Integer Linear Programming reformulations for the linear ordering problem

Nicolas Dupin

▶ **To cite this version:**

Nicolas Dupin. Integer Linear Programming reformulations for the linear ordering problem. 2022. hal-03607145

HAL Id: hal-03607145
https://hal.science/hal-03607145

Preprint submitted on 12 Mar 2022

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Integer Linear Programming reformulations for the linear ordering problem

Nicolas Dupin[1]*[0000-0003-3775-5629]

Université Paris-Saclay, LISN, 91405, Orsay, France
nicolas.dupin@universite-paris-saclay.fr

Abstract. This article studies the linear ordering problem, with applications in social choice theory and databases for biological datasets. Integer Linear Programming (ILP) formulations are available for the linear ordering problem and some extensions. ILP reformulations are proposed, showing relations with the Asymmetric Travel Salesman Problem. If a strictly tighter ILP formulation is found, numerical results justify the quality of the reference formulation for the problem in the Branch& Bound convergence, and the quality of the continuous relaxation to design rounding heuristics. This work offers perspectives to design matheuristics for the linear ordering problem and extensions.

Keywords : Optimization; Integer Linear Programming; Linear ordering problem ; Median of permutations; Consensus ranking; Polyhedral analysis

1 Introduction

A bridge between optimization and Machine Learning (ML) exists to optimize training parameters of ML models, using continuous optimization and metaheuristics [17, 18]. Discrete and exact optimization, especially Integer Linear Programming (ILP), is also useful to model and solve specific variants of clustering or selection problems [5, 6]. In this paper, another application of ILP to learning is studied: the Linear Ordering Problem (LOP). LOP aims to define a common and consensus ranking based on pairwise preferences, which is used in social choice theory. If many applications deal with a small number of items to rank, bio-informatics applications solve large size particular instances of LOP as median of permutations [2, 3]. An ILP formulation is available for LOP with constraints defining facets [10, 11]. An extension of LOP considering ties relies on this ILP formulation [2]. Current and recent works focus for consensus ranking in biological datasets, and use specific data characteristics of these median of permutation problems for an efficient resolution [1, 13].

This paper aims to analyze the limit of state-of-the art ILP solvers to solve LOP instances for exact resolution. Several alternative ILP reformulations are designed using similarities and recent results on the Asymmetric Travelling Salesman Problem (ATSP) from [14]. Comparison of Linear Programming (LP) illustrates and validates the work on polyhedral analysis, as in [14]. The implication on the characteristics of ILP solving using modern ILP solvers is analyzed, as in [4], as well as quality of LP relaxation for variable fixing matheuristics, as in [7].

2 Problem statement and reference ILP formulation

LOP consists in defining a permutation of N items indexed in $[\![1; N]\!]$, based on pairwise preferences. $w_{i,j} \geqslant 0$ denotes the preference between items i and j: i is preferred to j if $w_{i,j}$ is high (higher than $w_{j,i}$). A ranking is evaluated with the sum of $w_{i,j}$ in the $\frac{N(N-1)}{2}$ pairwise preferences implied by the ranking. Each permutation of $[\![1; N]\!]$ encodes a solution of LOP, there are thus $N!$ feasible solutions. The reference ILP formulation uses binary variables $x_{i,j} \in \{0, 1\}$ such that $x_{i,j} = 1$ if and only if item i is ranked before item j in the consensus permutation [11]. Such encoding allows to compute the ranking, the rank of item i is $1 + \sum_{j \neq i} x_{j,i}$. ILP formulation from [11] uses $O(N^2)$ variables and $O(N^3)$ constraints:

$$\max_{x \geqslant 0} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{1}$$

$$x_{i,j} + x_{j,i} = 1 \qquad \forall i < j, \tag{2}$$

$$x_{i,j} + x_{j,k} + x_{k,i} \leqslant 2 \qquad \forall i \neq j \neq k, \tag{3}$$

$$x_{i,j} \in \{0, 1\}, \qquad \forall i \neq j \tag{4}$$

Constraints (2) model that either i is preferred to j, either j is preferred to i. Constraints (3) ensure that $x_{i,j}$ variables encode a permutation: if i is before j and j before k, i.e. $x_{i,j} = 1$ and $x_{j,k} = 1$, then i must be before k, i.e. $x_{i,k} = 1$ which is equivalent to $x_{k,i} = 0$ using (2). Constraints (2) and (3) are proven to be facet defining under some conditions [11]. Note that some alternative equivalent ILP were formulated. Firstly, the problem is here defined as a maximization, whereas it is considered as a minimization of disagreement in [2]. Considering $w'_{i,j} = M - w_{i,j}$ where M is an upper bound of the weights $w_{i,j}$, or $w'_{i,j} = w_{j,i}$, allows to consider the same problem as minimization or maximization. Secondly, equations (3) are replaced by $x_{i,k} - x_{i,j} - x_{j,k} \geqslant -1$. Formulation (3) is symmetrical, and was also used for the ATSP [16]. To see the equivalence, we use that $-x_{i,k} = x_{k,i} - 1$:

$$\begin{aligned} x_{i,k} - x_{i,j} - x_{j,k} \geqslant -1 &\iff -x_{i,k} + x_{i,j} + x_{j,k} \leqslant 1 \\ x_{i,k} - x_{i,j} - x_{j,k} \geqslant -1 &\iff x_{k,i} - 1 + x_{i,j} + x_{j,k} \leqslant 1 \\ x_{i,k} - x_{i,j} - x_{j,k} \geqslant -1 &\iff x_{k,i} + x_{i,j} + x_{j,k} \leqslant 2 \end{aligned}$$

3 From ATSP to consensus ranking, tighter formulations

LOP and ATSP have similarities: basic encoding of a feasible solution is a permutation on $[\![1; N]\!]$, and order matters for cost computation. If any LOP solution is directly a permutation, ATSP solutions are Hamiltonian oriented cycles. LOP solutions can be projected in a cycle structure, adding a fictive node 0 such that $w_{0,i} = w_{i,0} = 0$ opening and closing the cycle: $x_{0,i} = 1$ (resp $x_{i,0} = 1$) expresses that i is the first (resp last) item of the linear ordering. This section aims to

use polyhedral work from the TSP to tighten the reference formulation [14]. The major difference between ATSP and LOP is the objective function. For ATSP, binary variables $y_{i,j} \in \{0,1\}$ are defined such that such that $y_{i,j} = 1$ if and only if the next item immediately after i is item j, for $i \neq j \in [\![1; N]\!]$. We do not introduce a node 0 to distinguish formulations where the fictive node is not necessary, we define binary variables $f_i, l_i \in \{0,1\}$ such that $f_i = x_{0,i} = 1$ (resp $l_i = x_{i,0} = 1$) denotes that item i is the first (resp last) in the linear ordering. Having these variables x, y, l, f induce directly another ILP formulation for LOP, denoted SSB for ATSP [16]:

$$\max_{x,y,f,l} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{5}$$

$$x_{i,j} + x_{j,k} + x_{k,i} \leqslant 2 \qquad \forall i \neq j \neq k, \tag{6}$$

$$y_{i,j} \leqslant x_{i,j} \qquad \forall i \neq j, \tag{7}$$

$$x_{i,j} + x_{j,i} = 1 \qquad \forall i < j, \tag{8}$$

$$\textstyle\sum_i f_i = 1 \tag{9}$$

$$\textstyle\sum_i l_i = 1 \tag{10}$$

$$l_i + \textstyle\sum_{j \neq i} y_{i,j} = 1 \qquad \forall i, \tag{11}$$

$$f_i + \textstyle\sum_{j \neq i} y_{j,i} = 1 \qquad \forall i, \tag{12}$$

$$x_{i,j}, y_{i,j} \in \{0,1\}, \qquad \forall i \neq j \tag{13}$$

$$f_i, l_i \in \{0,1\}, \qquad \forall i \tag{14}$$

The only difference with the SSB formulation of the TSP is the objective function minimizing a weighted sum of y, f, l variables for ATSP [16]. Constraints (6) and (8) from the SSB formulation are the identical with the reference ILP formulation for LOP. Constraints (11) and (12) are TSP elementary flow constraints: for each item there is a unique predecessor and a unique successor, 0 as node successor or predecessor implies using variables f_i, l_i. Unicity constraints (9) and (10) are ATSP elementary flow constraints arriving to and leaving from the fictive node. SSB can be tighten in the SSB2 formulation, replacing constraints (6) by tighter constraints (15) from [16]:

$$\forall i \neq j \neq k, \quad x_{i,j} + y_{i,j} + x_{j,k} + x_{k,i} \leqslant 2 \tag{15}$$

Constraints (16) are also tightening strictly SSB2 formulation [16]:

$$\forall i, \quad f_i + l_i \leqslant 1 \tag{16}$$

Constraints (16) are sub-tour cuts between node 0 and item $i > 0$. Sub-tours between two cities may have a crucial impact in the resolution, as in [5]. Having variables x and constraints (6) and the tighter variants implies the other sub-tours between two items. Indeed, $y_{i,j} + y_{j,i} \leqslant x_{i,j} + x_{j,i} = 1$.

