Integer Linear Programming Reformulations for the Linear Ordering Problem

Nicolas Dupin(✉)

Univ Angers, LERIA, SFR MATHSTIC, 49000 Angers, France
nicolas.dupin@univ-angers.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 linear ordering 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. The quality of the continuous relaxation allows to design rounding heuristics, it offers perspectives to design matheuristics.

Keywords: Optimization · Integer Linear Programming · Linear ordering · 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 [18,19]. Discrete and exact optimization, especially Integer Linear Programming (ILP), is also useful to model and solve specific variants of clustering or selection problems for ML [5,6]. In this paper, another application of ILP to learning is studied: the Linear Ordering Problem (LOP). LOP is used in social choice theory to define a common consensus ranking based on pairwise preferences. If many applications deal with a small number of items to rank, bio-informatics applications solve specific LOP instances of large size as medians 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 on consensus ranking for biological datasets, and use specific data characteristics of these median of permutation problems for an efficient resolution [1,14]. This paper analyzes the limits of state-of-the art ILP solvers to solve LOP instances. Several alternative ILP formulations are designed using recent results on the Asymmetric Traveling Salesman Problem (ATSP) from [15]. Comparison of Linear Programming (LP) relaxations illustrates and validates polyhedral analyses, as in [15]. The practical implication of polyhedral work is analyzed on the resolution using modern ILP solvers, as in [4]. Lastly, the quality of LP relaxation is used to design variable fixing matheuristics, as in [7]. Variants of variable definitions and ILP Formulations are recalled in Tables 1 and 2.

Table 1. Definitions of variables in the ILP formulations

Variables	Definitions
$x_{i,j} \in \{0,1\}$	$x_{i,j} = 1$ iff item $i \neq j$ is ranked before j
$y_{i,j} \in \{0,1\}$	$y_{i,j} = 1$ iff item $i \neq j$ is ranked immediately before j
$f_i \in \{0,1\}$	$f_i = 1$ iff item i is the first item of the ranking
$l_i \in \{0,1\}$	$f_i = 1$ iff item i is the last item of the ranking
$n_i \in [0, N-1]$	$n_i - 1$ gives the position of item i in the ranking
$z_{i,j,k} \in \{0,1\}$	$z_{i,j,k} = 1$ for $i \neq k$ and $j \neq k$ iff i is ranked before j and item j is ranked immediately before k
$z'_{i,j,k} \in \{0,1\}$	$z'_{i,j,k} = 1$ for $i \neq j \neq k$ iff i is ranked before j and j is before k

2 Problem Statement and Reference ILP Formulation

LOP consists in defining a permutation of N items indexed in $[\![1;N]\!]$, while maximizing the likelihood with given 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 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 it implies. Each permutation of $[\![1;N]\!]$ encodes a solution of LOP, there are $N!$ feasible solutions. The reference ILP formulation, given and analyzed in [10,11], 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. With such encoding, one computes the rank of each item i with $1 + \sum_{j \neq i} x_{j,i}$. ILP formulation from [11] uses $O(N^2)$ variables and $O(N^3)$ constraints:

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

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

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

Constraints (2) model that either i is preferred to j, or 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 $w'_{i,j} = w_{j,i}$ or $w'_{i,j} = M - w_{i,j}$, where M is an upper bound of weights $w_{i,j}$, it allows to transform minimization into maximization. Secondly, equations (3) are equivalently written as $x_{i,k} - x_{i,j} - x_{j,k} \geqslant -1$ in [2]. Formulation (3) is symmetrical, and was also used for the ATSP [17]. To see the equivalence, we use that $-x_{i,k} = x_{k,i} - 1$:

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

3 From ATSP to Consensus Ranking, Tighter Formulations

LOP and ATSP feasible solutions may be encoded as permutations of $[\![1;N]\!]$, order matters for cost computations. If any LOP solution is 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 uses polyhedral work from ATSP to tighten the reference formulation for LOP [15]. For ATSP, binary variables $y_{i,j} \in \{0,1\}$ are defined such that $y_{i,j} = 1$ if and only if j is next item immediately after item i, for $i \neq j \in [\![1;N]\!]$. Equivalently to consider a fictive node 0, 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 induces another ILP formulation for LOP, denoted SSB for ATSP [17]:

$$\max_{x,y,f,l} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{4}$$
$$x_{i,j} + x_{j,k} + x_{k,i} \leqslant 2 \qquad \forall i \neq j \neq k, \tag{5}$$
$$y_{i,j} \leqslant x_{i,j} \qquad \forall i \neq j, \tag{6}$$
$$x_{i,j} + x_{j,i} = 1 \qquad \forall i < j, \tag{7}$$
$$\textstyle\sum_i f_i = 1 \tag{8}$$
$$\textstyle\sum_i l_i = 1 \tag{9}$$
$$l_i + \textstyle\sum_{j \neq i} y_{i,j} = 1 \qquad \forall i, \tag{10}$$
$$f_i + \textstyle\sum_{j \neq i} y_{j,i} = 1 \qquad \forall i, \tag{11}$$

Objective function differs from ATSP: a weighted sum of y, f, l variables is minimized for ATSP [17]. For ATSP, x variables were used only to cut subtours, whereas it is necessary for LOP to write a linear objective function. The constraints are identical for ATSP and LOP once variables x, y, f, l are used. Constraints (5) and (7) reuse (2) and (3). Constraints (10) and (11) are ATSP elementary flow constraints: for each item there is a unique predecessor and a unique successor, 0 as node successor or predecessor implies variables f_i, l_i are used. Unicity constraints (8) and (9) are ATSP elementary flow constraints arriving to and leaving from the fictive node 0. SSB can be tightened in the SSB2 formulation, replacing constraints (5) by tighter constraints (12) from [17]:

$$\forall i \neq j \neq k, \quad x_{i,j} + y_{i,j} + x_{j,k} + x_{k,i} \leqslant 2 \tag{12}$$

Sub-tours between two cities (or items) may have a crucial impact in the resolution, as in [5]. Having variables x and constraints (5) and the tighter variants implies the other sub-tours between two items. Indeed, $y_{i,j}+y_{j,i} \leqslant x_{i,j}+x_{j,i} = 1$. Constraints (13) are sub-tour cuts between node 0 and each item $i > 0$, these are known to tighten strictly SSB2 formulation [17]:

$$\forall i, \quad f_i + l_i \leqslant 1 \tag{13}$$

Another formulation was proposed for ATSP without constraints (5), 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:

$$\forall i \neq j \neq k, \quad y_{i,j} + y_{j,i} - x_{k,j} + x_{k,i} \leqslant 1 \tag{14}$$

$$\forall i \neq j \neq k, \quad y_{i,j} + y_{k,j} + y_{i,k} - x_{k,j} + x_{k,i} \leqslant 1 \tag{15}$$

Tightening only with (14) and (15) induce respectively GP2 and GP3 formulations for ATSP [9]. A strictly tighter formulation, denoted GP4, is obtained with both sets of constraints [15]. A strictly tighter formulation is also obtained adding (12) to (14) and (15) for ATSP. Numerical issues are to determine whether the quality of LP relaxation is significantly improved after tightening for LOP.

