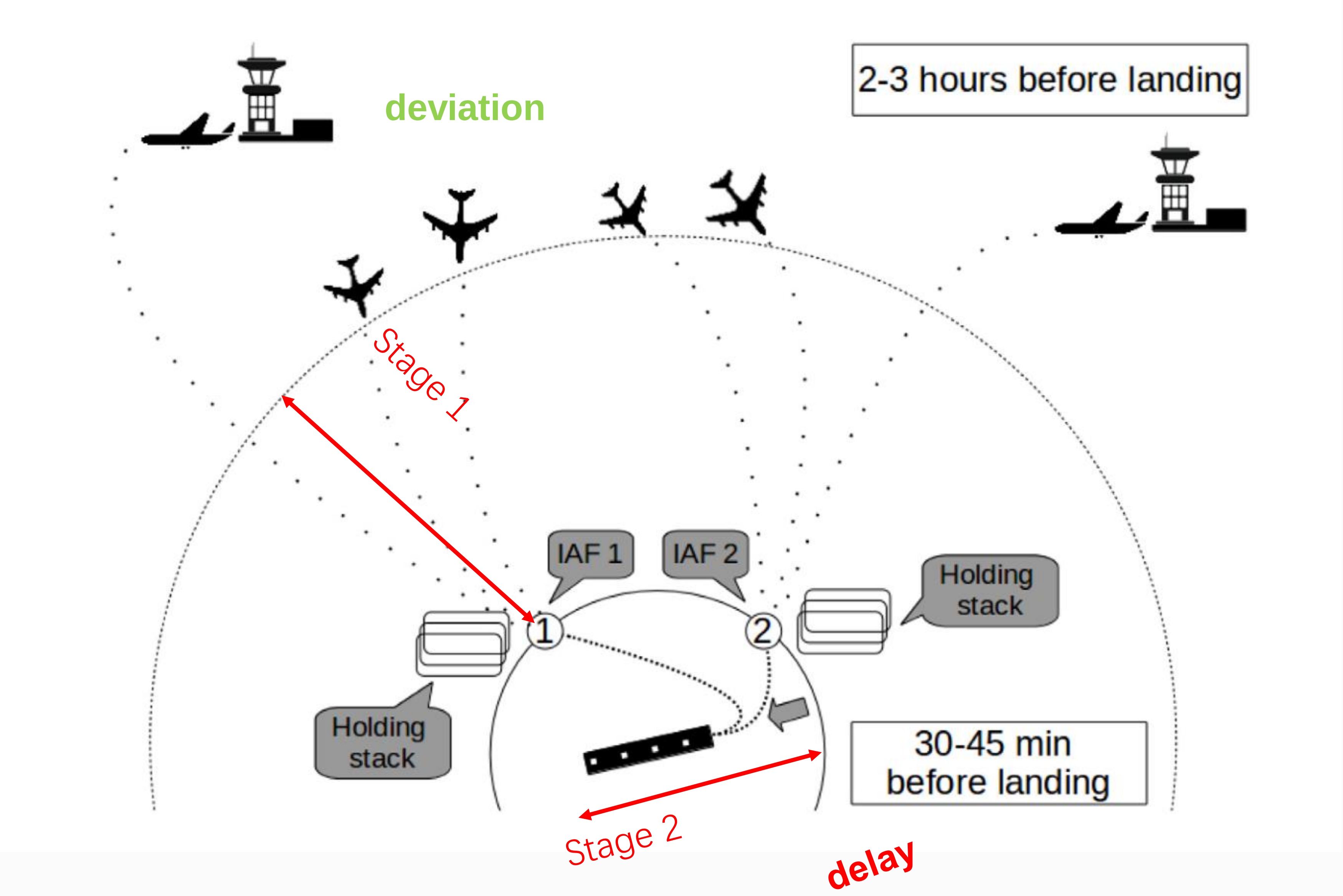
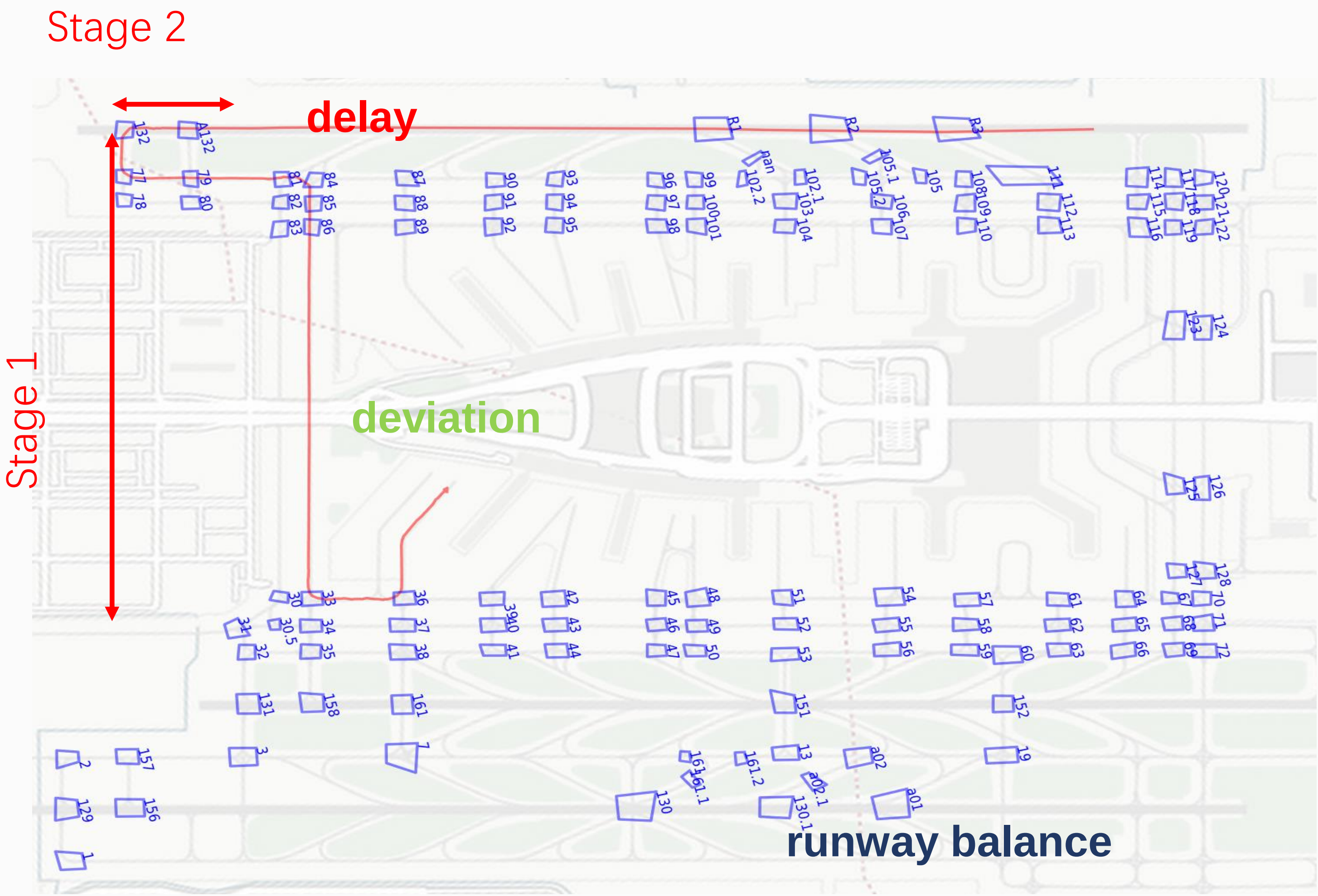
Integrated runway scheduling under operational time uncertainty: A case study in ZGGG