Another formulation was proposed for ATSP without constraints (6), but with linking constraints $y_{i,j} - x_{k,j} + x_{k,i} \leqslant 1$, to induce the same set of feasible solution [9]. These constraints can be tightened in two different ways:

$$\forall i \neq j \neq k, \quad y_{i,j} + y_{j,i} - x_{k,j} + x_{k,i} \leqslant 1 \tag{17}$$

$$\forall i \neq j \neq k, \quad y_{i,j} + y_{k,j} + y_{i,k} - x_{k,j} + x_{k,i} \leqslant 1 \tag{18}$$

Tightening only with (17) and (18) induce respectively GP2 and GP3 formulations for ATSP. A strictly tighter formulation, denoted GP4, is obtained with both sets of constraints [14]. A strictly tighter formulation is also obtained adding (15) to (17) and (18 for ATSP. These constraints are valid for LOP, numerical issues are to determine whether the quality of LP relaxation with such tighter formulation is significantly improved.

4 Other ILP reformulations

In this section, alternative ILP reformulations for LOP are provided, adapting also formulations from ATSP. Firstly, a formulation with $O(N^2)$ variables and constraints is given, before three-index formulations with $O(N^3)$ variables.

4.1 ILP formulation with $O(N^2)$ variables and constraints

Similarly with the famous MTZ formulation [12], $O(N)$ additional variables $n_i \in [\![0, N-1]\!]$ can directly indicate the position of the item in the ranking :

$$\max_{x,n \geqslant 0} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{19}$$

$$x_{i,j} + x_{j,i} = 1 \qquad \forall i < j, \tag{20}$$

$$n_j + N \times (1 - x_{i,j}) \geqslant n_i + 1 \; \forall i \neq j, \tag{21}$$

$$n_i + \sum_{j \neq i} x_{i,j} = N - 1 \qquad \forall i, \tag{22}$$

$$x_{i,j} \in \{0, 1\}, \qquad \forall i \neq j \tag{23}$$

$$n_i \in [0, N-1] \qquad \forall i \tag{24}$$

Objective function (19) and constraints (20) are like previously. Constraints (21) are similar with MTZ constraints: if i is ranked before j, i.e. $x_{i,j} = 1$, then it implies $n_j \geqslant n_i + 1$, n is a "big M" in this linear constraint. If constraints (20) and (21) are sufficient to induce feasible solutions for the ILP, constraints (22) completes (21) without using any "big M". Indeed, $c_i = \sum_{j \neq i} x_{i,j}$ counts the number of items after i, so that there is the invariant $c_i + n_i = N - 1$. A numerical issue is to determine if constraints (22) have a positive impact on the quality of the LP relaxation. As big M constraints are reputed to be weak and inducing poor LP relaxations (we refer to [5]), a numerical issue is to determine the difference with the continuous relaxation when relaxing also constraints (21) and (22). One note that this relaxation has a trivial optimal solution, considering

for $i \neq j$ the value $x_{i,j} = 1$ if and only if $w_{i,j} \geqslant w_{j,i}$). It induces that following upper bound is valid, and also greater than any LP relaxation for LOP:

$$UB = \sum_{i<j} \max(w_{i,j}, w_{j,i}) \tag{25}$$

Note that as for MTZ formulation, variables n_i can be declared as continuous, feasibility of (21) and bounds (24) implies that n variables are integer.

4.2 Three-index Flow formulation

Gouveia and Pires proposed also a three index formulation for ATSP, which is tighter than GP2, GP3, GP4 [9]. One can adapt this formulation to LOP using binary variables $z_{i,j,k} \in \{0,1\}$ for $i \neq j \neq k$ defined with $z_{i,j,k} = 1$ if and only if i is ranked before j (not necessarily immediately before) and j is ranked immediately before k. Like previously, first and last items in the ranking are marked with binaries $f_i, l_i \in \{0,1\}$. Previous binaries $x_{i,j}, y_{i,j} \in \{0,1\}$ are then defined by $x_{i,j} = \sum_k z_{i,j,k} + l_j$ and $y_{i,j} = z_{i,i,j}$.

