Airline Crew Scheduling by Column Generation

Guy Desaulniers and François Soumis

École Polytechnique, Canada

Spring School on Combinatorial Optimization in Logistics
Montréal, May 17-20, 2010
Collaborators

- Khaled Boubaker, MSc student
- Issmail Elhallaoui, PhD student and post-doc
- Alain Hertz, professor
- Abdelmoutalib Metrane, post-doc
- Mohammed Saddoune, PhD student
- Personnel from Ad Opt/Kronos
Outline

1. Airline planning process
2. Crew pairing and column generation
3. Crew assignment and dynamic constraint aggregation
4. Integrated crew scheduling and bi-dynamic constraint aggregation
5. Conclusions and future work
Outline

1. Airline planning process
2. Crew pairing and column generation
3. Crew assignment and dynamic constraint aggregation
4. Integrated crew scheduling and bi-dynamic constraint aggregation
5. Conclusions and future work
Operations research provides several tools to the airlines for planning their operations.

This planning process aims at maximizing the airline profits.

However, it is a complex process that involves several departments:

- Marketing for flight scheduling
- Scheduling for aircraft assignment and routing
- Flight operations for crew scheduling
Operations research provides several tools to the airlines for planning their operations.

This planning process aims at maximizing the airline profits.

However, it is a complex process that involves several departments:
- Marketing for flight scheduling
- Scheduling for aircraft assignment and routing
- Flight operations for crew scheduling
To simplify it, this process is typically divided into four main steps:

1 - Flight scheduling

- Given aircraft availability per type, determine the schedule of flights to operate over a given season in order to maximize anticipated profits
- Cyclic *weekly problem* with exceptions for holidays
- Starts usually from the previous year schedule
- Requires a passenger flow model to evaluate revenues
  - Estimated passenger demand per day and OD
  - Passenger preferences (departure time, number of legs, etc.)
  - Seating capacity per flight
- Output: number of passengers per itinerary
2 - Fleet assignment

- Given a weekly flight schedule, aircraft availability per type, estimated profits for each flight and aircraft type, find the **aircraft type** to assign to each flight so as to maximize total estimated profits while ensuring aircraft flow conservation in the network.
- Cyclic **weekly problem** with exceptions for holidays.
- Profits per flight and type are approximated via a passenger flow model (iterative process).
3 - Aircraft routing

- Given a flight schedule for an aircraft type and aircraft availability for this type, determine aircraft routes that satisfy short-term maintenance requirements.
- One maintenance (inspection lasting about 4 hours) at most at every three or four days.
- Cyclic weekly problem or dated monthly problem.
- In general, feasibility problem or through value maximization problem.
4 - Crew scheduling

- Given a monthly flight schedule for an aircraft type, determine the schedules of the crew members
- Typically divided into two steps: crew pairing and crew assignment

Remarks

- This decomposition process is heuristic and often requires backtracking
- For small- and medium-sized airlines, certain steps can be integrated
4 - Crew scheduling

- Given a monthly flight schedule for an aircraft type, determine the schedules of the crew members
- Typically divided into two steps: crew pairing and crew assignment

Remarks

- This decomposition process is heuristic and often requires backtracking
- For small- and medium-sized airlines, certain steps can be integrated
Outline

1. Airline planning process
2. Crew pairing and column generation
3. Crew assignment and dynamic constraint aggregation
4. Integrated crew scheduling and bi-dynamic constraint aggregation
5. Conclusions and future work
Crew pairing problem

Separable by

- Crew category: cockpit (pilots and copilots) and cabin (flight directors and attendants)
- Aircraft type or family of types

Definitions

- Crews are assigned to a few crew bases
- A pairing is a sequence of duties interspersed with rest periods, starting and ending at a crew base
- A duty is a sequence of flights, deadheads and connections forming a working day
- A deadhead is a flight (of any company) on which the crew travels as passengers
Crew pairing problem

Separable by

- Crew category: cockpit (pilots and copilots) and cabin (flight directors and attendants)
- Aircraft type or family of types

Definitions

- Crews are assigned to a few crew bases
- A pairing is a sequence of duties interspersed with rest periods, starting and ending at a crew base
- A duty is a sequence of flights, deadheads and connections forming a working day
- A deadhead is a flight (of any company) on which the crew travels as passengers
The cost of a pairing is a complex function of:

- flying time per duty (minimum guaranteed)
- span of each duty
- span of the pairing
- deadhead costs
- accommodation fees (for rests outside the base)
A pairing is feasible if it satisfies all safety and collective agreement rules such as:

- maximum number of calendar days in a pairing
- maximum number of duties in a pairing
- minimum rest time between two consecutive duties
- maximum number of landings per duty
- maximum span of a duty
- maximum flying time per duty
- minimum connection time between two consecutive flights
Crew pairing problem statement

- **Given**
  - a set of scheduled flights
  - safety and working rules
  - minimum or maximum credited hours per crew base
- **Find** least-cost feasible crew pairings
- **Such that**
  - each flight is covered by an active crew
  - the minimum or maximum credited hours per crew base is respected

**Remark**
A typical crew pairing problem instance is defined for a one-month dated flight schedule and solved one month ahead of time.
Crew pairing problem statement

- **Given**
  - a set of scheduled flights
  - safety and working rules
  - minimum or maximum credited hours per crew base

- **Find** least-cost feasible crew pairings

- **Such that**
  - each flight is covered by an active crew
  - the minimum or maximum credited hours per crew base is respected

**Remark**

A typical crew pairing problem instance is defined for a one-month dated flight schedule and solved one month ahead of time
Traditional sequential approach

- Solve a cyclic daily problem
- Unfold the computed solution over a typical week (one copy of each pairing for each week day) and remove infeasible copies
- Solve a cyclic weekly problem preserving as much as possible the unfolded daily solution
- Unfold the computed weekly solution over the month (one copy of each pairing for each week of the month) and remove infeasible copies
- Solve the dated monthly problem preserving as much as possible the unfolded weekly solution
Minimize \[ \sum_{p \in P} c_p y_p \] \hspace{1cm} (1)

s.t. \[ \sum_{p \in P} a_{fp} y_p = 1, \quad \forall f \in F \] \hspace{1cm} (2)

\[ \sum_{p \in P} b_{qp} y_p = e_q, \quad \forall q \in Q \] \hspace{1cm} (3)

\[ y_p \in \{0, 1\}, \quad \forall p \in P \] \hspace{1cm} (4)