Zhuoming Du,

College of Civil Aviation,

Nanjing University of Aeronautics and Astronautics (NUAA)


2-3 hours before landing
deviation
Stage 1
IAF 1
IAF 2
1
2
Holding stack
Holding stack
30-45 min before landing
Stage 2
delay

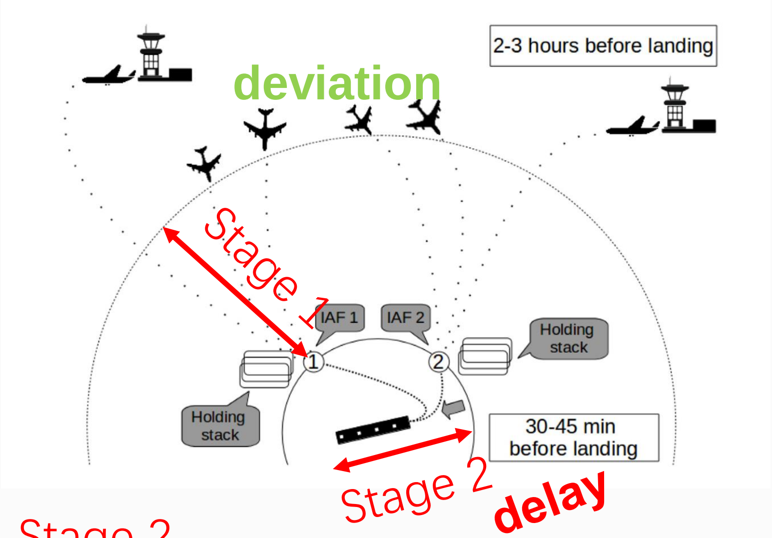

Stage 2
delay
Stage 1
deviation
runway balance

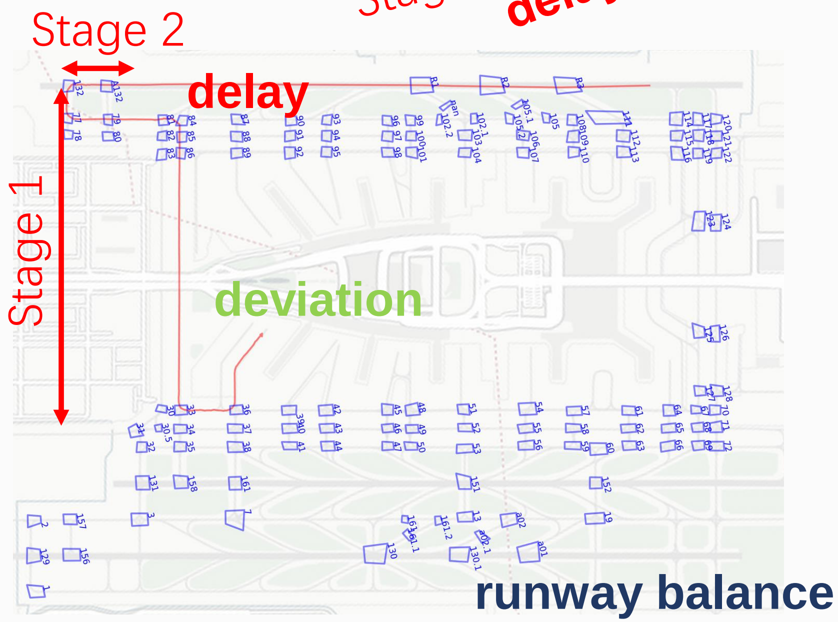
Integrated runway scheduling

- Objective: deviation, runway balance and delay

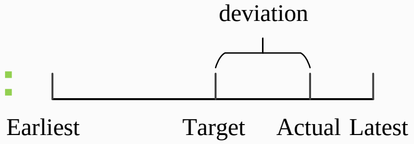
- First stage:
- Decision 1: entering time/pushback time
- Decision 2: landing/departing runway

- Second stage:
- Decision 1: runway sequence
- Decision 2: landing/ take off time

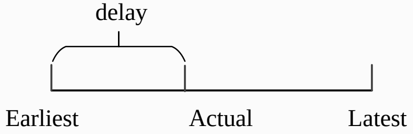




Deviation:



Delay:



Sets

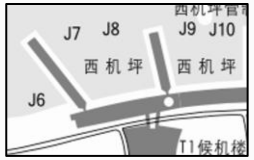
Sets	
F	Set of aircraft in the time horizon, $i \in F$
A	Subset of F, set of arrival aircraft
D	Subset of F, set of departure aircraft
R	Set of runways, $r \in R$

Decision variables

For departures:

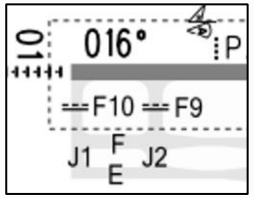
t_i

- Pushback time (or SOBT)



x_i

- Runway threshold time

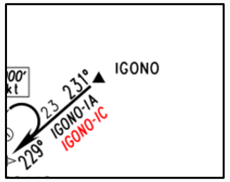


r_i

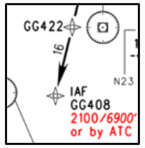
- Take off time (or STOT)

For arrivals:

