MILP global search algorithm for Aircraft ground movement optimization

Pushkar Godbole

Dept. of Aerospace Engineering
IIT Bombay

Guide
Prof. Abhiram Ranade
Prof. Rajkumar Pant

October 26, 2013

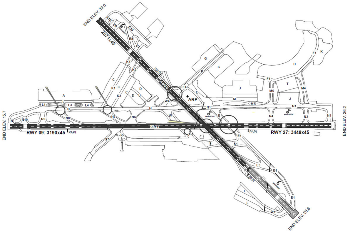
Section 1

Introduction

Air Traffic Control (ATC)

Ground movement

- Arriving & Departing aircrafts with **tentative** times
- Origin - Destination points
- Route through network of taxiways
- Solve overlap conflicts; minimize overall delay



Motivation

Motivation

Current ATC

- Prominently manual scheduling by ATCs: First come first serve (myopic)
- Over-scheduling by airlines leading to unrestrained push-backs
- Scheduling primarily empirical and based on ‘gut’
- 12.5% flight delays in USA in 2012 estimated to have occurred near airports

Motivation

Current ATC

- Prominently manual scheduling by ATCs: First come first serve (myopic)
- Over-scheduling by airlines leading to unrestrained push-backs
- Scheduling primarily empirical and based on 'gut'
- 12.5% flight delays in USA in 2012 estimated to have occurred near airports

Need

- Testable and realistic method for optimal scheduling
- Run-time comparable to current semi-automated systems
- Globally optimal schedule to benchmark the method

Attempts: Commercial

- **CDM** (Collaborative Decision Making):

  Collaborative data assimilation and dissemination

- **ASMGCS** (Advanced Surface Movement Guidance and Control Systems):

  Automated surface surveillance and scheduling

- **NextGen** (Next Generation Air Transportation System):

  De-centralized collaborative decision making and planning

Attempts: Academic

Macro-optimization

- *Capacity estimation:* Generate airport capacity envelope
- *Push-back optimization:* Generate optimal push-backs for aircrafts at terminals
- *Weather mitigation:* Models to predict weather and improve decision making

Micro-optimization

- *Stochastic:* Use heuristics and intelligent elimination to generate schedule
- *Deterministic\Semi-deterministic:* Cut down solution space to deterministically solvable one; find optimal schedule

Micro-optimization

- Airport map interpreted as a graph
- Detailed schedule for each aircraft; time stamps at nodes
- Better delay minimization than macro-optimization

Concerns

- Highly dynamic; daily changing schedules may cause discomfort to ATCs and pilots
- Heavy need of ATC - pilot interaction over an already congested radio network
- Justifiable mistrust amongst ATCs on an automated system; heavy testing required

Micro-optimization

Stochastic

- **Genetic algorithm** (Liu):
  - First N paths considered for each aircraft
  - Binary decision variable corresponding to each arc
  - Output: Chromosome chain of decision variables
- **Genetic + A*** (Gotteland):
  - Genetic algorithm for path selection for each aircraft
  - A* for prioritizing aircrafts during conflicts
- **Bacterial Foraging** (Bijal):
  - Dijkstra's algorithm for shortest path allocation
  - Bacterial Foraging (BAFO) for conflict resolution & delay allocation

Micro-optimization

Deterministic & Semi-deterministic

- **MILP** (Smeltink):
  - Fixed (shortest) paths allocated to aircrafts
  - MILP model used for conflict resolution and delay allocation
  - Rolling Horizons for realistic run-times
- **Time Dependent Shortest Path (TDSP)** (Baik):
  - Aircrafts prioritized on first come first serve
  - One aircraft scheduled per iteration
  - TDSP generated from a time-space graph
- **MILP+TDSP** (Gupta):
  - TDSP used to generate schedule
  - Paths allocation based on TDSP schedule
  - MILP used for conflict resolution

Section 2

Global Search

Global optimum

Need of global optimum to benchmark other methods

Global optimum

Need of global optimum to benchmark other methods

Each aircraft

- has multiple choices of routes
- can wait anywhere on the route to solve overlap conflicts
- can wait for any duration anywhere on the route to do so

Global optimum

Need of global optimum to benchmark other methods

Each aircraft

- has multiple choices of routes
- can wait anywhere on the route to solve overlap conflicts
- can wait for any duration anywhere on the route to do so

Infinitely many solutions?

