Traveling Salesman Problem Formulations with $N \log N$ Number of Binary Variables

Thomas A. Pogiatzis[a], *Vassilios S. Vassiliadis*[a]*, *Raul Conejeros*[b]

[a]Department of Chemical Engineering and Biotechnology, University of Cambridge, Pembroke Street, Cambridge CB2 3RA, UK.
[b]Escuela de Ingenieria Bioquimica, Pontificia Universidad Catolica de Valparaiso, Av. Brasil 2147, Valparaiso, Chile.

Abstract

This paper presents a novel formulation for the Traveling Salesman Problem (TSP), utilizing a binary list data-structure allocating cities to its leaves to form sequentially the tour of the problem. The structure allows the elimination of subtours from the formulation and at the same time reducing the number of binary variables to $\mathcal{O}(N \log_2 N)$. The expense is the increase in the constraint set cardinality measuring at $\mathcal{O}(N^2 \log_2 N)$, and in the introduction of at most $\mathcal{O}(N^2 \log_2 N)$ continuous variables. The value of the proposed original formulation is that for the first time a minimal number of binary degrees of freedom is recognized for the TSP. Although computationally the resulting TSP formulation is not competitive, this work presents a new methodology of structuring the information describing the problem which may lead to future developments exploiting it. The scheme is equivalent to binary expansion of tour-locations which may be applicable to other standard TSP formulations, thus allowing there also the same reduction in the number of binary variables.

Keywords:

Traveling Salesman Problem, mixed integer-linear programming, binary list, subtour elimination

1 Introduction

The Traveling Salesman Problem is a well-studied central problem in optimization theory. Mathematical Programming formulations of the problem are among others the following: Miller et al. (1960), Gavish and Graves (1978)and Claus (1984).

Our formulation is an assignment type one, following the approach of Millar and Cyrus (2000), admitting only complete tours as solutions.

The rest of our paper is organized as follows. At first, in Section 2 we lay the basics of the novel formulation and in Section 3 we present the formulation for Manhattan distance

*Corresponding Author: vsv20@cam.ac.uk

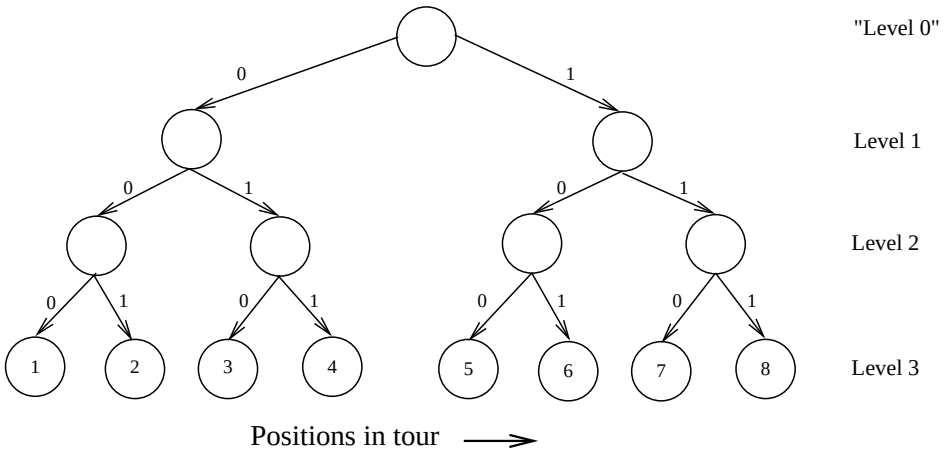

Fig. 2.1: 8 node tree structure

problems. Moving on, in Section 4 we lay down the problem for the Asymmetric Traveling Salesman Problem (ATSP) and in Section 5 we present the new formulation. Finally, in Section 6 we examine the validity of the proposed formulations by solving some small problems and in Section 7 we present conclusions and proposed future work.

2 Basics of the new formulation

This number of cities, without of loss of generality as demonstrated in a later section, is assumed to be precisely given by an integer transformation of the form:

$$N = 2^{NL} \tag{2.1a}$$

which equivalently is precisely

$$NL = \log_2 N \tag{2.1b}$$

In case N is not an exact integer power of 2, NL is computed by:

$$NL = \lceil \log_2 N \rceil \tag{2.2}$$

NL should thus be replaced appropriately from equation (2.2) in the following sections, particularly in the calculation of the number of constraints and variables appearing in the model. Terms containing $\log_2 N$ should be replaced by $\lceil \log_2 N \rceil$.

The ordering of cities is based on a sequencing that places cities at the leaves of a binary list constructed precisely so that it has equal number of nodes at its last layer (leaves) to the number of cities.

The above leads naturally to a binary tree representation which for $NL = 3$ *i.e.* $N = 2^3 = 8$ objects is shown in Figure 2.1:

Thus for NL Layers, the last layer ($l = NL$) is the one that contains the raw objects that are to be linked. It is clear that one should keep track of the *left-right* direction from the head of the tree that an object "travels" to be allocated to the appropriate most elementary pairing node at level NL.

In particular, this scheme is equivalent to repeated partitions of the set of all cities N into left-right orientation from Level 1 till level NL. The properties of the partition are

such that exactly half of the objects are in the left and half in the right orientation. We begin by allocating a variable $r_{il} \in \{0, 1\}$ such that:

$$r_{il} = \begin{cases} 0 & \text{if city is left allocated} \\ 1 & \text{if city is right allocated} \end{cases} \tag{2.3}$$

$$l = 1, 2, \ldots, NL; \quad i = 1, 2, \ldots, N$$

At each level l of the tree, the group of cities is allocated a left and right orientation. Only, and exactly, half of the cities are of left routing and half of right routing at each such level. Hence this leads to the following set of constraints:

$$\sum_{i=1}^{N} r_{il} = \frac{N}{2}; \quad l = 1, 2, \ldots, NL \tag{2.4}$$

It is noted however that this partitioning is not sufficient to allocate uniquely a binary string to a city. The nodes have to be examined sequentially and depending on their parents, sub-allocations have to be made so that nodes emanating from the same parent must split equally between the value of 0 and 1. A simpler way is adopted in the next section by introduction of the city to position allocation variables z_{ik}. Constraints (2.4) may still be used to tighten the feasible region of the optimization problems.

