Dynamic Demand-Capacity Balancing Using Constraint-Based Local Search

Farshid Hassani Bijarbooneh
Pierre Flener Justin Pearson

ASTRA Research Group
Department of Information Technology
Uppsala University, Sweden

www.it.uu.se/research/group/astra

8th Innovative Research Workshop & Exhibition
Brétigny-sur-Orge (France), 3 December 2009

Outline

Outline

UPPSALA UNIVERSITET

Objective & Motivation

- Tactical planning, in quasi-real-time,
- for the entire European airspace,

UPPSALA UNIVERSITET

Objective & Motivation

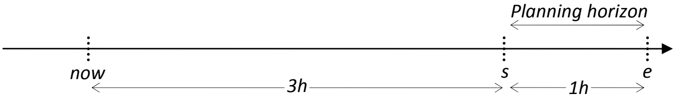
- Tactical planning, in quasi-real-time,
- for the entire European airspace,
- necessary for future flight volumes:

Objective & Motivation

- Tactical planning, in quasi-real-time,
- for the entire European airspace,
- necessary for future flight volumes:
- minimise the total ground-holding
  (e.g., by maximum 120 minutes per flight),

Objective & Motivation

- Tactical planning, in quasi-real-time,
- for the entire European airspace,
- necessary for future flight volumes:
- minimise the total ground-holding
  (e.g., by maximum 120 minutes per flight),
- such that all the capacity constraints are satisfied,
- within a rolling horizon (of, e.g., one hour)
  that starts, e.g., three hours from now:



- Ideally: also balance demands on portions of airspace.

UPPSALA UNIVERSITET

European ATM at Present

- Flight planning is done globally,
  at the strategic and tactical levels:
  - at the Central Flow Management Unit (CFMU),
  - but without achieving optimal global flow, and under almost certainly incorrect data estimates.

UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

Conclusion

European ATM at Present

- Flight planning is done globally, at the strategic and tactical levels:
  - at the Central Flow Management Unit (CFMU),
  - but without achieving optimal global flow, and under almost certainly incorrect data estimates.
- Flight control is done locally, at the operational level:
  - at regional air-traffic control centres (ATCCs),
  - but without a global view when re-planning flights, even though much more precise data are available.

UPPSALA UNIVERSITET

Objective & Motivation

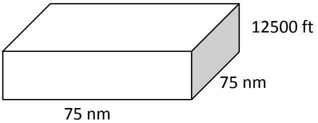
Model

Experiments

Conclusion

European ATM in the Future?

- Year 2030: 50,000 flights/day (now: 30,000 flights/day)
- The airspace might be partitioned into a 3D-grid of same-sized box-shaped cells (as building blocks for a new sectorisation), e.g., 75 nm $\times$ 75 nm $\times$ 12500 ft:
  - 4 layers
  - 4,600 cells
  - 700,000 cell entries per day