4 Other ILP Reformulations

In this section, alternative ILP reformulations for LOP are provided, adapting other 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 MTZ formulation [13], $O(N)$ additional variables $n_i \in [\![0, N-1]\!]$ directly indicate the position of the item in the ranking :

$$\max_{x,n \geqslant 0} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{16}$$

$$x_{i,j} + x_{j,i} = 1 \qquad \forall i < j, \tag{17}$$

$$n_j + N \times (1 - x_{i,j}) \geqslant n_i + 1 \; \forall i \neq j, \tag{18}$$

$$n_i + \sum_{j \neq i} x_{i,j} = N - 1 \qquad \forall i, \tag{19}$$

$$n_i \in [0, N-1] \qquad \forall i \tag{20}$$

Note that as for MTZ formulation, variables n_i can be declared as continuous, feasibility of (18) and bounds (20) implies $n_i \in [\![0, N-1]\!]$. Objective function (16) and constraints (17) are unchanged. Constraints (18) 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 (17) and (18) are sufficient to induce feasible solutions for the ILP, constraints (19) complete (18) without using any “big M”. Indeed, $c_i = \sum_{j \neq i} x_{i,j}$ counts the number of items after i, so that for each i, $c_i + n_i = N - 1$. As big M constraints are reputed to be weak and inducing poor LP relaxations, a numerical issue is to determine the difference with the continuous relaxation when relaxing also constraints (18) and (19).

Note that this relaxation has trivial optimal solutions, considering $x_{i,j} = 1$ and $x_{j,i} = 0$ for $i \neq j$ such that $w_{i,j} \geqslant w_{j,i}$. Hence, following upper bound is valid, and also larger than any LP relaxation for LOP:

$$UB = \sum_{i<j} \max(w_{i,j}, w_{j,i}) \tag{21}$$

4.2 Three-Index Flow Formulation

Another three index formulation, tighter than GP2, GP3 and GP4, was proposed for ATSP [9]. Adapting this formulation to LOP, one uses binary variables $z_{i,j,k} \in \{0,1\}$ for $i \neq k$ and $j \neq k$ 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. First and last items are still marked with binaries $f_i, l_i \in \{0,1\}$. 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 $y_{i,j} = z_{i,i,j}$.

$$\max_{z,l,f \geqslant 0} \quad \sum_{i \neq j} w_{i,j} \left(l_i + \sum_k z_{i,j,k} \right) \tag{22}$$

$$l_i + \textstyle\sum_k z_{i,j,k} + l_j + \sum_k z_{j,i,k} = 1 \qquad \forall i < j, \tag{23}$$

$$\textstyle\sum_i f_i = 1 \tag{24}$$

$$\textstyle\sum_i l_i = 1 \tag{25}$$

$$l_i + \textstyle\sum_{j \neq i} z_{i,i,j} = 1 \qquad \forall i, \tag{26}$$

$$f_i + \textstyle\sum_{j \neq i} z_{j,j,i} = 1 \qquad \forall i, \tag{27}$$

$$z_{i,j,k} \leqslant z_{j,j,k} \qquad \forall i,j,k, \tag{28}$$

Constraints (23) and (24)–(27) are respectively constraints (7) and (8)–(11) replacing x, y occurrences by the linear expressions using z, l variables. A similar operation allows to write the objective function (22) using z, l variables. Constraints (28) 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 (28). It is possible to preserve the validity of the ILP while having only $O(N^2)$ constraints replacing flow constraints (28) by the aggregated version:

$$\forall j,k, \quad \sum_i z_{i,j,k} \leqslant N z_{j,j,k} \tag{29}$$

4.3 Another Three-Index Flow Formulation

In another three-index formulation, binary variables $z'_{i,j,k} \in \{0,1\}$ are defined such that for $i \neq j \neq k$, $z'_{i,j,k} = 1$ if and only if items i, j, k are ranked in this order. In this ILP formulation, we keep variables $x_{i,j}$, it induces the valid ILP formulation for LOP:

$$\max_{x,z \geqslant 0} \quad \sum_{i \neq j} w_{i,j} x_{i,j} \tag{30}$$

$$3z'_{i,j,k} \leqslant x_{i,j} + x_{j,k} + x_{i,k} \qquad \forall i \neq j \neq k \tag{31}$$

$$z'_{i,j,k} + z'_{i,k,j} + z'_{j,i,k} + z'_{j,k,i} + z'_{k,j,i} + z'_{k,i,j} = 1 \qquad \forall i \neq j \neq k, \tag{32}$$

