SIMULATIONS

André-Luc BEYLOT
ENSEEIHT
Département Télécommunications et Réseaux

Plan Général

- Introduction - Les différents types de simulation
- Génération des nombres pseudo-aléatoires
- Intervalles de confiance
- Simulation Parallèle, simulations d'événements rares
- Outils de Simulation

Introduction

- Evaluation des performances des réseaux : Un réseau que l'on veut optimiser, dimensionner
  - Modélisation Faire sortir du protocole ou du réseau ce qui est important
  - Etude du modèle
    - Résolution Mathématique Lier le délai en fonction de la charge, ...
    - Simulation sur ordinateur
- Modèles comporte des ressources (logicielles ou matérielles) utilisées par des entités (paquets, trames, appels, cellules) Ressources partagées par des entités
- Critères de Performances à évaluer
  - Relatifs à l'utilisation des ressources (nombre moyen de paquets, % de temps d'occupation d'un lien, débit ...)
  - Relatifs aux entités traitées (temps de réponse, taux de perte)

Définition

- Une simulation est une expérience sur ordinateur qui permet de reproduire (par programme) le fonctionnement du système modélisé (on simule le modèle pas le système) pendant une durée *T* On reproduit le fonctionnement du modèle, pas du réseau

  On espère que la fonction soit une fonction affine, que l'augmentation soit linéaire -> le réseau n'est pas en train de s'emballer
  Temps d'exécution de même ordre de grandeur pour coupler des simulateurs avec des vrais réseaux

- Elle permet d'étudier l'évolution d'un certain nombre de processus stochastiques qui sont
  - à temps continu : ceux relatifs aux ressources (X_t)
  - à temps discret : $X_1, X_2, \ldots, X_n$ Délai du premier paquet, nème paquet...

Différents Types de Simulation

- Par Evénements (à événements discrets) - Asynchrone :
  - Gestion d'un échéancier : prochains événements, date
  - Extraction de l'événement *e* de date minimale
  - Insertion de nouveaux événements générés par *e*

C'est la méthode la plus couramment utilisée

Événement e qui engendre d'autres événements

Structure de données bien adaptée pour gérer un échéancier: dépend du problème qu'on traite, souvent des arbres

- Par Pas de Temps (incrément fixe) - Synchrone :
  - On regarde l'état du système à chaque pas de temps
  - Simultanéité des Evénements

Difficulté: choix du pas de temps
Fixer le temps de temps comme étant égal au pas de l'horloge

Déterminer l'ordre dans lesquels les événements se sont passés

Exemples :

Réseaux Mobiles/Satellites (propagation, déplacements)

ATM : paquets de taille identiques

SDH synchrone ATM asynchrone
on remplie des cellules vides pour rester synchrone

Exemple Simple - File M/M/1

On part d'un système vide, premier événement arrivée d'un premier paquet

Nb_paq_traités = 0 ; Nb_paq = 0 ;
T_Rep_Total = 0 ; C_Nb_paq = 0 ;
T_Inter_Arriv = Expo(λ) ; Premier paquet expo(lambda)
Insère(Ech, Arrivée, T_Inter_Arriv) ;
Evt = Extraire(Echéancier) ;
T_cr = T_Inter_Arriv ;
T_pred = 0 ;
Tant que Temps_cr < T_Max **Faire**
  C_Nb_paq += Nb_paq × (T_cr – T_pred);
  Si (Evt.type = Arrivée) **Alors**
    T_Emission = Expo(m.C) ;
    Créer(Paq_cr) ;
    Paq_cr.arrivée = T_cr ; Paq_cr.service = T_Emission ;
    Insère(file, Paq_cr) ;
    Si Nb_Paq = 0 **Alors**
     Insère(Ech, Départ, T_Emission + T_cr) ;
    Fin Si
    Nb_paq ++ ; T_Inter_Arriv = Expo(l) ;
    Insère(Ech, Arrivée, T_Inter_Arriv + T_cr) ;
  Sinon *{C'est un départ de paquet}*
    Nb_paq_traités ++ ; Paq_cr = File.tête ;
    T_Rep = T_cr – Paquet.date ;
    T_Rep_Total = T_Rep_Total + T_Rep ;
    Nb_Paq -- ;
    File.tête = Paq_cr.suivant ; Supprimer(Paq_cr) ;
    Si Nb_Paq > 0 **Alors**
      Insère(Ech, Départ, Temps_cr + File.tête.service) ;
    Fin Si
  Fin Si
  Supprimer(Evt) ;
  Evt = Extraire(Ech) ;
  T_pred = T_cr ;
  T_cr = Evt.date ;
Fin Tant que
T_cr=T_Max;
C_Nb_paq += Nb_paq x (T_cr – T_pred) ;
T_Rép_Moyen = T_Rep_Total/Nb_paq_traités ;
Nb_Paquets_Moyen = C_Nb_paq / T_Max ;

