Algorithms for Managing Sector Congestion with the Airspace Restriction Planner

Ray Jakobovits, Metron Aviation

Parimal Kopardekar, NASA Ames Research Center

Jason Burke, Metron Aviation

Bob Hoffman, Metron Aviation

US/Europe ATM Conference, Barcelona

July 2007

Outline

- Overview of Airspace Restriction Planner and current research objectives
- Major observations
- Issues underlying algorithm selection
- Algorithms considered
- Multi-sector example
- Future research

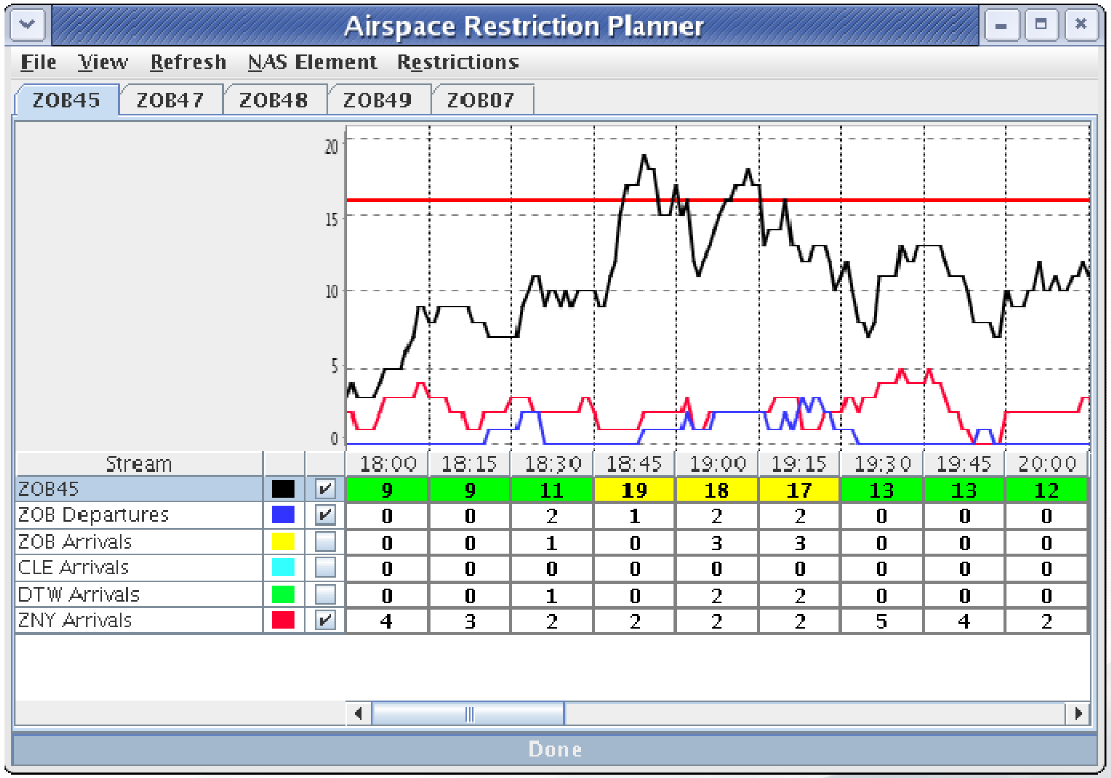
The Sector Congestion Problem

- The Air Traffic Management (ATM) problem of concern: airspace sectors have limited capacity. Traffic overloads must be anticipated and prevented so controllers do not become overwhelmed.
- Our perspective is that of the Traffic Management Coordinator (TMC) at the Air Route Traffic Control Center (ARTCC), as opposed to NAS-wide strategic flow control via the Command Center.
- Decision support for evaluating traffic flow restriction options is very limited.
  - TMCs use the Traffic Situation Display (TSD) to monitor sector demand.
  - Restrictions are applied ‘as needed,’ such as miles-in-trail (MIT), departure approval requests (APREQs), departure spacing programs (DSP), internal ground holds, and reroutes.

