ODD Simulação de fluxo de fila

1. Propósito

Questão: Qual é o propósito deste modelo?

Resposta: O propósito do modelo é simular o fluxo de estudantes através de vários estágios em um sistema definido. A simulação permitirá analisar o comportamento da fila de atendimento de forma controlada e replicável, sem interferir diretamente no funcionamento real do restaurante.

Através da simulação, será possível testar diferentes cenários e estratégias de melhoria operacional, avaliando seus impactos antes de serem implementados na prática. A simulação possibilita a compreensão mais profunda dos fatores que influenciam os tempos de espera, como padrões de chegada dos comensais, capacidade de atendimento, *layout* do restaurante etc.

O objetivo principal do estudo das filas nesse restaurante é avaliar o potencial de aplicação da metodologia *Lean Thinking* para melhorar a eficiência operacional do restaurante, reduzindo os tempos de espera e proporcionando uma experiência mais satisfatória aos usuários. A simulação permitirá testar e comparar diferentes estratégias de melhoria. Ao final, espera-se que a simulação forneça subsídios concretos para a implementação de soluções efetivas, alinhadas com as necessidades e expectativas dos comensais. Basicamente o lean thinking é: identificação do gargalo/problema, identificação de sua causa raiz e resolução do problema com base na eliminação de desperdícios. Esses desperdícios são os mais diversos, como o tempo de espera ou a demora para realizar o pagamento, que geram os gargalos. Ou seja, os desperdícios geram os gargalos. Não precisa saber mais que isso sobre lean)

Além disso, o *NetLogo*, como ferramenta de modelagem e simulação, oferece recursos visuais para representar o comportamento da fila de forma clara e intuitiva. Isso facilita a compreensão dos resultados por parte de diferentes indivíduos, como gestores, funcionários e alunos, promovendo o engajamento e a tomada de decisões mais assertivas.

Portanto, a simulação da fila de atendimento utilizando o *NetLogo* apresenta-se como uma ferramenta importante para uma análise profunda e controlada dos fatores que impactam a experiência do usuário, contribuindo para uma gestão eficiente do restaurante e possibilitando a proposição de melhorias efetivas, reduzindo o risco de intervenções malsucedidas.

2. Entidades, variáveis de estado e escalas

Questão: Que tipos de entidades estão no modelo? Por quais variáveis ​​de estado, ou atributos, essas entidades são caracterizadas? Quais são as resoluções e extensões temporais e espaciais do modelo?

**Respostas:**

* **Entidades:**
  + **Estudantes:** Representam os indivíduos que passam pelos diferentes estágios do sistema.
  + **Estágios:** Representam os diferentes patches (ENTRADA, PAGAMENTO, B1 a B7, SAIDA) pelos quais os estudantes passam.
* **Variáveis de Estado:**
  + **Estudantes:**
    - stage: O estágio atual onde o estudante se encontra.
    - time-in-stage: O tempo que o estudante passou no estágio atual.
    - next-stage: O próximo estágio para onde o estudante se moverá
    - total-time: O tempo total que o estudante passou desde o estágio ENTRADA.
  + **Patches:**
    - pcolor: Cor do patch representando o estágio.
    - plabel: Nome do estágio para exibição.
  + **Globals**
    - stage: Identificador do estágio representado pelo patch.
    - students-count: Contagem de estudantes presentes no estágio.
    - stage-names: Nomes dos estágios para exibição
    - total-times: Lista para armazenar os tempos totais gastos em cada estágio
    - total-counts: Lista para contar quantos estudantes passaram por cada estágio
    - average-times: Lista para armazenar os tempos médios de cada estágio
    - total-students: Total de estudantes que passaram pelo sistema
    - completion-times: Lista para armazenar os tempos de conclusão de cada estudante
    - student-average-times: Lista para armazenar os tempos médios de cada estudante
    - std-student-times: Lista de desvios padrão de cada estudante.
* **Temporal Resolution and Extent:**
  + O modelo avança em passos discretos de tempo (ticks).
  + Cada tick representa um pequeno incremento de tempo (ex: 1 segundo).
  + A simulação continua até que todos os estudantes tenham passado pelo estágio SAIDA.
  + A simulação continua até que todos os estudantes tenham passado pelo sistema.
  + Não há um limite de tempo pré-definido; a duração depende do número de estudantes e da eficiência do sistema.
* **Resolução Espacial:**
  + O espaço é representado por patches, cada um correspondendo a um estágio específico.
  + Os estudantes se movem de um patch para outro, representando a transição entre estágios.