Exemple Simple - File M/M/1

On stock le nombre de paquet dans la file

{Initialisations}
Proch_arrivée = Expo(*l*) ;
Proch_départ = infini ;
Nb_paq = 0 ;
C_Nb_paq = 0 ;
T_cr = Proch_arrivée ;
T_pred = 0
Proch_Evt = arrivée ;

Tant que T_cr < T_Max **Faire**
  C_Nb_paq += Nb_paq x (T_cr – T_pred) ;
  Si (Proch_Evt = arrivée) **Alors**
    Si Nb_Paquets = 0 **Alors**
      T_Emission = Expo(*m*.C) ;
      Proch_départ = T_cr + T_Emission ;
    Fin Si
    Nb_paq --;
    T_Inter_Arriv = Expo(λ) ;
    Proch_arrivée = T_Inter_Arriv + T_cr ;
  Sinon *{C'est un départ de paquet}*
    Nb_Paq -- ;
    Si Nb_Paq > 0 **Alors**
      T_Emission = Expo(*m*.C) ;
      Proch_départ = T_cr + T_Emission ;
    Sinon
      Prochain_départ = infini ;
    Fin Si
  Fin Si
  T_pred = T_cr ;
  Si (Proch_départ < Proch_arrivée) **Alors**
    T_cr = Proch_départ ;
    Proch_Evt = départ ;
  Sinon
    T_cr = Proch_arrivée ;
    Proch_Evt = arrivée ;
  Fin Si
Fin Tant que
T_cr=T_Max ;
C_Nb_paq += Nb_paq x (T_cr – T_pred) ;
Nb_Paq_Moy = C_Nb_paq / T_Max ;

Pour une file G/G/1 on a le même programme on doit juste changer la loi Expo

1 élément dans l'échéancier quand la file est vide, 2 quand la file n'est pas vide avec prochaine arrivée et prochain départ

Temps d'exécution du programme de simulation est divisé par 10

Problèmes liés aux Simulations

- Validité du modèle

ne pourra pas être traitée avec la simulation car la simulation reproduit le fonctionnement du modèle et si le modèle est de mauvaise qualité les résultats ne reflèteront pas la réalité

Valider le modèle en faisant des mesures

Mesures passives: on place une sonde dans le réseau, pas invasif mais plus difficile pour mesurer le délai, on est qu'à un point de passage

Mesures actives: se fait sur le trafic, on envoie des sondes, paquets qui ne sont pas des paquets de données avec lesquels on fait des mesurers. Le trafic de mesure rajoute de la congestion dans le réseau

- Nombre pseudo-aléatoires

Problème des simulateurs ne viennent pas des générateurs de nombres PA

- Etablissement du régime permanent

Problème: il n'existe pas forcément un temps moyen du système

- Une simulation est une expérience
  - Intervalles de confiance

- Troncatures et Arrondis

Nombres Pseudo-Aléatoires

- Fonctions “random” : Nombres uniformément répartis entre 0 et 1 (ou entre 0 et MAX)