\( P \): set of all feasible pairings;
\( c_p \): cost of pairing \( p \in P \);
\( F \): set of flights to cover;
\( a_{fp} \): binary parameter taking value 1 if flight \( f \in F \) is actively covered in pairing \( p \in P \) and 0 otherwise;
\( Q \): set of indices for the side constraints;
\( b_{qp} \): contribution of pairing \( p \in P \) to side constraint \( q \in Q \);
\( e_q \): right-hand side of side constraint \( q \in Q \);
\( y_p \): binary variable taking value 1 if pairing \( p \) is selected and 0 otherwise.
A priori generation of a subset of pairings

- Anbil et al. (1992), Hoffman and Padberg (1993), Chu et al. (1997)

Dynamic column generation based on resource-constrained shortest path subproblems

- Desaulniers et al. (1997), Vance et al. (1997), Barnhart and Shenoi (1998)

Dynamic column generation based on truncated depth-first search enumeration


Robust crew scheduling

Column generation

Basics

- Column generation is used to solve the linear relaxation of model (1)-(4), which is called the master problem.
- Iterative method alternating between a restricted master problem (RMP) and several subproblems.
- At iteration $i$, the RMP is simply the master problem restricted to a subset of its variables.
- The RMP is solved by a linear programming solver to produce a primal and a dual solution.
This primal solution is optimal for the master problem if the reduced costs of all variables are nonnegative.

This condition holds for all known variables appearing in the RMP.

\[ \tilde{C}_p \geq 0 \] 

\[ \tilde{C}_p \text{ ??} \]
Basics (cont’d)

- One or several **subproblems** must be solved, using a specialized algorithm, to determine if it holds for the unknown variables.
- Each subproblem searches for a **least reduced cost variable** among a subset of variables.

- If the least reduced cost is nonnegative for all subproblems, then the primal solution of the current RMP is also **optimal for the master problem** and the column generation algorithm stops.
- Otherwise, negative reduced cost variables identified by the subproblems are **added to the RMP** before starting a new iteration.
Basics (cont’d)

- One or several subproblems must be solved, using a specialized algorithm, to determine if it holds for the unknown variables.

- Each subproblem searches for a least reduced cost variable among a subset of variables.

- If the least reduced cost is nonnegative for all subproblems, then the primal solution of the current RMP is also optimal for the master problem and the column generation algorithm stops.

- Otherwise, negative reduced cost variables identified by the subproblems are added to the RMP before starting a new iteration.
Starts by solving the RMP with
- Artificial variables or
- Initial set of columns containing a feasible RMP solution
Subproblems

For the weekly crew pairing problem

- There is one subproblem per crew base and per day of the week.
- Such a subproblem is a shortest path problem with resource constraints that aims at finding a feasible pairing starting on the corresponding day at the corresponding base with the least reduced cost.
- A subproblem is defined over an acyclic time-space network.
Airline Crew Scheduling by Column Generation

Crew pairing and column generation

Column generation

Legend for nodes:
- Departure node
- Arrival node
- Source node
- Sink node

Legend for arcs:
- Flight arc
- Waiting arc
- Rest arc
- Deadhead arc
- Start of duty arc
- Start and end of pairing arcs
Path feasibility

- Each feasible pairing starting at this base and on this day is represented by a source-to-sink path in this network.
- However, not all paths correspond to a feasible pairing.
- Resource constraints are used to restrict path feasibility.
- A resource is a quantity that varies along a path and its value must fall within a prespecified resource window (interval) at each node.
Path feasibility

- Each feasible pairing starting at this base and on this day is represented by a source-to-sink path in this network.
- However, not all paths correspond to a feasible pairing.
- **Resource constraints** are used to restrict path feasibility.
- A **resource** is a quantity that varies along a path and its value must fall within a prespecified **resource window** (interval) at each node.
Resource constraints: example

A resource can be used to limit the number of landings per duty (maximum 5)

- The resource window is \([0,0]\) at the source node and \([0,5]\) at every other node
- The resource increases by 1 on every flight and deadhead arc
- It is reset to 0 on every rest arc

Remark

For a real-life crew pairing problem, between 5 and 10 resources are needed
Resource constraints: example

A resource can be used to limit the number of landings per duty (maximum 5)

- The resource window is [0,0] at the source node and [0,5] at every other node
- The resource increases by 1 on every flight and deadhead arc
- It is reset to 0 on every rest arc

Remark

For a real-life crew pairing problem, between 5 and 10 resources are needed
Arc (reduced) costs

- The cost of a feasible path \( p \) must be equal to the reduced cost of the corresponding variable \( y_p \)
- Every arc bears its original cost, possibly modified by dual values
- The dual variable \( \pi_f \) associated with (2) for flight \( f \) is subtracted on every flight arc representing \( f \)
- The dual variables associated with (3) must also be subtracted
  - For instance, if a constraint (3) limits the number of credited hours for a base \( w \), then \( b_a \sigma_w \) must be subtracted on an arc \( a \) in a subproblem for base \( w \) where \( b_a \) is the number of credited hours allocated on arc \( a \) and \( \sigma_w \) the corresponding dual variable
The shortest path subproblems with resource constraints are solved by a labeling algorithm.

- A label represents a partial path starting at the source node.
- A label contains one component per resource and one (reduced) cost component.
- Labels are extended along the arcs using label extension functions to create longer partial paths.
- Dominance rules are used to discard non-promising (non-Pareto-optimal) labels.
Consider a label \( E_i = (Z_i, L_i, F_i, S_i) \) with:

- \( Z_i \): reduced cost of the partial path
- \( L_i \): number of landings in current duty up to \( i \)
- \( F_i \): flying time in current duty up to \( i \)
- \( S_i \): span of current duty up to \( i \)

\[
E_i = (225, 1, 250, 295) \\
E_j = (-275, 2, 340, 385)
\]

- Consider a flight arc \((i, j)\) for 90-minute flight \( f \)
  \[ \text{cost} = -\pi_f = -500 \]
  \[
  E_i = (225, 1, 250, 295) \\
  E_j = (-275, 2, 340, 385)
  \]

- Consider a rest arc \((j, l)\) for a 12-hour rest
  \[ \text{cost} = 0 \]
  \[
  E_j = (-275, 2, 340, 385) \\
  E_l = (-165, 0, 0, 0)
  \]
Consider three labels associated with partial paths ending at the same node:

- \( E_1 = (-275, 2, 340, 385) \)
- \( E_2 = (-320, 2, 225, 360) \)
- \( E_3 = (-100, 1, 200, 245) \)

Assuming that all extension functions are non-decreasing:

- \( E_2 \) dominates \( E_1 \) which can be discarded.
- \( E_2 \) does not dominate \( E_3 \) and vice versa.

Labels \( E_2 \) and \( E_3 \) must be kept.