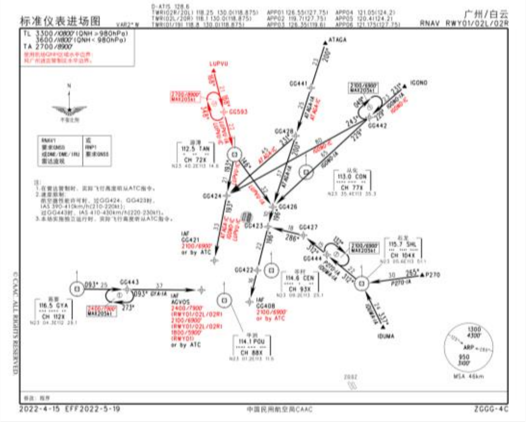
- Entry time



- IAF time



- Landing time



Decision variables

δ_{ij}

- Binary variable, 1 if $r_i < r_j$

y_{ir}

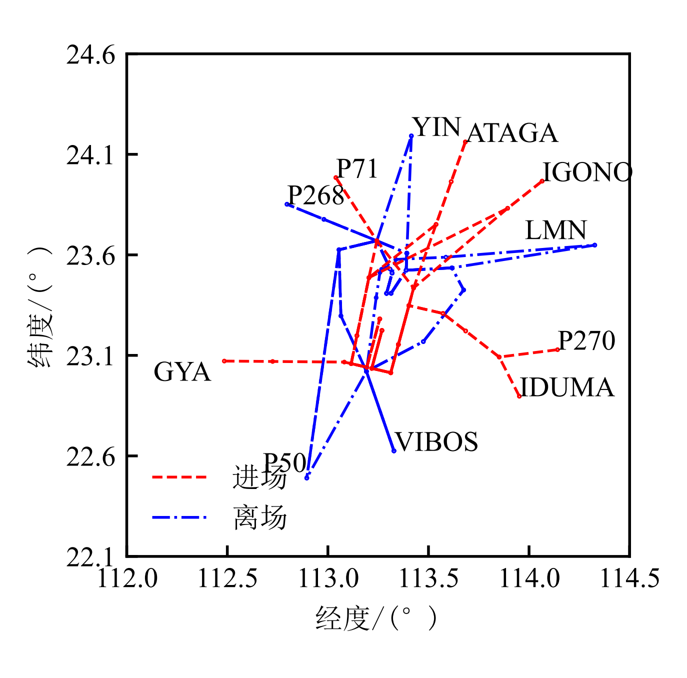
- Binary variable, 1 if runway r is assigned to ac. i

z_{ij}

- Binary variable, 1 if ac. i and ac. j use the same runway

Constraints

STAR



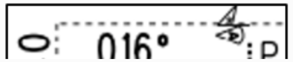

```python
for i in ALL:
    if ac_list[i].ad == 'd':
        m.addConstr(y[i, 2] == 0)
    else:
        m.addConstr(y[i, 1] == 0)
        if ac_list[i].entryfix == 'P270' or ac_list[i].entryfix == 'IDUMA':
            m.addConstr(y[i, 0] == 0)
        if ac_list[i].entryfix == 'GYA':
            m.addConstr(y[i, 2] == 0)
```

Time window

t_i

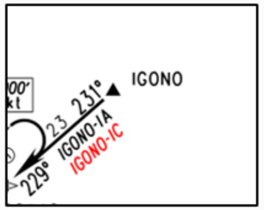
For departures:

- Pushback time (or SOBT)



For arrivals:

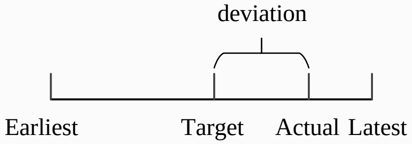
- Entry time




```
m.addConstrs((t[i] >= lb_t[i] for i in ALL), name: "lb")
m.addConstrs((t[i] <= ub_t[i] for i in ALL), name: "ub")
```

Parameters	LB	UB
Entry Time Window	-60	600
Push-back Time Window	-60	600
(A) Approach Time Window	500	800
(D) Waiting Time Window	0	300
Arrival Flight time	800	1600
Taxi-out Time	600	800

$$E_i \le t_i \le L_i, \forall i \in F$$



Runway related

$$\sum_{r\in R} y_{ir} = 1, \forall i \in F, \forall r \in R$$

$$z_{ij} = \sum_{r\in R} y_{ir} \cdot y_{jr}, \forall i, j \in F$$

```
# 唯一性约束
m.addConstrs((gp.quicksum(y[i, r] for r in R) == 1 for i in ALL), name: "unique1")
# m.addConstrs(z[i, j] >= y[i, r] + y[j, r] - 1 for i in ALL for j in ALL for r in R if i != j)
m.addConstrs(z[i, j] ==gp.quicksum( y[i, r] * y[j, r] for r in R) for i in ALL for j in ALL  if i > j)
```

$$z_{ij} \geq y_{ir} + y_{jr} - 1, \forall i, j \in F, r \in R$$

Two-stage stochastic programming

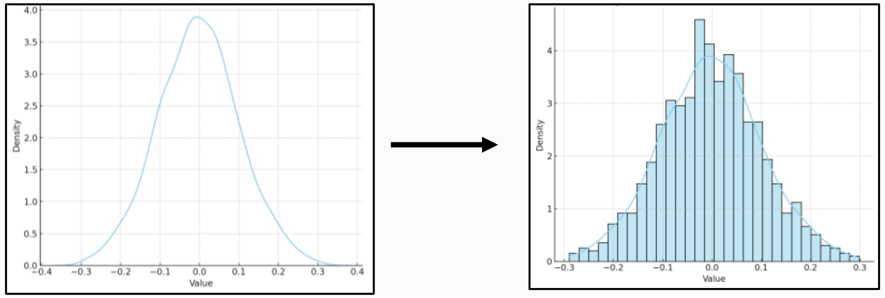
- Objective: deviation, **runway balance** and **delay**

Assume first stage taxi time and flight time

$$t \sim P(t_a, t_b)$$

- First stage:
- Decision 1: entering time/pushback time
- Decision 2: landing/departing runway

Reformulate the continuous distribution using a scenario based sampling.



