Linear Lexicographic Optimization and Preferential Bidding System

Nour ElHouda Tellache, Frédéric Meunier, Axel Parmentier

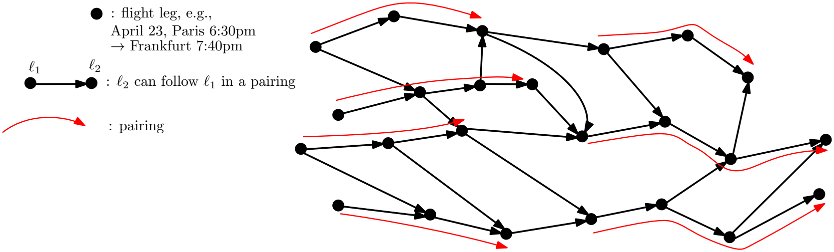
April 24, 2023

Supported by the “chaire Air France–École des Ponts”

Trends in Computational Discrete Optimization, ICERM

Plan

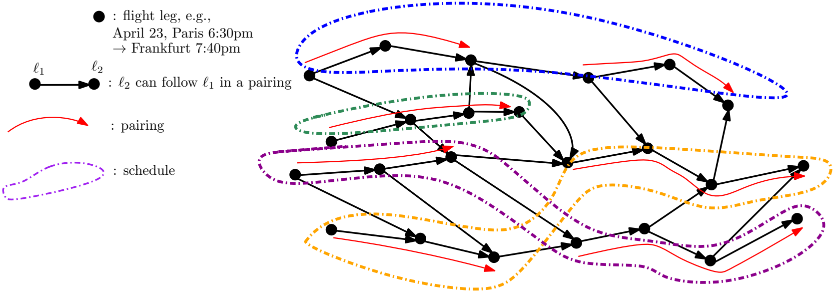
Scheduling problem in Airline Management



- The scheduling problem aims at building the schedules of the pilots for a given month.
- It comes after a few preliminary steps.
- **Input.** Digraph with pairings.
- **Output.** Feasible schedules for the pilots.

Schedule = sequence of pairings

Scheduling problem in Airline Management



Schedule is feasible (at Air France):

- No two pairings overlap in time.
- Number of days on $\leqslant 17$.
- Days off must include at least 7 consecutive days.
- Number of flight hours $\leqslant 85$.
- Number of working hours $\leqslant 55$ for every sequence of 7 consecutive days.

Preferential Bidding System in Airline Management

Preferential Bidding System: used by some airlines to build and assign schedules to their pilots.

Pilots “bid”: they give scores to the pairings.

Then:

- Assign to the most senior pilot the best possible schedule according to his scores, with the constraint that the remaining instance is still feasible.
- Assign to the second most senior pilots the best possible schedule according to his scores, with the constraint that the remaining instance is still feasible.
- Etc.

Not used at Air France.

A lexicographic optimization problem

Input.

- Collection of pairings
- Pilots numbered from 1 (most senior) to m (less senior)
- Scores g_{ip} (given by pilot i to pairing p).

Task. Find an assignment $\sigma \colon \{\text{pilots}\} \to \{\text{feasible schedules}\}$ so that

- $\sigma(\text{pilots})$ forms a partition of the pairings.
- the score of the pilots is lexicographically maximal, i.e.,

$$(c_{1\sigma(1)}, \ldots, c_{m\sigma(m)}) \text{ is lexicographically maximal.}$$

$$c_{is} = \text{score of schedule } s \text{ for pilot } i := \sum_{p \in s} g_{ip} \,.$$

Lexicographic order

Lexicographic order on $\mathbb{R}^m$:

$$\boldsymbol{x} >_{\text{lex}} \boldsymbol{y} \quad \text{if there is } j \text{ with } x_j > y_j \text{ and } x_k = y_k \text{ for all } k < j.$$

$$(2,1,3) >_{\text{lex}} (2,0,5) \quad \text{and} \quad (2,1,3) \not>_{\text{lex}} (2,2,0)\,.$$

A lexicographic problem

Modeling via an ILP:

$$
\begin{array}{lll}
\text{lexmax} & C\boldsymbol{x} & \\
\text{s.t.} & \displaystyle\sum_{s\in\{\text{schedules}\}} x_{is} = 1 & i \in \{\text{pilots}\} \\
& \displaystyle\sum_{i\in\{\text{pilots}\}} \sum_{s\in\{\text{schedules}\}:\ s\ni p} x_{is} = 1 & p \in \{\text{pairings}\} \\
& x_{is} \in \{0,1\} & i \in \{\text{pilots}\}, s \in \{\text{schedules}\}
\end{array}
$$

- $C = (c_{is})_{i,s}$
- "lexmax $C\boldsymbol{x}$": we want to maximize lexicographically the vector $(\sum_s c_{1s}x_{1s}, \sum_s c_{2s}x_{2s}, \ldots, \sum_s c_{ms}x_{ms})$.