More complex dominance rule in practice because extension functions are not necessarily non-decreasing.
Heuristics

In practice

- When there are many resources, use an **aggressive dominance rule** (eliminate labels that should be kept)
- To avoid tailing-off, **stop column generation prematurely** when the objective function decrease in the last iterations is insufficient
Column generation is embedded into a branch-and-bound tree to obtain integer solutions → Branch-and-price method

Branching decisions for exact method
- Not easy and efficient to impose $y_p = 0$
- Impose or forbid two flights to be covered consecutively (inter-task or follow-on branching)

Branching decisions for heuristic method
- Set $y_p = 1$ for the highest fractional variable $y_p$
- Set the highest fractional inter-task flow to 1
Column generation is embedded into a branch-and-bound tree to obtain integer solutions → Branch-and-price method

**Branching decisions for exact method**
- **Not easy and efficient** to impose $y_p = 0$
- Impose or forbid two flights to be covered consecutively (inter-task or follow-on branching)

**Branching decisions for heuristic method**
- Set $y_p = 1$ for the highest fractional variable $y_p$
- Set the highest fractional inter-task flow to 1
Column generation is embedded into a branch-and-bound tree to obtain integer solutions \( \Rightarrow \) Branch-and-price method

<table>
<thead>
<tr>
<th>Branching decisions for exact method</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Not easy and efficient to impose ( y_p = 0 )</td>
</tr>
<tr>
<td>- Impose or forbid two flights to be covered consecutively (inter-task or follow-on branching)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Branching decisions for heuristic method</th>
</tr>
</thead>
<tbody>
<tr>
<td>- Set ( y_p = 1 ) for the highest fractional variable ( y_p )</td>
</tr>
<tr>
<td>- Set the highest fractional inter-task flow to 1</td>
</tr>
</tbody>
</table>
Saddoune et al. (2009) propose to solve the crew pairing problem directly over the month using a **rolling horizon procedure**

- Divide the month into overlapping time slices
- In chronological order, solve the problem restricted to each time slice taking into account the solution of the previous slice
- This solution imposes initial conditions for the current slice problem
Computational results

## Irregular instance characteristics

<table>
<thead>
<tr>
<th>Instance</th>
<th>Daily</th>
<th>Weekly</th>
<th>Monthly</th>
<th>Bases</th>
<th>Stations</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>21</td>
<td>175</td>
<td>1011</td>
<td>3</td>
<td>26</td>
</tr>
<tr>
<td>I2</td>
<td>39</td>
<td>338</td>
<td>1463</td>
<td>3</td>
<td>35</td>
</tr>
<tr>
<td>I3</td>
<td>50</td>
<td>412</td>
<td>1793</td>
<td>3</td>
<td>41</td>
</tr>
<tr>
<td>I4</td>
<td>146</td>
<td>1202</td>
<td>5466</td>
<td>3</td>
<td>49</td>
</tr>
<tr>
<td>I5</td>
<td>158</td>
<td>1229</td>
<td>5639</td>
<td>3</td>
<td>34</td>
</tr>
<tr>
<td>I6</td>
<td>162</td>
<td>1274</td>
<td>5755</td>
<td>3</td>
<td>52</td>
</tr>
<tr>
<td>I7</td>
<td>206</td>
<td>1637</td>
<td>7527</td>
<td>3</td>
<td>54</td>
</tr>
</tbody>
</table>
Figure: The weekly regularity

Figure: The monthly regularity
## Solution Fat (additional credited flying time)

<table>
<thead>
<tr>
<th>Instance</th>
<th>3P</th>
<th>RH</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>9.2%</td>
<td>6.7%</td>
<td>27.2%</td>
</tr>
<tr>
<td>I2</td>
<td>13.7%</td>
<td>8.5%</td>
<td>40.0%</td>
</tr>
<tr>
<td>I3</td>
<td>10.9%</td>
<td>7.6%</td>
<td>30.2%</td>
</tr>
<tr>
<td>I4</td>
<td>7.3%</td>
<td>5.0%</td>
<td>31.5%</td>
</tr>
<tr>
<td>I5</td>
<td>2.5%</td>
<td>1.2%</td>
<td>52.0%</td>
</tr>
<tr>
<td>I6</td>
<td>4.2%</td>
<td>3.2%</td>
<td>23.8%</td>
</tr>
<tr>
<td>I7</td>
<td>4.1%</td>
<td>2.7%</td>
<td>34.1%</td>
</tr>
</tbody>
</table>

**Average** 34.1%

3P: three-phase  
RH: rolling horizon
### CPU times (in minutes)

<table>
<thead>
<tr>
<th>Instance</th>
<th>Daily</th>
<th>Weekly</th>
<th>Monthly</th>
<th>Total</th>
<th>RH</th>
<th>Reduction</th>
</tr>
</thead>
<tbody>
<tr>
<td>I1</td>
<td>&lt; 0.1</td>
<td>0.2</td>
<td>3.4</td>
<td>3.6</td>
<td>3.3</td>
<td>8.3%</td>
</tr>
<tr>
<td>I2</td>
<td>0.2</td>
<td>1.7</td>
<td>6.8</td>
<td>8.7</td>
<td>5.4</td>
<td>37.9%</td>
</tr>
<tr>
<td>I3</td>
<td>0.5</td>
<td>3.8</td>
<td>18.2</td>
<td>22.5</td>
<td>15.9</td>
<td>29.3%</td>
</tr>
<tr>
<td>I4</td>
<td>10.8</td>
<td>200.0</td>
<td>823.5</td>
<td>1034.3</td>
<td>756.6</td>
<td>26.8%</td>
</tr>
<tr>
<td>I5</td>
<td>2.6</td>
<td>89.1</td>
<td>284.1</td>
<td>375.8</td>
<td>222.9</td>
<td>40.7%</td>
</tr>
<tr>
<td>I6</td>
<td>3.4</td>
<td>101.7</td>
<td>313.7</td>
<td>418.8</td>
<td>292.7</td>
<td>30.1%</td>
</tr>
<tr>
<td>I7</td>
<td>4.4</td>
<td>140.2</td>
<td>535.3</td>
<td>679.9</td>
<td>493.0</td>
<td>27.5%</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td><strong>28.4%</strong></td>
</tr>
</tbody>
</table>

Guy Desaulniers and François Soumis (École Polytechnique, Canada)
### Flight number repetitions

