

Expertises digitale et logicielle

# ITR Project : Ordonnancement Off-line pour la Protection des Poules

**Auteur:** Alan Altruy

---

*Cours*

S-INFO-111 : Informatique Temps-Réel

*Professeur*

Gauvain DEVILLEZ

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>1</b>
<b>2</b>	<b>Ordonnancement Off-line</b>	<b>1</b>
2.1	Paramètres des tâches . . . . .	1
2.1.1	Tâche Renard ( $T_{renard}$ ) . . . . .	1
2.1.2	Tâche Aigle ( $T_{aigle}$ ) . . . . .	1
2.2	Hyperpériode . . . . .	1
2.3	Faisabilité . . . . .	1
2.4	Diagramme de Gantt . . . . .	1
<b>3</b>	<b>Implémentation et contraintes</b>	<b>2</b>
<b>4</b>	<b>Conclusion</b>	<b>2</b>

## 1 Introduction

Ce projet vise à développer un système automatique de protection des poules contre deux menaces principales : le renard et l'aigle. Chaque menace attaque périodiquement et doit être détectée et chassée à l'aide d'un senseur et d'une alarme. Le senseur et l'alarme ne peuvent être utilisés simultanément et ne traitent qu'une direction à la fois, ce qui impose un fonctionnement mono-processeur.

## 2 Ordonnancement Off-line

### 2.1 Paramètres des tâches

- $\text{STEP\_TIME} = \text{FOX\_TIME} / 8$  (unité de temps pour l'ordonnancement)

#### 2.1.1 Tâche Renard ( $T_{renard}$ )

- Période :  $\psi_{renard} = 8$  unités
- Durée pire cas :  $\tau_{renard} = 4$  unités
- Échéance :  $\delta_{renard} = 8$  unités

Étapes (pire cas) :

- (a) Sense direction 1 (north)
- (b) Sense direction 2 (south)
- (c) Sense direction 3 (east)
- (d) Alarm pour chasser le renard

#### 2.1.2 Tâche Aigle ( $T_{aigle}$ )

- Période :  $\psi_{aigle} = 4$  unités
- Durée pire cas :  $\tau_{aigle} = 2$  unités
- Échéance :  $\delta_{aigle} = 4$  unités

Étapes (pire cas) :

- (a) Sense direction (above)
- (b) Alarm pour chasser l'aigle

### 2.2 Hyperpériode

$$\text{Hyperpériode} = \text{PPCM}(\psi_{renard}, \psi_{aigle}) = 8 \text{ unités}$$

### 2.3 Faisabilité

$$U = \frac{\tau_{renard}}{\psi_{renard}} + \frac{\tau_{aigle}}{\psi_{aigle}} = 1.0$$

Le processeur est pleinement occupé, donc l'ordonnancement est possible mais sans marge.

### 2.4 Diagramme de Gantt

- Algorithme : Rate Monotonic (priorité à l'Aigle)
- Prémption : le Renard est interrompu si l'Aigle doit s'exécuter

Cycle par cycle :

- $t = 0-2$  : Aigle s'exécute
- $t = 2-4$  : Renard s'exécute

- $t = 4-6$  : Aigle reprend
- $t = 6-8$  : Renard termine

0	1	2	3	4	5	6	7
Aigle	Aigle	Renard	Renard	Aigle	Aigle	Renard	Renard

### 3 Implémentation et contraintes

- L'implémentation doit respecter l'ordonnancement off-line.
- Les fonctions `sense` et `sound_alarm` sont mutuellement exclusives et mono-direction.
- Une troisième tâche peut être déclenchée pour remplacer les poules perdues en utilisant `get_chickens` et `add_chicken`. Cette tâche ne perturbe pas l'ordonnancement principal.

### 4 Conclusion

Toutes les tâches peuvent être exécutées dans leurs délais respectifs avec un taux d'utilisation de 100%. La préemption et les contraintes mono-direction/mutex sont respectées. La tâche additionnelle de remplacement des poules profite des temps libres sans affecter l'ordonnancement.