**3. Process overview and scheduling**

**Questão:** Quem (ou seja, que entidade) faz o quê e em que ordem? Quando as variáveis de estado são atualizadas? Como o tempo é modelado, como passos discretos ou como um continuum onde ambos os processos contínuos e eventos discretos podem ocorrer?

**Resposta:**

* **Setup:** Inicializa a simulação, criando os patches e os estudantes.
* **Create-sector:** Define os patches que representam cada estágio.
* **Create-public:** Cria os estudantes e os posiciona no estágio inicial (ENTRY).
* **Go:** Função principal de execução que move os estudantes e exibe os tempos médios.
* **Move-students:** Lógica de movimento dos estudantes entre os estágios.
* **Update-stage-time:** Atualiza o tempo total e a contagem de estudantes em cada estágio.
* **possibly-create-new-students:** Verifica a possibilidade de criar um estudante.
* **update-stage-time:** Controle para o incremento de tempo dos estágios.
* **Average-time:** Calcula o tempo médio para um dado estágio.
* **Display-average-times:** Exibe os tempos médios para todos os estágios.
* **random-triangular:** Calcula a triangular do tempo de cada estágio.

*Quem Faz o Quê e em Que Ordem?*

**1. Entidade Principal: Estudantes**

* **Ações dos Estudantes:**
  + **Mover-se entre estágios:** Os estudantes se movem de um estágio (patch) para o próximo com base no tempo passado no estágio atual e nas condições de disponibilidade do próximo estágio.
  + **Atualizar variáveis de estado:\*\* A cada tick, os estudantes atualizam seu `time-in-stage` e, ao mover-se, atualizam seu `stage` e `total-time`.**
  + **Concluir a passagem pelo sistema: Ao atingir o estágio final ("SAÍDA"), os estudantes calculam e registram seus tempos totais e médios.**

**2. Entidade Secundária: Patches (Estágios)**

* **Ações dos Patches:**
  + **Servir como estágios no sistema:** Cada patch representa um estágio específico onde os estudantes permanecem por um certo tempo antes de se moverem para o próximo estágio.
  + **Coletar dados de passagem:** Os patches registram o tempo total que cada estudante passa neles e quantos estudantes passam por cada estágio.

**3. Globais (Variáveis e Procedimentos)**

* **Ações Globais:**
  + **Configurar o ambiente:** Na inicialização, o ambiente é configurado com os patches representando os estágios e um grupo inicial de estudantes é criado.
  + **Atualizar métricas globais:** A cada tick, as variáveis globais que coletam e calculam dados agregados sobre os estudantes são atualizadas.
  + **Criar estudantes**: Com base em uma probabilidade predefinida, novos estudantes podem ser adicionados ao sistema a cada tick.

*Atualização das Variáveis de Estado*

**1. Estudantes:**

* **`stage`:** Atualizado quando um estudante se move para o próximo estágio.
* **`time-in-stage`:** Incrementado a cada tick enquanto o estudante está no mesmo estágio.
* **`next-stage`:** Determinado quando o estudante está pronto para se mover para o próximo estágio.
* **`total-time`:** Incrementado a cada tick, representando o tempo total no sistema.

**2. Globais:**

* **`total-times`, `total-counts`:** Atualizados quando um estudante se move para o próximo estágio.
* **`average-times`:** Calculados periodicamente com base nos `total-times` e `total-counts`.
* **`completion-times`, `student-average-times`:** Atualizados quando um estudante conclui sua passagem pelo sistema.
* **`std-student-times`:** Calculados com base nas variáveis de tempo dos estudantes.

*Modelagem do Tempo*

**1. Passos Discretos:**

* **Ticks:** O tempo no modelo é avançado em unidades discretas chamadas "ticks".
* **Eventos:** A cada tick, os estudantes podem se mover para o próximo estágio, e variáveis de estado são atualizadas.
* **Atualização das Variáveis:** A cada tick, todas as variáveis de estado dos estudantes são atualizadas, e as métricas globais são recalculadas conforme necessário.

**2. Eventos Discretos:**

* **Movimentação de Estudantes**: Ocorre de forma discreta quando os estudantes atendem às condições para mover-se para o próximo estágio.
* **Criação de Novos Estudantes**: Pode ocorrer a cada tick com base em uma probabilidade definida.

