

## IF505 Simulation par événements discrets



Histoire d'ordinateurs

IF505 Simulation par  
événements discrets

Classe - 2

### Un Ordinateur Antique

Un ordinateur un peu antique opère en traitement par lots et en multi-tâches. Cet ordinateur est configuré avec  $k = 4$  initiateurs (lots), chacun pouvant démarrer une tâche (max. de 4 tâches simultanées). Dans chaque initiateurs (lots), chaque tâche possède son propre temps d'exécution (distribution empirique). Il y a trois classes de priorité, les tâches de classe 1 étant de plus haute priorité, et les tâches de classe 3 de plus basse priorité.

Les tâches sont emmagasinées dans une ou plusieurs files d'attente (dépend de l'implantation) en fonction de la priorité et selon le plus petit temps d'exécution. Le temps de service (exécution) pour une tâche de classe  $i$  est distribué selon une distribution empirique continue. Chaque classe possède son processus d'arrivée, i.e. le temps d'inter-arrivée pour les tâches de classe  $i$  est distribué exponentiellement avec une moyenne  $r(i)$  minutes.

Classe $i$	Moyenne $r(i)$
1	0.2
2	1.6
3	5.4



ÉCOLE  
POLYTECHNIQUE  
MONTREAL

## Un Ordinateur Antique

Le temps d'exécution pour chacune des tâches dans les différentes classes suivent une **distribution empirique** et ces données sont conservées dans un fichier (format SIMSCRIPT):

```
Classe 1 0.25 0.8 0.30 2.0 0.15 2.5 0.20 4.0 0.10 6.0 *
Classe 2 0.15 1.0 0.30 2.5 0.35 4.5 0.15 6.0 0.05 8.0 *
Classe 3 0.30 10.0 0.40 12.0 0.30 14.0 *
```

Le processeur (CPU) pour l'exécution d'au maximum  $k = 4$  tâches simultanément utilise la technique du tourniquet [*time-slicing*] en attribuant une tranche de temps à chacune des tâches en plus de changement du contexte [*overhead*]. Les tâches de plus haute priorité réquisitionnent le processeur avant les tâches de priorité inférieure.

Lors de l'exécution, étant donné les priorités, les tâches de plus haute priorité se verront attribuer toutes les tranches de temps jusqu'à un maximum de  $t=6$  tranches; ensuite, leur priorité est réduite d'une unité. Ensuite; les tâches de priorité intermédiaire se voient attribuer toutes les tranches de temps jusqu'à un maximum de  $t=4$  tranches et leur priorité est réduite d'une unité. Une tâche ne peut avoir une priorité négative.

## Un Ordinateur Antique

Tous les paramètres (variables) pour lesquels des valeurs ne sont pas spécifiées dans cette description sont considérés comme des valeurs obtenues d'un fichier (SIMU12).

Initialement, le système est vide. La simulation doit durer exactement 720 minutes. Écrire un modèle SIMSCRIPT qui

- + pour chaque classe, calcule les temps d'attente minimum, maximum, et moyen,
- + calcule le pourcentage d'utilisation de l'UCT (CPU) et le temps moyen d'exécution des tâches par classe de tâche.

Utilisez des suites différentes pour les arrivées et les temps d'exécution.

N.B. Dans la solution de ce problème, on demande de mettre l'accent sur la structure du programme informatique, i.e. identifiez en premier les processus, les ressources, les entités, les statistiques, etc... Les déclarations de variables ne sont pas les parties les plus importantes ainsi que les entrées/sorties pour les paramètres et les résultats.

## Un Ordinateur Antique

+ Stockage des données (En Simscript)

Classe(i) r(i) Pr(i) Tex(i) Tatt(i) Tserv(i)

