Persistent Surveillance Problem

ALUMNO: GERMÁN FERNÁNDEZ

AYUDANTE: NICOLÁS ARMIJO

PROFESORA: MARÍA CRISTINA RIFF

Introducción

Escenario: Se representa como una grilla $m \times n$.

Estrategia: Utilización de **k** UAVs en una ventana de tiempo **T**,

los drones pueden moverse a las 8 Casillas vecinas o mantenerse.

Casillas:

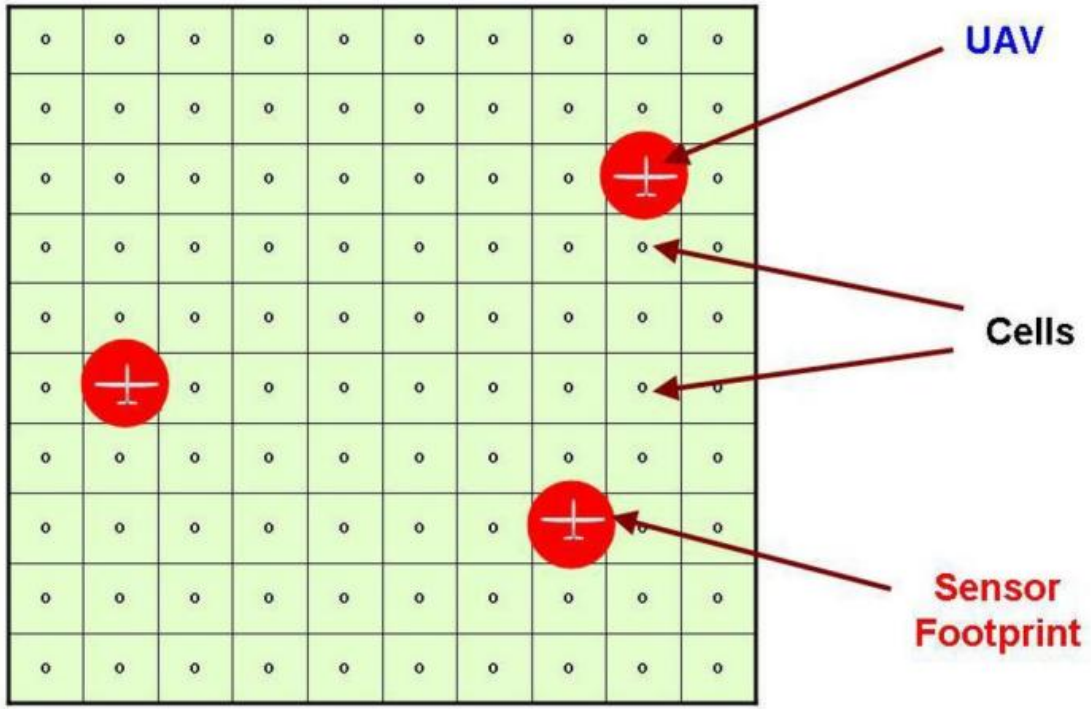
- **Bases:** Dónde inician los drones (con urgencia 0)

- **Con Urgencia:** Son aquellas que los drones deben cubrir

  y su urgencia aumenta en U cada tick

- **Obstáculos:** Son casillas en las cuales los drones no pueden sobrevolar

- **Libres:** Tienen urgencia 0.



Objetivo y restricciones

Objetivo: **minimizar la urgencia acumulada** durante T ticks.

Restricciones:

- Cada dron parte desde una base.

- No pueden pasar por obstáculos.

- No pueden coincidir dos drones en la misma celda.

- Sin restricción de batería ni señal.

Técnica: Tabú Search

- Representación, Función de Evaluación y Movimiento

- Tiene como parámetro el tamaño de la lista tabú

- **Criterio de aspiración:** Se permite aceptar un movimiento si este mejora la mejor solución global a pesar de si este es considerado tabú.

Representación

$$X = \begin{bmatrix} x_{1,1} & x_{1,2} & \cdots & x_{1,T} \\ x_{2,1} & x_{2,2} & \cdots & x_{2,T} \\ \vdots & \vdots & \ddots & \vdots \\ x_{k,1} & x_{k,2} & \cdots & x_{k,T} \end{bmatrix}$$

$x_{ij} \coloneqq Movimiento\ realizado\ por\ el\ dron\ i\ en\ el\ instante\ j.$

$i \in \{Norte, Noreste, Este,\ Sureste, Sur, Suroeste, Oeste, Noroeste, Halt\},$

$j \in \{1, 2, \ldots, T\}$

Función de Evaluación

$$\text{Minimizar } f(S) = \sum_{t=0}^{T} \sum_{(i,j)\in C} u_{ij}(t)$$

$$u_{ij}(t+1) = \begin{cases} 0, & \text{si la celda es visitada por algún dron en el tick } t, \\ u_{ij}(t) + w_{ij}, & \text{en caso contrario.} \end{cases}$$

donde:

- S: solución representada por la matriz $k \times T$ de movimientos.
- $u_{ij}(t)$: urgencia de la celda (i, j) en el instante t.
- w_{ij}: urgencia en que aumenta la celda (i, j) durante cada tick.
- C: Conjunto de celdas que deben ser monitoreadas.

