IFT 3245
Simulation et modèles

Fabian Bastin
DIRO
Université de Montréal

Automne 2016

Approche par événements

Hypothèse: système dans lequel les variables d'état ne peuvent changer qu'en un nombre dénombrable de points dans le temps.

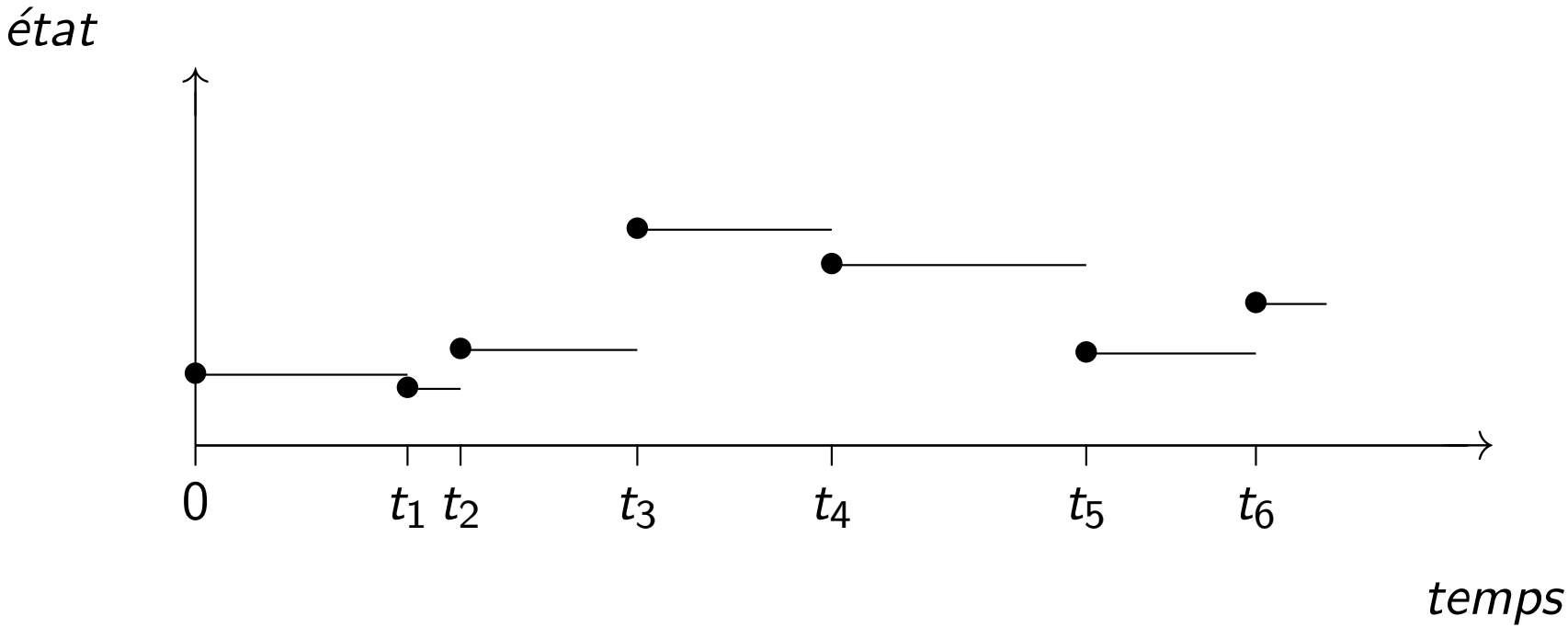
Suite d'événements $e_0, e_1, e_2 \ldots$, survenant aux instants $0 \leq t_0 \leq t_1 \leq t_2 \leq \ldots$.

$\mathcal{S}_i$: état du système immédiatement après e_i.

temps de la simulation: valeur courante de t_i.

$(t_i, \mathcal{S}_i)$ doit contenir assez d'information pour poursuivre la simulation (sauf les valeurs des variables aléatoires générées lors des événements e_j pour $j > i$).

