ON TIGHTENING THE RELAXATIONS
OF MILLER-TUCKER-ZEMLIN FORMULATIONS
FOR ASYMMETRIC TRAVELING SALESMAN PROBLEMS

HANIF D. SHERALI

Department of Industrial and Systems Engineering (0118), Virginia Polytechnic Institute and State University,
Blacksburg, Virginia 24061

PATRICK J. DRISCOLL

Department of Systems Engineering, U.S. Military Academy, West Point, New York 10996, fp5543@usma.edu

(Received August 1997; revisions received June 2000, February 2001; accepted March 2001)

This paper is concerned with applying the Reformulation-Linearization Technique (RLT) to derive tighter relaxations for the Asymmetric Traveling Salesman Problem (ATSP) formulation that is based on the Miller-Tucker-Zemlin (MTZ) subtour elimination constraints. The MTZ constraints yield a compact representation for the Traveling Salesman Problem (TSP), and their use is particularly attractive in various routing and scheduling contexts that have an embedded ATSP structure. However, it is well recognized that these constraints yield weak relaxations, and with this motivation, Desrochers and Laporte (1991) have lifted the MTZ constraints into facets of the underlying ATSP polytope. We show that a novel application of the RLT process from a nonstandard MTZ representation of the ATSP reveals a new formulation of this problem that is compact, and yet theoretically as well as computationally dominates the lifted-MTZ formulation of Desrochers and Laporte. This approach is also extended to derive tight formulations for the Precedence Constrained Asymmetric Traveling Salesman Problem (PCATSP), based on the MTZ subtour elimination constraints. Additional classes of valid inequalities are also developed for both these versions of the ATSP, and further ideas for developing tighter representations are suggested for future investigations.

A classic statement of the Traveling Salesman Problem (TSP) can be given as follows: A salesman is required to visit \( n \) cities, indexed by \( 1, 2, \ldots, n \), leaving from some base city, say city 1, visiting each of the other \( (n-1) \) cities exactly once, and then returning to city 1. Typically, the problem seeks to identify an itinerary that minimizes the total distance traveled by the salesman. In graph theoretical terminology, this problem is equivalent to determining a least cost Hamiltonian circuit on a complete graph \( K_N \equiv G(N, A) \), where \( N = \{1, \ldots, n\} \) is the set of nodes or vertices, and \( A = \{(i, j) : i, j \in N, i \neq j\} \) is the set of arcs. It is commonly assumed that the matrix of costs, or distances, \( c_{ij} \), is positive, where \( c_{ij} \) represents the cost of traversing from node \( i \) to node \( j \). In this paper, we assume that \( c_{ij} \neq c_{ji} \), possibly, so that we are explicitly addressing the Asymmetric Traveling Salesman Problem (ATSP).

There exists a wide body of literature on this class of problems (see, for example, Lawler et al. 1992 and Padberg and Sung 1992). Apart from purely heuristic approaches, exact solution methods employ two general strategies for resolving the inherent integrality requirement of ATSP within a branch-and-bound/cut process. The first focuses on identifying and exploiting efficient cutting planes during run-time, whose purpose is to successively reduce the size of the feasible polyhedral region. The second strategy seeks to tighten the polyhedral representation of the initial formulation as much as possible prior to initiating any computational solution procedure. This latter strategy is motivated by the prevalence within commercial software of the practice of using the bounds produced by the linear programming relaxation of the initial formulation to guide branching decisions. In this environment, regardless of the run-time actions taken, it is always in a modeler’s best interest to be working with the tightest, tractable, problem formulation available. The results presented in this paper are based on this motivation.

Our focus in this paper will be on a particular formulation of the ATSP based on the Miller-Tucker-Zemlin (MTZ) (1960) subtour elimination constraints, given as follows, where \( x \equiv (x_{ij} : i = 1, \ldots, n, j = 1, \ldots, n, i \neq j) \), and where \( x_{ij} \) takes on a value of 1 if the salesman transitions from node \( i \) to node \( j \), and 0 otherwise.

ATSP-MTZ:

\[
\begin{align*}
\text{Minimize} \quad & \sum_{i,j} c_{ij} x_{ij} : x \in X_{\text{ASSN}} \cap S_{\text{MTZ}}, \ x \text{ binary} \\
\text{subject to} \quad & \sum_{j \neq i} x_{ij} = 1 \quad \forall i = 1, \ldots, n, \\
& \sum_{i \neq j} x_{ij} = 1 \quad \forall j = 1, \ldots, n, \ x \geq 0
\end{align*}
\]

where the set \( X_{\text{ASSN}} \) represents the standard assignment constraints given by

\[
X_{\text{ASSN}} = \left\{ x : \sum_{j \neq i} x_{ij} = 1 \quad \forall i = 1, \ldots, n, \\
\sum_{i \neq j} x_{ij} = 1 \quad \forall j = 1, \ldots, n, \ x \geq 0 \right\}
\]
and where $S_{\text{MTZ}}$ represents the MTZ subtour elimination constraints:

$$S_{\text{MTZ}} = \{ x : \exists a \text{ collection } \{ u_1, u_2, \ldots, u_n \} \text{ with } u_1 \equiv 0$$

$$\text{and } 1 \leq u_j \leq (n-1) \text{ for } j = 2, \ldots, n,$$

$$\text{such that } u_j \geq (u_j + 1) - (n-1)(1-x_{ij}) \forall i, j \geq 2, i \neq j \}.$$

The physical interpretation of the indices $u_i$, $i = 1, \ldots, n$, is that they represent the rank-order in which the nodes/cities are visited, with the base city being assigned a rank of zero. The constraints defining $S_{\text{MTZ}}$ prevent subtours by providing a simple contradiction upon surrogating the last set of constraints in (2) over nodes in any subtour that does not involve the base city 1. The attractiveness of (2) lies in its compact polynomial representation ($O(n^3)$ constraints), which is particularly useful when the ATSP arises as an embedded structure within the context of a larger problem, such as in routing (see Desrochers and Laporte 1991), machine scheduling (see Sherali et al. 1990), or distribution problems (see Sherali et al. 1996), for example. However, it is well known that the MTZ constraints produce a weak linear programming (LP) relaxation (see Langevin et al. 1990 and Padberg and Sung 1992, for example). Using the Fourier-Motzkin elimination to project several TSP formulations into a common variable space, Orman and Williams (1999) recently demonstrated that the MTZ formulation yields an LP relaxation polytope which contains some of seven existing formulations, thereby underscoring its weakness.

In contrast, the Dantzig-Fulkerson-Johnson (DFJ) (1954) formulation (ATSP-DFJ), in which the subtour elimination constraints in ATSP-MTZ are replaced by

$$S_{\text{DFJ}} = \left\{ x : \sum_{(i,j) \in S} x_{ij} \leq |S| - 1 \forall S \subseteq N, \ 2 \leq |S| \leq (n-1) \right\},$$

provides a far tighter LP relaxation, but at the expense of introducing an exponential number of constraints. This necessitates the use of a relaxation strategy whenever these subtour elimination constraints are employed, wherein elements defining (3) are generated only as needed, i.e., to defeat subtours as they arise in the course of solving the problem.

In the aforementioned applications where the ATSP is only a substructure within the overall models, generating subtour elimination constraints as needed can be inconvenient, and so, elements of the DFJ constraints are frequently used to simply augment the MTZ constraints that principally support the model. With this motivation, Desrochers and Laporte (1991) lifted the MTZ constraints into facets of the underlying ATSP polytope, relying on the result of Grötschel and Padberg (1979) that the two-city subtour elimination constraints in (3) are facet-defining inequalities for $n \geq 6$. This revised formulation (ATSP-DL) of ATSP-MTZ replaces the subtour elimination constraints by the lifted inequalities: $S_{\text{DL}} = \{ x : \exists a \text{ collection } \{ u_1, \ldots, u_n \}$ with $u_1 \equiv 0$, such that

$$u_j \geq (u_j + 1) - (n-1)(1-x_{ij}) + (n-3)x_{ij} \forall i, j \geq 2, i \neq j,$$

$$1+(1-x_{ij})+(n-3)x_{ij} \leq u_j \leq (n-1)-(n-3)x_{ij}-(1-x_{ij}) \forall j = 2, \ldots, n.$$

Desrochers and Laporte (1991) established that (4) are facet defining, while (5) are valid inequalities. Driscoll (1995) demonstrated that a similar argument can be used to show that (5) also define facets of the ATSP polytope. More recently, Gouveia and Pires (1996) have developed a formulation based on a disaggregation of the MTZ constraints for ATSP, whose linear programming relaxation is characterized by a set of circuit inequalities for the ATSP as defined in Grötschel and Padberg (1985). These inequalities are then lifted into several facet defining inequalities that are not dominated by the subtour elimination constraints of DFJ.

In this paper, we first develop in §1 an even tighter representation of ATSP-MTZ. We begin with a nonstandard restatement of ATSP-MTZ that involves nonlinear product terms. To this problem, we apply a specialized version of the Reformulation-Linearization Technique (RLT) of Sherali and Adams (1990, 1994), exploiting inherent special structures as in Sherali et al. (1998). The resulting formulation of ATSP affords a new $O(n^3)$ representation of this problem that is shown to both theoretically as well as computationally dominate the lifted-MTZ formulation ATSP-DL of Desrochers and Laporte (1991). Section 2 provides our related computational results, and suggests even tighter representations that are still polynomial in size, although appreciably larger, being of $O(n^5)$ and $O(n^6)$. The actual implementation of these latter ideas requires a deeper study that is recommended for future research.