- Utilisent des générateurs congruentiels :
  - x(k+1) = (41358 x(k)) mod(2^31 - 1)
  - x(k+1) = (8088405 x(k) + 1) mod(2^31)

Périodiques et pseudo périodiques, pas indépendants

- Suites Très longues, bien pour les simulations
- Ne conviennent pas dans le domaine de la sécurité (trop prévisibles)
  - n=p*q, p et q spéciaux, p=2p1+1, p1=2p2+1, p, p1, p2 premiers
  - x(k+1) = x(k)^2 mod(n), Utiliser les log2(n) derniers bits
  - Réaliser le XOR avec le mot à coder

Nombres Pseudo-Aléatoires

- Pour générer des nombres selon une certaine loi :
  - Inversion de la fonction de répartition
- Exemples :
  - Variables continues : Exponentielle : $y = -\ln(x) \ / \ l$
  - Variables discrètes : Bernoulli $y = 1_{\{x>p\}}$

Fonction de répartition d'une valeur aléatoire continue positive

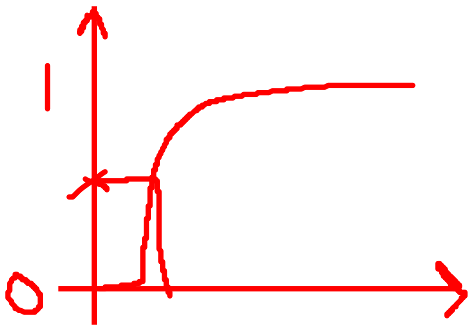
Inversion fonction de répartition

y = FX(x) = 1 - exp(-lambx)

1 - y = exp(-lambx)

ln(1-y) = -lambx

x = -ln(1-y)/lamb



Validation des Simulations

- La simulation dure un temps fini T
- Quelle confiance doit-on donner au résultat présenté ?

IL FAUT DETERMINER L'INTERVALLE DE CONFIANCE

- ATTENTION :
  - C'est une validation de la simulation
  - Ce n'est pas une validation du modèle

Buts et Hypothèses

- But de la simulation : obtenir une estimation de

Estimer les critères de performances

$$Z = \lim_{t \to \infty} E\left[Y(X_t)\right] \qquad Z = \lim_{n \to \infty} E\left[Y(X_n)\right]$$

Pi(k) = lim(t -> inf) E(1|{xt = k})

- Résultat d'une simulation

Moyenne temporelle

$$Z_T = \frac{1}{T} \int_{t=0}^{T} Y(X_t)\,dt \qquad Z_n = \frac{1}{n} \sum_{i=1}^{n} Y(X_i)$$

Moyenne arythmétique

Variables continues

Variables discretes

- Hypothèses :
  - Processus Strictement Ergodiques $\quad \forall Y, Z_T \xrightarrow[T \to \infty]{} Z$

Si on remplace y par la fonction identité, Zt va tendre vers T

  - Asymptotiquement Stationnaires du Second Ordre

$$\lim_{t \to \infty} E\left[X_t\right] = m < \infty \qquad \lim_{t \to \infty} E\left[X_t X_{t+h}\right] = C(h), \text{continue en } 0$$

Conclusion qu'on peut tirer d'une simulation: on ne peut pas obtenir des valeurs maximales

Intervalle de Confiance

- Soit Z_T un estimateur de Z

Intervalle de confiance = une valeur & un niveau de confiance

- L'intervalle de confiance pour un niveau de confiance $1 - \eta$ est le réel ε positif tel que

$$\Pr\left[\left|Z - Z_T\right| < \varepsilon\right] \geq 1 - \eta$$

Plus on veut un niveau de confiance important et plus la durées simulée va être grande