Distribution empirique pour chaque classe

+ Evénements

Fin de la simulation  
Imprimer les résultats

0 min 720 min Temps

Durée de la simulation



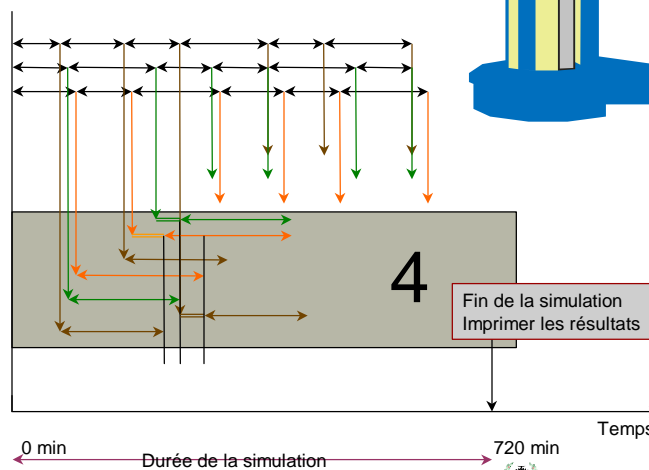
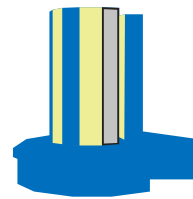
## Un Ordinateur Antique

+ Processus

Classe	
1	Exp(r(1))
2	Exp(r(2))
3	Exp(r(3))
.	

+ Ressource

Ressource  
CPU (1)  
Capacité (4)



## Un Ordinateur Antique

+ Mesures (Statistiques)

4 Temps Attente

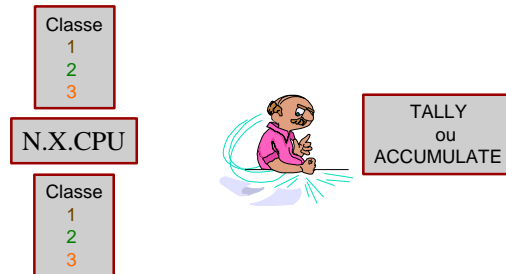
\_\_Min

\_\_Max

\_\_Moyenne

4 Utilisation CPU

4 Temps Exécution



## Un Ordinateur Antique

Preamble

Normally mode is undefined

Last column is 132

Permanent Entities.....

Every Classe has	a Cl.Numero,	" Numéro de la classe
	a Cl.Interarrivee,	" Moyenne des interarrivées
	a Cl.Priorite,	" Priorité de chaque classe
	a Cl.TempsExecution,	" Temps exécution
	a Cl.TempsAttente,	" Temps d'attente
	and a T.Execution.F	random linear variable

Define Cl.Numero, Cl.Priorite as integer variable

Define Cl.Interarrivee, Cl.TempsExecution, Cl.TempsAttente as double variable

Define T.Execution.F as a double stream 10 variable

## Un Ordinateur Antique

### Resources .....

Every Lot has a Lt.Capacite  
 Define Lt.Capacite as an integer variable

Every CPU has a CPU.Overhead, " Overhead pour changement de contexte  
 a CPU.Tranche " Tranche de temps allouée ... chaque tâche  
 Define CPU.Overhead, CPU.Tranche as double variable

### Processes.....

Every Arrivee has a Ar.NumeroClasse " Processus d'arrivée pour une classe  
 Define Ar.NumeroClasse as an integer variable

Every Tache has a Tk.NumeroClasse, " Numéro de la classe  
 a Tk.TempsArrivee, " Temps d'arrivée de la tâche  
 a Tk.TempsExecution, " Temps d'exécution de la tâche  
 a Tk.Priorite " Priorité de la tâche

Define Tk.NumeroClasse, Tk.Priorite as integer variable

Define Tk.TempsArrivee, Tk.TempsExecution as double variable

## Un Ordinateur Antique

Event include Fin.Simulation

### " Les prototypes de fonctions

Define AjusterPriorite as a routine giving 2 arguments  
 yielding 2 arguments

### " Les Statistiques

Tally T.AttMoy as the mean,  
 T.AttMax as the maximum,  
 T.AttMin as the minimum of Cl.TempsAttente

Accumulate Utilisation.CPU as the mean of N.X.CPU

Tally T.ExMoy as the mean of Cl.TempsExecution

### " Déclarations globales

Define Duree.Simulation as a double variable

Define Secondes.V as a double variable

Define .Seconds to mean / 60.0 Minutes

Define .Donnee to mean 10

Define Forever to mean while ( 1 = 1)

End " Preamble

## Un Ordinateur Antique

### Main

Call Initialiser

for each Classe

do

    Activate an Arrivee ( Classe ) now

loop

Schedule a Fin.Simulation in Duree.Simulation minutes

Start Simulation

End "Main

## Un Ordinateur Antique

### Routine Initialiser

use .Donnee for input

Let Secondes.V = 60.0

Read N.Classe

Start new input record

Create all Classe

for each Classe

do

    Read Cl.Numero(Classe), Cl.Priorite (Classe)