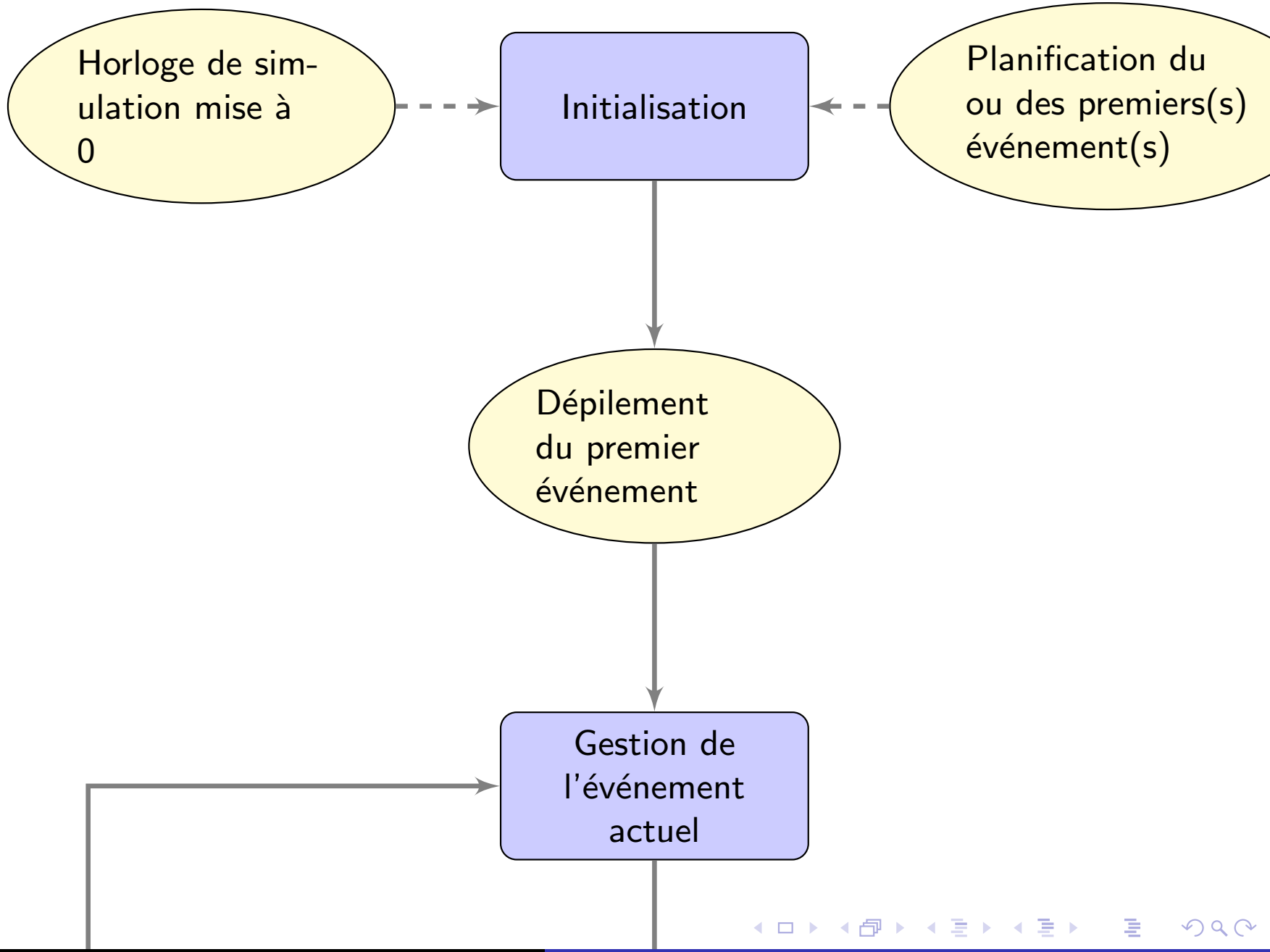
Evénements discrets

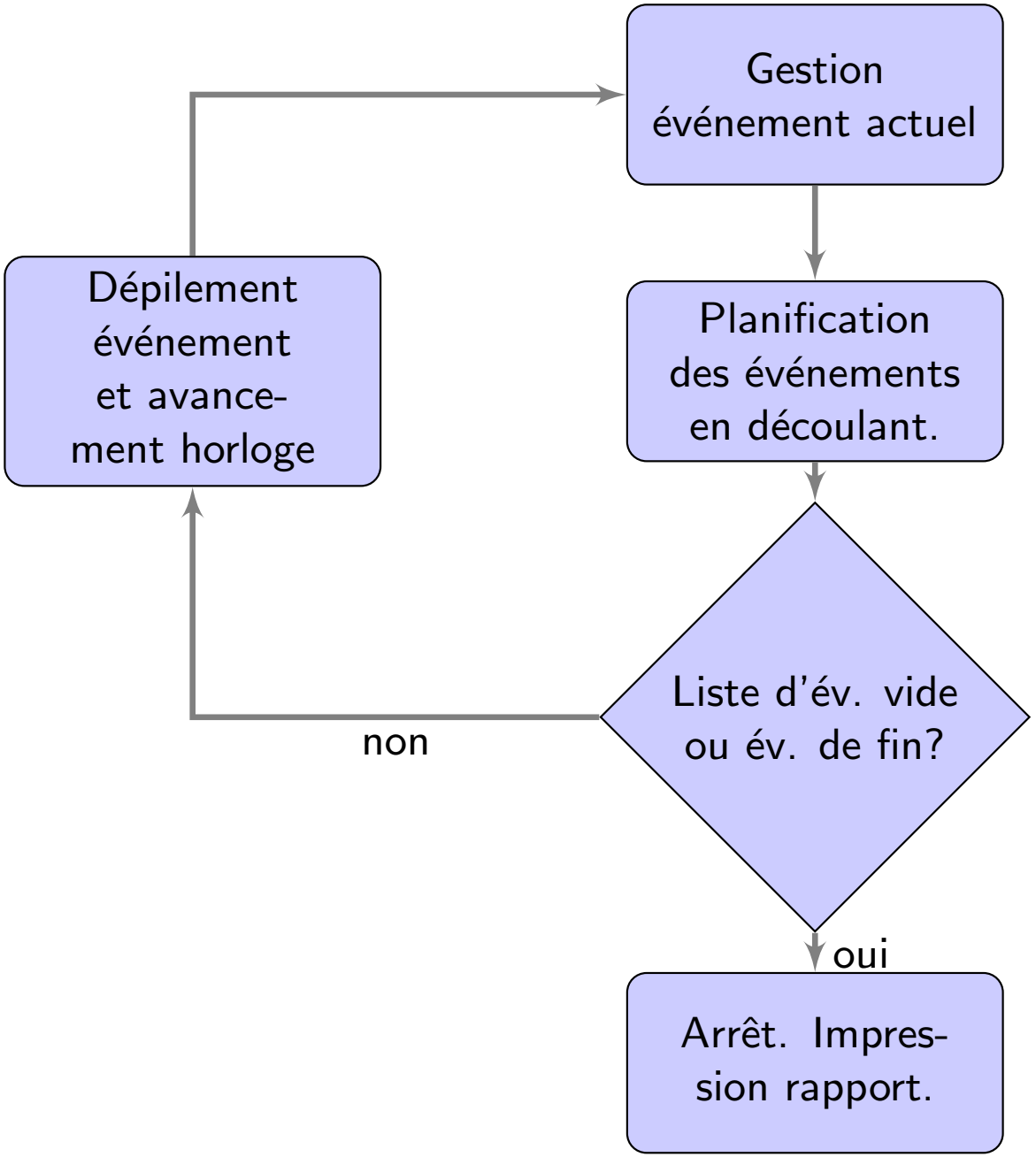


Organisation générale

- Programme principal.
- Horloge de simulation.
- Liste d'événements.
- Routines de génération de nombres aléatoires.
- Compteurs statistiques.
- Génération de rapport.

Evénement: action, schedule, cancel.


Horloge de simulation mise à 0
Initialisation
Planification du ou des premiers(s) événement(s)
Dépilement du premier événement
Gestion de l'événement actuel


Gestion
événement actuel
Planification
des événements
en découlant.
Liste d'év. vide
ou év. de fin?
non
oui
Dépilement
événement
et avance-
ment horloge
Arrêt. Impres-
sion rapport.

Exemple: file GI/G/1

Considérons une file $GI/G/1$, comprenant donc un seul serveur, devant lequel les clients arrivent un à un et sont servis un à la fois, en ordre FIFO. De plus,

- S_i est la durée de service du client i;
- A_i est la durée entre les arrivées des clients i et $i+1$.

Les S_i et les A_i sont mutuellement indépendantes, et de fonctions de répartition G et F, respectivement. Le premier client arrive au temps A_0 dans un système vide. Nous souhaitons simuler ce système pour une durée T et calculer l'attente moyenne par client et la longueur moyenne de la file d'attente. Les types d'événements sont

1. arrivée;
2. départ;
3. fin de la simulation.

Les variables aléatoires (indépendentes) à générer sont $A_1, A_2, A_3, \ldots$ et $S_1, S_2, S_3, \ldots$.

Exemple: file GI/G/1

- W_i, le temps d'attente du client i;
- $Q(t)$, la longueur de la file d'attente au temps t;
- $N_c(t)$, le nombre de clients ayant débuté leur service durant l'intervalle $[0, t]$.

Supposons que nous voulons calculer, pour l'intervalle $[0, T]$, l'*attente moyenne* par client,

$$\overline{W}_{N_c(T)} = \frac{1}{N_c(T)} \sum_{i=1}^{N_c(T)} W_i;$$

et la *longueur moyenne* de la file d'attente:

$$\overline{Q}_T = \frac{1}{T} \int_0^T Q(t)dt.$$

Exemple: file GI/G/1

Cette dernière intégrale est facile à calculer par simulation.