Section 3 extends our analysis to the Precedence Constrained ATSP (PCATSP). This problem, also referred to as the sequential ordering problem, incorporates precedence relationships within ATSP that require certain cities to be visited before others. Precedence constraints of this type often appear in scheduling applications (Gillies and Liu 1995) and delivery and routing applications (Psaraftis 1980), and have also more recently been applied to assigning locations to microprocessor components on a printed circuit board (Al-Majoub 1996). By examining the polyhedral structure of the PCATSP, Ascheuer et al. (1990) and Escudero et al. (1991) have been able to identify and lift various valid cutting planes stemming from this problem’s formulation. Balas et al. (1995) have also derived several families of valid inequalities, and have proposed various
lified versions of the DFJ subtour elimination constraints for the PCATSP, along with polynomial-time separation algorithms for important subfamilies of these valid inequalities.

The observation that motivates our particular approach is that the rank ordering imposed by the MTZ node labeling scheme provides a natural facility for precedence ordering. Hence, with the same motivation as for the ATSP, we extend our analysis to derive new, tight, \( O(n^2) \) representations for the MTZ-based formulation of the PCATSP.

In addition, we also present several new classes of valid inequalities for this problem. Finally, §4 provides some computational results that exhibit the effect of our new approach on tightening the LP relaxation of the PCATSP. Section 5 concludes this paper.

1. APPLICATION OF RLT TO TIGHTEN ATSP-DL

Consider the following restatement of the ATSP-MTZ formulation introduced previously:

\[
\text{ATSP-MTZ2: } \text{Minimize } \sum_i \sum_j c_{ij} x_{ij}
\]

subject to:

\[
x \in X_{\text{ASSN}} \text{ of Equation (1)}, \tag{6}
\]

\[
u_{ij} x_{ij} = (u_i + 1)x_{ij} \quad \forall i, j \geq 2, \; i \neq j, \tag{7}
\]

\[
u_{ij} x_{ij} = x_{ij} \quad \forall j \geq 2, \tag{8}
\]

\[
u_{ij} x_{ij} = (n-1)x_{ij} \quad \forall j \geq 2, \tag{9}
\]

\[
1 \leq u_{ij} \leq (n-1) \quad \forall j \geq 2, \tag{10}
\]

\[x \text{ binary}. \tag{11}\]

Observe that the nonlinear constraints (7)–(10) are subtour elimination constraints that, similar to the (linear) MTZ constraints given by (2), enforce that for any \( j \geq 2, u_{ij} = 1 \) if \( x_{ij} = 1 \), and \( u_{ij} = (n-1) \) if \( x_{ij} = 0 \), and that \( u_{ij} = (u_i + 1) \) if \( x_{ij} = 1 \) for all \( i, j \geq 2, i \neq j \). These conditions, in combination with (6) and (11), defeat any subtour that does not involve the base city 1, thereby admitting only Hamiltonian circuits.

Let us now apply a specialized version of RLT to ATSP-MTZ2, following Sherali et al. (1998). To contain the size of the resulting relaxation, we shall apply only a partial first-level version of this approach that will maintain \( O(n^2) \) variables and constraints. Our approach is to first reformulate ATSP-MTZ2 by generating additional (nonlinear) implied constraints. Upon applying a subsequent linearization which uses a substitution of variables in place of each distinct nonlinear term, we expose useful relationships among the new and original variables that tend to enforce the nonlinear relationships. Specifically, we perform the following operations.

Reformulation Phase. Construct additional sets of constraints via (R1)–(R4) stated below.

(R1) Using the assignment constraints (6), construct the valid equalities \( u_{ij} \sum_{j \neq i} x_{ij} = 0 \) for all \( i \geq 2, \) and \( u_{ij} \sum_{j \neq i} x_{ij} = 0 \) for all \( j \geq 2. \)

(R2) For each \( j \geq 2, \) multiply the inequality \( (u_{ij} - 1) \geq 0 \) of (10) by (i) \( x_{ij} \geq 0 \) for each \( i \geq 2, i \neq j; \) and by (ii) \( (1 - x_{ij} - x_{ji}) \geq 0 \) for each \( i \geq 2, i \neq j. \) Notice that these multiplications clearly yield valid inequalities of type (i) \( (u_{ij} - 1)x_{ij} \geq 0 \) for all \( i, j \geq 2, i \neq j; \) and (ii) \( (u_{ij} - 1)(1 - x_{ij} - x_{ji}) \geq 0 \) for all \( i, j \geq 2, i \neq j, \) respectively, since they are simply products of nonnegative factors. Observe that the inequalities used as multipliers in (ii) are the two-city DFJ subtour elimination constraints defined in (3). Since \( \{x_{ij} + x_{ji} \leq 1, x_{ij} \geq 0, x_{ji} \geq 0\} \) implies the set of constraints \( \{0 \leq x_{ij} \leq 1, 0 \leq x_{ji} \leq 1\}, \) though not vice-versa, the use of the factors \( (1 - x_{ij} - x_{ji}) \) generates potentially tighter relaxations than that obtained using the simple bound-factors \( (1 - x_{ij}) \) and \( (1 - x_{ji}) \) as discussed more generally in Sherali et al. (1998). Note that other known valid inequalities that imply these bound-factors could be used as RLT factors in this context. However, one needs to balance this strategy with the number of distinct nonlinear terms produced in order to curtail the size of the resulting relaxation obtained after applying the ensuing linearization phase.

(R3) Similar to (R2), using the upper bounding conditions in (10), construct the valid inequalities (i) \( (n - 1 - u_{ij})x_{ij} \geq 0 \) for all \( i, j \geq 2, i \neq j; \) and (ii) \( (n - 1 - u_{ij})(1 - x_{ij} - x_{ji}) \geq 0 \) for all \( i, j \geq 2, i \neq j. \)

(R4) For each \( j \geq 2, \) construct the base city product constraints \( (u_{ij} - 1)(1 - x_{ij} - x_{ji}) \geq 0 \) and \( (n - 2 - u_{ij})(1 - x_{ij} - x_{ji}) \geq 0 \). When \( x_{ij} = 1 \) or \( x_{ji} = 1, \) these constraints are trivially valid. Likewise, when \( x_{ij} = 0 \) and \( x_{ji} = 0, \) since we must then have \( 2 \leq u_{ij} \leq (n - 2), \) these constraints are again valid. Notice that although we could have additionally generated the valid product constraints \( (u_{ij} - 2)x_{ij} \geq 0 \) and \( (n - 2 - u_{ij})x_{ij} \geq 0 \) for all \( i, j \geq 2, i \neq j, \) in the same spirit, these constraints would yield the same restrictions as in (R2)(i) and (R3)(i), respectively, because of relationship (7).

Linearization Phase. Linearize ATSP-MTZ2 along with the new classes of constraints (R1)–(R4) generated above by using the substitution

\[
y_{ij} = u_i x_{ij} \quad \text{and} \quad z_{ij} = u_j x_{ij} \quad \forall i, j \geq 2, i \neq j, \tag{12}\]

and, as in (8) and (9), by replacing

\[
u_{ij} x_{ij} \text{ by } x_{ij} \quad \text{and} \quad u_j x_{ij} \text{ by } (n-1)x_{ij} \quad \forall j \geq 2. \tag{13}\]

Additionally, noting that under (7) and (12), we can eliminate \( z_{ij} \) from the resulting problem through the relationship

\[
z_{ij} = y_{ij} + x_{ij} \quad \forall i, j \geq 2, i \neq j, \tag{14}\]

R1this RLT process yields the formulation ATSP-SD given below. Proposition 1 then shows ATSP-SD to be a valid statement of ATSP that produces a tighter LP relaxation.
than does ATSP-DL, which in turn has been shown to produce a tightened representation for ATSP-MTZ.

ATSP-SD: Minimize \( \sum_{i,j} c_{ij} x_{ij} \)

subject to:

\[
x \in X_{\text{ASSN}},
\]

\[
\sum_{j \neq i, i \neq 1} y_{ij} + (n - 1)x_{i1} = u_i \quad \forall i \geq 2,
\]

\[
\sum_{j \neq i, i \neq 1} y_{ji} + 1 = u_j \quad \forall j \geq 2,
\]

\[
x_{ij} \leq y_{ij} \leq (n - 2)x_{ij} \quad \forall i,j \geq 2, i \neq j,
\]

\[
u_j + (n - 2)x_{ij} - (n - 1)(1 - x_{ji}) \leq y_j + y_{ji} \leq u_j - (1 - x_{ji}) \quad \forall i,j \geq 2, i \neq j,
\]

\[
1 + (1 - x_{ij}) + (n - 3)x_{ji} \leq u_j \leq (n - 1) - (n - 3)x_{ij} - (1 - x_{ji}) \quad \forall j \geq 2,
\]

\[x \text{ binary.}
\]

Under the linearization (12), (13), and substitution (14), the reformulation step (R1) produces the constraints (16) and (17). Steps (R2)(i) and (R3)(i) yield the lower and upper bounding restrictions in (18), respectively. Steps (R3)(ii) and (R2)(ii) similarly yield the lower and upper bounding restrictions in (19), respectively, and the two sets of constraints generated at step (R4) produce the bounding inequalities given in (20). Note that the new variable \( y_{ij} \) represents the rank-order (starting with rank 0) of the arc \( (i,j) \) in the tour, if it appears in the tour, being zero otherwise. The interpretation of each constraint is thereby evident by considering the possible zero or one values for the accompanying \( x \)-variable. For example, consider (19), and suppose that \( x_{ij} = 1 \) and \( x_{ji} = 0 \) for \( i,j \geq 2 \) (note that the surrogate of the two inequalities in (19) implies that \( x_{ij} + x_{ji} \leq 1 \) for \( n \geq 3 \)). Then, (19) reduces to \( y_{ij} + y_{ji} = (u_j - 1) \). The interpretation of the \( y \)-variables confirms this relationship since we must have \( y_{ji} = 0 \) (this is implied by (18) because \( x_{ji} = 0 \)), and \( y_{ij} = u_j \) (this is implied by (16), noting (15), (18), and (21)), leading to the valid requirement that \( u_i = u_j - 1 \) in this case. Proposition 1 below offers a more formal proof of validation for this formulation.

