Cuts over Extended Formulations by Flow Discretization

Eduardo Uchoa

Departamento de Engenharia de Produção,
Universidade Federal Fluminense, Brazil.
uchoa@producao.uff.br

1st July, 2011

Abstract

Large-sized extended formulations have the potential for providing high-quality bounds on some combinatorial optimization problems where the natural formulations perform poorly. This chapter discusses the use of some families of cuts that have been recently applied on extended formulations that are obtained by the discretization of the continuous variables occurring in the natural formulation of the Fixed Charge Network Flow Problem. By interpreting the flows as being link capacities, loads or times, similar cuts can be used on problems like the Capacitated Minimum Spanning Tree, Vehicle Routing and Parallel-Machine Scheduling.

1 Introduction

Many Combinatorial Optimization Problems (COPs) can be cast as follows. We are given a finite set $N = \{1, \ldots, n\}$, costs c_j for each $j \in N$, and a set $\mathcal{F}$ of feasible subsets of N. The problem is to find a set $S \in \mathcal{F}$ that minimizes $\sum_{j \in S} c_j$. Solving such COPs is not a trivial task because the cardinality of $\mathcal{F}$ is far too large to allow an explicit description; those sets are defined by a property satisfied by their elements. Consider, for example, the following problems:

- The Travelling Salesman Problem (TSP) is a COP where N is the edge-set of an undirected graph, c_j are edge costs and $\mathcal{F}$ is associated to the hamiltonian circuits in that graph;
- The Steiner Problem in Graphs (SPG) is a COP where N is the edge-set of an undirected graph, c_j are positive edge costs and $\mathcal{F}$ is associated to the subgraphs connecting a given subset of required vertices in that graph (the minimal such subgraphs are known as Steiner trees).

One of the most effective approaches for handling NP-hard COPs, like the TSP or the SPG, is formulating them as Integer Programs (IPs) and applying branch-and-bound or branch-and-cut algorithms. The "natural way" of formulating such problems is by defining an integer variable x_j for each $j \in N$ and finding a suitable set of constraints to represent $\mathcal{F}$:

$$\text{Min} \quad \sum_{j \in N} c_j x_j \tag{1a}$$

$$\text{S.t.} \quad Ax \geq b \tag{1b}$$

$$x_j \text{ integer} \quad \forall j \in N. \tag{1c}$$

The efficiency of solving those IPs depends crucially on the quality of the chosen formulation. The classical way of improving a formulation is by polyhedral investigation. Define $P \subset R^n$ as the convex hull of the integral solutions of the formulation. One looks for families of inequalities defining facets of P. As the families may contain an exponential number of inequalities, one also needs to devise separation procedures. It is typical that the simpler families of facet-defining inequalities can be separated in polynomial time. More complex families usually define NP-hard separation problems. In those cases, heuristic separation procedures must be devised.

The approach just described had a big success in the solution of the TSP, the most notorious COP. Dantzig, Fulkerson and Johnson [17] outlined the following natural TSP formulation for a graph $G = (V, E)$:

$$\text{Min} \quad \sum_{j \in E} c_e x_e \tag{2a}$$

$$\text{S.t.} \quad \sum_{e \in \delta(v)} x_e = 2 \quad \forall v \in V \tag{2b}$$

$$\sum_{e \in \delta(S)} x_e \geq 2 \quad \forall S \subset V \tag{2c}$$

$$x_e \text{ integer} \quad \forall e \in E. \tag{2d}$$

Constraints (2c) are known as subtour elimination inequalities. Since the seventies, several authors are performing a deep investigation of the TSP polyhedra. Many families of facet-defining inequalities are now known, including 2-matching, comb, path, clique-tree, bipartition, ladder, and domino-parity inequalities [33]. Subtour elimination and 2-matching inequalities can be separated in polynomial time; separation heuristics for the more complex families have been proposed. All that research effort went along with increasingly impressive practical results. In 1978, the largest solved instance had 120 vertices. In 2009, all the TSPLIB instances with up to 85,900 vertices could be solved to optimality by very sophisticated branch-and-cut implementations [4].

It is tempting to view the success on the TSP of the classical approach — formulation over the natural variables + polyhedral investigation + effective separation — as an indication that it may work well on every other COP. Unfortunately, this does not appear to be true. The SPG, another very well-known COP, illustrates this fact. The SPG over a graph $G = (V, E)$ and a required set $T \subseteq V$ has the following natural formulation by Aneja [2]:

$$\text{Min} \quad \sum_{j \in E} c_e x_e \tag{3a}$$

$$\text{S.t.} \quad \sum_{e \in \delta(S)} x_e \geq 1 \quad \forall S \subset V, S \setminus T \neq \emptyset, T \setminus S \neq \emptyset \tag{3b}$$

$$x_e \geq 0 \quad \forall e \in E \tag{3c}$$

$$x_e \text{ integer} \quad \forall e \in E. \tag{3d}$$

Constraints (3b) are cut inequalities. They can be separated in polynomial time. Several other families of facet-defining inequalities were found, including partition, odd-hole, anti-hole and wheel inequalities [12, 13], for which heuristic separation procedures were provided. In spite of those efforts, the performance of branch-and-cut algorithms using those inequalities is not really good. What is the difference between the TSP and the SPG?

- The linear relaxation of the natural TSP formulation (2) is already quite strong in practice. If one also adds the 2-matching inequalities, one can obtain in polynomial time a lower bound that is less than 0.5% away from the optimal on typical TSPLIB instances. This leaves a small gap to be closed by the heuristic separation of complex cuts and, possibly, by branching. The overall branch-and-cut performance can be very good.

- On the other hand, the linear relaxation of the natural SPG formulation (3) is poor. Lower bounds more than 10% away from the optimal are typical on SteinLIB instances. To make things worse, there are no other known constraints that can be separated in polynomial time; even the relatively simpler partition inequalities define an NP-hard separation problem. This leaves a large gap to be closed by the heuristic separation of increasingly complex cuts and by branching.

A better SPG formulation is obtained by using an additional set of variables [51]. Define a directed graph $G_D = (V, A)$ where A contains a pair of opposite arcs (i, j) and (j, i) for each edge $e = \{i, j\} \in E$. Choose a vertex $r \in T$. It can be seen that there is a

one-to-one correspondence between Steiner trees in G and Steiner arborescences (directed trees) in G_D rooted at r. Define a binary variable y_a for each $a \in A$. The formulation is:

$$\text{Min} \quad \sum_{j \in E} c_e x_e \tag{4a}$$

$$\text{S.t.} \quad \sum_{e \in \delta^-(S)} y_a \geq 1 \qquad \forall S \subset V, r \notin S, T \cap S \neq \emptyset \tag{4b}$$

$$y_a \geq 0 \qquad \forall a \in A \tag{4c}$$

$$x_e - y_{ij} - y_{ji} = 0 \quad \forall e = \{i, j\} \in E \tag{4d}$$

$$x_e \text{ integer} \qquad \forall e \in E. \tag{4e}$$

Directed cut inequalities (4b) are much stronger than their undirected counterparts (3b). The gaps of the linear relaxation of formulation (4) are seldom more than 0.1% on SteinLIB instances (except on artificial instances created with the intent of being hard). In fact, the most effective SPG codes of today, which are capable of solving many instances with tenths of thousands of vertices, usually do not bother to separate cuts other than (4b); their algorithmic effort is focused on devising sophisticated graph reductions and dual ascent procedures in order to speedup the solution of that very strong linear relaxation [43, 44]. Formulation (4) is an example of an extended formulation.