Regular instances of weekly problem

<table>
<thead>
<tr>
<th>Instance</th>
<th>NR</th>
<th>WR</th>
<th>Reduction</th>
<th>No.</th>
<th>%</th>
</tr>
</thead>
<tbody>
<tr>
<td>R1</td>
<td>6.9%</td>
<td>7.5%</td>
<td>−8.7%</td>
<td>36</td>
<td>100.0%</td>
</tr>
<tr>
<td>R2</td>
<td>20.9%</td>
<td>19.8%</td>
<td>5.3%</td>
<td>26</td>
<td>38.5%</td>
</tr>
<tr>
<td>R3</td>
<td>26.7%</td>
<td>10.3%</td>
<td>61.4%</td>
<td>34</td>
<td>51.3%</td>
</tr>
<tr>
<td>R4</td>
<td>5.6%</td>
<td>4.6%</td>
<td>17.9%</td>
<td>51</td>
<td>61.8%</td>
</tr>
<tr>
<td>R5</td>
<td>4.1%</td>
<td>4.0%</td>
<td>2.4%</td>
<td>51</td>
<td>26.4%</td>
</tr>
<tr>
<td>R6</td>
<td>4.8%</td>
<td>4.6%</td>
<td>4.2%</td>
<td>63</td>
<td>38.5%</td>
</tr>
<tr>
<td>R7</td>
<td>7.2%</td>
<td>5.2%</td>
<td>27.8%</td>
<td>77</td>
<td>25.4%</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>10.9%</strong></td>
<td><strong>8.0%</strong></td>
<td><strong>15.8%</strong></td>
<td><strong>48.3</strong></td>
<td><strong>48.8%</strong></td>
</tr>
</tbody>
</table>
Outline

1. Airline planning process
2. Crew pairing and column generation
3. Crew assignment and dynamic constraint aggregation
4. Integrated crew scheduling and bi-dynamic constraint aggregation
5. Conclusions and future work
Crew assignment problem

Separable by

- Crew member category (pilot, copilot, flight director, and flight attendant)
- Aircraft type or family of types
- Crew base

- Each crew member is assigned to a base and must cover pairings assigned to this base
- A schedule is a sequence of pairings, rests and days off that may also contain vacation and training periods
- A schedule is feasible if it satisfies all safety and collective agreement rules, including those pertaining to crew pairings
Crew assignment problem

Separable by

- Crew member category (pilot, copilot, flight director, and flight attendant)
- Aircraft type or family of types
- Crew base

Each crew member is assigned to a base and must cover pairings assigned to this base

A **schedule** is a sequence of pairings, rests and days off that may also contain vacation and training periods

A schedule is feasible if it satisfies all **safety and collective agreement rules**, including those pertaining to crew pairings
Examples of rules

- Minimum and maximum number of credited hours per month
- Minimum number of days off per month
- Minimum rest time between two pairings
- Maximum number of consecutive working days
- Valid day off patterns for the month (2 consecutive days at least 4 times, including one weekend)
Three construction modes

**Bidline**
- Anonymous schedules with equity (credited hours and days off) are built first
- Crew members bid on these schedules
- Schedules are assigned to the members according to their bids and seniority

**Rostering**
- Takes into account
  - Vacation and training periods
  - Unfinished pairings from the previous month
  - Employees’ preferences (in some cases)
- Personalized schedules that maximize the total of the employees’ preferences or balance credited hours and days off
Three construction modes

Bidline

- Anonymous schedules with equity (credited hours and days off) are built first
- Crew members bid on these schedules
- Schedules are assigned to the members according to their bids and seniority

Rostering

- Takes into account
  - Vacation and training periods
  - Unfinished pairings from the previous month
  - Employees’ preferences (in some cases)
- Personalized schedules that maximize the total of the employees’ preferences or balance credited hours and days off
Preferential bidding

- Takes into account
  - Vacation and training periods
  - Unfinished pairings from the previous month
  - Sophisticated weighted preferences
    - 1000 to fly to Paris
    - 200 for a day off on the 22nd
    - 2000 for not working with Guy

- Sequential construction of schedules that maximize the total of the weights obtained by each employee from the most senior to the most junior
Crew assignment problem statement

- **Given**
  - Set of pairings
  - For each pairing
    - Total number of employees required
    - Minimum number of employees per language (flight attendants)
  - Set of available crew members
  - For each crew member (except for bidline)
    - Vacation and training periods
    - Spoken languages (if needed)
    - Weights for preferences (if needed)
  - Safety and working rules
Construct feasible schedules according to the selected mode

Such that

- A schedule is constructed for each employee
- Each pairing is covered with the right number of employees (total and minimum per language)

Remark

A typical crew assignment problem instance is defined for a one-month dated flight schedule and solved one month ahead of time
Construct feasible schedules according to the selected mode

Such that

- A schedule is constructed for each employee
- Each pairing is covered with the right number of employees (total and minimum per language)

Remark

A typical crew assignment problem instance is defined for a one-month dated flight schedule and solved one month ahead of time
Bidline

- Jarrah and Diamond (1997): a priori column generation
- Campbell et al. (1997): simulated annealing
- Christou et al. (1998): genetic algorithm
- Weir and Johnson (2004): MIP three-phase method
- Boubaker et al. (2010): column generation/dynamic constraint aggregation
Rostering

- Set partitioning approaches

- Column generation methods

Preferential bidding

- Column generation methods
  - Gamache et al. (1998), Achour et al. (2007)
Rostering

- Set partitioning approaches

- Column generation methods

Preferential bidding

- Column generation methods
  - Gamache et al. (1998), Achour et al. (2007)
Bidline for pilots: set partitioning type model

\[ \text{Minimize} \quad \sum_{b \in B} c_b x_b \]
\[ \text{s.t.} \quad \sum_{b \in B} a_{pb} x_b = 1, \quad \forall p \in P \]
\[ \sum_{b \in B} x_b = m \]
\[ x_b \in \{0, 1\}, \quad \forall b \in B \]

*P*: set of pairings to cover
*B*: set of feasible bidlines
*c*<sub>*b*</sub>: cost of bidline *b* \( \in B \)
*a*<sub>*p*b</sub>: binary parameter taking value 1 if bidline *b* \( \in B \) contains pairing *p*, and 0 otherwise
*m*: number of available pilots
*x*<sub>*b*</sub>: binary variable equal to 1 if bidline *b* is selected and 0 otherwise
Column generation for bidline

Subproblems

- Anonymous schedules yield a single subproblem
- In Boubaker et al. (2010), several subproblems because of a complex objective function (minimizing a weighted sum of the variances of the number of days off per schedule and of the number of credited hours per schedule)
- Subproblems are shortest path problems with resource constraints defined on an acyclic network
- Between 4 and 8 resources for a real-life problem
Airline Crew Scheduling by Column Generation