Airspace Restriction Planner (ARP)

- ARP is an application within NASA's Future Air Traffic Management Concepts Evaluation Tool (FACET) for managing sector congestion. Features include:

- Traffic characterization display: demand from specific aircraft streams.
- Controllers can sort through regional TFM control options:
  - MITs, Time-Based Metering, reroutes, departure restrictions, capping, and tunneling.
- View control option effectiveness and impacts on individual flights.



Algorithm Testbed

- Our long-range goal is to embed automated algorithms in ARP that help controllers sort through combinations of restrictions that balance demands and capacities.
- The Automated Airspace Restriction Planner (AARP) is currently a MATLAB prototype, integrated with ARP through FACET, for studying advanced airspace restriction strategies.
- Our research to date has focused on algorithms for assigning upstream delays.
  - Dispatching rules.
  - Classical optimization approaches.
  - Hybrid methods.

Two Possible Approaches

Dispatching Rules:

- Also called “heuristic,” “scheduling rule,” or “sequencing rule.”
- Sort flights into a queue then process them one by one.
- Suboptimal, but fast, simple, intuitive.
- Example: Ration-by-Schedule (RBS) for Ground Delay Programs (GDP’s).

Classical Optimization:

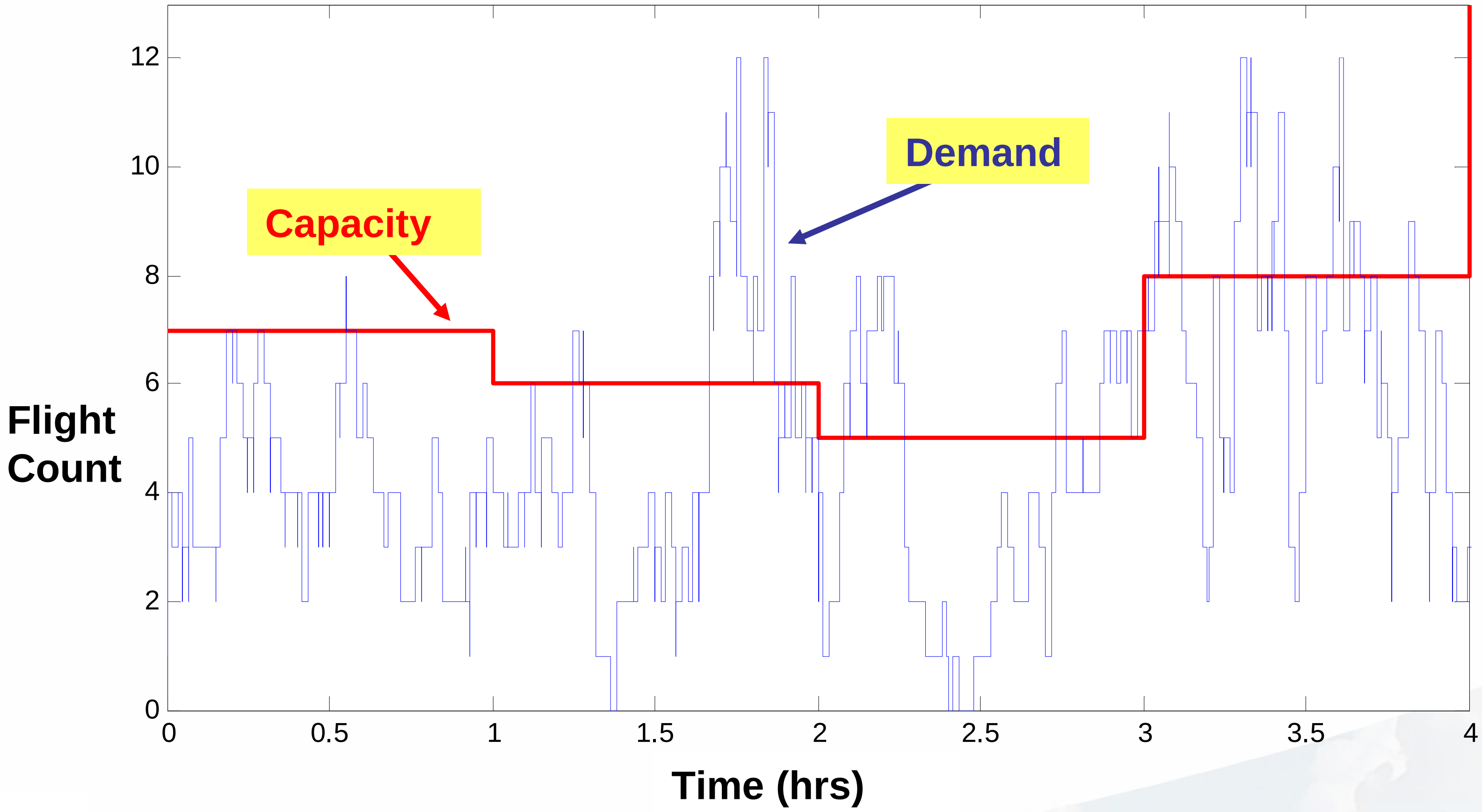
- Math programs to solve the traffic flow management problem or the traffic flow management with rerouting problem.
- Time is discretized into intervals of equal length.
- Decision variables are usually arrival at a sector by or at a certain time interval.
- Integer constraints used to guarantee feasibility lead to potentially long computation times.