Surely, an extended formulation, already having variables other than the natural variables, may be further extended. We define an *extended formulation* with respect to a given *original formulation* — a natural formulation or an already extended one — as follows. Suppose that there are n original variables x and p extended variables y. Suppose there are m constraints, that may involve only x variables, only y variables, or both sets of variables. We also assume that the cost function and the integrality requirements are only on the x variables, which is reasonable, because the original formulation could do that.

$$\text{Min} \quad cx \tag{5a}$$

$$\text{S.t.} \quad Ax + By \geq b \tag{5b}$$

$$x \text{ integer} \tag{5c}$$

An extended formulation can be compared with a formulation on the original variables by projecting its associated polyhedra onto the x space. Let $Q = \{(x, y) \in R^n \times R^p : Ax + By \geq b\}$. The *projection of Q onto x* is defined as $\text{proj}_x(Q) = \{x \in R^n : \exists y \in R^p, Ax + By \geq b\}$. The *projection cone of Q onto x* is defined as $C_x(Q) = \{u \in R^m_+ : uB = 0\}$. For each $u \in C_x(Q)$, it is clear that $(uA)x \geq (ub)$ is a valid inequality in the original space. The following result states that all inequalities that define the polyhedron $\text{proj}_x(Q)$ can be obtained by suitable u multipliers that cancel the y variables:

Theorem 1 *If $Q = \{(x, y) \in R^n \times R^p : Ax + By \geq b\}$, then $proj_x(Q) = \{x \in R^n : (uA)x \geq (ub), \forall u \in C_x(Q)\}$.*

Theorem 1 can be applied in different ways. One can use it "offline" during a polyhedral investigation: giving a suitable set of multiplies in $C_x(Q)$ it is possible to prove that a certain family of inequalities in the x space (already known or not) is implied by the extended formulation. One can also use it "on-the-fly", during an optimization, to separate

violated cuts. Suppose that we want to find a cutting plane $(uA)x \geq (ub)$ separating a given point $\bar{x} \in R^n$ from $\text{proj}_x(Q)$. This can be done by solving the following LP on the u variables:

$$z^* = \text{Min} \quad u(A\bar{x} - b) \tag{6a}$$
$$\text{S.t.} \quad uB = 0 \tag{6b}$$
$$\sum_{i=1}^{m} u_i = 1 \tag{6c}$$
$$u \geq 0. \tag{6d}$$

If $z^* \geq 0$ then $\bar{x} \in \text{proj}_x(Q)$, otherwise a violated cut is obtained. Constraint (6c) is a possible normalization of the u multipliers, to avoid LP unboundedness. This technique is a particular case of the well-known Benders decomposition.

Given the above, we can analyze the relation between undirected and directed SPG formulations in more depth. Let $(V_1, \ldots, V_K)$ be a partition of V such that $V_k \cap T \neq 0$, for $k = 1, \ldots, K$. Define $\delta(V_1, \ldots, V_K)$ as $\{(u, v) \in E : u \in V_i, v \in V_j, i \neq j\}$. The corresponding partition inequality is:

$$\sum_{e \in \delta(V_1, \ldots, V_K)} x_e \geq K - 1. \tag{7}$$

The cut inequalities (3b) correspond to the case $K = 2$. Chopra and Rao [12] showed that partition inequalities define facets of the STP polyhedra under mild conditions. The following projection show that (7) can be obtained from the directed formulation. Without loss of generality, suppose that $r \in V_1$. Define a 0-1 multiplier vector u where the following constraints receive multiplier 1: (i) for $k = 2, \ldots, K$, the constraint of (4b) for $S = V_k$; (ii) for every arc $a \in \delta^+(V_1)$, the corresponding constraint in (4c); (iii) for every edge in $\delta(V_2, \ldots, V_K)$, the corresponding constraint in (4d). Goemans [23] showed that the directed formulation implies not only partition and odd-holes inequalities, but also a whole zoo of previously unknown facet-defining inequalities for the Steiner polytope. Those new families of inequalities are very complex. It is possible to obtain inequalities where coefficients take all integral values between 1 and a chosen odd number. We can use the SPG example to state some facts on the potential benefits of working with extended formulations.

Potential Advantage 1 *It is possible that very complex families of facet-defining inequalities on the natural variables can be obtained by the projection of quite simple inequalities on the extended variables.*

When working with an extended formulation, one can still use the known inequalities on the original variables. However, even when the inequality defines a facet of original polyhedron, it is often obvious that instead of putting zero coefficients on all extended variables, one can lift several such coefficients. For example, the directed cut inequalities (4b) can be viewed as a direct lifting of the undirected cut inequalities (3b).

Potential Advantage 2 *It is possible that an inequality on the natural variables, even when facet-defining, can be immediately lifted when expressed in the extended variables.*

The proof by Grötschel et al. [27] that the separation of partition inequalities is NP-hard looks paradoxical, after it was shown that those inequalities can be obtained by the projection of inequalities that can be separated in polynomial time. The correct interpretation is that cutting only with partition inequalities is hard, but cutting with a larger family of cuts that include the partition inequalities is not. This can be done by applying a Benders decomposition over the directed formulation. Of course, one can just solve the relaxation of the extended formulation and drop the y variables to obtain a point $\bar{x} \in \text{proj}_x(Q)$.

Potential Advantage 3 *It is possible that the separation of a certain family of inequalities in the original variables is NP-hard, and yet, a superfamily of valid inequalities can separated in polynomial time by projection from an extended formulation.*

The directed cut formulation actually formulates a more general problem, the Steiner Problem in Directed Graphs (SPDG), that includes the SPG as a particular case. In fact, a good branch-and-cut algorithm for the SPG over that formulation is also very effective on solving the SPDG.

Potential Advantage 4 *An extended formulation can always be viewed as a natural formulation for a more general problem (defined on the extended variable space) that includes the original COP as a particular case. This only requires allowing additional costs and integrality requirements in the extended variables. The resulting generalized problems may be interesting and also have practical importance.*

On the other hand, there are also disadvantages in working with extended formulations. The increased number of variables is the most evident disadvantage. In the SPG, the directed formulation has only two times more variables than the directed formulation (the x variables can be eliminated). Moreover, there is a one-to-one correspondence between undirected (3b) and directed (4b) cuts and both families are separated by similar algorithms. Therefore, solving the directed relaxation is not much harder than solving the undirected one. The large gains in terms of bound quality are almost for free. Unhappily, this situation is atypical. The most general techniques currently known for obtaining extended formulations (like Sherali-Adams [45] and Lovász-Schrijver [30]) multiply the number of variables and constraints by factors of at least n. Even more particular techniques valid for specific classes of COPS, like the ones that will be presented in Section 2, usually multiply the size of the formulation by large (and non-constant) factors.

Potential Disadvantage 1 *For most COPs, significantly stronger extended formulations are significantly larger.*

There are more subtle disadvantages in large extended formulations. In the case of a compact extended formulation, it would be possible to solve the LP relaxation by a variant of the simplex algorithm. However, such LPs are often highly degenerate and harder than other LPs with similar size. The interior-point LP methods are not much affected by degeneracy, but the lack of hot-start capabilities limits their use in a branch-and-bound algorithm. The typical degeneracy of the LPs from large extended formulations is related to the fact that their solutions (primal and dual) can be very sparse. The fact that most of the variables have the same cost (zero) also contributes to degeneracy.

Potential Disadvantage 2 *The LPs from large extended formulations can be highly degenerate.*

In the case of an extended formulation with an exponential number of constraints, one also has to perform separation. In that case, cut convergence can be a serious issue. Suppose that we want to cut a fractional point $\bar{x}$. Adding a violated inequality $ax \geq b$ in the natural variable space will certainly cut that point and probably will move the objective function. On the other hand, each point $\bar{x}$ is associated with a polyhedron $Q_y(\bar{x}) = \{y \in R^p : By \geq b - A\bar{x}\}$. It is quite possible that a single inequality $ay \geq b$ will not cut all points in that polyhedron. In fact, in large extended spaces, there are frequent situations when one needs several cut iterations to move significantly the objective function. This phenomenon can be explained in an alternative way. It is often the case that a strong or even facet-defining inequality $ax \geq b$ can be obtained from an extended formulation by combining (in the projection operation) several inequalities in the y space. This means that certain fractional solutions that would be cut by $ax \geq b$ will only by cut after a whole set of extended inequalities are added.

Potential Disadvantage 3 *The cut separation in large extended formulations often exhibits convergence problems.*

Slow cut convergence can also occur on natural formulations, this is called "tailing-off". The name suggests that this behavior is typical of the final iterations, when one is already close to the optimal linear solution of the formulation. The proposed remedy is just stop cutting and proceed to branching. However, when working with large extended formulations, the slow cut convergence can manifest much earlier. A premature branching may yield a bound that is not close to the potential of the extended formulation.

2 The Fixed Charge Network Flow Problem: natural and extended formulations

The Fixed Charge Network Flow (FCNF) Problem is an NP-hard generalization of the minimum cost network flow problem where arcs costs are split into two parts: a non-negative fixed cost, incurred if the arc carries any positive amount of flow; and a cost proportional to the flow. A remarkably large set of COPs can be modeled directly as FCNF problems or as FCNF problems with a few simple additional constraints. This includes several network design (for example, the SPG), vehicle routing (for example, the TSP), production planning and scheduling problems.

2.1 Natural formulation

Let the network be defined by graph $G = (V, A)$, positive fixed costs c_a, proportional costs f_a, and capacities u_a for each $a \in A$; and demands d_i for each $i \in V$. A vertex with negative demand is a source vertex; it is required that $\sum_{i \in V} d_i = 0$. The natural formulation for this problem works over continuous flow variables w and binary decision

variables x:

$$\text{Min} \quad \sum_{a \in A} (c_a x_a + f_a w_a) \tag{8a}$$

S.t.

$$\sum_{a \in \delta^-(i)} w_a - \sum_{a \in \delta^+(i)} w_a = d_i \quad \forall\, i \in V \tag{8b}$$

$$0 \leq w_a \leq u_a x_a \qquad \forall\, a \in A \tag{8c}$$

$$x_a \in \{0, 1\} \qquad \forall\, a \in A. \tag{8d}$$

The linear relaxation of (8) can yield poor bounds because of the weak coupling between the x and w variables by constraints (8c). In fact, x_a will always be equal to w_a/u_a in that relaxation. The polyhedra corresponding to formulation (8) has been extensively studied; several families of valid mixed-integer inequalities are known [41].