Legend for nodes:
- pairing start
- pairing end
- midnight
- source
- sink

Legend for arcs:
- pairing
- post-courrier
- day off
- start of schedule
- end of schedule
- start of pairing
Dynamic constraint aggregation (DCA)

Introduced in

- Aims at reducing degeneracy in linear programs with set partitioning constraints by working with lower-dimensional bases
- Can be combined with column generation (reduce degeneracy in RMP)
Dynamic constraint aggregation (DCA)

Introduced in


- Aims at **reducing degeneracy** in linear programs with set partitioning constraints by working with lower-dimensional bases
- Can be combined with column generation (reduce degeneracy in RMP)
Basic concepts of constraint aggregation

- Only for set partitioning constraints
- Tasks are partitioned into clusters $\Rightarrow$ task partition

- Guy Desaulniers and François Soumis (École Polytechnique, Canada)
Task clusters

- Task partition built according to a subset of columns
- One task cluster per color (identical rows)
- Higher degeneracy generally yields less clusters
A cluster is represented by a single constraint in the **Aggregated Restricted Master Problem (ARMP)**

<table>
<thead>
<tr>
<th>BASIC</th>
<th>NON BASIC</th>
<th>TASKS</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 1</td>
<td></td>
<td>1/2</td>
</tr>
<tr>
<td>0 1 1</td>
<td></td>
<td>0</td>
</tr>
<tr>
<td>1 1 0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

\[ 1/2 = 1 \]
\[ 0 = 1 \]
\[ 1 = 1 \]
A column is **compatible** with a partition if, for each cluster, it covers all its tasks or none of them.
Airline Crew Scheduling by Column Generation

Crew assignment and dynamic constraint aggregation

Dynamic constraint aggregation

Partition Q

Cluster 1

Cluster 2

Cluster 3

Cluster 4

Cluster 5

Red path

Blue path

Green path is incompatible with Q

are compatible with Q

Guy Desaulniers and François Soumis (École Polytechnique, Canada)
The ARMP contains only compatible columns

<table>
<thead>
<tr>
<th>BASIC</th>
<th>NON BASIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 1</td>
<td>1 0 0 1</td>
</tr>
<tr>
<td>0 1 1</td>
<td>0 1 0 1</td>
</tr>
<tr>
<td>....</td>
<td>....</td>
</tr>
<tr>
<td>1 1 0</td>
<td>0 0 1 1</td>
</tr>
<tr>
<td>0 1 1</td>
<td>0 1 0 0</td>
</tr>
</tbody>
</table>

1 = 1, 0 = 1, 1 = 1
Algorithm

- Set an initial task partition
- Construct the ARMP

Solve the ARMP

Primal solution & Aggregated dual values

Disaggregate dual values

Solve the subproblems

Change the partition?

No

- Update partition (disaggregate or aggregate)
### Aggregated dual values

The image shows a matrix table with the following structure:

<table>
<thead>
<tr>
<th>BASIC</th>
<th>NON BASIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 1 1</td>
<td>0 0 0 1</td>
</tr>
<tr>
<td>0 1 1 0</td>
<td>1 0 1 0</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>1 1 0 0</td>
<td>0 1 1 0</td>
</tr>
</tbody>
</table>

The dual values are indicated for each row and column.

- $\alpha_1 = 1$
- $\alpha_4 = 1$
- $\alpha_6 = 1$

The image also includes a graphical representation of the dual values with arrows pointing from the matrix to the dual values. The matrix is labeled with tasks and columns categorized as basic and non-basic.
Disaggregated dual values

<table>
<thead>
<tr>
<th>BASIC</th>
<th>NON BASIC</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 0 1 1 1 0</td>
<td>1 1 0 0 1 0 1 1 0 0</td>
</tr>
<tr>
<td>1 0 1 1 1 1</td>
<td>1 0 1 0 1 0 0 0 1 1</td>
</tr>
<tr>
<td>1 0 1 0 1 1</td>
<td>1 1 0 0 1 0 0 0 0 1</td>
</tr>
<tr>
<td>0 1 1 1 1 0</td>
<td>1 0 1 1 0 0 0 1 1 0 0</td>
</tr>
<tr>
<td>0 1 1 1 0 0</td>
<td>1 0 1 1 1 1 0 1 0 0 1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TASKS</th>
<th>1/2</th>
<th>0</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 1 0 0 1 0</td>
<td>1 0 1 0 0 1 1 1 1</td>
<td></td>
</tr>
<tr>
<td>1 1 0 1 1 0 1</td>
<td>1 0 1 0 0 1 0 0 1 1</td>
<td></td>
</tr>
</tbody>
</table>

\[ \alpha_1 = 1 \]
\[ \alpha_2 = 1 \]
\[ \alpha_3 = 1 \]
\[ \alpha_4 = 1 \]
\[ \alpha_5 = 1 \]
\[ \alpha_6 = 1 \]
**Dual variable disaggregation**

- \( L \): set of cluster indices; \( W_I \): set of tasks in cluster \( I \)
- \( P \): selected subset of incompatible columns

\[
\alpha_I = \sum_{w \in W_I} \alpha_I^w, \quad \forall I \in L \tag{5}
\]

\[
c_p \geq \sum_{I \in L} \sum_{w \in W_I} a_{lp}^w \alpha_I^w, \quad \forall p \in P \tag{6}
\]

- (5) ensure the same (nonnegative) reduced cost for all compatible variables in the ARMP
- (6) ensure a nonnegative reduced cost for the subset \( P \) of incompatible variables
Dual variables can be disaggregated by solving a linear system of equalities and inequalities.

Difficult to solve in general.

With assumptions on the columns in subset P, solving this system becomes equivalent to solving a series of shortest path problems.
Incompatibility types

(Arbitrary) ordered tasks in clusters

- \( p_1 \): S-incompatible
- \( p_2 \): E-incompatible
- \( p_3 \): M-incompatible
- \( p_4 \): SE-incompatible
- \( p_5 \): ES-incompatible
- \( p_6 \): O-incompatible
Variable substitution

\[ \beta_l^w = \sum_{j=1}^{w} \alpha_j^l, \quad \forall w \in W_l, \ l \in L \]

For example, for a cluster with 4 ordered tasks:

\[
\begin{align*}
\beta_1^1 &= \alpha_1^1 \\
\beta_1^2 &= \alpha_1^1 + \alpha_2^1 \\
\beta_1^3 &= \alpha_1^1 + \alpha_2^1 + \alpha_3^1 \\
\beta_1^4 &= \alpha_1^1 + \alpha_2^1 + \alpha_3^1 + \alpha_4^1
\end{align*}
\]