Constraints (31) are linking constraints among variables x, z: $z'_{i,j,k} = 1$ implies $x_{i,j} = x_{j,k} = x_{i,k} = 1$. Constraints (32) express that each triplet $i \neq j \neq k$ is assigned in exactly one order in a permutation, replacing constraints of type $x_{k,i} + x_{i,j} + x_{j,k} \leqslant 2$. Constraints (32) 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:

$$z_{i,j,k} + z_{i,k,j} + z_{j,i,k} + z_{j,k,i} + z_{k,j,i} + z_{k,i,j} \leqslant 1 \tag{33}$$

Table 2. Summary of implemented formulations, their denomination, the sets and asymptotic number of variables and constraints

Formulation	Variables	Constraints	nbVariables	nbConstraints
LOP_ref	x	(2), (3)	$O(N^2)$	$O(N^3)$
LOP_SSB2	x, f, l, y	(6)–(11), (12, (13)	$O(N^2)$	$O(N^3)$
LOP_GP3	x, f, l, y	(6)–(11), (15)	$O(N^2)$	$O(N^3)$
LOP_MTZ	x, n	(17), (19)	$O(N^2)$	$O(N^2)$
LOP_flowGP	z, f, l	(23)–(27), (28)	$O(N^3)$	$O(N^3)$
LOP_flowGP_aggr	z, f, l	(23)–(27), (29)	$O(N^3)$	$O(N^2)$
LOP_flow2	x, z'	(31), (32)	$O(N^3)$	$O(N^3)$

5 Computational Experiments and Results

Numerical experiments were proceeded using a workstation with a dual processor Intel Xeon E5-2650 v2@2.60GHz, using at most 16 cores and 32 threads in total. Cplex version 20.1 was used to solve LPs and ILPs. Cplex was called using OPL modeling and OPL script languages. LocalSolver in its version 10.5 was used as a heuristic solver benchmark to compare primal solutions when 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. For reuse and reproducibility, OPL and LocalSolver codes and generated instances are available online at https://github.com/ndupin/linearOrdering.

5.1 Data Generation and Characteristics

It was necessary to generate specific instances for this study. As mentioned by [1,14], instance characteristics are crucial in the resolution difficulty. In many social choice applications and datasets, N is small, exact resolution with formulation LOP_ref 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 value, and LP relaxations of every ILP formulation give the integer optimal solution. For this numerical study, as in [15], 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 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 in $[\![0, 100]\!]$ 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 computed 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. Actually, the results were very similar for ILP formulations to the 1-median trivial instances. Real-life structured instances for median of permutations are much easier than random instances. The generators allow to analyze the impact of structured instances.

Number of items N was generated with values $N \in \{20, 30, 40, 50, 100\}$. For $N \in \{20, 30, 40\}$, the Best Known Solution (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 does not find a proven optimal solution in one hour, we note that LocalSolver is also efficient in short time limits. 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 gaps to BKS, denoted BKS(i):

$$gap = \frac{|\ v(i) - BKS(i)\ |}{BKS(i)} \tag{34}$$

5.2 Comparing LP Relaxations

To analyze the quality of polyhedral descriptions recalled in Table 2, Table 3 presents gaps of LP relaxations of ILP formulations for LOP and the naive upper bound (21). Table 4 presents the computation time for LP relaxations,

Table 3. Comparison of the average gaps to the BKS for the LP relaxations of formulations recalled in Table 2 and the naive upper bound (21)

Instances	(21)	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%

to highlight the impact of the number of variables and constraints recalled in Table 2. These tables illustrate the difficulty of instances, aShuf are 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 [15], (3) induces much better LP relaxations for LOP than (14) and (15). Adding (14) and (15) in ILP formulations with (3) or (12) does not induce any difference in the quality of LP relaxation. It explains why in Table 2, we remove constraints of type (3) to compare quality of LP relaxations. An explanation is the different nature of LOP and ATSP problems because of different objective functions: if polyhedrons defined by constraints are identical, objective functions with weighted sums in x or y change the projection on the space of interest.