2.2 The Multi-Commodity Extended Reformulation

Another standard way of strengthening the FCNF formulation is by decomposing the flows according to their destinations [41]. Suppose w.l.o.g. that there is a single source vertex $s \in V$ (if this is not the case, a supervertex concentrating all sources is created). Let $D \subset V$ be the vertices with positive demand. Define one commodity for each $d \in D$, with corresponding variables w_a^d. Define d_i^d as being equal to d_i if $i = d$ and equal to 0 otherwise. The extended formulation follows:

$$\text{Min} \quad \sum_{a \in A} (c_a x_a + f_a \sum_{d \in D} w_a^d) \tag{9a}$$

S.t.

$$\sum_{a \in \delta^-(i)} w_a^d - \sum_{a \in \delta^+(i)} w_a^d = d_i^d \quad \forall\, i \in V, i \neq s;\ \forall\, d \in D \tag{9b}$$

$$0 \leq w_a^d \leq \min\{u_a, d_d\} x_a \quad \forall\, a \in A;\ \forall\, d \in D \tag{9c}$$

$$x_a \in \{0, 1\} \qquad \forall\, a \in A. \tag{9d}$$

The multicommodity reformulation of the FCNF (9), at the expense of having $O(|D|)$ times more variables and constraints, can be significantly stronger because constraints (9c) are a tighter link between the x and w variables.

As an example, the SPG can be formulated over a directed graph $G = (V, A)$ as a FCNF as follows. The root terminal is a source with demand $1 - |T|$, the other terminals have unitary demand; the arc capacities are set to $|T| - 1$; the fixed costs are the arc costs, there are no flow costs. While the single flow formulation has a very weak linear relaxation, its multicommodity reformulation is very strong [15, 51], being equivalent (by projection) to the dicut formulation (4).

A possible way of dealing with the large size of a multicommodity reformulation is by Benders decomposition [16].

2.3 The Discretized-Flow Extended Reformulation

Suppose a FCNF where all arc capacities and all vertex demands are integral. Then, there exists an optimal solution where all w variables are integral. In those cases, one

may reformulate the problem using only binary variables x_a^d meaning that a carries a flow of d units. The formulation follows:

$$\text{Min} \quad \sum_{a \in A} \sum_{d=1}^{u_a} (c_a + df_a) x_a^d \tag{10a}$$

S.t.

$$\sum_{a \in \delta^-(i)} \sum_{d=1}^{u_a} dx_a^d - \sum_{a \in \delta^+(i)} \sum_{d=1}^{u_a} dx_a^d = d_i \quad \forall\, i \in V \tag{10b}$$

$$\sum_{d=1}^{u_a} x_a^d \leq 1 \qquad \forall\, a \in A \tag{10c}$$

$$x_a^d \in \{0,1\} \qquad \forall\, a \in A;\ d = 1, \ldots, u_a. \tag{10d}$$

Equations (10b) give the flow-balance over the discretized variables. Let $C = \max\{u_a : a \in A\}$. To provide a simpler notation of this formulation, we define a directed multigraph $G_C = (V, A_C)$, where A_C contains arcs $(i,j)^d$, for each $(i,j) \in A$, $d = 1, \ldots, u_a$. When working with variables x_a^d it is assumed that $\delta^-(\cdot)$ and $\delta^+(\cdot)$ are subsets of A_C. For example, equations (10b) can be written as:

$$\sum_{a^d \in \delta^-(i)} dx_a^d - \sum_{a^d \in \delta^+(i)} dx_a^d = d_i \quad \forall\, i \in V.$$

It can be shown that the linear relaxation of formulations (8) and (10) provide the same bound. Since the discretized formulation can have much more variables, an increase by a pseudo-polynomial factor of $O(C)$, at first sight there is no advantage at all in it. However, the large number of variables on the new reformulation (and the fact that all continuous variables are eliminated) allows the derivation of stronger families of valid inequalities.

2.3.1 Extended Capacity Cuts

The sum of the flow-balance equations (10b) corresponding to the vertices in a set $S \in V$ is:

$$\sum_{a^d \in \delta^-(S)} dx_a^d - \sum_{a^d \in \delta^+(S)} dx_a^d = d(S), \tag{11}$$

where $d(S) = \sum_{i \in S} d_i$. The *Extended Capacity Cuts (ECCs)* for a given set S are defined as the inequalities that are valid to the polyhedron $P(S)$ defined by the 0-1 solutions of (11) [48]. Those equations are a rich source of strong cuts. We define the *Homogeneous Extended Capacity Cuts (HECCs)* as the subset of the ECCs where all entering variables with the same flow value have the same coefficients, the same happening with the leaving variables. For a given set S, define aggregated variables y^d and z^d as follows:

$$y^d = \sum_{a^{d'} \in \delta^-(S) \,:\, d' = d} x_a^d \qquad \forall\, d = 1, \ldots, C \tag{12}$$

$$z^d = \sum_{a^{d'} \in \delta^+(S) \,:\, d' = d} x_a^d \qquad \forall\, d = 1, \ldots, C. \tag{13}$$

The flow-balance equation over those variables is:

$$\sum_{d=1}^{C} dy^d - \sum_{d=1}^{C} dz^d = d(S). \tag{14}$$

For each possible pair of values of C and $D = d(S)$, we may define the polyhedron $P(C, D)$ induced by the non-negative integral solutions of (14). The inequalities that are valid for those polyhedra are HECCs. Suppose that we have a heuristic that provides candidate sets S. We describe three approaches to separate HECCs over those sets:

1. For small values of C, say, up to 10, we can actually compute the facets of $P(C, D)$, for different values of D, and store them in tables for posterior separation. For a given set S, the separation procedure must only check if one of those facets is violated.

2. Recently, Dash, Fukasawa and Günlük [18] performed a deep study of polyhedra $P(C, D)$, which they called the *Master Equality Polyhedra*. In particular, they give a pseudo-polynomial characterization of the polar of such polyhedra. This means that one can separate a point from $P(C, D)$ by solving a linear program of pseudo-polynomial size ($O(C)$ variables and $O(C^3)$ constraints). For each candidate set S, one LP must be solved. The approach is practical for moderate values of C.

3. For larger values of C, one can separate weaker cuts (not guaranteed to define facets of $P(C, D)$ by rounding: relax the equation (14) corresponding to S to $\geq$, multiply all coefficients by a rational constant $r = a/b$, apply integer rounding and check if the resulting inequality is violated. It was proved in [49] that there are at most $O(C^2)$ multiplies r that may yield distict *Rounded Homogeneous Extended Capacity Cuts*.

As will be seen in the following sections, the ECCs can be effectively used on quite different COPs related to the FCNF, as long as they are formulated over similar discretized variables. In fact, it can proved that many known inequalities for those problems are equivalent to (or dominated by) particular ECCs. For example, consider the SPG formulation as a discretized FCNF. In that case $C = |T| - 1$. Let S be a set that contains at least one terminal vertex but not the root, so $D = d(S) \geq 1$. By applying the rounding procedure with multiplier $r = 1/C$ over the corresponding equality (14), all coefficients corresponding to $\delta^-(S)$ are rounded to one. By applying the identities $x_a = \sum_{d=1}^{u_a} x_a^d$, for $a \in A$, the resulting inequality projects into the directed cut inequality (4b) over S.

Of course, taking the particular structure of the COPs into account, additional new families of cuts on those variables can be derived.

3 The CMST over the Capacity-Indexed Formulation

Let $G = (V, E)$ be an undirected graph with vertices $V = \{0, 1, \ldots, n\}$ and $m = |E|$ edges. Vertex 0 is the *root*. Each remaining vertex i is associated with a positive integer demand d_i. When all such demands are equal to 1, the instance is said to be unitary. Root demand d_0 is defined as zero. Each edge $e \in E$ has a nonnegative cost c_e. Given a positive integer C greater than or equal to the maximum demand, the Capacitated Minimum Spanning Tree (CMST) problem consists of finding a minimum cost spanning tree for G such that the total demand of the vertices in each subtree hanging from the

root does not exceed C. An interesting generalization of the CMST is the Multi-Level Capacitated Minimum Spanning Tree (MLCMST) problem [22] where the edge costs are given by c_e^d, where d, $1 \leq d \leq C$, is the total demand of the subtree hanging from edge e. The MLCMST problem models the situation where edges are communication links and the chosen technology for each link depends on the total traffic that will use that link.