Major Observations

- There is no “perfect” algorithm.
- When considering volumes of airspace, resource demands (sector dwell times) vary considerably from flight to flight.
  - Dispatching rules can produce very inefficient solutions when resource demands vary.
- Time discretization introduces inefficiencies.
  - Small time steps lead to excessive optimization run times (hours/days).
- A hybrid algorithm appears to capture the best of both models.
  - Linear program for “near optimality.”
  - Dispatching rule to force feasibility.

A Single-Sector Problem

- ZOB48, 328 flights, 4 hour time period.



111 flight-minutes of excess demand

Single-Sector Solution

Algorithm	Total Assigned Delay	Max. Assigned Delay	No. of Flights Delayed
Dispatching Rule	398 minutes	12 minutes	92
Integer Program 100 Time Steps (~2.5 minutes per step)	724 minutes	79 minutes	63
Integer Program 400 Time Steps (~30 seconds per step)	253 minutes	37 minutes	71
Hybrid Algorithm (LP + dispatching rule) 1,000 time Steps (~15 seconds per step)	242 minutes	32 minutes	98

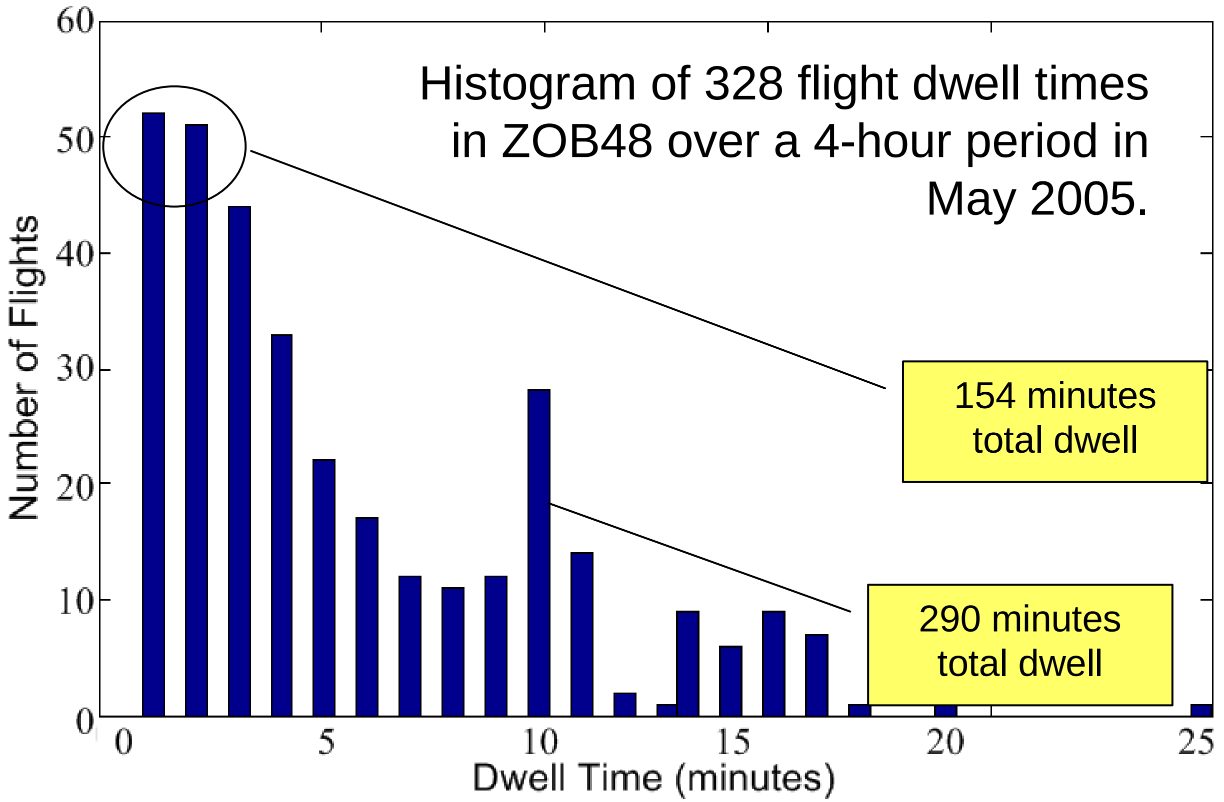
- **Hybrid algorithm reduces total delay compared with other algorithms**
- **Hybrid algorithm (1000 time steps) runs about 20 times faster than integer program (400 time steps)**