UPPSALA UNIVERSITET

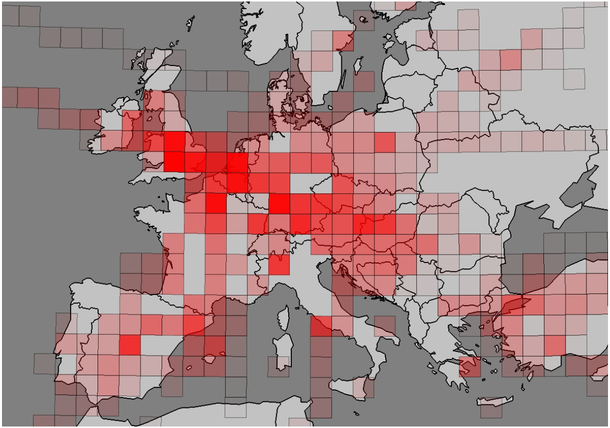
Objective & Motivation

Model

Experiments

Conclusion

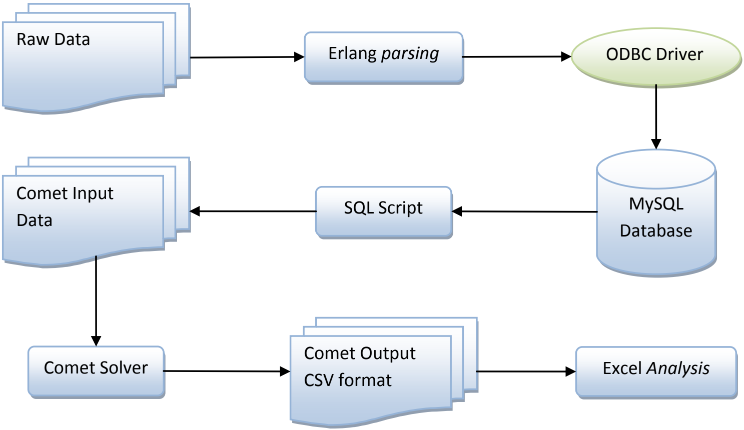
3D Cells over Europe



European airspace divided into cells

Outline

Dataflow

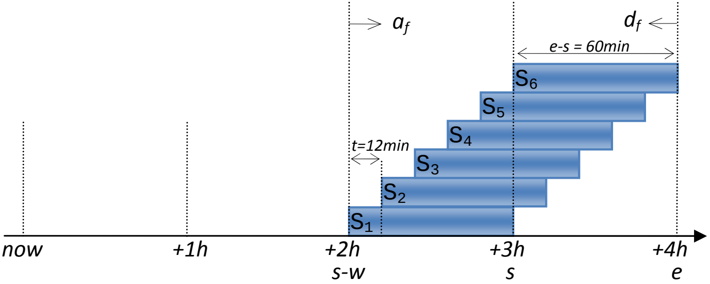

Raw Data
Erlang *parsing*
ODBC Driver
MySQL Database
SQL Script
Comet Input Data
Comet Solver
Comet Output CSV format
Excel *Analysis*

Sliding Windows

- Capacity = max number of entering flights per hour.
- Constraints: at any given moment,
  no capacity is exceeded within the last hour.

Sliding Windows

- Capacity = max number of entering flights per hour.
- Constraints: at any given moment, no capacity is exceeded within the last hour.
- In practice, we sample every t minutes, e.g., $t = 12$:



UPPSALA UNIVERSITET

Variables, Constraints, and Objective

- Decision variables:
  for each non-airborne flight:
  a ground-holding delay within $0 \ldots 120$ minutes.

UPPSALA UNIVERSITET

Variables, Constraints, and Objective

- Decision variables:
  for each non-airborne flight:
  a ground-holding delay within $0 \ldots 120$ minutes.
- Constraints:
  for each cell and each sliding window:
  # airborne flights $+$ # re-planned flights $\leq$ cell capacity.

Variables, Constraints, and Objective

- Decision variables:
  for each non-airborne flight:
  a ground-holding delay within $0 \ldots 120$ minutes.
- Constraints:
  for each cell and each sliding window:
  # airborne flights + # re-planned flights $\leq$ cell capacity.
- Objective function, to be minimised:

  $\alpha \cdot$ sum(all delays) $+ \beta \cdot$ violations(all constraints)

  where α and β are weights.

Typical Problem Instance

Horizon 9 to 10 pm, for a projected data-set of year 2030:

- Decision variables: 4,295 (= # non-airborne flights)
- Relevant cells: 2,294
- Cell entries: 51,879
- Constraints: 1,936

UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

Conclusion

Three-State Heuristic

- State 1, initially:
  - Select a **delay** based on a probability function.
  - Select a **flight** (that achieves the largest decrease in violations) for the selected delay.

UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

Conclusion

Three-State Heuristic

- State 1, initially:
  - Select a **delay** based on a probability function.
  - Select a **flight** (that achieves the largest decrease in violations) for the selected delay.
- State 2, when violations drop below a threshold:
  - Select the most violating **flight**.
  - Select a **delay** (that achieves the largest decrease in violations) for the selected flight.

UPPSALA UNIVERSITET

Three-State Heuristic

- State 1, initially:
  - Select a **delay** based on a probability function.
  - Select a **flight** (that achieves the largest decrease in violations) for the selected delay.
- State 2, when violations drop below a threshold:
  - Select the most violating **flight**.
  - Select a **delay** (that achieves the largest decrease in violations) for the selected flight.
- State 3, when violations drop below a lower threshold (very close to satisfaction):
  - Select a (**flight**, **delay**) pair (that achieves the largest decrease in violations).