A lexicographic problem

Modeling via an ILP:

$$\begin{array}{lll} \text{lexmax} & C\boldsymbol{x} & \\ \text{s.t.} & \displaystyle\sum_{s \in \{\text{schedules}\}} x_{is} = 1 & i \in \{\text{pilots}\} \\ & \displaystyle\sum_{i \in \{\text{pilots}\}} \sum_{s \in \{\text{schedules}\}:\ s \ni p} x_{is} = 1 & p \in \{\text{pairings}\} \\ & x_{is} \in \{0,1\} & i \in \{\text{pilots}\}, s \in \{\text{schedules}\} \end{array}$$

Challenges:

- The number of schedules is huge: column generation.
- The objective is lexicographic.
- It is an integer program.

Plan

Approaches in the literature

Sequential approach (Gamache et al. (1998, 2007)):

- Solve the problem with objective function restricted to the most senior pilot.
- Solve the problem with objective function restricted to the second most senior pilot, with extra constraint fixing the score of the most senior.
- And so on.

(The sequential approach is also used by standard ILP solvers.)

Weighting approach:

- Single objective function where the score of each pilot gets a weight.
- Require weights increasing exponentially with the number of pilots.
- Prevents any use of this approach for realistic instances.

Proposed approach

If there were just a standard one-dimensional objective function, a standard approach would use

- an algorithm for the linear relaxation $\Rightarrow$ upper bound u.
- use the upper bound u to solve the original problem (e.g., branch-and-bound).

We keep the same approach:

- an algorithm for the linear relaxation $\Rightarrow$ upper bound $\boldsymbol{u} \in \mathbb{R}^m$.
- use the upper bound $\boldsymbol{u}$ to solve the original problem.

For the linear relaxation, we propose a column generation method:

- master problem: linear lexicographic programming
- slave problem: resource-constrained shortest paths with lexicographic costs

Plan

Basics in linear programming

$$\begin{aligned} \max \quad & \boldsymbol{c} \cdot \boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geqslant \boldsymbol{0} \end{aligned}$$

A is an $k \times n$ matrix with independent rows.

A feasible basis is a subset B of $[n]$ such that

- A_B is non-singular. (A_B is the submatrix with columns indexed by B.)
- $A_B^{-1}\boldsymbol{b} \geqslant \boldsymbol{0}$.

Solution associated with B: $\boldsymbol{y}_B = A_B^{-1}\boldsymbol{b}$ and $\boldsymbol{y}_{[n]\setminus B} = \boldsymbol{0}$

A basis B is primal-dual feasible if $c_j - \boldsymbol{c}_B^\top A_B^{-1}\boldsymbol{a}_j \leqslant 0$ for all $j \in [n]$.
($\boldsymbol{a}_j$ is the j-th column of A.)

Primal-dual feasible bases

Theorem Dantzig 1947

If B is primal-dual feasible, then the solution associated with B is optimal. Moreover, when the program is bounded, such a basis B always exists.

This theorem is the key result one which column generation is built.

If a new column $\boldsymbol{a}$ is added to a linear program, with a cost c, any idea on how it might improve the objective value? (column generation)

Answer given by reduced cost $= c - \boldsymbol{c}_B^\top A_B^{-1} \boldsymbol{a}$ when B is primal-dual feasible.

Basics in linear lexicographic programming

Setting proposed by Isermann 1982

$$\begin{aligned} \text{lexmax} \quad & C\boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geqslant \boldsymbol{0} \end{aligned}$$

A is an $k \times n$ matrix with independent rows.
C is an $m \times n$ matrix.

A feasible basis is a subset B of $[n]$ such that

- A_B is non-singular. (A_B is the submatrix with columns indexed by B.)
- $A_B^{-1}\boldsymbol{b} \geqslant \boldsymbol{0}$.

Solution associated with B: $\quad \boldsymbol{y}_B = A_B^{-1}\boldsymbol{b} \quad$ and $\quad \boldsymbol{y}_{[n]\setminus B} = \boldsymbol{0}$

A basis B is primal-dual feasible if $\boldsymbol{c}_j - C_B^\top A_B^{-1}\boldsymbol{a}_j \leqslant_{\text{lex}} \boldsymbol{0}$ for all $j \in [n]$.

Primal-dual feasible bases (lexicographic programming)

Lemma Isermann 1982

Any solution associated with a primal-dual feasible basis is optimal.

Lemma Tellache, M., Parmentier 2023

When the program is bounded, there always exists a primal-dual feasible basis.

It paves the way for column generation in linear lexicographic programming.

