On the integrality Gap of Small Asymmetric Travelling Salesman Problems: A Polyhedral and Computational Approach

Eleonora Vercesi[a,b,*], Janos Barta[a,c], Luca Maria Gambardella[a,b], Stefano Gualandi[d], Monaldo Mastrolilli[a,c]

[a] *Istituto Dalle Molle di studi sull'Intelligenza artificiale (IDSIA USI-SUPSI), Via La Santa 1, 6900, Lugano, Switzerland*
[b] *Faculty of Informatics, Università della Svizzera italiana, Via La Santa 1, 6900, Lugano, Switzerland*
[c] *Dipartimento Tecnologie innovative, Scuola universitaria professionale della Svizzera italiana, Via La Santa 1, 6900, Lugano, Switzerland*
[d] *Department of Mathematics "Felice Casorati", Universit of Paviia, Via Ferrata 5, 27100, Pavia, Italy*

Abstract

In this paper, we investigate the integrality gap of the Asymmetric Traveling Salesman Problem (ATSP) with respect to the linear relaxation given by the Asymmetric Subtour Elimination Problem (ASEP) for small-sized instances. In particular, we focus on the geometric properties and symmetries of the ASEP polytope (P^n_{ASEP}) and its vertices. The polytope's symmetries are exploited to design a heuristic pivoting algorithm for the search of vertices where the integrality gap is maximized. Furthermore, a general procedure for the extension of vertices from P^n_{ASEP} to P^{n+1}_{ASEP} is defined. The generated vertices improve the known lower bounds of the integrality gap for $16 \leq n \leq 22$ and, provide small hard-to-solve ATSP instances.

Keywords: Asymmetric Traveling Salesman Problem, Integrality Gap
2000 MSC: 90C05,
2000 MSC: 90C10.

[*] Corresponding author

Email addresses: `eleonora.vercesi@usi.ch` (Eleonora Vercesi), `janos.barta@supsi.ch` (Janos Barta), `luca.gambardella@usi.ch` (Luca Maria Gambardella), `stefano.gualandi@unipv.it` (Stefano Gualandi), `monaldo@idsia.ch` (Monaldo Mastrolilli)

1. Introduction

Mathematical programming relaxations and especially linear programming relaxations have played a central role both in solving combinatorial optimization problems in practice (e.g. see, [1]) and in the design and analysis of approximation algorithms (see, e.g., [29]). When solving an instance, an important concept is the *integrality gap* with respect to the linear relaxation, which is the maximum ratio between the solution quality of the Integer Linear Program (ILP) and its Linear Program (LP) relaxation. For many integer linear programs, the integrality gap of the linear relaxation is equal to the approximation ratio of the best algorithm as well as the hardness of the approximation ratio and essentially represents the inherent limits of the considered relaxation. To some extent, instances having a large integrality gap are the hard instances for the class of approaches based on linear programming. With this respect, the following facts can be observed from the literature.

- Proving integrality gaps for LP relaxations of NP-hard optimization problems is a difficult task that is usually undertaken on a case-by-case basis (e.g., see the case of the Vertex Cover [27]). Very few and limited attempts have been made to use computer-assisted analysis for this difficult goal.

- For several notable and important examples, like the Traveling Salesman Problem (TSP), our integrality gap comprehension resisted the persistent attack during the last decades of many researchers. For the Symmetric Travelling Salesman Problem (STSP), several advances have been made in specific cases, such as the STSP having only costs 1 and 2 [23], and the cubic and subcubic STSP [6], where the integrality gap is proved to be equal to $\frac{4}{3}$. For the asymmetric case, [12, 9] independently shows that the lower bound for the integrality gap is 2, and no further improvements have been made. More specifically, [12] also provides specific lower bounds for $n \leq 15$. This is the first time an IG greater than $\frac{4}{3}$ is shown for $n = 9$. In terms of the upper bound, the best-known bound is constant ad equal to 22 [28].

Within this paper, we investigate the integrality gap *computationally* for the Asymmetric Traveling Salesman Problem. More specifically, we investigate and exploit aspects of polyhedral theory to reduce the integrality gap search space. Then, we combine this information to effectively search the pruned space. This approach allows us to get computer-aided construction of bad instances, namely, problem instances having large integrality gap values, for the considered LP relaxation which improves upon the best-known lower bounds for small n (see Charikar, Goemans, and Karloff [9] and Elliot-Magwood [12]). The aim is to obtain a new and better understanding of lower bounds on the corresponding integrality gaps. Besides, we present hard-to-solve ATSP instances for the state-of-the-art solver Concorde.

The problem.. Given a weighted directed graph, a Hamiltonian cycle is a cycle that visits each vertex exactly once. The Asymmetric Traveling Salesman Problem (ATSP) involves finding a directed Hamiltonian cycle of minimum cost. In practice, the currently most efficient exact algorithm for solving ATSP is based on an Integer Linear Programming (ILP) [26, 14, 13]. Let $K_n = (V, A)$ be the complete directed graph with n nodes and $m = n(n-1)$ arcs, that is, $V := \{1, \ldots, n\}$ and $A := \{(i, j) \mid i, j \in V, i \neq j\}$, each having a weight $c_{ij} \in \mathbb{R}^+$. Whenever $c_{ij} = c_{ji}$ for all arcs (i, j), then we have a Symmetric TSP instance. Whenever $c_{ij} \neq c_{ji}$, but the cost vector satisfies the triangle inequality $c_{ij} \leq c_{ik} + c_{kj}, \forall i, j, k \in V$, we have a pseudo-quasi metric TSP instance, since the cost vector induces a *pseudo-quasi metric* [22]. Note that any instance of ATSP on a complete graph K_n is completely defined by its cost vector $\boldsymbol{c} \in \mathbb{R}^m_+$. Given $S \subseteq V$, let $\delta(S)$ be the cut induced by S, namely $\delta(S) := \{(i, j) \mid i \in S, j \notin S\}$. For convenience, we denote $\delta(i) = \delta(\{i\})$.

In this paper, we study the LP relaxation of the Dantzig-Fulkerson-Johnson (DFJ) formulation [11] for solving the ATSP for small values of n.

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} c_{ij} x_{ij} & & & (1)\\
\text{s.t.} \quad & \sum_{i\in V} x_{ij} = 1 & & \forall j \in V & (2)\\
& \sum_{j\in V} x_{ij} = 1 & & \forall i \in V & (3)\\
& \sum_{i\in \delta(S)} x_{ij} \geq 1 & & \forall S \subset V \text{ such that } 2 \leq |S| \leq n-2 & (4)\\
& x_{ij} \in \{0, 1\} & & \forall (i, j) \in A, & (5)
\end{aligned}
$$

where $x_{ij} \in \{0,1\}$ is equal to 1 if arc (i,j) is one optimal cycle, and 0 otherwise. Constraints (2)–(3) are the in-degree and out-degree constraints, that force each node to have exactly one predecessor and one successor. Constraints (4) are the *Subtour Elimination Constraints*, which avoid the presence of sub-cycles (subtours). For a survey on ATSP formulations, we refer the reader to [26]. In the literature, the solution of the LP relaxation of (1)–(5) is called the *Asymmetric Subtour Elimination Problem* (ASEP), while the corresponding feasibility region is the *ASEP Polytope*, which is defined as follows.

$$P^n_{ASEP} := \{\boldsymbol{x} \in \mathbb{R}^m \mid (2)\text{–}(4), \boldsymbol{x} \geq 0\}. \tag{6}$$

The Integrality Gap (IG) for the ATSP on n nodes is

$$\alpha_n := \sup_{c \in \mathbb{R}^m_+} \frac{\text{ATSP}(\boldsymbol{c})}{\text{ASEP}(\boldsymbol{c})}, \tag{7}$$

where ATSP($\boldsymbol{c}$) is the optimal value of (1)–(5) and ASEP($\boldsymbol{c}$) is the optimal value of its LP relaxation. In general, the IG for the ATSP is

$$\alpha := \sup_{n \in \mathbb{N}} \alpha_n.$$

In [12], it is shown that for the general ATSP, a non-negative cost vector $\boldsymbol{c}$ exists such that ATSP($\boldsymbol{c}$) > 0 and ASEP($\boldsymbol{c}$) $= 0$. This implies that the integrality gap tends to $+\infty$. For this reason, the works studying the integrality gap of ATSP always assume that the cost vector satisfies the triangle inequality $c_{ij} \leq c_{ik} + c_{kj}, \forall i,j,k \in V$ and, hence, the costs induce a pseudo-quasi metric. Herein, we restrict our attention to pseudo-quasi metric ATSP, but we will call them ATSP for short, to be consistent with the notation of the literature.

A long-standing conjecture stated that $\alpha \leq \frac{4}{3}$ (e.g., see [8]). However, this conjecture was disproved independently in [12] and [9]. Both works show $\alpha \geq 2$ by presenting two families of ATSP instances with $\alpha(\boldsymbol{c}) \to 2$ for $n \to \infty$. For $n \leq 3$, we have that $\alpha_n = 1$, while for $4 \leq n \leq 7$ the exact value was computed in [12]. The authors of [12, 7] provide also lower bounds of α_n for $n \leq 15$. In particular, they show that $\alpha > \frac{4}{3}$, since they prove (compute) that $\alpha_n \geq \frac{11}{8}$ for $n = 9$. Lower bounds were also provided by Charikar et al. [9] for arbitrarily large n, but they are rather weak for $n \leq 25$. We remark that the literature is more extensive for the Symmetric TSP (STSP)

[5, 19, 20, 31, 30]. For instance, [31, 30, 20] propose STSP instances that have a large integrality gap and are hard-to-solve for the state-of-art solver Concorde [1]. However, no theoretical proof is available for the exact value of the integrality gap, neither for the STSP nor the ATSP. Furthermore, the relation between computational complexity and large integrality gap is still unclear even in the symmetric case.

Main contributions.. This paper has three main contributions. First, we identify, for the first time, a group of symmetries of the ASEP polytope. We exploit this symmetry to design pivoting strategies that explore vertices of P^n_{ASEP}. Second, we provide new lower bounds for the integrality gap α_n for the ATSP for $16 \leq n \leq 22$ by using a new pivoting algorithm which exploits the symmetries of the vertices of P^n_{ASEP}, combined with an inductive algorithm that generates vertices of P^{n+1}_{ASEP} from a vertex of P^n_{ASEP}. Our bounds improve those provided in [12] and [9]. Third, by using our new inductive algorithm, we generate hard ATSP instances, where complexity is measured with respect to the Concorde solver for STSP [1], after an appropriate transformation of the ATSP instance.

The outline of this paper is as follows. Section 2 reviews the background material. Section 3 studies the symmetries of the polytope P^n_{ASEP}. In Section 4, we explain how we use the algebraic structure introduced to perform a symmetry-breaking heuristic pivoting. Furthermore, we introduce a new operator that generates vertices of P^{n+1}_{ASEP} from a vertex of P^n_{ASEP}. In Section 5 we present the results of our approach, by analyzing the structure of the obtained vertices and exhibiting new lower bounds for $16 \leq n \leq 22$. Finally, in Section 6 we conclude the paper with a perspective on future works.