Movimiento

Los movimientos se codifican de forma discreta y ordenada en sentido horario, y la generación de vecinos avanza iterando primero en el tiempo y luego por dron, realizando un único cambio por vecino dentro de la matriz S.

Solución inicial y conjunto de soluciones

Solución inicial: Es aleatoria y factible.

Conjunto de soluciones: Factibles.

Escenario en código

```
struct Celda {
    char tipo; // Tipo de celda: '.', 'O', 'U', 'B'
    int urgencia; // Tasa de actualizacin de urgencia
    vector<int> movimientos; // Direcciones posibles

    // Constructor por defecto
    Celda()
        : tipo('.'), urgencia(0), movimientos(9, 0)
    {
        movimientos[8] = 1; // Se permite permanecer en la misma celda
    }
};
```

Cada celda puede ser:

- **'.'** : celda libre.
- **'O'** : obstáculo (no transitable).
- **'U'** : celda con urgencia distinta de cero.
- **'B'** : base donde inicia un dron.

Listing 1: Estructura general de la clase Grid

```
class Grid {
private:
    // --- Atributos principales ---
    int filas, cols; // Dimensiones de la grilla
    int n_obstaculos, n_urgencias, n_bases; // Cantidades de cada tipo
    vector<pair<int,int>> obstaculos; // Coordenadas de obstculos
    vector<tuple<int,int,int>> urgencias; // (fila, columna, valor)
    vector<tuple<int,int,int>> bases; // (id, fila, columna)
    vector<vector<Celda>> grid; // Matriz de celdas
    int urgenciaTotalInicial = 0; // Suma inicial de urgencias
    unordered_map<int, int> histUrgencias; // Histograma de urgencias

public:
    // --- Constructor ---
    Grid(string instancia); // Construye la grilla desde archivo

    // --- Metodos principales ---
    void print() const; // Muestra la grilla en consola
    bool esRutaFactible(
        const vector<int>& basesIDs,
        const vector<vector<int>>& movimientos,
        int T
    ) const; // Verifica restricciones

    // --- Getters ---
    int getFilas() const;
    int getCols() const;
    int getNBases() const;
    int getUrgenciaTotalInicial() const;
    int getNUrgencias() const;
    const unordered_map<int,int>& getHistUrgencias() const;

    // --- Accesos a bases ---
    vector<pair<int,int>> getBases() const;
    pair<int,int> getBasePorID(int id) const;

    // --- Accesos a celdas ---
    char getTipo(int i, int j) const;
    int getUrgencia(int i, int j) const;
};
```

```cpp
// --- Representa un movimiento tabu individual ---
struct MovimientoTabu { int i, t, mov; };

// --- Genera vecinos factibles de una solucion dada ---
// Recorre la matriz kxT y cambia un unico movimiento por vez.
// Omite movimientos marcados como tabu y valida factibilidad con Grid.
vector<vector<vector<int>>> generarVecinos(
    const vector<vector<int>>& sol,
    const Grid& grid, int k, int T,
    const vector<int>& basesIDs,
    const deque<MovimientoTabu>& listaTabu,
    int mejorValor);

// --- Evalua una solucion sumando las urgencias acumuladas ---
// Usa la formula: w * [(N_w - N_{v,w}) * T(T+1)/2 +  t_v(t_v+1)/2]
// Considera las ultimas visitas de cada dron a cada celda urgente.
ll funcionEvaluacion(
    const vector<vector<int>>& sol,
    const Grid& grid, int k, int T,
    const vector<int>& basesIDs);

// --- Resultado de una busqueda tabu ---
// Almacena la mejor solucion y su valor asociado.
struct ResultadoTabu {
    vector<vector<int>> mejorSol;
    ll mejorValor;
};

// --- Algoritmo principal de Busqueda Tabu ---
// Explora vecinos evitando ciclos, mantiene lista tabu y aspiracion.
// Actualiza la mejor solucion encontrada durante las iteraciones.
ResultadoTabu tabuSearch(
    const Grid& grid,
    vector<vector<int>> solucionInicial,
    int iterMax, int tabuTenencia,
    int k, int T, const vector<int>& bases);

// --- Genera una solucion inicial factible ---
// Asigna movimientos aleatorios a cada dron asegurando factibilidad.
// Reintenta hasta cierto limite y, si falla, usa la solucin quieta.
vector<vector<int>> generarSolucionInicial(
    const Grid& grid, int k, int T,
    const vector<int>& basesIDs);
```