Em resumo, o modelo utiliza passos discretos (ticks) para avançar o tempo, e a cada tick, todas as ações e atualizações de variáveis de estado ocorrem de forma síncrona.

**4. Design concepts**

*Design Concept para a Simulação de Fila*

Processos:

* Movimento dos Estudantes: Os estudantes se movem de um estágio para o próximo, dependendo de certas condições.
* Atualização de Tempos: A cada tick, o tempo no estágio e o tempo total dos estudantes são atualizados.
* Criação de Novos Estudantes: Novos estudantes podem ser adicionados ao sistema com uma certa probabilidade.

Concepção e Métodos de Implementação:

* Inicialização: Configura o ambiente inicial, define os patches para cada estágio e cria um conjunto inicial de estudantes.
* Loop Principal: Controla a movimentação dos estudantes, a atualização de tempos e a possível criação de novos estudantes a cada tick.
* Movimentação Condicional: Os estudantes se movem para o próximo estágio se não houver congestionamento e se o tempo no estágio atual for suficiente.

Sazonalidade e Variação Estocástica:

* Sazonalidade: Não foi especificada sazonalidade no modelo, mas poderia ser incorporada ajustando a probabilidade de chegada de novos estudantes ao longo do tempo.
* Variação Estocástica: A criação de novos estudantes e o tempo gasto em cada estágio são modelados estocasticamente usando uma distribuição triangular.

Observação:

* Métricas: Tempos totais e médios nos estágios, número de estudantes em cada estágio, tempos de conclusão dos estudantes.
* Visualização: O modelo exibe os tempos médios de espera e de conclusão, além da distribuição dos estudantes pelos estágios.

*Implementação dos Conceitos de Design*

Inicialização do Modelo:

Função inicializar\_ambiente():

Limpar todas as variáveis e o ambiente

Configurar patches para cada estágio

Inicializar variáveis globais com valores iniciais

Criação de Estudantes:

Função criar\_estudantes(tamanho):

Para i de 1 até tamanho:

Criar estudante

Definir posição inicial do estudante como entrada

Inicializar variáveis do estudante (estágio, tempo\_no\_estágio, próximo\_estágio, tempo\_total)

Movimentação e Atualização:

Função mover\_estudantes():

Para cada estudante:

Obter o índice do estágio atual do estudante

Incrementar o tempo total do estudante

Se o estudante está no último estágio:

Armazenar o tempo de conclusão do estudante

Calcular e armazenar o tempo médio do estudante

Incrementar o total de estudantes

Remover o estudante do ambiente

Caso contrário:

Definir o próximo estágio

Se as condições para mover são atendidas:

Mover o estudante para o próximo estágio

Atualizar o estágio atual do estudante

Reiniciar o tempo no estágio

Caso contrário:

Incrementar o tempo no estágio atual

Criação de Novos Estudantes:

Função possivelmente\_criar\_novos\_estudantes():

Se uma nova criação de estudante for determinada:

Chamar a função criar\_estudantes(1)

*Parametrização e Teste*

Parametrização:

* Parâmetros foram definidos com base em valores hipotéticos. Testes adicionais com dados reais podem ser realizados para ajustar os parâmetros.

Teste:

* Testes de validação do modelo podem incluir a verificação dos tempos médios calculados, análise de congestionamento em estágios e variação estocástica.

Esses conceitos de design garantem que o modelo é estruturado de maneira clara, eficiente e fácil de entender, permitindo ajustes e melhorias conforme necessário.

**5. Inicialização**

**Questões:** Qual é o estado inicial do mundo do modelo, ou seja, no tempo t = 0 de uma execução de simulação? Em detalhes, quantas entidades de que tipo existem inicialmente, e quais são os valores exatos de suas variáveis de estado (ou como foram definidos estocasticamente)? A inicialização é sempre a mesma ou pode variar entre simulações? Os valores iniciais são escolhidos arbitrariamente ou com base em dados?  
**Resposta:**

*No tempo t=0 de uma execução de simulação, o estado inicial do mundo do modelo é definido da seguinte maneira:*