Issues for Selection of an Approach

- Issue A: Sector dwell times vary from flight to flight
  - Algorithms should consider dwell time in allocating sector resources.

- Dwell time varies from 1 minute to 25 minutes for ZOB48*.

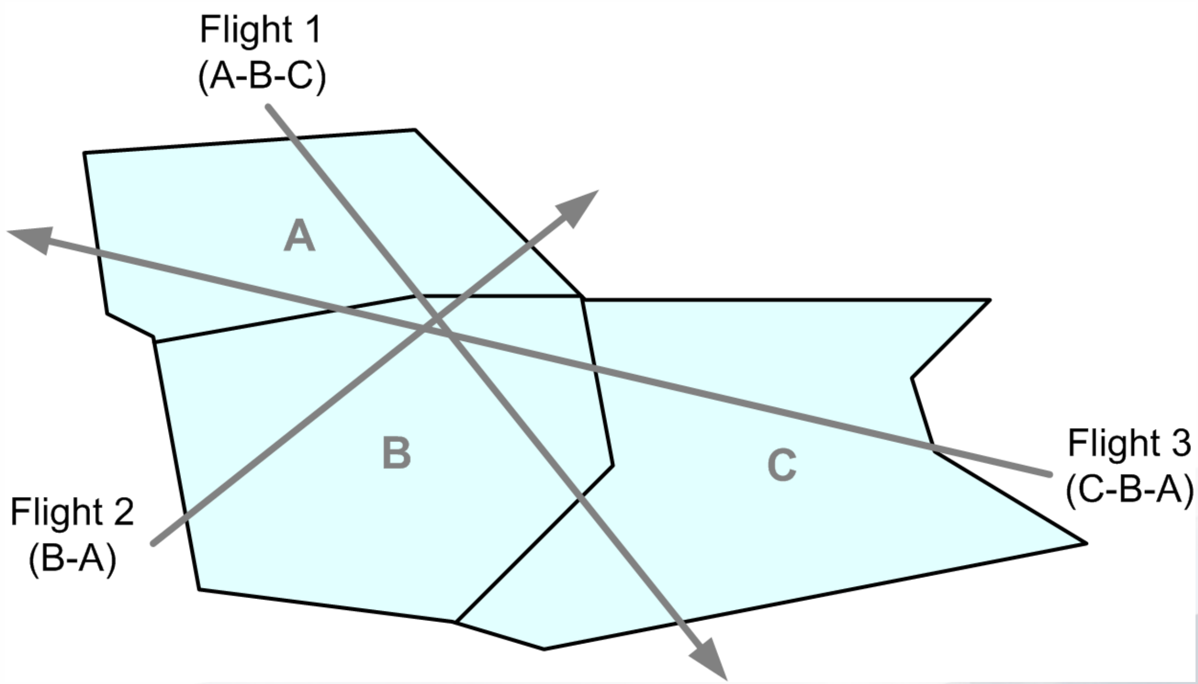
- A small fraction of the flights can use up a lot of the capacity.