- Second stage:
- Decision 1: runway sequence
- Decision 2: landing/ take off time

After realization the uncertain random variables, fix stage 1 decision and do the second stage for each scenario, and iteratively adding cutting planes into stage 1.

Or solve two stages at a time!

Deterministic

- Taxi time = constant
- Flight time = constant

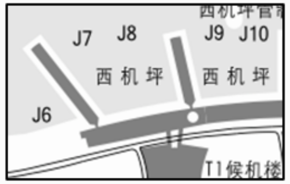
Operational time

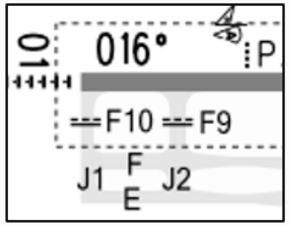
t_i

x_i

For departures:

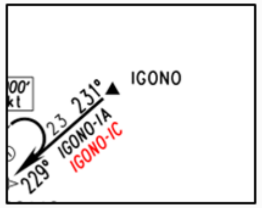
- Pushback time (or SOBT)
- Runway threshold time





For arrivals:

- Entry time
- IAF time



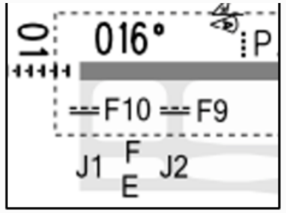
Parameters	LB	UB
Entry Time Window	-60	600
Push-back Time Window	-60	600
(A) Approach Time Window	500	800
(D) Waiting Time Window	0	300
Arrival Flight time	800	1600
Taxi-out Time	600	800

```
## stage 2
m.addConstrs(x[s, i] == t[i] + ds[s][i] for s in S for i in ALL)
```

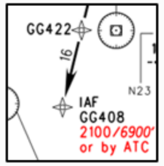
$$x_i = t_i + \tilde{d}_i, \forall i \in F$$

Dep. Hold & arr. flight time window

x_i • Runway threshold time



• IAF time



r_i • Take off time (or STOT)

• Landing time

$$x_i + \underline{u} \le r_i \le x_i + \overline{u}, \forall i \in A$$

Parameters	LB	UB
Entry Time Window	-60	600
Push-back Time Window	-60	600
(A) Approach Time Window	500	800
(D) Waiting Time Window	0	300
Arrival Flight time	800	1600
Taxi-out Time	600	800

```
for i in ALL:
    if ac_list[i].ad == 'a':
        m.addConstrs(x[s, i] + lb_ua <= r[s, i] for s in S)
        m.addConstrs(r[s, i] <= x[s, i] + ub_ua for s in S)
    elif ac_list[i].ad == 'd':
        m.addConstrs(x[s, i] + lb_ud <= r[s, i] for s in S)
        m.addConstrs(r[s, i] <= x[s, i] + ub_ud for s in S)
```

sequence of runway

$$r_j \geq r_i + z_{ij} sep_{ij} - \delta_{ji} M, \forall i, j \in F$$

$$\delta_{ij} + \delta_{ji} = 1, \forall i, j \in F, i > j$$

Table 2
Single-runway separation requirements according to aircraft categories and to operations (in seconds). A refers to Arrival, D refers to Departure, and C refers to Crossing. H refers to Heavy, M refers to Medium, and L refers to Light. For example, the minimum runway separation between an "A-H" (Arrival-Heavy) and a "D-M" (Departure-Medium) is 60 s.

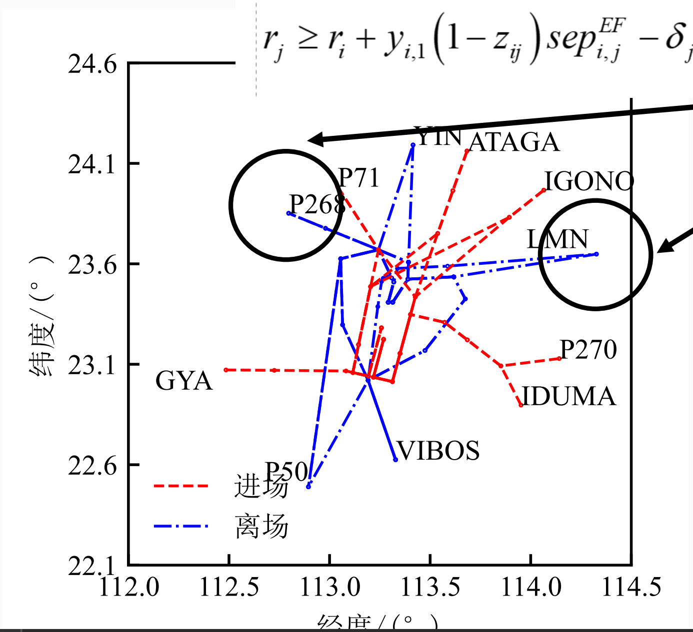
Operation-category		Trailing aircraft						
		A-H	A-M	A-L	D-H	D-M	D-L	C
Leading aircraft	A-H	96	157	207	60	60	60	–
	A-M	60	69	123	60	60	60	–
	A-L	60	69	82	60	60	60	–
	D-H	60	60	60	96	120	120	60
	D-M	60	60	60	60	60	60	60
	D-L	60	60	60	60	60	60	60
	C	–	–	–	40	40	40	10

```
m.addConstrs(
    r[s, j] >= r[s, i] + z[i, j] * sep[i, j] - delta[s, j, i] * 10000 for s in S for i in ALL for j in ALL if
    i != j)
m.addConstrs(delta[s, j, i] + delta[s, i, j] == 1 for s in S for i in ALL for j in ALL if i > j)
```