In the case that N is not an exact integer power of 2, equations (2.4) need to be replaced by:

$$\sum_{i=1}^{N} r_{il} = \sum_{k=1}^{N} t_{kl}; \quad l = 1, 2, \ldots, NL \tag{2.5}$$

Parameters t_{kl} are calculated by Algorithm 1 presented in the next section.

Finally it is noted that the position of city i, designated by variables POS_i, is given by:

$$POS_i = 1 + \sum_{l=1}^{NL} 2^{NL-l} \cdot r_{il}; \quad i = 1, 2, \ldots, N \tag{2.6}$$

To facilitate allocation of each city to a tour location, new continuous variables z_{ik} are defined such that they are forced-binary by the variables r_{il}. Index i is associated with the city, and index k with the leaf on the binary tree.

To facilitate notation we introduce the following functions:

$$\alpha(t) = \begin{cases} +1 & \text{if} \quad t = 0 \\ 0 & \text{if} \quad t = 1 \end{cases} \tag{2.7a}$$

$$\beta(t) = \begin{cases} -1 & \text{if} \quad t = 0 \\ +1 & \text{if} \quad t = 1 \end{cases} \tag{2.7b}$$

The binary tree leaves are allocated a binary string describing the routing the r_{il} variables indicate to reach each one of them. To compare which object i is allocated to

Algorithm 1 Nodal binary string analysis

```
for k = 1 to N do
temp = k − 1
for l = NL to 1 do
t_kl = temp mod 2
temp = ⌊temp/2⌋
end for
end for
```

which node k through the object i set of variables r_{il} for $l = 1, 2, \ldots, NL$, we define the following "target" binary strings for each leaf-node according to Algorithm 1.

With the above, the following constraints are written to define the values of the induced binary variables z_{ik}:

$$z_{ik} \leq \alpha(t_{kl}) + \beta(t_{kl}) \cdot r_{il}; \quad l = 1, 2, \ldots, NL \tag{2.8a}$$

$$z_{ik} \geq 1 - \sum_{l=1}^{NL} (\alpha(1 - t_{kl}) + \beta(1 - t_{kl}) \cdot r_{il}) \tag{2.8b}$$

$$i = 1, 2, \ldots, N; \quad k = 1, 2, \ldots, N$$

There are $(NL + 1) \cdot N^2 = (\log_2 N + 1) \cdot N^2$ constraints in the equations above.

To guarantee uniqueness of the allocation of each position binary string to exactly one city, similar to the work of Millar and Cyrus (2000), we need to include the uniqueness of the allocation of each city to each position and vice versa:

$$\sum_{i=1}^{N} z_{ik} = 1; \quad k = 1, 2, \ldots, N \tag{2.8c}$$

$$\sum_{k=1}^{N} z_{ik} = 1; \quad i = 1, 2, \ldots, N \tag{2.8d}$$

There are $2N$ constraints defined in the equations above.

3 The Manhattan distance case formulation

For the Manhattan distance case each city is described by a pair of coordinates:

$$x_i^{(0)}, y_i^{(0)}$$

Each of the nodes of the last layer (leaves) of the binary tree is allocated accordingly an (x, y) coordinate pair, depending on which city was routed to that leaf:

$$x_k = \sum_{i=1}^{N} x_i^{(0)} \cdot z_{ik} \tag{3.1a}$$

$$y_k = \sum_{i=1}^{N} y_i^{(0)} \cdot z_{ik} \tag{3.1b}$$

$$k = 1, 2, \ldots, N$$

There are $2 \cdot N$ equations in the set above.

The computation of city-to-city distances requires the use of two variables per leaf of the binary list, namely $l_j^{(x)}$ and $l_j^{(y)}$.

$$-l_k^{(x)} \leq \quad x_k - x_{k+1} \quad \leq +l_k^{(x)} \tag{3.2a}$$

$$-l_k^{(y)} \leq \quad y_k - y_{k+1} \quad \leq +l_k^{(y)} \tag{3.2b}$$

$$k = 1, 2, \ldots, (N-1)$$

$$-l_N^{(x)} \leq \quad x_N - x_1 \quad \leq +l_N^{(x)} \tag{3.2c}$$

$$-l_N^{(y)} \leq \quad y_N - y_1 \quad \leq +l_N^{(y)} \tag{3.2d}$$

There are $4 \cdot N$ inequalities in the set above.

It is abundantly clear that from the uniqueness of the allocations z_{ij} (one-to-one city-to-node), and from the fact that distances are taken consecutively node-to-node at the last layer of the binary list, that the above procedure does not permit the existence of subtours.

3.1 Symmetry breaking

Consider the case where a solution (integer feasible) contains the following tour of cities:

$$1 \to 3 \to 4 \to 2 \to 1$$

Where it appears that city 1 is the beginning of the route. This is placed readily on the proposed binary list structure leaves; however there is a problem of alternative solutions. (1,3,4,2,1) is equivalent in distance in (3,4,2,1,3) *etc.* To avoid this situation, and observing that every city is contained precisely once in any tour, the first node (leaf 1) of the binary data-structure is fixed to be pointed to by city 1 (in fact any city would do):

$$r_{1,l} = 0,\ l = 1, 2, \ldots, NL \tag{3.3}$$

There are NL equalities defined above.

Another source of symmetry is the fact that a tour may be traversed either clockwise or counter-clockwise, even for the fixed first node case. So for example (1,2,3,4,1) is the same as (1,4,3,2,1). To break this case as well, we enforce that the *index* of the city placed in node $N-1$ be larger (or smaller) than the index of the city placed in node 2. To extract such information the following constraint is used:

$$\sum_{i=1}^{N} i \cdot z_{i,2} \leq \sum_{i=1}^{N} i \cdot z_{i,(N-1)} \tag{3.4}$$

This is a single inequality constraint, using the object indices as weights for discrimination.

