Estruturas de Kripke

Aula para disciplina de Métodos Formais

Gabriela Moreira

Departamento de Ciência da Computação - DCC
Universidade do Estado de Santa Catarina - UDESC

17 de março de 2025

Conteúdo

Outline

Sistema de transições: Definição

- Abstrações que descrevem o comportamento de sistemas com precisão matemática e sem ambiguidade (BAIER; KATOEN, 2008) .
- Podem ser vistos como grafos dirigidos onde
  - Os nós são **estados**
  - As arestas são **transições**

Um **estado** descreve as informações de um sistema em um momento específico.

Uma **transição** descreve como um sistema pode mudar de um estado para outro.

Sistema de transições: Definição formal

Um sistema de transições é definido pela tripla $(S, \rightarrow, I)$ onde

- S é um conjunto de estados,
- $\rightarrow \subseteq S \times S$ é uma relação de transições, e
- $I \subseteq S$ é um conjunto de estados iniciais.

Um **comportamento** ou **execução** ρ de um sistema de transições é uma sequência de estados tal que

$$\rho = s_0, s_1, \ldots \ \ \textit{tal que } s_i \rightarrow s_{i+1} \textit{ para todo } i \geq 0$$

Sistemas de transições finito

Um sistema de transições é dito **finito** se e somente se S é finito.

Pergunta: Comportamentos de sistemas de transições finitos são sempre finitos?

Determinismo e Não-Determinismo

O conjunto de **sucessores** de um estado s é definido por $Post(s) = \{s' \in S \mid s \rightarrow s'\}$.

Um sistema de transições é dito **determinístico** se e somente se $|I| \leq 1 \wedge \forall s \in S : |Post(s)| \leq 1$. Ou seja:

- Tem apenas um estado inicial, e
- Todo estado tem, no máximo, um sucessor.

Não-Determinismo acontece quando há múltiplos estados iniciais $|I| > 1$ ou múltiplos sucessores para o mesmo estado ($|Post(s)| > 1$).

Outline

Estruturas de Kripke

Estruturas de Kripke são um tipo de sistema de transições com uma restrição adicional:

A relação $\rightarrow$ deve ser total

ou seja

$$\forall s \in S, \exists s' \in S : s \rightarrow s'$$

Estruturas de Kripke

Estruturas de Kripke são um tipo de sistema de transições com uma restrição adicional:

A relação $\rightarrow$ *deve ser total*

ou seja

$\forall s \in S, \exists s' \in S : s \rightarrow s'$

Estados terminais

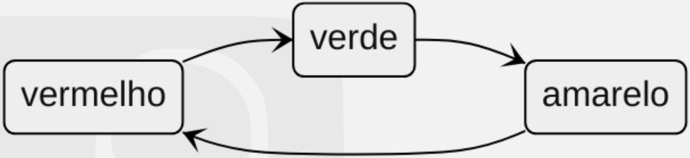
Em sistemas de transições, um estado é dito terminal se $Post(s) = \emptyset$.

Isso não é possível em estruturas de Kripke, devido a restrição acima. Em estruturas de Kripke, estados terminais são aqueles que possuem apenas transições para si mesmo, ou seja $Post(s) = \{s\}$

Exemplo: Semáforo

Um semáforo pode ser representado por uma estrutura de Kripke $(S, \to, I)$ onde

- $S = \{verde, amarelo, vermelho\}$
- $\to = \{verde \to amarelo, amarelo \to vermelho, vermelho \to verde\}$
- $I = \{vermelho\}$



Exercício: 🚦🚦 Dois semáforos

Um sistema com **dois** semáforos pode ser representado por uma estrutura de Kripke $(S, \rightarrow, I)$ onde ...

Exercício: 🚦🚦🚦 Três semáforos

Exercício: 🚦🚦🚦 Três semáforos

- Vish!

Exercício: 🚦🚦🚦 Três semáforos

- Vish!
- Muita coisa pra escrever, certo?
- Um jeito melhor: linguagens de especificação

Exercício: 🚦🚦🚦 Três semáforos

- Vish!
- Muita coisa pra escrever, certo?
- Um jeito melhor: linguagens de especificação

Vamos perceber algumas generalizações

1. Cada semáforo deve iniciar vermelho, e fazer o caminho vermelho $\rightarrow$ verde $\rightarrow$ amarelo enquanto os outros permanecem vermelhos.
2. Quando um semáforo fecha, queremos que **outro** semáforo abra
   - Com três semáforos, deve haver um revezamento que garanta que cada um vai abrir de vez em quando.