The CMST can be formulated as a FCNF with additional degree constraints. This section will summarize the experience with cuts over a capacity-indexed formulation for both the CMST [49] and the MLCMST [50].

3.1 Formulations and valid inequalities

3.1.1 Arc formulation

Although the CMST is defined on an undirected graph, as in most network design problems where this is possible, it is much preferable to use a directed formulation. Define a directed graph $G_D = (V, A)$, where A has a pair of opposite arcs (i, j) and (j, i) for each edge $e = \{i, j\} \in E$, excepting edges $\{0, i\}$ adjacent to the root, which are transformed into a single arc $(0, i)$. The arc costs correspond to the original edge costs. Now, in graph G_D, one looks for a minimum cost capacitated spanning arborescence directed from the root to each other vertex. Such an arborescence corresponds to a minimum cost capacitated spanning tree in the original graph G.

The Arc Formulation uses binary variables x_a to indicate whether arc a belongs to the optimal solution. This is the original formulation (already an extended formulation) that will be compared with the Capacity-Indexed Formulation. Denote the set of non-root vertices by $V_+ = \{1, \ldots, n\}$. For any set $S \subseteq V$, we let $d(S) = \sum_{i \in S} d_i$, $k(S) = \lceil d(S)/C \rceil$. The formulation follows [28]:

$$\text{Minimize} \qquad \sum_{a \in A} c_a x_a \tag{15a}$$

$$\text{S.t.}$$

$$\sum_{\delta^-(i))} x_a = 1 \qquad \forall\, i \in V_+ \tag{15b}$$

$$\sum_{(\delta^-(S))} x_a \geq k(S) \quad \forall\, S \subseteq V_+ \tag{15c}$$

$$x_a \in \{0, 1\} \qquad \forall\, a \in A. \tag{15d}$$

The *In-Degree* constraints (15b) state that exactly one arc must enter each non-root vertex. *Capacity Cuts* (15c) state that at least $k(S)$ arcs must enter each set S. Another useful family of constraints are the *Root Cutset inequalities*. Define $S_\alpha = \{i \in V \setminus S : k(S \cup \{i\}) = k(S)\}$ and $S_\beta = (V \setminus S) \setminus S_\alpha$. Note that the root always belongs to S_α. The Root Cutset inequalities are

$$\tfrac{k(S)+1}{k(S)} \sum_{\delta^-(S) \cap \delta^+(S_\alpha)} x_a + \sum_{\delta^-(S) \cap \delta^+(S_\beta)} x_a \geq k(S) + 1 \quad \forall\, S \subseteq V_+. \tag{16}$$

Constraints (16) are actually a strengthening of the Capacity Cuts, based on the observation that if one of the subtrees covering S comes from a higher demand vertex in S_β, at least $k(S) + 1$ subtrees must enter S. Other families of valid inequalities that can

potentially improve the arc formulation are known, including several variants of the so-called *Multistar* constraints [5, 28, 25, 26]. Even with all such inequalities, branch-and-cut algorithms over the arc formulation fail on many instances with only 50 vertices.

3.1.2 The Capacity-Indexed Formulation

Gouveia [24] presented the following capacity-indexed formulation for the CMST (also valid for the MLCMST), inspired by an earlier formulation for the TSP [20]. Let binary variables x_a^d indicate that arc $a = (i, j)$ belongs to the optimal arborescence and that the total demand of all vertices in the sub-arborescence rooted in j is exactly d. In other words, x_a^d equal to 1 means that a link of capacity d needs to be installed over arc a. Note that variables x_{ij}^d with $d > C - d(i)$ can be removed. We define a directed multigraph $G_C = (V, A_C)$, where A_C contains arcs $(i, j)^d$, for each $(i, j) \in A$, $d = 1, \ldots, C - d(i)$.

$$\text{Minimize} \qquad \sum_{a^d \in A_C} c_a^d x_a^d \tag{17a}$$

S.t.

$$\sum_{a^d \in \delta^-(i)} x_a^d = 1 \qquad \forall\, i \in V_+ \tag{17b}$$

$$\sum_{a^d \in \delta^-(i)} d x_a^d - \sum_{a^d \in \delta^+(i)} d x_a^d = d_i \quad \forall\, i \in V_+ \tag{17c}$$

$$x_a^d \in \{0, 1\} \qquad \forall\, a^d \in A. \tag{17d}$$

Formulation (17) can also be viewed as a discretized FCNF formulation (like (10)) with additional in-degree constraints (17b). This formulation has only $2n$ constraints, but $O(mC)$ variables. The potential advantage of the capacity-indexed formulation is to allow the derivation and separation of new families of cuts defined over this pseudo-polynomially large extended variable space.

3.1.3 Extended Capacity Cuts

The ECCs are very effective cuts for Formulation (17). It can be proved that most known CMST inequalities for the arc formulation are equivalent to (or dominated by) particular cases of ECCs. For example, it can be shown that a Root Cutset (16) over a set S is dominated by the following ECC:

$$\frac{k(S)+1}{k(S)} \sum_{a^d \in \delta^-(S)\,:\, d > d^*} x_a^d + \sum_{a^d \in \delta^-(S)\,:\, d \le d^*} x_a^d \ge k(S) + 1, \tag{18}$$

where $d^* = d(S) - C(k(S) - 1) - 1$.

The ECCs also include several previously unknown inequalities. For example, consider an instance with $C = 5$ and set S with $d(S) = 6$. The polyhedron corresponding to the aggregated balance-equation (14) is:

$$P(5,6) = \left\{ \begin{array}{rrrrrrrrrcl} y^1 & +2y^2 & +3y^3 & +4y^4 & +5y^5 & -z^1 & -2z^2 & -3z^3 & -4z^4 & = & 6, \\ y^1 & +y^2 & +y^3 & +y^4 & +y^5 & & & & & \ge & 2 \\ y^1 & +2y^2 & +2y^3 & +3y^4 & +3y^5 & & & -z^3 & -2z^4 & \ge & 4 \\ 2y^1 & +2y^2 & +3y^3 & +4y^4 & +4y^5 & & -z^2 & -2z^3 & -2z^4 & \ge & 6 \\ & & & & & & & (y,z) & & \ge & 0 \end{array} \right\}.$$

It can be seen that one of the three non-trivial facets of $P(5,6)$ defines a Capacity Cut, the other two non-trivial facets define new cuts.

3.1.4 Fenchel cuts over small neighborhoods

The so-called *Fenchel cuts* for integer programming were introduced by Boyd in the early nineties [11]. These cuts are characterized for being separated by solving a linear program where the variables correspond to the coefficients of the desired cut, maximizing the violation with respect to the current fractional solution subject to not cutting any integer feasible solution. Of course, Fenchel cut separation can not be applied to a whole IP, separating a single cut would be more expensive than solving the original problem. In practice, those cuts are separated with respect to substructures present in the IP, typically knapsack-like constraints [10]. Applegate et al. [3] separated Fenchel cuts for the TSP by performing node contractions that shrink the original graph into a much smaller graph and by considering the solutions of the graphical TSP (a relaxation of the TSP that allows multiple visits to a node) on the shrunk graph. They were called *local cuts* (in spite of the fact that, after unshrinking, the resulting cut may have non-zero coefficients spread over the original graph). We propose separating Fenchel cuts for the CMST by considering small parts of a fractional solution on the extended capacity-indexed space.

For a set $S \subset V_+$, define the neighborhood of S as the arc-set $N(S) = \delta^-(S) \cup \delta^+(S) \cup A(S)$, where $A(S)$ is the set of arcs with both endpoints in S. Define $x(S)$ as the subset of the variables x_a^d where $a \in N(S)$. Let $Q(S)$ be the set composed by the 0-1 incidence vectors that correspond to possible integral values for the variables in $x(S)$ and are maximal with respect to the number of values 1. If $\bar{x}$ is the current fractional solution in the branch-and-cut, we denote by $\bar{x}(S)$ its restriction to $N(S)$. In a similar way, let α be a vector of coefficients associated to the x variables and $\alpha(S)$ its restriction to $N(S)$. If the solution of the following linear program over variables $\alpha(S)$ yields $z^* > 1$, then $\alpha.x \leq 1$ (the positions of α not in $\alpha(S)$ are completed with zero) is a valid violated cut.

$$\begin{aligned} \text{Maximize } z = \quad & \bar{x}(S).\alpha(S) && \text{(19a)} \\ \text{S.t.} \quad & \\ & q.\alpha(S) \leq 1 \quad (\forall\, q \in Q(S)), && \text{(19b)} \\ & \alpha \geq 0 && \text{(19c)} \end{aligned}$$

The separation of such kinds of Fenchel cuts may be practical for sets S of small cardinality. One can further restrict $N(S)$ to the arcs with positive value in the current fractional solution, reducing the number of solutions in $Q(S)$ to be considered. Moreover, only sets S that are connected with respect to $\bar{x}$ need to be considered.