Solving a linear lexicographic program

We want to solve

$$\begin{aligned} \text{lexmax} \quad & C\boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geqslant \mathbf{0} \end{aligned}$$

Solve = find a primal-dual feasible basis.

Isermann (1982) proposed a simplex method to solve linear lexicographic program.

We use an alternative standard method, which can rely on standard solvers.

Solving a linear lexicographic program

Standard way to solve a linear lexicographic program (e.g., Akgül 1984):

$S^{(1)} := [n]$;
for $\ell = 1, \ldots, m$ **do**
  Solve with any off-the-shelf solver

$$\begin{aligned} \max \quad & \boldsymbol{c}^{\ell}_{S^{(\ell)}} \cdot \boldsymbol{x} \\ \text{s.t.} \quad & A_{S^{(\ell)}} \boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \geqslant \boldsymbol{0}\,; \end{aligned} \qquad (\mathrm{P}_\ell)$$

  $B^{(\ell)} :=$ any primal-dual feasible basis of (P_ℓ);
  $S^{(\ell+1)} := \{ j \in S^{(\ell)} \colon c^{\ell}_j = \boldsymbol{c}^{\ell\top}_{B^{(\ell)}} A^{-1}_{B^{(\ell)}} \boldsymbol{a}_j \}$;
return $B^{(m)}$;

$\boldsymbol{c}^{\ell}$ is the ℓ-th row of C ($= \ell$-th objective function).

Solving a linear lexicographic program

Proposition Tellache, M., Parmentier 2023

The basis $B^{(m)}$ is a primal-dual feasible basis of the linear lexicographic program.

The proof uses the following (easy) lemma from standard linear programming.

Lemma

Let B be a feasible basis. Let S be the components with zero reduced cost and B' a feasible basis included in S. Then the reduced costs computed with respect to B' are equal to those computed with respect to B.

Column generation (lexicographic setting)

We want to solve

$$\begin{aligned} \text{lexmax} \quad & C\boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \qquad \text{(P)} \\ & \boldsymbol{x} \geqslant \boldsymbol{0}\,, \end{aligned}$$

where A has a huge number of columns.

We solve instead

$$\begin{aligned} \text{lexmax} \quad & C'\boldsymbol{x}' \\ \text{s.t.} \quad & A'\boldsymbol{x}' = \boldsymbol{b} \qquad \text{(P')} \\ & \boldsymbol{x}' \geqslant \boldsymbol{0}\,, \end{aligned}$$

where A' and C' are the matrices A and C limited to a subset J of columns.

While there is $\bar{j} \notin J$ such that $\boldsymbol{c}_{\bar{j}} - {C_B}^{\top} {A_B}^{-1} \boldsymbol{a}_{\bar{j}} >_{\text{lex}} \boldsymbol{0}$ (Slave problem):

- Add $\bar{j}$ to J.
- Solve (P'). (Master problem)

Plan

Original problem (reminder)

$$
\begin{aligned}
\text{lexmax} \quad & C\boldsymbol{x} & \\
\text{s.t.} \quad & \sum_{s \in \{\text{schedules}\}} x_{is} = 1 & i \in \{\text{pilots}\} \\
& \sum_{i \in \{\text{pilots}\}} \; \sum_{s \in \{\text{schedules}\}:\ s \ni p} x_{is} = 1 & p \in \{\text{pairings}\} \\
& {\color{red} x_{is} \in \{0, 1\}} & i \in \{\text{pilots}\}, s \in \{\text{schedules}\}
\end{aligned}
$$

Integer part

$$\begin{aligned} \text{lexmax} \quad & C\boldsymbol{x} \\ \text{s.t.} \quad & A\boldsymbol{x} = \boldsymbol{b} \\ & \boldsymbol{x} \in \mathbb{Z}_+^n \,. \end{aligned}$$

Let $\boldsymbol{\ell}, \boldsymbol{u} \in \mathbb{R}^k$ be respectively lower and upper bounds of the linear relaxation.

Lemma Tellache, M., Parmentier 2021+

Consider a primal-dual feasible basis B, and a non-basic variable x_j. If $\boldsymbol{c}_j - C_B^\top A_B^{-1} \boldsymbol{a}_j <_{\text{lex}} \boldsymbol{\ell} - \boldsymbol{u}$, then $x_j = 0$ in all optimal solutions.

It is the extension of a classical trick (Dantzig, Fulkerson, and Johnson (1954)) to the lexicographic setting.