separation of Exit Fix (EF)

$$r_j \ge r_i + y_{i,0}\left(1 - z_{ij}\right) sep^{EF}_{i,j} - \delta_{ji} M, \forall i, j \in F, EF \in \{\text{LMN}, \ldots\}$$

$$r_j \ge r_i + y_{i,1}\left(1 - z_{ij}\right) sep^{EF}_{i,j} - \delta_{ji} M, \forall i, j \in F, EF \in \{\text{P268}, \ldots\}$$



Dep. From 01, 02L to LMN or P268 require additional separation

- e.g., if ac. i from 01 is earlier than ac. j from 02L,
- then, $t_j \ge t_i + sep^{\text{EF}}_{i,j}$

where $sep^{\text{EF}}_{i,j} = 240\text{s}$

```
for i in ALL:
    for j in ALL:
        if ac_list[i].entryfix =='LMN' and ac_list[j].entryfix =='LMN' :
            m.addConstrs(r[s, j] >= r[s, i] + y[i,1]*(1-z[i, j]) * 240 - delta[s, j, i] * 10000 for s in S for i in ALL for j in ALL if i != j)
        elif ac_list[i].entryfix =='P268' and ac_list[j].entryfix =='P268' :
            m.addConstrs(r[s, j] >= r[s, i] + y[i,1]*(1-z[i, j]) * 240 - delta[s, j, i] * 10000 for s in S for i in ALL for j in ALL if i != j)
```

$$r_j \geq r_i + y_{i,0}\left(1-z_{ij}\right)sep_{i,j}^{EF} - \delta_{ji}M, \forall i,j \in F, EF \in \{\mathrm{LMN},...\}$$

$$r_j \geq r_i + y_{i,1}\left(1-z_{ij}\right)sep_{i,j}^{EF} - \delta_{ji}M, \forall i,j \in F, EF \in \{\mathrm{P268},...\}$$

```
for i in ALL:
    for j in ALL:
        if ac_list[i].entryfix =='LMN' and ac_list[j].entryfix =='LMN' :
            m.addConstrs(r[s, j] >= r[s, i] + y[i,1]*(1-z[i, j]) * 240 - delta[s, j, i] * 10000 for s in S for i in ALL for j in ALL if i != j)
        elif ac_list[i].entryfix =='P268' and ac_list[j].entryfix =='P268' :
            m.addConstrs(r[s, j] >= r[s, i] + y[i,1]*(1-z[i, j]) * 240 - delta[s, j, i] * 10000 for s in S for i in ALL for j in ALL if i != j)
```

Objectives

Early or delay?

$$\Delta \geq \left| t_i - t_i^{\text{preferred}} \right|, \forall i \in F$$

$$\min_{t,y,z} \left[\left(Z_{\max} - Z_{\min} \right) + \Delta \right] +$$

$$\alpha_i \geqslant T_i - x_i \qquad i = 1, \ldots, P, \qquad (14)$$

$$0 \leqslant \alpha_i \leqslant T_i - E_i \qquad i = 1, \ldots, P, \qquad (15)$$

$$\beta_i \geqslant x_i - T_i \qquad i = 1, \ldots, P, \qquad (16)$$

$$0 \leqslant \beta_i \leqslant L_i - T_i \qquad i = 1, \ldots, P, \qquad (17)$$

$$x_i = T_i - \alpha_i + \beta_i \qquad i = 1, \ldots, P, \qquad (18)$$

- Dep. From 01, 02L to LMN or P268 require additional separation
- e.g., if ac. i from 01 is earlier than ac. j from 02L,
- then, $t_j \geq t_i + sep_{i,j}^{\text{EF}}$