3.2 Objective function

The objective function is the last and simplest item to add to our formulation.

$$\min L = \sum_{k=1}^{N} \left(l_k^{(x)} + l_k^{(y)} \right) \tag{3.5}$$

3.3 Additional tightening constraints

Additional tightening constraints, particularly for the case where N is not an exact power of 2 are introduced by limiting the number of the position allocated to each city i as follows:

$$1 \leq \sum_{l=1}^{NL} 2^{NL-l} \cdot r_{il} \leq N-1; \quad i = 2, 3, \ldots, N \tag{3.6}$$

These arise from the restriction that the position of a city can range from 2 to N (since city 1 is fixed to position 1), such that $2 \leq POS_i \leq N$. There are $2(N-1)$ such constraints.

4 The general Asymmetric Traveling Salesman Problem (ATSP)

The case where the distances are given by a general matrix, including the Asymmetric Traveling Salesman Problem (ATSP), necessitates the use of binary variables x_{ij}[1], $i, j = 1, 2, \ldots, N$, indicating the presence of an arc (i, j) in the optimal tour. The objective function of the problem is thus given by:

$$\min \sum_{i=1}^{N} \sum_{j=1}^{N} c_{ij} \cdot x_{ij} \tag{4.1}$$

Similar to the use of allocation variables of cities to the position in the tour presented by Millar and Cyrus (2000), we may use the city-position variables z_{ik}, defined in equations (2.8a) and (2.8d) to force the variables x_{ij} to be 1 if a corresponding arc (i, j) is present. This yields the following constraints:

$$z_{ik} + z_{j,k+1} - 1 \quad \leq \quad x_{ij}; \quad k = 1, 2, \ldots, (N-1) \tag{4.2a}$$

$$z_{i,N} + z_{j,1} - 1 \quad \leq \quad x_{ij} \tag{4.2b}$$

$$i = 1, 2, \ldots, N; \quad j = 1, 2, \ldots, N$$

Equations (4.2a) and (4.2b) force variables x_{ij} to take the value of one if arc (i, j) is in the tour. Else the lower bound relaxes. Because of the objective minimization in equation (4.1), assuming all the c_{ij} are positive numbers, the variables x_{ij} are naturally

[1] These variables are not to be confused with the variables $x_i^{(0)}$ and x_k used to represent city coordinates in the case of the Manhattan distance TSP problem defined in earlier sections.

driven to their lower bound of zero if the constraints added do not enforce a value of 1. Hence variables x_{ij} do not need to be binary, only continuous with bounds $0 \leq x_{ij} \leq 1$.

This formulation requires thus the introduction of N^2 variables and N^3 additional enforcing constraints.

The number of the enforcing constraints in equations (4.2a) and (4.2b) is thus going to be the issue examined next, in order to see if an alternative formulation may be employed that does not require such a large number of them ($\mathcal{O}(N^3)$).

4.1 Conditions of adjacency with city to position allocation

Using our binary tree location fixing, we utilize variables $r_{il} \in \{0, 1\}$ as in section 2. Consider two cities i and j for which the positions in the tour are given as functions of the r_{il} and r_{jl} variables:

$$POS_i = 1 + \sum_{l-1}^{NL} 2^{NL-l} \cdot r_{il} \tag{4.3}$$

$$POS_j = 1 + \sum_{l-1}^{NL} 2^{NL-l} \cdot r_{jl} \tag{4.4}$$

The above equations are simply converting the binary string (number) associated with each position to decimal base. For the cities to be adjacent, hence for arc (i, j) to be in the tour, we must have:

$$POS_j - POS_i = 1 \tag{4.5}$$

or

$$\sum_{l-1}^{NL} 2^{NL-l} \cdot (r_{jl} - r_{il}) = 1 \tag{4.6}$$

Clearly what we are seeking is that if condition in equation (4.6) holds, then the corresponding x_{ij} variable must be forced to 1, else it is left loose within its bounds:

$$\sum_{l-1}^{NL} 2^{NL-l} \cdot (r_{jl} - r_{il}) = \begin{cases} 1, & x_{ij} = 1 \\ \neq 0, & 0 \leq x_{ij} \leq 1 \ \text{(or set to 0)} \end{cases} \tag{4.7}$$

What we are after is a lower number of constraints derived by the indices i, j, l such that their number is given by $N^2 f(NL)$ where the product resulting from the inclusion of $f(NL)$ is less than N^3. It is noted that $NL = \log_2 N$.

4.1.1 Adjacency of binary numbers: motivation

The strings represented by the variables r_{il} and r_{jl} for $l = 1, 2, \ldots, NL$ for the two cities i and j may be thought of as integer numbers in binary format. If we consider two consecutive numbers in binary representation we find the following:

For string $r_i \in \{0, 1\}^{NL}$ we consider the number encoded, NUM_i, to be given by the following equation:

$$NUM_i = \sum_{l=1}^{NL} 2^{NL-l} \cdot r_{il} \tag{4.8}$$

With NL bits we thus encode numbers $0 \leq NUM_i \leq 2^{NL} - 1$. It is noted that highest power of 2 corresponds to level $l = 1$ and the lowest power of 2 corresponds to level $l = NL$.

Any number NUM_i in this range is analyzed such that we locate the lowest power bit that is equal to 0, corresponding to some level l'. In other words we demand that:

$$l' = \max_{l=1,2,\ldots,NL} l \text{ such that } r_{il} = 0 \tag{4.9}$$

For l in the range $l' < l \leq NL$ the definition of equation (4.9) implies that $r_{il} = 1$, unless $l' = NL$ in which case there are no higher level bits. This is depicted in Figure 4.1.

The next number to NUM_i is obtained by adding 1 to it, so that in terms of its binary digits, all digits l in the range $l' < l \leq NL$ change to the value of 0 from 1, while the digit in level l' changes from 0 to the value of 1. Lower level digits, *i.e.* for $1 \leq l < l'$ remain unchanged and equal to their previous values.

It is noted that the range for NUM_i in Figure 4.1 is $0 \leq NUM_i \leq 2^{NL} - 2$. This is so as the case of $NUM_i = 2^{NL} - 1$ by addition of 1 would require one more bit to represent the resulting number. The proof in the following section is based on these observations.