2. Background material

A key subproblem of our approach is the computation of the maximum integrality gap over all possible pseudo-quasi metric cost vectors $\boldsymbol{c} \in \mathbb{R}^m_+$ for a single vertex of the ASEP polytope. This subproblem was first introduced in [5] for the symmetric TSP (STSP) and later in [12] for the asymmetric TSP (ATSP). The main idea is to divide the cost vector $\boldsymbol{c}$ by the optimal value ATSP($\boldsymbol{c}$), so that the definition of IG reduces to $\alpha_n := \sup_{\boldsymbol{c}} \frac{1}{\text{ASEP}(\boldsymbol{c})}$, which is equivalent to solve $\inf_{\boldsymbol{c}} \text{ASEP}(\boldsymbol{c})$. Note that the operation of dividing the costs by ATSP($\boldsymbol{c}$) maintains the triangle inequalities and preserves the value of the IG.

The problem of computing the maximum α_n of K_n for the pseudo-quasi metric (pq-metric) ATSP is as follows [12].

$$\frac{1}{\alpha_n} := \min_{\substack{\boldsymbol{c} \text{ is pq-metric,} \\ \text{ATSP}(\boldsymbol{c})=1}} \text{ASEP}(\boldsymbol{c}) = \min_{\boldsymbol{x} \in P^n_{ASEP}} \min_{\substack{\text{is pq-metric,} \\ \text{ATSP}(\boldsymbol{c})=1}} \boldsymbol{x}^T \boldsymbol{c}. \tag{8}$$

To solve the problem (8) for a fixed (small) value of n, the authors in [12] enumerate the vertices of P^n_{ASEP}, and for each vertex $\boldsymbol{x} \in P^n_{ASEP}$, they solve $\min \left\{ \boldsymbol{c}^T \boldsymbol{x} \mid \boldsymbol{c} \text{ is pq-metric, ATSP}(\boldsymbol{c}) = 1 \right\}$. In [12], the latter inner problem is called Gap($\boldsymbol{x}$). Intuitively Gap($\boldsymbol{x}$) is related to the IG for a fixed vertex x and it is defined as

$$\begin{aligned}
\text{Gap}(\boldsymbol{x}) := \min & \sum_{(i,j) \in A} x_{ij} c_{ij} && (9) \\
\text{s.t.} & \sum_{(i,j) \in A} z_{ij} c_{ij} \geq 1 \quad \forall \boldsymbol{z} \in \mathcal{T}^n_{ASEP} && (10) \\
& c_{ij} \leq c_{ik} + c_{jk} \quad \forall i, j, k \in V && (11) \\
& c_{ij} \geq 0 \quad \forall (i,j) \in A && (12) \\
& c_{ij} - y_i^{\text{out}} - y_j^{\text{in}} - \sum_{S \in \mathcal{S}_{ij}(\boldsymbol{x})} d_S \geq 0 \quad \forall (i,j) \in A \backslash A(\boldsymbol{x}) && (13) \\
& c_{ij} - y_i^{\text{out}} - y_j^{\text{in}} - \sum_{S \in \mathcal{S}_{ij}(\boldsymbol{x}^*)} d_S = 0 \quad \forall (i,j) \in A(\boldsymbol{x}) && (14) \\
& d_S \geq 0 \quad \forall S \in \bigcup_{(i,j) \in A} \mathcal{S}_{ij}(\boldsymbol{x}), && (15)
\end{aligned}$$

where $\mathcal{T}^n_{\text{ASEP}}$ is the collection of every possible Hamiltonian cycle of K_n, and $\boldsymbol{z} \in \{0,1\}^m$ are incidence vectors of elements of $\mathcal{T}^n_{\text{ASEP}}$. We remark that the variables are cost c_{ij}, while x_{ij} and z_{ij} are given. Constraints (10) ensure that the optimal solution $\boldsymbol{c}^*$ of Gap($\boldsymbol{x}$) yields ATSP$(\boldsymbol{c}^*) = 1$. Constraints (13)–(15) derive from dualizing the linear program relaxation of (2)–(4). Here, we define the following subset of nodes $S_{ij(\boldsymbol{x})}$ and of arcs $A(\boldsymbol{x})$ as follows: $\mathcal{S}_{ij}(\boldsymbol{x}) = \{S \subset A | (i,j) \in \delta(S), \boldsymbol{x}(\delta(S)) = 1, 2 \leq |S| \leq n-2\}$ and $A(\boldsymbol{x}) = \{(i,j) \in A \mid x_{ij} > 0\}$. Constraints (13)–(15) ensure that

$$\boldsymbol{x}^* \in \arg \min_{\boldsymbol{x} \in P^n_{ASEP}} \boldsymbol{c}^* \boldsymbol{x} \Leftrightarrow \boldsymbol{c}^* \in \arg \min_{\boldsymbol{c}, \boldsymbol{y}, \boldsymbol{d} \text{ satisfy } (10)\text{–}(15)} \text{Gap}\left(\boldsymbol{x}^*\right).$$

Thus, the arg min $\boldsymbol{c}^*$ of the program Gap $(\boldsymbol{x}^*)$ is such that, once ASEP $(\boldsymbol{c}^*)$ is solved, the minimum is attained precisely at $\boldsymbol{x}^*$ (for details, see [12]).

3. Symmetry group of $\boldsymbol{P^n_{ASEP}}$

Polyhedral aspects of the TSP have been extensively studied in the past decades. In particular, in [16, 24, 4] the polyhedral properties of the convex hull of the integer vertices, the so-called "natural polytope", of the STSP and the ATSP have been investigated starting from the assignment polytope. More specifically, it has been shown that the diameter of the assignment polytope is two, which in turn implies that the diameter of the TSP polytope is at most 2. However, since an integer formulation is not known for the TSP, the focus is on the relationship between a valid formulation of the TSP (such as, in this paper, that of Dantzig-Fulkerson-Johnson [11]) and its linear relaxation. In this section, we focus in particular on the fractional vertices and the symmetry properties of the subtour elimination polytope P^n_{ASEP}.

As remarked in [12], the vertices of P^n_{ASEP} can be subdivided into equivalence classes through permutations of the nodes of K_n. In this section, we define explicitly a symmetry group of P^n_{ASEP} based on permutations. The classes of isomorphic vertices turn out to be the orbits generated by this symmetry group.

By observing the definition of P^n_{ASEP}, an important consideration can be made: due to the structure of the constraints, they are not affected by a permutation of the node indices. More precisely, it is well known that any relabeling of the nodes $i \in V$ induces just an internal permutation of the constraint groups (2), (3) and (4), which leaves the feasible region unchanged. Consequently, it is possible to identify a group of symmetries of the polytope P^n_{ASEP} induced by the symmetric group S_n of permutations of the nodes $i \in V$. To describe the symmetries of P^n_{ASEP} explicitly, it is useful to convert the feasible solutions into matrix form. More precisely, we can rewrite any feasible solution $\boldsymbol{x} \in P^n_{ASEP}$ as a matrix $\boldsymbol{X} \in [0,1]^{n \times n}$, with the corresponding components x_{ij} for $i \neq j$ and setting $x_{ij} = 0$ for $i = j$.

Now, consider the symmetric group of permutations on n elements S_n. Let $\pi \in S_n$, such that $\pi : i \mapsto \pi(i)$, be a permutation of the nodes $i \in V$. For any feasible solution $\boldsymbol{x} \in P^n_{ASEP}$, with matrix representation $\boldsymbol{X}$, it is possible to generate a new feasible solution $\boldsymbol{x'}$, expressed by a matrix $\boldsymbol{X'}$, by applying the permutation π to the nodes of the graph K_n. It must hold

$$x'_{\pi(i)\pi(j)} = x_{ij} \quad \forall i, j \in V. \tag{16}$$

This means that $\boldsymbol{X'}$ is obtained by permuting rows and columns of $\boldsymbol{X}$ according to π. Let us define the permutation matrix $\boldsymbol{P}_\pi = (p_{ij})$ associated to $\pi \in S_n$ as

$$p_{ij} = \begin{cases} 1 & \text{if } i = \pi(j) \\ 0 & \text{otherwise.} \end{cases}$$

In the product $\boldsymbol{P_\pi X}$, the permutation matrix $\boldsymbol{P_\pi}$ permutes the rows of $\boldsymbol{X}$ according to π. Since also the columns of $\boldsymbol{X}$ have to be permuted, we apply $\boldsymbol{P_\pi}$ to the transpose of $\boldsymbol{P_\pi X}$. Thus

$$\boldsymbol{X'} = \left(\boldsymbol{P_\pi}(\boldsymbol{P_\pi X})^T\right)^T = (\boldsymbol{P_\pi X})\boldsymbol{P_\pi^T}= \boldsymbol{P_\pi X P_\pi^T}.$$

A notion that we will widely use is the *isomorphism between digraphs.* Hence, let us recall some helpful definitions.

Definition 1 (Support digraph). Let $\boldsymbol{x} \in P^n_{ASEP}$. The weighted support digraph $\boldsymbol{x}$ is the graph $H(\boldsymbol{x})$ defined on the set of nodes V, having an arc (i, j) with the weight x_{ij}, if and only if $x_{ij} > 0$.

Definition 2 (Isomorphism between weighted graphs). Two graphs $D = (V, A_D)$ and $F = (V, A_F)$ on n nodes are isomorphic if there exists a permutation π of V such that $(u, v) \in A_D \iff (\pi(u), \pi(v)) \in A_F$. Furthermore, the weights of the edges must satisfy $c_{\pi(u)\pi(v)} = c_{uv}, \forall u, v \in V$.

Example 1. *Consider $\boldsymbol{x} \in P^4_{ASEP}$ defined by*

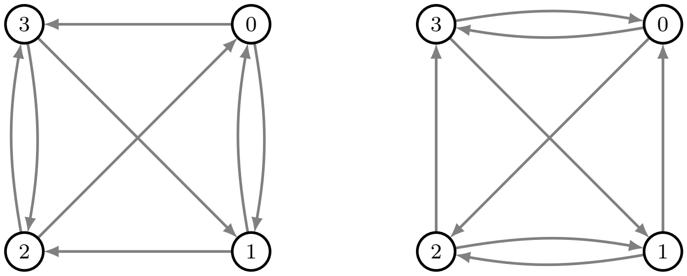
$$\boldsymbol{x} = \left(\frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2}, \frac{1}{2}, 0, \frac{1}{2}, 0, \frac{1}{2}, 0, \frac{1}{2}, \frac{1}{2}\right)^T.$$

Its matrix version is hence

$$\boldsymbol{X} = \begin{pmatrix} 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ \frac{1}{2} & 0 & 0 & \frac{1}{2} \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 \end{pmatrix}.$$

This feasible solution is associated with the support graph in Figure 1, left. Let $\pi = (0\ 1\ 2\ 3)$, that is the permutation such that $\pi(0) = 1,\ \pi(1) = 2,\ \pi(2) = 3,\ \pi(3) = 0$. Thus, in this case

$$\boldsymbol{P}_\pi = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}$$



(a)

(b)

Figure 1: Two isomorphic vertices obtained via vertex permutation. Each arc is weighted $\frac{1}{2}$.

and we obtain

$$\boldsymbol{X}' = \boldsymbol{P}_\pi \boldsymbol{X} \boldsymbol{P}_\pi^T = \begin{pmatrix} 0 & 0 & \frac{1}{2} & \frac{1}{2} \\ \frac{1}{2} & 0 & \frac{1}{2} & 0 \\ 0 & \frac{1}{2} & 0 & \frac{1}{2} \\ \frac{1}{2} & \frac{1}{2} & 0 & 0 \end{pmatrix},$$

which corresponds to the graph in Figure 1 on the right.