Constraints (5) become

\[ \beta_l^{\mid W_l \mid} = \alpha_l, \ \forall l \in L \]
Variable substitution

\[ \beta_l^w = \sum_{j=1}^{w} \alpha_j^l, \quad \forall w \in W_l, \ l \in L \]

For example, for a cluster with 4 ordered tasks:

\[ \beta_1^1 = \alpha_1^1 \]
\[ \beta_1^2 = \alpha_1^1 + \alpha_2^1 \]
\[ \beta_1^3 = \alpha_1^1 + \alpha_2^1 + \alpha_3^1 \]
\[ \beta_1^4 = \alpha_1^1 + \alpha_2^1 + \alpha_3^1 + \alpha_4^1 \]

Constraints (5) become

\[ \beta_l^{\mid W_l \mid} = \alpha_l, \quad \forall l \in L \]
Transformation of constraints (6)

<table>
<thead>
<tr>
<th>Category</th>
<th>Transformed constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$ is S-incompatible</td>
<td>$\beta_i^m \leq c_p - \sum_{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is E-incompatible</td>
<td>$-\beta_i^{m-1} \leq c_p - \alpha_i - \sum_{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is M-incompatible</td>
<td>$\beta_i^n - \beta_i^{m-1} \leq c_p - \sum_{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is SE-incompatible</td>
<td>$\beta_i^m - \beta_i^{m+n-1} \leq c_p - \alpha_i - \sum_{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is ES-incompatible</td>
<td>$\beta_j^n - \beta_i^{m-1} \leq c_p - \alpha_i - \sum_{l \in L_p} \alpha_l$</td>
</tr>
</tbody>
</table>

- Limiting the subset $P$ to these types, the linear system (5)–(6) transforms into a set of inequalities corresponding to the constraints of the dual of a shortest path problem.
- This dual might be infeasible $\implies$ Reduce $P$. 
Transformation of constraints (6)

<table>
<thead>
<tr>
<th>Category</th>
<th>Transformed constraint</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p$ is S-incompatible</td>
<td>$\beta^m_i \leq c_p - \sum_{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is E-incompatible</td>
<td>$-\beta^{m-1}<em>i \leq c_p - \alpha_i - \sum</em>{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is M-incompatible</td>
<td>$\beta^n_i - \beta^{m-1}<em>i \leq c_p - \sum</em>{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is SE-incompatible</td>
<td>$\beta^m_i - \beta^{m+n-1}<em>i \leq c_p - \alpha_i - \sum</em>{l \in L_p} \alpha_l$</td>
</tr>
<tr>
<td>$p$ is ES-incompatible</td>
<td>$\beta^n_j - \beta^{m-1}<em>i \leq c_p - \alpha_i - \sum</em>{l \in L_p} \alpha_l$</td>
</tr>
</tbody>
</table>

- Limiting the subset $P$ to these types, the linear system (5)–(6) transforms into a set of inequalities corresponding to the constraints of the dual of a shortest path problem.
- This dual might be infeasible $\implies$ Reduce $P$.
Algorithm (once again)

- Set an initial task partition
- Construct the ARMP
  - Solve the ARMP
    - Primal solution & Aggregated dual values
    - Disaggregate dual values
    - Solve the subproblems
    - Change the partition? No
- Update partition (disaggregate or aggregate)
A task partition is built using a subset $C$ of columns

For the initial partition, $C$ is composed of columns (feasible or not) obtained from
- A heuristic solution
- Logical reasoning

Disaggregate the partition when there exist incompatible variables with large negative reduced costs (compared to the reduced costs of the compatible variables)

Aggregate the partition when degeneracy becomes important
When **disaggregating**
- The current subset $C$ is augmented by a small number of incompatible columns that have negative reduced costs

When **aggregating**
- The current subset $C$ is replaced by the set of variables with a positive value in the current master problem solution
Multi-phase DCA: Concepts

Introduced in


- In practice, incompatible variables often price out favorably
  - May yield fast disaggregation
- Partial pricing strategy that favors slow disaggregation
- With each column, associate a number of incompatibilities
  - Approximation of the number of additional clusters needed to become compatible
- In phase $k$: price only variables with $k$ incompatibilities or less
Multi-phase DCA: Concepts

Introduced in

- In practice, incompatible variables often price out favorably
  - May yield fast disaggregation
- Partial pricing strategy that favors slow disaggregation
- With each column, associate a number of incompatibilities
  - Approximation of the number of additional clusters needed to become compatible
- In phase $k$: price only variables with $k$ incompatibilities or less
A resource is used to limit the number of incompatibilities in a path.
MPDCA algorithm

1. Create an initial partition \( Q \), set phase \( k = 0 \)
2. Solve the ARMP
3. Compute disaggregated dual variables
4. Price the \( p \)-incompatible variables \( (p < k+1) \)
5. Check if there is a negative reduced cost?

   - No, stop if the last one (optimal solution)
   - Yes

6. Change the partition?
   - No
   - Yes

7. Update the partition \( Q \)

- Initial task partition computed from a tabu search solution
- Heuristic branching (column fixing)
### Solution process statistics

<table>
<thead>
<tr>
<th>Instance</th>
<th>BP heuristic</th>
<th>DCA heuristic</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Times (s)</td>
<td>Numbers of</td>
</tr>
<tr>
<td></td>
<td>Total</td>
<td>RMP</td>
</tr>
<tr>
<td>1187/228</td>
<td>3852</td>
<td>2503</td>
</tr>
<tr>
<td>1507/289</td>
<td>9230</td>
<td>6715</td>
</tr>
<tr>
<td>2165/416</td>
<td>43625</td>
<td>36174</td>
</tr>
<tr>
<td>2924/564</td>
<td>95215</td>
<td>86228</td>
</tr>
</tbody>
</table>

### Computational results

- **BP heuristic**
  - Instance 1187/228: 2607 iterations, 228 nodes explored, 1188 fractional variables
  - Instance 1507/289: 3260 iterations, 287 nodes explored, 1488 fractional variables
  - Instance 2165/416: 5131 iterations, 416 nodes explored, 2154 fractional variables
  - Instance 2924/564: 6408 iterations, 563 nodes explored, 2914 fractional variables

- **DCA heuristic**
  - Instance 1187/228: 861 iterations, 156 nodes explored, 344 fractional variables
  - Instance 1507/289: 1157 iterations, 191 nodes explored, 374 fractional variables
  - Instance 2165/416: 1896 iterations, 317 nodes explored, 617 fractional variables
  - Instance 2924/564: 3149 iterations, 440 nodes explored, 608 fractional variables
### Solution quality statistics