Pour chaque *client*, nous créons un *objet* contenant l'instant d'arrivée et la durée de service.

Variables d'état: liste des clients en attente et la liste les clients en service.

+ horloge de simulation, compteurs statistiques (au besoin), *liste* des événements futurs prévus, par ordre chronologique, et une procédure pour chaque type d'événement.

File d'attente

Arrivée Générer A selon F et prévoir une autre arrivée dans A unités de temps;
Créer le nouveau client et noter son instant d'arrivée;
Générer sa durée de service S selon G;
Si (serveur est occupé) **alors**
  Insérer ce client dans la liste des clients en attente;
sinon
  Insérer le client dans la liste des clients en service;
  Prévoir son départ dans S unités de temps;
Mise à jour des statistiques voulues.

Départ Enlever le client de la liste des clients en service;
Si la file d'attente n'est pas vide **alors**
  Enlever le premier client de la liste d'attente;
  L'insérer dans la liste des clients en service;
  Récupérer son S et prévoir son départ dans S unités de temps;
Mise à jour des statistiques voulues.

Fin-de-la-Simulation Imprimer un rapport et terminer le programme.

Simulation

Démarrer la simulation:

- initialiser des variables et compteurs,
- prévoir la "Fin-de-la-Simulation" au temps T
- générer A selon F et prévoir la première "Arrivée" dans A unités de temps,
- lancement de la simulation.

L'exécution de la simulation consiste simplement à répéter la boucle suivante:
Répéter: exécuter le prochain événement dans la liste d'événements
jusqu'à: la liste des événements prévus est vide,
ou bien un événement stoppe la simulation.

Simulation sans liste d'événements.

Pas toujours indispensable de recourir à l'approche par événements.

Exemple: récurrence de Lindley:

$$W_1 = 0, \quad W_{i+1} = \max(0,\ W_i + S_i - A_{i+1}).$$

Nous pouvons ainsi facilement simuler pour un nombre fixe de clients (au lieu d'un horizon de temps T fixe).

Le formulation du programme basée sur la récurrence de Lindsley revient à traiter un problème d'intégration sur $[0,1)^t$.

Récurrence de Lindley

Supposons que nous souhaitions estimer $E[\overline{W}_{100}]$ par $\overline{W}_{100}$:

$$\overline{W}_{100} = \frac{1}{100} \sum_{i=1}^{1} 00 W_i.$$

Il nous faut $A_1, S_1, A_2, S_2, \ldots, A_{99}, S_{99}$. Si on pose $A_i = F^{-1}(U_{2i-1})$ et $S_i = G^{-1}(U_{2i})$, où U_j, $j = 1, \ldots, 99$, sont des variables aléatoires uniformes sur $(0, 1)$, alors $\overline{W}_{100} = f(U_1, \ldots, U_{198})$ (f étant de forme inconnue dans le cas présent).

Nous avons donc $t = 198$, i.e.,

$$E[\overline{W}_{100}] = \int_{[0,1)^{198}} f(\boldsymbol{u}) d\boldsymbol{u}.$$

Nombre aléatoire de clients

Par contre, si nous voulons simuler $\overline{W}_{N_c(T)}$, le nombre de clients $N_c(T)$ est aléatoire et non borné. Nous devrions donc choisir $t = \infty$ comme dimension.

Approche par processus

Un processus est une séquence temporellement ordonnée d'événements interreliés, séparés par certains intervalles de temps, qui décrit l'expérience entière d'une "entité" comme celle-ci évolue à travers un "système". Un système ou un modèle de simulation peut avoir différents types de processus.

Une routine sera associée à chaque processus du modèle, décrivant son histoire entière à travers le système.

Au contraire de l'approche par événements, une routine de processus contient explicitement le passage du temps simulé et a généralement de multiples points d'entrée.

Processus vs Evenements

Une simulation utilisant l'approche par processus évolue aussi au cours du temps en exécutant les événements dans l'ordre de leur occurence.
En interne, les approches par événement et par processus sont donc très similaires.

Pour l'utilisateur, le raisonnement soit différent, et puisse parfois (mais pas toujours) être plus naturel pour l'approche par processus.

En outre, l'approche par processus est souvent plus lente en terme de temps d'exécution, puisque le simulateur doit au final traiter des événements, et gérer des processus pouvant être concurrents.

SimJulia

SimJulia travaille principalement avec les processus, mais gère également la notion d'événements.

Exemple

```
julia> using SimJulia

julia> function car(env::Environment)
         while true
           println("Start parking at $(now(env))")
           parking_duration = 5.0
           yield(Timeout(env, parking_duration))
           println("Start driving at $(now(env))")
           trip_duration = 2.0
           yield(Timeout(env, trip_duration))
         end
       end
```