4.1.2 Condition of adjacency derived from the binary string representation

Looking at the binary tree allocation of positions and the relationship between positions in the tour (leaves in the tree), such as in Figure 2.1, the following holds:

Theorem 1. *Given two cities i and j and the binary representation of their position in the tour by variable sets $r_{il}, r_{jl} \in \{0, 1\}$ with $l = 1, 2, \ldots NL$, then if and only if the cities are allocated adjacently such that the position of city j is greater by 1 from the position of city i (i.e. cities i and j are in consecutive positions), the following properties hold:*

Property A:

There exists exactly one and only one l', with $l' = 1, 2, \ldots, NL$, such that

$$r_{il'} = 0 \text{ and } r_{jl'} = 1 \tag{4.10}$$

Property B:

For $1 \leq l < l'$ $(l = 1, 2, \ldots, (l' - 1))$

$$r_{il} = r_{jl} \text{ (either both 0, or both 1)} \tag{4.11}$$

Property C:

For $l' < l \leq NL$ $(l = (l' + 1), (l' + 2), \ldots, NL)$

$$r_{il} = 1 \text{ and } r_{jl} = 0 \tag{4.12}$$

The converse also holds: if the three properties do not hold simultaneously for a pair of cities i and j then the cities are not adjacent in the tour, such that arc (i, j) is not present in the tour.

$Level\ l$	1	2	...	$l'-1$	l'	$l'+1$	...	$NL-1$	NL
Power of 2	2^{NL-1}	2^{NL-2}	...	$2^{NL-(l'-1)}$	$2^{NL-l'}$	$2^{NL-(l'+1)}$	...	2^1	2^0
NUM_i	$r_{i,1}$	$r_{i,2}$	...	$r_{i,(l'-1)}$	0	1	...	1	1
NUM_i+1	$r_{i,1}$	$r_{i,2}$	...	$r_{i,(l'-1)}$	1	0	...	0	0

Fig. 4.1: Binary number representation of number NUM_i in bits and addition of 1 to it.

Lemma. *The integer numbers NUM_i and NUM_j are described by a binary string. If $NUM_j > NUM_i$ then there exists an l' such that*

$$l' = \min_{l=1,2,..,NL} l$$

for which the following hold

$$r_{il'} = 0,\ r_{jl'} = 1$$
$$r_{il} = r_{jl},\ l = 1, 2, ..., l' - 1$$

The following formulae hold and are going to be used in the proof.

$$\sum_{i=1}^{k} 2^{k-i} = 2^k - 1 \tag{4.13}$$

$$\sum_{i=1}^{m} 2^{k-i} = 2^k - 2^{k-m} \tag{4.14}$$

$$\sum_{i=m+1}^{k} 2^{k-i} = \sum_{i=1}^{k} 2^{k-i} - \sum_{i=1}^{m} 2^{k-i} = 2^k - 1 - (2^k - 2^{k-m}) = 2^{k-m} - 1 \tag{4.15}$$

Proof. The integer numbers NUM_i and NUM_j, were $NUM_j > NUM_i$, are described by a binary string. Assuming that Properties A, B & C hold simultaneously then

$$NUM_j - NUM_i = \sum_{l=1}^{l'-1} 2^{NL-l}(r_{jl} - r_{il}) - 2^{NL-l'}(r_{jl} - r_{il}) + \sum_{l=l'+1}^{NL} 2^{NL-l}(r_{jl} - r_{il}) \Rightarrow$$

$$\Rightarrow NUM_j - NUM_i = 2^{NL-l'} - \sum_{l=l'+1}^{NL} 2^{NL-l} = 2^{NL-l'} - (2^{NL-l'} - 1) \Rightarrow$$
$$\Rightarrow NUM_j - NUM_i = 1$$

Thus, the integer numbers NUM_i and NUM_j occupy consecutive leaves of the binary tree.

The converse must also be true, meaning that if properties A, B & C do not hold simultaneously then the integer numbers NUM_i and NUM_j are not adjacent. To prove this, we consider the cases were the three properties do not hold simultaneously.

1. Property A does not hold

   Based on our initial assumption that $NUM_j > NUM_i$, there are two possible scenarios:

   (a) $r_{il} = r_{jl},\ l = 1, 2, ..., NL$ thus $i = j$

   (b) $r_{il} = 0$ and $r_{jl} = 1$, for more than one index $l \in [1, NL]$. Let us examine the case where this occurs for two levels in that there exist $l' < l'' \leq NL$ such that

   $$r_{il'} = 0, \quad r_{jl'} = 1$$
   $$r_{il''} = 0, \quad r_{jl''} = 1$$

Also, because $j > i$ it is obvious that $r_{il} = r_{jl}$, $l = 1, 2, ...l' - 1$. Then,

$$\begin{aligned} \Rightarrow NUM_j - NUM_i &= \sum_{l=1}^{l'-1} 2^{NL-l}(r_{jl} - r_{il}) + 2^{NL-l'}(r_{jl'} - r_{il'}) \\ &+ \sum_{l=l'+1}^{l''-1} 2^{NL-l}(r_{jl} - r_{il}) \\ &+ 2^{NL-l''}(r_{jl''} - r_{il''}) + \sum_{l=l''+1}^{NL} 2^{NL-l}(r_{jl} - r_{il}) \Rightarrow \end{aligned}$$

$$\begin{aligned} \Rightarrow \quad NUM_j - NUM_i = \; & 2^{NL-l'} + \sum_{l=l'+1}^{l''-1} 2^{NL-l}(r_{jl} - r_{il}) \\ & + 2^{NL-l''} + \sum_{l=l''+1}^{NL} 2^{NL-l}(r_{jl} - r_{il}) \end{aligned}$$

