Capítulo 24: Introdução à Simulação

The best advice to those about to embark on a very large simulation is often the same as Punch's famous advice to those about to marry: Don't!

— Bratley, Fox, and Schrage (1986)

Simulação: questões fundamentais

- Quais são os erros mais comuns em simulação e porquê muitas simulações falham?
- Que linguagem deveria ser usada para desenvolver um modelo de simulação?
- Quais são os diversos tipos de simulação?
- Como programar (*escalonar*) eventos numa simulação?
- Como verificar e validar um modelo?
- Como determinar se a simulação atingiu um estado permanente?
- Por quanto tempo devemos executar uma simulação?
- Como gerar números aleatórios uniformemente distribuídos?
- Como verificar se um determinado gerador de números aleatórios é bom?
- Como selecionar sementes para um gerador de números aleatórios?
- Como gerar variáveis aleatórias de acordo com uma determinada distribuição?
- Que distribuições devem ser utilizadas e quando?

Simulação

- Técnica útil para a análise de desempenho de sistemas computacionais.
- Especialmente:
  - se o sistema não estiver disponível,
  - para prever o desempenho de diversas alternativas,
  - facilidade de efetuar comparações para uma maior variedade de cargas e de ambientes.

Erros comuns em simulação

- *Usar um nível inapropriado de detalhe:* nem sempre um modelo mais detalhado é um melhor modelo.

  Problemas com demasiados detalhes:

  - necessita de mais tempo para ser desenvolvido
  - aumenta a probabilidade de erros e o tempo gasto em identificá-los
  - maior tempo de execução
  - necessita de um conhecimento mais detalhado dos parâmetros de entrada, que se não estiverem disponíveis, tornam o modelo impreciso

  É melhor começar com um modelo menos detalhado, obter alguns resultados, estudar as sensibilidades e introduzir maiores detalhes nas áreas que causem maior impacto nos resultados.

- *Uso de linguagem inadequada:*

  - **linguagens de simulação** necessitam de menos tempo para o desenvolvimento do modelo e facilitam a verificação e análise estatística.
  - **linguagens de propósito geral** são mais portáteis e possibilitam um maior controle sobre a eficiência e tempo de execução da simulação.

Erros comuns em simulação

- *Modelos não verificados:* erros de programação podem tornar os resultados sem sentido.
- *Modelos inválidos:* o modelo pode não representar corretamente o sistema real por causa de hipóteses incorretas. É preciso que o modelo seja validado.
- *Tratamento inadequado das condições iniciais:* a parte inicial de uma simulação geralmente não é representativa do comportamento do sistema em regime permanente.
- *Simulações demasiado curtas:* os resultados são muito dependentes das condições iniciais e podem não ser representativos do sistema real.
- *Geradores de números aleatórios ruins:* é mais seguro utilizar geradores conhecidos que já foram vastamente analisados.
- *Seleção inadequada das sementes:* o uso inadequado de sementes dos geradores de números aleatórios pode levar, por exemplo, a correlações indesejadas.

Outras causas de erros

Qualquer programa quando estiver rodando, estará obsoleto.
Se um programa for útil, terá que ser alterado.
A complexidade de um programa cresce até superar a capacidade do programador que deve fazer a sua manutenção.
— Datamation 1968

Adicionar mais pessoas a um projeto de software que está atrasado fará com que ele atrase ainda mais.
— Fred Brooks

- Estimativa inadequada do tempo necessário
- Falta de objetivo claro
- Mistura incompleta de talentos essenciais
  - Liderança de projeto
  - Modelagem e Estatística
  - Programação
  - Conhecimento do sistema que está sendo modelado
- Nível inadequado de participação do usuário
- Documentação Obsoleta ou Inexistente
- Incapacidade de Gerenciar o Desenvolvimento de um Grande e Complexo Programa de Computador
- Resultados Misteriosos