$$\max_{z,l,f \geqslant 0} \quad \sum_{i \neq j} w_{i,j} \left(l_i + \sum_k z_{i,j,k} \right) \tag{26}$$

$$l_i + \textstyle\sum_k z_{i,j,k} + l_j + \sum_k z_{j,i,k} = 1 \quad \forall i < j, \tag{27}$$

$$\textstyle\sum_i f_i = 1 \tag{28}$$

$$\textstyle\sum_i l_i = 1 \tag{29}$$

$$l_i + \textstyle\sum_{j \neq i} z_{i,i,j} = 1 \quad \forall i, \tag{30}$$

$$f_i + \textstyle\sum_{j \neq i} z_{j,j,i} = 1 \quad \forall i, \tag{31}$$

$$z_{i,j,k} \leqslant z_{j,j,k} \quad \forall i, j, k, \tag{32}$$

$$l_i, f_i, \in \{0,1\}, \quad \forall i \tag{33}$$

$$z_{i,j,k} \in \{0,1\} \quad \forall i, j, k \tag{34}$$

Objective function and constraints (28)- (31) are unchanged from the ATSP structures, rewritten using z, f, l variables. Constraints (27) are constraints $x_{i,j} + x_{j,i} = 1$ Constraints (32) model that $z_{i,j,k} = 1$ implies that j is ranked just before k and thus $z_{j,j,k} = 1$. This formulation has $O(N^3)$ variables and $O(N^3)$ constraints only because of constraints (32). It is possible to have only $O(N^2)$ constraints and preserving the validity of the integer program replacing flow constraints (32) by the aggregated version:

$$\forall j, k, \quad \sum_i z_{i,j,k} \leqslant N z_{j,j,k} \tag{35}$$

4.3 Another three-index flow formulation

$z'_{i,j,k} \in \{0,1\}$ defined for $i \neq j \neq k$ with $z'_{i,j,k} = 1$ if and only if genes i, j, k are ranked in this order. Note that improvements of LP relaxation were obtained

using similar reformulation in [15]. In this ILP formulation, we keep variables $x_{i,j}$ $z'_{i,j,k} = 1$ implies $x_{i,j} = x_{j,k} = x_{i,k} = 1$. it induces the valid ILP formulation for LOP:

$$\max_{x,z \geqslant 0} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{36}$$

$$3z'_{i,j,k} \leqslant x_{i,j} + x_{j,k} + x_{i,k} \qquad \forall i \neq j \neq k \tag{37}$$

$$z'_{i,j,k} + z'_{i,k,j} + z'_{j,i,k} + z'_{j,k,i} + z'_{k,j,i} + z'_{k,i,j} = 1 \qquad \forall i \neq j \neq k, \tag{38}$$

$$x_{i,j} \in \{0,1\} \qquad \forall i \neq j, \tag{39}$$

$$z'_{i,j,k} \in \{0,1\} \qquad \forall i \neq j \neq k \tag{40}$$

Constraints (37) are linking constraints related to the definition of variables x, z: $z'_{i,j,k} = 1$ implies $x_{i,j} = x_{j,k} = x_{i,k} = 1$. Constraints (38) express that each triplet $i \neq j \neq k$ is assigned in exactly one order in a permutation, replacing constraints of type $x_{k,i} + x_{i,j} + x_{j,k}$. Constraints (38) induce that this ILP formulation has also $O(N^3)$ variables and $O(N^3)$ constraints. Note that a similar constraint can be defined as cut for the previous ILP formulation, with an inequality:

$$z_{i,j,k} + z_{i,k,j} + z_{j,i,k} + z_{j,k,i} + z_{k,j,i} + z_{k,i,j} \leqslant 1 \tag{41}$$

Table 1. Implemented formulations, denomination, variables, constraints

Formulation	Variables	Constraints	nbVar	nbConstraints
LOP_ref	x	(2), (3)	$O(N^2)$	$O(N^3)$
LOP_SSB2	x, f, l, y	(7) - (12), (15, (16)	$O(N^2)$	$O(N^3)$
LOP_GP3	x, f, l, y	(7) - (12), (18)	$O(N^2)$	$O(N^3)$
LOP_MTZ	x, n	(20), (22)	$O(N^2)$	$O(N^2)$
LOP_flowGP	z, f, l	(27)- (31),(32)	$O(N^3)$	$O(N^3)$
LOP_flowGP_aggr	z, f, l	(27)- (31),(35)	$O(N^3)$	$O(N^2)$
LOP_flow2	x, z'	(37), (38)	$O(N^3)$	$O(N^3)$

5 Computational experiments and results

Numerical experiments were proceeded using a workstation with a dual processor Intel Xeon E5-2650 v2@2.60GHz, for 16 cores and 32 threads in total. Cplex was used to solve LPs and ILPs, in its version 20.1. Cplex was called using OPL modeling language and OPL script. LocalSolver in its version 10.5 was used as a heuristic solver benchmark to be able to compare primal solutions in cases where optimal solutions are not proven. The maximal time limit for Cplex and LocalSolver was set to one hour, Cplex was used with its default

parameters. It was necessary to generate specific instances for this study, it is presented before the analysis of numerical experiments. Code and instances will be available online.