The minimum of this subtraction is achieved when $r_{il} = 1$ and $r_{jl} = 0$ in both summations. Following that,

$$NUM_j - NUM_i \geq 2^{NL-l'} - \sum_{l=l'+1}^{l''-1} 2^{NL-l} + 2^{NL-l''} - \sum_{l=l''+1}^{NL} 2^{NL-l} \Rightarrow$$

$$\begin{aligned} \Rightarrow \quad NUM_j - NUM_i \geq \; & 2^{NL-l'} - (\sum_{l=1}^{NL} 2^{NL-l} - \sum_{l=1}^{l'} 2^{NL-l} - \sum_{l=l''}^{NL} 2^{NL-l}) \\ & + 2^{NL-l''} - (2^{NL-l''} - 1) \Rightarrow \end{aligned}$$

$$\begin{aligned} \Rightarrow \quad NUM_j - NUM_i \geq \; & 2^{NL-l'} - (2^{NL} - 1) + (2^{NL} - 2^{NL-l'}) \\ & + (2^{NL-(l''-1)} - 1) + 1 \Rightarrow \end{aligned}$$

$$\Rightarrow NUM_j - NUM_i \geq 2 \cdot 2^{NL-l''} + 1 \Rightarrow$$

$$\Rightarrow NUM_j - NUM_i \geq 3$$

It is proven, accordingly, that if this occurs for more than two layers then $NUM_j - NUM_i > 1$.

2. Property B does not hold

   Assume that there exists a l', with $l' = 1, 2, \ldots, NL$, such that $r_{il'} = 0$ and $r_{jl'} = 1$. Then, there are two possible scenarios:

   (a) there exists a l'' such that

   $$l'' = \min_{l=1,2,\ldots,l'-1} l$$

   for which the following hold

   $$r_{il''} = 0, \; r_{jl''} = 1$$

   which contradicts our initial assumption that Property A holds.

(b) there exists a l'' such that

$$l'' = \min_{l=1,2,\ldots,l'-1} l$$

for which the following hold

$$r_{il''} = 1,\ r_{jl''} = 0$$

which contradicts our initial assumption that $NUM_j > NUM_i$.

3. Property C does not hold

Assume that there exists a l', with $l' = 1, 2, \ldots, NL$, such that $r_{il'} = 0$ and $r_{jl'} = 1$. Also, assume Property B holds. Then, there are two possible scenarios:

(a) there exists a l'' such that

$$l'' = \min_{l=l'+1,l'+2,\ldots,NL} l$$

for which the following hold

$$r_{il''} = 0,\ r_{jl''} = 1$$

which violates Property A and was examined earlier.

(b) there exists a l'' such that

$$l'' = \min_{l=l'+1,l'+2,\ldots,NL} l$$

for which the following holds

$$r_{il''} = r_{jl''}$$

Following that,

$$\begin{aligned} \Rightarrow NUM_j - NUM_i &= \sum_{l=1}^{l'-1} 2^{NL-l}(r_{jl} - r_{il}) + 2^{NL-l'}(r_{jl'} - r_{il'}) \\ &\quad + \sum_{l=l'+1}^{l''-1} 2^{NL-l}(r_{jl} - r_{il}) \\ &\quad + 2^{NL-l''}(r_{jl''} - r_{il''}) + \sum_{l=l''+1}^{NL} 2^{NL-l}(r_{jl} - r_{il}) \Rightarrow \end{aligned}$$

$$\Rightarrow NUM_j - NUM_i = 2^{NL-l'} + \sum_{l=l'+1}^{l''-1} 2^{NL-l}(r_{jl} - r_{il}) + \sum_{l=l''+1}^{NL} 2^{NL-l}(r_{jl} - r_{il})$$

The minimum of this subtraction is achieved when $r_{il} = 1$ and $r_{jl} = 0$ in both summations. Then,

$$\begin{aligned} NUM_j - NUM_i \geq\ & 2^{NL-l'} - (2^{NL} - 1) + (2^{NL} - 2^{NL-l'}) + (2^{NL-(l''-1)} - 1) \\ & -(2^{Nl-l''} - 1) \Rightarrow \end{aligned}$$

$$\Rightarrow NUM_j - NUM_i \geq 2.2^{NL-l''} - 2^{NL-l''} + 1 \Rightarrow$$
$$\Rightarrow NUM_j - NUM_i \geq 2^{NL-l''} + 1 \Rightarrow$$
$$\Rightarrow NUM_j - NUM_i \geq 2$$

Thus the theorem is proven. □

5 Formulation of the ATSP problem with binary tree allocation of positions to the tour

The formulation presented in this section is based on the comparison of the binary string representing the position of each city. For this task we have two alternative formulation approaches.

As comparisons of two strings may made easily only for cities within the extreme positions (left-most and right-most) contained in the tour representation, we consider in the first formulation approach problems such that the number of cities is given by:

$$N = 2^{NL} - 1 \tag{5.1}$$

We augment the distance matrix by repeating the first city as the $(N+1)^{\text{th}}$ city and utilize continuous variables indicating the presence of an arc as:

$$0 \leq x_{ij} \leq 1; \quad i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N+1) \tag{5.2}$$

The above equation defines N^2 continuous variables.

This is so because we have no cities feeding into the original city 1 which is designated arbitrarily as the start of the tour, hence $x_{i,1}$ do not appear as variables. Similarly, there are no cities linked after city $(N+1)$, which is a repetition of city 1 as the end of the tour (return to origin of tour), hence there are no variables $x_{(N+1),j}$.

In case $N+1$ is not an exact integer power of 2 we calculate NL by:

$$NL = \lceil \log_2(N+1) \rceil \tag{5.3}$$

This should be replaced appropriately in the following sections for the calculation of the number of constraints and variables appearing in the model, as applicable. Terms containing $\log_2(N+1)$ should be replaced by $\lceil \log_2(N+1) \rceil$.

The following sections outline the construction of the optimization model. The formulation results in a MILP problem.

5.1 Objective function

The objective function is comprised by the cost associated with the presence of each arc. City $(N+1)$ is simply a repetition of city 1 (arbitrary choice of start/end of tour). There are no arcs counted originating from the additional city $(N+1)$, as it is the terminus of the tour.

$$\min \sum_{i=1}^{N} \sum_{j=2}^{N} c_{ij} \cdot x_{ij} + \sum_{i=1}^{N} c_{i,1} \cdot x_{i,(N+1)} \tag{5.4}$$

It is noted that in the summation we could have demanded that $j \neq i \wedge (i,j) \neq (1,(N+1))$.

