only sketch the main ideas. What is most interesting in the contest of the present paper, however, is the fact that this separation algorithm, as well
as many others in the literature, is based on dynamic programming. Since
dynamic programming can always be thought of as a shortest path problem
on a layered graph, it can be cast as an LP, and hence is suitable for our
technique of compact formulations.

A triangle in B is a set of lines originating from a common point u, on
one side of B (i.e., V₁ or V₂), and ending in a range i, i + 1, ..., j on the
other side. Let a₁ < a₂ < ... < aₖ ∈ V₁ and c₁ > c₂ > ... > cₖ ∈ V₂, with
|h – k| ≤ 1. Connecting a₁ to c₁ to a₂ to c₂..., we obtain a “zigzag” path P in
B (similarly, the path could start from c₁, to a₁, to c₂,...). It is shown in [2]
that the union of the triangles with tip in cᵢ and base aᵢ, aᵢ₊₁, ..., aᵢ₊₁ or tip
in aⱼ and base cⱼ, cⱼ₊₁, ..., cⱼ₋₁ is a clique in B, and all cliques are of this
form. Further, for a maximal clique it is always a₁ = 1, aₖ = n and c₁ = n,
cₖ = 1. Orient each line [i, j] ∈ B, i ∈ V₁ and j ∈ V₂, in both ways, obtaining
the arcs (i, j) and (j, i). Then arc lengths l⁺(i, j) and l⁻(j, i) can be defined
(see [2] for details) as suitable sums of some x-variables, depending on i and
j, so that the length l⁺(a₁, c₁) + l⁻(c₁, a₂) + l⁺(a₂, c₂) + ... of a zigzag path
is equal to the x-value of the clique it individuates. Hence, to find the most
violated clique inequality, we look for the longest zigzag path. Such path can
be found by solving the following dynamic programming relations:

\[
V(i, j, \rightarrow) = \max\{l⁺(i, j) + V(i + 1, j, \leftarrow), V(i, j + 1, \rightarrow)\} \quad (46)
\]

\[
V(i, j, \leftarrow) = \max\{l⁻(j, i) + V(i, j + 1, \rightarrow), V(i + 1, j, \leftarrow)\} \quad (47)
\]

where \(V(i, j, \rightarrow)\) is the length of a longest zigzag path starting at \(i \in V₁\)
and using nodes of \(V₂\) only within \(n – j + 1, n – j, ..., 2, 1\). Similarly, \(V(i, j, \leftarrow)\)
is the length of a longest zigzag path starting at \(n – j + 1 \in V₂\) and using
nodes of \(V₁\) only within \(i, i + 1, ..., n\).

The boundary conditions needed are the values of \(V(n, n, \rightarrow) = l⁺(n, 1)\)
and \(V(n, n, \leftarrow) = l⁻(1, n)\). The recurrence can be solved backwards from
\((n, n)\), in time \(O(n²)\). At the end, the maximum between \(V(1, 1, \rightarrow)\) and
\(V(1, 1, \leftarrow)\) is the length of the longest zigzag path.

Define variables \(y_{ij}^{\rightarrow}\) and \(y_{ij}^{\leftarrow}\) to represent \(V(i, j, \rightarrow)\) and \(V(i, j, \leftarrow)\) respectively. We have constraints

\[
y_{nn}^{\rightarrow} = l⁺(n, 1) \quad (48)
\]

\[
y_{nn}^{\leftarrow} = l⁻(1, n) \quad (49)
\]

\[
y_{ij}^{\rightarrow} \geq l⁺(i, j) + y_{i+1,j}^{\leftarrow} \quad i \in V₁, j \in V₂, i < n \quad (50)
\]

\[
y_{ij}^{\leftarrow} \geq y_{i,j+1}^{\rightarrow} \quad i \in V₁, j \in V₂, i < n \quad (51)
\]
Constraints (54) and (55) force all zigzag paths to have length less than one. Constraints (48) and (49) are the boundary conditions, while (50), (51) and (52), (53), implement (46) and (47) respectively. Together, constraints (48)-(55) can be replaced for the exponentially many clique inequalities (45) for a compact formulation in all LPs which have non-crossing matchings as part of the model.