Lista de verificações

1. Verificações antes de desenvolver uma simulação:
   (a) O objetivo da simulação está claramente especificado?
   (b) O nível de detalhes do modelo está adequado para o objetivo?
   (c) O time de simulação inclui pessoas com experiência de liderança de projetos, modelagem, programação e sistemas computacionais?
   (d) Foi alocado tempo suficiente para este projeto?
2. Verificações durante o desenvolvimento:
   (a) O gerador de números aleatórios que está sendo utilizado na simuação, foi testado em relação à uniformidade e independência?
   (b) O modelo está sendo revisado periodicamente com o usuário final?
   (c) O modelo está documentado?
3. Verificações depois que a simulação está sendo executada:
   (a) A duração da simulação está apropriada
   (b) Foram removidos os transientes iniciais?
   (c) O modelo foi completamente verificado?
   (d) O modelo foi validado antes da utilização de seus resultados?
   (e) Se há resultados inesperados, eles foram validados?
   (f) As sementes foram escolhidas de modo que as seqüências de valores aleatórios não se sobreponham?

Terminologia

- **Variáveis de estado:** definem o estado do sistema.

  A simulação pode ser continuada a partir do conhecimento das variáveis de estado.

  Exemplo: comprimento da fila de jobs.

- **Evento:** mudança no estado do sistema.

  Exemplos: chegada de um job,
  início de uma nova execução,
  partida do job.

- **Modelo de Tempo Contínuo:** o estado do sistema está definido em todos os instantes.

  Exemplo: Modelo de escalonamento de CPU.

- **Modelo de Tempo Discreto:** o estado do sistema está definido apenas em instantes particulares.

  Exemplo: Número de estudantes que assistem este curso.

Terminologia

- **Modelos de Estado Contínuo e de Estado Discreto:** dependendo de se as variáveis de estado são contínuas ou discretas.

  Exemplo: Tempo gasto estudando uma determinada matéria vs Número de estudantes.

  Modelo de estado discreto = **Modelo de Eventos Discretos**

  Modelo de estado contínuo = **Modelo de Eventos Contínuos**

- Continuidade de tempo $\neq$ Continuidade de estado

  Combinações possíveis de modelos:

  - estado discreto/tempo discreto
  - estado discreto/tempo contínuo
  - estado contínuo/tempo discreto
  - estado contínuo/tempo contínuo

Terminologia

- **Modelos Determinísticos e Probabilísticos:** nos modelos determinísticos, os resultados podem ser previstos com certeza.

- **Modelos Estáticos e Dinâmicos:** modelos estáticos são aqueles nos quais o tempo não é uma variável.

  Exemplo: $E = mc^2$ vs Modelo de Escalonamento de CPU

- **Modelos Lineares e Não-Lineares:**

  Saída = f(Entrada)

Terminologia

- **Modelos Abertos e Fechados:** nos modelos abertos a entrada é externa ao modelo e independente do mesmo.

- **Modelos Estáveis e Instáveis:**

  Estável $\Rightarrow$ atinge estado permanente
  Instável $\Rightarrow$ muda continuamente de comportamento.

Modelos de Sistemas Computacionais

- Tempo contínuo
- Estados discretos
- Probabilístico
- Dinâmico
- Não-linear
- Aberto ou fechado
- Estável ou instável

Seleção de uma Linguagem para Simulação

1. Linguagem de simulação
2. Linguagem de propósito geral
3. Extensão de uma linguagem de propósito geral
4. Pacote de simulação

Linguagens de Simulação

- Economizam tempo de desenvolvimento
- Recursos embutidos para:
  - avançar o tempo
  - programar eventos
  - manipulação de entidades
  - geração de valores aleatórios
  - coleta de dados estatísticos
  - geração de relatórios
- Mais tempo para questões específicas do sistema
- Código modular, bastante legível

Linguagem de Propósito Geral

- Familiaridade do analista
- Grande disponibilidade
- Início imediato
- Tempo gasto com o desenvolvimento de rotinas para tratamento de eventos e geração de valores aleatórios
- Outras questões:
  - Eficiência
  - Flexibilidade
  - Portabilidade

Recomendação: Aprenda pelo menos uma linguagem de simulação.