* Trajectories computed using FACET

Issues for Selection of an Approach - II

- Issue B: It is not obvious how to sort flights using a dispatching rule for multiple sectors.
  - Flights may not only pass through volumes of airspace going in different directions, but may also enter different sectors according to different arrival sequences.
  - Dispatching rules produce better solutions when flights are sorted in time order.



Issues for Selection of an Approach - III

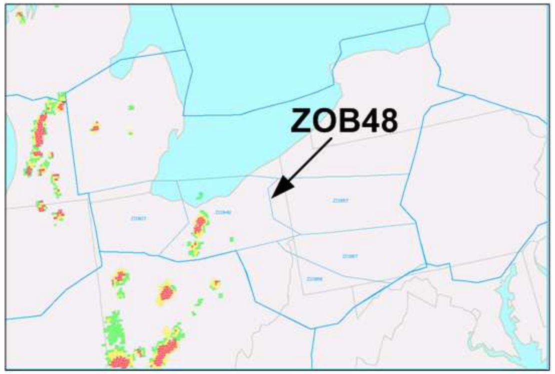
- Issue C: Model granularity vs. computational speed.
  - Math programs with large time intervals run fast but can waste capacity. Formulations with small time intervals produce better solutions but run slowly.
- Issue D: Delay cost evaluation.
  - Models driven by delay-minimizing objective functions require an expression of ground delay costs and air (en route) delay costs.
  - Heuristics struggle with this tradeoff, whereas math programs can weight the tradeoff in the objective function (e.g., air delay minute = 2 x ground delay minute).
- Issue E: Interconnectivity among sectors.
  - Delays imposed on flights prior to entering one sector may trigger over-demand situations in other sectors (the “whack-a-mole” problem). This discourages use of heuristics that solve demand-capacity problems one sector at a time.

AARP Algorithm Descriptions

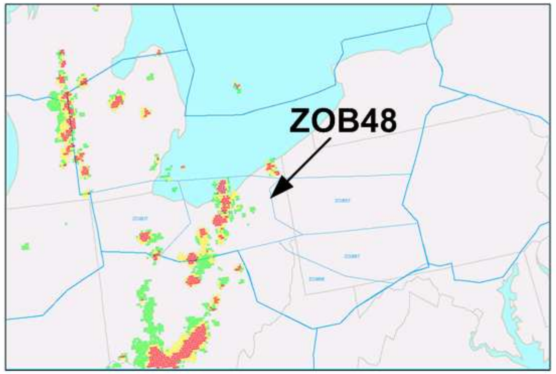
- Model 1: Single-pass dispatching algorithm.
  - Order flights by earliest arrival time to the set of sectors. Compute minimum upstream delay for each flight.
- Model 2: Multi-pass dispatching algorithm.
  - Sort sectors in order of demand/capacity imbalance. Apply Model 1 to worst sector. Re-compute demands and repeat until all imbalances are eliminated.
- Model 3: Integer program.
  - Minimize total (weighted) upstream delay subject to capacity constraints.
- Model 4: Hybrid algorithm
  - Linear program + multi-pass dispatching algorithm.

Multi-Sector Example

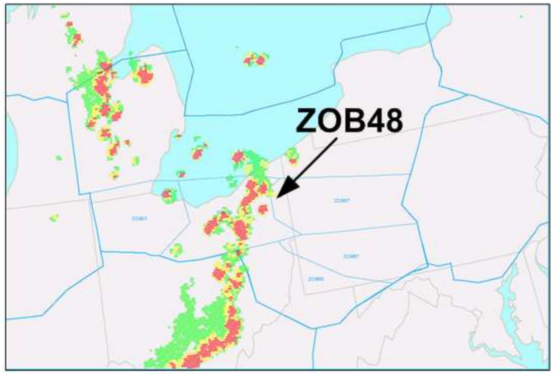
- Severe weather impacting ZOB Center on June 14, 2005.



