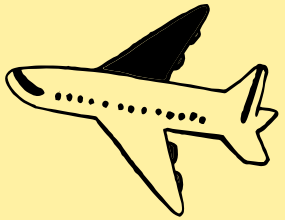


AIRCRAFT LANDING SCHEDULING PROBLEM

Javiera Villarroel
Departamento de Informática
Universidad Técnica Federico Santa María

AIRCRAFT LANDING SCHEDULING PROBLEM

- ALSP busca crear una planificación al encontrar la secuencia de aterrizaje de los aviones en sus respectivas pistas de aterrizaje que haga el mejor uso de estas **minimizando los costos**.
- Se debe decidir el tiempo y pista de aterrizaje **para cada avión.**
  - Ventana de tiempo limitada por el tiempo más temprano a más tardío de aterrizaje
  - Tiempo ideal de aterrizaje → Velocidad de crucero
  - Penalización por desviación del tiempo ideal
  - Tiempos de separación entre el aterrizaje de un avión y otro, delimitados por **consideraciones aerodinámicas**.

- **Objetivo:** Minimizar el costo total por la desviación del tiempo ideal de aterrizaje de cada avión.

Boeing 747

MODELO MATEMÁTICO

PARÁMETROS

P: *Número de aviones*
E_i: *Tiempo más temprano de aterrizaje para el avión i*
L_i: *Tiempo más tardío de aterrizaje para el avión i*
T_i: *Tiempo ideal de aterrizaje para el avión i*
g_i: *Costo de penalización por unidad de tiempo de aterrizaje antes del tiempo objetivo* T_i *para el avión i*
h_i: *Costo de penalización por unidad de tiempo de aterrizaje después del tiempo objetivo* T_i *para el avión i*
S_{ij}: *Tiempo de separación requerido entre el aterrizaje del avión i y el aterrizaje del avión j*

VARIABLES

X_i: *Tiempo de aterrizaje para el avión i*
α_i: *Unidades en que el avion i aterriza antes de* T_i
β_i: *Unidades en que el avion i aterriza después de* T_i

FUNCIÓN OBJETIVO

$$mín = \sum_{i=1}^{P} (g_i \alpha_i + h_i \beta_i)$$

MODELO MATEMÁTICO

RESTRICCIONES

- Cada avión debe aterrizar dentro de su ventana de tiempo

$$E_i \leq X_i \leq L_i$$

- El avión i debe aterrizar antes que el avión j, o el avión j debe aterrizar antes que el avión i

$$\delta_{ij} + \delta_{ji} = 1$$

- Separación de tiempos de aterrizaje

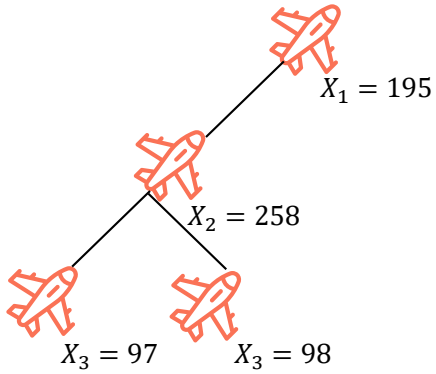
$$X_j \geq X_i + S_{ij} - M\delta_{ji}$$

RESTRICCIONES BLANDAS

Cada avión, en lo posible, debería aterrizar en su tiempo ideal de aterrizaje T_i.

PROPUESTA DE SOLUCIÓN

- **Técnica completa**
  - Garantizan obtener el óptimo global
  - Tienen asociado un árbol de búsqueda, estructura que permite construir instanciaciones
- **Backtracking con retorno guiado por conflictos**
  - Para cada variable se guarda un conjunto de conflictos $Conf(X_i)$
  - Para cada valor inconsistente se registra en el conjunto la variable más prematuramente instanciada en conflicto con el intento de instanciación.
  - Cuando no quedan valores a intentar, el punto de regreso será la variable más recientemente instanciada en $Conf(X_i)$.
  - Cuando se encuentra una solución, el salto inteligente se desactiva.



REPRESENTACIÓN

ESTRUCTURA AVIÓN

- Tiempo ideal de aterrizaje T_i
- Tiempo más temprano de aterrizaje E_i
- Tiempo más tardío de aterrizaje L_i
- Penalización por aterrizar antes g_i
- Penalización por aterrizar después h_i
- Penalización promedio
- Arreglo con separación entre aviones S_{ij}
- Tiempo instanciado X_i
- Arreglo con dominio de tiempos de aterrizaje para el avión i
- Lista enlazada de conflictos

ESTRUCTURA SOLUCIÓN

Arreglo de largo P con las variables instanciadas X_i
Costo total de la solución

REPRESENTACIÓN

ORDENES DE INSTANCIACIÓN

- **Variables**
  - Instanciar variables ordenadas por penalización promedio de mayor a menor.