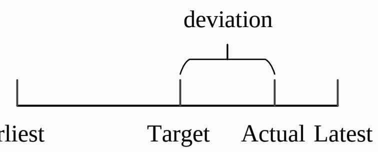
where $sep_{i,j}^{\text{EF}} = 240\text{s}$

比直接用绝对值好在哪里？

```
m.addConstrs(alpha[i] >= target[i] - t[i] for i in ALL)
m.addConstrs(beta[i] >= t[i] - target[i] for i in ALL)
m.addConstrs(alpha[i] <= target[i] - lb_t[i] for i in ALL)
m.addConstrs(beta[i] <= ub_t[i] - target[i] for i in ALL)
m.addConstrs(t[i] == target[i] - alpha[i] + beta[i] for i in ALL)
m.addConstrs((xmax[i] == alpha[i] + beta[i]) for i in ALL)

obj1 = sum(xmax[i] for i in ALL)
```

Deviation:




```
m.addConstrs((z[i, j] >= y[i, j] - U[i, j] for i in S for j in Acf), 'Z-')
m.addConstrs((z[i, j] >= U[i, j] - y[i, j] for i in S for j in Acf), 'Z-')
```


duzhuoming@nuaa.edu.cn

Runway balance

```
# obj
m.addConstrs((Zmax >= gp.quicksum(y[i, r] for i in ALL)) for r in R)
m.addConstrs((Zmin <= gp.quicksum(y[i, r] for i in ALL)) for r in R)
```

```
obj2 = Zmax - Zmin
```

$$Z_{\max} \geq \sum_{i \in F} y_{ir}$$

$$Z_{\min} \leq \sum_{i \in F} y_{ir}$$

$$\min_{t,y,z} \left[\left(Z_{\max} - Z_{\min} \right) \right.$$

Dep. Hold and arr. flt. time

- 注意 这个目标是受随机变量影响的，即受离散后的场景影响

```
obj3 = gp.quicksum(r[s, i] - x[s, i] for s in S for i in ALL)
```

$$Q(t,y,z,d) := \min_{x,r,\delta} \sum_{i \in F} \eta_i$$

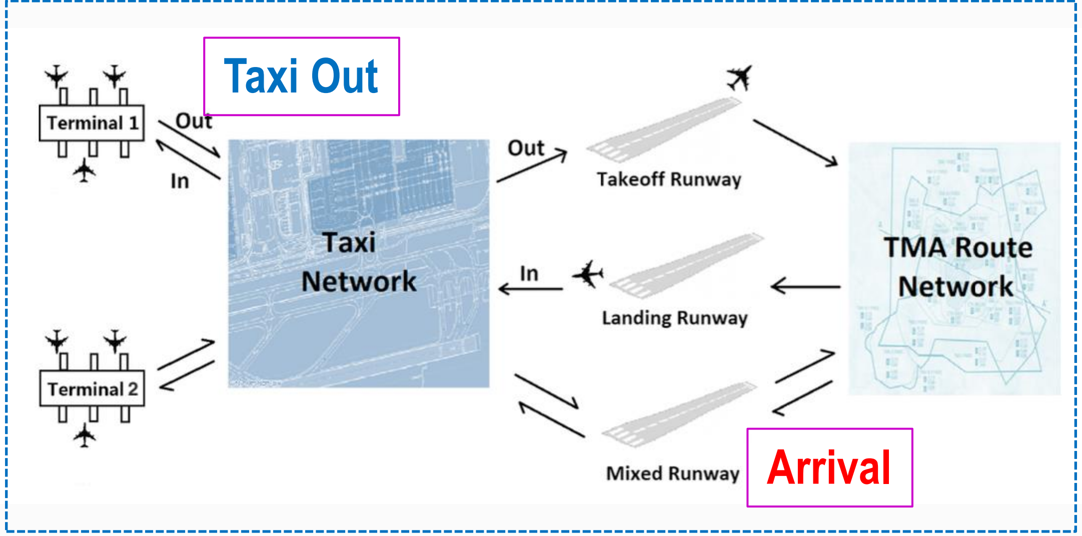
$$\eta_i \geq r_i - x_i, \forall i \in F$$

Sum of 3 obj.

```
m.setObjective(obj2 + obj1 * weight + 1 / len(S) * obj3, GRB.MINIMIZE)
```

$$\min_{t,y,z}\left[\left(Z_{\max}-Z_{\min}\right)+\Delta\right]+\sup_{\mathbb{P}\in\mathcal{F}(d,u,s)}\mathbb{E}_{\mathbb{P}}[Q(x,y,z,\delta,p,q,d)]$$