Complejidades

Función	Complejidad Temporal	Descripción
`generarSolucionInicial()`	$\mathcal{O}(I \cdot k \cdot T)$	Crea una solución inicial factible.
`esRutaFactible()`	$\mathcal{O}(k \cdot T \log k)$	Verifica límites, obstáculos y colisiones.
`funcionEvaluacion()`	$\mathcal{O}(k \cdot T + U)$	Calcula la urgencia acumulada total.
`generarVecinos()`	$\mathcal{O}(k^2 \cdot T^2)$	Explora vecinos modificando un único movimiento.
`tabuSearch()`	$\mathcal{O}(I_{\max} \cdot k^2 \cdot T^2)$	Ejecuta la búsqueda tabú completa.

Table 1: Complejidad temporal resumida de las funciones principales del sistema PSP-UAV.

Asignación drones-bases

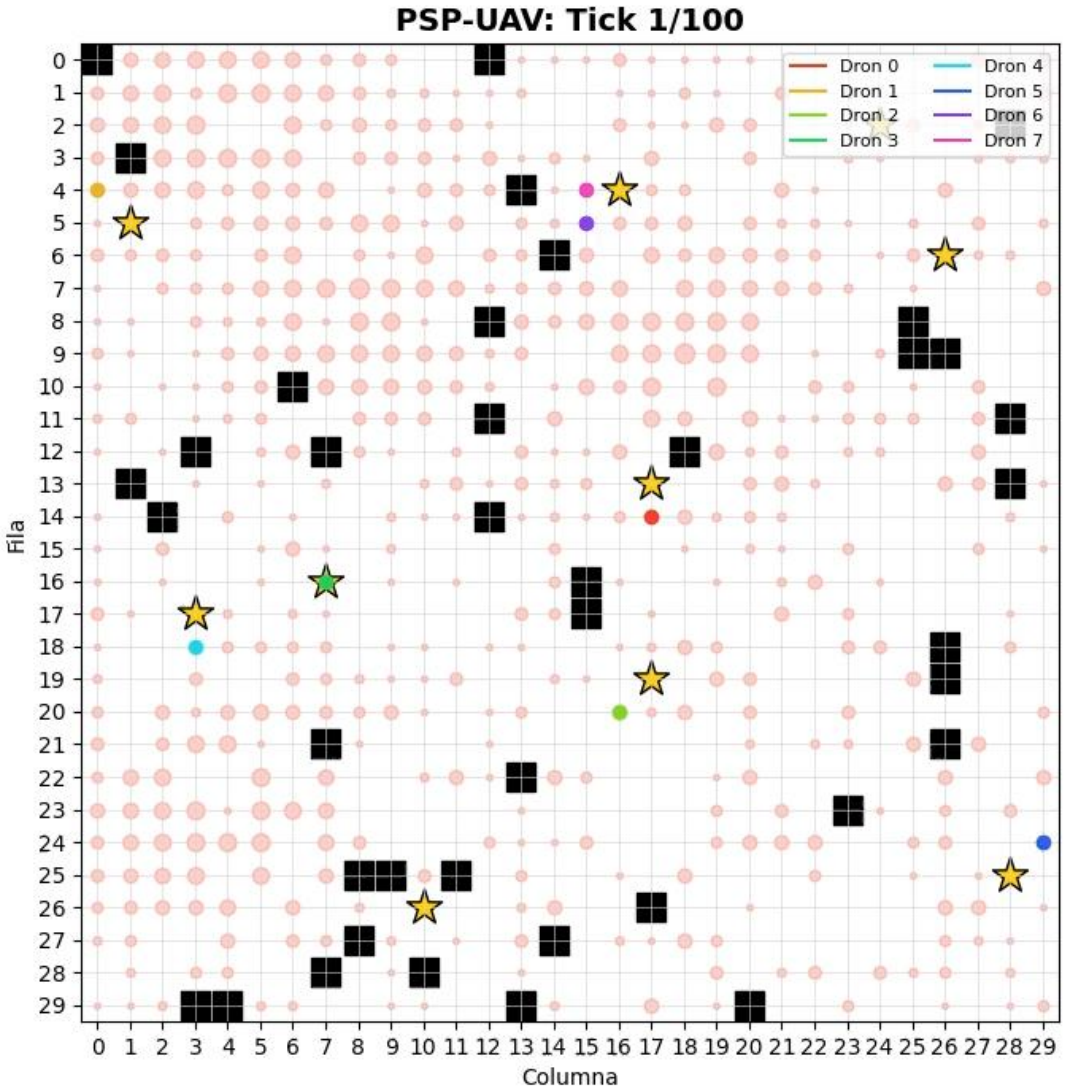
La elección de bases se maneja mediante una **búsqueda combinatoria estocástica**, donde se generan todas las combinaciones posibles drones–bases y se selecciona un **subconjunto aleatorio sin repetición** para su evaluación.