* Patches:
  + Todos os patches são inicializados com a cor pcolor=66.
  + Os patches que representam os estágios (patches específicos) são rotulados com nomes de estágios específicos (ENTRADA, PAGAMENTO, B1, B2, B3, B4, B5, B6, B7, SAÍDA).
* Estudantes:
  + Um grupo inicial de estudantes é criado. A quantidade de estudantes é definida pela variável initial-group-size, que deve ser especificada antes da execução do modelo.
  + Cada estudante é inicializado com as seguintes variáveis de estado:
    - shape = "person": Forma dos estudantes.
    - xcor = -11, ycor = -16: Posição inicial dos estudantes.
    - stage = patch -10 -16: Estágio inicial definido para ser o patch que representa "ENTRADA".
    - time-in-stage = 0: Tempo inicial no estágio definido como zero.
    - next-stage = nobody: Próximo estágio inicialmente definido como nenhum.
    - total-time = 0: Tempo total inicialmente definido como zero.
* Globais:
  + stages: Lista de patches que representam os estágios, inicializada com patches específicos.
  + stage-names: Lista de nomes dos estágios para exibição.
  + total-times: Lista de tempos totais inicializados com zeros, com tamanho igual ao número de estágios.
  + total-counts: Lista de contagens de estudantes inicializados com zeros, com tamanho igual ao número de estágios.
  + average-times: Lista de tempos médios inicializados com zeros, com tamanho igual ao número de estágios.
  + total-students = 0: Contador de estudantes total inicializado como zero.
  + completion-times: Lista de tempos de conclusão inicializada vazia.
  + student-average-times: Lista de tempos médios dos estudantes inicializada vazia.
  + std-student-times = []: Lista de desvios padrão dos tempos dos estudantes, embora não esteja em uso no código refatorado.

*Em detalhes, quantas entidades de que tipo existem inicialmente, e quais são os valores exatos de suas variáveis de estado (ou como foram definidos estocasticamente)?*

Estudantes: Inicialmente, são criados initial-group-size estudantes, cada um com as variáveis de estado definidas conforme descrito acima. O valor de initial-group-size deve ser especificado antes da execução do modelo e determina quantos estudantes serão criados inicialmente.

Patches: Há 10 patches específicos que representam os estágios do sistema, cada um com um nome correspondente.

Variáveis Globais: Inicializadas conforme descrito acima, com listas apropriadas e contadores definidos para manter o estado da simulação.

*A inicialização é sempre a mesma ou pode variar entre simulações?*

A inicialização dos estudantes pode variar entre simulações, dependendo do valor de initial-group-size, que pode ser diferente a cada execução. Além disso, a criação estocástica de novos estudantes durante a simulação (possibly-create-new-students) e o tempo que os estudantes passam em cada estágio (random-triangular) introduzem variabilidade na simulação.

*Os valores iniciais são escolhidos arbitrariamente ou com base em dados?*

Os valores iniciais, como o número inicial de estudantes (initial-group-size), são definidos arbitrariamente pelo usuário antes da execução do modelo. A posição inicial dos estudantes e os tempos iniciais no estágio são fixos, mas o tempo que os estudantes passam em cada estágio é determinado estocasticamente usando uma distribuição triangular definida pela função random-triangular, que introduz variabilidade com base em parâmetros fornecidos (mínimo, máximo e modo).

**6. Dados de entrada**

**Questão:** O modelo usa entrada de fontes externas, como arquivos de dados ou outros modelos, para representar processos que mudam ao longo do tempo?

**Resposta**: O modelo não utiliza dados de entrada externos para representar processos que mudam ao longo do tempo.

**7. Submodelos**

**Questões:** Quais são, em detalhes, os submodelos que representam os processos listados em 'Visão geral do processo e agendamento'? Quais são os parâmetros do modelo, suas dimensões e valores de referência? Como os submodelos foram projetados ou escolhidos, e como foram parametrizados e testados?  
**Resposta:**

1. **Setup (Inicialização)**

   - **Função:** `setup`

   - **Descrição:**  Inicializa o ambiente e cria o grupo inicial de estudantes. Define patches para cada estágio e configura variáveis globais.

   - **Parâmetros:**

     - `initial-group-size`: Define o número inicial de estudantes.

     - `stages`, `stage-names`, `total-times`, `total-counts`, `average-times`, `total-students`, `completion-times`, `student-average-times`, `std-student-times`: Variáveis globais para controlar o estado da simulação.

2. **Criação dos Estágios**

   - **Função:** `create-sector`

   - **Descrição:** Rotula os patches que representam os estágios com os nomes apropriados.