For example, Figure 1 depicts the arcs with positive value in a fractional solution restricted to $N(S)$, where $S = \{1, 2\}$. All those arcs have a value of 1/3, the numbers next to the arrows are the capacity indices. The instance has unitary demands and $C = 3$.

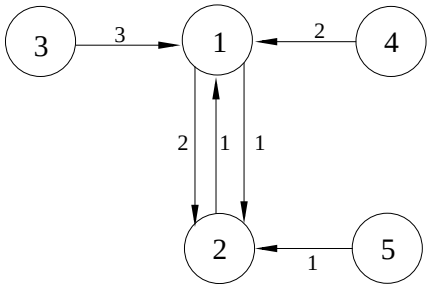

Figure 1: Partial fractional solution over $N(\{1,2\})$, all depicted arcs have value 1/3.

We set the following LP:

$$\begin{aligned}
\text{Maximize } z = \quad & 1/3(\alpha^1_{12} + \alpha^2_{12} + \alpha^1_{21} + \alpha^3_{31} + \alpha^2_{41} + \alpha^1_{52}) && \text{(20a)}\\
\text{S.t.} \quad & \\
& \alpha^1_{12} + \alpha^3_{31} \leq 1 && \text{(20b)}\\
& \alpha^1_{12} + \alpha^2_{41} \leq 1 && \text{(20c)}\\
& \alpha^2_{12} + \alpha^3_{31} \leq 1 && \text{(20d)}\\
& \alpha^3_{31} + \alpha^1_{52} \leq 1 && \text{(20e)}\\
& \alpha^2_{41} + \alpha^1_{52} \leq 1 && \text{(20f)}\\
& \alpha^1_{21} \leq 1 && \text{(20g)}\\
& 0 \leq \alpha. && \text{(20h)}
\end{aligned}$$

Solving that LP, one obtains $z^* = 4/3$ and the following violated inequality:

$$x^1_{12} + x^2_{12} + x^1_{21} + x^1_{52} \leq 1$$

In this particular case, the derived Fenchel cut has a clear structure (a clique cut). In fact, it could be lifted to a stronger cut by including positive coefficients for some variables in $N(S)$ with $\bar{x}^d_a = 0$, that were not considered in the separation. For example, x^2_{21} (and several other variables) could have their coefficients increased to 1 by noticing that this variable is incompatible with all the variables already with coefficient 1. In practice, it is often not practical to perform lifting. The difficulty of lifting lies in the fact that general Fenchel cuts do not have a clear structure. For example, Figure 2 depicts the support graph of a more complex cut found over a certain fractional solution of a benchmark CMST instance. Again, the numbers next to the arrows represent the d indices. The cut is:

$$3x^1_{12} + 3x^2_{12} + x^1_{13} + x^3_{13} + x^1_{21} + x^1_{23} + x^3_{23} + x^3_{25} + x^1_{41} + 2x^3_{62} \leq 4.$$

3.2 Experimental Results

Handling a CMST formulation with $O(mC)$ variables, when C can be large, is certainly an issue. One possibility is performing column generation. When combined with cut separation, it leads to a branch-cut-and-price algorithm. Instead of performing column

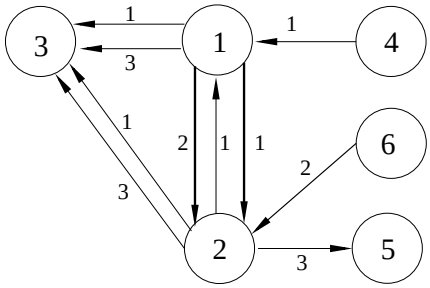

Figure 2: Support graph of a more complex Fenchel cut.

generation over individual variables x_a^d, in [49] it was decided to perform column generation over a combinatorial structure that correspond to sets of x_a^d variables, in order to better capture the structure of the problem. A *q-arb* is an arborescence-like structure on G_C, having degree 1 at the root and with total demand at most C, but allowing some vertices to appear more than once. More precisely,

Definition 1 *A* q-arb rooted at a vertex $i \neq 0$ *can be:*

- *The vertex* i *alone. In this case, the q-arb demand is* d_i*.*
- *The vertex* i *connected to* k *other q-arbs with demand* D_k *rooted at distinct vertices* $v_1, \ldots, v_k$ *by arcs* $(i, v_k)^{D_k} \in A$*. The demand of this q-arb is* d_i *plus the demand of its* k *sub-q-arbs and must not exceed* C*.*

Finally, a q-arb rooted at *0, or just a* q-arb*, is a q-arb rooted at a vertex* $i \neq 0$ *with demand* D *plus the arc* $(0, i)^D$*.*

Number all possible q-arbs from 1 to p. Associate a non-negative variable λ_j with the j-th q-arb. Define q_a^{jd} as 1 if arc a^d appears in j-th q-arb and 0 otherwise. An alternative formulation with an exponential number of variables is:

$$\text{Minimize} \quad \sum_{a^d \in A_C} c_a^d x_a^d \tag{21a}$$

S.t.

$$\sum_{j=1}^{p} q_a^{jd} \lambda_j - x_a^d = 0 \quad \forall\, a^d \in A_C \tag{21b}$$

$$\sum_{a^d \in \delta^-(i)} x_a^d = 1 \qquad \forall\, i \in V_+ \tag{21c}$$

$$\lambda_j \geq 0 \qquad j = 1, \ldots, p \tag{21d}$$

$$x_a^d \in \{0, 1\} \qquad \forall\, a^d \in A_C. \tag{21e}$$

It can be shown that equalities (17c) are implicitly given by (21b). Other (unknown) inequalities are also implied by (21b), leading to a stronger formulation. By relaxing the integrality constraints and eliminating the x variables, the so-called Dantzig-Wolfe Master

(DWM) Problem is obtained:

$$\text{Minimize} \quad \sum_{j=1}^{p}\Big(\sum_{a^d \in A_C} q_a^{dj} c_a^d\Big)\,\lambda_j \tag{22a}$$

$$\text{S.t.}$$

$$\sum_{j=1}^{p}\Big(\sum_{a^d \in \delta^-(i)} q_a^{dj}\Big)\,\lambda_j = 1 \quad \forall\, i \in V_+ \tag{22b}$$

$$\lambda_j \geq 0 \qquad\qquad j = 1, \ldots, p. \tag{22c}$$

The pricing of q-arbs can be performed in $O(mC^2)$, which can be expensive for large values of C. Therefore, it is important to also have fast pricing heuristics. A generic cut $\sum_{a^d \in A_C} \alpha_a^d x_a^d \geq b$ can be introduced in the DWM as $\sum_{j=1}^{p}(\sum_{a^d \in A_C} \alpha_a^d q_a^{dj})\lambda_j \geq b$.

Table 1 (adapted from [49]) presents a detailed comparison of lower bounding methods over a sample of 30 CMST instances from the ORLIB. The te80 instances have $n = 80$, unitary demands and $C = 5$, 10 or 20, there are 5 instances per combination. The cm50 instances have $n = 50$, non-unitary demands and $C = 200$, 400 or 800, there are also 5 instances per combination. Columns **Arc** correspond to the linear relaxation of the arc formulation (15), separating Root Cutset inequalities (16), which are slightly stronger than Capacity Cuts (15c). Columns **DWM** corresponds to solution of the DWM (22). Columns **+ RCs** correspond to the introduction of Root Cutset cuts in the DWM. Finally, Columns **+ ECCs** correspond to also separating ECCs. For each method, the average percent gap with respect to the optimal solution and the average time in seconds (on a Pentium IV 3.0GHz processor) are provided. Some comments about those results:

Table 1: Gaps and times on CMST instances.

	Arc		DWM		+ RCs		+ ECCs	
Instances	Gap	T	Gap	T	Gap	T	Gap	T
te80-5	5.86	70.1	1.09	1.77	0.33	6.46	**0.16**	20.9
te80-10	4.04	19.1	2.24	7.30	0.86	25.3	**0.36**	223.8
te80-20	1.19	2.10	4.61	31.5	0.94	199.4	**0.53**	638.8
cm50-200	5.10	3.01	3.48	7.38	0.82	31.9	**0.22**	115.8
cm50-400	3.02	0.74	7.60	74.9	1.49	320.9	**0.36**	2251
cm50-800	0.73	0.32	13.21	817	0.62	6679	**0.20**	38715

- Comparing columns **Arc** and **+ RCs**, it can be seen that the additional structure provided by the q-arbs makes the arc formulation significantly stronger, specially when the instances are tight, i.e., the value of C allow few vertices in each subtree. However, when C is large, the expensive pricing of q-arbs makes the method very time consuming. In fact, instances with $C = 800$ are much better solved by a branch-and-cut over the Arc Formulation.