5.1 Data generation and characteristics

As mentioned by [1, 13], characteristics of instances is crucial in the difficulty of instances. In many social choice applications and datasets, N is small, so that exact resolution with the reference ILP formulation is almost instantaneous. For the biological application, N is very large but median of permutations among similar permutations is easier than general instances. In the extreme case where $w_{i,j}$ coefficients encode a permutation (median of 1-permutation, trivial problem), trivial bounds UB give the optimal, and all the LP relaxation of the ILP formulations give the integer optimal solution. For this numerical study, as in [14], quality of polyhedral descriptions are analyzed on the implications on the quality of LP relaxation using diversified directions of the objective function. Three generators were used for the goal of this study:

- `aleaUniform` (denoted aUnif): $w_{i,j}$ for $i \neq j$ are randomly generated with a uniform law in $[\![0, 100]\!]$.
- `aleaSum100` (denoted aSum): uniform generation in $[\![0, 100]\!]$ such that $w_{i,j} + w_{j,i} = 100$: for $i < j$ $w_{i,j}$ is randomly generated $[\![0, 100]\!]$ and $w_{j,i}$ is then set to $w_{j,i} = 100 - w_{i,j}$.
- `aleaShuffle` (denoted aShuf): $\max(N/2, 20)$ random permutations are generated (with Python function shuffle), $w_{i,j}$ are then computes using Kendall-τ distance and Kemeny ranking, as in [1–3].

A fourth generator was coded, as in `aleaShuffle` but generating small perturbations around a random permutation, with small (and not so small perturbations). Actually, the results were very similar for all the ILP formulations, as for the 1-median trivial instances, these instances were much easier. This illustrates that real-life instances for median of permutations are much easier than unstructured and random instances. The three generators allows to analyze the impact of some structures of instances in the difficulty and quality of ILP and LP resolution.

Number of items N was generated with values $N \in \{20, 30, 40, 50, 100\}$. For each generator and value of N, 30 instances are generated and results are given in average for each group of 30 similar instances, with the denomination XX$-N$ where XX $\in$ {aUnif,aSum,aShuf}. Lower and upper bounds $v(i)$ on instance i are compared with the following gap indicator to the Best Known Solution (BKS), denotes BKS(i):

$$gap = \frac{\mid v(i) - BKS(i) \mid}{BKS(i)} \tag{42}$$

For $N \in \{20, 30, 40\}$, BKS are optimal solutions proven by Cplex. For $N \in \{50, 100\}$, LocalSolver always provides the BKS. There is also no counter-example where LocalSolver do not find an optimally proven solution in one hour, we note that LocalSolver is also very efficient in short time limits.

Table 2. Comparison of the average gaps to the BKS for the LP relaxations of formulations recalled in Table 1 and the naive upper bound (25)

Instances	(25)	ref/SSB2	GP3	MTZ	flow-GP	flow-GP-agg	flow2
aUnif-20	12,86 %	0,02 %	10,49 %	10,84 %	5,22 %	11,53 %	12,86 %
aUnif-30	14,86 %	0,17 %	13,20 %	13,39 %	7,34 %	13,98 %	14,86 %
aUnif-40	16,98 %	0,60 %	15,68 %	15,78 %	9,22 %	16,16 %	16,98 %
aUnif-50	17,84 %	1,12 %	16,80 %	16,86 %	10,22 %	17,17 %	17,84 %
aUnif-100	21,65 %	3,17 %	21,10 %	21,11 %	-	21,25 %	21,65 %
aSum-20	19,00 %	0,05 %	15,56 %	16,13 %	7,26 %	17,23 %	19,00 %
aSum-30	21,96 %	0,31 %	19,53 %	19,84 %	10,25 %	20,45 %	21,96 %
aSum-40	24,40 %	1,18 %	22,52 %	22,70 %	12,53 %	23,21 %	24,40 %
aSum-50	26,24 %	2,25 %	24,71 %	24,83 %	14,16 %	25,23 %	26,24 %
aSum-100	31,44 %	4,97 %	30,64 %	30,65 %	-	30,84 %	31,44 %
aShuf-30	1,93 %	0,00 %	1,39 %	1,42 %	0,29 %	1,89 %	1,93 %
aShuf-40	1,44 %	0,00 %	1,18 %	1,18 %	0,42 %	1,42 %	1,44 %
aShuf-50	1,41 %	0,00 %	1,18 %	1,18 %	0,44 %	1,40 %	1,41 %
aShuf-100	1,80 %	0,02 %	1,65 %	1,55 %	-	1,80 %	1,80 %

5.2 Comparing LP relaxations

To analyze the quality of polyhedral descriptions of ILP formulations, Table 2 presents gaps of LP relaxations of ILP formulations for LOP and the naive upper bound (25), following a similar methodology to [14]. Table 3 presents the computation time to calculate LP relaxation, to appreciate the impact of the number of variables and constraints recalled in Table 1. These tables illustrate the difficulty of instances, aShuf are quite easy instances with good naive upper bounds and LP relaxations. Datasets aSum and aUnif induce more difficulties with worse continuous bounds, and aSum is even more difficult than aUnif.

