A Recovery Algorithm for a Disrupted Airline Schedule

Niklaus Eggenberg
Matteo Salani and Prof. M. Bierlaire

In collaboration with APM Technologies
Index

- Airline Scheduling in general
- The Disrupted Schedule Recovery Problem (DSRP)
- The Column Generation (CG) approach
- The Recovery Network Model (RNM)
- The pricing algorithm
- Future Work and Conclusions
Airline Scheduling Approach

Route Choice

- Fleet Assignment
- Tail Assignment
- Crew Pairing
- Crew Roistering
- Passenger Routing (Catering)
Disrupted Schedule Recovery

\[ t_0 + T \]

Schedule \( S_0 \)

Disruption

Recovery Decision
Definitions

• *Disruption*
  
event making a schedule unrealizable

• *Recovery*
  
action to get back to initial schedule

• *Recovery Period (T)*
  
time needed to recover initial schedule
Definitions

• *Recovery Plan*
  
  set of actions to recover disrupted schedule

• *Recovery Scheme (r)*
  
  set of actions for a resource (plane)
Hypothesis

• consider only fleet and tail assignment
• no repositioning flights
• no early departure for flights
• work with universal time (UMT)
• initial state of resources are known
• no irregularity until end of recovery period
• maintenance forced by resource consumption
Column Generation

• column = recovery scheme

• recovery scheme \( r \) = way to link Initial State to Final State with succession of flights and maintenances

• suppose set of all possible schemes \( R \) known

• find optimal combination of schemes
Master Problem (IMP)

\[
\begin{align*}
\min \quad & z_{MP} = \sum_{r \in R} c_r x_r + \sum_{f \in F} c_f y_f \\
\text{s. t.} \quad & \sum_{r \in R} b^f_r x_r + y_f = 1 \quad \forall f \in F \\
& \sum_{r \in R} b^{s^a}_r x_r = 1 \quad \forall s^a \in S^a, \forall a \in A \\
& \sum_{r \in R} b^p_r x_r \leq 1 \quad \forall p \in P \\
x_r \in \{0,1\} \quad \forall r \in R \\
y_f \in \{0,1\} \quad \forall f \in F
\end{align*}
\]
Solving the Master Problem

I. Solve IMP with **Branch and Bound**

II. Solve linear relaxation LP at each node:
   - Restrict LP to sub-set $R' \subseteq R$
   - Solve RLP
   - Find $b_r \in R \setminus R'$ minimizing reduced cost
   - Insert column if $r.c. < 0$ and resolve RLP
The Pricing Problem

Find column $b_r \in R \setminus R'$ minimizing reduced cost $\tilde{c}_r^p$

$$\min_{r \in R} \tilde{c}_r^p = c_p^r - \sum_{f \in F} b_r^f \lambda_f - \sum_{a \in A} \sum_{s^a \in S^a} b_r^{s^a} \eta_{s^a} + b_r^p \mu_p \quad \forall p \in P$$

Recovery Network Model

Solve Resource Constrained Elementary Shortest Path Problem (RCESPP)
The Recovery Network (Argüello et al. 97)

• Time-space network
• One network for every plane
• Source node corresponding to initial state
• Sinks corresponding to expected final states
• 3 arc types (NEVER horizontal):
  1. Flight arcs
  2. Maintenance arcs
  3. Termination arcs (vertical)
Avoiding Vertical Arcs

RN – Vertical Elimination
Source and Sink Nodes

Plane P1, initial state = [GVA, 0800]
Expected States : [GVA, 1800] and [BUD, 1500]
Flight and Maintenance Arcs

flight F1: GVA to NY at 1200
Arc Costs

- Flight arcs: \[ c = c^f - \lambda_f \]
- Maintenance arcs: \[ c = c^f + c^M - \lambda_f \]
- Termination arcs: \[ c = -\eta_s a \]
Recovery Network Properties

- No horizontal arcs
- No vertical arcs except termination arcs
- Node only at earliest availability time
- Grounding time included in arc length (3 types)
- Maintenances are integrated before flight if possible
Solving the Pricing Problem

• Find RCESP in Recovery Networks

• 3 phase algorithm:
  1. RN Generation
  2. RN Preprocessing
  3. Solve RCESPP
1. RN Generation

- proceed *dynamically*: create source, proceed by increasing time
- for all *feasible* flights create arc and node
- process nodes until all nodes are explored
1. RN Generation
Feasible Flights

- departure airport must match
- departure and landing in airports’ activity slots
- control parameters
Control Parameters

• maximum delay bound
• time discretisation (keep earliest availability time)
• minimal time before reaching sinks
• maximal waiting time (D. Messina)
Network Cleaning

• proceed *forwards*, remove nodes *without predecessors* (!source!)