- Comparing columns **+RCs** and **+ ECCs**, it can be seen that the separation of ECCs can obtain even smaller gaps. Several previously open instances from the literature could only be solved after the addition of those cuts.

The branch-and-cut in the article [50] avoids the expensive pricing of q-arbs by working directly over Formulation (17), which is only practical for relatively small values of C (say, up to 30). In order to compensate for the lack of the q-arb structure, ECCs were separated in a more aggressive way and Fenchel cuts over neighborhoods of size two were also added. The small gaps obtained allowed solving to optimality most MLCMST instances from the literature with up to 150 vertices. In this chapter, we report results obtained by the same algorithm over CMST instances with unitary demands. Section 1 mentions that cut separation over large extended formulations is prone to convergence problems (Potential Disadvantage 3). Those problems do happen in this case. In fact, the cutting procedure never converges in reasonable times and has to be stopped prematurely by a tailing-off criterion. Columns **1.0/10 iter** report gaps and times (in seconds, on a single core of an i5 M430@2.27GHz processor) by stopping the cutting when the last 10 cut rounds fail to improve the lower bound by at least 1.0 units. Columns **0.1/10 iter** and **0.04/10 iter** correspond to stopping when the last 10 cut rounds could not improve the bound by 0.1 and 0.04 units, respectively.

Table 2: Gaps and times on unitary CMST instances, Formulation (17) separating ECCs and Fenchel Cuts.

	1.0/10 iter		0.1/10 iter		0.04/10 iter	
Instances	Gap	T	Gap	T	Gap	T
te80-5	0.10	48.1	0.09	82.3	**0.07**	294.8
te80-10	0.30	451.7	0.23	1856.3	**0.22**	5188.0
te80-20	0.27	285.4	0.23	748.7	**0.18**	4628.8
All te80	0.22	261.7	0.18	895.8	**0.15**	3370.5

Some comments about the results in Table 2:

- A strong separation on the capacity-indexed space can more than compensate for the lack of q-arbs. The gaps for the te80 instances in Table 2 are smaller than those in Table 1.

- The larger times are directly related to increased difficulties with cut convergence. For solving relatively easy instances it is better to stop earlier, for example, using the 1 unit minimum improvement per 10 iterations criterion, and proceed to branching. However, when solving hard instances to optimality, spending a long time in the root node in order to get a very good lower bound pays.

- The comparison with the results obtained by the Arc Formulation is striking. The average gap of that original formulation over all the 15 te80 instances is 3.70%. The average gap of the capacity-indexed formulation, after adding cuts that can only be expressed over that extended space, can be much smaller.

4 Other Applications

4.1 The VRP over the Load-Indexed Formulation

Let $G = (V, A)$ be a directed graph with vertices $V = \{0, 1, \ldots, n\}$ and $m = |A|$ arcs. Vertex 0 is the *depot*. Each remaining vertex i is a *customer*, associated with a positive integer demand d_i. Depot demand d_0 is defined as zero. Each arc $a \in A$ has a nonnegative travel cost c_a. Given a positive integer C, the *Asymmetric Capacitated Vehicle Routing Problem* (ACVRP) problem consists of finding a set of routes satisfying the following constraints: (i) each route starts and ends at the depot, (ii) each customer is visited by a single vehicle, and (iii) the sum of the demands of the customers in any route is at most C. The goal is to design a set of routes in such a way that the sum of travel costs is minimized. The classical symmetric CVRP is a particular case of the ACVRP.

The ACVRP admits a discretized FCNF based formulation very similar to Formulation (17), by including outdegree constraints (equal to 1) and additional arcs $(i, 0)^0$ for every $i \in V_+$. This formulation corresponds to an interpretation where a vehicle leaves the depot with a certain load, makes deliveries in each customer visited, and returns to the depot empty. Variables x_{ij}^d indicate that the vehicle traverses arc (i, j) with a load of d units. However, by disaggregating equations (17c), a stronger formulation over those variables is obtained:

$$\text{Minimize} \quad \sum_{a^d \in A_C} c_a x_a^d \tag{23a}$$

$$\text{S.t.}$$

$$\sum_{a^d \in \delta^-(i)} x_a^d = 1 \qquad \forall\, i \in V_+ \tag{23b}$$

$$\sum_{a^d \in \delta^-(i)} x_a^d - \sum_{a^{d-d_i} \in \delta^+(i)} x_a^{d-d_i} = 0 \quad \forall\, i \in V_+;\ \forall d = d_i, \ldots, C \tag{23c}$$

$$x_a^d \in \{0, 1\} \qquad \forall\, a^d \in A. \tag{23d}$$

It is worthy mentioning that the load-indexed formulation (23) is a natural formulation for a generalized problem, where travel costs in each arc can depend on the vehicle load. There are very few articles in the VRP literature addressing that issue [9], in spite of the fact that the fuel consumption rate of a typical truck varies by a factor of almost two, depending on its load.

A q-route [14] is a structure that can be defined on graph G_C as follows: a walk starting and ending at the depot vertex 0, such that an arc $(j, i)^d$ entering a customer vertex i must be followed by an arc $(i, k)^{d-d_i}$ leaving i. The q-routes are a relaxation of the actual routes because they allow customer vertices to be visited more than once. Number all possible q-routes from 1 to p. Associate a non-negative variable λ_j with the j-th q-route. Define q_a^{jd} as 1 if arc a^d appears in j-th q-route and 0 otherwise. With a new interpretation (q-routes instead of q-arbs), Formulation (21) is valid for the VRP and equivalent to Formulation (23). The pricing of q-routes can be performed in $O(mC)$, which is reasonable even for large values of C. Moreover, significantly stronger formulations are obtained by working with q-routes without s-cycles, where vertices in the walk may only repeat after $s + 1$ arcs. Those q-routes without s-cycles can be priced in $O(s!s^2mC)$ [29], which can still be reasonable for small values of s.

4.1.1 Capacitated VRP

A number of authors investigated the natural edge formulation for the CVRP and constructed sophisticated branch-and-cut algorithms [34]. However, some instances with only 50 vertices remained open until 2003. The branch-cut-and-price presented in [21] obtained significantly better results by pricing q-routes without 3-cycles and separating the same inequalities (in the edge space) used in [31]; all instances from the literature with up to 135 vertices were solved to optimality. After that, Baldacci et al. [7, 6] presented a better algorithm, that obtains stronger lower bounds by pricing elementary (without any subcycles) routes and separating cuts over the λ variables of the DWM. Instead of branching, routes with reduced cost smaller than the duality gap are enumerated to construct a set-partitioning problem that is given to a MIP solver. However, in spite of solving all the instances already solved in [21], usually in much less time, the new algorithm could not solve any of the larger open instances. A possible explanation is that this new algorithm is based on very clever implementations of algorithmic components that have an exponential worst-case complexity (according to the definition proposed in [42], cutting over the DWM variables is non-robust). On the instances with up to 135 vertices and relatively small routes (having less than 12 customers on average) that complexity is tamed and the overall algorithm is very efficient. In particular, because the lower bounds are very strong, few routes are enumerated. On the other hand, on instances with more than 150 vertices and long routes (more than 12 customers on average) the exponential complexity begins to manifest.

This means that there exists an interest of improving the branch-cut-and-price in [21] by also separating cuts over the load-indexed formulation. In principle, as the main components of such algorithm (those that can not be replaced by an heuristic without compromising the overall optimality) would have a pseudo-polynomial complexity (they depend on C), its performance may scale better on larger instances. Some results obtained by separating ECCs and Triangle Clique Cuts (also described in [38]) appear in [37]. Improved results, by also separating a new generalization of the Capacity Cuts in the extended space, were presented in [35] and are repeated in Table 3. The first row reports average gaps over a set of 14 medium-sized representative instances from the literature, having from 50 to 101 vertices. The second row reports average gaps over 3 larger instances: M-n151-k12, M-n200-k16, and M-n200-k17. Those instances are still open, the gaps are calculated over the best known solutions. The second and third columns correspond to the root node of the algorithms in [31] and [21], respectively. Column **ER** correspond to taking that branch-cut-an-price and replacing the pricing of q-routes for an exponential pricing of elementary routes, still separating only the cuts in the edge space. Column **ExtC** corresponds to keeping the pseudo-polynomial pricing of q-routes without 3-cycles, but separating the additional cuts in the extended space. Last column corresponds to the lower bound in [7]. Some comments about the results in Table 3:

- On medium-sized instances, it is interesting to see that adding cuts in the extended space can be more effective than performing an exponential-complexity pricing of elementary routes. On those instances, the gaps in [7], due to the cutting over the DWM variables, are even smaller.