N semáforos em TLA+

MODULE *Semaforos*

EXTENDS *Integers*, *FiniteSets*
VARIABLE *cores*, *proximo*
CONSTANT *SEMAFOROS*

$$
\begin{aligned}
FicaVerde(s) \triangleq\ & \land\ proximo = s \\
& \land\ \forall\, s2 \in SEMAFOROS : cores[s2] = \text{“vermelho”} \\
& \land\ cores' = [cores \text{ EXCEPT } ![s] = \text{“verde”}] \\
& \land\ proximo' = (s + 1)\%Cardinality(SEMAFOROS)
\end{aligned}
$$

$$
\begin{aligned}
FicaAmarelo(s) \triangleq\ & \land\ cores[s] = \text{“verde”} \\
& \land\ cores' = [cores \text{ EXCEPT } ![s] = \text{“amarelo”}] \\
& \land\ \text{UNCHANGED } \langle proximo \rangle
\end{aligned}
$$

$$
\begin{aligned}
FicaVermelho(s) \triangleq\ & \land\ cores[s] = \text{“amarelo”} \\
& \land\ cores' = [cores \text{ EXCEPT } ![s] = \text{“vermelho”}] \\
& \land\ \text{UNCHANGED } \langle proximo \rangle
\end{aligned}
$$

$$Init \triangleq cores = [s \in SEMAFOROS \mapsto \text{“vermelho”}] \land proximo = 0$$

$$Next \triangleq \exists\, s \in SEMAFOROS : FicaVerde(s) \lor FicaAmarelo(s) \lor FicaVermelho(s)$$

$$Spec \triangleq Init \land \Box[Next]_{\langle cores,\, proximo \rangle}$$

N semáforos em Quint

Especificação completa no GitHub.

```
module semaforos {
  type Cor = Vermelho | Verde | Amarelo
  type Semaforo = int

  var cores: Semaforo -> Cor
  var proximo: Semaforo

  const SEMAFOROS: Set[Semaforo]

  action fica_verde(s: Semaforo): bool = all {
    proximo == s,
    SEMAFOROS.forall(s2 => cores.get(s2) == Vermelho),

    cores' = cores.set(s, Verde),
    proximo' = (s + 1) % SEMAFOROS.size(),
  }

  ...
```

Perguntas

1. Todos os exemplos de semáforos (1, 2, 3 e N) são sistemas de transições. Quais deles são Estruturas de Kripke?

Perguntas

1. Todos os exemplos de semáforos (1, 2, 3 e N) são sistemas de transições. Quais deles são Estruturas de Kripke?
   - 💡 Para um sistema de transições ser uma estrutura de Kripke, $\rightarrow$ deve ser total:
     - $\forall s \in S, \exists s' \in S : s \rightarrow s'$

Perguntas

1. Todos os exemplos de semáforos (1, 2, 3 e N) são sistemas de transições. Quais deles são Estruturas de Kripke?
   - 💡 Para um sistema de transições ser uma estrutura de Kripke, $\rightarrow$ deve ser total:
     - $\forall s \in S, \exists s' \in S : s \rightarrow s'$
2. Os sistemas de semáforos são finitos?

Perguntas

1. Todos os exemplos de semáforos (1, 2, 3 e N) são sistemas de transições. Quais deles são Estruturas de Kripke?
    - Para um sistema de transições ser uma estrutura de Kripke, $\rightarrow$ deve ser total:
        - $\forall s \in S, \exists s' \in S : s \rightarrow s'$
2. Os sistemas de semáforos são finitos?
    - Um sistema de transições é dito **finito** se e somente se S é finito.

Perguntas

1. Todos os exemplos de semáforos (1, 2, 3 e N) são sistemas de transições. Quais deles são Estruturas de Kripke?
   - 💡 Para um sistema de transições ser uma estrutura de Kripke, $\rightarrow$ deve ser total:
     - $\forall s \in S, \exists s' \in S : s \rightarrow s'$
2. Os sistemas de semáforos são finitos?
   - 💡 Um sistema de transições é dito **finito** se e somente se S é finito.
3. Nossas definições de semáforo são determinísticas?