5.2 Binary data-structure related variables and partitioning of their values

We repeat here equations (2.8a)-(2.8d) indicating the partitioning of the city position binary string, appropriate for the case of equation (5.1):

$$z_{ik} \leq \alpha(t_{kl}) + \beta(t_{kl}) \cdot r_{il}; \quad l = 1, 2, \ldots, NL \tag{5.5a}$$

$$z_{ik} \geq 1 - \sum_{l=1}^{NL} (\alpha(1 - t_{kl}) + \beta(1 - t_{kl}) \cdot r_{il}) \tag{5.5b}$$

$$i = 1, 2, \ldots, (N+1); \quad k = 1, 2, \ldots, (N+1)$$

There are $(NL+1) \cdot (N+1)^2 = (N+1)^2(\log_2(N+1)+1)$ constraints in the above setting.

The parameters t_{kl} are found by running Algorithm 1 using $N+1$ positions and the corresponding NL levels.

The binary variables appearing are:

$$r_{il} \in \{0, 1\}; \quad i = 1, 2, \ldots, (N+1); \quad l = 1, 2, \ldots, NL \tag{5.5c}$$

There are $(N+1)NL = (N+1)\log_2(N+1)$ variables defined in the equation above.

The continuous variables appearing are:

$$0 \leq z_{ik} \leq 1; \quad i = 1, 2, \ldots, (N+1); \quad k = 1, 2, \ldots, (N+1) \tag{5.5d}$$

There are $(N+1)^2$ continuous variables defined in the equation above.

To guarantee uniqueness of the allocation of each position binary string to exactly one city, we need to include the uniqueness of the allocation of each city to each position:

$$\sum_{i=1}^{N+1} z_{ik} = 1; \quad k = 1, 2, \ldots, (N+1) \tag{5.5e}$$

$$\sum_{k=1}^{N+1} z_{ik} = 1; \quad i = 1, 2, \ldots, (N+1) \tag{5.5f}$$

There are $2(N+1)$ constraints defined in the equations above.

To force city 1 to position 1 of the tour, and city $(N+1)$ to position $(N+1)$ of the tour we introduce the following constraints:

$$r_{1,l} = 0; \quad l = 1, 2, \ldots, NL \tag{5.6a}$$

$$r_{(N+1),l} = 1; \quad l = 1, 2, \ldots, NL \tag{5.6b}$$

It is noted that when N is not an exact integer power of 2, equation (5.6b) above should be replaced by the following:

$$r_{(N+1),l} = t_{(N+1),l}; \quad l = 1, 2, \ldots, NL \tag{5.7}$$

There are $2NL = 2\log_2(N+1)$ constraints defined in the equations above (these constraints may be enforced by fixing the bounds of the associated variables).

It is noted that we could enforce the same condition by setting:

$$z_{1,1} = 1 \text{ and } z_{(N+1),(N+1)} = 1 \tag{5.8}$$

5.3 Checking equality of the binary string elements of city i and city j

To check the conditions of Theorem 1 it is necessary to construct a testing of the equality of the r_{il} and r_{jl} binary string variables of any two cities i and j. This is done by the following constraints, where the two variables are either tested to be equal to 0 or to 1. Any of the two cases forces a new variable to take the value of 1, else if neither case holds the variable is left to be loose in its bounds.

$$1 - (r_{il} + r_{jl}) \leq E_{ijl} \tag{5.9a}$$

$$(r_{il} + r_{jl}) - 1 \leq E_{ijl} \tag{5.9b}$$

$$i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N+1); j \neq i \wedge (i,j) \neq (1, (N+1));$$

$$l = 1, 2, \ldots, (NL - 1)$$

There are $(N^2 - N)(NL - 1) = (N^2 - N)(NL - 1) = (N^2 - N)(\log_2(N+1) - 1)$ variables E_{ijl}, with bounds $0 \leq E_{ijl} \leq 1$, and $2(N^2 - N)(\log_2(N+1) - 1)$ constraints defined in the above equations. The conditions and variables are not required for the last layer, $l = NL$.

5.4 Checking the conditions of Theorem 1 to enforce variables x_{ij}

Here logical checks are employed to enforce lower bounds equal to 1 on the x_{ij} variables, if the three conditions of Theorem 1 are met for each pair of arcs (i, j). We present two ways of doing this, one which employs additional continuous variables and one which is immediate. The conditions as written hold for $NL \geq 3$.

5.4.1 Use of auxiliary variables to test and enforce logical conditions

Property A

$$(1 - r_{il'}) + r_{jl'} - 1 \leq A_{ijl'} \tag{5.10}$$

$$i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N+1); j \neq i \wedge (i,j) \neq (1, (N+1));$$

$$l' = 1, 2, \ldots, NL$$

There are $(N^2 - N)NL = (N^2 - N)\log_2(N+1)$ constraints defined above. The number of variables $A_{ijl'}$, with $0 \leq A_{ijl'} \leq 1$, is also equal to $(N^2 - N)\log_2(N+1)$.

Property B

$$\sum_{l=1}^{l'-1} E_{ijl} - (l'-1) + 1 \leq B_{ijl'} \tag{5.11}$$

$$i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N+1); j \neq i \wedge (i,j) \neq (1,(N+1));$$

$$l' = 2, 3, \ldots, NL$$

There are $(N^2-N)(NL-1) = (N^2-N)(\log_2(N+1)-1)$ constraints defined above. The number of variables $B_{ijl'}$, with $0 \leq B_{ijl'} \leq 1$, is also equal to $(N^2-N)(\log_2(N+1)-1)$.

Property C

$$\sum_{l=l'+1}^{NL} [r_{il} + (1 - r_{jl})] - 2(NL - (l'+1) + 1) + 1 \leq C_{ijl'} \tag{5.12}$$

$$i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N+1); j \neq i \wedge (i,j) \neq (1,(N+1));$$

$$l' = 1, 2, \ldots, (NL-1)$$

There are $(N^2-N)(NL-1) = (N^2-N)(\log_2(N+1)-1)$ constraints defined above. The number of variables $C_{ijl'}$, with $0 \leq C_{ijl'} \leq 1$, is also equal to $(N^2-N)(\log_2(N+1)-1)$.