- BienAymé-Tchebychev

$$Si \quad E\left[\left|Z - Z_T\right|^{\alpha}\right] < \infty \quad alors \quad \Pr\left[\left|Z_T - Z\right| \geq \varepsilon\right] \leq \frac{E\left[\left|Z_T - Z\right|^{\alpha}\right]}{\varepsilon^{\alpha}}$$

Bienaymé - Tchebychev

- Hypothèse : la variance de l'estimateur $\sigma^2_{Z_T}$ existe

- α = 2 $$\Pr\left[|Z_T - Z| \geq \varepsilon\right] \leq \frac{E\left[|Z_T - Z|^2\right]}{\varepsilon^2}$$

- Quand T est grand on remplace Z par $E[Z_T]$

$$\frac{E\left[|Z_T - Z|^2\right]}{\varepsilon^2} \to \frac{\sigma^2_{Z_T}}{\varepsilon^2} \qquad \Pr\left[|Z_T - Z| < \varepsilon\right] \geq 1 - \frac{\sigma^2_{Z_T}}{\varepsilon^2}$$

- Pour η donné, on obtient alors $$\varepsilon = \frac{\sigma_{Z_T}}{\sqrt{\eta}}$$

- exemple : 1 - η = 95%, $\frac{1}{\sqrt{\eta}} \approx 4.5$ $\qquad \varepsilon \approx 4.5\sigma_{Z_T}$

Pour un niveau de confiance de 95% l'intervalle est 4,5 * l'écart type de l'estimateur

Utilisation de la loi normale

$$Z_T \sim N\left(E[Z_T] = Z, \sigma^2_{Z_T}\right) \qquad \frac{Z_T - Z}{\sigma_{Z_T}} \sim N(0,1)$$

$$\Pr\left[\left|\frac{Z_T - Z}{\sigma_{Z_T}}\right| < d\right] \leq \pi_d - \pi_{-d} \qquad \pi_x = \frac{1}{2\pi} \int_{u=-\infty}^{x} e^{\frac{-u^2}{2}} du$$

$$\varepsilon = \sigma_{Z_T} d \qquad 1 - \eta = \pi_d - \pi_{-d}$$

Problème qu'on a est que dans la pratique le plus dur est de calculer l'écart type de l'estimateur

- Exemple $1 - \eta = 95\%$, $\varepsilon \approx 1.96\sigma_{Z_T} \sim 2\sigma_{Z_T}$

- Rq : Dans la pratique, le plus dur n'est pas de savoir si l'on peut appliquer la loi normale c'est de calculer la variance de l'estimateur !

On estime l'écart type de l'estimateur

Batch Mean (Le Gall)

- Si le processus simulé est Asymptotiquement Stationnaire du Second Ordre

$$s^2_{Z_T} \sim \frac{A}{T}$$

Plus la simulation va être longue plus l'estimateur va être précis

Une seule simulation de durée T et on la découpe en n t et faire l'estimation sur chaque t -> n réalisations du même estimateur

- Pour déterminer la variance de l’estimateur, on découpe la simulation en plusieurs blocs de durée T_1.
  - Z_i la réalisation de Z_{T1} sur le ième bloc.
  - Variance empirique comme estimateur de la variance

Estimateur sans biais

$$\hat{s}^2_{Z_{T_1}} = \frac{1}{n-1}\left[\sum_{i=1}^{n} Z_i^2 - nZ^2_{nT_1}\right] \qquad s^2_{Z_T} \sim \frac{T_1}{T}\hat{s}^2_{Z_{T_1}}$$

- On peut alors utiliser les résultats précédents

$$e = 4.5\frac{\sqrt{\hat{s}^2_{Z_{T_1}}}}{\sqrt{n}} \qquad e = 2\frac{\sqrt{\hat{s}^2_{Z_{T_1}}}}{\sqrt{n}}$$