4 Conclusions

This paper introduces a formal setting, compact optimization and compact separation, for the formulation of polynomial-size LP relaxations of integer programming models with an exponential number of constraints or variables. We have shown by means of some examples how this can be done. Most known exponential IP formulations can be cast this way. However, there is a major exception: we have not been able to formulate a compact separation of the weighted, non-bipartite, matching problem, and in fact we conjecture that no such compact separation is possible. This is unfortunate, since otherwise it is possible that a strong theorem can be proved, analogous to Grötschel, Lovász and Schrijver, that is, optimization is possible if and only if compact separation is possible. In a sense, this would also have provided a simpler proof of Grötschel, Lovász and Schrijver's theorem.

One of the examples presented here, introduces the path formulation of the symmetric TSP. We have shown how its LP relaxation has the same value of the subtour–elimination relaxation. We intend to investigate on a possible tightening of the path formulation which may lead to a better bound.

5 Acknowledgment

Many thanks to Matteo Fischetti and Alberto Caprara for providing us with useful feedback. This work was performed while the second author was visiting Sandia National Labs, on a leave from the University of Padua, Department of Electronics and Informatics.

\[
y_{ij}^+ + y_{i+1,j}^+ - y_{ji}^+ - y_{i+1,j}^+ \leq 1
\]

Conclusions

This paper introduces a formal setting, compact optimization and compact separation, for the formulation of polynomial-size LP relaxations of integer programming models with an exponential number of constraints or variables. We have shown by means of some examples how this can be done. Most known exponential IP formulations can be cast this way. However, there is a major exception: we have not been able to formulate a compact separation of the weighted, non-bipartite, matching problem, and in fact we conjecture that no such compact separation is possible. This is unfortunate, since otherwise it is possible that a strong theorem can be proved, analogous to Grötschel, Lovász and Schrijver, that is, optimization is possible if and only if compact separation is possible. In a sense, this would also have provided a simpler proof of Grötschel, Lovász and Schrijver's theorem.

One of the examples presented here, introduces the path formulation of the symmetric TSP. We have shown how its LP relaxation has the same value of the subtour–elimination relaxation. We intend to investigate on a possible tightening of the path formulation which may lead to a better bound.

5 Acknowledgment

Many thanks to Matteo Fischetti and Alberto Caprara for providing us with useful feedback. This work was performed while the second author was visiting Sandia National Labs, on a leave from the University of Padua, Department of Electronics and Informatics.
References


Distribution

1. External Distribution

1 Prof. Egon Balas
   GSIA, Carnegie Mellon University
   Pittsburgh, PA 15213-3890
1 Dr. Alberto Caprara
   DEIS, Università di Bologna, Viale Risorgimento 2
   40136 Bologna, Italy
1 Prof. Matteo Fischetti
   Dipartimento di Elettronica e Informatica, Via Gradeno 6/a
   35131 Padova, Italy,
1 Prof. R. Ravi
   GSIA, Carnegie Mellon University
   Pittsburgh, PA 15213-3890
1 Dr. Franca Rinaldi
   Dipartimento di Matematica e Informatica, Viale delle Scienze 206
   33100 Udine, Italy
1 Prof. Giorgio Romanin Jacur
   Dipartimento di Elettronica e Informatica, Via Gradeno 6/a
   35131 Padova, Italy,
1 Prof. Paolo Serafini
   Dipartimento di Matematica e Informatica, Viale delle Scienze 206
   33100 Udine, Italy

2. Internal Distribution

10 MS1110 R. Carr, 9211
10 MS1110 G. Lancia, 9214
  1 MS1110 D. Womble, 9214
  1 MS1110 B. Hart, 9211
  1 MS1110 C. Phillips, 9211
  1 9018 Central Technical Files, 8945-1
2 0899 Technical Library, 9616
  1 0612 Review and Approval Desk, 9612 for DOE/OSTI