Contrary to ATSP where GP2, GP3, SSB2 are not redundant [14], (3) induce much better LP relaxations for LOP than (17) and (18). Adding (17) and (18) in ILP formulations with (3) or (15) does not induce any difference in the quality of LP relaxation. The explanation for the difference between ATSP and LOP is the different nature of the objective function, polyhedron defined by constraints are identical, but objective function in x or y changes the projection on the space of interest. It explains why in Table 1, we remove constraints of type (3) to highlight the difference of quality of LP relaxation.

The standard flow formulation flow-GP improves significantly the quality of LP relaxation of GP3, as for the ATSP, but it is still significantly worse than SSB formulations. Computation time of LP relaxation is much higher with flow-GP, for $N = 100$ computations were stopped in one hour without termination. With aggregation (35) instead of (32), LP relaxation is computed quickly, but the quality of LP relaxation is dramatically decreased, the continuous bounds are very close to the naive upper bounds. MTZ adaptation has the quickest LP relaxation, but the continuous bounds are close to the ones of GP3. Last flow formulation always provides exactly the naive upper bounds, constraints (41) does not tighten flow-GP formulation.

Table 3. Comparison of the average time (in seconds) to compute LP relaxations for ILP formulations recalled in Table 1

Instances	ref	SSB2	GP3	MTZ	flow-GP	flow-GP-agg	flow2
aUnif-20	0,04	0,14	0,32	0,00	1,21	0,06	0,07
aUnif-30	0,28	0,75	1,08	0,01	7,62	0,18	0,25
aUnif-40	0,49	2,27	3,58	0,03	38,60	0,53	0,83
aUnif-50	0,95	5,59	10,55	0,12	173,49	1,36	2,32
aUnif-100	26,63	278	839	1,04	-	18,24	54,61
aSum-20	0,04	0,17	0,33	0,00	1,20	0,06	0,07
aSum-30	0,29	0,77	1,08	0,01	7,48	0,19	0,25
aSum-40	0,50	2,12	3,43	0,03	37,10	0,55	0,83
aSum-50	0,95	5,65	10,74	0,12	168,24	1,40	2,27
aSum-100	27	282,46	819	1,75	-	17,40	55,63
aShuf-30	0,06	0,24	1,04	0,01	6,18	0,18	0,27
aShuf-40	0,16	0,84	3,73	0,04	26,86	0,49	0,74
aShuf-50	0,33	2,17	11,63	0,13	98,88	1,32	2,13
aShuf-100	17,7	353,5	1360	1,76	-	16,45	51,34

LP relaxation of the reference formulation is of an excellent quality, which illustrates polyhedral results and proven facets from [11]. In Table 1, reference and SSB2 formulations have the same values: except on three instances, LP relaxation are the same (with a tolerance to numerical errors on the last digit). On instance number 27 in aUnif-20 and instances number 17 and 29 in aSum-20, SSB2 improves the reference formulation around 0.01%, making a difference of one unit in the integer ceil rounding of the continuous relaxation. With additional experiments, the difference is only due to (3) instead of (15), no difference was observe adding only (16). With this result, this shows that SSB2 formulation for LOP is in theory strictly tighter than the reference formulation, but with small and rare improvements and a larger computation time for the LP relaxation.

5.3 Comparing Branch&Bound convergences

Table 4 analyzes the impact of Cplex cuts and heuristics at the root node, before branching in the Branch&Bound (B&B) tree, for ILP formulations ref and SSB2. If SSB2 improves slightly LP relaxation quality, additional variables and constraints can help modern ILP solvers detecting other structures for cut generation, as in [4]. For LOP, computations at the root node of B&B tree are much slower with SSB2, coherently with the double number of variables, but the efficiency of cuts and primal heuristics is significantly worse with the heavier SSB2 formulation. Having a larger ILP model, heavier matrix operations for generation of cutting planes are needed by Cplex, and this stop earlier cuts that would have been generated using the reference formulation, the size of ILP matrix is crucial here. Note also that Table 4 shows that few improvement of LP relaxation is provided at the root node of B&B tree.