Perguntas

1. Todos os exemplos de semáforos (1, 2, 3 e N) são sistemas de transições. Quais deles são Estruturas de Kripke?
   - 💡 Para um sistema de transições ser uma estrutura de Kripke, $\rightarrow$ deve ser total:
     - $\forall s \in S, \exists s' \in S : s \rightarrow s'$
2. Os sistemas de semáforos são finitos?
   - 💡 Um sistema de transições é dito **finito** se e somente se S é finito.
3. Nossas definições de semáforo são determinísticas?
   - 💡 O conjunto de **sucessores** de um estado s é definido por $Post(s) = \{s' \in S \mid s \rightarrow s'\}$.
   - 💡 Sistema é deterministico sse $|I| \leq 1 \wedge \forall s \in S : |Post(s)| \leq 1$

Outline

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \rightarrow s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \land \forall s \in S : |Post(s)| \leq 1$

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \to s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \wedge \forall s \in S : |Post(s)| \leq 1$

1. Qualquer estado pode ser um estado inicial. Se definirmos isso ($I = S$), temos não determinismo

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \rightarrow s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \wedge \forall s \in S : |Post(s)| \leq 1$

1. Qualquer estado pode ser um estado inicial. Se definirmos isso ($I = S$), temos não determinismo
    - $|I| \leq 1$ não é satisfeito

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \to s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \wedge \forall s \in S : |Post(s)| \leq 1$

1. Qualquer estado pode ser um estado inicial. Se definirmos isso ($I = S$), temos não determinismo
   - $|I| \leq 1$ não é satisfeito
2. Caso o primeiro semáforo a abrir não esteja definido

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \to s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \land \forall s \in S : |Post(s)| \leq 1$

1. Qualquer estado pode ser um estado inicial. Se definirmos isso ($I = S$), temos não determinismo
   - $|I| \leq 1$ não é satisfeito
2. Caso o primeiro semáforo a abrir não esteja definido
   - $|Post(1 : vermelho\ e\ \ldots\ e\ N : vermelho\ e\ próximo : indefinido)| \leq 1$ não é satisfeito

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \to s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \land \forall s \in S : |Post(s)| \leq 1$

1. Qualquer estado pode ser um estado inicial. Se definirmos isso ($I = S$), temos não determinismo
   - $|I| \leq 1$ não é satisfeito
2. Caso o primeiro semáforo a abrir não esteja definido
   - $|Post(1 : vermelho \text{ } e \text{ } ... \text{ } e \text{ } N : vermelho \text{ } e \text{ } próximo : indefinido)| \leq 1$ não é satisfeito
3. Caso a definição de próximo seja removida

Não determinismo nos semáforos

Como seriam semáforos com não determinismo?

- $Post(s) = \{s' \in S \mid s \to s'\}$.
- Sistema é deterministico sse $|I| \leq 1 \land \forall s \in S : |Post(s)| \leq 1$

1. Qualquer estado pode ser um estado inicial. Se definirmos isso ($I = S$), temos não determinismo
   - $|I| \leq 1$ não é satisfeito
2. Caso o primeiro semáforo a abrir não esteja definido
   - $|Post(1 : vermelho \text{ e } \ldots \text{ e } N : vermelho \text{ e } próximo : indefinido)| \leq 1$ não é satisfeito
3. Caso a definição de próximo seja removida
   - $|Post(1 : vermelho \text{ e } \ldots \text{ e } N : vermelho)| \leq 1$ não é satisfeito

Não determinismo na realidade

Onde podemos encontrar não determinismo em sistemas de software?

Definindo a fronteira

Ao especificar um sistema, especialmente quando há não determinismo, é preciso definir uma fronteira.

- Até aquela fronteira, fatores externos não especificados determinam o que acontece.
- O não determinismo é uma forma de abstrair esses fatores externos
  - i.e. De A, vou pra B ou C. Isso depende de algum fator externo. Se é a jogada de um dado ou o input de um usuário, não me importa.
  - Se isso me importa, então vou modelar a jogada de dado.

Exemplo: Notas de alunos

Vamos considerar duas fronteiras diferentes:

1. O professor coloca uma nota no SIGA. Se a nota for ≥ 7, o aluno passa.
    - Não determinismo no input do professor

Exemplo: Notas de alunos

Vamos considerar duas fronteiras diferentes:

1. O professor coloca uma nota no SIGA. Se a nota for ≥ 7, o aluno passa.
   - Não determinismo no input do professor
2. O aluno pode ou não prestar atenção nas aulas. Se prestar atenção, vai se dar bem na prova, sua nota será maior que 7, e portanto vai passar.
   - Não determinismo nas escolhas do aluno
   - A nota que o professor dá é **determinada** pelas escolhas do aluno

Exemplo: Notas de alunos

Vamos considerar duas fronteiras diferentes:

1. O professor coloca uma nota no SIGA. Se a nota for ≥ 7, o aluno passa.
   - Não determinismo no input do professor
2. O aluno pode ou não prestar atenção nas aulas. Se prestar atenção, vai se dar bem na prova, sua nota será maior que 7, e portanto vai passar.
   - Não determinismo nas escolhas do aluno
   - A nota que o professor dá é **determinada** pelas escolhas do aluno

No caso (2) estamos detalhando mais o mundo externo fora do SIGA, enquanto no (1) a fronteira é na interface do SIGA.