Extensão de uma linguagem de propósito geral

Exemplos: GASP (para FORTRAN) e SMPL (para C)

- Coleção de rotinas para tratar tarefas de simulação
- Compromisso entre eficiência, flexibilidade e portabilidade.

Pacotes de Simulação

Exemplos: QNET4, RESQ e BONeS

- Diálogo de entrada
- Biblioteca de estruturas de daos, rotinas e algoritmos
- Grande economia de tempo
- Inflexível $\Rightarrow$ Simplificação

Tipos de Linguagens de Simulação

1. Linguagens de simulação contínuas:
   CSMP, DYNAMO
   Equações diferenciais
   Usadas em engenharia química

2. Linguagens de simulação de eventos discretos:
   SIMULA e GPSS

3. Combinadas:
   SIMSCRIPT e GASP.
   Permitem simulações discretas, contínuas ou combinadas.

Tipos de Simulações

1. Emulação: utilizando hardware ou firmware
   Exemplos: emulador de terminal, emulador de processador
   Envolve basicamente questões de projeto de hardware
2. Simulação de Monte Carlo
3. Simulação Dirigida por Traces
4. Simulação de Eventos Discretos

Método de Monte Carlo

Origem: em homenagem ao Conde Montgomery de Carlo, jogador e gerador de números aleatórios italiano (1792-1838).

Um método de animar certos ambientes estatísticos e numéricos através do registro de apostas no resultado de uma computação.

— The Devil's DP Dictionary
McGraw Hill (1981)

Simulação de Monte Carlo

- Simulação estática (sem eixo do tempo)
- Para modelar fenômenos probabilísticos
- Necessita de números pseudo-aleatórios
- Usado para avaliar expressões não probabilísticas usando métodos probabilísticos.

Exemplo de Simulação de Monte Carlo

Deseja-se obter o resultado da seguinte integral:

$$I = \int_0^2 e^{-x^2} dx$$

$$x \sim \text{Uniform(0,2)}$$

$$\text{Função densidade } f(x) = \frac{1}{2} \text{ iff } 0 \leq x \leq 2$$

$$y = 2e^{-x^2}$$

$$\begin{aligned} E(y) &= \int_0^2 2e^{-x^2} f(x) dx \\ &= \int_0^2 2e^{-x^2} \frac{1}{2} dx \\ &= \int_0^2 e^{-x^2} dx \\ &= I \end{aligned}$$

Portanto, I pode ser calculada através da geração de números aleatórios x_i, cálculo de y_i, e tomando a média como a seguir:

$$x_i \sim \text{Uniform(0,2)}$$

$$y_i = 2e^{-x_i^2}$$

$$I = E(y) = \frac{1}{n} \sum_{i=1}^{n} y_i$$

Simulação Dirigida por Traces

- Trace = registro de eventos de um sistema ordenado de acordo com o tempo
- Simulação dirigida por Traces = utiliza os traces como entrada
- Utilizada
  para analisar ou ajustar algoritmos de gerenciamento de recursos:

  Paginação, caches, escalonamento de CPU, prevenção de *deadlock*, alocação dinâmica de memória.
- Exemplo: Trace = Padrões de referência de páginas
- Deve ser independente do sistema em estudo

  Exemplo: um trace de páginas recuperadas depende da dimensão do conjunto de trabalho e da política de substituição de páginas.

  Não é bom para o estudo de outras políticas de substituição de páginas

  É melhor usar as páginas que foram referenciadas.