Sobre cada combinación seleccionada se aplica la **metaheurística Tabu Search**, ejecutándose de forma **paralela y asíncrona** para identificar la asignación con la menor urgencia acumulada.

Resultados

```
==================== RESULTADO FINAL ====================
Mejor asignación de drones-bases:
  Dron 0 -> Base 3
  Dron 1 -> Base 8
  Dron 2 -> Base 0
  Dron 3 -> Base 7
  Dron 4 -> Base 2
  Dron 5 -> Base 5
  Dron 6 -> Base 4
  Dron 7 -> Base 4
Costo total acumulado: 4686834
=========================================================

==================== TRAYECTORIAS ====================
```



```
================== TRAYECTORIAS ==================
Dron 0 (Base 3) - posición inicial (13,17)
Movimientos: ↓ ↓ • ↓ ↓ ↘ ↙ • ↓ ← • • ↑ ↑ ↘ → • ↗ ↙ → → → ↓ ↓ → • ↖ • ↖ ↑ ↗ ↗ • ↑ → ↑ ↖ ↙ ↙ ← ↗ ↗ • ↑ ↗ • → ↖ ↙ ← ↙ → ↗ ← • ↑ • ↓ ↗ ↗ ↑ ↗ ↗ ← ↓ ← ↑ ← ↗ → ↘ → ↑ → ↘ ← ↙ ↑ ↓ ↗ • ↗ ↙ • ↙ ← ↙ ↘ ↘ ↓ ↘ ↗ ↓ ↙ → ↘ ↙ ↖ • •
Trayectoria (fila,columna): (14,17) -> (15,17) -> (15,17) -> (16,17) -> (17,17) -> (18,18) -> (19,17) -> (19,17) -> (20,17) -> (20,16) -> (20,16) -> (20,16) -> (19,16) -> (18,16) -> (19,17) -> (19,18) -> (19,18) -> (18,19) -> (19,18) -> (19,19) -> (19,20) -> (19,21) -> (20,21) -> (21,21) -> (21,22) -> (21,22) -> (20,21) -> (20,21) -> (19,20) -> (18,20) -> (17,21) -> (16,22) -> (16,22) -> (15,22) -> (15,23) -> (14,23) -> (13,22) -> (14,21) -> (15,20) -> (15,19) -> (14,20) -> (13,21) -> (13,21) -> (12,21) -> (11,22) -> (11,22) -> (11,23) -> (10,22) -> (11,21) -> (11,20) -> (12,19) -> (12,20) -> (11,21) -> (11,20) -> (11,20) -> (10,20) -> (10,20) -> (11,20) -> (10,21) -> (9,22) -> (8,22) -> (7,23) -> (6,24) -> (6,23) -> (7,23) -> (7,22) -> (6,22) -> (6,21) -> (5,22) -> (5,23) -> (6,24) -> (6,25) -> (5,25) -> (5,26) -> (6,27) -> (6,26) -> (7,25) -> (6,25) -> (7,25) -> (6,26) -> (6,26) -> (5,27) -> (6,26) -> (6,26) -> (7,25) -> (7,24) -> (8,23) -> (9,24) -> (10,25) -> (11,25) -> (12,26) -> (11,27) -> (12,27) -> (13,26) -> (13,27) -> (14,28) -> (15,27) -> (14,26) -> (14,26) -> (14,26)
------------------------------------------------------------
Dron 1 (Base 8) - posición inicial (5,1)
Movimientos: ↖ ↓ ↑ ↑ ↑ ↗ ↑ ↓ ↗ → ↓ • ↓ ← ↓ ↗ ↘ ↘ ↓ ↓ ↓ • ↙ ↑ ↘ ↗ • ↑ ↘ ↑ ↗ → ↑ ↑ ↖ ↙ • ← • ← → • ↘ ← • ↘ → ↓ ↓ ↓ ↓ ↖ ↙ ↗ → • ↗ • ↖ ↓ ↓ ↗ ↗ ↙ ↘ ↘ → ↗ ↑ ← ← ↙ ← ↘ ↙ ← ↗ ↙ ↖ ↘ ← ↘ → ↘ ↓ • ↑ ↑ ↗ ↓ ↓ ↓ ↗ ↘ ↑ ↗ ↗ ↗ ↓ ↖