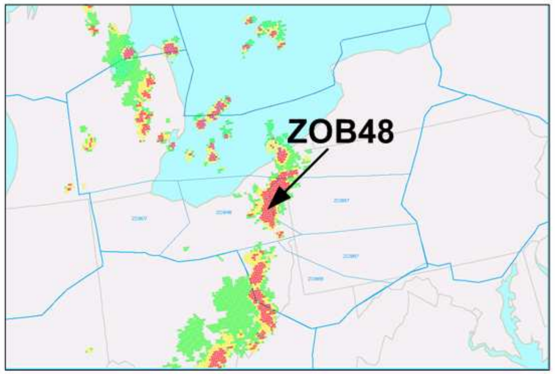
18:00



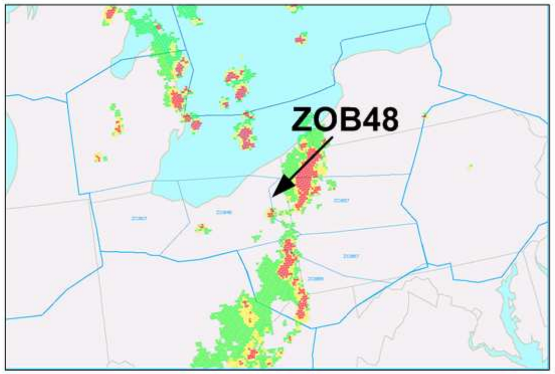
19:00



20:00



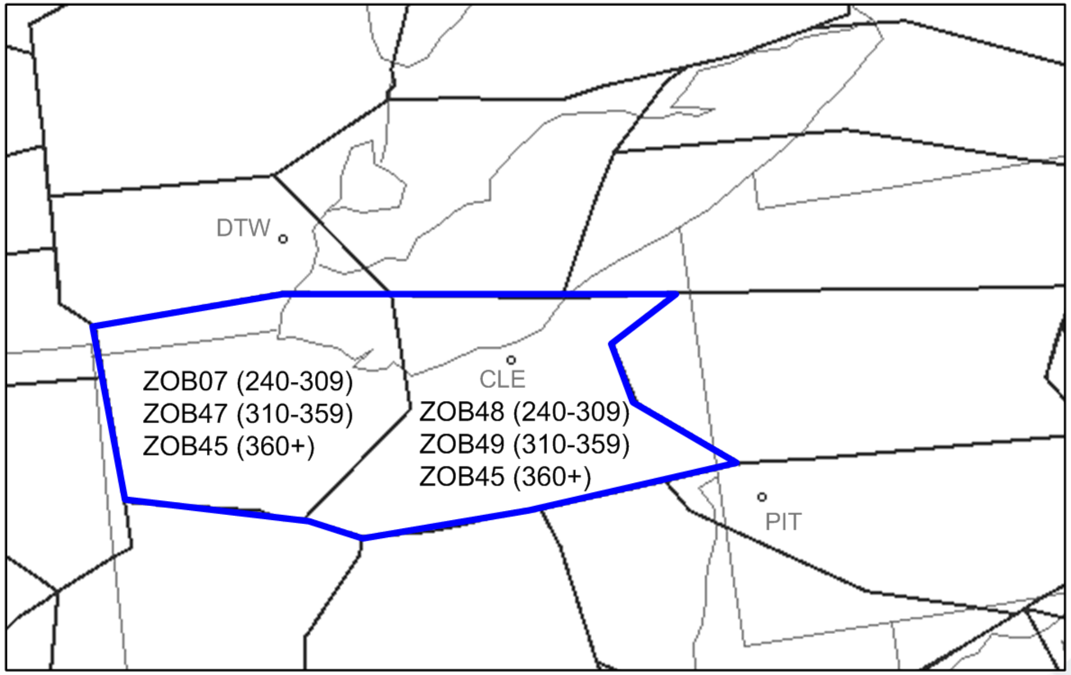
21:00



22:00

Multi-Sector Example (continued)