<table>
<thead>
<tr>
<th>Instance</th>
<th>Credited hours</th>
<th>Days off</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Mean</td>
<td>Min</td>
</tr>
<tr>
<td><strong>BP heuristic</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1187/228</td>
<td>75.2</td>
<td>71.1</td>
</tr>
<tr>
<td>1507/289</td>
<td>75.2</td>
<td>71.2</td>
</tr>
<tr>
<td>2165/416</td>
<td>75.2</td>
<td>71.0</td>
</tr>
<tr>
<td>2924/564</td>
<td>75.1</td>
<td>71.0</td>
</tr>
<tr>
<td><strong>DCA heuristic</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1187/228</td>
<td>75.2</td>
<td>73.0</td>
</tr>
<tr>
<td>1507/289</td>
<td>75.2</td>
<td>72.6</td>
</tr>
<tr>
<td>2165/416</td>
<td>75.2</td>
<td>73.0</td>
</tr>
<tr>
<td>2924/564</td>
<td>75.1</td>
<td>71.9</td>
</tr>
</tbody>
</table>

Average value of fixed variables: **0.91 with DCA, 0.71 with BP**
Outline

1. Airline planning process
2. Crew pairing and column generation
3. Crew assignment and dynamic constraint aggregation
4. Integrated crew scheduling and bi-dynamic constraint aggregation
5. Conclusions and future work
Traditional sequential approach

Weaknesses

- Pairings are built without fully taking into account schedule feasibility rules
- Less choices to construct schedules
Integrated approach

Strengths

- Schedules are built directly from the flights (more choices)
- Schedule feasibility rules are considered throughout the process
- Crew assignment is not subject to a separation by crew base
Only one paper on integrated crew scheduling

- Zeghal and Minoux (2006)
  - Exact and heuristic branch-and-bound methods
  - Assume that
    - All duties can be enumerated a priori
    - Deadheads can be introduced as needed without additional costs
    - Small-sized instances (up to 210 flights or 40 crew members)
Model for integrated problem (bidline, pilots)

\[ \text{Minimize} \quad \sum_{b \in B} \sum_{s \in S^b} c_s x_s + \beta \sum_{b \in B} y_b \]

\[ \text{s.t.} \quad \sum_{b \in B} \sum_{s \in S^b} a_{fs} x_s = 1, \quad \forall f \in F \]

\[ \sum_{s \in S^b} x_s - y_b \leq q_b, \quad \forall b \in B \]

\[ x_s \in \{0, 1\}, \quad \forall b \in B, \ s \in S^b \]

\[ y_b \geq 0, \quad \forall b \in B \]

- \( F \): set of flights to cover;
- \( B \): set of crew bases;
- \( q_b \): number of pilots available at base \( b \);
- \( \beta \): penalty cost for each additional pilot;
- \( S^b \): set of feasible schedules for pilots at base \( b \);
- \( c_s \): cost of schedule \( s \);
- \( a_{fs} \): binary parameter equal to 1 if flight \( f \) is actively covered in schedule \( s \) and 0 otherwise;
- \( x_s \): binary variable taking value 1 if schedule \( s \) is selected and 0 otherwise;
- \( y_b \): surplus variable indicating the number of extra pilots required at the base \( b \).
## Integrated problem: Complexity?

### Difficulties

1. Large number of set partitioning constraints
2. Large number of tasks per path yielding high degeneracy

<table>
<thead>
<tr>
<th>Size</th>
<th>Pairing (weekly)</th>
<th>Assignment</th>
<th>Integrated</th>
</tr>
</thead>
<tbody>
<tr>
<td>200 flights/day</td>
<td>1400</td>
<td>600</td>
<td>6000</td>
</tr>
<tr>
<td>1000 flights/day</td>
<td>7000</td>
<td>3000</td>
<td>30000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Problem</th>
<th>Tasks</th>
<th>Paths</th>
<th>Tasks/path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pairing (weekly)</td>
<td>flights</td>
<td>pairings</td>
<td>~10</td>
</tr>
<tr>
<td>Assignment</td>
<td>pairings</td>
<td>schedules</td>
<td>~5</td>
</tr>
<tr>
<td>Integrated</td>
<td>flights</td>
<td>schedules</td>
<td>~50</td>
</tr>
</tbody>
</table>
Column generation: subproblems

- One-month horizon
- One subproblem per crew base
- Between 9 and 18 resources
Bi-dynamic constraint aggregation

Introduced in

- With MPDCA, most of the computational time is spent solving the subproblems.
- To avoid fast disaggregation, forbid the pricing of columns that would force the disaggregation of certain clusters.
- This is another partial pricing strategy.
Bi-dynamic constraint aggregation

Introduced in


- With MPDCA, most of the computational time is spent solving the subproblems
- To *avoid fast disaggregation*, forbid the pricing of columns that would force the disaggregation of certain clusters
- This is another *partial pricing strategy*
Main ideas

- Reduce the subproblem networks according to current task partition
  - Select a certain number of task clusters
  - Remove all incompatible arcs associated with these clusters
- If no negative reduced cost columns, solve with complete networks
Cluster selection based on

- Aggregated dual values of the clusters
- Cluster reduced costs
- Neighborhoods to increase interaction between generated columns
Cluster selection without neighborhoods

Legend:
- cluster to disaggregate
- aggregated cluster
- flight
- incompatible arc
Cluster selection based on neighborhoods

Legend:
- cluster to disaggregate
- aggregated cluster
- flight
- incompatible arc
Algorithm 1 : BDCA with neighborhoods (BDCA-N)

1: Select an initial neighborhood $N$
2: Set $k := 0$, $Z_{old} := \infty$, $Z := \infty$
3: Create an initial partition $Q$ and build the ARMP
4: Solve the ARMP with artificial variables
5: repeat
6: Select a subset of clusters $U$ according to $RL$ and $N$
7: $\text{negRedCostCols} := \text{LocalSearch}(Z, Q, k, U)$
8: if $\text{negRedCostCols} = \text{false}$ then
9: $k := k + 1$
10: else if $Z_{old} - Z \leq \delta$ then
11: Select a new neighborhood $N$
12: $Z_{old} := Z$
13: until $k > k_{max}$
Algorithm 2 : \textit{LocalSearch}(Z, Q, k, U)

1: \textit{curIter} := 1
2: repeat
3: \text{Compute disaggregated dual values}
4: \textit{redNetwork} := true
5: Reduce the subproblems according to \textit{U} and solve them in phase \textit{k}
6: if \text{negative reduced cost columns found} then
7: \text{Update partition \textit{Q} (if needed) and the ARMP}
8: \text{Solve the ARMP and update \textit{Z}}
9: \textit{curIter} := \textit{curIter} + 1
10: else if \textit{redNetwork} = true and \textit{k} > 0 then
11: \textit{redNetwork} := false
12: \text{Solve the complete subproblems in phase \textit{k} and go to Step 6}
13: else
14: return \textit{false}
15: until \textit{curIter} > \text{maxIter}_k
16: return \textit{true}
Computational results