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, computations were stopped in one hour without termination for $N = 100$. With aggregation (29) instead of (28), LP relaxation is computed quickly, but the quality of LP relaxation is dramatically decreased, the continuous bounds are close to the naive upper bounds (21). 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 (21), constraints (33) do not tighten flow-GP formulation, this result differ from [16].

Table 4. Comparison of the average time (in seconds) to compute LP relaxations for ILP formulations recalled in Table 2

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 LOP_ref is of an excellent quality, which illustrates polyhedral results and proven facets from [11]. In Table 2, LOP_ref 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 (12), no difference was observe adding only (13). These results proves that LOP_SSB2 is in theory strictly tighter than LOP_ref, but with small and rare improvements.

5.3 Comparing Branch and Bound Convergences

Now, we compare the impact of modeling LOP with LOP_ref and LOP_SSB2, in the Branch&Bound (B&B) convergence. Table 5 analyzes the impact of Cplex cuts and heuristics at the root node, before branching in the B&B tree. If LOP_SSB2 improves slightly LP relaxation quality, the open question is to determine if additional variables and constraints 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 higher number of variables, but the efficiency of cuts and primal heuristics is significantly worse with the heavier SSB2 formulation. Having a larger ILP model, slower matrix operations for generation of cutting planes are needed by Cplex, and this stops earlier cuts that would have been generated using LOP_ref formulation, the size

Table 5. Comparison of Lower Bounds (LB) and Upper Bounds (UB) of formulations LOP_ref and LOP_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 ref,SSB2	UB	LB ref	time	UB	LB SSB2	time
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

of ILP matrix is crucial here, contrary to [4]. Table 5 shows that few improvement of LP relaxation is provided at the root node of B&B tree, cuts are not very efficient to improve the LP relaxation, which was of a good quality. This explains the difference in the B&B convergence in one hour allowing branching, LOP_ref formulation is largely superior. For some instances with $N = 40$ or $N = 50$, LOP_ref can converge in ten minutes whereas a significant gap between lower and upper bounds remains after one hour for LOP_SSB2. This definitively validates the LOP_ref formulation as baseline ILP model for [2].

5.4 Variable Fixing Heuristics

The excellent quality of the LP relaxation with LOP_ref formulation allows to use continuous solutions of LP relaxation to design primal heuristics 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. One may use a VF preprocessing for variables with an integer value in the continuous relaxation, expecting that these integer decisions are good. Generally, it makes a difference to apply VF preprocessing on zeros and ones in the LP relaxation, as in [7]. There are in general many possibilities of VF preprocessing, considering also specific rules to select a subset of variable to fix [7].

For LOP, imposing $x_{i,j} = 1$ (resp 0) implies $x_{j,i} = 0$ (resp 1) with constraints (2), so that fixing only ones or only zeros are equivalent to fix all the integer variables, contrary to [7]. Note that constraints (3) may induce continuous solutions $x_{i,j} = x_{j,k} = x_{k,i} = 2/3$, so that rounding such variables induces infeasibility on (3) constraints. This property does not hold when rounding to ones only 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 0.8. Actually, there were slight differences for these two strategies. Experiments also used the quick MTZ relaxation, this significantly degraded the performance of the VF heuristic.

Table 6 compares the gap to BKS and computation time using the VF preprocessing to LOP_ref formulation. For small and easy instances where LOP_ref 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 than 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 an advanced phase of the B&B convergence.

Table 6. 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	time (sec)	LB	time (sec)
	ref		ref + VF	
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

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 reference formulation, but is fewly improved after with cuts and branching. Note that these reformulation issues were an open question raised by [12]. Also, primal heuristics are not efficient on the problem, a basic VF matheuristic significantly improves the primal solutions for difficult instances. Furthermore, this paper illustrates the graduated difficulty of instances, structured instances from the biological application as median of permutations are easier that random instances of LOP.