These elements explain the difference in the B&B convergence in one hour allowing branching, the reference ILP formulation is largely superior. For some

Table 4. Comparison of Lower Bounds (LB) and Upper Bounds (UB) of formulations ref and SSB2 after Cplex cuts and heuristics at the root node (i.e. before branching). Common UB with the LP relaxation are also provided for comparison

	LP	UB	LB	time	UB	LB	time
	ref,SSB2	ref			SSB2		
aUnif-20	0,02%	0,00%	0,00 %	0,1	0,00 %	0,00 %	0,4
aUnif-30	0,17%	0,00%	0,00 %	1,5	0,09 %	0,50 %	14
aUnif-40	0,60%	0,44%	0,22 %	30,5	0,52 %	4,58 %	159
aUnif-50	1,12%	0,96%	0,82 %	212,9	0,98 %	5,27 %	1336
aUnif-100	3,17%	3,07%	5,49 %	3600	3,15 %	7,37 %	3600
aSum-20	0,05%	0,00%	0,00 %	0,12	0,00 %	0,00 %	0,55
aSum-30	0,31%	0,02%	0,02 %	2,0	0,16 %	0,79 %	20
aSum-40	1,18%	0,75%	0,32 %	64	0,95 %	5,08 %	296
aSum-50	2,25%	1,82%	0,95 %	297	1,95 %	6,72 %	989
aSum-100	4,97%	4,82%	6,87 %	3600	4,95 %	9,62 %	3600
aShuf-30	0,00%	0,00%	0,00 %	0,14	0,00 %	0,00 %	0,86
aShuf-40	0,00%	0,00%	0,00 %	0,37	0,00 %	0,00 %	2,9
aShuf-50	0,00%	0,00%	0,00 %	0,90	0,00 %	0,00 %	8
aShuf-100	0,02%	0,02%	0,02 %	477	0,02 %	6,02 %	3395

instances with $N = 40$ or $N = 50$, ref formulation can converge in ten minutes whereas a significant gap between lower and upper bounds remains after one hour. This validates the reference formulation as baseline ILP model for [2].

5.4 Variable Fixing heuristics

The excellent quality of the LP relaxation with the reference formulation allows to investigate the quality of heuristics using continuous solutions of the LP relaxation, as in [7]. Variable Fixing (VF) denotes here a heuristic reduction of the search space based on the LP relaxation, to set integer values to variables in the ILP resolution based on some values of the LP relaxation. For instance, on may fix a variable fixing preprocessing for variables with an integer value in the continuous relaxation, expecting that these integer decisions are good.

In general, it makes a difference to apply VF preprocessing on zeros ans ones in the LP relaxation, as in [7]. This makes in general many possibilities of VF preprocessing, also with specific rules to select a subset of variable to fix [7]. For LOP, imposing $x_{i,j} = 1$ implies fixation $x_{j,i} = 0$ with constraints (2). Note also that constraints (3) may induce having continuous solution with variables $x_{i,j} = x_{j,k} = x_{k,i} = 2/3$, so that rounding to ones variables lower that $2/3$ induce direct infeasibility on the corresponding (3) constraint. This property does not hold rounding to ones variables that are superior to 0.7 Hence, two VF strategies were implemented, on one hand fixing the integer value, and on the other hand considering the threshold for rounding to 0.7. Actually, there were slight difference for these two strategies. Experiments were also done using the quick MTZ relaxation for the LP relaxation, this was significantly degrading the performance of the VF heuristic.

Table 5. Comparison of gaps to BKS and computation time of Cplex in ILP solving using the reference formulation, without and with Variable Fixing (VF) preprocessing on integer values in the LP relaxation of the reference formulation. BKS are optimums for $N \leqslant 40$, for $N \geqslant 50$ BKS were given by LocalSolver

Instances	LB ref	time (sec)	LB ref + VF	time (sec)
aUnif-20	0,00 %	0,1	0,01 %	0,04
aUnif-30	0,00 %	1,5	0,04 %	0,62
aUnif-40	0,00 %	30,5	0,17 %	13
aUnif-100	5,49 %	3600	2,36 %	3600
aSum-20	0,00 %	0,13	0,05 %	0,06
aSum-30	0,00 %	2,1	0,09 %	0,77
aSum-40	0,00 %	63,5	0,43 %	12,7
aSum-100	6,87 %	3600	3,14 %	3600
aShuf-30	0,00 %	0,13	0,00 %	0,03
aShuf-40	0,00 %	0,37	0,00 %	0,08
aShuf-50	0,00 %	0,92	0,00 %	0,12
aShuf-100	0,00 %	1820	0,00 %	35,7