Observe that the particular valid constraints generated during the reformulation phase produce only those types of nonlinear product terms (of the type \( u_i x_{ij} \) and \( u_j x_{ji} \)) that are already present in ATSP-MTZ2. We have pointedly avoided other valid product constraints that would produce additional distinct product terms in order to curtail the size of the resulting relaxation. (Section 2 addresses the issue of going beyond this to generate tighter, higher-dimensional representations.) The particular constraints generated in Step (R4) of the reformulation phase are special instances of new \textit{conditional-logic} based inequalities introduced by Sherali et al. (1998). Note also that the constraints generated in steps (R1), (R2)(ii), (R3)(ii), and (R4) can be viewed in a broader sense as special cases of \textit{generalized factor product constraints} developed in Sherali et al. (1998). Each of these constructions go beyond the original RLT approach of Sherali and Adams (1990, 1994) that is based on using the simple bound-factors \( x_{ij} \) and \( (1 - x_{ij}) \), as discussed above.

**Proposition 1.** The formulation ATSP-SD is a valid model for the Asymmetric Traveling Salesman Problem. Moreover, it yields an LP relaxation that is tighter than the LP relaxation of ATSP-DL.

**Proof.** The validity of the various constraints generated in ATSP-SD follows by construction and the arguments presented within the reformulation phase of the RLT process. Hence, in order to establish both the assertions of the proposition, it is sufficient to show that the constraints (15)–(20) imply the constraints of ATSP-DL in the continuous sense. Note that constraints (15) and (20) are common for both formulations. Hence, it remains to show that constraints (4) are implied by (15)–(20).

For any pair \( i,j \geq 2, i \neq j \), examine the lower bounding restriction of (19). With \( i \) and \( j \) interchanged, multiply this by \(-1\), and surrogate it with the upper bounding restriction in (19). This yields

\[
(y_{ij} - y_{ji}) + (y_{ji} - y_{ij}) \leq u_j - u_i - 1 + (3 - n)x_{ji} + (n - 1)(1 - x_{ij}) \forall i,j \geq 2, i \neq j,
\]

i.e., \( u_j \geq (u_i + 1) - (n - 1)(1 - x_{ij}) + (n - 3)x_{ji} \) for all \( i,j \geq 2, i \neq j \), which is (4). This completes the proof. \( \square \)

The following proposition is actually implicit in the construction of ATSP-SD, and is also implied by Proposition 1. However, we state and prove this here explicitly to expose the structure of this problem.

**Proposition 2.** The continuous (LP) relaxation of ATSP-SD implies the DFJ subtour elimination constraints of (3) for \( |S| = 2 \), when \( n \geq 5 \).

**Proof.** The two inequalities of (20) imply that \((n - 1) - (n - 3)x_{ij} - (1 - x_{ji}) \geq 1 + (1 - x_{ij}) + (n - 3)x_{ji} \), i.e.,

\[
(n - 4)(x_{ij} + x_{ji} - 1) \leq 0 \quad \forall j \geq 2.
\]

Since \( n \geq 5 \), this implies that (3) holds true for \( S = \{i,j\}, j \geq 2 \). Next, consider \( S = \{i,j\} \) for \( i,j \geq 2, i \neq j \). As above, the two inequalities in (19) imply that \( u_j - (1 - x_{ji}) \geq u_j + (n - 2)x_{ij} - (n - 1)(1 - x_{ji}) \), or that

\[
(n - 2)(x_{ij} + x_{ji} - 1) \leq 0
\]

which yields (3) for this set \( S \) when \( n \geq 3 \). This completes the proof. \( \square \)
2. COMPUTATIONAL RESULTS AND EXTENSIONS FOR HIGHER-ORDER RLT RELAXATIONS

We conducted the following computational experiments to examine whether the lower bound produced by ATSP-SD was consistently as tight or tighter than that of the formulation it theoretically dominates and to examine whether or not such a tightened bound would have the desired effect of reducing the number of nodes processed to achieve optimality. We used both formulations ATSP-SD (sd) and ATSP-DL (dl) to solve standard test cases from TSPLIB (due to M. Fischetti and B. Repetto, available through Rice University) in order to assess their relative effectiveness in closing the inherent LP-IP gap. The exceptionally weak performance of ATSP-MTZ is both well known and documented, and will not be repeated here since the polyhedral relationship it theoretically dominates and to examine whether or not such a tightened bound would have the desired effect of reducing the number of nodes processed to achieve optimality. We used both formulations ATSP-SD (sd) and ATSP-DL (dl) to solve standard test cases from TSPLIB (due to M. Fischetti and B. Repetto, available through Rice University) in order to assess their relative effectiveness in closing the inherent LP-IP gap. The exceptionally weak performance of ATSP-MTZ is both well known and documented, and will not be repeated here since the polyhedral relationship $P_{SD} \subseteq P_{DL} \subseteq P_{MTZ}$ established in the theoretical results of the previous section implies that any lower bound achieved in which $Z_{LP_{SD}} \geq Z_{LP_{DL}}$ necessarily has $Z_{LP_{SD}} \geq Z_{LP_{MTZ}}$.

The computations for the test problems in this section were performed on a Dell XPS 450 computer having 328 Mb RAM, running the NT 4.0 operating system. The General Algebraic Modeling System (GAMS), version 2.25 (1992), was used to generate the models, and the CPLEX MIP solver, version 6.0, was used to solve the resulting problems. Table 1 displays the results obtained using both the default CPLEX MIP solver settings and the custom settings (optfile) chosen to assist in overcoming the difficulties posed by the degeneracy introduced by RLT (see Sherali and Adams 1996). Within optfile, the default CPLEX MIP solver settings were used, except for the following choices for the stated options:

- Branching priority: $x_{ij}$ variables;
- Node selection: best estimate of the integer objective value once integer infeasibilities are removed;
- Initial LP solve: dual;
- Variable selection: based on pseudoreduced costs;
- Pricing strategy: steepest edge with slack initial norms.

Table 1 presents the performance results on six TSPLIB problems, indicating the value $Z_{IP}$ of the confirmed optimum integer objective function value, the lower bound obtained by solving the LP relaxation ($Z_{LP}$), the Best Found integer solution, the total number of nodes processed (Nodes), and the cpu seconds (CPU). The numerical value appended to the problem name indicates the number of cities involved in the problem. An asterisk (*) indicates that the solver terminated prior to identifying a confirmed optimal solution from exceeding either a memory limitation, or a preset node or cpu limit. Several key observations are worth noting. First, the computational evidence confirms that the RLT-tightened ATSP formulation ATSP-SD can be relied upon to produce consistently superior lower bounds at node 0, which is a positive factor in any enumeration strategy.

Second, ATSP-SD using the optfile settings consistently achieved the desired goal of processing the smallest number of nodes to reach confirmed optimality. The cost realized to achieve this goal can be partially attributed to the longer cpu times consumed by ATSP-SD in solving the relaxation for each node subproblem. This was to be expected since RLT formulations tend to be degenerate and ill-conditioned, and can benefit by specialized procedures for solving the associated LP relaxations (see Sherali and Adams 1996). In order to reverse this trend in cpu effort based on the observed significant reduction in the total number of nodes enumerated in a branch-and-bound/cut scheme, one would need to rely on improvements in LP technology to better handle degeneracy and ill-conditioning effects, or use a suitable Lagrangian dual implementation to more rapidly compute lower bounds. This aspect is proposed for future investigation.

Despite the increase in cpu time experienced by the RLT formulation, we find these results encouraging because it is our belief that a stronger formulation will ultimately benefit the solution process by providing tighter representations at each node and reducing the overall number of nodes required to be explored even if current LP technology cannot fully exploit this structure. A smaller number of nodes also generally equates to a reduction in the amount of memory required to store unexplored node information. Case-in-point can be seen in Table 1 with ftv33 using ATSP-DL under the optfile settings. The solution process for this run terminated prematurely because of exceeding memory limitations caused by an excessive growth in the enumeration tree, thereby crippling ATSP-DL’s attempt to identify the confirmed optimal solution value of 1,286. In contrast, ATSP-SD with the optfile settings identified the confirmed optimal solution in both less cpu time and with fewer nodes processed.

For two particularly challenging problem instances from the TSPLIB tested shown in Table 2, neither ATSP-SD nor ATSP-DL were able to close the LP-IP gap. In p43, ATSP-SD encountered the preset cpu limit (100,000 cpu seconds), while ATSP-DL terminated early by exceeding memory limitations due again to the extensive growth in the branching tree (46,723 nodes). The superior lower bound (846.9) achieved by the stronger formulation ATSP-SD, coupled with deeper gap resolution at each node, enabled ATSP-SD to achieve a greater degree of LP-IP gap closure than that attained by ATSP-DL, which is evident upon examination of the absolute gap (abs) and relative gap (rel) remaining to be resolved. Similar comparative performance was also evident in the 100-city problem kr124, in which both formulations terminated by encountering memory limitations. However, prior to doing so, ATSP-SD dramatically outperformed ATSP-DL in closing the LP-IP gap. Hence, we might expect a relatively superior performance for the tighter formulation ATSP-SD when solving large challenging problem instances under a more relaxed optimality tolerance.