For any node relabeling $\pi \in S_n$ we can define the corresponding permutation of a solution $\boldsymbol{X}$ as

$$\begin{aligned} g_\pi : [0,1]^{n \times n} &\to [0,1]^{n \times n} \\ \boldsymbol{X} &\mapsto \boldsymbol{P}_\pi \boldsymbol{X} \boldsymbol{P}_\pi^T. \end{aligned} \tag{17}$$

With a slight abuse of notation, we will use both $g_\pi(\boldsymbol{x})$ and $g_\pi(\boldsymbol{X})$ interchangeably, denoting the *row and column permutation* of $\boldsymbol{X}$, according to π.

First of all, we observe that the map is well defined in $P^n_{_{ASEP}}$, namely $\boldsymbol{x} \in P^n_{_{ASEP}}$ implies $g_\pi(\boldsymbol{x}) \in P^n_{_{ASEP}}$. Hence, all the constraints are satisfied also for $g_\pi(\boldsymbol{x})$ by a simple "shuffle" of the rows: degree constraints i become degree constraints $\pi(i)$ and subtour elimination constraints associated to $\delta(S),\ S = \{s_1, \ldots, s_r\}$ become $\delta(\pi(S)),\ \pi(S) := \{\pi(s_1), \ldots, \pi(s_r)\}$. The isomorphism of vertices, observed in [12], can now be extended to the whole polytope $P^n_{_{ASEP}}$. Hence, it trivially follows

Lemma 1. *Let $x \in P^n_{_{ASEP}}$ and $\pi \in S_n$. The support graphs $H(\boldsymbol{x})$ and $H(g_\pi(\boldsymbol{x}))$ are isomorphic.*

Let be $G(n) = \{g_\pi \mid \pi \in S_n\}$ the set of all transformations g_π. The next theorem shows that $G(n)$ is a group of symmetries of the polytope.

Theorem 2. *$G(n)$ is a group of isometries acting on of $P^n_{_{ASEP}}$.*

Proof. We begin by showing the closure of $G(n)$ under composition. Let $\pi_1, \pi_2 \in S_n$. By equation (17) we have

$$\begin{aligned} g_{\pi_1} g_{\pi_2}(\boldsymbol{X}) &= g_{\pi_1}\left(\boldsymbol{P}_{\pi_2}\boldsymbol{X}\boldsymbol{P}^T_{\pi_2}\right) = \boldsymbol{P}_{\pi_1}\boldsymbol{P}_{\pi_2}\boldsymbol{X}\boldsymbol{P}^T_{\pi_2}\boldsymbol{P}^T_{\pi_1} \\ &= \left(\boldsymbol{P}_{\pi_1\pi_2}\right)\boldsymbol{X}\left(\boldsymbol{P}_{\pi_1\pi_2}\right)^T = g_{\pi_1\pi_2}(\boldsymbol{X}) \in G(n). \end{aligned} \tag{18}$$

It is not difficult to verify that the symmetric group S_n induces the group structure on $G(n)$. In particular, by equation (18) we have that the identity element of $G(n)$ is g_{id} and the inverse element of g_π is $g_{\pi^{-1}}$. Moreover, equation (18) states that $G(n)$ and S_n are isomorphic groups. By equation (16) it can be observed that for any feasible solution $\boldsymbol{x} \in P^n_{_{ASEP}}$ the solution $\boldsymbol{x}' = g_\pi(\boldsymbol{x}) \in P^n_{_{ASEP}}$ is obtained by a permutation of the components of $\boldsymbol{x}$ based on π. Therefore, it is immediately clear that g_π preserves the Euclidean distance, that is $||g_\pi(\boldsymbol{x}) - g_\pi(\boldsymbol{y})|| = ||\boldsymbol{x} - \boldsymbol{y}||$, $\forall \boldsymbol{x}, \boldsymbol{y} \in P^n_{_{ASEP}}, \pi \in S_n$.

□

As explained in Section 2, our main goal is the computation of the integrality gap of vertices. The following corollary focuses on the action of $G(n)$ on the set of vertices, which will be denoted through the manuscript with $\mathcal{X}^n_{_{ASEP}}$.

Corollary 3. *Let $\boldsymbol{x} \in \mathcal{X}^n_{_{ASEP}}$ and $g_\pi \in G(n)$. Then $g_\pi(\boldsymbol{x}) \in \mathcal{X}^n_{_{ASEP}}$.*

Proof. g_π is an isometry and isometries map vertices into vertices. □

Since the group $G(n)$ acts on the set of vertices $\mathcal{X}^n_{_{ASEP}}$, it is of interest to study the *orbit* of each vertex $\boldsymbol{x} \in \mathcal{X}^n_{_{ASEP}}$, that is the set

$$O_{\boldsymbol{x}} = \{g_\pi(\boldsymbol{x}) \mid g_\pi \in G(n)\}, \tag{19}$$

and the *stabilizer* of $G(n)$ with respect to $\boldsymbol{x} \in \mathcal{X}^n_{_{ASEP}}$, that is the subgroup of $G(n)$ defined as

$$G_{\boldsymbol{x}} = \{g_\pi \in G(n) \mid g_\pi(\boldsymbol{x}) = \boldsymbol{x}\}.$$

Combining Lemma 1 and (19), we can conclude that vertices of the polytope $P^n_{_{ASEP}}$ belonging to the same orbit, have isomorphic support graphs. In other terms, the isomorphism classes of vertices introduced in [12] are the orbits of the vertices with respect to the group $G(n)$.

By the so-called *orbit-stabilizer theorem* we have the relation $|G(n)| = |G_{\boldsymbol{x}}||O_{\boldsymbol{x}}|$. Further details on the orbit-stabilizer theorem, can be found for instance in [2]. A natural question concerns the role of the stabilizer of each vertex, and if it leads to some implications in combinatorial questions, such as the integrality gap. For example, what could be the relationship between the stabilizer of a vertex and the maximum integrality gap achievable at that vertex? The case of the integer vertices is particularly interesting. As already pointed out, integer vertices of P^n_{ASEP} correspond to Hamiltonian cycles of K_n. However, Hamiltonian cycles differ from each other only by a relabeling of the nodes. Therefore we can prove the following property.

Lemma 4. *Let $\boldsymbol{x} \in \mathcal{T}^n_{ASEP}$. Then, it holds $O_{\boldsymbol{x}} = \mathcal{T}^n_{ASEP}$.*

Proof. Let $\boldsymbol{x_1}, \boldsymbol{x_2} \in \mathcal{T}^n_{ASEP}$. Since $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ correspond to Hamiltonian cycles in K_n, there exists a permutation π, such that $g_\pi(\boldsymbol{x_1}) = \boldsymbol{x_2}$. It follows that $\boldsymbol{x_1}$ and $\boldsymbol{x_2}$ belong to the same orbit of $G(n)$. □

A consequence of Lemma 4 is that integer vertices build a unique orbit $O_{\boldsymbol{x}}$ with $|O_{\boldsymbol{x}}| = (n-1)!$. By the orbit-stabilizer theorem it follows that $|G_{\boldsymbol{x}}| = n, \ \forall \boldsymbol{x} \in \mathcal{T}^n_{ASEP}$.

More specifically, it holds the following

Lemma 5. *If $\boldsymbol{x} \in \mathcal{T}^n_{ASEP}$, then $G_{\boldsymbol{x}}$ is the group*

$$\langle (1\,2 \ldots n) \rangle$$

of all the cyclic permutations, that are

$$\tau_k(i) = (i+k) \mod n.$$

Proof. As these permutation are the k-shift of nodes, $k \in \{0, 1, 2, \ldots, n\}$, it holds

$$\tau_{k_1} \circ \tau_{k_2} = \tau_{k_3} \qquad k_3 = (k_1 + k_2) \mod n.$$

□

Unfortunately, as further discussed in Section 5.3, the stabilizer of a fractional vertex $\boldsymbol{x}$ of P^n_{ASEP} does not seem to follow a general recipe.

In Section 6, we conjecture that vertices having a large integrality gap are the ones having large but not trivial stabilizers, hence, it could be relevant to know in advance the structure of the stabilizer to guess the "promising" vertices.

It is worth to remark that the orbits of $G(n)$ form large classes of isomorphic vertices, hence, P^n_{ASEP} can be considered a highly symmetric polytope.

The intrinsic equivalence of vertices belonging to the same orbit is clearly visible in the following lemma.

Lemma 6. *Let $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}^n_{ASEP}$ such that $\boldsymbol{y} \in O_{\boldsymbol{x}}$, then $Gap(\boldsymbol{x}) = Gap(\boldsymbol{y})$.*

Proof. If $\boldsymbol{x}, \boldsymbol{y} \in O_{\boldsymbol{x}}$, then there exists $g_\pi \in G(n)$, such that $\boldsymbol{x} = g_\pi(\boldsymbol{y})$. Let $\boldsymbol{c}^* \in \arg\min Gap(\boldsymbol{x})$.

$$Gap(\boldsymbol{x}) = \boldsymbol{c}^{*T}\boldsymbol{x} = g_\pi(\boldsymbol{c}^*)^T g_\pi(\boldsymbol{x}) = Gap(\boldsymbol{y}).$$

The first equation holds by definition, while the second equation remains unchanged even if both terms of the scalar product are permuted. The final equation is derived from the fact that if there exists a $\boldsymbol{c}'$ value such that $\boldsymbol{c}'^T\boldsymbol{x}' < \boldsymbol{c}^{*^T}\boldsymbol{x}$, then $g_\pi^{-1}(\boldsymbol{c}')$ would provide a solution of lower cost than $\boldsymbol{c}^{*^T}$, thus rendering the latter non-optimal. □

Note that this result has already been proved in [12], using a different strategy. Finally, we add a result that justifies our symmetry-breaking simplex algorithm presented in the next section.

Definition 3. The set of all vertices adjacent to $\boldsymbol{x} \in \mathcal{X}^n_{ASEP}$ is called the **neighborhood of $\boldsymbol{x}$** and is denoted by $\mathcal{N}(\boldsymbol{x})$.