- **Valores del dominio**
  - Instanciar valores del dominio según unidad de tiempo alejado del tiempo ideal de aterrizaje, de menor a mayor alejamiento.
  - Ejemplo: Considerando $E_i: 5, L_i: 9, T_i: 7$, el orden será: [7 6 8 5 9]

MOVIMIENTOS

ESTRUCTURA SALTO

- ¿Salto inteligente está activado?
- ¿Hubo un fallo?
- Variable a la cual saltar

Algorithm 1 restricciones

Require: avión, posición variable, aviones
Ensure: bool, True o False
  for i = 0; i < posición variable; i++ **do**
    if (avión.xi >= aviones[i].xi && avión.xi < aviones[i].xi + separacion con avión i) || (avión.xi <= aviones[i].xi && aviones[i].xi < avión.xi + separacion con avión i) **then**
      se agrega posición i de variable a lista de conflictos de avión
      return *False*
    end if
  end for
  return *True*

EXPERIMENTOS

- **Idea inicial de instanciación**
  - Instanciación de variables en el orden entregado (avión 1 a P)
  - Instanciación de valores de dominio desde el tiempo más temprano de aterrizaje al más tardío, es decir, $[E_i, L_i]$
- Comparación con **ideal inicial vs propuesta final** en ordenes de instanciación
- 5 instancias que se ejecutaron durante **5 minutos**
  - *airline1:* 10 aviones
  - *airline2:* 15 aviones
  - *airline3:* 20 aviones
  - *airline7:* 44 aviones
  - *airline11:* 200 aviones

EXPERIMENTOS

- Para la instancia 1 hubo **32515 saltos inteligentes** para la propuesta 1 y **32558** para la propuesta 2.

```
Costo: 3650.000000
Tiempo total de ejecución: 300.000001 [s]
Instanciaciones totales en la ejecución: 521884989
Tiempo avión 1: 129
Tiempo avión 2: 195
Tiempo avión 3: 89
Tiempo avión 4: 97
Tiempo avión 5: 110
Tiempo avión 6: 144
Tiempo avión 7: 152
Tiempo avión 8: 160
Tiempo avión 9: 168
Tiempo avión 10: 180
```

Idea inicial

```
Costo: 1150.000000
Tiempo total de ejecución: 300.000001 [s]
Instanciaciones totales en la ejecución: 492506962
Tiempo avión 3: 98
Tiempo avión 4: 106
Tiempo avión 5: 123
Tiempo avión 6: 135
Tiempo avión 7: 143
Tiempo avión 8: 151
Tiempo avión 9: 159
Tiempo avión 10: 180
Tiempo avión 1: 195
Tiempo avión 2: 258
```

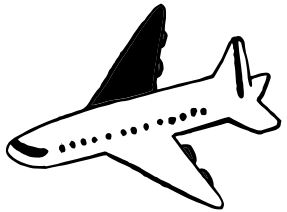
Propuesta final

EXPERIMENTOS

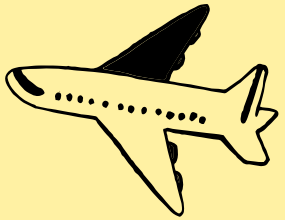
Instancia	Costo propuesta 1	Costo propuesta 2	Instanciaciones propuesta 1	Instanciaciones propuesta 2
airline1	3650	1150	521884989	492506962
airline2	8420	1720	448881681	443895932
airline3	21960	2090	446333049	424358738
airline7	3163	0	322892922	44
airline11	77103.335938	30704.789062	94756696	272325763

- Impacto de las heurísticas de ordenamiento en caso de no ejecutar en su totalidad la técnica completa.
- Ventaja del **salto inteligente versus salto cronológico.**
- Ventaja de las técnicas incompletas.

CONCLUSIONES



- El tráfico aéreo ha experimentado un gran aumento y un crecimiento sostenido, lo que implica un **aumento en la cantidad de datos** que el ALSP debe procesar.
  - Explosión combinatorial
  - Cantidad de aviones
  - Ventana de tiempo de aterrizaje
  - Memoria para almacenar el conjunto de conflictos
- Beneficio de la heurística utilizada en Backtracking con salto inteligente, sin embargo, la gran cantidad de datos hace difícil la ejecución total de las **técnicas completas.**
- Ventaja de las técnicas incompletas ante las técnicas completas.
- Mejoras en las heurísticas de ordenamiento utilizadas.



AIRCRAFT LANDING SCHEDULING PROBLEM

Javiera Villarroel
Departamento de Informática
Universidad Técnica Federico Santa María