Trayectoria (fila,columna): (4,0) -> (5,0) -> (4,0) -> (3,0) -> (2,0) -> (1,1) -> (0,1) -> (1,1) -> (0,2) -> (0,3) -> (1,3) -> (1,3) -> (2,3) -> (2,2) -> (3,2) -> (2,3) -> (3,4) -> (4,5) -> (5,5) -> (6,5) -> (7,5) -> (7,5) -> (8,4) -> (7,4) -> (8,5) -> (7,6) -> (7,6) -> (6,6) -> (7,7) -> (6,7) -> (5,8) -> (5,9) -> (4,9) -> (3,9) -> (2,8) -> (3,7) -> (3,7) -> (3,6) -> (3,6) -> (3,5) -> (3,6) -> (3,6) -> (4,7) -> (4,6) -> (4,6) -> (5,7) -> (5,8) -> (6,8) -> (7,8) -> (8,8) -> (9,8) -> (8,7) -> (9,6) -> (8,7) -> (8,8) -> (8,8) -> (7,9) -> (7,9) -> (6,8) -> (7,8) -> (8,8) -> (7,9) -> (6,10) -> (7,9) -> (8,10) -> (9,11) -> (9,12) -> (8,13) -> (7,13) -> (7,12) -> (7,11) -> (8,10) -> (8,9) -> (9,10) -> (10,9) -> (10,8) -> (9,9) -> (10,8) -> (9,7) -> (10,8) -> (10,7) -> (11,8) -> (11,9) -> (12,10) -> (13,10) -> (13,10) -> (12,10) -> (11,10) -> (10,11) -> (11,11) -> (12,11) -> (13,11) -> (12,12) -> (13,13) -> (12,13) -> (11,14) -> (10,15) -> (9,16) -> (10,16) -> (9,15)
------------------------------------------------------------
Dron 2 (Base 0) - posición inicial (19,17)
Movimientos: ↙ ↗ ↑ ↗ ↑ ↓ ↗ • ↙ ↗ ↖ ↓ ↙ ↖ ↖ → ↑ ↘ ↙ ↗ ↗ ↙ ↙ ↗ ↖ ↘ ↖ ↗ ↑ ↑ ↑ ↑ ↘ ↑ ↖ ↑ • ← ↗ ← ↓ → ← → ↘ → ↘ ↗ ↙ ↘ ↙ ↖ ↖ ↑ → • ↗ • ↘ ← ↙ ↖ ↑ ↑ → • ↖ ↘ → ↑ ↑ → ← ← ← ← ↑ ↑ ↑ ↖ → • → → ↓ → ↓ ↑ ↑ → → ↙ ↘ ↘ ↙ ↙ ↙ ↘ ↖ ↖
Trayectoria (fila,columna): (20,16) -> (19,17) -> (18,17) -> (17,18) -> (16,18) -> (17,18) -> (16,19) -> (16,19) -> (17,18) -> (16,19) -> (15,18) -> (16,18) -> (17,17) -> (16,16) -> (15,15) -> (15,16) -> (14,16) -> (15,17) -> (16,16) -> (15,17) -> (14,18) -> (15,17) -> (16,16) -> (15,17) -> (14,16) -> (15,17) -> (14,16) -> (13,17) -> (12,17) -> (11,17) -> (10,17) -> (9,17) -> (10,18) -> (9,18) -> (8,17) -> (7,17) -> (7,17) -> (7,16) -> (6,17) -> (6,16) -> (7,16) -> (7,17) -> (7,16) -> (7,17) -> (8,18) -> (8,19) -> (9,20) -> (8,21) -> (9,20) -> (10,21) -> (11,20) -> (10,19) -> (9,18) -> (8,18) -> (8,19) -> (8,19) -> (7,20) -> (7,20) -> (8,21) -> (8,20) -> (9,19) -> (8,18) -> (7,18) -> (6,18) -> (6,19) -> (6,19) -> (5,18) -> (6,19) -> (6,20) -> (5,20) -> (4,20) -> (4,21) -> (4,20) -> (4,19) -> (4,18) -> (4,17) -> (3,17) -> (2,17) -> (1,17) -> (0,16) -> (0,17) -> (0,17) -> (0,18) -> (0,19) -> (1,19) -> (1,20) -> (2,20) -> (1,20) -> (0,20) -> (0,21) -> (0,22) -> (1,21) -> (2,22) -> (3,23) -> (4,22) -> (5,21) -> (6,20) -> (7,21) -> (6,20) -> (5,19)
------------------------------------------------------------
Dron 3 (Base 7) - posición inicial (16,7)
