Autômatos de Pilha

Definição

Movimentações do PDA

Linguagens do PDA

PDA's Determinístico

Tradução dos slides do Prof. Jeffrey D. Ullman (Stanford University)

Autômato de Pilha

- AS CFG's podem ser convertidas em Autômatos de Pilha (PDA) equivalentes e *vice-versa*, que aceitam as linguagens livres de contexto.
- Somente PDA não-determinístico define todas as CFL's.
- Mas a versão determinística modela um analisador sintático.
  - Muitas linguagens de programação podem ser reconhecidas por PDA's determinístico.

PDA: Definição Informal

- Pense em um ϵ-NFA com o poder adicional para manipular uma pilha.
- Movimentações são determinadas por:
  1. O estado atual (do seu "NFA"),
  2. O símbolo de entrada (ou ϵ), e
  3. O símbolo presente no topo da pilha.

PDA: Definição Informal

- Sendo não-determinístico, o PDA pode ter uma escolha da próxima movimentação.
- Em cada escolha, o PDA pode:
  1. Mudar o estado, e também
  2. Substituir o símbolo no topo da pilha por uma sequência de zero ou mais símbolos.
     - Zero símbolos (ϵ)= extração (“pop”) da pilha
     - Um símbolo = altera o topo da pilha
     - Dois ou mais símbolos = altera o topo da pilha e inseri um ou mais novos símbolos na pilha (“push”)

PDA: Definição Formal

- Um PDA é descrito por:
  1. Um conjunto finito de *estados* (Q).
  2. Um conjunto finito de *símbolos de entrada* (Σ).
  3. Um *alfabeto da pilha* finito (Γ).
  4. Uma *função de transição* (δ).
  5. Um *estado inicial* (q_0, em Q).
  6. Um *símbolo de início* (Z_0, em Γ).
  7. Um conjunto de *estados finais* ($F \subseteq Q$).

Convenções

- a, b, … são símbolos de entrada.
  - às vezes permitimos ϵ como um possível valor.
- …, X, Y, Z são símbolos da pilha.
- …, w, x, y, z são strings de símbolos de entrada.
- α, β,… são strings de símbolos da pilha.

A Função de Transição

- Toma três argumentos:
  1. Um estado q, em Q.
  2. Um símbolo de entrada a, em Σ ou a=ϵ.
  3. Um símbolo da pilha X, em Γ.
- $\delta(q, a, X)$ é um conjunto de zero ou mais pares da forma (p, α).
  - p é o novo estado; α é o string de símbolos da pilha.

Ações de um PDA

- Se $\delta(q, a, X)$ contêm (p, α) entre suas ações, então uma ação que o PDA pode fazer no estado q, com *a* de entrada, e X no topo da pilha é :

1. Mudar o estado para p.
2. Remover *a* da frente da entrada (mas *a* pode ser ϵ).
3. Substituir X no topo da pilha por α.

Exemplo: PDA

- Monte um PDA que aceite $\{0^n1^n \mid n \geq 1\}$.
- Os estados:
  - q = estado inicial. Estamos no estado q se vimos somente 0's até o momento.
  - p = vimos pelo menos um 1 e agora podemos proceguir somente se as entradas são 1's.
  - f = estado final; aceitar.

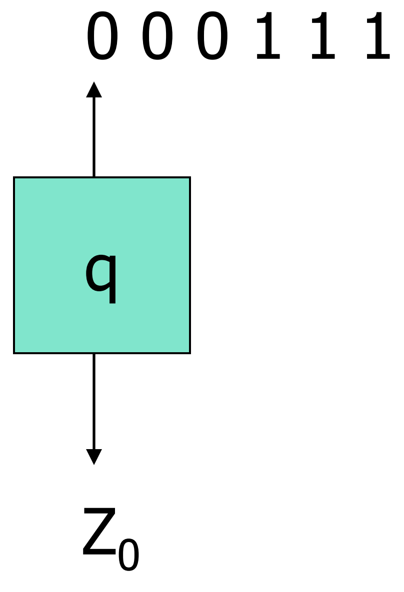
Exemplo: PDA – (2)

- Os símbolos da pilha:
  - Z_0 = símbolo inicial. Também marca a parte inferior da pilha, assim sabemos quando contamos o mesmo número de 1’s e 0’s.
  - X = marcador, usado para contar o número de 0’s visto na entrada.