   - **Parâmetros:**

     - `stages`: Lista de patches que representam os estágios.

     - `stage-names`: Lista de nomes dos estágios.

3. **Criação do Público (Estudantes)**

   - **Função:** `create-public`

   - **Descrição:** Cria os estudantes iniciais no estágio de entrada.

   - **Parâmetros:**

     - `initial-size`: Número de estudantes a serem criados inicialmente.

4. **Simulação Principal**

   - **Função:** `go`

   - **Descrição:** Controla o loop principal da simulação. Chama submodelos para mover os estudantes e possivelmente criar estudantes.

   - **Parâmetros:** Nenhum diretamente; depende de variáveis globais e do estado atual da simulação.

5. **Movimento dos Estudantes**

   - **Função:** `move-students`

   - **Descrição:** Move estudantes entre estágios e atualiza o tempo gasto em cada estágio.

   - **Parâmetros:**

     - `move-threshold`: Valor para determinar quando um estudante pode se mover para o próximo estágio (definido estocasticamente).

     - `stages`, `total-times`, `total-counts`, `completion-times`, `student-average-times`, `total-students`: Variáveis globais usadas para controlar o movimento e contabilização dos estudantes.

6. **Possível Criação de Novos Estudantes**

   - **Função:** `possibly-create-new-students`

   - **Descrição:** Possivelmente cria estudantes com base em uma probabilidade.

   - **Parâmetros:**

     - `new-student-prob`: Probabilidade de criação de novos estudantes.

7. **Atualização do Tempo nos Estágios**

   - **Função:** `update-stage-time`

   - **Descrição:** Atualiza o tempo total gasto e a contagem de estudantes em cada estágio.

   - **Parâmetros:**

     - `index`: Índice do estágio.

     - `time`: Tempo gasto no estágio.

8. **Cálculo de Tempo Médio**

   - **Função:** `average-time`

   - **Descrição:** Calcula o tempo médio gasto em um estágio.

   - **Parâmetros:**

     - `total-time-val`: Tempo total gasto no estágio.

     - `total-count-val`: Número total de estudantes que passaram pelo estágio.

9. **Exibição dos Tempos Médios**

   - **Função:** `display-average-times`

   - **Descrição:** Exibe os tempos médios calculados para cada estágio.

   - **Parâmetros:** Nenhum diretamente; usa variáveis globais para exibição dos resultados.

10. **Geração de Números Aleatórios Triangulares**

    - **Função:** `random-triangular`

    - **Descrição:** Gera números aleatórios usando uma distribuição triangular.

    - **Parâmetros:**

      - `min-val`: Valor mínimo da distribuição.

      - `max-val`: Valor máximo da distribuição.

      - `mode`: Valor mais provável (modo) da distribuição.

Parâmetros do Modelo, Dimensões e Valores de Referência

1. `**initial-group-size**`: Número inicial de estudantes. Valor de referência deve ser definido pelo usuário.

2. `**move-threshold**`: Limite para a movimentação dos estudantes para o próximo estágio. Exemplo de valor: `10.8`.

3. `**new-student-prob**`: Probabilidade de criação de novos estudantes. Exemplo de valor: `0.1`.

4. **Distribuição Triangular (usada em** `**random-triangular**`**):**

   - `min-val`: Valor mínimo. Exemplo: `0.1`.

   - `max-val`: Valor máximo. Exemplo: `11`.

   - `mode`: Valor mais provável. Exemplo: `10.8`.

*Design e Parametrização dos Submodelos*

Os submodelos foram projetados com base nos seguintes princípios:

- **Modularidade:** Cada função realiza uma tarefa específica e é chamada no momento apropriado do loop principal da simulação (`go`).

- **Estocasticidade:** Alguns processos são definidos estocasticamente para simular a variabilidade real, como o tempo gasto em estágios (`random-triangular`) e a criação de novos estudantes (`possibly-create-new-students`).

- **Atualização Incremental:** O tempo total e as contagens são atualizadas incrementando os valores conforme os estudantes se movem pelos estágios.

*Testes e Validação*

-**Testes Unitários:** Cada submodelo foi testado individualmente para garantir que funciona conforme esperado.

- **Validação:** A simulação foi rodada várias vezes para verificar se os resultados são consistentes e plausíveis. Comparações com dados empíricos (se disponíveis) ou análises estatísticas (ex.: médias, desvios padrão) podem ser usadas para validar o comportamento do modelo.