Lemma 7. *Let $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{X}^n_{ASEP}$, $\pi \in S_n$. Then, $\boldsymbol{y} \in \mathcal{N}(\boldsymbol{x}) \Leftrightarrow g_\pi(\boldsymbol{y}) \in \mathcal{N}(g_\pi(\boldsymbol{x}))$.*

Proof. First, we use the transformation π to sort the rows of the constraint matrix, by mapping each degree constraint associated with i to $\pi(i)$, and each subtour elimination constraint associated with $S = \{s_1, \ldots, s_f\}$ to $\pi(S) := \{s_1, \ldots, s_f\}$ and each non-negative constraint accordingly. With a slight abuse of notation, we will call $\pi(i)$ the mapping of the i^{th} constraint. Note that $\boldsymbol{y}$ shares exactly $r(n)-1$ linearly independent and tight constraints with $\boldsymbol{x}$, let $i_1, \ldots, i_{n_y-1}, i_{n_y}$ are this set of constraints of $\boldsymbol{y}$ and $i_1, \ldots, i_{n_y-1}, i_{n_x}$ is the one of $\boldsymbol{x}^3$, then, $\pi(i_1), \ldots, \pi\left(i_{n_y-1}\right), \pi\left(i_{n_y}\right)$ is associated to $g_\pi(\boldsymbol{y}$ and $\pi(i_1), \ldots, \pi\left(i_{n_y-1}\right), \pi(i_{n_x})$ is hence associated with $g_\pi(\boldsymbol{x})$. Thus, $g_\pi(\boldsymbol{y}) \in \mathcal{N}(g_\pi)(\boldsymbol{x}))$. □

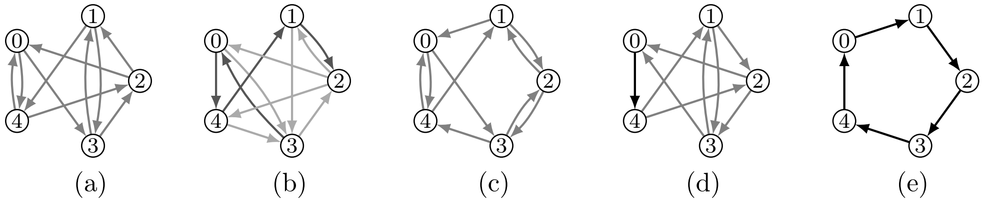


Figure 2: Support graphs of the 5 isomorphism classes for $n = 5$. The arcs have a grey level depending on the value of the corresponding x_i. See Table 2 for details: while (a) and (c) have all $x_i = \frac{1}{2}$ (light grey); in the support graph (b) 4 arcs correspond to $x_i = \frac{2}{3}$ (dark grey) and 7 to $x_i = \frac{1}{3}$; (d) 8 arcs have $x_i = \frac{1}{2}$ and 1 arc $x_i = 1$; (e) all $x_i = 1$.

Substantially, Lemma 7 states that the maps g_π preserve the adjacency of vertices. In other words, vertices belonging to the same orbit are equivalent also in terms of their neighborhoods.

4. Computing vertices with an large integrality gap

The orbits of the vertices of P^n_{ASEP} introduced in the previous section are here used to design a computational strategy for heuristically generating vertices of P^n_{ASEP}. In the next paragraphs, first, we introduce our pivoting algorithm that exploits the vertex symmetries to avoid the exploration of isomorphic vertices. Then, we introduce a new iterative procedure that generates vertices of P^{n+1}_{ASEP} starting from vertices of P^n_{ASEP}.

4.1. Pivoting by symmetry-breaking

In this subsection, we illustrate the new pivoting algorithm, which attempts to avoid the exploration of new vertices that are isomorphic to vertices already visited. The pivoting algorithm will be denoted by Pivoting($\boldsymbol{x}, T$), where the meaning of the variable T will be clarified later, and it is described in the following.

The main idea is simple: we start with a basic feasible solution, that is a vertex $\boldsymbol{x} \in P^n_{ASEP}$, and we explore all vertices in the neighborhood $\mathcal{N}(\boldsymbol{x})$ one at a time, by enumerating (or by sampling) the possible pivoting steps for that vertex. If the new vertex obtained by pivoting is not isomorphic to any vertex already explored, then we solve the optimization problem (9)–(15) to find the maximal integrality gap for that new vertex, and we record the corresponding orbit. We iterate the neighborhood search either with a complete enumeration for small values of n (e.g., $n \leq 8$), or with a random sampling strategy for $n > 8$. Our procedure is iterative, namely, it continues to iterate vertex-by-vertex, exploring each time the neighborhood of each vertex. The input parameters are:

- M, the maximum number of iterations of the algorithm, equivalent to the maximum number of vertices we pivot on.
- T_{tot}, timelimit for the whole iterations.
- T_{it}, timelimit for a single iteration.

Parameters T_{it} balance the tradeoff between *exploration* and *exploitation.* Small values of T_{it} allow for exploring only a few elements adjacent to a given vertex $\boldsymbol{x}$ and quickly moving on to the next neighboring vertex to be explored. On the other hand, high values of T_{it} insist heavily on the neighborhood of a vertex $\boldsymbol{x}$. We continue to iterate the procedure until either the timelimit T_{tot} is hit or the maximum number M of iteration is reached.

We named this algorithm "explore/exploit" for two reasons: the parameter T_{it} governs the exploitation of a single vertex. For small values of T_{it}, we do not focus much on the neighborhood of a single vertex, instead preferring to pivot on multiple vertices. However, for large values of T_{it}, we continue to build a single $\mathcal{N}(\boldsymbol{x})$, having less time to explore different areas of $P^n_{_{ASEP}}$.

Hence, given a time limit of T, our function

$$\mathcal{N}'(\boldsymbol{x}) = \text{Pivoting}(\boldsymbol{x}, T) \tag{20}$$

returns only a subset of $\mathcal{N}(\boldsymbol{x})$, namely the adjacent vertices that the strategy is capable of finding within a time limit.

4.2. Generating vertices using loop breaking procedure

In this subsection, we present our new iterative algorithm to compute a vertex of $P^{n+1}_{_{ASEP}}$ by starting from a vertex of $P^n_{_{ASEP}}$. First, we recover two definitions and a lemma from [12] that we use to prove our new result. Then, we introduce our loop-breaking procedure.

Definition 4 ([12], Chap. 3). Let $S \subset V$ and $\boldsymbol{x} \in P^n_{_{ASEP}}$. If $\delta(\boldsymbol{x}(S)) := \sum_{(i,j)\in\delta(S)} x_e = 1$, then, S is called a **tight set**.

Note that a tight set is a subset of vertices that induces a cut that satisfies a subtour elimination constraint with equality.

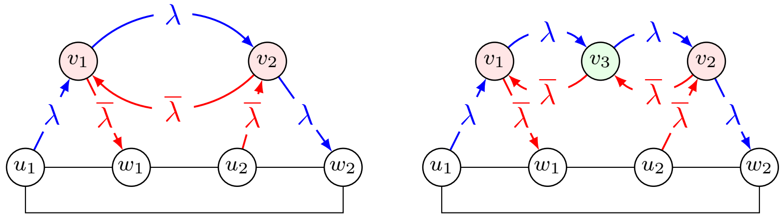
Algorithm 1: Generating vertices of $P^n_{_{ASEP}}$ trough the explore-exploit algorithm.

```
Input: $n$, Number of nodes of the ATSP instance under study
Input: $R := \{\boldsymbol{x}_i\}_{i \in I}$, list of vertices available for $n-1$
Input: $M$, Maximum number of iteration of algorithm
Input: $T_{tot}$, time limit for the whole iteration
Input: $T_{it}$, timelimit for the single iteration
Output: $R'$, a collection of non isomorphic vertices of $P^n_{_{ASEP}}$
$R'$ = Extend($R$)
$R'$ = SortByNumberOfZeros($R'$)
$i = 0$
$\boldsymbol{x}_0 = R'[i]$
$Ps = []$
while $i < M$ and time $< T_{tot}$ do
    if $\boldsymbol{x}_0$ is not isomorphic to any vertex in $Ps$ then
        $Ps$.append($\boldsymbol{x}_0$)
        $\mathcal{N}'(\boldsymbol{x_0})$ = Pivoting($\boldsymbol{x}_0, T_{it}$)
        for $\boldsymbol{y} \in \mathcal{N}(\boldsymbol{x_0})$ do
            if $\boldsymbol{y}$ is not isomorphic to any vertex in $R'$ then
                $R'$.insert($\boldsymbol{y}$)
            end
        end
        $i \leftarrow i + 1$
        $Ps$.append($R'[i]$)
    end
end
return $R'$
```

Definition 5 ([12], Chap. 3). Let $S \subset V$ be a tight set and $\boldsymbol{x} \in P^{n+|S|-1}_{_{ASEP}}$. Then, we can collapse the set S into node w as follows:

$$(\boldsymbol{x} \downarrow_w (S))_{uv} = \begin{cases} \sum_{s \in S} x_{us} & \text{if } v = w, \\ \sum_{s \in S} x_{sv} & \text{if } u = w, \\ x_{uv} & \text{otherwise.} \end{cases}$$

Lemma 8 ([12], Prop. 3.3.1)**.** *Let $\boldsymbol{x} \in P^n_{_{ASEP}}$ and $S \subset V$ be a tight set of $\boldsymbol{x}$. Then, $\boldsymbol{x} \downarrow_w (S)$ belongs to $P^{n-|S|+1}_{_{ASEP}}$.*



(a) Vertex $\boldsymbol{x} \in P^n_{ASEP}$.

(b) New vertex $\boldsymbol{x} \in P^{n+1}_{ASEP}$.

Figure 3: Example of λ-loop breaking $(\boldsymbol{x} \uparrow^{v_3} (v_1, v_2))_{uv}$, with $\overline{\lambda} = 1 - \lambda$.

We are now ready to formally introduce λ-loops.

Definition 6 (λ-loop). Let $\boldsymbol{x} \in P^n_{ASEP}$ and let $v_1, v_2 \in V$ such that $x_{v_1 v_2} = \lambda$ and $x_{v_2 v_1} = 1 - \lambda$, then, we say that $\boldsymbol{x}$ **contains a λ-loop** $\overleftrightarrow{v_1 v_2}$.

The following trivially follows.

Lemma 9. *Consider $\boldsymbol{x}$ a point in P^n_{ASEP} containing the λ-loop $\overleftrightarrow{v_1 v_2}$, then $S = \{v_1, v_2\}$ is a tight set.*

Proof. The amount of flow entering v_1 is λ, while the amount of flow entering v_2 is $1 - \lambda$. Hence, $\boldsymbol{x}(\delta(S)) = 1$. □

4.2.1. A new λ-loop breaking procedure

Our algorithm for generating vertices of P^{n+1}_{ASEP} from P^n_{ASEP} exploits λ-loop in the following λ-loop breaking procedure.

Definition 7. Let $\boldsymbol{x} \in P^n_{ASEP}$ that contains a λ-loop $\overleftrightarrow{v_1 v_2}$. Then, the **$\lambda$-loop breaking procedure** generate a point in $\mathbb{R}^{(n+1)n}$ adding the node v_3 as follows :

$$(\boldsymbol{x} \uparrow^{v_3} (v_1, v_2))_{uv} = \begin{cases} \lambda & \text{if } uv \in \{v_1 v_3,\ v_3 v_2\}, \\ 1 - \lambda & \text{if } uv \in \{v_3 v_1,\ v_2 v_3\}, \\ 0 & \text{if } uv \in \{v_1 v_2,\ v_2 v_1\} \\ 0 & \text{if } v = v_3 \text{ and } u \notin \{v_1, v_2\} \\ x_{uv} & \text{otherwise.} \end{cases}$$

The λ-loop breaking procedure has the following properties.