It is also instructive to note that allowing CPLEX to automatically scan for set packing binary (SOS3) structures seriously impeded solution progress for both ATSP-SD and ATSP-DL, especially on larger problem instances.
Table 1. Lower bounds and performance of ATSP-SD and ATSP-DL on TSPLIB problems.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Formulation</th>
<th>(Z_{LP})</th>
<th>Best Found</th>
<th>Nodes</th>
<th>CPU</th>
<th>Option</th>
</tr>
</thead>
<tbody>
<tr>
<td>br17</td>
<td>ATSP-DL</td>
<td>87.0</td>
<td>87</td>
<td>8</td>
<td>1.6</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td>Z(L) = 87</td>
<td>ATSP-DL</td>
<td>87.0</td>
<td>26</td>
<td>0.4</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATSP-S</td>
<td>87.0</td>
<td>10</td>
<td>4.4</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td></td>
<td>ATSP-DL</td>
<td>87.0</td>
<td>31</td>
<td>0.3</td>
<td>default</td>
</tr>
<tr>
<td>ftv33</td>
<td>ATSP-S</td>
<td>1,224.5</td>
<td>1,286</td>
<td>4,753</td>
<td>49,641.5</td>
<td>optfile</td>
</tr>
<tr>
<td>(Z_{LP} = 1,286)</td>
<td>ATSP-DL</td>
<td>1,217.2</td>
<td>1,338.0*</td>
<td>527,357</td>
<td>705,488.8</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td>ATSP-S</td>
<td>1,224.5</td>
<td>1,286</td>
<td>18,702</td>
<td>13,933.7</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>ATSP-DL</td>
<td>1,217.2</td>
<td>1,286</td>
<td>86,955</td>
<td>1,495.6</td>
<td>default</td>
</tr>
<tr>
<td>ftv35</td>
<td>ATSP-S</td>
<td>1,415.5</td>
<td>1,473</td>
<td>2,112</td>
<td>1,111.2</td>
<td>optfile</td>
</tr>
<tr>
<td>(Z_{LP} = 1,473)</td>
<td>ATSP-DL</td>
<td>1,413.5</td>
<td>1,473</td>
<td>24,776</td>
<td>471.6</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td>ATSP-S</td>
<td>1,415.5</td>
<td>1,473</td>
<td>6,114</td>
<td>5,427.0</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>ATSP-DL</td>
<td>1,413.5</td>
<td>1,473</td>
<td>40,699</td>
<td>657.7</td>
<td>default</td>
</tr>
<tr>
<td>ftv38</td>
<td>ATSP-S</td>
<td>1,480.0</td>
<td>1,530</td>
<td>2,704</td>
<td>2,333.6</td>
<td>optfile</td>
</tr>
<tr>
<td>(Z_{LP} = 1,530)</td>
<td>ATSP-DL</td>
<td>1,477.1</td>
<td>1,530</td>
<td>16,937</td>
<td>428.7</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td>ATSP-S</td>
<td>1,480.0</td>
<td>1,530</td>
<td>5,286</td>
<td>4,934.9</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>ATSP-DL</td>
<td>1,477.1</td>
<td>1,530</td>
<td>56,087</td>
<td>1,177.5</td>
<td>default</td>
</tr>
<tr>
<td>ftv44</td>
<td>ATSP-S</td>
<td>1,573.7</td>
<td>1,613</td>
<td>1,839</td>
<td>2,141.7</td>
<td>optfile</td>
</tr>
<tr>
<td>(Z_{LP} = 1,613)</td>
<td>ATSP-DL</td>
<td>1,573.7</td>
<td>1,613</td>
<td>5,967</td>
<td>164.7</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td>ATSP-S</td>
<td>1,573.7</td>
<td>1,613</td>
<td>3,658</td>
<td>9,319.6</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>ATSP-DL</td>
<td>1,573.7</td>
<td>1,613</td>
<td>9,504</td>
<td>239.4</td>
<td>default</td>
</tr>
<tr>
<td>ry48</td>
<td>ATSP-S</td>
<td>11,520.2</td>
<td>11,635</td>
<td>149</td>
<td>360.9</td>
<td>optfile</td>
</tr>
<tr>
<td>(Z_{LP} = 11,635)</td>
<td>ATSP-DL</td>
<td>11,513.2</td>
<td>11,635</td>
<td>189</td>
<td>8.7</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td>ATSP-S</td>
<td>11,520.2</td>
<td>11,635</td>
<td>390</td>
<td>1,047.5</td>
<td>default</td>
</tr>
<tr>
<td></td>
<td>ATSP-DL</td>
<td>11,513.2</td>
<td>11,635</td>
<td>356</td>
<td>15.9</td>
<td>default</td>
</tr>
</tbody>
</table>

Table 3 shows the effect of this option on performance for both ATSP-SD and ATSP-DL using problem ftv44 as an illustration. Note that although both ATSP-SD and ATSP-DL obtained the same initial lower bound, the detrimental effect of this option on ATSP-DL is far more pronounced. ATSP-DL consumed significantly greater effort than ATSP-SD and exceeded memory limitations in the process. It is suspected that the CPLEX-imposed order for processing the model constraints was the culprit in this case, because it directly affects the prioritization of dual variables used within the dual simplex solution option chosen for solving the relaxation. This observation underscores the dependence of the relative performance of various alternative formulations on the particular system parameter options and strategies used by any selected solver, thereby further validating the strategy of providing a solver with the tightest possible formulation at the onset.

We now suggest two insightful ideas that may be used to enhance the computational performance of ATSP-SD beyond tightening the lower bound of the LP relaxation. In an actual implementation in which the formulation ATSP-SD is used to solve ATSP, two particular expedients can greatly enhance the effectiveness of the solution procedure and are necessary to solve larger problem instances. First, additional strong valid inequalities can be used to augment this formulation in order to further tighten its LP relaxation. For example, one can generate lifted cycle inequalities as in Grötschel (1977), or two-matching and comb inequalities as in Crowder and Padberg (1980). Second, note that a branch-and-cut approach wherein at some node of the enumeration tree, we might have several \(x\)-variables fixed at zero or one, it is preferable to regenerate the RLT constraints rather than simply perform logical tests on the root node’s formulation itself. That is, by examining the structure of the subgraph corresponding to the fixed variables, one can tighten the bounds on the \(\mu\)-variables, and use these tightened bounds in concert with the RLT scheme described above to update the various coefficients of the RLT constraints in ATSP-SD. This type of preprocessing of node subproblems combines the usual conditional logic tests with the RLT process and can thereby potentially yield tighter reduced representations. This contrasts with using a standard MIP package to solve the formulation of ATSP-SD.

In the same spirit, one can examine more complete first-order RLT relaxations, or even perhaps higher-order RLT relaxations, or even perhaps higher-order RLT relaxations.
relaxations, in order to strengthen the problem representation. Recall that our motivation in generating ATSP-SD via only a partial first-order RLT application was to maintain an $O(n^2)$ formulation. If we were to permit products of $u_i$ with $x_{pq}$ where $j$ is different from $p$ and $q$, as opposed to the types of products $u_ix_j$ and $u_jx_i$ permitted in deriving ATSP-SD, we could obtain a tighter, but $O(n^3)$, representation.

Here, in the reformulation phase, in addition to the types of product-constraints used for ATSP-SD, the assignment constraints defining (6) could be multiplied by each $u$-variable, and the constraints (10) could be multiplied by each $x$-variable and by each two-city, and perhaps three-city, DFJ subtour elimination constraint, using conditional logic based lifted products as before. Furthermore, while maintaining an $O(n^3)$ formulation, we could also permit the creation of product terms of the type $x_{ij}x_{jk}$ for $i \neq j \neq k$ via the RLT process. This would allow the generation of RLT constraints using products of (6) with suitable $x$-variables, and in addition, permit us to include linearized versions of the following types (22) and (23) of quadratic valid inequalities based on three-city interactions. These constraints represent lifted relationships that can be validly imposed on the $u$-variables based on node-triplets, similar to (4) and (5) that are based on node-pairs.

**Proposition 3.** The multilinear (quadratic) constraints

\[ u_i \geq 2 - x_{ik} + x_{ij}x_{jk} \quad (22) \]

and

\[ u_k \geq u_i + 2 - (n - 1)(2 - x_{ij} - x_{jk}) \]

\[ + (n - 2)(1 - x_{ij})(1 - x_{jk}) + (n - 1)x_{jk} + (n - 4)(x_{ki}x_{ij} + x_{kj}x_{ki} + x_{ij}x_{ji}) \quad (23) \]

with $i, j, k \geq 2$ are valid constraints for ATSP, for all distinct $i, j, k \geq 2$.

**Proof.** First consider constraint (22) for any $i, j, k$. If $x_{j}x_{jk} = 0$, then if $x_{ik} = 1$, this constraint is implied by (10), while if $x_{jk} = 0$, then $u_k \geq 2$ is a valid lower bound. On the other hand, if $x_{j}x_{jk} = 1$, then we must have $x_{ik} = 0$ and also, in this case, $u_k \geq 3$ is valid. This verifies the validity of (22).