Méthode de Réplication

On fait n simulations

- Méthode des “blocs indépendants”
- On fait n simulations de durée $T_1 \Rightarrow Z_i$
- Les Zi étant i.i.d + lois des grands nombres

$$Z_T = \frac{1}{n}\sum_{i=1}^{n} Z_i$$

$$\sigma^2_{Z_T} = \frac{1}{n}\sigma^2_{Z_{T_1}}, E[Z_T] = E[Z_{T_1}]$$

$$Z_T \sim N\left(Z, \sigma^2_{Z_T}\right)$$

La moyenne arithmétique converge vers une loi normale

$$\hat{\sigma}^2_{Z_{T_1}} = \frac{1}{n-1}\left[\sum_{i=1}^{n}\left(Z_i - \overline{Z}\right)^2\right]$$

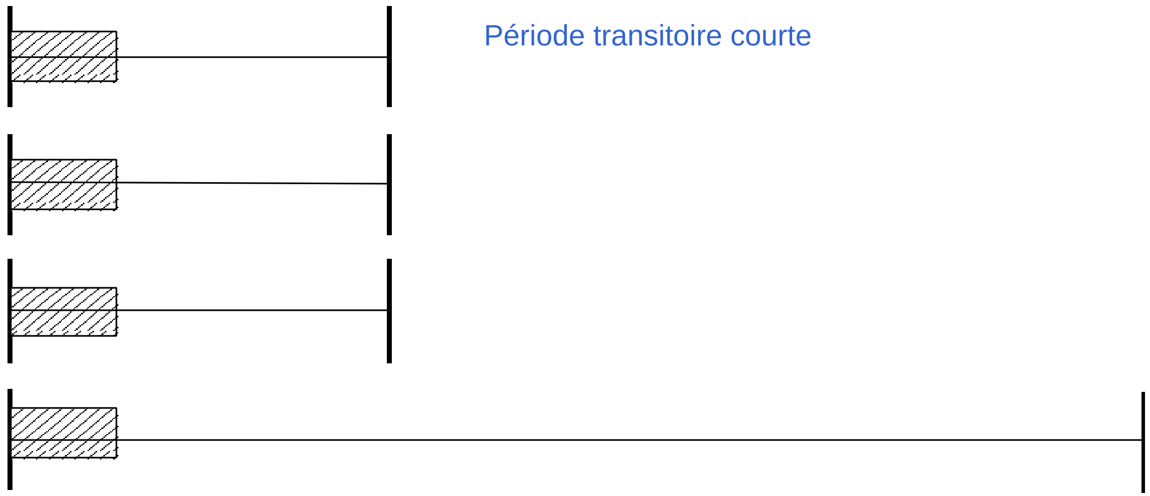
- Ce qui conduit aux mêmes résultats

$$e = 2\frac{\sqrt{\hat{\sigma}^2_{Z_{T_1}}}}{\sqrt{n}}$$

Comparaison des 2 méthodes

- Batch Mean
  - Z_i corrélés parce qu'on fait qu'une simulation donc on vide pas le système à la fin d'un bloc
    On peut enlever les covariances
  - Période Transitoire Courte

- Réplication
  - Période Transitoire à chaque simulation
  - Jusqu'où va l'indépendance ?



Période transitoire courte

Points de Renouvellement

- C'est un instant / l'évolution du système à partir de cet instant est indépendante du passé
  - Exemple : File d'attente vide et arrivées poissonniennes

- Les résultats sont indépendants sur les différents blocs

$$E[f(X)] = E[Y]/E[a]$$

  - Y : intégrale de la fonction à estimer sur la durée de la période
  - α : durée de la période
  - E[Y] et E[α] estimés par la moyenne arithmétique

- PB MAJEUR : dans un “vrai” système, ces points sont difficiles à repérer (nombreux tests à effectuer)