- 5 ZOB Sectors (ZOB45, ZOB47, ZOB48, ZOB49, ZOB07):
  - Arrivals and Departures for DTW, PIT, and CVG.
  - Arrivals for CLE.
  - Eastbound flows to New York City, Westbound flows to Chicago.
  - Flows between Midwest and Philadelphia and Washington D.C.
  - Other over-flights.
- 4 hours, 977 flights, demands generated by FACET.
- 20-minute maximum en route delay constraint



- Target capacities for two cases:

A. Wx capacities derived from actual traffic (85% of 1-hr. avg. traffic levels).

B. Capacities set to Monitor Alert Parameter (MAP) level for each sector minus a constant.

Multi-Sector Example (continued)

WX Capacities (Excess Demand: 870 flight minutes)

Algorithm	Total Assigned Delay	Max. Assigned Enroute Delay	Max. Assigned Ground Delay	No. of Flights Delayed
Single-pass Dispatching Rule	6,005 minutes	39 minutes	46 minutes	379
Multiple-pass Dispatching Rule	6,463 minutes	39 minutes	49 minutes	398
Hybrid Algorithm	4,672 minutes	21 minutes	181 minutes	425

MAP - 6 (Excess Demand: 710 flight minutes)

Algorithm	Total Assigned Delay	Max. Assigned Enroute Delay	Max. Assigned Ground Delay	No. of Flights Delayed
Single-pass Dispatching Rule	3,229 minutes	23 minutes	31 minutes	283
Multiple-pass Dispatching Rule	3,155 minutes	23 minutes	28 minutes	305
Hybrid Algorithm	2,344 minutes	24 minutes	130 minutes	310

Multi-Sector Example (continued)

WX Capacities (Excess Demand: 870 flight minutes)

Algorithm	Total Assigned Delay	Max. Assigned Enroute Delay	Max. Assigned Ground Delay	No. of Flights Delayed
Single-pass Dispatching Rule	6,005 minutes	39 minutes	46 minutes	379
Multiple-pass Dispatching Rule	6,463 minutes	39 minutes	49 minutes	398
Hybrid Algorithm	4,672 minutes	21 minutes	181 minutes	425

MAP - 6 (Excess Demand: 710 flight minutes)

Algorithm	Total Assigned Delay	Max. Assigned Enroute Delay	Max. Assigned Ground Delay	No. of Flights Delayed
Single-pass Dispatching Rule	3,229 minutes	23 minutes	31 minutes	283
Multiple-pass Dispatching Rule	3,155 minutes	23 minutes	28 minutes	305
Hybrid Algorithm	2,344 minutes	24 minutes	130 minutes	310

- **Significant efficiency improvement with hybrid algorithm**
- **No clear winner between the two dispatching rules**

Multi-Sector Example (continued)

WX Capacities (Excess Demand: 870 flight minutes)

Algorithm	Total Assigned Delay	Max. Assigned Enroute Delay	Max. Assigned Ground Delay	No. of Flights Delayed
Single-pass Dispatching Rule	6,005 minutes	39 minutes	46 minutes	379
Multiple-pass Dispatching Rule	6,463 minutes	39 minutes	49 minutes	398
Hybrid Algorithm	4,672 minutes	21 minutes	181 minutes	425

MAP - 6 (Excess Demand: 710 flight minutes)

Algorithm	Total Assigned Delay	Max. Assigned Enroute Delay	Max. Assigned Ground Delay	No. of Flights Delayed
Single-pass Dispatching Rule	3,229 minutes	23 minutes	31 minutes	283
Multiple-pass Dispatching Rule	3,155 minutes	23 minutes	28 minutes	305
Hybrid Algorithm	2,344 minutes	24 minutes	130 minutes	310

Hybrid algorithm enforces en route delay constraints (at the expense of additional ground delay)

Summary and Future Work

- Summary
  - We have identified key issues associated with algorithm selection for sector congestion management.
  - Our preliminary results indicate that the hybrid algorithm, which combines heuristics with a linear program, strikes a suitable balance between optimal solution and fast run time.
  - The hybrid approach also enables consideration of additional constraints, such as limits on maximum assigned delays.