- Initial task partition from the crew pairing solution
- Phases 0 and 1 only
- Column generation stops if objective value decreases by less than 0.01% in 10 iterations
- BDCA-N-1 version forces phase 1 if premature halt in phase 0
Computational results


- Initial task partition from the crew pairing solution
- Phases 0 and 1 only
- Column generation stops if objective value decreases by less than 0.01% in 10 iterations
- BDCA-N-1 version forces phase 1 if premature halt in phase 0
### Instance characteristics

<table>
<thead>
<tr>
<th>Instance</th>
<th>Number of flights</th>
<th>Number of bases</th>
<th>Number of stations</th>
</tr>
</thead>
<tbody>
<tr>
<td>/1</td>
<td>1011</td>
<td>3</td>
<td>26</td>
</tr>
<tr>
<td>/2</td>
<td>1463</td>
<td>3</td>
<td>35</td>
</tr>
<tr>
<td>/3</td>
<td>1793</td>
<td>3</td>
<td>41</td>
</tr>
<tr>
<td>/4</td>
<td>5466</td>
<td>3</td>
<td>49</td>
</tr>
<tr>
<td>/5</td>
<td>5639</td>
<td>3</td>
<td>34</td>
</tr>
<tr>
<td>/6</td>
<td>5755</td>
<td>3</td>
<td>52</td>
</tr>
<tr>
<td>/7</td>
<td>7527</td>
<td>3</td>
<td>54</td>
</tr>
</tbody>
</table>

**Remark**
- For instance /1, standard column generation took more than 12 hours of CPU time (sequential approach took 4 minutes).
- For all other instances, computation was stopped after 2 days.
Instance characteristics

<table>
<thead>
<tr>
<th>Instance</th>
<th>Number of flights</th>
<th>Number of bases</th>
<th>Number of stations</th>
</tr>
</thead>
<tbody>
<tr>
<td>/1</td>
<td>1011</td>
<td>3</td>
<td>26</td>
</tr>
<tr>
<td>/2</td>
<td>1463</td>
<td>3</td>
<td>35</td>
</tr>
<tr>
<td>/3</td>
<td>1793</td>
<td>3</td>
<td>41</td>
</tr>
<tr>
<td>/4</td>
<td>5466</td>
<td>3</td>
<td>49</td>
</tr>
<tr>
<td>/5</td>
<td>5639</td>
<td>3</td>
<td>34</td>
</tr>
<tr>
<td>/6</td>
<td>5755</td>
<td>3</td>
<td>52</td>
</tr>
<tr>
<td>/7</td>
<td>7527</td>
<td>3</td>
<td>54</td>
</tr>
</tbody>
</table>

Remark

- For instance /1, standard column generation took more than 12 hours of CPU time (sequential approach took 4 minutes).
- For all other instances, computation was stopped after 2 days.
## Comparative results

**SEQ: sequential approach (crew pairing, then crew assignment)**

<table>
<thead>
<tr>
<th>Instance</th>
<th>BDCA-N approach</th>
<th></th>
<th>BDCA-N-1 approach</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>CPU ratio</td>
<td>Savings (%)</td>
<td>CPU ratio</td>
<td>Savings (%)</td>
</tr>
<tr>
<td></td>
<td>BDCA-N/SEQ</td>
<td>Cost</td>
<td>BDCA-N-1/SEQ</td>
<td>Cost</td>
</tr>
<tr>
<td>I1</td>
<td>1.7</td>
<td>5.74</td>
<td>1.9</td>
<td>5.72</td>
</tr>
<tr>
<td>I2</td>
<td>2.5</td>
<td>3.60</td>
<td>3.2</td>
<td>3.28</td>
</tr>
<tr>
<td>I3</td>
<td>3.0</td>
<td>3.07</td>
<td>5.3</td>
<td>4.46</td>
</tr>
<tr>
<td>I4</td>
<td>1.8</td>
<td>3.42</td>
<td>2.1</td>
<td>4.02</td>
</tr>
<tr>
<td>I5</td>
<td>6.0</td>
<td>4.09</td>
<td>6.6</td>
<td>4.97</td>
</tr>
<tr>
<td>I6</td>
<td>3.0</td>
<td>6.75</td>
<td>4.2</td>
<td>8.78</td>
</tr>
<tr>
<td>I7</td>
<td>3.0</td>
<td>1.50</td>
<td>3.2</td>
<td>2.14</td>
</tr>
<tr>
<td><strong>Average</strong></td>
<td><strong>3.0</strong></td>
<td><strong>4.02</strong></td>
<td><strong>3.8</strong></td>
<td><strong>4.76</strong></td>
</tr>
</tbody>
</table>

CPU ratio indicates the ratio of the CPU time of the approach to the CPU time of the sequential approach. Savings (%) are the percentage reduction in cost compared to the sequential approach.
### Average results comparison

<table>
<thead>
<tr>
<th>Instance</th>
<th>CPU ratio (BDCA-N/SEQ)</th>
<th>Savings (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MPDCA</td>
<td>6.8</td>
<td>3.37</td>
</tr>
<tr>
<td>BDCAn</td>
<td>2.6</td>
<td>2.63</td>
</tr>
<tr>
<td>BDCA-N</td>
<td>3.0</td>
<td>4.02</td>
</tr>
<tr>
<td>BDCA-N-1</td>
<td>3.8</td>
<td>4.76</td>
</tr>
</tbody>
</table>
Outline

1. Airline planning process
2. Crew pairing and column generation
3. Crew assignment and dynamic constraint aggregation
4. Integrated crew scheduling and bi-dynamic constraint aggregation
5. Conclusions and future work
Conclusion

- For several decades, airline crew scheduling has been a fertile ground for operations research.
- Operations research tools have helped reducing crew costs and increasing crew member quality of life.
- Research on this topic remains very active because there still remain many challenges.
Future work

- Improve dynamic constraint aggregation
  - Dual variable disaggregation
  - Neighborhood exploration
- Experiment IPS (improved primal simplex) method on certain crew scheduling problems
  - IPS is a generalization of DCA to generic linear programs
  - It uses a reduced problem and a complementary problem to identify good incompatible variables
- Integrated crew scheduling in a rostering context (personalized schedules)
  - Pilots only
  - Pilots and copilots simultaneously
Thank you!