Vantagens das simulações dirigidas por traces

1. Credibilidade
2. Validação fácil:

   Basta comparar o resultado de simulação com o medido
3. Carga de trabalho precisa:

   Modela correlações e interferências
4. Compromissos detalhados:

   Carga de trabalho detalhada $\Rightarrow$ Permite estudar pequenas alterações nos algoritmos
5. Menor aleatoriedade:

   Trace $\Rightarrow$ entrada determinística $\Rightarrow$ Menor número de repetições
6. Comparação Justa: Melhor do que entrada aleatória
7. Semelhança com a Implementação Real: o modelo dirigido por traces é semelhante ao sistema real $\Rightarrow$ Permite compreender a complexidade da implementação

Desvantagens das simulações dirigidas por traces

1. Complexidade: mais detalhada
2. Representatividade:
   A carga de trabalho varia com o tempo e com o equipamento
3. Finitude:
   Poucos minutos de trace enchem um disco
4. Ponto único de validação:
   Um trace = um ponto
5. Detalhe
6. Compromissos:
   É difícil alterar a carga de trabalho

Simulação de Eventos Discretos

Concentração de substâncias químicas $\Rightarrow$ Simulação de eventos contínuos

Número de jobs $\Rightarrow$ Eventos discretos

Estados discretos $\neq$ tempo discreto

Componentes de Simulação de Eventos Discretos

1. Escalonador de Eventos
   (a) Programa o evento X para o instante T.
   (b) Congela o evento X durante o intervalo de tempo dt.
   (c) Cancela um evento X previamente programado.
   (d) Congela o evento X indefinidamente
   (e) Programa um evento congelado indefinidamente.
2. Relógio de Simulação e Mecanismo de Avanço do Tempo
   (a) abordagem baseada em unidades de tempo
   (b) abordagem dirigida a eventos
3. Variáveis de Estado do Sistema

   Global = Número de jobs
   Local = Tempo de CPU necessário para um dado job
4. Rotinas associadas aos eventos: Uma para cada evento.

   Exemplo: chegada de jobs, escalonamento de jobs e partida de jobs

Componentes de Simulação de Eventos Discretos

5. Rotinas de Entrada: Obtenção dos parâmetros do modelo Variação dos parâmetros dentro de uma certa faixa.
6. Gerador de Relatórios
7. Rotinas de Inicialização:
   Atribui o estado inicial. Inicializa as sementes.
8. Rotinas de Trace: Podem ser ligadas ou desligadas
9. Gerenciamento Dinâmico de Memória: Coleta de Lixo
10. Programa Principal

Algoritmos de Conjunto de Eventos

Conjunto de Eventos = Lista ordenada do registro dos eventos futuros

Operações Básicas:

- Inserção de um novo evento
- Remoção do próximo evento a ser executado

1. Lista Ligada Ordenada: SIMULA, GPSS, GASP IV e SMPL

Busca da esquerda para a direita

Algoritmos de Conjunto de Eventos

2. Lista Linear Indexada:

   Vetor de índices $\Rightarrow$ Não é necessário fazer uma busca para encontrar a sublista.

   Intervalos (Δt) fixos ou variáveis

   Apenas a primeira lista é mantida ordenada

3. Fila de Calendários: Todos os eventos de $1^{\underline{\text{O}}}$ de Janeiro encontram-se numa mesma página, seja os de 1997 que os de 1998.

4. Estruturas em Árvores: Árvore binária $\Rightarrow \log_2 n$

Algoritmos de Conjunto de Eventos

5. Heap: o evento é um nó de uma árvore binária

Até dois filhos
O instante de ocorrência do evento associado a cada nó é menor do que o de qualquer de seus filhos $\Rightarrow$ Raiz é o próximo.
Heaps podem ser armazenadas como vetores
Os filhos de um nó na posição i encontram-se nas posições $2i$ e $2i+1$

Critérios de Escolha da Estrutura mais Apropriada

Qual a alternativa mais eficiente?

- Lista ligada simples: menos do que 20 eventos
- Listas lineares indexadas: entre 20 e 120 eventos
- Heaps: mais do que 120 eventos