Lemma 10. *$S = \{v_1, v_3\}$ is a tight set for $\boldsymbol{x} \uparrow^{v_3} (v_1, v_2)$.*

Proof. Let $\boldsymbol{x}' = \boldsymbol{x} \uparrow^{v_3} (v_1, v_2)$

$$\boldsymbol{x}'(\delta(S)) = \sum_{\substack{b \in v \\ b \neq v_3}} x_{v_1 b} + \sum_{\substack{b \in r \\ b \neq v_1}} x_{v_3 b}$$

given that

$$\begin{aligned} &\sum_{b \in V} x_{v_1 b} = 1 \Rightarrow \sum_{\substack{b \in V \\ b \neq v_3}} x_{v_1 b} = 1 - x_{v_1 v_3} = 1 - \lambda, \\ &\sum_{b \in V} x_{v_3 b} = 1 \Rightarrow \sum_{\substack{b \in V \\ b \neq v_1}} x_{v_3 b} = 1 - x_{v_3 v_1} = 1 - (1 - \lambda) = \lambda, \\ &\Rightarrow \boldsymbol{x}'(\delta(S)) = 1 - \lambda + \lambda = 1. \end{aligned}$$

□

Lemma 11. *If $\boldsymbol{x}$ is a vertex of P^n_{ASEP} that contains a λ-loop $\overleftrightarrow{v_1 v_2}$, then $\boldsymbol{x} \uparrow^{v_3} (v_1, v_2)$ is a vertex of P^{n+1}_{ASEP}.*

Proof. Let $\boldsymbol{x}' := \boldsymbol{x} \uparrow^{v_3} (v_1, v_2)$. Clearly $\boldsymbol{x}'$ is feasible. To show that is a vertex, assume by contradiction that there exists $\boldsymbol{y}', \boldsymbol{z}' \in P^{n+1}_{ASEP}$ and $\mu \in (0, 1)$ such that $\boldsymbol{x}' = \mu \boldsymbol{y}' + (1 - \mu)\boldsymbol{z}'$. Let $S = \{v_1, v_3\}$. Note that

$$\boldsymbol{x}'(\delta(S)) = 1 = \mu \boldsymbol{y}'(\delta(S)) + (1 - \mu)\boldsymbol{z}'(\delta(S)). \tag{21}$$

As $\boldsymbol{y}', \boldsymbol{z}'$ are feasible, we have $\boldsymbol{y}'(\delta(S)) \geq 1$ and $\boldsymbol{z}'(\delta(S)) \geq 1$. As the equality should hold, we have then $\boldsymbol{z}'(\delta(S)) = \boldsymbol{y}'(\delta(S)) = 1$. Thus, we have

$$\boldsymbol{y} :=\downarrow_{v_1} \boldsymbol{y}'(S) \quad \text{ and } \quad \boldsymbol{z} :=\downarrow_{v_1} \boldsymbol{z}'(S).$$

Thanks to Lemma 8, $\boldsymbol{z}, \boldsymbol{y} \in P^n_{ASEP}$. Clearly, $\boldsymbol{x} = \mu \boldsymbol{y} + (1 - \mu)\boldsymbol{z}$. This can be verified entry by entry. As an example, consider the case $e = uv_1$:

$$\begin{aligned} \mu y_{uv_1} + (1 - \mu) z_{uv_1} &= \mu \left(y'_{uv_1} + y'_{uv_3} \right) + (1 - \mu) \left(z'_{uv_1} + z'_{uv_3} \right) \\ &= \left[\mu y'_{uv_1} + (1 - \mu) z'_{uv_1} \right] + \left[\mu y'_{uv_3} + (1 - \mu) z'_{uv_3} \right] \\ &= x'_{uv_1} + x'_{uv_3} = x_{uv_1}. \end{aligned}$$

Thus, $\boldsymbol{x}$ is not a vertex of P^n_{ASEP}, and we get a contradiction. □

Remark 1. Note that the converse unfortunately does not hold, e.g., if you collapse a λ-loop into a single not, you are not guaranteed to obtain a vertex. This will be particularly relevant in Section 5.4 where we collapse instead λ-loops: after doing that, we have to check that what we obtain is again a vertex.

4.2.2. How we used this procedure

If we are able to compute an initial vertex $\boldsymbol{x} \in P^n_{ASEP}$ having a $\lambda-$loop, then, we can iteratively apply the λ-loop breaking procedure to obtain a sequence of vertices $\boldsymbol{x}_k, \boldsymbol{x}_{k+1}, \dots, \boldsymbol{x}_{k+t}$ that belongs to $P^{n+1}_{ASEP}, P^{n+2}_{ASEP}, \dots, P^{n+t}_{ASEP}$, respectively.

Despite being simple, this iterative procedure is effective in generating ATSP instances with a large integrality gap, as we show in Section 5.1. Note that a similar idea was explored in [12], where the author introduced a *2-jack insertion* procedure, where a node satisfying precise hypotheses is replaced with a λ-loop, with the strict condition $\lambda = \frac{1}{2}$. Our strategy is more general since we allow any value of $\lambda \in (0, 1)$. Our experimental results show that our procedure is very effective in finding vertices with large integrality gaps (see Section 5.1). Note that the reverse move, namely, making a λ-loop *collapse* into one single point, does not always lead to a vertex, but it always leads to a feasible point thanks to Lemma 8.

4.3. The full algorithm

For a formal description of the proposed procedure, see Algorithm 1. The Pivoting function mentioned in line 9 of Algorithm 1 is the one of Equation (20). The whole procedure can be described in words as follows. Starting from $n = 5$ and the full collection of non-isomorphic vertices of P^5_{ASEP} (See Figure 2) that we can exhaustively generate using the software Polymake [3]:

1. We apply the procedure of λ-loop break obtaining vertices of P^{n+1}_{ASEP}.
2. We complete each of these vertices with the slack variables.
3. We order all the vertices found in this way by the number of zeros, from the one with the fewest zeros to the one with the most zeros.
4. Starting from the first one, we begin to apply the Pivoting strategy as described in Section 4.1 and Equation 20. Specifically, we enumerate all possible combinations of variables that can form a feasible basis and attempt to include each of the nonbasic variables in the basis.
5. When we reach the time limit T_{it}, we have a subset of adjacent vertices denoted as $\mathcal{N}'(\boldsymbol{x}) \subseteq \mathcal{N}(\boldsymbol{x})$. At this point, two things can occur:
   (a) $\mathcal{N}'(\boldsymbol{x}) = \emptyset$: in this case, we move on to the next vertex in the ordered list and start again from step 4.
   (b) $\mathcal{N}'(\boldsymbol{x}) \neq \emptyset$: in this case, we take each vertex from this set and add it to the input list - if does not belong to any of the orbits already explored - in order to maintain the list sorted by the number of zeros. Then, we start again from step 4.

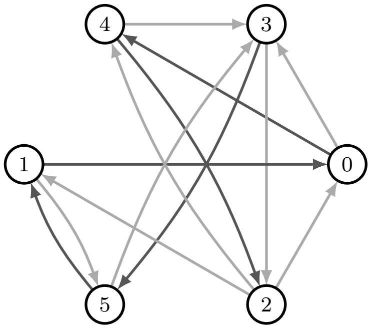


Figure 4: Starting point for the pivoting algorithm for $n = 6$.

6. We continue iteratively until either T_{tot} or M are reached.

Note that, thanks to Lemma 7, we do not need to pivot twice on isomorphic vertices, as they share the same neighborhood. In the remainder of this section, we show step-by-step how the algorithm described in Section 4 works in practice from $n = 5$ to $n = 6$. First, we need a starting vertex. To do so, we recover it using the breaking loop procedure from $n = 5$.

This gives us a representative for 6 different orbits and, among them, we choose the one with the smallest number of zeros. Figure 4 reports its support graph. Such vertex has entries equal to $\frac{k}{3}$, $k \in \{0, \dots, 5\}$. We start by pivoting from this vertex and collecting all its neighbors. Then, among its neighbors, we move to the one having the smaller number of zeros, and we proceed iteratively. Differently from the pivoting pipeline presented in [12], we can recover at least one representative for each orbit.

5. Computational results

This section shows the results of our experiments, focusing on the loop-breaking procedure's impact on expanding loops and the effect on the integrality gap. Additionally, it examines the neighborhood and the stabilizer of vertices for small values of n, focusing on the relation with the integrality gap.

Implementation details.. All the experiments run on an x86_64 architecture with a 13th Gen Intel(R) Core(TM) i5-13600 processor, offering 20 CPU cores with 32-bit and 64-bit operation modes. The pivoting algorithm is implemented in C++, while the iterative procedure is in Python. We use the Eigen library [17] to deal with matrices and MPFR [15] for the operations in multiple precision. The integrality gap is computed by solving the linear program (9)–(15) using Gurobi v9.5.0 [18].

5.1. Combining the symmetry breaking pivoting and the λ-loop breaking procedure

Our algorithm has some strengths and weaknesses.

One of the strengths is that compared to the Pivoting algorithm introduced by [12], for $n = 6$, we manage to quickly recover at least one representative for all orbits. Another strength is the impact of the λ-loop breaking procedure in quickly identifying vertices with a large integrality gap. Figure 5 illustrates this phenomenon in the transition from $n = 6$ to $n = 7$. Experimentally, we observe that the integrality gap of the vertices obtained through λ-loop breaking is always greater or equal to the starting one. Another strength is that the λ-loop breaking procedure alone is capable of finding the vertex with the maximum integrality gap for each n. For example, in Figure 7, it can be seen that the vertex with the maximum integrality gap was obtained at iteration 0, implying that the rest of the search, although leading to many points with large integrality gaps, is subordinated to what is obtained at step 0.

Among the weaknesses, we observe that although the solutions we found with the λ-loop loops procedure are associated with large integrality gaps, they have a lot of zeros. In fact, moving from n to $n + 1$, we switch from considering points from dimension $n(n-1)$ to points in $\mathbb{R}^{(n+1)n}$, adding hence $2n$ entries. All of them are 0, but 2. So we add $2n - 2$ zeros to our vertices. Hence, it is hard to explore the full neighborhood, due to the great amount of feasible basis. Lastly, we were able to push our pivoting procedure until $n = 11$. After that, it becomes infeasible to even partially enumerate at least one neighborhood of the vertex. Figure 7 illustrates how the duration of each iteration increases and how the number of new orbits found decreases with each iteration.

5.2. Neighborhood exploration

This section is devoted to studying the neighborhood of some vertices of $n \in \{4, 5, 6\}$ and deducing local properties. Before diving into the details, let us recall the definition of a polyhedral graph.

Definition 8 (Polyhedral graph). A polyhedral graph is an undirected graph formed from the vertices and edges of a convex polyhedron.