- On large-sized instances, the gaps by the last two methods are almost identical. Un-

Table 3: Gaps over CVRP instances.

Instances	[31]	[21]	ER	ExtC	[7]
medium	3.02	1.28	1.07	0.81	0.54
large	-	1.73	-	1.44	1.44

happily, they are still not small enough to allow solving the instances to optimality in reasonable times.

- As happened in the CMST, the separation of cuts in the extended space is plagued by convergence problems and has to be stopped by a tailing-off criterion.

4.1.2 Heterogeneous Fleet VRP

The Heterogeneous Fleet Vehicle Routing Problem (HFVRP) is a generalization of the CVRP. Instead of assuming that all vehicles are identical, there is an availability of several vehicle types, with different characteristics. This generalization is very important to the operations research practice, since most actual vehicle routing applications deal with heterogeneous fleets.

The HFVRP was considered to be much harder than the CVRP; instances with only 20 customers were not solved to optimality until 2007. In particular, it is not easy to derive strong inequalities in the natural variables analogous to the Capacity Cuts, since the customers in a set S may be visited by vehicles with distinct capacities (see [52]). A branch-cut-and-price over the load-indexed extended formulation represented a breakthrough, solving instances with up to 75 customers [36, 38]. The separation of ECCs and Triangle Clique Cuts is crucial to the performance of the algorithm. For example, in the FIX variant (where the vehicles only differ by their capacities and fixed costs) those cuts reduce the average gap from 1.60% to 0.44%. The effectiveness of the ECCs in this problem is due to the fact that a variable x_a^d, where $a \in \delta^-(S)$ or $a \in \delta^+(S)$, indicates that a load of d units is entering/leaving S by arc a. The capacity of the vehicle that is actually carrying that load is irrelevant to the derivation of ECCs.

A more recent algorithm [8], based on pricing of elementary routes and cutting in the DWM variables, is a little better and can solve some instances with 100 customers.

4.1.3 Split Delivery VRP

The Split Delivery Vehicle Routing Problem (SDVRP) is a relaxation of the CVRP where a customer can be visited by more than one vehicle. In most practical situations it is not possible (or convenient) to split deliveries. However, there are cases when splitting deliveries is feasible and indeed brings significant savings. The SDVRP is even harder than the HFVRP, some instances with only 32 customers still remain open.

The branch-cut-and-price in [32] works over a formulation that further extends the load-indexed formulation. Variables x_a^{qd} indicate the number of vehicles that traverse arc $a = (i, j)$ with exactly q units of load and deliver d units at customer j. ECCs are separated. The results are mixed. While the algorithm consistently found lower bounds better than those in the literature, those bounds are still not good enough for solving most

instances to optimality. In a harder set of instances, the average bound was improved from 6.5% to 5.0%. In another set, the improvement was from 3.2% to 1.6%. Perhaps, other cuts (currently unknown) that take the additional delivery dimension in the variables into account could be more effective.

4.1.4 Time Dependent TSP

The Time Dependent TSP (TDTSP) is a generalization of the Asymmetric TSP where the arc costs depend on their *position* in the route with respect to a depot vertex 0. An important particular case of the TDSTP is the Traveling Deliveryman Problem (TDP), known also as the Cumulative TSP or as the Minimum Latency Problem. The TDP looks for a route that minimizes the average time to visit a customer. The TDP is much more difficult than the classical TSP; most instances with only 60 vertices could not be solved until 2010.

The Picard-Queyranne formulation for the TDTSP [40] is equivalent to the load-indexed formulation (23) with unitary demands. The article [1] contains a polyhedral investigation of the TDTSP over that formulation. Three families of (usually) facet-defining inequalities were identified: Admissible Flows, Lifted Subtours and Triangle Cliques (the same already used on other VRPs). A branch-cut-and-price algorithm, pricing q-routes without 5-cycles and separating those cuts was implemented. On tests with the TDP instances from the TSPLIB having from 42 to 107 vertices, the cuts reduced the average gap from 2.26% to 0.60%. Most of those instances were solved to optimality.

4.2 The Parallel Machine Scheduling Problem over the Arc-Time Indexed Formulation

Let $J = \{1, \ldots, n\}$ be a set of jobs to be processed in a set of parallel identical machines $M = \{1, \ldots, m\}$ without preemption. Each job has a positive integral processing time p_j and is associated with a cost function $f_j(C_j)$ over its completion time. Each machine can process at most one job at a time and each job must be processed by a single machine. The general parallel identical machine scheduling problem consists of sequencing the jobs in the machines (perhaps introducing idle times) in order to minimize $\sum_{j=1}^{n} f_j(C_j)$.

Many classical scheduling problems are particular cases of the above problem. For example, in the weighted-tardiness scheduling problem each job has a due date d_j and a weight w_j, and the cost function of job j is defined as $w_j T_j$, where $T_j = \max\{0, C_j - d_j\}$ is the tardiness of job j with respect to its due date. This problem is referred in the scheduling literature as $1||\sum w_j T_j$ for the single machine case and as $P||\sum w_j T_j$ for the parallel identical machines case.

The Arc-Time Indexed Formulation (ATIF) was independently proposed in [47], [46], and [39] (the first two works only address the single machine case). It has variables x_{ij}^t, meaning that job i completes and job j starts at time t on the same machine. The Time Indexed Formulation (TIF), that only has variables x_j^t indicating that a job j starts at time t, was proposed earlier in [19]. It was verified that the ATIF can be significantly stronger than the TIF, specially in the single machine case. Moreover, in [39] it was found that the ATIF is almost isomorphic to the load-indexed formulation for the VRP: the machines correspond to the vehicles, jobs to the customers, and the processing times

to customer demands. This means that known VRP inequalities for the load-indexed formulation can be easily adapted and used to further strengthen the ATIF. Table 4 presents results on $1||\sum w_j T_j$ and $P||\sum w_j T_j$ instances from the ORLIB. The columns are the average gaps with respect to the best known solutions (half of the instances with $m > 1$ and $n = 100$ are still open) of the linear relaxations of the TIF, the ATIF, and the ATIF with additional separation of ECCs. It can be seen that the ATIF is already very strong when $m = 1$, however, when $m > 1$, it is not much stronger than the TIF. In those cases, the separation of ECCs (which can only be done in the arc-time space) is quite effective.

Table 4: Gaps over $1||\sum w_j T_j$ and $P||\sum w_j T_j$ instances.

n	m	TIF	ATIF	+ECCs
40	1	0.68	0.00	0.00
	2	1.53	1.24	0.04
	4	0.54	0.41	0.20
50	1	0.74	0.00	0.00
	2	0.53	0.49	0.09
	4	0.53	0.49	0.27
100	1	0.52	0.02	0.00
	2	1.80	0.69	0.42
	4	0.51	0.49	0.36

The particularly large size of the ATIF on those instances is certainly an issue. There are $O(n^3 p_{avg}/m)$ variables in that formulation, where p_{avg} is the average processing time of a job. In the ORLIB instances, $p_{avg} = 50.5$, thus instances with $n = 100$ have many millions of variables. In contrast, the load-indexed formulation has $O(n^2 C)$ variables, and C is seldom more than 200 in the CVRP instances from the literature. A number of computational techniques for coping with that difficulty are described in [39].

5 Conclusions

This chapter tried to present in a unified framework a number of works by the author (and several collaborators) that use cuts over extended formulations for some network design, vehicle routing and scheduling problems. All those formulations can be viewed as particular cases of a discretized FCNF reformulation, possibly with additional degree constraints. In fact, a single family of cuts, the ECCs, that are derived only from the FCNF structure, turned out to be effective on all those problems. Moreover, it is now quite clear that the ECCs and some other families of problem-specific cuts over those extended spaces can be significantly stronger than currently known cuts over the original spaces. It must be stressed that the potential gains in terms of reduced duality gaps come together with non-trivial computational difficulties. Besides having to deal with a pseudo-polynomially large number of variables, one also has to face frequent problems with slow cut convergence.

Acknowledgements: EU received support from CNPq grant 304533/02-5 and FAPERJ grant E-26/110.550/2010. The author thanks Anand Subramanian and Artur Pessoa for correcting errors in the manuscript.

References

[1] H. Abeledo, R. Fukasawa, A. Pessoa, and E. Uchoa. *The Time Dependent Traveling Salesman Problem: Polyhedra and Algorithm*. Optimization online, 2010.

[2] Y. Aneja. An integer linear programming approach to the Steiner problem in graphs. *Networks*, 10:167–178, 1980.

[3] D. Applegate, R. Bixby, V. Chvátal, and W. Cook. TSP cuts which do not conform to the template paradigm. *Computational Combinatorial Optimization*, pages 261–303, 2001.