UPPSALA UNIVERSITET

Objective & Motivation

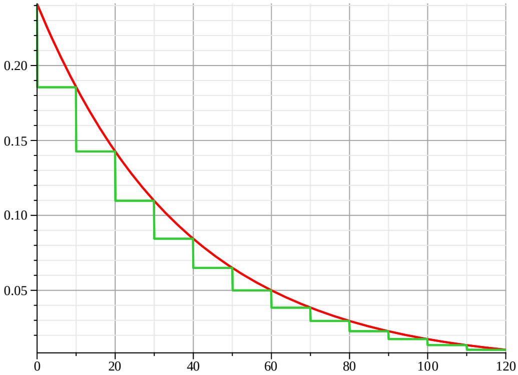
Model

Experiments

Conclusion

State 1 of Heuristic

Select a delay based on a probability function:



Longer delays are less likely than shorter delays.

Outline

Platform

- Constraint solver: local-search back-end of Comet (available at www.dynadec.com), which offers a very-high-level language for modelling the constraints and programming search (meta-)heuristics.
- Operating system: Linux Ubuntu 9.04 (32-bit).
- CPU: Intel Core 2 Duo T7300 2GHz, 2MB cache.
- Memory: 4GB (only 2GB available to Comet).

UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

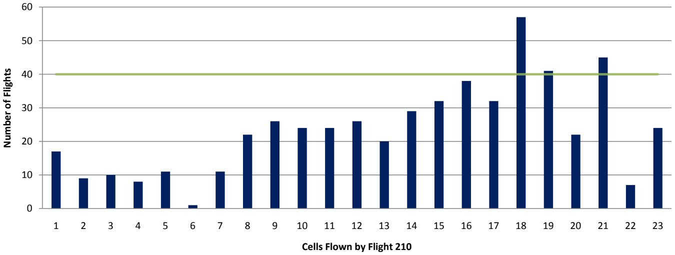
Conclusion

Planned Cell Demands (Layer 3)

- Before optimisation, for the 9–10 pm horizon, in layer 3:

	Mean	Std Dev	Max
before	15.7	24.6	166

- For instance (when all cells have capacity 40):



UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

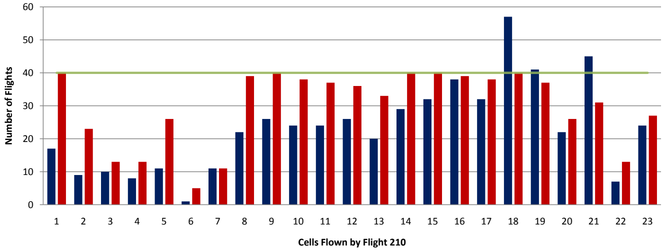
Conclusion

Optimised Cell Demands (Layer 3)

- After optimisation, for the 9–10 pm horizon, in layer 3:

	Mean	Std Dev	Max
before	15.7	24.6	166
after	11.0	12.0	40

- For instance (when all cells have capacity 40):

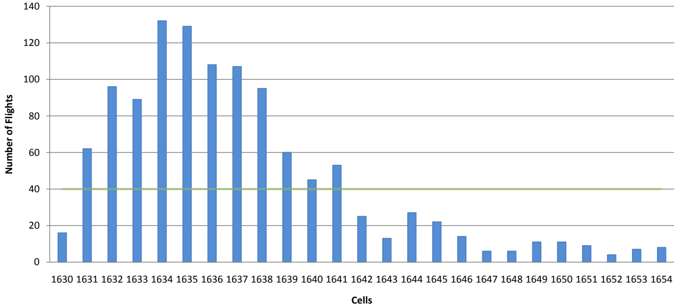


Planned Cell Demands (All Layers)

- Before optimisation, for the 9–10 pm horizon, all layers:

	Mean	Std Dev	Max
before	12.4	22.4	235

- For instance (when all cells have capacity 40):

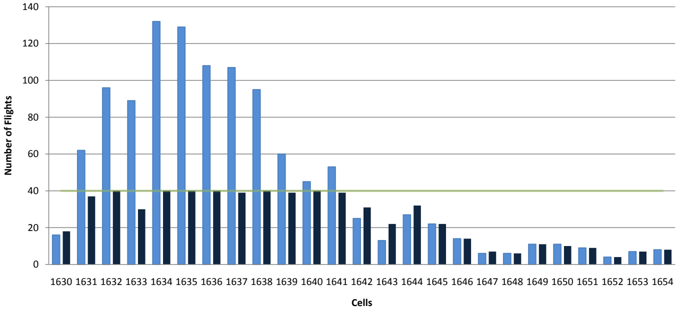


Optimised Cell Demands (All Layers)