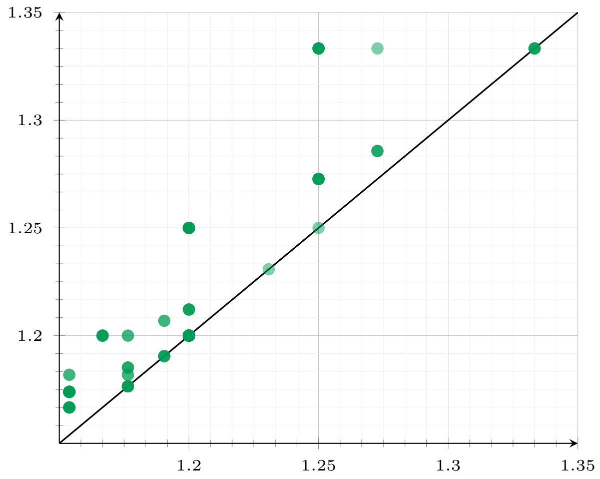


Figure 5: On the x axis, we represent the integrality gap of the vertices for $n = 6$ that have a λ-loop. For each of these vertices, we plot the integrality gap of the vertices obtained by the λ-loop breaking procedure on the y axis. The fact that all points lie (non-strictly) above the line $y = x$ implies that the λ-loop breaking procedure is highly effective in increasing the integrality gap.

Table 1: Orbit structure for $n = 4$. Columns: cardinality of the orbit, type of components, frequency of each component, and integrality gap attained at the elements of that orbit.

$\lvert O_{\boldsymbol{x}} \rvert$	Components		Frequencies		IG
6	0	$\frac{1}{2}$	4	8	6/5
6	0	1	8	4	1

For $n = 4$, we recovered using Polymake [3] all the vertices have been enumerated in [12] using PORTA [10]. More specifically, we have 12 vertices in total and two orbits. With this small number of vertices and orbits is hence easy to exhaustively study the neighborhood of each point. Figure 6 show the polyhedral graph of P^4_{ASEP}. Nodes from 0 to 5 represent the non-integer vertices, while nodes from 6 to 11 represent the tours. We can observe that each non-integer vertex has among the adjacent vertices, always an integer one. Interestingly, each tour is connected to all the vertices but one.

For $n = 5$, we recover 384 vertices, as already done in [12]. We choose a representative for each class and study its neighborhood. This can be done w.l.o.g thanks to Lemma 7. As expected, the number of neighbors is related to the degeneracy of the vertex, that is number of zeros among both

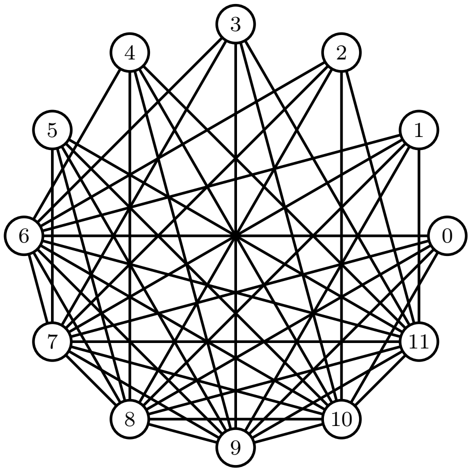


Figure 6: Polyhedral graph for P^4_{ASEP}.

Table 2: Orbit structure for $n = 5$. Columns: Label the orbits according to Figure 2, Cardinality of the orbit, type of components, frequency of each component, integrality gap attained at the elements of that orbit, number of tight sets, neighborhood size, and stabilizer.

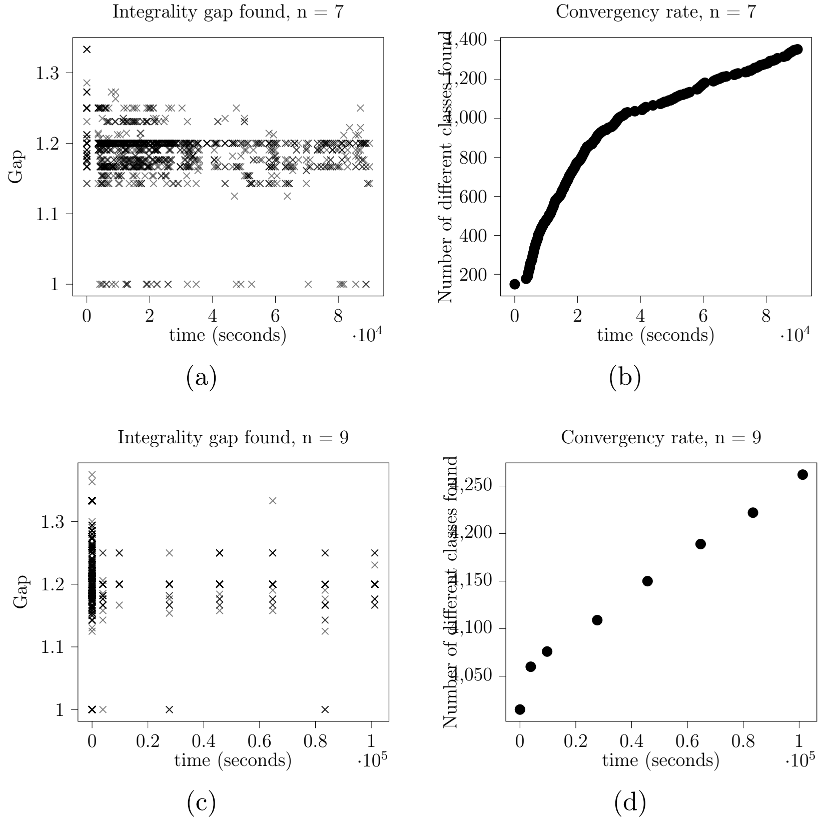
	$\lvert O_{\boldsymbol{x}}\rvert$	Components			Frequencies			IG	tight sets	$\lvert\mathcal{N}(\boldsymbol{x})\rvert$	$G_{\boldsymbol{x}}$
(a)	60	0	6/5		10	10		5/4	6	28	$\langle(0\ 1)(4\ 3)\rangle$
(b)	120	0	$\frac{1}{3}$	$\frac{2}{3}$	9	7	4	6/5	4	20	$\langle id\rangle$
(c)	60	0	$\frac{1}{2}$		10	10		6/5	4	28	$\langle(0\ 4)(1\ 3)\rangle$
(d)	120	0	1	$\frac{1}{2}$	11	1	8	6/5	4	23	$\langle id\rangle$
(e)	24	0	1		15	5		1	10	148	$\langle(0\ 1\ 2\ 3\ 4)\rangle$

arc and slack variables. Interestingly, each vertex has at least one adjacent representative for each equivalence class. Furthermore, the vertices (d) and (e) also have a representative of themselves among the neighbors.

In the case of $n = 6$, we have 90 orbits. The orbit structure can be found in Table A.5 and A.6, in the appendix. Neighborhoods of these vertices cannot be exhaustively explored. When the number of zeros is close to 18, the exhaustive listing of the whole neighborhood quickly leads to an out-of-memory error.

Note that, from our preliminary test, we state the following conjecture:

Conjecture 1. *For each $\boldsymbol{x}$ vertex, $\mathcal{N}(\boldsymbol{x})$ contains at least one tour.*



(a)

(b)

(c)

(d)

Figure 7: Left: time vs integrality gap found. The darker the ×, the bigger the number of vertices found leading to that integrality gap. We note that the highest integrality gap is found at the very beginning of the procedure. Right: Time vs Number of classes of isomorphism. We note that for $n = 7$ the algorithm continuously finds new and new vertices; for $n = 9$, new vertices are more and more rare.

If this conjecture were true, solely pivoting on the integer vertex would lead to the full V-description of the polytope: by using Lemma 7 the neighborhood of an integer vertex will contain at least one representative of each orbit. We can then list all the vertices by fully describing each orbit. This can be done just *theoretically*, as in practice, listing the full neighborhood of a vertex with just 6 nodes is infeasible.

5.3. Symmetries and conjectured relation with the integrality gap

Table 1, 2, A.5, A.6 show the orbit structure of $n = 4, 5, 6$. In all cases, we observe that the maximum integrality gap has been attained at one-half integer vertex with a relatively small orbit.

Table 1 reports the structure of the orbit we found for $n = 4$. As there is only one half-integer solution, the maximum integrality gap is hence attained at that vertex.

For $n = 5$, we have represented all the vertices in Figure 2. The one attaining the highest integrality gap is vertex (a). Looking at Table 2, we understand that vertex (a) and (c) have the same number of non-zeros and zeros entries. The main differences lie in the number of λ-loops and tight sets (see Definition 4). Tight sets are associated with slack variables $\overline{x}_S = 0$. In particular, vertex (a) has 6 tight sets, while vertex (c) has only 4. For $n = 6$, there are 90 orbits, see Table A.5 and A.6. The vertex attaining the maximum integrality gap is again half-integer, which in principle does not seem very different from other half-integer vertices having low integrality gap (See, e.g, the middle of Table A.6). We observe that all the half-integer vertices have 10 tight sets. The two vertices leading the two highest integrality gaps have instead a higher number of tight sets (Line 1 and 3 of Table A.5). More specifically, the two half integer vertices maximizing the integrality gap have, respectively, 10 and 12 tight sets.

For these small number of nodes, we also explicitly compute the stabilizer. For $n = 4$, the stabilizer of any representative of the non-integer orbit is given by the vertices of the subcycles in the support graph. For instance, referring to Figure 1, left, it holds $G_x = \langle (0\ 3\ 1\ 2) \rangle$.

All the stabilizers are trivial for $n = 5$, but the ones of the two half-integer vertices. Table 2 reports all the stabilizers explicitly computed. The vertices attaining the maximum gap are the ones whose stabilizer is isomorphic to $\mathbb{Z}_2$ and, more specifically, it acts swapping the extreme of the two cycles. For (c), the situation is analogous, by considering the two chained λ-loops $\overleftrightarrow{03}$ and $\overleftrightarrow{04}$ as one.

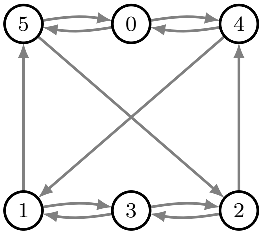


Figure 8: Vertex obtaining the maximum integrality gap for $n = 6$.

For $n = 6$, the vertex attaining the maximum integrality gap has been represented in Figure 8. The stabilizer of this vertex is isomorphic to $\mathbb{Z}_4$, more specifically is the group generated from (03)(1425). Even in this case, the stabilizer "swaps" the extreme nodes of the two chained λ - loops and the middle nodes 0 and 3.

For $n = 7$, our heuristic procedure does not recover all the vertices, hence we can do considerations only among the ones we were able to obtain, which are 1356 out of 3748. According to [12], 5 different isomorphism classes are attaining the maximum gap of $\frac{4}{3}$: our heuristic finds all of them. As already observed in [12], 3 out of 5 vertices have entries in $\{0, 0.5\}$ (from now on, we will denote vertices having entries in $\{0, 0.5\}$ as *half-integer*) while the others in $\{0, 0.5, 1\}$ (*integer-half-integer*). Even in this case, the number of tight sets in vertices maximizing the gap is high, namely, 12 and 16, although not maximal, as there exist pure half-integer vertices having 14 and 16 tight sets. A similar argument holds for half-integer vertices.