Table 5 compares the gap to BKS and computation time using the VF preprocessing to the basic reference formulation. For small and easy instances where the reference formulation gives optimal solutions, the degradation of the objective function is small with the VF heuristic, speeding up significantly the computation time. For the largest instances with $N = 100$, VF matheuristic is significantly better that the exact resolution, illustrating the difficulty of the ILP solver to find good primal solutions with its primal heuristics. The primal solutions of matheuristic are in this case also significantly worse than the ones of LocalSolver, the VF speed up is not sufficient to reach and advanced phase of the B&B convergence

6 Conclusions and perspectives

If the reference ILP formulation seemed to be improvable using ATSP results, only a slightly tighter ILP formulation is obtained after this reformulation work. Analyzing the ILP convergence with a modern ILP solver shows that the LP relaxation is of an excellent quality with the LP relaxation, but is fewly improved after. Also, primal heuristics are not efficient on the problem, a basic VF matheuristic improves significantly the primal solutions for difficult instances. Also, this paper illustrates the graduated difficulty of instances.

These results offer perspectives for the biological application [1]. Matheuristics can be used in this context, also with the extension ties, and using properties of their easier median of permutation instances allowing specific reduction space operators [1, 13]. Perspectives are also to combine matheuristics and local search approaches which are efficient for the problem, as shown by LocalSolver benchmark on this study, and also with [8].

References

1. P. Andrieu, B. Brancotte, L. Bulteau, S. Cohen-Boulakia, A. Denise, A. Pierrot, and S. Vialette. Efficient, robust and effective rank aggregation for massive biological datasets. *Future Generation Computer Systems*, 2021.
2. B. Brancotte, B. Yang, G. Blin, S. Cohen Boulakia, A. Denise, and S. Hamel. Rank aggregation with ties: Experiments and analysis. *Proc. of the VLDB Endowment (PVLDB)*, 8(11):2051, Aug 2015.
3. S. Cohen-Boulakia, A. Denise, and S. Hamel. Using medians to generate consensus rankings for biological data. In *International Conference on Scientific and Statistical Database Management*, pages 73–90. Springer, 2011.
4. N. Dupin. Tighter MIP formulations for the discretised unit commitment problem with min-stop ramping constraints. *EURO Journal on Computational Optimization*, 5(1):149–176, 2017.
5. N. Dupin, R. Parize, and E. Talbi. Matheuristics and Column Generation for a Basic Technician Routing Problem. *Algorithms*, 14(11):313, 2021.
6. N. Dupin and E. Talbi. Machine learning-guided dual heuristics and new lower bounds for the refueling and maintenance planning problem of nuclear power plants. *Algorithms*, 13(8):185, 2020.
7. N. Dupin and E. Talbi. Parallel matheuristics for the discrete unit commitment problem with min-stop ramping constraints. *International Transactions in Operational Research*, 27(1):219–244, 2020.
8. C. Garcia, D. Pérez-Brito, V. Campos, and R. Martí. Variable neighborhood search for the linear ordering problem. *Computers & OR*, 33(12):3549–3565, 2006.
9. L. Gouveia and J. Pires. The asymmetric travelling salesman problem and a reformulation of the miller–tucker–zemlin constraints. *Euro J of Oper Res*, 112(1):134–146, 1999.
10. M. Grötschel, M. Jünger, and G. Reinelt. A cutting plane algorithm for the linear ordering problem. *Operations research*, 32(6):1195–1220, 1984.
11. M. Grötschel, M. Jünger, and G. Reinelt. Facets of the linear ordering polytope. *Mathematical programming*, 33(1):43–60, 1985.
12. C. Miller, A. Tucker, and R. Zemlin. Integer programming formulation of traveling salesman problems. *Journal of the ACM*, 7(4):326–329, 1960.
13. R. Milosz and S. Hamel. Space reduction constraints for the median of permutations problem. *Discrete Applied Mathematics*, 280:201–213, 2020.
14. T. Öncan, I. Altınel, and G. Laporte. A comparative analysis of several asymmetric traveling salesman problem formulations. *Computers & OR*, 36(3):637–654, 2009.
15. F. Peschiera, R. Dell, J. Royset, A. Haït, N. Dupin, and O. Battaïa. A novel solution approach with ML-based pseudo-cuts for the Flight and Maintenance Planning problem. *OR Spectrum*, 43(3):635–664, 2021.
16. S. Sarin, H. Sherali, and A. Bhootra. New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints. *Operations research letters*, 33(1):62–70, 2005.
17. E. Talbi. Combining metaheuristics with mathematical programming, constraint programming and machine learning. *Annals of OR*, 240(1):171–215, 2016.
18. E. Talbi. Machine learning into metaheuristics: A survey and taxonomy. *ACM Computing Surveys (CSUR)*, 54(6):1–32, 2021.