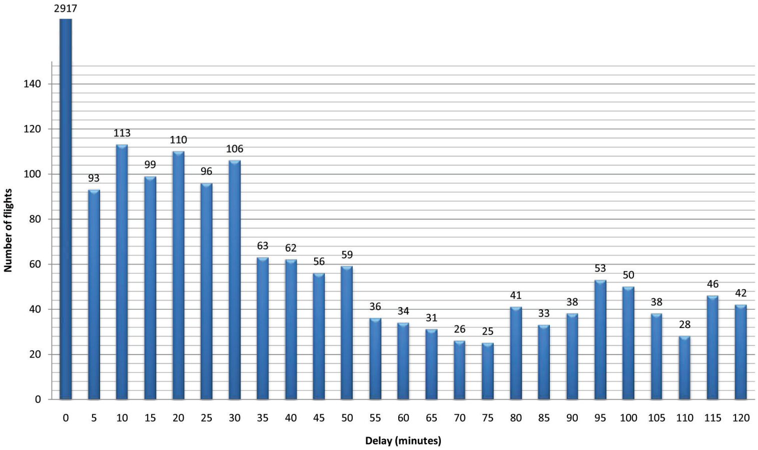
- After optimisation, for the 9–10 pm horizon, all layers:

	Mean	Std Dev	Max
before	12.4	22.4	235
after	8.4	10.4	40

- For instance (when all cells have capacity 40):



Distribution of Delays



More Results

- All the capacity constraints can be satisfied.
- By-product:
  standard deviation of cell demands shrinks significantly.
- Quasi-real-time performance.

Horizon	Run-time	# Waiting	# Airborne
9 – 10 pm	140 sec	4,295	463
5 – 6 pm	620 sec	8,806	811

Horizon	Total Delay	Avg Delay	Demand Dev
9 – 10 pm	65,457 min	13.76 min	−35%
5 – 6 pm	246,267 min	25.61 min	−43%

UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

Conclusion

Outline

UPPSALA UNIVERSITET

Summary and Future Work

- Constraint programming can be used to model and solve large ATM problem instances efficiently.

UPPSALA UNIVERSITET

Summary and Future Work

- Constraint programming can be used to model and solve large ATM problem instances efficiently.
- Extensions:
  - Increase realism by adding extra constraints.
  - Enforce a notion of first-scheduled-first-served.
  - Enforce *load* constraints ($\rightarrow$ less total delay) (load = max number of flights simultaneously present).
  - Vertical re-routing of flights along the planned 2D route.

UPPSALA UNIVERSITET

Summary and Future Work

- Constraint programming can be used to model and solve large ATM problem instances efficiently.
- Extensions:
  - Increase realism by adding extra constraints.
  - Enforce a notion of first-scheduled-first-served.
  - Enforce *load* constraints ($\rightarrow$ less total delay) (load = max number of flights simultaneously present).
  - Vertical re-routing of flights along the planned 2D route.
- Need for a tight integration of planning and control! (Witness huge capacity violations in CFMU plans, and witness our unacceptably high average delays.)

Summary and Future Work

- Constraint programming can be used to model and solve large ATM problem instances efficiently.
- Extensions:
  - Increase realism by adding extra constraints.
  - Enforce a notion of first-scheduled-first-served.
  - Enforce *load* constraints (→ less total delay) (load = max number of flights simultaneously present).
  - Vertical re-routing of flights along the planned 2D route.
- Need for a tight integration of planning and control! (Witness huge capacity violations in CFMU plans, and witness our unacceptably high average delays.)
- Need for a dynamic adjustment of capacity to demand, rather than of demand to capacity?

UPPSALA UNIVERSITET

Objective & Motivation

Model

Experiments

Conclusion

Acknowledgements

- This research project has been co-financed by EUROCONTROL under its Care INO III programme grant 08-121447-C.
- Many thanks to Franck Ballerini, Marc Bisiaux, Marc Dalichampt, Hamid Kadour, Serge Manchon, and Leïla Zerrouki at the EUROCONTROL Experimental Centre for the definition of the problem, the data-set used, and the feedback on our progress.