For $n = 8$, we identified 41 vertices with a maximum integrality gap of $\frac{4}{3}$. In [12], there were 43 of such vertices, but one of them was not half-integer. Observe that their method was specifically tailored to find *all* the half-integer vertices and *potentially* other, whereas our approach allows for more flexibility. Specifically, we discovered 17 pure half-integer orbits, 16 half-orbits, and 8 orbits with components in $\{0, 0.25, 0.5, 0.75\}$. While this suggests the possibility that non-half-integer vertices may also result in the maximum integrality gap, it appears to be an isolated case.

More specifically, for $n \geq 9$, the vertex attaining the maximum gap is always unique and a pure half-integer vertex. Based on our recent discussion, we have gathered the following empirical evidence:

- Among the vertices achieving the maximum integrality gap, there is at least one half-integer.

- Furthermore, vertices with a large stabilizer and a large number of tight sets appear to maximize the integrality.

5.4. New lower bound on the integrality gap

Table 3 presents the results of the combined symmetry-breaking pivoting and λ-loop breaking algorithm. Starting from a vertex of P^6_{ASEP}, our combined algorithm alternates the exploration of vertices of P^n_{ASEP} with the generation of a new vertex of P^{n+1}_{ASEP}

With our combined algorithm, we can recompute all the lower bounds of α^{LB}_n up to $n = 15$, and we compute newer lower bounds for $n \in \{16, 18, 20, 22\}$ Note that for n odd, [12] introduces a family of ATSP instances having $\alpha_n = \frac{3k+1}{2k+1}$ where $n = 3 + 2(k + 1)$, which gives new lower bounds for $n = 17, 19, 21$. However, all the lower bounds in [12] are obtained only by exploring half-integer vertices, while our procedure can generate non-half-integer vertices, thanks to the λ-loop breaking procedure. However, except for the case $n = 8$, where we found a non-half integer vertex maximizing the gap, such a maximum is always achieved in correspondence with a half-integer vertex.

To obtain Table 3, we proceed as follows: starting from a vertex explicitly given by [12] for $n = 18$, having an integrality gap of 1.5, we make each λ-loop *collapse* to one single node, and check time by time if the so-obtained *fesible* point is a vertex. Afterward, we expand each λ-loop obtaining vertices for $19 \leq n \leq 22$. These two procedures built Table 3, where the lower bounds have been improved with respect to the state of the art for $n \in \{16, 17, 19, 20, 21, 22\}$. We recall that, although our exploration allows different types of fractional vertices, the maximum gap is always attained on a half-integer vertex.

5.5. Hard-ATSPLIB instances

Whenever we solve the Gap($\boldsymbol{x}$) problem to compute the maximum integrality gap for a given vertex, we generate an ATSP instance which could be challenging in practice for the state-of-the-art solver Concorde [1]. Hence, we have saved several small hard ATSP instances, which we share online at `https://github.com/eleonoravercesi/HardATSPLIB`. Several studies in the literature have examined the empirical hardness of the STSP concerning the integrality gap, such as [20, 31, 30]. However, to the best of our knowledge, no such studies have been conducted on the ATSP. Note that every time we compute the maximum possible integrality gap attained at a given

Table 3: State of the art on the lower bounds $\alpha_n^{LB} \leq \alpha_n$ for ATSP, with $11 \leq n \leq 22$. In bold, the new best lower bounds are obtained with our approach.

n	From [12]	New	n	From [12]	New
11	10/7 (1.429)	10/7 (1.429)	17	19/13 (1.461)	**55/37 (1.486)**
12	56/39 (1.436)	56/39 (1.436)	18	3/2 (1.500)	3/2 (1.500)
13	13/9 (1.444)	13/9 (1.444)	19	22/15 (1.466)	**3/2 (1.500)**
14	100/69 (1.449)	100/69 (1.449)	20	-	**3/2 (1.500)**
15	16/11 (1.454)	16/11 (1.454)	21	25/17 (1.470)	**3/2 (1.500)**
16	-	**28/19 (1.474)**	22	-	**3/2 (1.500)**

Table 4: Results of 10 runs of Concorde with 10 different seeds and different modalities. Columns represent the number of nodes of the ATSP, the mean of the runtimes and the B&B nodes, and the same values for an instance of the TSPLIB having $2n$ number of nodes.

n	Standard run		DF		No local cuts		DF and No local cuts	
	Time (s)	B&B nds	Time (s)	B&B nds	Time (s)	B&B nds	Time (s)	B&B nds
7	0.7	1.0	0.7	1.0	0.1	7.0	0.1	7.2
8	2.6	1.0	2.6	1.0	0.1	11.2	0.1	11.2
9	3.4	6.6	3.4	7.4	0.1	39.4	0.1	32.9
10	5.1	13.2	4.2	11.6	0.1	54.6	0.1	54.8
11	7.6	36.2	6.6	39.8	0.4	103.8	0.3	80.4
12	9.8	104.6	10.3	112.8	112.8	278.8	0.8	220.9
13	25.5	258.4	21.9	240.3	2.0	335.8	1.1	198.8
14	44.2	561.6	39.9	561.5	9.2	1468.6	4.3	778.0
15	57.1	661.2	42.8	516.4	7.9	1003.0	3.2	461.9
16	145.1	1551.0	92.5	1047.2	20.8	2112.0	6.0	773.4
17	259.9	2948.2	178.8	2152.7	35.1	2928.0	9.6	1104.5
18	548.7	6422.0	363.2	4678.0	246.5	14283.2	44.5	4321.7
19	1090.8	11675.8	668.8	8094.0	270.6	10398.0	57.7	4540.5

vertex $\boldsymbol{x}$ by solving problem (9)–(15), we obtain a cost vector $\boldsymbol{c}$ associated to an ATSP instance having ATSP($\boldsymbol{c}$) = 1. Hence, we can evaluate the computational complexity of each instance as generated in this way.

A core question is how to evaluate complexity from a computational perspective. Differently from the case of STSP, a native state-of-art solver is not available for ATSP. Previous work by [14] has shown promising results using a branch-and-cut algorithm that exploits facet-defining inequalities for ATSP. However, more recent studies [13, 26] suggest that Concorde, the state-of-art solver for the STSP, remains the most efficient method for solving ATSP. It is important to note that Concorde can only handle symmetric nonnegative and integer costs, but it is possible to transform any of the ATSP instances we obtained into an integer and non-negative STSP starting from the method proposed in [21]. First of all, we have observed that all the solutions we found solving $Gap(\boldsymbol{x})$ are rational, and hence it is possible to make the costs integer by multiplying all the entries by the common denominator. Hence, without loss of generality, we can consider all the solutions of $Gap(\boldsymbol{x})$ as integer vectors. Let $\overline{\boldsymbol{C}} = (\overline{c}_{ij})$ be a matrix derived from the cost vector as suggested in [21], namely:

$$\overline{c}_{ij} = \begin{cases} c_{ij} & i \neq j \\ -M & i = j, \end{cases} \tag{22}$$

where M denotes a large positive number. Consider the matrix $\boldsymbol{U}$, where all entries are set to infinity. We construct the following $\mathbb{R}^{2n\times 2n}$ matrix: We can create the following $\mathbb{R}^{2n\times 2n}$ matrix:

$$\tilde{\boldsymbol{C}} = \begin{bmatrix} \bar{\boldsymbol{C}} & \boldsymbol{U} \\ \boldsymbol{U} & \bar{\boldsymbol{C}} \end{bmatrix}.$$

Note that $\tilde{\boldsymbol{C}}$ may contain negative costs, which we do not want, as Concorde only performs with positive costs. Therefore, we shift all costs forward by M, namely making the minimum cost equal to 0. Unfortunately, in this framework, we only have *premetrics*, as we lose the triangle inequality in every triple involving two of the original nodes and one "doubled" node. However, since we have only performed an affine translation on each cost, the optimal tour does not change. The relationship between the original values of ATSP and STSP is hence

$$\text{ATSP}(\boldsymbol{c}) = \text{STSP}(\tilde{\boldsymbol{c}}) - nM. \tag{23}$$

Hence, we transform each ATSP into an STSP as discussed above, solve each instance using Concorde, and record the computational time and the integrality gap. Regarding the integrality gap, we observe that we only have a relation between the value of ATSP($\boldsymbol{c}$) and STSP($\tilde{\boldsymbol{c}}$), but nothing can be said a prior for SSEP($\tilde{\boldsymbol{c}}$) and ASEP($\boldsymbol{c}$). Remarkably, we observed that

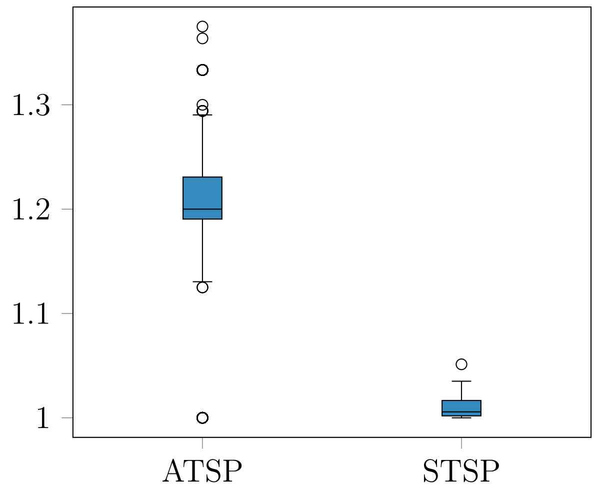


Figure 9: Distribution of the integrality gap from the ATSP to the STSP via the application of the Jonker-Volgenant-based procedure for $n = 9$.

after the above-discussed procedure, the resulting instances exhibit a reduced integrality gap. Figure 9 reports this information for $n = 9$.

In terms of computational complexity, we are hence able to retrieve some hard-to-solve instances. Table 4 reports the results for some hard instances with $7 \leq n \leq 19$ nodes generated by our approach. We do four different types of computation:

- We run Concorde as it is, 10 times with 10 different seeds and we record the runtime and the number of Branch & Bound (B&B) nodes. From now on, this would be called the "standard" setting.
- We add the flag `-d` flag that uses Deep-first (DF) branching instead of Breadth-First (BF). Adding this flag can prevent Concorde from writing search nodes to files, which could not be a good idea for small instances. Even in this case, we ran Concorde 10 times with 10 different seeds, collecting the results.
- We add the flag -C0 to disable local cuts. For small instances of the TSP, the computational overhead associated with generating and applying local cuts may outweigh the benefits they provide.
- We combine the flag -d -C0 together.

We compare both runtime and B&B nodes with the instances of the TSPLIB [25].

First, we observe that not using local cuts in small instances has a great benefit, as already known in the literature. First, we observe that not using local cuts in small instances has a great benefit. For instance, a 15-node instance from our library, when solved with Concorde "as it is" requires approximately one minute. Disabling local cuts can lead to a solution in less than 10 seconds. This time is further reduced if we prefer a DF strategy for branching. However, our instances still prove to be challenging even in their optimized version compared to instances from the TSPLIB and even with the hard instances introduced by [20]. By disabling local cuts and using DF instead of BF, Concorde takes less than 2 seconds to reach the optimal value with $n = 52$. Note that the STSP instances in the TSPLIB with less than 26 nodes, namely `burma14`, `ulysses16`, `ulysses22`, `gr24`, and `fri26`, and the ATSPLIB instance `br17` are all solved by Concorde in around 0.01 seconds with the standard setting.