Movimientos: • ← ↘ ↘ ↑ ↙ ← • ↑ ↓ ↑ ↓ • ↙ ↓ ↙ ↖ ↖ • ← • ← ↘ ↙ ↘ ↑ → ↗ ← ← ↘ → ↙ ← ↗ ↓ ← → ↑ • ↑ ↓ • ← ↖ ↘ ↘ ↙ ← • ↓ ↗ • → ↘ • ↘ ↗ ↑ ↙ • ↑ ↓ → ↓ ← • ↘ ↖ → • ↓ ↙ ↑ ↖ ↙ → → ↖ ↖ ↖ ↘ ↖ • • ↓ ↓ ↙ ↘ ↑ ↓ ↓ ↗ → ↘ ← ↙ ← ← ↑
Trayectoria (fila,columna): (16,7) -> (16,6) -> (17,7) -> (18,8) -> (17,8) -> (18,7) -> (18,6) -> (18,6) -> (17,6) -> (18,6) -> (17,6) -> (18,6) -> (18,6) -> (19,5) -> (20,5) -> (21,4) -> (20,3) -> (19,2) -> (19,2) -> (19,1) -> (19,1) -> (19,0) -> (20,1) -> (21,0) -> (22,1) -> (21,1) -> (21,2) -> (20,3) -> (20,2) -> (20,1) -> (21,2) -> (21,3) -> (22,2) -> (22,1) -> (21,2) -> (22,2) -> (22,1) -> (22,2) -> (21,2) -> (21,2) -> (20,2) -> (21,2) -> (21,2) -> (21,1) -> (20,0) -> (21,1) -> (22,2) -> (23,1) -> (23,0) -> (23,0) -> (24,0) -> (23,1) -> (23,1) -> (23,2) -> (24,3) -> (24,3) -> (25,4) -> (24,5) -> (23,5) -> (24,4) -> (24,4) -> (23,4) -> (24,4) -> (24,5) -> (25,5) -> (25,4) -> (25,4) -> (26,5) -> (25,4) -> (25,5) -> (25,5) -> (26,5) -> (27,4) -> (26,4) -> (25,3) -> (26,2) -> (26,3) -> (26,4) -> (25,3) -> (24,2) -> (23,1) -> (24,2) -> (23,1) -> (23,1) -> (23,1) -> (24,1) -> (25,1) -> (26,0) -> (27,1) -> (26,1) -> (27,1) -> (28,1) -> (27,2) -> (27,3) -> (28,4) -> (28,3) -> (29,2) -> (29,1) -> (29,0) -> (28,0)
------------------------------------------------------------
Dron 4 (Base 2) - posición inicial (17,3)
Movimientos: ↓ ↖ ↗ ↙ ↓ ↗ ↓ ↑ ↙ ↘ ↗ ↖ ↓ → ← ↗ ↘ ↘ ↑ → ↘ ↘ • ↓ • ↘ ↗ ↗ ↘ ↗ → ↙ → ↑ ← ↘ ← ↖ ↓ ↘ ↓ ↑ ↖ ↙ ↙ → ← ↑ ↘ ↓ ↖ ↓ ↗ ↑ ↑ ↗ ↑ ← ↓ ↘ ↗ ↑ ↖ → ↑ ← ↙ • ↙ ↑ ↗ ↖ ↖ • ← ← ↑ ↙ ← • ← ↑ ← ← ↗ → • ↑ ↖ • ↑ ↓ → ↑ ↖ ↖ ↖ ↑ ↘ ↑
Trayectoria (fila,columna): (18,3) -> (17,2) -> (16,3) -> (17,2) -> (18,2) -> (17,3) -> (18,3) -> (17,3) -> (18,2) -> (19,3) -> (18,4) -> (17,3) -> (18,3) -> (18,4) -> (18,3) -> (17,4) -> (18,5) -> (19,6) -> (18,6) -> (18,7) -> (19,8) -> (20,9) -> (20,9) -> (21,9) -> (21,9) -> (22,10) -> (21,11) -> (20,12) -> (21,13) -> (20,14) -> (20,15) -> (21,14) -> (21,15) -> (20,15) -> (20,14) -> (21,15) -> (21,14) -> (20,13) -> (21,13) -> (22,14) -> (23,14) -> (22,14) -> (21,13) -> (22,12) -> (23,11) -> (23,12) -> (23,11) -> (22,11) -> (23,12) -> (24,12) -> (23,11) -> (24,11) -> (23,12) -> (22,12) -> (21,12) -> (20,13) -> (19,13) -> (19,12) -> (20,12) -> (21,13) -> (20,14) -> (19,14) -> (18,13) -> (18,14) -> (17,14) -> (17,13) -> (18,12) -> (18,12) -> (19,11) -> (18,11) -> (17,12) -> (16,11) -> (15,10) -> (15,10) -> (15,9) -> (15,8) -> (14,8) -> (15,7) -> (15,6) -> (15,6) -> (15,5) -> (14,5) -> (14,4) -> (14,3) -> (13,4) -> (13,5) -> (13,5) -> (12,5) -> (11,4) -> (11,4) -> (10,4) -> (11,4) -> (11,5) -> (10,5) -> (9,4) -> (8,3) -> (7,2) -> (6,2) -> (7,3) -> (6,3)