Overall method

1. Solve the linear relaxation to optimality via column generation $\longrightarrow$ upper bound $= \boldsymbol{u}$:
   master and slave problems
2. Solve the integer version (restricted to the previous columns) $\longrightarrow$ lower bound $= \boldsymbol{\ell}$:
   with an off-the-shelf solver
3. Add columns with reduced cost $\geqslant_{\text{lex}} \boldsymbol{\ell} - \boldsymbol{u}$.
   previous lemma, and slave problem, again
4. Solve this new integer version $\longrightarrow$ optimal solution.
   with an off-the-shelf solver, again

Plan

Slave problem

In our context, columns are pairs (i, s) with $i \in \{\text{pilots}\}$ and $s \in \{\text{schedules}\}$.

The slave problem can be treated for each pilot i separately and modeled as

$$\operatorname*{lexmax}_{s \in \{\text{schedules}\}} \boldsymbol{c}_{is} - \boldsymbol{c}_B^\top A_B^{-1} \boldsymbol{a}_{is} \,.$$

The slave problem becomes a resource constrained shortest path problem, with lexicographic costs.

Lex-Resource constrained shortest path problem

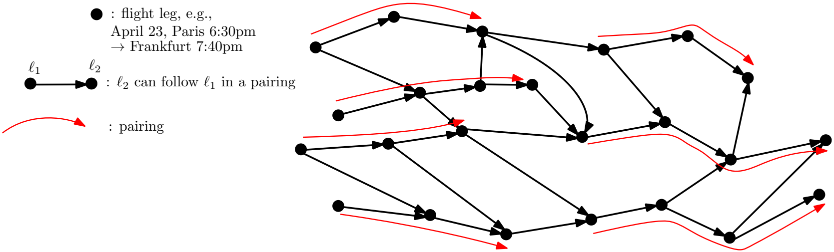
Input.

- Directed graph $D = (V, A)$ with two special vertices o and d
- Partial ordered set $(R, \preccurlyeq)$ of resources with a unique maximal element $\hat{\mathbf{1}}$
- Cost function $c\colon R \to \overline{\mathbb{R}}^m$

Task. Compute a feasible o-d path P (i.e., with $r_P \neq \hat{\mathbf{1}}$), with minimal $c(r_P)$.

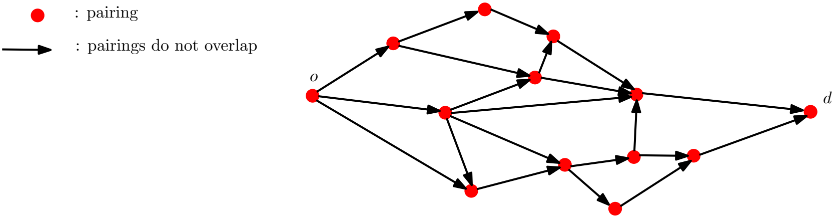
The resource r_P of a path P is defined with an extra function $\oplus$ "summing" the resources along P. The function is also given in input.

We are not working with this graph:



Graph

...but with that graph:



Resource $r \in R$ on each arc:

- number of days on of head pairing
- number of flight hours of head pairing
- 7 days off between tail and head pairings? (Boolean)
- etc.
- cost of head pairing

Bounds to discard paths

Theorem Parmentier 2019

Suppose $(R, \preccurlyeq)$ is a lattice and D is acyclic. Then the $b_v \in R$ solutions of

$$\begin{cases} b_d = \hat{\mathbf{0}} \\ b_v = \bigwedge_{(v,w)\in\delta^+(v)} \left(r_{(v,w)} \oplus b_w\right). \end{cases}$$

are such that $b_v \preccurlyeq r_Q$ for all feasible v-d-path Q. Moreover, such b_v can be computed in polynomial time.

When c is non-decreasing, such bounds can be used to discard paths. Given an o-v path P:

if $c(r_P \oplus b_v) \geqslant$ best cost so far, then P can be discarded.

(!!!Problem formulated as a shortest path problem!!!)

Plan

Experimental results

Server with 192 GB of RAM and 32 cores at 3.30 GHz
Gurobi 9.02

Results of the overall method.

	#pilots	#inst.	#opt.	Average					
				Total (s)	Master pb. (s)	Pricing pbs. (s)	ILLPs (s)	Most sen. w/gap	Gap (%)
I_1	17	5	5	12.0	0.7	1.1	10.1	-	-
I_2	25	5	5	57.0	3.1	42.7	11.2	-	-
I_3	50	5	5	250.4	21.0	121.8	107.6	-	-
I_4	70	5	2	1,118.4	61.3	224.7	832.5	39.3	62.8
I_5	80	5	4	1,967.8	105.1	975.7	887.0	54.0	8.3
I_6	90	5	3	3,576.5	151.1	1,594.3	1,830.2	42.0	22.9
I_7	100	5	2	3,223.4	262.4	606.4	2,354.6	51.3	25.6
I_8	150	1	1	79,556.0	3,011.7	28,657.9	47,886.4	-	-

Thank you