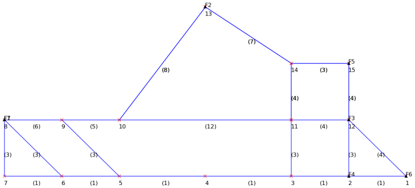
Requirements

Algorithm must:

- deliver at least one globally optimal solution: any other solution at best may be as good
- take a (possibly long but) finite run-time
- deliver a realistic solution
- be such that, approach ensures global optimality of solution

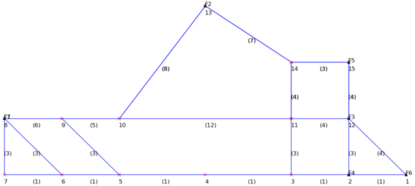
Assumptions

Assumptions



Map discretized to arcs & nodes

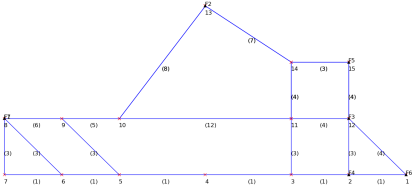
Assumptions



Map discretized to arcs & nodes

Time generally discretized into slots (10 secs)

Assumptions



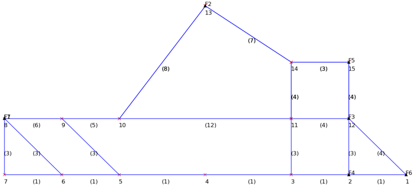
A **B**

Waiting: **A** and **B** are equivalent w.r.t. delay

Map discretized to arcs & nodes

Time generally discretized into slots (10 secs)

Assumptions



Map discretized to arcs & nodes

Time generally discretized into slots (10 secs)

A

B

Waiting: **A** and **B** are equivalent w.r.t. delay

- Fixed origin-destination & start-time
- Infinite acceleration & Constant velocities
- Aircrafts: Point objects

Parameters

Parameters

Inputs

- Map : Interpreted as a graph
- Aircrafts
    - Starting time
    - Origin
    - Destination
    - Aircraft speed
    - Trailing separation
    - Priority

Parameters

Inputs

- Map : Interpreted as a graph
- Aircrafts
  - Starting time
  - Origin
  - Destination
  - Aircraft speed
  - Trailing separation
  - Priority

Objective

Total time : Summation over the destination times of all aircrafts

Parameters

Inputs

- Map : Interpreted as a graph
- Aircrafts
    - Starting time
    - Origin
    - Destination
    - Aircraft speed
    - Trailing separation
    - Priority

Objective

Total time : Summation over the destination times of all aircrafts

Output

Schedule : Time stamps at every node for each aircraft

Algorithm

- Path Finding
- Flight-Path combinations
- Local Mimima
- Global Minimum
- Post processing

Algorithm

Path Finding

- All origin-destination pairs identified from given aircrafts
- DFS applied for each pair: all possible paths explored
- DFS breaks on cycle-detection to avoid infinite loops

Flight-Path combinations

Local Mimima

Global Minimum

Post processing

Algorithm

Path Finding

Flight-Path combinations

All possible combinations of flight-path allocations generated from aircrafts and corresponding path pools

Local Mimima

Global Minimum

Post processing

Algorithm

Path Finding

Flight-Path combinations

Local Mimima

- For every flight-path combination, MILP run
- Conflict free schedule deterministically generated
- Local minimum for particular flight-path combination identified

Global Minimum

Post processing

Algorithm

Path Finding

Flight-Path combinations

Local Mimima

Global Minimum

All local minima compared and global minimum identified

Post processing

Algorithm

Path Finding

Flight-Path combinations

Local Mimima

Global Minimum

Post processing

- Feasibility check run on solution to detect spacio-temporal overlaps
- Feasible, globally optimal solution animated in real-time
- 3D space-time plot of solution generated

Section 3

MILP Design

Mathematically

Identify a schedule with minimum total delay for given flight-path combination

t_{ik} : Time when aircraft i reaches node k / enters arc k
P_i : Priority of aircraft i; landing aircrafts have higher priority