Runway Assignment under Time Uncertainty



Obj1:
Balance of RWY

Obj2:
Deviation Time

Obj3:
Delay Time

$$\min_{t,y,z}\left[\boxed{\left(Z_{\max}-Z_{\min}\right)}+\Delta\right]+\sup_{\mathbb{P}\in\mathcal{F}(d,u,s)}\mathbb{E}_{\mathbb{P}}[Q(x,y,z,\delta,p,q,d)]$$

$$E_i \le t_i \le L_i, \forall i \in F$$

$$\sum_{r\in R} y_{ir} = 1, \forall i \in F, \forall r \in R$$

$$z_{ij} = \sum_{r\in R} y_{ir}\cdot y_{jr}, \forall i, j \in F$$

$$z_{ij} = z_{ji}, \forall i, j \in F, i > j$$

$$Z_{\max} \ge \sum_{i\in F} y_{ir}$$

$$Z_{\min} \le \sum_{i\in F} y_{ir}$$

$$\boxed{\Delta \ge \left|t_i - t_i^{\text{preferred}}\right|, \forall i \in F}$$

$$Q(t,y,z,d) := \min_{x,r,\delta}\sum_{i\in F}\eta_i$$

$$x_i = t_i + \tilde{d}_i, \forall i \in F$$

$$x_i + \underline{u} \le r_i \le x_i + \bar{u}, \forall i \in A$$

$$r_j \ge r_i + z_{ij} sep_{ij} - \delta_{ji} M, \forall i, j \in F$$

$$\delta_{ij} + \delta_{ji} = 1, \forall i, j \in F, i > j$$

$$\boxed{\eta_i \ge r_i - x_i, \forall i \in F}$$

$$r_j \ge r_i + y_{i,0}\left(1 - z_{ij}\right) sep_{i,j}^{EF} - \delta_{ji} M, \forall i, j \in F, EF \in \{\text{LMN},\ldots\}$$

$$r_j \ge r_i + y_{i,1}\left(1 - z_{ij}\right) sep_{i,j}^{EF} - \delta_{ji} M, \forall i, j \in F, EF \in \{\text{P268},\ldots\}$$

Parameters	LB	UB
Entry Time Window	-60	600
Push-back Time Window	-60	600
(A) Approach Time Window	500	800
(D) Waiting Time Window	0	300
Arrival Flight time	800	1600
Taxi-out Time	600	800

$$\mathcal{F} = \left\{ \mathbb{P} \in \mathcal{P}_0(\mathbb{R}^A \times \mathbb{R}^A \times S) \left| \begin{array}{l} (\tilde{\mathbf{d}}, \tilde{\mathbf{u}}, \tilde{s}) \sim \mathbb{P} \\ \mathbb{E}_{\mathbb{P}}[\tilde{\mathbf{d}} | \tilde{s} = s] = \boldsymbol{\mu}_s \\ \mathbb{E}_{\mathbb{P}}[\tilde{\mathbf{u}} | \tilde{s} = s] = 0 \\ \mathbb{P}[(\tilde{\mathbf{d}}, \tilde{\mathbf{u}}) \in \mathcal{Z}_s | \tilde{s} = s] = 1 \\ \mathbb{P}[\tilde{s} = s] = \omega_s \end{array} \right. , \forall s \in S \right\}$$

$$\mathcal{Z}_s = \left\{ (\mathbf{d}, \mathbf{u}) \in \mathbb{R}^A \times \mathbb{R}^A : \mathbf{d} \in [\underline{\mathbf{d}}_s, \bar{\mathbf{d}}_s], (d_i - \mu_i)^2 \le 0, \forall i \in A \right\}$$

Sample Average

Sample Robust

$$\mathcal{F} = \left\{ \mathbb{P} \in \mathcal{P}_0(\mathbb{R}^A \times \mathbb{R}^A \times S) \left| \begin{array}{l} (\tilde{\mathbf{d}}, \tilde{\mathbf{u}}, \tilde{s}) \sim \mathbb{P} \\ \mathbb{E}_{\mathbb{P}}[\tilde{\mathbf{d}} | \tilde{s} = s] = \boldsymbol{\mu}_s \\ \mathbb{E}_{\mathbb{P}}[\tilde{\mathbf{u}} | \tilde{s} = s] \le \phi_s \\ \mathbb{P}[(\tilde{\mathbf{d}}, \tilde{\mathbf{u}}) \in \mathcal{Z}_s | \tilde{s} = s] = 1 \\ \mathbb{P}[\tilde{s} = s] = \omega_s \end{array} \right. , \forall s \in S \right\}$$

$$\mathcal{Z}_s = \left\{ (\mathbf{d}, \mathbf{u}) \in \mathbb{R}^A \times \mathbb{R}^A : \mathbf{d} \in [\underline{\mathbf{d}}_s, \bar{\mathbf{d}}_s], (d_i - \mu_i)^2 \le u_i, \forall i \in A \right\}$$

Thank you for your time!

Zhuoming Du,
College of Civil Aviation,
Nanjing University of Aeronautics and Astronautics (NUAA)