NON

Remarques Pratiques

- Intervalles de confiance absolus et relatifs

$$e_r = eZ_T \qquad e_r \sim \frac{C}{\sqrt{T}}$$

- On se fixe un intervalle de confiance à atteindre γ
  - 1ère simulation de durée T donne e_r
  - 2ème simulation de durée $T' = T\left(\frac{e_r}{\gamma}\right)^2$
- Intervalles de confiance sur somme/produit/quotient : idem calculs d 'incertitude en physique, on différencie et on prend les valeurs absolues
  - Sommes/différences : additionne intervalles absolus
  - Produit : additionne intervalles relatifs

Simulation par roulette

Chaîne de markov de grande taille dur de résoudre le système linéaire

On note les fréquences de passages qu'on a dans les états de la chaîne

- Quand tout est exponentiel, on a juste besoin de connaître l'état courant du système
- Inf de lois exponentielles est exponentiel
- On fait les moyennes sur les événements (PASTA)

- Exemple : File M/M/C/C Formule d'Erland B
  - Simulation Classique, on doit stocker la date de fin de tous les appels en cours + date prochaine arrivée d'appel
  - Par roulette : uniquement le nombre d'appels en cours

Exemple Simple - File M/M/C/C

```
Nb_rejets = 0 ; Nb_appels = 0 ;T_cr = 0 ;
En_cours = 0 ; Proch_arrivée=Expo(l) ;
Evt = Créer_Evt(Arrivée, Proch_arrivée) ; Insérer_Evt(Evt) ; 1er_Evt=Evt ;
Tant que T_cr < T_Max Faire
  Si Type(1er_Evt) = Arrivée Alors
      Nb_appels ++ ; Proch_arrivée=Expo(l) + T_cr ;
      Evt = Créer_Evt(Arrivée, Proch_arrivée) ;  Insérer_Evt(Evt) ;
      Si En_cours < C Alors
         En_cours ++; Fin_appel = Expo(m) + T_cr ;
         Evt=Créer_Evt(Fin, Fin_appel) ;  Insérer_Evt(Evt) ;
     Sinon
         Nb_rejets ++;
     Fin si
  Sinon
    En_cours — ;
  Fin Si
Evt=1er_evt ; 1er_evt=Suivant(1er_evt) ; T_cr=Date(1er_evt) ; Supprimer(Evt) ;
Fin Tant que
Estimation_rejet = Nb_rejets/Nb_appels ;
```

Roulette - File M/M/C/C

On se positionne dans la chaîne, taux d'arrivée lambda, taux de départ mu, 2mu jusqu'à Cmu. On part de n'importe où dans la chaîne. On tire avec la loi de bernouilli

Nb_rejets = 0 ; Nb_appels = 0
Nb_Evt = 0 ; En_cours = 0 ;
Tant que Nb_Evt < N_Max **Faire**
  Si (Bernoulli(*l*/(En_cours . *m* + *l*) = 1) **Alors**
    Nb_appels ++ ;
    Si En_cours < *C* **Alors**
      En_cours ++ ;
    Sinon
      Nb_rejets ++ ;
    Fin si
  Sinon
    En_cours — ;
  Fin si
  Nb_Evt ++ ;
Fin tant que
Estimation_rejet = Nb_rejets/Nb_appels

Simulation parallèle/distribuée

Parallèle: répartir le réseau qu'on simule. Tel processeur simule tel commutateur. On découpe le réseau en plusieurs sous-réseaux et chacune des machine traite une partie du réseau. Chaque processeur va avoir son échéancier

- Découper le réseau en sous-réseaux / envoi de clients assez rares entre sous-réseaux

  Si la date de l'événement est inférieure à la date de l'échéancier, on revient en arrière: méthode optimiste

- Communication par envois de messages
- Politique Pessimiste :
  - avance l 'échéancier quand on est sûr qu'aucun evt avec une date inférieure ne viendra perturber le système
- Politique Optimiste :
  - on avance l 'échéancier, si un événement avec une date inférieure parvient, on revient en arrière