Enforcement of x_{ij} constraints

$$A_{ij,1} + C_{ij,1} - 1 \leq x_{ij}; \quad (\text{for } l' = 1) \tag{5.13a}$$

$$A_{ijl'} + B_{ijl'} + C_{ijl'} - 2 \leq x_{ij}; \quad l' = 2, 3, \ldots, (NL-1) \tag{5.13b}$$

$$A_{ij,NL} + B_{ij,NL} - 1 \leq x_{ij}; \quad (\text{for } l' = NL) \tag{5.13c}$$

$$i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N+1); j \neq i \wedge (i,j) \neq (1,(N+1))$$

There are $(N^2-N)NL = (N^2-N)\log_2(N+1)$ constraints defined above.

5.4.2 Direct enforcement of logical conditions

Direct enforcement of of the adjacency conditions follows from the previous and analysis and is presented immediately below.

$$x_{ij} \geq \underbrace{[(1 - r_{i,1}) + r_{j,1} - 2]}_{\text{Property A}} + \underbrace{\left[\sum_{l=2}^{NL} [r_{il} + (1 - r_{jl})] - 2(NL - (1+1) + 1)\right]}_{\text{Property C}} + 1;$$

$$(\text{for } l' = 1) \tag{5.14a}$$

$$x_{ij} \geq \underbrace{[(1 - r_{il'}) + r_{jl'} - 2]}_{\text{Property A}} + \underbrace{\left[\sum_{l=1}^{l'-1} E_{ijl} - (l' - 1)\right]}_{\text{Property B}} + \underbrace{\left[\sum_{l=l'+1}^{NL} [r_{il} + (1 - r_{jl})] - 2(NL - (l' + 1) + 1)\right]}_{\text{Property C}} + 1;$$

$$l' = 2, 3, \ldots, (NL - 1) \tag{5.14b}$$

$$x_{ij} \geq \underbrace{[(1 - r_{i,NL}) + r_{j,NL} - 2]}_{\text{Property A}} + \underbrace{\left[\sum_{l=1}^{NL-1} E_{ijl} - (NL - 1)\right]}_{\text{Property B}} + 1;$$

$$(\text{for } l' = NL) \tag{5.14c}$$

$$i = 1, 2, \ldots, N; \quad j = 2, 3, \ldots, (N + 1); j \neq i \wedge (i, j) \neq (1, (N + 1))$$

There are $(N^2 - N)NL = (N^2 - N)\log_2(N + 1)$ constraints defined in the equations above.

5.5 Tightening constraints

Different types of constraints (redundant) are added to the resulting MILP formulation for the ATSP so as to tighten the feasible domain.

5.5.1 Arc variables x_{ij}

Allocation of arcs (i, j) through summations of the x_{ij} variables

$$\sum_{i=1}^{N} x_{ij} = 1; \quad j = 2, 3, \ldots (N + 1) \tag{5.15a}$$

$$\sum_{j=2}^{N+1} x_{ij} = 1; \quad i = 1, 2, \ldots N \tag{5.15b}$$

There are $2N$ constraints defined in the equations above.

Elimination of self-referential arcs and of link of city 1 to city $N+1$

$$x_{ii} = 0; \quad i = 2, 3, \ldots, N; \quad j = 2, 3, \ldots, N \tag{5.16a}$$

$$x_{1,(N+1)} = 0 \tag{5.16b}$$

The above is automatically enforced by not including any forcing constraints for the variables appearing and by the requirement of minimization of the objective function.

5.5.2 Partitioning of binary string variables r_{il} and city positions

$$\sum_{i=1}^{N+1} r_{il} = \frac{N+1}{2}; \quad l = 1, 2, \ldots, NL \tag{5.17}$$

There are $NL = \log_2(N+1)$ constraints of this form. In the case that $N+1$ is not an exact integer power of 2, equations (5.17) need to be replaced by:

$$\sum_{i=1}^{N+1} r_{il} = \sum_{k=1}^{N+1} t_{kl}; \quad l = 1, 2, \ldots, NL \tag{5.18}$$

Additional tightening constraints, particularly for the case where $N+1$ is not an exact power of 2 are introduced by limiting the number of the position allocated to each city i as follows:

$$1 \leq \sum_{l=1}^{NL} 2^{NL-l} \cdot r_{il} \leq N-1; \quad i = 2, 3, \ldots, N \tag{5.19}$$

These arise from the restriction that the position of a city may range from 2 to N (since cities 1 and $N+1$ are fixed to positions 1 and $N+1$, respectively), such that $2 \leq POS_i \leq N$. There are $2(N-1)$ such constraints.

5.5.3 Symmetry breaking (case of symmetric TSP with distances given by a matrix)

For the Asymmetric TSP the following constraints should not be used, as distances from city i to city j are not the same when i and j are reversed (there is directionality in the ATSP). If this distances happen to be symmetric and we are dealing with a TSP whose distances are given by a matrix, then the following constraints are used.

Similar to the discussion in section 3.1, we introduce the following constraint to force the index of the second city in the tour to be smaller than the last one (penultimate city in the augmented tour representation) closing the tour by returning to city $(N+1)$:

$$\sum_{i=1}^{N} i \cdot z_{i,2} \leq \sum_{i=1}^{N} i \cdot z_{i,N} \tag{5.20}$$

An alternative and equivalent constraint for the ATSP case is to base the comparison on the x_{ij} arc presence variables:

$$\sum_{j=2}^{N} j \cdot x_{1,j} \leq \sum_{i=2}^{N} i \cdot x_{i,(N+1)} \tag{5.21}$$

City	x	y
1	10.58416945	−51.07803567
2	12.64040569	11.61855973
3	−4.72246031	64.37609131
4	28.77288143	−43.37664821
5	0.132976303	3.24127766
6	−95.34669232	80.51883235
7	0.34608131	20.55363708
8	9.07583291	0.493862256

Tab. 1: Coordinates for 8-cities problem.