• proceed *backwards*, remove nodes *without successors* (!sinks!)
Tested Instance

- 48 flights
- 9 airports
- 5 planes
- $T = 2880$ minutes
## Delay Bound Parameter DB

<table>
<thead>
<tr>
<th>DB [min]</th>
<th>2880 clean</th>
<th>1440 clean</th>
<th>720 clean</th>
<th>360 clean</th>
<th>180 clean</th>
<th>90 clean</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>2117 1939</td>
<td>1995 1834</td>
<td>1611 1443</td>
<td>965 842</td>
<td>473 412</td>
<td>306 255</td>
</tr>
<tr>
<td># arcs</td>
<td>17825 16791</td>
<td>14851 14301</td>
<td>10836 10366</td>
<td>6773 6507</td>
<td>3348 3247</td>
<td>1986 1916</td>
</tr>
<tr>
<td>flight</td>
<td>11775 11258</td>
<td>9780 9518</td>
<td>7069 6843</td>
<td>4278 4158</td>
<td>2056 2009</td>
<td>1205 1181</td>
</tr>
<tr>
<td>maint</td>
<td>6050 5533</td>
<td>5071 4783</td>
<td>3767 3523</td>
<td>2495 2349</td>
<td>1292 1238</td>
<td>781 735</td>
</tr>
<tr>
<td># sinks</td>
<td>5 5</td>
<td>5 5</td>
<td>5 5</td>
<td>5 5</td>
<td>5 5</td>
<td>5 5</td>
</tr>
<tr>
<td># term arcs</td>
<td>2780 2780</td>
<td>2780 2780</td>
<td>2610 2610</td>
<td>2120 2120</td>
<td>1335 1335</td>
<td>840 840</td>
</tr>
<tr>
<td>time [s]</td>
<td>3.53 1.63</td>
<td>3 0.65</td>
<td>2.24 0.25</td>
<td>1.35 0.11</td>
<td>0.63 0.01</td>
<td>0.39 0.01</td>
</tr>
</tbody>
</table>
PP 1 – Generation – Display
# Time Discretisation Parameter $\Delta$

<table>
<thead>
<tr>
<th>$\Delta$ [min]</th>
<th>5 clean</th>
<th>15 clean</th>
<th>30 clean</th>
<th>60 clean</th>
<th>120 clean</th>
<th>240 clean</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>2117</td>
<td>1939</td>
<td>781</td>
<td>721</td>
<td>416</td>
<td>386</td>
</tr>
<tr>
<td># arcs</td>
<td>17825</td>
<td>16791</td>
<td>6776</td>
<td>6430</td>
<td>3673</td>
<td>3503</td>
</tr>
<tr>
<td>flight</td>
<td>11775</td>
<td>11258</td>
<td>4455</td>
<td>4282</td>
<td>2411</td>
<td>2326</td>
</tr>
<tr>
<td>maint</td>
<td>6050</td>
<td>5533</td>
<td>2321</td>
<td>2148</td>
<td>1262</td>
<td>1177</td>
</tr>
<tr>
<td># sinks</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td># term arcs</td>
<td>2780</td>
<td>2780</td>
<td>1055</td>
<td>1055</td>
<td>570</td>
<td>570</td>
</tr>
<tr>
<td>time [s]</td>
<td>3.61</td>
<td>1.63</td>
<td>1.26</td>
<td>0.08</td>
<td>0.68</td>
<td>0.02</td>
</tr>
</tbody>
</table>
### Min Time before Sink Absorption $\Phi$