6. Conclusions

In this paper, we have introduced and implemented a new symmetry-breaking pivoting algorithm and a new λ-loop breaking procedure that permits the exploration of vertices of the asymmetric subtour elimination polytope yielding a large integrality gap. The symmetry-breaking pivoting exploits the class of isomorphism of the vertices of P^n_{ASEP} that we completely calculated for a small value of n. Checking whether two vertices of P^n_{ASEP} are isomorphic is currently one of the two computational bottlenecks of our procedure. For each non-isomorphic vertex we visit, we solve an instance of the Gap($\boldsymbol{x}$) problem. With our new algorithm, we can compute new lower bounds for α_n^{LB} for $n \leq 22$ by exploring not only half-integer vertices.

In addition, we solved the instances yielding the largest integrality gap with Concorde, and comparing the runtime, it is clear that those instances are challenging for a state-of-the-art TSP solver.

In the future, we plan to explore the unresolved issue addressed in this study, which involves developing a procedure that yields the stabilizer based on a vertex and creating a strategy that leverages symmetries to produce vertices that are considered *noteworthy* in terms of the integrality gap. Note that for small values of n the integrality gap values returned by the two families of instances proposed and studied in [9, 12] are improved. For these two families, the integrality gap converges to 2. Therefore, if the improvement obtained in this paper for small values of n could be "uniformly" observed also for large values of n this would lead to an integrality gap greater than 2. Of course, this is just an intriguing hypothesis for future research.

Acknowledgments

The work of M. Mastrolilli, L. M. Gambardella and E. Vercesi has been supported by the Swiss National Science Foundation project n. 200021_212929 / 1 "Computational methods for integrality gaps analysis", Project code 36RAGAP. S. Gualandi acknowledges the contribution of the National Recovery and Resilience Plan, Mission 4 Component 2 - Investment 1.4 - NATIONAL CENTER FOR HPC, BIG DATA AND QUANTUM COMPUTING, spoke 6. We thank Giovanni Rinaldi for pointing out the interesting references [24] and [4]. We also thank the anonymous referees of IPCO2024 for having provided precious insights to improve this work.

References

[1] Applegate, D., Bixby, R., Cook, W., Chvátal, V., 1998. On the solution of traveling salesman problems. Documenta Mathematica , 645–656URL: `http://eudml.org/doc/233207`.

[2] Artin, M., 2011. Algebra. Pearson Education, Upper Saddle River, NJ.

[3] Assarf, B., Gawrilow, E., Herr, K., Joswig, M., Lorenz, B., Paffenholz, A., Rehn, T., 2017. Computing convex hulls and counting integer points with polymake. Math. Program. Comput. 9, 1–38.

[4] Balinski, M.L., Russakoff, A., 1974. On the assignment polytope. SIAM Review 16, 516–525. URL: `https://doi.org/10.1137/1016083`, doi:`10.1137/1016083`, `arXiv:https://doi.org/10.1137/1016083`.

[5] Benoit, G., Boyd, S., 2008. Finding the exact integrality gap for small Traveling Salesman Problems. Mathematics of Operations Research 33, 921–931.

[6] Boyd, S., Sitters, R., van der Ster, S., Stougie, L., 2011. Tsp on cubic and subcubic graphs, in: Integer Programming and Combinatoral Optimization: 15th International Conference, IPCO 2011, New York, NY, USA, June 15-17, 2011. Proceedings 15, Springer. pp. 65–77.

[7] Boyd, S.C., Elliott-Magwood, P., 2005. Computing the integrality gap of the asymmetric travelling salesman problem. Electron. Notes Discret. Math. 19, 241–247.

[8] Carr, R., Vempala, S., 2004. On the Held-Karp relaxation for the Asymmetric and Symmetric Traveling Salesman Problems. Mathematical Programming 100, 569–587.

[9] Charikar, M., Goemans, M.X., Karloff, H., 2006. On the integrality ratio for the Asymmetric Traveling Salesman Problem. Mathematics of Operations Research 31, 245–252.

[10] Christof, T., 2009. Porta. http://www.iwr.uni-heidelberg.de/groups/comopt/software/PORTA/index. html .

[11] Dantzig, G., Fulkerson, R., Johnson, S., 1954. Solution of a large-scale traveling-salesman problem. Journal of the operations research society of America 2, 393–410.

[12] Elliott-Magwood, P., 2008. The integrality gap of the Asymmetric Travelling Salesman Problem. Ph.D. thesis. University of Ottawa (Canada).

[13] Fischetti, M., Lodi, A., Toth, P., 2007. Exact methods for the Asymmetric Traveling Salesman Problem. The traveling salesman problem and its variations , 169–205.

[14] Fischetti, M., Toth, P., 1997. A polyhedral approach to the Asymmetric Traveling Salesman Problem. Management Science 43, 1520–1536.

[15] Fousse, L., Hanrot, G., Lefèvre, V., Pélissier, P., Zimmermann, P., 2007. MPFR: A multiple-precision binary floating-point library with correct rounding. ACM Transactions on Mathematical Software (TOMS) 33, 13–es.

[16] Grötschel, M., Padberg, M.W., 1985. The traveling salesman problem. John Wiley & Sons Ltd. chapter Polyhedral theory.

[17] Guennebaud, G., Jacob, B., 2010. Eigen v3. `http://eigen.tuxfamily.org`.

[18] Gurobi Optimization, LLC, 2023. Gurobi Optimizer Reference Manual. URL: `https://www.gurobi.com`.

[19] Hougardy, S., 2014. On the integrality ratio of the subtour LP for Euclidean TSP. Operations Research Letters 42, 495–499.

[20] Hougardy, S., Zhong, X., 2021. Hard to solve instances of the Euclidean Traveling Salesman Problem. Mathematical Programming Computation 13, 51–74.

[21] Jonker, R., Volgenant, T., 1983. Transforming Asymmetric into Symmetric Traveling Salesman Problems. Operations Research Letters 2, 161–163.

[22] Lawvere, F.W., 1973. Metric spaces, generalized logic, and closed categories. Rendiconti del Seminario Matematico e Fisico di Milano 43, 135–166.

[23] Mnich, M., Mömke, T., 2018. Improved integrality gap upper bounds for traveling salesperson problems with distances one and two. European journal of operational research 266, 436–457.

[24] Padberg, M., Rao, M., 1974. The travelling salesman problem and a class of polyhedra of diameter two. Mathematical Programming 7, 32–45.

[25] Reinelt, G., 1991. Tsplib—a traveling salesman problem library. ORSA journal on computing 3, 376–384.

[26] Roberti, R., Toth, P., 2012. Models and algorithms for the Asymmetric Traveling Salesman Problem: an experimental comparison. EURO Journal on Transportation and Logistics 1, 113–133.

[27] Singh, M., 2019. Integrality gap of the vertex cover linear programming relaxation. Operations Research Letters 47, 288–290.

[28] Traub, V., Vygen, J., 2020. An improved approximation algorithm for atsp, in: Proceedings of the 52nd annual ACM SIGACT symposium on theory of computing, pp. 1–13.

[29] Vazirani, V.V., 2001. Approximation Algorithms. volume 1. Springer, Berlin.

[30] Vercesi, E., Gualandi, S., Mastrolilli, M., Gambardella, L.M., 2023. On the generation of metric TSP instances with a large integrality gap by branch-and-cut. Mathematical Programming Computation 15, 389–416.

[31] Zhong, X., 2021. Lower Bounds on the Integraliy Ratio of the Subtour LP for the Traveling Salesman Problem. arXiv preprint arXiv:2102.04765 .

Appendix A. Detailed description of the orbits for for $n = 6$

Table A.5: Orbit structure for $n = 6$, top 21 having the highest integrality gap. Columns: cardinality of the orbit, type of components, frequency of each component, and integrality gap attained at the elements of that orbit.

$\lvert O_{\boldsymbol{x}} \rvert$	Components				Frequencies				IG
180	0	1/2			18	12			4/3
120	0	1/3	2/3		18	6	6		9/7
360	0	1/2			18	12			14/11
360	0	1/2			18	12			5/4
720	0	1/2			18	12			5/4
180	0	1/2			18	12			5/4
720	0	1	1/2		19	1	10		5/4
720	0	1	1/2		19	1	10		5/4
360	0	1	1/2		19	1	10		5/4
720	0	1/4	3/4	1/2	17	5	3	5	16/13
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	17	5	3	5	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	16	6	2	6	6/5
720	0	1/4	3/4	1/2	17	5	3	5	6/5

Table A.6: Orbit structure for $n = 6$, bottom 69 having the lowest integrality gap. Columns: cardinality of the orbit, type of components, frequency of each component, and integrality gap attained at the elements of that orbit.

$\lvert O_{\boldsymbol{x}} \rvert$	Components					Frequencies					IG
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
240	0	1/3	2/3			18	6	6			6/5
720	0	1/3	2/3			16	10	4			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			17	8	5			6/5
720	0	1/3	2/3			16	10	4			6/5
120	0	1/2				18	12				6/5
720	0	1/2				18	12				6/5
720	0	1/2				18	12				6/5
720	0	1/2				18	12				6/5
720	0	1/2				18	12				6/5
720	0	1	1/3	2/3		18	1	7	4		6/5
720	0	1	1/3	2/3		18	1	7	4		6/5
720	0	1	1/3	2/3		18	1	7	4		6/5
720	0	1	1/3	2/3		18	1	7	4		6/5
360	0	1	1/2			19	1	10			6/5
720	0	1	1/2			19	1	10			6/5
720	0	1	1/2			19	1	10			6/5
720	0	1	1/3	2/3		18	1	7	4		6/5
360	0	1	1/2			20	2	8			6/5
720	0	1	1/2			20	2	8			6/5
720	0	1	1/2			20	2	8			6/5
720	0	1/5	4/5	2/5	3/5	16	5	2	4	3	25/21
720	0	1/5	4/5	2/5	3/5	16	5	2	4	3	25/21
720	0	1/5	4/5	2/5	3/5	16	5	2	4	3	25/21
720	0	1/4	3/4	1/2		16	7	3	4		20/17
720	0	1/4	3/4	1/2		16	7	3	4		20/17
720	0	1/4	3/4	1/2		16	7	3	4		20/17
720	0	1/4	3/4	1/2		16	7	3	4		20/17
720	0	1/4	3/4	1/2		16	7	3	4		20/17
720	0	1/4	3/4			16	9	5			20/17
720	0	1/3	2/3			16	10	4			7/6
120	0	1/2				18	12				7/6
720	0	1/3	2/3			16	10	4			15/13
720	0	1/3	2/3			17	8	5			15/13
360	0	1/3	2/3			16	10	4			15/13
720	0	1/3	2/3			16	10	4			15/13
720	0	1/3	2/3			16	10	4			15/13
720	0	1/3	2/3			16	10	4			15/13
720	0	1/3	2/3			16	10	4			15/13
360	0	1/3	2/3			16	10	4			15/13
720	0	1/3	2/3			17	8	5			15/13
720	0	1/3	2/3			17	8	5			15/13
120	0	1/3	2/3			18	6	6			15/13
120	0	1				24	6				1