Next consider the derivation of (23). First, surrogating the MTZ constraints (2) corresponding to the combinations $(i, j)$ and $(j, k)$, we obtain the valid inequality

\[ u_i \geq u_j + 2 - (n - 1)(2 - x_{ij} - x_{jk}). \quad (24) \]

We will show that (23) can be obtained by lifting (24) in a sequential term-by-term fashion. To this end, consider first a tightening of (24) by finding the maximum value of $\theta$ such that

\[ u_k \geq u_i + 2 - (n - 1)(2 - x_{ij} - x_{jk}) + \theta(1 - x_{ij})(1 - x_{jk}) \quad (25) \]

is valid when $x_{ij} = x_{jk} = 0$, when this term is possibly nonzero. Such a value of $\theta$ is given by solving the problem

\[ \theta = \min \{u_k - u_i - 2 + (n - 1)(2 - x_{ij} - x_{jk}) : \]

\[ 2 \leq u_k \leq n, 2 \leq u_i \leq n, x \text{ binary}, x_{ij} = x_{jk} = 0\} . \]

This yields the value $\theta = 2 - n - 2 + (n - 1)2 = (n - 2)$ for use in (25). Next, lifting (25) with a term $\theta x_{jk}$, notice that when $x_{jk} = 1$, $u_k = u_i + 1$, and $x_{ij} = x_{jk} = 0$, we can take $\theta' = \{(u_i + 1) - u_i - 2 + (n - 1)(2 - (n - 2)) = (n - 1)\}$. Finally, let us determine the largest coefficient $\theta''$ for the term $(x_{ki}x_{ij} + x_{kj}x_{ki} + x_{ij}x_{ji})$ in (23) to make this constraint valid. When $x_{ij} = x_{ij} = 1$, we have $u_i = u_i + 1$, $x_{jk} = 0$, the other two quadratic terms $x_{jk}x_{ij}$ and $x_{ij}x_{ji}$ are zeroes. This implies that we must have $\theta'' \leq -1 - 2 + (n - 1) = (n - 4)$. Similarly, when $x_{jk} = x_{ki} = 1$, we get $u_i = u_i + 1$, $x_{ij} = 0$, $x_{jk} = 0$, and the other two quadratic terms, $x_{jk}x_{ij}$ and $x_{ij}x_{ji}$, are zeroes. Hence, we again need $\theta'' \leq -1 - 2 + (n - 1) = (n - 4)$. Finally, when $x_{jk} = x_{ki} = 1$, we get $u_i = u_i + 2$, $x_{ij} = 0$, $x_{jk} = 0$, and the other two quadratic terms, $x_{ij}x_{ij}$ and $x_{ij}x_{ji}$, are zeroes. This yields $\theta'' \leq -2 - 2 + (n - 1) = (n - 4)$. From these three cases, it follows that we can take $\theta'' = (n - 4)$, and this completes the proof.

In a similar vein, if we permit a full first-order RLT relaxation, including the construction of generalized RLT product constraints as in Sherali et al. (1998), we would create an $O(n^4)$ representation that would include $x$-variable products of the type $x_{ij}x_{kl}$ for all indices $i < k$, $j \neq l$. Recently, Ramachandran and Pekny (1996) have considered such full-first-order as well as second-order RLT constructions for deriving strong bounds for solving the quadratic assignment problem (QAP). Since instances of QAP are challenging even for moderate values of $n \times (n \geq 15)$, this is a justifiable strategy. However, for the ATSP, such RLT representations might be far too large for the size of the problems of interest, excepting further significant advances in LP technology. Rather, such higher-order representations might be better used to generate a selection of strong valid inequalities by way of projection onto the original variable space through suitable surrogates.

Table 3. Effect of automatic SOS3 scanning and CPLEX prioritization on ftv44.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Formulation</th>
<th>$Z_{LP}$</th>
<th>Best Found</th>
<th>Nodes</th>
<th>CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>ftv44</td>
<td>ATSP-SD</td>
<td>1,573.75</td>
<td>1,613</td>
<td>3,149</td>
<td>8,250.46</td>
</tr>
<tr>
<td>with SOS3</td>
<td>ATSP-DL</td>
<td>1,573.75</td>
<td>1,786</td>
<td>1,595,923</td>
<td>60,374.0</td>
</tr>
<tr>
<td>ftv44</td>
<td>ATSP-SD</td>
<td>1,573.75</td>
<td>1,613</td>
<td>294</td>
<td>1,386.2</td>
</tr>
<tr>
<td>without SOS3</td>
<td>ATSP-DL</td>
<td>1,573.75</td>
<td>1,613</td>
<td>384</td>
<td>261.9</td>
</tr>
</tbody>
</table>
example, constraint (22) could be surrogated with the RLT constraint generated via the product \((1 - x_{ij})(1 - x_{jk}) \geq 0\) namely, with the constraint \(x_{ij}x_{jk} \geq x_{ij} + x_{jk} - 1\), to yield the valid inequality in the \((x, u)\)-space:
\[
u_k \geq 1 + x_{ij} + x_{jk} - x_{ik}.
\]

Constraints such as (26) could be generated along with other known valid inequalities using suitable separation problems in order to develop an overall algorithmic approach for the ATSP or its variants, driven principally by the formulation ATSP-SD.

While our focus here has been on the ATSP, as in Desrochers and Laporte (1991), similar improvements appear possible for the Vehicle Routing Problem (VRP), the distance constrained VRP, and the VRP with time-windows, using the lifted MTZ framework. In particular, we illustrate this by applying the foregoing concepts to the precedence constrained traveling salesman problem in §3 below.

3. APPLICATION OF RLT TO DERIVE AN MTZ-BASED FORMULATION FOR PCATSP

We now extend our derivation of ATSP-SD via ATSP-MTZ2 to the case of the Precedence Constrained Asymmetric Traveling Salesman Problem (PCATSP). Here, superimposed on the basic ATSP structure, there exist certain precedence relationships, denoted \(i < j\) for designated node pairs \(i, j \geq 2\), that require node \(i\) to be visited before node \(j\). Again, node 1 is taken as the “base city” at which the Hamiltonian tour must commence and terminate. By assuming that \(c_{ij} = c_{ji} = 0\) for all \(j = 2, 3, \ldots, n\), the problem can be equivalently stated as seeking a least-cost traversal of cities 2, \ldots, \(n\) in some order, that is feasible to the precedence constraints.

Let \(G_p = (N \setminus \{1\}, A)\) denote the precedence graph on the node set \(N \setminus \{1\} \equiv 2, \ldots, n\), with arc set \(A\), where \((i, j) \in A \iff i < j\). Arc set \(A\) contains only those arcs representing precedence constraints. We assume that \(G_p\) is circuit-free, so that a permutation exists that is feasible to the given precedence relationships. Note that \(G_p\) is also assumed to be transitive, i.e., if \(i < j\) and \(j < k\) are two direct precedences specified, then the implied precedence relationship \(i < k\) holds, and arc \((i, k)\) is in \(G_p\) as well. Such an arc \((i, k)\) will be called a transition arc. We will denote the set of transition arcs by \(A_t\). Let \(G_p^t\) denote the graph \(G_p\) with all transition arcs removed. Hence, \(G_p^t\) contains no arc \((i, k)\) for which there exists an alternate path from \(i\) to \(k\) in \(G_p\).

Furthermore, let \(F(j)\) denote the set of nodes \(i \in \{2, \ldots, n\}\) for which there exists an arc \((j, i)\) in \(G_p^t\). \(F(j)\) is called the forward-star of node \(j\). Let \(R(j)\) denote the set of reverse-star nodes \(i \in \{2, \ldots, n\}\) in \(G_p\), such that there exists an arc \((i, j)\) in \(G_p\). Define \(F^*(j)\) and \(R^*(j)\) similarly with respect to \(G_p^t\). Finally, let \(f_j = |F(j)|\), \(r_j = |R(j)|\), \(f^*_j = |F^*(j)|\), and \(r^*_j = |R^*(j)|\) for all \(j \geq 2\), and for any \(S_1, S_2 \subseteq N\), \(S_1 \cap S_2 = \emptyset\), let \(x(S_1, S_2) \equiv \sum_{i \in S_1, j \in S_2} x_{ij}\).

For example, in Figure 1, which represents one \(G_p\) structure used for TSPLIB problem br17, \(A = \{(6, 8), (13, 15), (5, 2), (5, 3)\}\), \(F(5) = \{2, 16, 3, 10, 14\}\), \(F^*(5) = \{16, 10, 14\}\), \(R(3) = \{5, 16\}\), and \(R^*(3) = \{16\}\).

Recognizing the similarity between precedence ordering and the ordering structure of the MTZ subtour elimination constraints, our motivation is to exploit the structure provided by the two precedence graphs, \(G_p\) and \(G_p^t\), to tighten the linear programming relaxation of the precedence constrained ATSP (PCATSP). Using the concept of ATSP-MTZ2, the PCATSP version can be formulated as follows, where we have used the same set of variables \((x, u)\) as defined for ATSP-MTZ2.

PCATSP-MTZ2: Minimize \(\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}\)

subject to:

\[
\begin{align*}
    x & \in X_{ASSN}, \\
    u_j x_{ij} & = (u_i + 1) x_{ij} \quad \forall \; i, j \geq 2, i \neq j, (j, i) \notin G_p, \\
    u_j x_{ij} & = x_{ij} \quad \forall \; j = 2, \ldots, n, R^*(j) = \emptyset, \\
    u_j x_{ij} & = (n - 1) x_{ij} \quad \forall \; j = 2, \ldots, n, F^*(j) = \emptyset, \\
    u_j & \geq u_i + 1 \quad \forall \; (i, j) \in G_p^t, \\
    x_{ij} & = 0 \quad \forall \; (i, j) \in G_p^t, \\
    x_{ij} & = x_{ji} = 0 \quad \forall \; (i, j) \in A_t, \\
    x_{ij} & = 0 \quad \forall \; j = 2, \ldots, n, R^*(j) = \emptyset, \\
    x_{ij} & = 0 \quad \forall \; j = 2, \ldots, n, F^*(j) = \emptyset, \\
    r_j + 1 & \leq u_j \leq (n - 1) - f_j \quad \forall \; j \geq 2, \\
\end{align*}
\]

Constraint (27)–(30) are as in the ATSP-MTZ formulation, excluding null relationships. Constraints (32)–(35) prohibit transitions that are disallowed because of the given precedence relationships. Constraints (31) explicitly recognize that there should exist at least a one-step node labeling gap with respect to the \(u\)-variables in \(G_p^t\). Finally, constraints (36) explicitly represent the bounds on \(u_j\) that are implied by \(G_p\), and (37) states the binary restrictions on \(x\).
We will now apply RLT to PCATSP-MTZ2, generating suitable valid product constraints in the reformulation phase that create only $O(n^3)$ product terms that are inherent within (28)–(30). Then, in the linearization phase, we shall use the substitution given by (12), (13), and (14) as before, but additionally recognize that for any variable $x_{ij}$ that is restricted to be zero by (32) and (33), we have that the corresponding RLT variables $y_{ij}$ and $z_{ij}$ are also zero for all $i, j \geq 2$, $i \neq j$. The reformulation phase constructions (R1)–(R5) are stated below, followed by the resulting linear mixed-integer formulation of PCATSP, denoted PCATSD.