    Read T.Execution.F(Classe)

loop

Let N.CPU = 1

Create every CPU

Let U.CPU(1) = 1.0

Read CPU.Overhead(1), CPU.Tranche(1)

Let N.Lot = 1

Create every Lot

Read Lt.Capacite (1)

Let U.Lot = Lt.Capacite(1)

Read Duree.Simulation

End " Initialiser

## Un Ordinateur Antique

```
Event Fin.Simulation
  Call Imprimer.Resultats
  Stop
End " Fin.Simulation

Function AjusterPriorite      giving      Nb.Tranche, PR
                              yielding    Nb.Tranche1, PR1

  Define Nb.Tranche, PR as integer variable
  Define Nb.Tranche1, PR1 as integer variable

  Add 1 to Nb.Tranche
  IF Nb.Tranche = 6 or Nb.Tranche = 10,
    Subtract 1 from PR
  Endif
  Let PR1 = Max.F ( 1, PR)
  Let Nb.Tranche1 = Nb.Tranche
End " AjusterPriorite
```

## Un Ordinateur Antique

```
Process Arrivee ( PourClasse)
  Define PourClasse      as an integer variable
  Define Nb.Tranche      as an integer variable

  Forever
  do
    Wait Exponential.F ( Cl.Interarrivee ( PourClasse), PourClasse) Minutes
    Activate a Tache ( PourClasse, TIME.V, T.Execution.F(PourClasse),
                     N.Classe - PourClasse + 1) Now
  loop

  End "Arrivee
```

## Un Ordinateur Antique

```

Process Tache ( Class, TempsArrivee, TempsExecution, PR)
Define Class, PR      as integer variable
Define TempsArrivee, TempsExecution  as double variable

Request 1 Lot(1) with priority PR      " K tâches à la fois dans le CPU
Let Cl.TempsAttente (Class) = TIME.V - TempsArrivee " Calculer le temps d'attente

While (TempsExecution > CPU.Tranche(1)) " Tant que l'exécution n'est pas terminée
do
  Request 1 CPU(1) with priority PR      " Le CPU alloue une tranche de temps
  work CPU.Overhead(1) + CPU.Tranche(1) .Seconds " à la tâche
  Call AjusterPriorite ( Nb.Tranche, PR, Nb.Tranche, PR)
  TempsExecution = TempsExecution - CPU.Tranche(1)
  " Diminuer le temps d'exécution de la tranche
  Relinquish 1 CPU(1)      " Laisser le CPU à une autre tâche
loop

Request 1 CPU(1) with priority PR " Terminer l'exécution de cette tâche
work (CPU.Overhead(1) + CPU.Tranche(1)) .Seconds
Relinquish 1 CPU(1)

Relinquish 1 Lot(1)      " Laisser la place à une autre tâche
Let Cl.TempsExecution(UneClasse) = TempsExecution " Prendre des statistiques
End " Tache
    
```

## Un Ordinateur Antique

```

Routine Imprimer.Resultats
Print 2 lines with Hour.F(TIME.V), Minute.F(TIME.V),
      Mod.F(TIME.V * Hours.V * Minutes.V * Secondes.V, 60),
      Utilisation.CPU      Thus
La simulation se termine ... **.*.*
Le pourcentage d'utilisation du CPU est ***.**** %
Print 3 lines thus
Classe Priorit,      Temps Attente      Temps Ex,cution
      moyen      Min      Max      moyen
-----
For each Classe
do
  Print 1 line with Classe, Cl.Priorite(Classe) , T.AttMoy(Classe), T.AttMin(Classe),
  T.AttMax(Classe), T.ExMoy(Classe) thus
  *****      *****      ***** *      ***** *      ***** *      ***** *
loop
End " Imprimer.Resultats
    
```

## Un Ordinateur Antique

) Fichiers

! Define .Donnee to mean 10 (Preamble)

! use .Donnee for input (Initialiser)



) SIMU10

3 " Nombre de classes

1 3 0.25 0.8 0.30 2.0 0.15 2.5 0.20 4.0 0.10 6.0 \*

2 2 0.15 1.0 0.30 2.5 0.35 4.5 0.15 6.0 0.05 8.0 \*

3 1 0.30 10.0 0.40 12.0 0.30 14.0 \*

0.015 0.1

4

720