<table>
<thead>
<tr>
<th>$\Phi$ [min]</th>
<th>0</th>
<th>0 clean</th>
<th>1440</th>
<th>clean</th>
<th>2160</th>
<th>clean</th>
<th>2520</th>
<th>clean</th>
<th>2700</th>
<th>clean</th>
<th>2790</th>
<th>clean</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong># nodes</strong></td>
<td>2117</td>
<td>1939</td>
<td>2117</td>
<td>1939</td>
<td>2117</td>
<td>1939</td>
<td>2117</td>
<td>1758</td>
<td>2117</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong># arcs</strong></td>
<td>17825</td>
<td>16791</td>
<td>17825</td>
<td>16791</td>
<td>17825</td>
<td>16791</td>
<td>17825</td>
<td>14670</td>
<td>17825</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>flight</strong></td>
<td>11775</td>
<td>11258</td>
<td>11775</td>
<td>11258</td>
<td>11775</td>
<td>11258</td>
<td>11775</td>
<td>9797</td>
<td>11775</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>maint</strong></td>
<td>6050</td>
<td>5533</td>
<td>6050</td>
<td>5533</td>
<td>6050</td>
<td>5533</td>
<td>6050</td>
<td>4873</td>
<td>6050</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong># sinks</strong></td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong># term arcs</strong></td>
<td>2780</td>
<td>2780</td>
<td>1630</td>
<td>1630</td>
<td>1050</td>
<td>1050</td>
<td>330</td>
<td>330</td>
<td>25</td>
<td>25</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td><strong>time [s]</strong></td>
<td>3.6</td>
<td>1.66</td>
<td>3.62</td>
<td>1.63</td>
<td>3.56</td>
<td>1.62</td>
<td>3.59</td>
<td>1.62</td>
<td>3.57</td>
<td>4.03</td>
<td>3.61</td>
<td>0.07</td>
</tr>
</tbody>
</table>
2. RN Preprocessing

- compute upper/lower **bounds** on costs and resource consumption (spp)
- remove unfeasible arcs according to resources
Example of Arc Removing

max flown hours between maintenances = 60 h

min flown hours = 55 h

flt = 3 h

flt = 6 h

Computed LB
## Without Maintenance Slot

<table>
<thead>
<tr>
<th>% resource used initially</th>
<th>0</th>
<th>clean</th>
<th>25</th>
<th>clean</th>
<th>50</th>
<th>clean</th>
<th>75</th>
<th>clean</th>
<th>100</th>
<th>clean</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>669</td>
<td>386</td>
<td>304</td>
<td>115</td>
<td>70</td>
<td>58</td>
<td>22</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># arcs</td>
<td>2586</td>
<td>1949</td>
<td>982</td>
<td>606</td>
<td>165</td>
<td>143</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>flight</td>
<td>2586</td>
<td>1949</td>
<td>982</td>
<td>606</td>
<td>165</td>
<td>143</td>
<td>21</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>maint</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td># sinks</td>
<td>5</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td># term arcs</td>
<td>270</td>
<td>270</td>
<td>104</td>
<td>104</td>
<td>39</td>
<td>39</td>
<td>2</td>
<td>2</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>time [s]</td>
<td>0.66</td>
<td>0.047</td>
<td>0.22</td>
<td>0.03</td>
<td>0.05</td>
<td>&lt; 0.01</td>
<td>&lt; 0.01</td>
<td>&lt; 0.01</td>
<td>&lt; 0.01</td>
<td>&lt; 0.01</td>
</tr>
</tbody>
</table>
With Maintenance at any Time