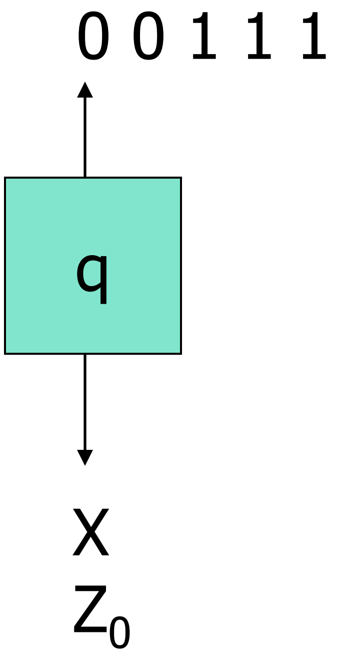
Exemplo: PDA – (3)

- As transições:
  - $\delta(q, 0, Z_0) = \{(q, XZ_0)\}$.
  - $\delta(q, 0, X) = \{(q, XX)\}$.
    - Estas duas regras produzem um X a ser "empurrado" na pilha para cada 0 lido da entrada.
  - $\delta(q, 1, X) = \{(p, \epsilon)\}$. Quando vê um 1, vai para o estado p e "extrai" um X.
  - $\delta(p, 1, X) = \{(p, \epsilon)\}$. "Extrair" um X por 1.
  - $\delta(p, \epsilon, Z_0) = \{(f, Z_0)\}$. Aceitar na pilha vazia.

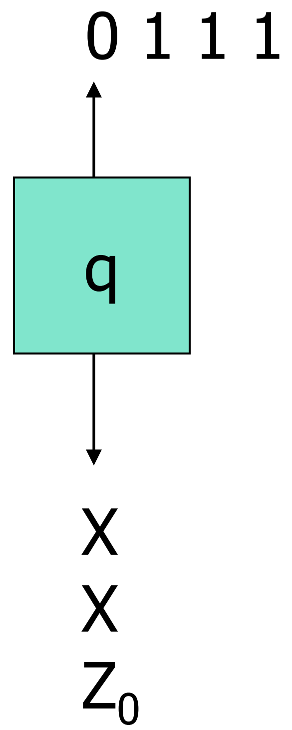
Ações do Exemplo PDA



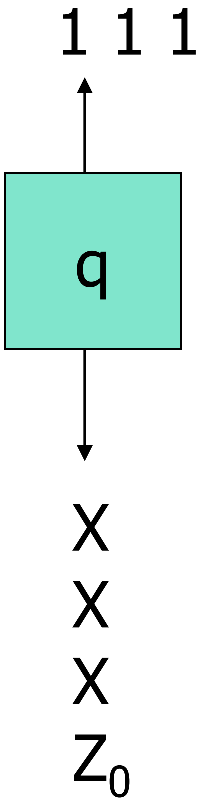
Ações do Exemplo PDA



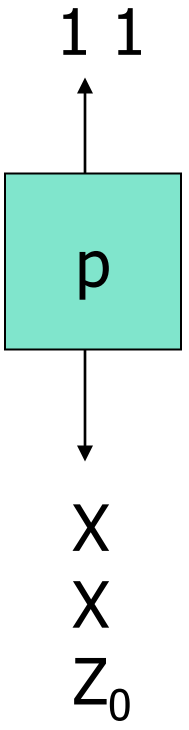
Ações do Exemplo PDA



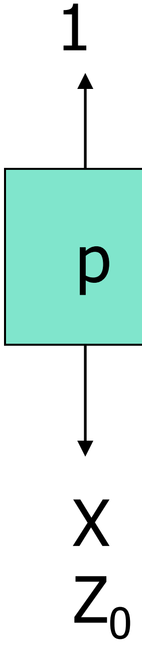
Ações do Exemplo PDA



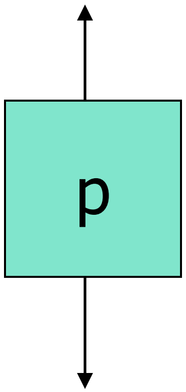
Ações do Exemplo PDA



Ações do Exemplo PDA

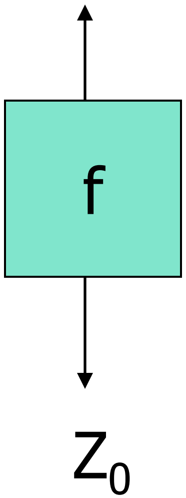


Ações do Exemplo PDA



Z_0

Ações do Exemplo PDA



Descrições Instantâneas

- Podemos formalizar as figuras vistas com uma *descrição instantânea* (ID).
- Uma ID é uma tripla (q, w, α), onde:
  1. q é o estado.
  2. w é a parte restante da entrada.
  3. α é o conteúdo da pilha (topo na extremidade esquerda de α).

Notação de "Dedução"

- Para dizer que o ID I pode se tornar o ID J em um movimento do PDA, escrevemos $I \vdash J$.
- Formalmente, $(q, aw, X\alpha) \vdash (p, w, \beta\alpha)$ para algum w e α, se $\delta(q, a, X)$ contém (p, β).
- Extender $\vdash$ para $\vdash^*$, significa "zero ou mais movimentos do PDA".
  - Base: $I \vdash^* I$.
  - Indução: Se $I \vdash^* J$ e $J \vdash K$, então $I \vdash^* K$.