The above constraint is based on the observation that city 1 is set as the start of the tour, so we scan in the LHS of the constraint the second city in the tour. In the RHS it is recognized that the artificially introduced city $(N+1)$, which is equivalent to city 1, forms the end of the tour, and thus we scan all possible cities that feed into it.

Either of these constraints (or both for more tightness) may be included into the formulation to break the symmetry of the solutions.

5.6 Alternative formulation

An alternative formulation for the ATSP problem which eliminates the need to consider $N+1$ cities is described here. All the constraints defined in Equations (5.5a)-(5.21) stay the same for number of cities N and $i, j, k = 1, 2, ..., N$ (*i.e.* replacing the maximum number of cities with N instead of $N+1$). Tightening constraints (5.15b) and (5.15a) need to be summed over $i, j = 1, 2, \ldots, N$ with the exclusion $i \neq j$. The objective function becomes

$$\min \sum_{i=1}^{N} \sum_{\substack{j=1 \\ j \neq i}}^{N} c_{ij} \cdot x_{ij} \tag{5.22}$$

and the constraint

$$x_{i,1} = z_{i,N};\ i = 1, 2, ..., N \tag{5.23}$$

needs to be added to enforce the presence of an arc between the city placed on the last leaf and the first city of the tour.

6 Results

All three formulations are coded in GAMS and tested using small size problems. The solver CPLEX has been used and the runs are done on an ASUS Chassis AMD Athlon Processor 2.21 GHz PC.

6.1 Manhattan formulation

To test the Manhattan formulation we implement two problems of 8 and 10 cities. For these problems the coordinates are chosen using a random number generator. The coordinates for the two problems are given respectively in Tables 1 and 2.

A summary of the results is given in Table 6.1.

City	x	y
1	0.938475958	−0.24007422
2	0.492295391	0.980246747
3	0.044364637	0.214636689
4	0.916432093	0.158259034
5	−0.54520066	0.473306820
6	0.49092077	−0.58925045
7	0.49092077	0.557613960
8	0.017379759	0.538031086
9	0.897945947	−0.254021207
10	−0.82659591	0.2146296389

Tab. 2: Coordinates for 10-cities problem.

Case	BV	CV	Constraints	Nodes	Iterations	CPU Time	Optimal Tour
8-cities	21	121	330	496	16841	0.063	511.432884
10-cities	36	177	593	2544	132626	0.11	6.669152

Tab. 3: Summary of results obtained using the Manhattan formulation.

6.2 ATSP

To test the ATSP formulations we implement three problems of size 8 and 10 cities. For the first two cases the intercity distances are chosen using a random number generator. The distance coefficients are given for the two problems in Tables 4 and 6.2, respectively.

A summary of the results is given in Table 6.2.

7 Conclusions

This paper presents a totally novel formulation, in which it is recognized that the underlying binary degrees of freedom in the general TSP problem is $N \log N$ which is the smallest number of binary variables from all published formulations for the TSP. This is based on a binary tree allocation of cities to their positions in the tour.

The proposed TSP formulation was found to converge very slowly for larger problems than the ones reported in this work, and the explanation for this is that the lower bounds produced in the Branch and Bound method were very loose. One explanation for this is

City	1	2	3	4	5	6	7	8
1	0	29	28	11	2	8	17	6
2	21	0	2	13	12	9	5	23
3	27	24	0	8	7	8	38	3
4	28	23	26	0	1	13	28	25
5	21	2	23	15	0	16	17	41
6	12	18	33	25	3	0	15	2
7	3	13	9	27	24	25	0	19
8	27	8	40	8	27	13	2	0

Tab. 4: Intercity distances for 8-cities problem.

City	1	2	3	4	5	6	7	8	9	10
1	0	21	27	22	6	35	33	33	24	41
2	78	0	34	23	46	48	4	2	13	22
3	32	7	0	10	7	11	6	47	10	10
4	21	13	3	0	47	40	48	21	45	17
5	13	27	12	8	0	16	32	5	18	15
6	25	6	10	34	35	0	21	49	4	22
7	25	7	22	25	39	27	0	12	16	10
8	22	28	30	26	28	23	31	0	45	33
9	24	5	39	5	21	5	32	26	0	16
10	29	3	7	31	21	4	26	35	30	0

Tab. 5: Intercity distances for 10-cities problem.

Case	BV	CV	Constraints	Nodes	Iterations	CPU Time	Optimal Tour
8-cities	21	247	695	209	6414	0.062	31
10-cities	36	488	1460	600	32152	0.079	70

Tab. 6: Summary of results obtained using ATSP formulation.

the nature of constraints (5.14a)-(5.14c) which enforce the conditions of Theorem 1 to set the x_{ij} variables to 1.

Although computationally the resulting TSP formulation is not competitive, this work presents a new methodology of structuring the information describing the problem which may lead to future developments exploiting it. The scheme is equivalent to binary expansion of tour-locations which may be applicable to other standard TSP formulations, thus allowing there also the same reduction in the number of binary variables.

The formulation proposed has a hierarchical structure, in that it may be viewed as repeated partitioning (halving) of the set of cities at each level of the binary tree representation, so that eventually single cities are allocated at tour positions sequentially. Future work will focus on possibilities of exploiting this property either in rigorous formulations or in deriving useful heuristic procedures.

Finally, Lagrangean Relaxation methods will be considered for our new formulation.

References

Claus, A., 1984. A new formulation for the travelling salesman problem. SIAM Journal on Algebraic and Discrete Methods 5 (1), 21–25.

Gavish, B., Graves, S. C., Jul. 1978. The travelling salesman problem and related problems.

Millar, H., Cyrus, P., 2000. An alternate formulation and lagrangian heuristic for the traveling salesman problem. In: ASAC-IFSAM 2000 Conference, Montreal, Quebec, Canada.

Miller, C. E., Tucker, A. W., Zemlin, R. A., 1960. Integer programming formulation of traveling salesman problems. J. ACM 7 (4), 326–329.