**R1.** Multiply the assignment constraints for each $i$ and $j$ with their corresponding variables $u_i$ and $u_j$, for all $i, j \geq 2$. This produces constraints (39) and (40), which are the same as (16) and (17).

**R2.** For each $j \geq 2$, construct the product of $(u_i - r_j - 1) \geq 0$ in (36) with (i) $x_{ij} \forall i \in R^*(j)$, (ii) $x_{ij} \forall i \neq j, i \geq 2$, $x_{ij} \neq 0$, (iii) $(1 - x_{ij}) \forall i \neq j, i \geq 2, (i, j) \notin A_s$, and (iv) $(1 - x_{ij})$ provided $R(j) \neq \emptyset$ and $F(j) = \emptyset$. Furthermore, construct the lifted product constraints (v) $(u_i - r_j - 2)x_{ij} \geq 0 \forall i \neq j, i \geq 2, i \notin R(j) \cup F(j)$, based on the case that if $x_{ij} = 1$, then we must have $u_j \geq r_j + 2$. Here, (i) produces (41), (ii) and (v) jointly produce (42) and (43), (iii) produces (47), and (iv) along with (13) produces (50). Moreover, product constraints of the types (i)–(v) for indices not contained herein are either null because of (32)–(35), or are implied by the ones considered herein as for ATSP-SD.

**R3.** Symmetric to (R2), for each $j \geq 2$, construct the product of $(n - 1 - f_j - u_j) \geq 0$ in (36) with (i) $x_{ij} \forall i \neq j, i \geq 2, x_{ij} \neq 0$; (ii) $x_{ij} \forall i \in F^*(j)$; (iii) $(1 - x_{ij}) \forall i \neq j, i \geq 2, (i, j) \notin A_s$, and (iv) $(1 - x_{ij})$ provided $F(j) \neq \emptyset$ and $R(j) = \emptyset$. Furthermore, construct the lifted product constraints (v) $(n - 2 - f_j - u_j)x_{ij} \geq 0 \forall i \neq j, i \geq 2, i \notin R(j) \cup F(j)$, based on the case that if $x_{ij} = 1$, then we must have $u_j \leq n - 2 - f_j$. Here, (i) together with (v) jointly produce (45) and (46), (ii) produces (44), (iii) produces (48), and (iv) produces (52). Note that in (44) we have reversed the indices $(i, j)$ to $(j, i)$ for consistency. Otherwise, (44) would have appeared as $y_{ji} \leq x_{ji}(n - 1 - f_i)$, for all $i \in F^*(j)$, $j \geq 2$.

**R4.** For each $j \geq 2$ such that $R(j) = \emptyset$, produce the lifted product constraint $(u_j - 1)(1 - x_{ij} - x_{ji}) \geq 0$, and similarly, for each $j \geq 2$ such that $F(j) = \emptyset$, produce the lifted product constraint $(n - 2 - u_i)(1 - x_{ij} - x_{ji}) \geq 0$. These are valid since, when $x_{ij} = x_{ji} = 0$, we must have $u_j \geq 2$ and $u_i \leq n - 2$, respectively. These products respectively produce the constraints (49) and (51).

**R5.** Finally, using (31), produce the lifted product constraints $(u_j - u_i - 2)(1 - x_{ij}) \geq 0 \forall (i, j) \in G_s^*$, based on the fact that if $x_{ij} = 0$ for any $(i, j) \in G_s^*$, then we must have $u_j \geq u_i + 2$. This produces constraints (53).

Applying the linearization (12), (13), and (14) as discussed above, and accommodating the restrictions (32)–(37), we obtain the proposed linear mixed-integer representation as given below. The validity of this model follows similar to the argument of Proposition 1.

**PCATSD:** Minimize $\sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_{ij}$

subject to:

\begin{align*}
\sum_{j \neq 1, j \neq i} y_{ij} + (n - 1)x_{ii} &= u_j \quad \forall i \geq 2, \quad (38) \\
\sum_{i \neq 1, i \neq j} y_{ij} + 1 &= u_j \quad \forall j \geq 2, \quad (39) \\
y_{ij} \geq r_j x_{ij} &\quad \forall i \in R^*(j), j \geq 2, \quad (40) \\
y_{ij} \geq x_{ij}[1 + \max(r_i, r_j)] &\quad \forall i \notin R(j) \cup F(j), \quad (i, j) \neq 2, \quad (41) \\
y_{ij} \geq x_{ij}[1 + r_i] &\quad \forall (i, j) \in G^*_s, \quad (42) \\
y_{ij} \leq x_{ij}[n - 1 - f_i] &\quad \forall j \in F^*(i), i \geq 2, \quad (43) \\
y_{ij} \leq x_{ij}[n - 2 - \max(f_i, f_j)] &\quad \forall j \notin R(i) \cup F(i), \quad (44) \\
y_{ij} \leq x_{ij}[n - 2 - f_j] &\quad \forall (i, j) \in G^*_s, \quad (45) \\
y_{ij} \leq y_{ji} &\quad u_j - (n - 1 - f_i)(1 - x_{ij}) + (n - 2 - f_j)x_{ij} \quad \forall (i, j) \in G^*_s, \quad (46) \\
y_{ij} \leq (r_j + 1) &\quad (n - 2 - r_j)x_{ij} \quad \forall j \geq 2, \quad (47) \\
y_{ij} \leq (r_j + 1) &\quad (n - 3)x_{ij} \quad \forall j \geq 2, \quad R(j) = \emptyset, \quad (48) \\
y_{ij} \leq (r_j + 1) &\quad (n - 2 - r_j)x_{ij} \quad \forall j \geq 2, \quad R(j) \neq \emptyset \quad \text{and} \quad F(j) = \emptyset, \quad (49) \\
y_{ij} \leq (n - 2) &\quad (n - 3)x_{ij} + x_{ji} \quad \forall j \geq 2, \quad F(j) = \emptyset, \quad (50) \\
y_{ij} \leq (n - 2) &\quad (n - 3)x_{ij} + x_{ji} \quad \forall j \geq 2, \quad F(j) \neq \emptyset \quad \text{and} \quad R(j) = \emptyset, \quad (51) \\
y_{ij} \leq u_i &\quad x_{ji} \quad \forall (i, j) \in G^*_s, \quad (52) \\
y_{ij} \leq u_i + x_{ji} &\quad 2 \quad \forall (i, j) \in G^*_s, \quad (53) \\
\text{constraints (32)–(37).} \quad (54)
\end{align*}

Similar to our comments for the ATSP, in a detailed branch-and-bound/cut algorithmic implementation for PCATSD based on the model PCATSD, the use of the aforementioned known classes of valid inequalities, as well as the use of selective higher-order representation constraints as discussed in §2 are of relevance. In addition, the following results develop further classes of valid constraints which can be used to augment the model PCATSD in order to further tighten its representation.

**Proposition 4.** The following classes of constraints are valid for PCATSD:

(a) $\sum_{j=2}^{n} u_j = n(n - 1)/2$ and $\sum_{j=5}^{n} u_j \geq |S|(|S| + 1)/2$ for all $S \subseteq \{2, \ldots, n\}$.

(b) $\sum_{j=i}^{n} u_j \geq \frac{f_j + \frac{(n-1)(n-2)}{2}}{2}(1 - x_{ij}) + \frac{(n+1)(n-2)}{2} x_{ij}$ for all $j \geq 2$ such that $R^*(j) = \emptyset$.
(c) \( \sum_{i \neq j} u_i \leq \left[ \frac{(n+1)(n-2)}{2} - r_j \right] (1 - x_{ij}) + \frac{(n-1)(n-2)}{2} x_{ij} \) for all \( j \geq 2 \) such that \( F^*(j) = \emptyset \).

**Proof.** Constraints (a) are based simply on the fact that the \( u_i \) values for \( j \geq 2 \) span the set \{1, 2, \ldots, \( n - 1 \)\} for any feasible solution. Similarly, for any \( j \geq 2 \) such that \( R^*(j) = \emptyset \) (else \( x_{ij} = 0 \)), constraint (b) asserts that

\[
\sum_{i \neq j} u_i \geq [1 + \cdots + (n+1) - (n - 1 - f_j)](1 - x_{ij})
\]

\[
+ [2 + \cdots + (n - 1)]x_{ij},
\]

based on the cases \( x_{ij} = 0 \) or 1. Likewise, for any \( j \geq 2 \) such that \( F^*(j) = \emptyset \) (else \( x_{ij} = 0 \)), examining the cases \( x_{ij} = 0 \) or 1, we get

\[
\sum_{i \neq j} u_i \leq [1 + \cdots + (n+1) - (1 + r_j)](1 - x_{ij})
\]

\[
+ [1 + \cdots + (n - 2)]x_{ij},
\]

which yields constraint (c). This completes the proof. \( \square \)

**Remark 1.** As evident from the proof of Proposition 4, constraints (a), (b), and (c) are also valid for ATSP whence \( r_j = f_j \equiv 0 \) for all \( j \geq 2 \). Furthermore, upper and lower bounds on \( \sum_{i \neq j} u_i \) can be likewise developed for cases (b) and (c), respectively, if necessary.

The node labeling scheme of the MTZ subtour elimination constraints inspires several new classes of valid inequalities for PCATSP when taken in conjunction with the cardinality of either the forward star \( F^*(j) \) or reverse star \( R^*(j) \) of a particular node \( j \geq 2 \). We present several such inequalities in what follows.