<table>
<thead>
<tr>
<th>% resource used initially</th>
<th>0</th>
<th>clean</th>
<th>25</th>
<th>clean</th>
<th>50</th>
<th>clean</th>
<th>75</th>
<th>clean</th>
<th>100</th>
<th>clean</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>2052</td>
<td>1822</td>
<td>2052</td>
<td>1822</td>
<td>2052</td>
<td>1822</td>
<td>2052</td>
<td>1822</td>
<td>2052</td>
<td>1822</td>
</tr>
<tr>
<td># arcs</td>
<td>16857</td>
<td>15749</td>
<td>16857</td>
<td>15749</td>
<td>16854</td>
<td>15746</td>
<td>16833</td>
<td>15725</td>
<td></td>
<td></td>
</tr>
<tr>
<td>flight</td>
<td>11047</td>
<td>10456</td>
<td>11047</td>
<td>10456</td>
<td>11044</td>
<td>10453</td>
<td>11023</td>
<td>10432</td>
<td></td>
<td></td>
</tr>
<tr>
<td>maint</td>
<td>5810</td>
<td>5293</td>
<td>5810</td>
<td>5293</td>
<td>5810</td>
<td>5293</td>
<td>5810</td>
<td>5293</td>
<td></td>
<td></td>
</tr>
<tr>
<td># sinks</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td># term arcs</td>
<td>2102</td>
<td>2102</td>
<td>2101</td>
<td>2101</td>
<td>2100</td>
<td>2100</td>
<td>2099</td>
<td>2099</td>
<td>2098</td>
<td>2098</td>
</tr>
<tr>
<td>time [s]</td>
<td>3.22</td>
<td>0.234</td>
<td>3.33</td>
<td>0.265</td>
<td>3.36</td>
<td>0.219</td>
<td>3.56</td>
<td>0.25</td>
<td>3.33</td>
<td>0.235</td>
</tr>
</tbody>
</table>
## With Maintenance Slot at Night

<table>
<thead>
<tr>
<th>% resource used initially</th>
<th>0 clean</th>
<th>25 clean</th>
<th>50 clean</th>
<th>75 clean</th>
<th>100 clean</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>737</td>
<td>435</td>
<td>655</td>
<td>338</td>
<td>589</td>
</tr>
<tr>
<td># arcs</td>
<td>4326</td>
<td>3638</td>
<td>2968</td>
<td>2263</td>
<td>2527</td>
</tr>
<tr>
<td>flight</td>
<td>2958</td>
<td>2270</td>
<td>2488</td>
<td>1783</td>
<td>2143</td>
</tr>
<tr>
<td>maint</td>
<td>1368</td>
<td>1368</td>
<td>480</td>
<td>480</td>
<td>384</td>
</tr>
<tr>
<td># sinks</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td># term arcs</td>
<td>572</td>
<td>572</td>
<td>327</td>
<td>327</td>
<td>287</td>
</tr>
<tr>
<td>time [s]</td>
<td>0.84</td>
<td>0.06</td>
<td>0.55</td>
<td>0.08</td>
<td>0.5</td>
</tr>
</tbody>
</table>
PP 2 – Preprocessing – Night maintenance with adapted parameters

DB = 720, Δ = 15, Φ = 1440

<table>
<thead>
<tr>
<th>% resource used initially</th>
<th>0 clean</th>
<th>25 clean</th>
<th>50 clean</th>
<th>75 clean</th>
<th>100 clean</th>
</tr>
</thead>
<tbody>
<tr>
<td># nodes</td>
<td>270</td>
<td>222</td>
<td>176</td>
<td>162</td>
<td>151</td>
</tr>
<tr>
<td># arcs</td>
<td>1515</td>
<td>957</td>
<td>675</td>
<td>429</td>
<td>408</td>
</tr>
<tr>
<td>flight</td>
<td>1035</td>
<td>732</td>
<td>480</td>
<td>414</td>
<td>393</td>
</tr>
<tr>
<td>maint</td>
<td>480</td>
<td>225</td>
<td>195</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td># sinks</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td># term arcs</td>
<td>70</td>
<td>69</td>
<td>69</td>
<td>69</td>
<td>69</td>
</tr>
<tr>
<td>time [s]</td>
<td>0.39</td>
<td>0.19</td>
<td>&lt; 0.01</td>
<td>0.14</td>
<td>0.01</td>
</tr>
</tbody>
</table>

*Note: Values indicate resource usage and computational time for different scenarios.*
3. Solve RCESPP

Use a dynamic programming algorithm proposed by Righini & Salani (2006), which is an extension of Desrochers et al. (1988).

Exploit bounds computed in preprocessing.
Future Work

• Test real instances (next month)

• Explore more widely RCESPP Solver

• Compare solution to real recovery decisions

• Include Algorithm in APM Framework
Conclusions

- Adapted model including maintenances
- Developed 3 phase algorithm to solve PP
- Get quick and easy to parameterize solutions
- Still need real-instance validation
THANKS for your attention

Feel free to ask!