Objective

Minimize : $\sum_{i=0}^{N} P_i t_{in}$
$N =$ Total no. of flights
$n =$ Last / Destination node in the path of flight i

Output

Values of t_{ik} for $i \in (0, N)$ & $k \in (0, n)$

Constraints

- Every aircraft has a maximum speed
- Aircrafts entering an arc from opposite directions must do so exclusively of each other in time
- Minimum separation equal to the minimum amongst (trailing separation of leading aircrafts) and (arc length) must always be maintained between an aircraft following another

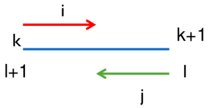
Constraints : Travel time

Travel time on an arc should be greater than or equal to the (arc length)/(aircraft speed)

- $t_{i(k+1)} - t_{ik} \geqslant \frac{length_{ik}}{speed_i}$

Constraints : Head-on collision

2 aircrafts entering an arc from opposite directions must have no temporal overlap



- $M \times x_a + t_{ik} - t_{j(l+1)} \geqslant 0$
- $M \times (1 - x_a) + t_{jl} - t_{i(k+1)} \geqslant 0$

$M \gg 0$ & $x_a \in [0, 1]$

J leads $(x_a = 0)$

$t_{ik} - t_{j(l+1)} \geqslant 0$

I leads $(x_a = 1)$

$t_{jl} - t_{i(k+1)} \geqslant 0$

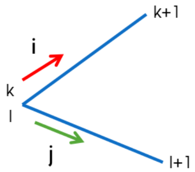
Constraints : Common Node 1

For aircraft i following aircraft j while entering arcs with 1^{st} node common:

if *length of the arc of* j *is less than it's trailing separation*:
then i must enter it's arc only after j has exited it's own

Else, if *length of arc of* j *is greater than or equal to it's trailing separation*:
i may enter it's arc only after j is at least at a distance equivalent to it's trailing separation from common node 1

and visa-versa.



Constraints : Common Node 1

- If j leads & length of arc l is less than sep$_j$, then i enters arc k only after j has exited arc l i.e. reached node $(l+1)$
- If j leads & length of arc l is greater than or equal to sep_j , then i enters arc k after j has traversed distance of at least sep_j on arc l
- If i leads & length of arc k is less than sep_i, then j enters arc l only after i has exited arc k i.e. reached node $(k+1)$
- If i leads & length of arc k is greater than or equal to sep_i , then j enters arc l after i has traversed distance of at least sep_i on arc k

- If $Length_l < sep_j$,

$$M \times x_b + t_{ik} - t_{j(l+1)} \geqslant 0$$

- Else (if $Length_l \geqslant sep_j$),

$$M \times x_b + t_{ik} - t_{jl} \geqslant \frac{sep_j}{speed_j}$$

- If $Length_k < sep_i$,

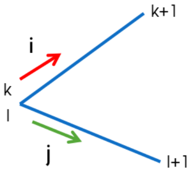
$$M \times (1 - x_b) + t_{jl} - t_{i(k+1)} \geqslant 0$$

- Else (if $Length_k \geqslant sep_i$),

$$M \times (1 - x_b) + t_{jl} - t_{ik} \geqslant \frac{sep_i}{speed_i}$$

$M \gg 0$ & $x_b \in [0, 1]$

Constraints : Common Node 1



	J leads $(x_b = 0)$	I leads $(x_b = 1)$	
$Length_l < sep_j$	$t_{ik} - t_{j(l+1)} \geqslant 0$	$t_{jl} - t_{i(k+1)} \geqslant 0$	$Length_k < sep_i$
$Length_l \geqslant sep_j$	$t_{ik} - t_{jl} \geqslant \frac{sep_j}{speed_j}$	$t_{jl} - t_{ik} \geqslant \frac{sep_i}{speed_i}$	$Length_k \geqslant sep_i$

$\frac{sep_j}{speed_j}$ is the time taken by j to traverse a distance of sep_j

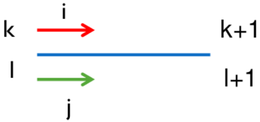
Constraints : Common Node 2

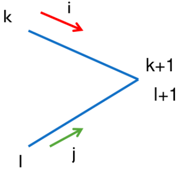
For an aircraft i following aircraft j while entering arcs with 2^{nd} node common:

if *length of the arc of i is less than j's trailing separation*: then i must enter it's arc only after j has exited it's own arc