**Proposition 5.** The inequalities

\[
\sum_{i \in F^*(j)} u_i \geq f_j^* u_j + \frac{f_j^*(f_j^* + 1)}{2}
\]

\[
+ f_j^* \left[ 1 - \sum_{i \in F^*(j)} x_{ji} \right] \quad \forall j \geq 2 \quad f_j^* \geq 2 \quad (55)
\]

are valid for PCATSP.

**Proof.** Consider any node \( j \geq 2 \) such that \( f_j^* \geq 2 \) and examine its forward star \( F^*(j) \). Since each of the nodes contained in \( F^*(j) \) must appear after \( j \) in some order because of their precedence requirements, we get

\[
\sum_{i \in F^*(j)} u_i \geq (u_j + 1) + (u_j + 2) + \cdots + (u_j + f_j^*)
\]

\[
= f_j^* u_j + \frac{f_j^* (f_j^* + 1)}{2}. \quad (56)
\]

However, notice that if \( \sum_{i \in F^*(j)} x_{ji} = 0 \), then each \( u_i \) in \( F^*(j) \) would take on a value at least one unit greater in each feasible permutation of \( j \) and its successors. Therefore, we can lift (56) into (55). This completes the proof. \( \square \)

Note that in Proposition 5, when \( f_j^* \equiv 1 \), constraint (55) is simply (53) (with \( i \) and \( j \) interchanged).

**Proposition 6.** The inequalities

\[
\sum_{i \in R^*(j)} u_i \leq r_j^* u_j - \frac{r_j^* (r_j^* + 1)}{2}
\]

\[
- r_j^* \left[ 1 - \sum_{i \in R^*(j)} x_{ji} \right] \quad \forall j \geq 2, r_j^* \geq 2 \quad (57)
\]

are valid for PCATSP.

**Proof.** Similar to that for Proposition 5. \( \square \)

**Proposition 7.** Inequalities (55) and (57) obtained by replacing \( F^*(j) \), \( R^*(j) \), \( f_j^* \), and \( r_j^* \) with their corresponding counterparts in \( G_p \) are valid inequalities for PCATSP.

**Proof.** Follows from Proposition 5 and Proposition 6. \( \square \)

**Proposition 8.** For any node \( j \geq 2 \), examine \( R(j) \) and consider the subgraph \( G^*(j) \) of \( G^*_p \) induced by \( R(j) \cup \{j\} \). Select any node \( k \in G^*(j) \) that has more than one path in \( G^*(j) \) to node \( j \). Let \( P_{kj} \) represent the set of nodes lying on the union of these paths, including node \( k \), but not node \( j \), and let \( P_{kj} \) represent the set \( \{2, \ldots, n\} \setminus \{j\} \setminus P_{kj} \). Then, the following inequality is valid:

\[
u_j \geq u_k + |P_{kj}| + \sum_{p \in P_{kj}, q \notin P_{kj}} x_{pq}. \quad (58)
\]

**Proof.** Observe that in any feasible tour between nodes \( k \) and \( j \), we must have the nodes in \( P_{kj} \) strung in some order. This yields the first two terms on the right-hand side of (58). Moreover, if any \( x_{pq} = 1 \), with \( p \in P_{kj} \) and \( q \notin P_{kj} \), then node \( q \) also exists between \( k \) and \( j \), further separating \( k \) and \( j \) by one. This yields the final term in (58), and the proof is complete. \( \square \)

**Remark 2.** For (58), we can obtain an equivalent class of inequalities by symmetrically using \( F(j) \) in place of \( R(j) \), since only a pair of nodes is being examined at a time. Furthermore, note that if there exists a unique path from \( k \) to \( j \) in \( G^*(j) \), then (58) is implied by summing the corresponding inequalities (53).

**4. COMPUTATIONAL RESULTS FOR PCATSP-SD**

We conducted the following computational experiments with PCATSP-SD and PCATSP-DL to study the effect of precedence structures on the lower bounds produced by solving these two formulations, as well as to identify specific precedence structures that impose a greater challenge. For this initial experiment, we included all the valid inequalities of Propositions 4–8 in the PCATSP-SD formulation. PCATSP-DL is defined by the constraints: \{(4), (27), (31)–(37), (49)–(52)\}. Again, test cases were run on a Dell XPS 450 computer having 328 Mb RAM, under the NT 4.0 operating system, using GAMS as the modeling front-end, and using CPLEX 6.0 as the MIP solver with the system parameter settings (optfile) described earlier as well as with the default CPLEX MIP settings.
Table 4. Precedence orderings for TSPLIB problems.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Precedence Orderings</th>
</tr>
</thead>
<tbody>
<tr>
<td>esc07</td>
<td>2 \prec_4 \prec_5, 2 \prec_6 \prec_5, 2 \prec_7 \prec_5</td>
</tr>
<tr>
<td>esc11</td>
<td>9 \prec_2 \prec_3 \prec_5 \prec_8, 6 \prec_3, 7 \prec_3, 7 \prec_10, 11 \prec_4</td>
</tr>
<tr>
<td>esc12</td>
<td>2 \prec_6, 3 \prec_6, 4 \prec_6, 6 \prec_12, 7 \prec_12, 8 \prec_12, 5 \prec_9, 10 \prec_9</td>
</tr>
<tr>
<td>esc25</td>
<td>6 \prec_2, 19 \prec_11 \prec_2, 12 \prec_2, 12 \prec_6, 12 \prec_8, 16 \prec_2, 25 \prec_5, 15 \prec_13, 22 \prec_17</td>
</tr>
<tr>
<td>esc25a</td>
<td>Same as for esc25, plus 2 \prec_13, 2 \prec_15, 8 \prec_17, 8 \prec_22</td>
</tr>
</tbody>
</table>

We first examined the bound produced by the three formulations when applied to five sequential ordering problems from the TSPLIB collection. Table 4 displays the precedence structure inherent in each problem. Note that $G_p$ includes these precedences plus the various implied transitive precedence relationships. Table 5 presents the lower bound ($Z_{LP}$) obtained at the initial node, the total number of nodes processed (Nodes) to obtain a confirmed optimal $Z_{LP}$ value, the total cpu seconds used (CPU) to solve the problem, and the solver settings used for each experimental run.

It is encouraging to note that the specialized formulation PCATSP-SD consistently provided a tighter lower bound than PCATSP-DL for all problems, thereby computationally confirming the theoretical results established earlier. These tightened bounds resulted in the enumeration of significantly fewer nodes overall, particularly as problem size increases. For problem esc07, PCATSP-SD and PCATSP-DL solved this instance at the initial node itself via their LP relaxations. For problems esc11, esc12, esc25, and esc25a, PCATSP-SD displayed a consistent pattern of improvement over PCATSP-DL with respect to the number of nodes enumerated. This dimension of performance translates directly into a capability for the formulation to tackle larger problems because fewer unexplored nodes are required to be maintained in the branching structure of a branch-and-bound/cut implementation. At the same time, however, note that PCATSP-SD required CPLEX to work harder at each node as indicated by the relatively higher overall cpu time consumed.

It is worth noting that problem esc25 provided an exception to the foregoing general pattern. In this case, the default CPLEX MIP settings induced significantly worse performance results for PCATSP-DL both in terms of cpu time and the nodes processed. The difference between the initial lower bounds obtained by PCATSP-SD and PCATSP-DL would not have predicated such a performance differential. Evidently, the tighter formulation PCATSP-SD responds better to the branching process, producing relatively stronger lower bounds.

We conducted a separate set of experiments using esc11 to evaluate the effectiveness of the valid inequalities of Propositions 4–8 by varying the set of constraints that are appended to the PCATSP-SD formulation. Simultaneously, we wanted to examine the sensitivity of this effectiveness to the CPLEX options settings chosen. Three CPLEX options settings were used: optfile (the settings described earlier), the default CPLEX settings without any branching priorities set on the variables (Def w/o), and the default CPLEX settings with branching priorities set the same as in the case of optfile (Def w) mentioned earlier (see §2). Table 6 presents the results obtained. Here, P4(b), for example denotes the set of valid inequalities (b) from Proposition 4, and so on.

A comparison between the two extreme cases in which none and all of the propositional constraints were appended to the PCATSP-SD formulation, shows that these additional constraints do not have a beneficial effect—the number of nodes processed increases (presumably due to altered branching decisions) and cpu time increases. We therefore set out to determine if a minimal set of these propositional constraints could be included with PCATSP-SD in order to realize both a lower bound tightening and a reduction in the nodes processed.

When constraints were introduced by classes, constraint set P4(b) and P5 had the greatest impact on reducing the number of nodes as well as cpu time, whereas constraint set P7(b) achieved the best additional tightening of the lower bound (yet adversely affected the number of nodes enumerated). We next examined the effects of including both P4(b)
Table 6. Effectiveness of the valid inequalities from Propositions 4–8 on PCATSP-SD.