------------------------------------------------------------
Dron 5 (Base 5) - posición inicial (25,28)
Movimientos: ↗ ↖ ↖ ↙ ↙ ↑ ↘ ↙ ↖ ← ↙ ↗ ↘ ↘ ↑ ↑ ↙ ↓ ↗ ↗ • ← ↑ ↙ ↗ ↑ → • ↗ ↘ ↑ ↘ ← ↑ • → ← • ↗ ↙ • ↖ ↑ ↘ ↓ ← ↗ → ↗ ← ↙ ↘ • → ↙ ↓ ↑ ↑ ↖ → ← • ↙ ↖ ← ← ← → ↙ ↖ ↘ ↙ → ↙ ↑ ↙ ↓ ↘ → ↑ → • ↑ ↖ → ↙ ↙ ↓ ↙ ↖ ↘ ↙ ↘ ↑ ↙ ↙ ↑ ←
Trayectoria (fila,columna): (24,29) -> (23,28) -> (22,27) -> (23,26) -> (24,25) -> (23,25) -> (24,26) -> (25,25) -> (24,24) -> (24,23) -> (25,22) -> (24,23) -> (25,24) -> (26,25) -> (25,25) -> (24,25) -> (25,24) -> (26,24) -> (25,25) -> (24,26) -> (24,26) -> (24,25) -> (23,25) -> (24,24) -> (23,25) -> (22,25) -> (22,26) -> (21,27) -> (22,28) -> (21,28) -> (22,29) -> (22,28) -> (21,28) -> (21,28) -> (21,29) -> (21,28) -> (21,28) -> (20,29) -> (21,28) -> (21,28) -> (20,27) -> (19,27) -> (20,28) -> (21,28) -> (21,27) -> (20,28) -> (20,29) -> (20,28) -> (19,29) -> (19,28) -> (19,27) -> (20,26) -> (21,27) -> (21,27) -> (21,28) -> (22,27) -> (23,27) -> (22,27) -> (21,27) -> (20,26) -> (20,27) -> (20,26) -> (20,26) -> (21,25) -> (20,24) -> (20,23) -> (20,22) -> (20,21) -> (20,22) -> (21,21) -> (20,20) -> (21,21) -> (22,20) -> (22,21) -> (23,20) -> (22,20) -> (23,19) -> (24,19) -> (25,20) -> (25,21) -> (24,21) -> (24,22) -> (24,22) -> (23,22) -> (22,21) -> (22,22) -> (23,21) -> (24,20) -> (25,20) -> (26,19) -> (25,18) -> (26,19) -> (27,18) -> (28,19) -> (27,19) -> (28,18) -> (29,17) -> (28,17) -> (28,16)
------------------------------------------------------------
Dron 6 (Base 4) - posición inicial (4,16)
Movimientos: ↙ ↓ ↗ • ↑ ↘ → ↘ → ↙ • ↓ ↙ ↖ ↑ ↙ • ← ↖ → • ↙ ↓ • ↗ ↙ ← ↑ • ↙ ← • ↙ ↓ ↖ ↙ • ↘ ↓ → ↖ ↓ ← ↘ ↘ ↙ ↑ ↓ ↘ ↑ ↘ ← • • ↑ ↙ ↓ ↘ ↓ ↑ ↖ ↙ ← ↘ ↗ → ↘ ← • ↙ ↖ ← ↗ ↓ ↖ ↙ ↙ ↓ ↘ ↖ ↘ ↘ • ↗ ↙ ↓ ↓ ↓ ← → ↙ • ↖ ↗ ↗ ↖ ↗ ← ↖
Trayectoria (fila,columna): (5,15) -> (6,15) -> (5,16) -> (5,16) -> (4,16) -> (5,17) -> (5,18) -> (6,19) -> (6,20) -> (7,19) -> (7,19) -> (8,19) -> (9,18) -> (8,17) -> (7,17) -> (8,16) -> (8,16) -> (8,15) -> (7,14) -> (7,15) -> (7,15) -> (8,14) -> (9,14) -> (9,14) -> (8,15) -> (9,14) -> (9,13) -> (8,13) -> (8,13) -> (9,12) -> (9,11) -> (9,11) -> (10,10) -> (11,10) -> (10,9) -> (11,8) -> (11,8) -> (12,9) -> (13,9) -> (13,10) -> (12,9) -> (13,9) -> (13,8) -> (14,9) -> (15,10) -> (16,9) -> (15,9) -> (16,9) -> (17,10) -> (16,10) -> (17,11) -> (17,10) -> (17,10) -> (17,10) -> (16,10) -> (17,9) -> (18,9) -> (19,10) -> (20,10) -> (19,10) -> (18,9) -> (19,8) -> (19,7) -> (20,8) -> (19,9) -> (19,10) -> (20,11) -> (20,10) -> (20,10) -> (21,9) -> (20,8) -> (20,7) -> (19,8) -> (20,8) -> (19,7) -> (20,6) -> (21,5) -> (22,5) -> (23,6) -> (22,5) -> (23,6) -> (24,7) -> (24,7) -> (23,8) -> (24,7) -> (25,7) -> (26,7) -> (27,7) -> (27,6) -> (27,7) -> (28,6) -> (28,6) -> (27,5) -> (26,6) -> (25,7) -> (24,8) -> (23,7) -> (22,8) -> (22,7) -> (21,6)
------------------------------------------------------------
Dron 7 (Base 4) - posición inicial (4,16)
Movimientos: ← • ↖ ↖ ↑ ← ↙ ↑ ↙ • ← ↖ ↙ ↑ ↙ ↗ • ↘ ↓ ↘ → ↗ ↙ ↗ → ↑ ↓ ← ↗ • ↗ ← ↗ → • • ↘ ↑ → ↓ ↗ ↗ ↙ ↘ • ↑ ↙ → ↓ ↑ ↓ ↑ ↙ ↙ → ↗ → ↑ ↓ • • ↖ • ↘ → ↗ ↖ ↘ → • ↖ ↘ ↗ ↓ ↗ ↙ ↖ ↗ ↘ ↓ → • → • → ↗ ↙ ↗ ↑ ← • ↓ ↙ ↘ ↙ → ↗ → ↓ •
Trayectoria (fila,columna): (4,15) -> (4,15) -> (3,14) -> (2,13) -> (1,13) -> (1,12) -> (2,11) -> (1,11) -> (2,10) -> (2,10) -> (2,9) -> (1,8) -> (2,7) -> (1,7) -> (2,6) -> (1,7) -> (1,7) -> (2,8) -> (3,8) -> (4,9) -> (4,10) -> (4,11) -> (5,10) -> (4,11) -> (4,12) -> (3,12) -> (4,12) -> (4,11) -> (3,12) -> (3,12) -> (2,13) -> (2,12) -> (1,13) -> (1,14) -> (1,14) -> (1,14) -> (2,15) -> (1,15) -> (1,16) -> (2,16) -> (1,17) -> (0,18) -> (1,17) -> (2,18) -> (2,18) -> (1,18) -> (2,17) -> (2,18) -> (3,18) -> (2,18) -> (3,18) -> (2,18) -> (3,17) -> (4,16) -> (4,17) -> (3,18) -> (3,19) -> (2,19) -> (3,19) -> (3,19) -> (3,19) -> (2,18) -> (2,18) -> (3,19) -> (3,20) -> (2,21) -> (1,20) -> (2,21) -> (2,22) -> (2,22) -> (1,21) -> (2,22) -> (1,23) -> (2,23) -> (1,24) -> (2,23) -> (1,22) -> (0,23) -> (1,24) -> (2,24) -> (2,25) -> (2,25) -> (2,26) -> (2,26) -> (2,27) -> (1,28) -> (2,27) -> (1,28) -> (0,28) -> (0,27) -> (0,27) -> (1,27) -> (2,26) -> (3,27) -> (4,26) -> (4,27) -> (3,28) -> (3,29) -> (4,29) -> (4,29)
```

Mejoras posibles

- **generarVecinos():** Debemos buscar reducir su complejidad viendo posible trabajar en conjunto de soluciones infactibles o analizando la factibilidad dinámicamente.

- En caso de que no se consiga una solución inicial factible, trabajar con un greedy que nos consiga una trayectoría factible para cada dron.

- Trabajar con la solución inicial en donde todas los movimientos inician en Halt tiene errores.

- Mejorar la visualización de la solución, corrigiendo ciertos errores, haciendo que los drones partan de las bases en el vídeo.

- Que el Código muestre cuánto se demora conseguir la mejor solución a través del Tabú Search.