Else, if *length of arc of i is greater than or equal to j's trailing separation*:
then j must exit it's arc when i is at least at a distance equal to j's trailing separation from the common node 2

and visa versa





Constraints : Common Node 2

- If j leads & length of arc k is less than sep_j, then i enters arc k only after j has exited arc l i.e. reached node $l+1$
- If j leads & length of arc k is greater than or equal to sep_j , then j exits arc l when the distance of i is at least equal to sep_j from node $k+1$
- If i leads & length of arc l is less than sep_i , then j enters arc l only after i has exited arc k i.e. reached node $k+1$.
- If i leads & length of arc l is greater than or equal to sep_i, then i exits arc k when the distance of j is at least equal to sep_i from node $l+1$

- If $Length_k < sep_j$,

$$M \times x_c + t_{ik} - t_{j(l+1)} \geqslant 0$$

- Else (if $Length_k \geqslant sep_j$),

$$M \times x_c + t_{i(k+1)} - t_{j(l+1)} \geqslant \frac{sep_j}{speed_i}$$

- If $Length_l < sep_i$,

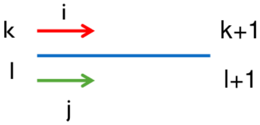
$$M \times (1 - x_c) + t_{jl} - t_{i(k+1)} \geqslant 0$$

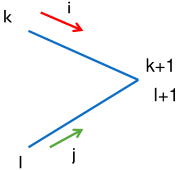
- Else (if $Length_l \geqslant sep_i$),

$$M \times (1 - x_c) + t_{j(l+1)} - t_{i(k+1)} \geqslant \frac{sep_i}{speed_j}$$

$$M \gg 0 \ \& \ x_c \in [0, 1]$$

Constraints : Common Node 2





	J leads ($x_c = 0$)	I leads ($x_c = 1$)	
$Length_k < sep_j$	$t_{ik} - t_{j(l+1)} \geqslant 0$	$t_{jl} - t_{i(k+1)} \geqslant 0$	$Length_l < sep_i$
$Length_k \geqslant sep_j$	$t_{i(k+1)} - t_{j(l+1)} \geqslant \frac{sep_j}{speed_i}$	$t_{j(l+1)} - t_{i(k+1)} \geqslant \frac{sep_i}{speed_j}$	$Length_l \geqslant sep_i$

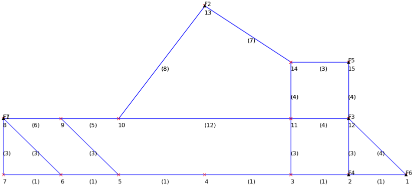
$\frac{sep_j}{speed_i}$ is the time taken by i to traverse a distance of sep_j

Section 4

Results

Bijal (CSIA)

Flight	Origin	Destination	Start time	Speed	Trailing sep	Priority
1	1	8	13	1	3	1
2	1	13	12	1	3	1
3	5	12	6	1	3	1
4	9	2	6	1	3	1
5	1	15	0	1	3	1
6	15	1	0	1	3	1
7	1	8	0	1	3	1



Cost

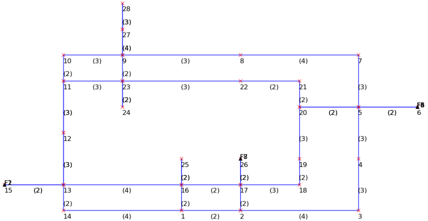
Bijal: 186

Global search: 166

Run time: 183 mins

Roling

Flight	Origin	Destination	Start time	Speed	Trailing sep	Priority
1	26	15	7	1	3	1
2	24	15	6	2	3	1
3	25	6	10	2	3	1
4	25	6	8	1	3	1
5	25	6	16	2	3	1
6	24	6	14	1	3	1
7	28	26	0	1	3	1
8	28	26	3	1	3	1



Cost

Roling: 362

Global search: 204

Run time: 51 mins

Future work

- Global optimum identified
- Global search can not be used on the fly: Heavy run-time
- Stage II: Need method to deliver fast and near-optimal solutions

Direction

- Hybrid approach: Stochastic + Deterministic
- Rolling horizons
- Learning from empirical data

Merci! :)