These results offer perspectives for the biological application also with the extension with ties [1]. Matheuristics can be used in this context, combined to specific reduction space operators related to the easier median of permutation instances [1,14]. 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 by [8,12], to solve larger instances ($N \geqslant 100$).

References

1. Andrieu, P., et al.: Efficient, robust and effective rank aggregation for massive biological datasets. Future Gener. Comput. Syst. **124**, 406–421 (2021)
2. Brancotte, B., Yang, B., Blin, G., Cohen-Boulakia, S., Denise, A., Hamel, S.: Rank aggregation with ties: experiments and analysis. Proc. VLDB Endowment (PVLDB) **8**(11), 1202–1213 (2015)
3. Cohen-Boulakia, S., Denise, A., Hamel, S.: Using medians to generate consensus rankings for biological data. In: Bayard Cushing, J., French, J., Bowers, S. (eds.) SSDBM 2011. LNCS, vol. 6809, pp. 73–90. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-22351-8_5
4. Dupin, N.: Tighter MIP formulations for the discretised unit commitment problem with min-stop ramping constraints. EURO J. Comput. Optim. **5**(1), 149–176 (2017)
5. Dupin, N., Parize, R., Talbi, E.: Matheuristics and column generation for a basic technician routing problem. Algorithms **14**(11), 313 (2021)
6. Dupin, N., Talbi, E.: 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. Dupin, N., Talbi, E.: Parallel matheuristics for the discrete unit commitment problem with min-stop ramping constraints. Int. Trans. Oper. Res. **27**(1), 219–244 (2020)
8. Garcia, C., Pérez-Brito, D., Campos, V., Martí, R.: Variable neighborhood search for the linear ordering problem. Comput. OR **33**(12), 3549–3565 (2006)
9. Gouveia, L., Pires, J.: The asymmetric travelling salesman problem and a reformulation of the Miller-Tucker-Zemlin constraints. Euro. J. Oper. Res. **112**(1), 134–146 (1999)
10. Grötschel, M., Jünger, M., Reinelt, G.: A cutting plane algorithm for the linear ordering problem. Oper. Res. **32**(6), 1195–1220 (1984)
11. Grötschel, M., Jünger, M., Reinelt, G.: Facets of the linear ordering polytope. Math. Program. **33**(1), 43–60 (1985). https://doi.org/10.1007/BF01582010
12. Martí, R., Reinelt, G.: Exact and Heuristic Methods in Combinatorial Optimization: A Study on the Linear Ordering and the Maximum Diversity Problem. Springer, Cham (2022)

13. Miller, C., Tucker, A., Zemlin, R.: Integer programming formulation of traveling salesman problems. J. ACM **7**(4), 326–329 (1960)
14. Milosz, R., Hamel, S.: Space reduction constraints for the median of permutations problem. Discrete Appl. Math. **280**, 201–213 (2020)
15. Öncan, T., Altınel, I., Laporte, G.: A comparative analysis of several asymmetric traveling salesman problem formulations. Comput. OR **36**(3), 637–654 (2009)
16. Peschiera, F., Dell, R., Royset, J., Haït, A., Dupin, N., Battaïa, O.: A novel solution approach with ML-based pseudo-cuts for the flight and maintenance planning problem. OR Spectrum **43**(3), 635–664 (2021)
17. Sarin, S., Sherali, H., Bhootra, A.: New tighter polynomial length formulations for the asymmetric traveling salesman problem with and without precedence constraints. Oper. Res. Lett. **33**(1), 62–70 (2005)
18. Talbi, E.: Combining metaheuristics with mathematical programming, constraint programming and machine learning. Ann. OR **240**(1), 171–215 (2016)
19. Talbi, E.: Machine learning into metaheuristics: a survey and taxonomy. ACM Comput. Surv. (CSUR) **54**(6), 1–32 (2021)