Exemplo: Notas de alunos

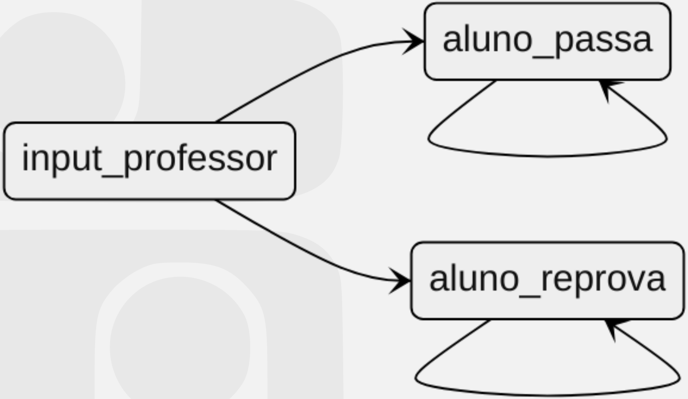
Vamos considerar duas fronteiras diferentes:

1. O professor coloca uma nota no SIGA. Se a nota for ≥ 7, o aluno passa.
   - Não determinismo no input do professor
2. O aluno pode ou não prestar atenção nas aulas. Se prestar atenção, vai se dar bem na prova, sua nota será maior que 7, e portanto vai passar.
   - Não determinismo nas escolhas do aluno
   - A nota que o professor dá é **determinada** pelas escolhas do aluno

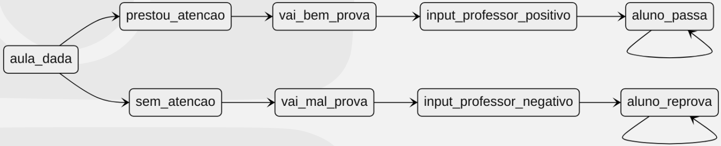
No caso (2) estamos detalhando mais o mundo externo fora do SIGA, enquanto no (1) a fronteira é na interface do SIGA.

O caso (1) é uma especificação do SIGA, enquanto o (2) fala mais sobre um sistema universitário.

Exemplo: Notas de alunos - input professor



Exemplo: Notas de alunos - escolhas dos alunos



Exemplo: Vôo com conexões

Versão 1:

Joinville $\rightarrow$ *São Paulo* $\rightarrow$ *Paris*

Exemplo: Vôo com conexões

Versão 1:

Joinville → São Paulo → Paris

Versão 2:

Check-in em Joinville → Despacho de Bagagem em Joinville → Check de Segurança em Joinville → Embarque em Joinville → Pouso em São Paulo → Check de Segurança em São Paulo → Embarque em São Paulo → Pouso em Paris → Retirada de bagagem em Paris

Exemplo: Vôo com conexões - Não determinismo

Onde poderia ter **não determinismo**?

- Chegar atrasado e perder o check-in
- Acharem uma bomba na bagagem
- Problemas técnicos no vôo
- Perder a conexão

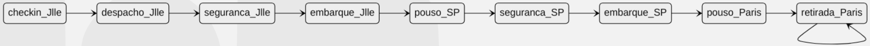
Exemplo: Vôo com conexões - Não determinismo

Onde poderia ter **não determinismo**?

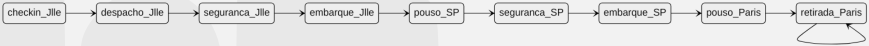
- Chegar atrasado e perder o check-in
- Acharem uma bomba na bagagem
- Problemas técnicos no vôo
- Perder a conexão

Podemos ter não determinismo em cada estado. Nos casos listados, podemos ou não determinar o que acontece. Cabe ao nível de detalhe, ou a **fronteira** da nossa modelagem.

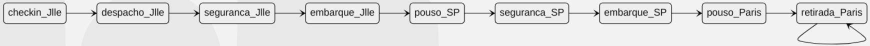
Exemplo: Vôo com conexões - Perdendo a conexão

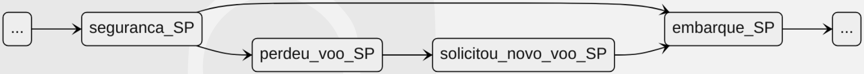


Exemplo: Vôo com conexões - Perdendo a conexão



Exemplo: Vôo com conexões - Perdendo a conexão





Referências

BAIER, C.; KATOEN, J.-P. **Principles of model checking**. Cambridge, MA: The MIT Press, 2008.

Estruturas de Kripke

Aula para disciplina de Métodos Formais

Gabriela Moreira

Departamento de Ciência da Computação - DCC
Universidade do Estado de Santa Catarina - UDESC

17 de março de 2025