[4] D. Applegate, R. Bixby, V. Chvátal, W. Cook, D. Espinoza, M. Goycoolea, and K. Helsgaun. Certification of an optimal TSP tour through 85,900 cities. *Operations Research Letters*, 37:11–15, 2009.

[5] J. Araque, L. Hall, and T. Magnanti. Capacitated trees, capacitated routing, and associated polyhedra. Technical Report OR232-90, MIT, Operations Research Center, 1990.

[6] R. Baldacci, E. Bartolini, A. Mingozzi, and R. Roberti. An exact solution framework for a broad class of vehicle routing problems. *Computational Management Science*, 7:229–268, 2010.

[7] R. Baldacci, N. Christofides, and A. Mingozzi. An exact algorithm for the vehicle routing problem based on the set partitioning formulation with additional cuts. *Mathematical Programming*, 115:351–385, 2008.

[8] R. Baldacci and A. Mingozzi. A unified exact method for solving different classes of vehicle routing problems. *Mathematical Programming*, 120:347–380, 2009.

[9] T. Bektas and G. Laporte. The pollution-routing problem. *Transportation Research B, To appear*, 2011.

[10] E.A. Boyd. Generating fenchel cutting planes for knapsack polyhedra. *SIAM journal on optimization*, 3:734, 1993.

[11] E.A. Boyd. Fenchel cutting planes for integer programs. *Operations Research*, 42:53–64, 1994.

[12] S. Chopra and M.R. Rao. *The Steiner tree problem I: Formulations, compositions and extension of facets*. Working paper, New York University, 1988.

[13] S. Chopra and M.R. Rao. *The Steiner tree problem II: Properties and classes of facets*. Working paper, New York University, 1988.

[14] N. Christofides, A. Mingozzi, and P. Toth. Exact algorithms for the vehicle routing problem, based on spanning tree and shortest path relaxations. *Mathematical Programming*, 20:255–282, 1981.

[15] A. Claus and N. Maculan. *Une nouvelle formulation du probleme de Steiner sur un graphe*. Université de Montréal, Centre de recherche sur les transports, 1983.

[16] A.M. Costa. A survey on benders decomposition applied to fixed-charge network design problems. *Computers & operations research*, 32:1429–1450, 2005.

[17] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Journal of the Operations Research Society of America*, 2:393–410, 1954.

[18] S. Dash, R. Fukasawa, and O. Günlük. On a generalization of the master cyclic group polyhedron. *Mathematical Programming*, 125:1–30, 2010.

[19] M. Dyer and L. Wolsey. Formulating the single machine sequencing problem with release dates as a mixed integer program. *Discrete Applied Mathematics*, 26:255–270, 1990.

[20] K. Fox, B. Gavish, and S. Graves. An n-constraint formulation of the (time-dependent) traveling salesman problem. *Operations Research*, 28:1018–1021, 1980.

[21] R. Fukasawa, H. Longo, J. Lysgaard, M. Poggi de Aragão, M. Reis, E. Uchoa, and R. F. Werneck. Robust branch-and-cut-and-price for the capacitated vehicle routing problem. *Mathematical Programming*, 106:491–511, 2006.

[22] I. Gamvros, B. L. Golden, and S. Raghavan. The multilevel capacitated minimum spanning tree problem. *INFORMS Journal on Computing*, 18:348–365, 2006.

[23] M. Goemans. The Steiner tree polytope and related polyhedra. *Mathematical programming*, 63:157–182, 1994.

[24] L. Gouveia. A 2n-constraint formulation for the capacitated minimal spanning tree problem. *Operations Research*, 43:130–141, 1995.

[25] L. Gouveia and L. Hall. Multistars and directed flow formulations. *Networks*, 40:188–201, 2002.

[26] L. Gouveia and M. Lopes. The capacitated minimum spanning tree problem: On improved multistar constraints. *European Journal of Operational Research*, 160:47–62, 2005.

[27] M. Grötschel, C.L. Monma, and M. Stoer. Facets for polyhedra arising in the design of communication networks with low-connectivity constraints. *SIAM Journal on Optimization*, 2:474, 1992.

[28] L. Hall. Experience with a cutting plane algorithm for the capacitated spanning tree problem. *INFORMS Jounal On Computing*, 8:219–234, 1996.

[29] S. Irnich and D. Villeneuve. The shortest-path problem with resource constraints and k-cycle elimination for k $\geq$ 3. *INFORMS Journal on Computing*, 18:391, 2006.

[30] L. Lovász and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM Journal on Optimization*, 1:166–190, 1991.

[31] J. Lysgaard, A. Letchford, and R. Eglese. A new branch-and-cut algorithm for the capacitated vehicle routing problem. *Mathematical Programming*, 100:423–445, 2004.

[32] L. Moreno, M. Poggi de Aragão, and E. Uchoa. Improved lower bounds for the split delivery vehicle routing problem. *Operations Research Letters*, 38:302–306, 2010.

[33] D. Naddef. Polyhedral theory and branch-and-cut algorithms for the symmetric TSP. *The traveling salesman problem and its variations*, pages 29–116, 2004.

[34] D. Naddef and G. Rinaldi. Branch-and-cut algorithms for the capacitated VRP. In P. Toth and D. Vigo, editors, *The Vehicle Routing Problem*, chapter 3, pages 53–84. SIAM, 2002.

[35] D. Pecin, A. Pessoa, M. Poggi de Aragão, and E. Uchoa. Experiments with new cuts on the VRP. In *Abstracts of the EURO XXIV*, page 73, Lisbon, 2010.

[36] A. Pessoa, M. Poggi de Aragão, and E. Uchoa. A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. In *Proceedings of the 6th international conference on Experimental algorithms*, pages 150–160. Springer-Verlag, 2007.

[37] A. Pessoa, M. Poggi de Aragão, and E. Uchoa. Robust branch-cut-and-price algorithms for vehicle routing problems. *The vehicle routing problem: Latest advances and new challenges*, pages 297–325, 2008.

[38] A. Pessoa, E. Uchoa, and M. Poggi de Aragão. A robust branch-cut-and-price algorithm for the heterogeneous fleet vehicle routing problem. *Networks*, 54:167–177, 2009.

[39] A. Pessoa, E. Uchoa, M. Poggi de Aragão, and R. Rodrigues. Exact algorithm over an arc-time-indexed formulation for parallel machine scheduling problems. *Mathematical Programming Computation*, 2:259–290, 2010.

[40] J.C. Picard and M. Queyranne. The time-dependent traveling salesman problem and its application to the tardiness problem in one-machine scheduling. *Operations Research*, 26:86–110, 1978.

[41] Y. Pochet and L.A. Wolsey. *Production planning by mixed integer programming*. Springer Verlag, 2006.

[42] M. Poggi de Aragão and E. Uchoa. Integer program reformulation for robust branch-and-cut-and-price. In *Annals of Mathematical Programming in Rio*, pages 56–61, Búzios, Brazil, 2003.

[43] M. Poggi de Aragão, E. Uchoa, and R.F. Werneck. Dual heuristics on the exact solution of large Steiner problems. *Electronic Notes in Discrete Mathematics*, 7:150–153, 2001.

[44] T. Polzin and S.V. Daneshmand. Improved algorithms for the Steiner problem in networks. *Discrete Applied Mathematics*, 112:263–300, 2001.

[45] H. Sherali and W. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM Journal on Discrete Mathematics*, 3:411, 1990.

[46] F. Sourd. New exact algorithms for one-machine earliness-tardiness scheduling. *INFORMS Journal on Computing*, 21:167–175, 2009.

[47] S. Tanaka, S. Fujikuma, and M. Araki. An exact algorithm for single-machine scheduling without machine idle time. *Journal of Scheduling*, 12:575–593, 2009.

[48] E. Uchoa. Robust branch-and-cut-and-price for the CMST problem and extended capacity cuts. Presentation in the MIP 2005 Workshop, Minneapolis, 2005. Available at http://www.ima.umn.edu/matter/W7.25-29.05/activities/Uchoa-Eduardo/cmst-ecc-IMA.pdf.

[49] E. Uchoa, R. Fukasawa, J. Lysgaard, A. Pessoa, M. Poggi de Aragão, and D. Andrade. Robust branch-cut-and-price for the capacitated minimum spanning tree problem over a large extended formulation. *Mathematical Programming*, 112:443–472, 2008.

[50] E. Uchoa, T. Toffolo, M.C. Souza, A. Martins, and R. Fukasawa. Branch-and-cut and hybrid local search for the multi-level capacitated minimum spanning tree problem. *Networks, To appear*, 2011.

[51] R.T. Wong. A dual ascent approach for Steiner tree problems on a directed graph. *Mathematical Programming*, 28:271–287, 1984.

[52] H. Yaman. Formulations and valid inequalities for the heterogeneous vehicle routing problem. *Mathematical programming*, 106:365–390, 2006.