<table>
<thead>
<tr>
<th>Problem</th>
<th>Constraint</th>
<th>( Z_{IP} )</th>
<th>Nodes</th>
<th>CPU</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>esc11</td>
<td>none</td>
<td>3,464.6</td>
<td>10</td>
<td>0.36</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>15</td>
<td>0.40</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>10</td>
<td>0.35</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>all</td>
<td>3,465.7</td>
<td>10</td>
<td>0.76</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.7</td>
<td>38</td>
<td>0.62</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.7</td>
<td>13</td>
<td>0.41</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P4(b)</td>
<td>3,464.6</td>
<td>7</td>
<td>0.34</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>7</td>
<td>0.33</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>7</td>
<td>0.34</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P4(c)</td>
<td>3,464.6</td>
<td>10</td>
<td>0.41</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>13</td>
<td>0.43</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>10</td>
<td>0.41</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P5</td>
<td>3,464.6</td>
<td>9</td>
<td>0.34</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>10</td>
<td>0.32</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>9</td>
<td>0.34</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P6</td>
<td>3,464.6</td>
<td>21</td>
<td>0.57</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>46</td>
<td>0.73</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>21</td>
<td>0.55</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P7(a)</td>
<td>3,464.6</td>
<td>10</td>
<td>0.34</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>12</td>
<td>0.33</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,464.6</td>
<td>10</td>
<td>0.33</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P7(b)</td>
<td>3,465.5</td>
<td>22</td>
<td>0.53</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.5</td>
<td>28</td>
<td>0.43</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.5</td>
<td>22</td>
<td>0.52</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P4(b), P7(b)</td>
<td>3,465.7</td>
<td>11</td>
<td>0.38</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.7</td>
<td>22</td>
<td>0.40</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.7</td>
<td>11</td>
<td>0.34</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P4(b), P7(b), P5</td>
<td>3,465.7</td>
<td>7</td>
<td>0.33</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.7</td>
<td>22</td>
<td>0.40</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3,465.7</td>
<td>10</td>
<td>0.34</td>
<td>def w/o priority</td>
</tr>
<tr>
<td>esc25</td>
<td>none</td>
<td>1,359.4</td>
<td>1,054</td>
<td>251.2</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,359.4</td>
<td>2,135</td>
<td>373.3</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,359.4</td>
<td>1,054</td>
<td>260.5</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td>P4(b), P7(b), P5</td>
<td>1,359.4</td>
<td>930</td>
<td>210.7</td>
<td>optfile</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,359.4</td>
<td>3,342</td>
<td>398.4</td>
<td>def w/o priority</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1,359.4</td>
<td>930</td>
<td>216.5</td>
<td>def w/o priority</td>
</tr>
</tbody>
</table>

and P7(b) with the idea of appending both a node reducing set and a bound tightening set. As can be seen in the results of Table 6, this produced a further improvement in the lower bound, while drawing closer to the case of using P4(b) alone with respect to number of nodes processed. However, by further incorporating P5 with this combination and using the optfile settings, we realized the tightest initial lower bound, the fewest number of nodes processed, and a near-best cpu time measure as well.

For the larger problem esc25, similar encouraging results were obtained with regard to both the cpu time and the number of nodes processed for the same combination of valid inequalities (see Table 6). It was interesting to note that for this problem instance, PCATSP-SD had already achieved the tightest lower bound; yet, appending constraints P4(b), P7(b), and P5 significantly reduced both the number of nodes processed and the overall cpu time under the optfile setting. Moreover, this latter setting appears to be the most favorable one over the three tested options, with the default setting using the priority branching rule being second best.

Finally, we created a new 25-city problem (esc25b) with the precedence structure shown in Table 7, and had both the PCATSP-SD and the PCATSP-DL formulations attack this problem using the optfile settings. This special precedence structure is motivated by modeling situations in which several cities that are geographically separated by large distances (perhaps in a rural setting) are required to be visited by a salesman. The order of visitation between these cities is purely a matter of minimizing cost, and not adhering to preference priorities. However, once the salesman arrives in a particular city, there exists a strong priority ordering for intracity site visitations that must be adhered to. This is an interesting modeling construct having broad applications.
outside of the obvious pick-up and delivery scenarios. For example, consider the case of nonintrusive medical diagnoses in which a physician can choose to conduct various diagnostic tests on a patient in any order. However, once a particular diagnostic test begins, there exists a sequence of steps to be followed that has a required preference ordering. Alternatively, in job shop scheduling applications, there can exist various assembly stations that must be visited by the work-in-progress in order to complete a product assembly. When these stations have very loose, or nonexistent priority orderings between the tasks, but each individual station’s subtasks have a well-defined precedence ordering, we again observe the same structure.

After both processes were allowed to run for approximately 64,000.0 cpu seconds, well into the “tailing off” effect region that typically arises in this class of problems, we stopped both runs, resulting in the data shown in Table 8. Although neither formulation closed the LP-IP gap, PCATSP-SD outperformed PCATSP-DL in all categories. At an intermediate point 14,372.6 cpu seconds into the run, the best found integer solution of PCATSP-DL was 6,900 with a corresponding tree size of 103.11 MB, whereas at the same point, PCATSP-SD identified a significantly better integer solution of 3,528 with a corresponding tree size of only 10.18 MB. At the stopping point whose data is presented in Table 8, this performance difference persisted: The enumeration tree for PCATSP-DL increased to 380.72 MB, representing 806,078 unexplored nodes, whereas the enumeration tree for PCATSP-SD grew to just 65.45 MB with a corresponding 99,407 unexplored nodes. For this problem instance, the benefit realized by the stronger PCATSP-SD formulation causing CPLEX to work harder at each node is apparent: a higher quality initial lower bound, a better integer solution, less nodes processed overall, a smaller enumeration tree, and a greater degree of gap closure than that achieved by PCATSP-DL. Investigation into the effectiveness of PCATSP-SD on a larger class of problems possessing this particular precedence structure is recommended for future research.

5. CONCLUSIONS

The objective of this paper was to present new, tightened formulations for two classes of the asymmetric traveling salesman problem, both of which exploit special structures afforded by the Miller-Tucker-Zemlin (1960) subtour elimination constraints. The MTZ formulation presented an interesting challenge in this vein as many researchers, most recently Orman and Williams (1999), have demonstrated that the MTZ subtour elimination constraints consistently yield the weakest formulation of all those available. Applying the specialized version of the Reformulation-Linearization Technique (RLT) of Sherali et al. (1998) directly to the MTZ constraints yielded the formulation ATSP-SD, which was shown to both theoretically as well as computationally dominate the bounds afforded by the lifted-MTZ formulation ATSP-DL of Desrochers and Laporte (1991).

The similarity between precedence ordering and the sequential ordering of the MTZ subtour elimination constraints provided the insight necessary to extend our RLT approach to the precedence constrained ATSP. In doing so, we developed the tightened formulation PCATSP-SD, which was shown to again generate significantly tightened lower bounds. In addition, we have introduced several new classes of valid inequalities for both the ATSP and the PCATSP, and have identified their relative effectiveness for solving problems when embedded in the PCATSP-SD formulation.

For both new formulations, we have presented encouraging preliminary computational results that exhibit the effective tightening of the lower bounds achieved by these formulations, resulting in a significant decrease in the number of nodes enumerated across the problem instances solved. This represents an important result in that the fewer the number of nodes enumerated in a search strategy, the smaller the size of the resulting branching structure, and the lesser the computer memory requirements for storing node-specific information. We recognize that because of the additional effort involved in solving the somewhat larger and ill-structured relaxations produced by these new model formulations, the overall cpu effort does not at present usually appear competitive with respect to the alternative weaker formulations. However, this situation could easily change in the future with advances in LP technology, or through the application of quick Lagrangian dual ascent schemes, or via new cut-generation strategies based on the classes of valid inequalities presented herein. Nonetheless, for certain more challenging instances, the proposed tighter formulation resulted in enumerating fewer nodes as well as consumed lesser cpu time. When such instances were prematurely terminated, it also produced significantly better quality incumbent solutions. Research into the computational benefits of further exploiting the structure of the precedence orderings, incorporating cuts generated from

<table>
<thead>
<tr>
<th>Problem</th>
<th>Precedence Orderings</th>
</tr>
</thead>
<tbody>
<tr>
<td>esc25b</td>
<td>3 &lt; 4 &lt; 5 &lt; 6 &lt; 7 &lt; 8, 13 &lt; 14 &lt; 15 &lt; 16</td>
</tr>
<tr>
<td></td>
<td>19 &lt; 20 &lt; 21 &lt; 22 &lt; 23</td>
</tr>
</tbody>
</table>

Table 7. Upper triangular precedence structure for esc25b.

For each problem instance, the proposed PCATSP-SD formulation resulted in enumerating fewer nodes as well as achieved a significantly better integer solution and lesser cpu time. When such instances were prematurely terminated, it also produced significantly better quality incumbent solutions. Research into the computational benefits of further exploiting the structure of the precedence orderings, incorporating cuts generated from

<table>
<thead>
<tr>
<th>Problem</th>
<th>Formulation</th>
<th>Z_{LP}</th>
<th>Best Found</th>
<th>Nodes</th>
<th>CPU</th>
<th>Abs Gap</th>
<th>Rel Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td>esc25b</td>
<td>PCATSP-SD</td>
<td>2,023.9</td>
<td>3,528</td>
<td>161,786</td>
<td>63,706.2</td>
<td>283.2</td>
<td>0.08</td>
</tr>
<tr>
<td></td>
<td>PCATSP-DL</td>
<td>1,925.1</td>
<td>4,311</td>
<td>1,339,899</td>
<td>64,103.9</td>
<td>1,483.5</td>
<td>0.52</td>
</tr>
</tbody>
</table>

Table 8. Lower bounds and performance statistics on esc25b.
these new classes of valid inequalities, and of embedding these tightened formulations into other classes of routing and scheduling problems are recommended for future studies.

ACKNOWLEDGMENTS

This material is based upon work supported by the National Science Foundation under grants DMI-9521398 and DMI-9812047, and by the Air Force Office of Scientific Research under grant F49620-96-1-0274. This research was also supported by the Mathematical Sciences Center of Excellence at the United States Military Academy at West Point, New York.

REFERENCES


Driscoll, P. J. 1995. A new hierarchy of relaxation for 0–1 mixed integer problems with application to some specially structured problems. Doctoral dissertation, Department of Industrial and Systems Engineering, Virginia Polytechnic Institute, Blacksburg, VA.


——, ——. 1996. Computational advances on using the reformulation-linearization technique (RLT) to solve various discrete and continuous nonconvex programming problems. OPTIMA 49 1–7.