$\Rightarrow$ PEU EFFICACE

Méthode optimiste ne marche pas

- Autre solution : une simulation sur chaque processeur + Réplication

  Plus du parallélisme

Simulation d'événements rares

- Simulations classiques permettent d 'estimer des probas
  - jusqu 'à 10-6 ou 10-7 avec des simulateurs optimisés
  - jusqu 'à 10-4 ou 10-5 avec des produits du marché

- Pbs : les réseaux sont de plus en plus fiables (BER ~ 10-12, perte de paquets ~ 10-9)

Simulateur ont du mal à simuler des probabilités aussi faibles

- Quelques solutions : Trouver des conditions nécessaires pour que l 'événement rare se produise

Pour que la file déborde, il faut au moins qu'elle soit à moitié pleine. On fait une simulation ou on calcule la probabilité qu'elle soit à moitié pleine. Puis une autre simulation où on calcule la probabilité que la file soit pleine sachant qu'elle est à moitié pleine

Simulation d'événements rares

- Exemple :
  - Pr[Taille du buffer = C] => Pr[taille du buffer ≥ N]
  - Deux simulations :
    - Déterminer : P1 = Pr[taille du buffer ≥ N]
    - Déterminer : P2 = Pr[Taille du buffer = C | taille du buffer ≥ N]
  - Pr[Taille du buffer = C] = P1 x P2

- PBS :
  - difficiles de trouver ces conditions ... sauf quand on a une solution mathématique (processus de naissance et de mort)
  - Simulations des probabilités conditionnelles
  - Pbs théoriques pas complètement résolus

Outils de Simulation

- 1ère génération : Langages de programmation classiques
  - C, C++, Pascal, Java
  - Avantages : rapidité d'exécution
  - Inconvénients : à réserver aux initiés, faible réutilisabilité

On change un minimum le modèle et ça ne marche plus

- 2ème génération : Les langages de simulation (ou de description de modèles)
  - QNAP2 : description de files d 'attente + simulations à evts discrets + résolution analytique
  - SIMULA, SIMSCRIPT II : langage à objets + simulation
  - Avantages : Plus besoin de décrire les objets de base, moteur de simulation, fonctions statistiques élémentaires
  - Inconvénients : il faut encore programmer, réutilisabilité faible

Outils de Simulation

- 3ème génération : Les outils modernes ont déjà des bibliothèques de modules prédéfinis présents
  Modèle de base de routeur, de liens, de protocoles connus
  - Interface graphique conviviale
  - Nombreux modules prédéfinis (bibliothèques)
  - Rapidité de mise en place des modèles
  - Peu de programmation : on « assemble » les morceaux

Problème de granularité: on ne fait que reproduire le fonctionnement de modèle. Modéliser c'est simplifier.
Pour une même configuration on a beaucoup de paramètres, modèles bien plus sophistiqués. Pour un même élément physique on peut avoir plusieurs modèles.
Si le niveau de granularité ne correspond pas au modèle posé, on a un modèle trop sophistiqué avec trop de paramètres

- Inconvénients :
  - usines à gaz : problème de granularité
  - outils développés par des informaticiens qui ne connaissent pas grand chose aux problèmes posés par les simulations !

Exemples

- OPNET : très orienté réseaux Incovénient: payant
  - 3 niveaux de description : réseau, nœud, processus
  - Au niveau le plus bas : automates finis
  - Grande Richesse des modèles de base (modules radio)
- BONeS : vient du domaine de la VLSI
  - proche d 'OPNET : blocs, automates
- SES Workbensh : orienté architectures parallèles
- Simulateur = moteur de simulation + code C (ou C++) décrivant les modules

Exemples (fin)

- MODSIM : lié à SIMSCRIPT
- MODLINE : lié à QNAP

- NS : Network Simulator
  - vient de l 'IETF, très lié à l'évolution de l 'Internet
  - Domaine Public
  - Modules en C++ avec Interface OTCL

- OPNET : industriels/ NS : universitaires