Exemplo: Dedução

- Usando o PDA do exemplo anterior, podemos descrever a sequência de movimento por: $(q, 000111, Z_0) \vdash (q, 00111, XZ_0) \vdash (q, 0111, XXZ_0) \vdash (q, 111, XXXZ_0) \vdash (p, 11, XXZ_0) \vdash (p, 1, XZ_0) \vdash (p, \epsilon, Z_0) \vdash (f, \epsilon, Z_0)$
- Assim, $(q, 000111, Z_0) \vdash^* (f, \epsilon, Z_0)$.
- O que aconteceria na entrada 0001111?

Resposta

Um PDA pode usar ϵ mesmo se resta entrada

- $(q, 0001111, Z_0) \vdash (q, 001111, XZ_0) \vdash (q, 01111, XXZ_0) \vdash (q, 1111, XXXZ_0) \vdash (p, 111, XXZ_0) \vdash (p, 11, XZ_0) \vdash (p, 1, Z_0) \vdash (f, 1, Z_0)$
- Observe o último ID não tem movimentos.
- 0001111 não é aceita, porque a entrada não é completamente consumida.

Notações para FA e PDA

- Representamos movimentos de um FA pelo δ extendido, o qual não menciona a entrada ainda a ser lida.
- Podemos escolher uma notação similar para PDA's, onde o estado do FA é substituído por uma combinação estado-pilha, como as figuras mostram.

Notações para FA e PDA

- Da mesma forma, podemos escolher uma notação FA com ID's.
  - Apenas remova o componente pilha.
- Por que a diferença? Teoria:
- FA tendem a modelar como protocolos, com entradas indefinidamente longas.
- PDA modela analisadores sintáticos, nos quais são dados programas fixos para processar.

Linguagem de um PDA

- A forma comum para definir a linguagem de um PDA é pelo *estado final*.
- Se P é um PDA, então L(P) é o conjunto de strings w tais que $(q_0, w, Z_0) \vdash^* (f, \epsilon, \alpha)$ para o estado final f e algum α.

Linguagem de um PDA – (2)

- Uma outra abordagem para definir a mesma linguagem de um PDA é por *pilha vazia*.
- Se P é um PDA, então N(P) é o conjunto de strings w tais que $(q_0, w, Z_0) \vdash^* (q, \epsilon, \epsilon)$ para algum estado q.

Equivalência de Definições de Linguagem

- L tem um PDA que a aceita pelo estado final se e somente se L tem um PDA que a aceita por pilha vazia.
  1. Se L = L(P), então existe um PDA P′ tal que L = N(P′).
  2. Se L = N(P), então existe um PDA P″ tal que L = L(P″).
- Porém, para um PDA P as linguagens que P aceita por pilha vazia e por estado final em geral são difirentes.

Prova: L(P) -> N(P’)

- P’ irá simular P.
- Se P aceita, P’ irá esvaziar sua pilha.
- P’ evita o esvaziamento acidental da pilha, utilizando um marcador especial de fundo da pilha para capturar o caso onde P esvaziou a pilha sem aceitar o string.

Prova: L(P) -> N(P')

- P' têm todos os estados, símbolos e movimentos de P, mais:
  1. Símbolo da pilha X_0, usado para guardar fundo da pilha contra esvaziamento acidental.
  2. Novo estado inicial s e um estado "apagar".
  3. $\delta(s, \epsilon, X_0) = \{(q_0, Z_0X_0)\}$. Começando de P.
  4. $\delta(f, \epsilon, X) = \delta(e, \epsilon, X) = \{(e, \epsilon)\}$ para algum estado final f de P e algum símbolo de pilha X.

Prova: N(P) -> L(P'')

- P'' simula P.
- P'' tem um especial marcador de fundo para capturar a situação onde P esvazia a pilha.
- Se assim for, P'' aceita.

Prova: N(P) -> L(P”)

- P” têm todos os estados, símbolos e movimentos de P, mais:
  1. Símbolo de pilha X_0, usado para guardar o fundo da pilha.
  2. Novo estado s e um estado final f.
  3. $\delta(s, \epsilon, X_0) = \{(q_0, Z_0X_0)\}$. Começando de P.
  4. $\delta(q, \epsilon, X_0) = \{(f, \epsilon)\}$ para algum estado q de P.

PDA's Deterministico

- Para ser determinístico, deve haver no máximo uma escolha de movimento para algum estado q, símbolo de entrada *a*, e símbolo de pilha X.
- Além disso, não deve haver uma escolha entre usar uma entrada ϵ ou uma entrada real.
- Formalmente, $\delta(q, a, X)$ e $\delta(q, \epsilon, X)$ não podem ser ambos não vazios.