- Future Work
  - Enhance algorithms to include multiple degrees of freedom including: rerouting, metering, tunneling, and altitude capping.
  - Conduct human-in-the-loop simulations to compare the benefits of automated restrictions compared with current human generated restrictions.

Backup Slides

Model 1: Single-pass Dispatching Algorithm

1. Process exempt flights and compute residual capacities for the set of sectors.
2. Order flights by earliest arrival time to the set of sectors.
3. Process each flight on the list once:
   a. Determine the minimum upstream delay that allows the flight to transit each sector on its planned route without exceeding the residual capacity of any sector in the set.
   b. Apply the resulting delay to the flight and decrement residual capacities for each sector.

Model 2: Multi-pass Dispatching Algorithm

1. Find the sector *s** with the greatest capacity-demand imbalance and pass it to Step 2.
2. Apply Model 1 (single-pass algorithm) to *s**. But each time a flight is processed, restrict its set of sectors to *s**. For each flight, pass final delay value to Step 3.
3. Update demand at every sector.
4. Repeat Steps 1 – 3 until all sectors are resolved.

<u>Notes on Model 2:</u>

1. Resolves capacity-demand imbalances one sector at a time.
2. Order of flights on dispatching list is based on earliest arrival to each sector and may change as sectors are processed.
3. Sectors may need to be revisited.
4. Convergence is ensured by the fact that flight delays can only increase.

Notation for Models 3 and 4

Set of time intervals $t = 1, 2, ..., T$

Set of flights $f = 1, 2, ..., F$

Set of sectors $s = 1, 2, ..., S$

Ψ_f = ordered set of sectors flight f must traverse

E_{fs} = earliest time f can enter sector s (scheduled arrival time)

τ_{fs} = dwell time of f in sector s

$C_s(t)$ = capacity of sector s during time interval t

ω = relative weight of air delay to ground delay

$\alpha_f = 1$ if f is airborne at time t, 0 else.

$x_{fts} = 1$ if flight f enters sector s during interval t, 0 else.

Model 3: Integer Program (and its LP Relaxation)

(0) Objective function: minimize weighted delay

$$\text{Minimize} \quad \sum_{i=1}^{N} d_f \left(1 + \alpha_f \omega\right)$$

subject to

(1) Define and limit flight delay

$$d_f = \sum_{t \geq E_{fs}} t\, x_{fts} - E_{fs} \leq \Delta_{\max} \quad \forall f, s \in \Psi_f$$

(2) Enforce sector capacities

$$\sum_{f} \sum_{k=\max(t-\tau_{is},1)}^{t} x_{fks} \leq C_{ts} \quad \forall t, \forall s$$

(3) Each flight must enter its designated sectors

$$\sum_{t \geq E_{fs}} x_{fts} = 1 \quad \forall f, s \in \Psi_f$$

(4) Binary decision variables

$$x_{fts} \in \{0,1\} \quad \forall f \forall t \forall s$$

(5) Bounded variables (for LP relaxation only)

$$x_{fts} \leq 1 \quad \forall f \forall t \forall s$$

Model 4: Hybrid Algorithm using LPR and multi-pass (Models 2 and 3)

1. Solve the linear program relaxation (LPR).

2. For each flight, apply the resulting delay d_f .

   (Note: demand/capacity imbalances may remain due to non-integer values of x_{fts} .)

3. Apply Model 2 (multi-pass dispatching algorithm) to the solution derived in Step 2. Output this feasible integer solution.

Comments on Model 4

- The solution derived in Step 2 may not be feasible. Multi-pass algorithm restores feasibility.
- The LPR gives Model 2 (multi-pass) a significant ‘jump start’ in the direction of an optimal solution, hence, improves on Model 2.
  - High percent of variables are integer in the solution.
  - Typically very close to a feasible integer solution.
  - The LP identifies flights that consume a disproportionate amount of capacity during high-demand time periods.
- With refined time intervals, this hybrid solution may yield less delay than the IP with coarse time intervals.
  - We could run the IP with refined time intervals, but this greatly increases the solver runtime .