PDA→MTTM

CSAS 4113 Final Project

Tijana Minic and Oliwia Kempinski

Seton Hall University

Design idea

- Represent the unconsumed input as tape 0 in the `MTTM`.

Design idea

- Represent the unconsumed input as tape 0 in the `MTTM`.
- Represent the stack as tape 1 in the `MTTM`.

Design idea

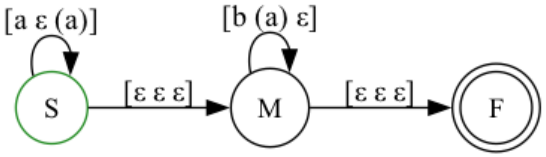
- Represent the unconsumed input as tape 0 in the `MTTM`.
- Represent the stack as tape 1 in the `MTTM`.
- Precondition of the `MTTM`:
  `tape 0 = (LM w) and t0h = 1`
  `tape 1 = (BLANK) and t1h = 0`

Design idea

- Represent the unconsumed input as tape 0 in the MTTM.
- Represent the stack as tape 1 in the MTTM.
- Precondition of the MTTM:
  tape 0 = (LM w) and t0h = 1
  tape 1 = (BLANK) and t1h = 0
- Let PDA = (make-ndpda K sig gam s F del).
  Let MTTM = (make-mttm K' sig' s F' del' a n)
  K' = K and
  new states needed for popping and pushing and
  a
  sig' = sig and
  gam
  F' = (list a)
  a = new accept state
  del' = transitions simulating transitions in del and
  transitions reading and writing blanks on all
  tapes from all states in F to a
  n = 2

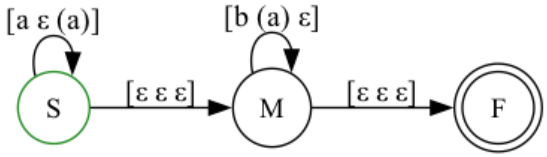
Example $a^n b^n$

- PDA:

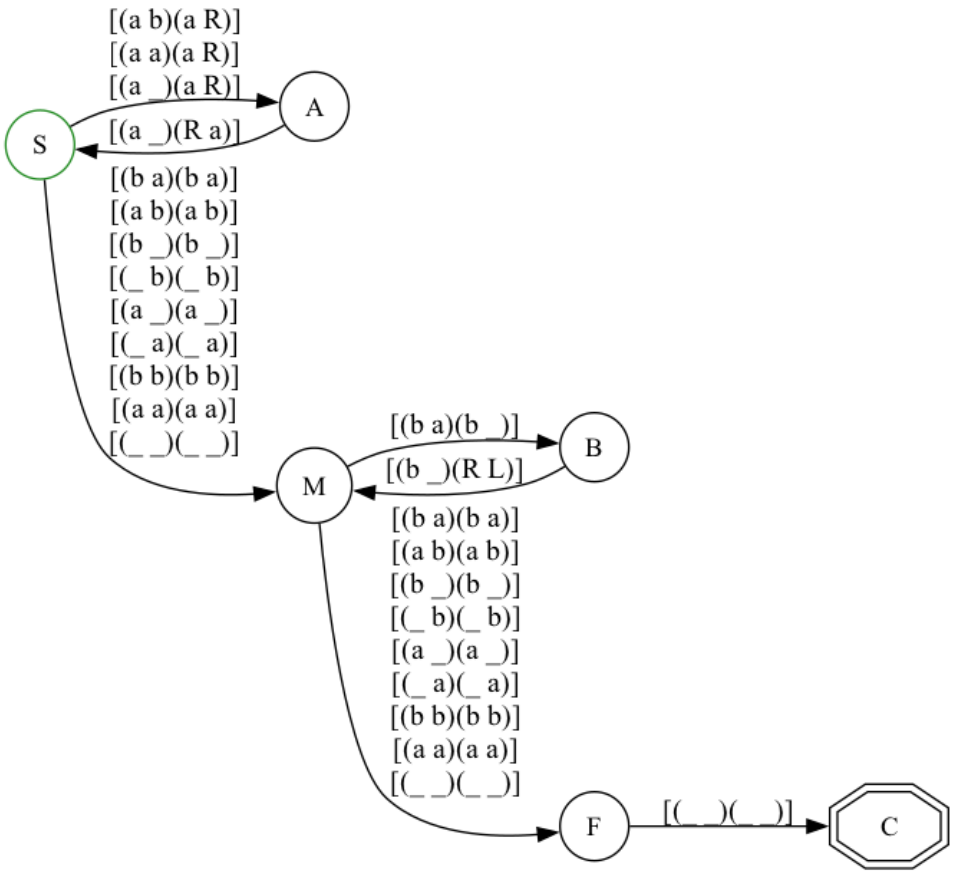


Example $a^n b^n$

- PDA:



- MTTM:



Constructor

```
;; pda -> mttm
;; Purpose: Given a pda, make an mttm
(define (pda->mttm p)
...
  (let* ((new-rules (new-rules-helper (sm-rules p) (sm-states p)))
         (new-states (get-states-from-mttm-rules new-rules))
         (new-final (gen-nt new-states))
         (new-rules2 (append (append-map
                              (lambda (x)
                                (list `((,x (,BLANK ,BLANK))
                                        (,new-final (,BLANK ,BLANK)))))
                              (sm-finals p)) new-rules)))
    (make-mttm (cons new-final new-states)
               (remove-duplicates (append (sm-sigma p) (sm-gamma p)))
               (sm-start p)
               (list new-final)
               new-rules2
               2
               new-final))
```

Reads nothing, pops nothing, pushes nothing

- tape 0: read tape element and write it.

Reads nothing, pops nothing, pushes nothing

- tape 0: read tape element and write it.
- tape 1: read tape element and write it.

Reads nothing, pops nothing, pushes nothing

- tape 0: read tape element and write it.
- tape 1: read tape element and write it.
- effect: stack and ui both remain unchanged.

Reads nothing, pops nothing, pushes nothing

```
;; pda-rule -> (listof mttm-rule)
;; Purpose: Make mttm rules for a pda rule that reads, pops,
            and pushes nothing
(define (new-empty-rules rule)
    (let* ((fromst (first (first rule)))
           (tost (first (second rule)))
           (sigma (cons BLANK (sm-sigma p)))
           (gamma (cons BLANK (sm-gamma p)))
           (new-reads-actions
            (append-map (lambda (x) (map (lambda (y) (list x y)) gamma)
      (map (lambda (x) `((,fromst ,x) (,tost ,x))) new-reads-actions)))
```

Reads something, pops nothing, pushes nothing

- tape 0: read `PDA` rule's element and move the head to the right.

Reads something, pops nothing, pushes nothing

- tape 0: read `PDA` rule's element and move the head to the right.
- tape 1: read tape element and write it.

Reads something, pops nothing, pushes nothing

- tape 0: read `PDA` rule's element and move the head to the right.
- tape 1: read tape element and write it.
- effect: stack is unchanged and the next element in ui has been consumed.

Reads something, pops nothing, pushes nothing

```
;; pda-rule -> (listof mttm-rule)
;; Purpose: Make mttm rules for a pda rule that only reads something

(define (new-read-rules rule)
  (let ((fromst (first (first rule)))
        (tost (first (second rule)))
        (read (second (first rule)))
        (sigma (cons BLANK (sm-sigma p))))
    (map (lambda (x) `((,fromst (,read ,x))
                       (,tost (R ,x)))) sigma)))
```

Reads nothing, pops something, pushes nothing

- create a new state for every element popped.

Reads nothing, pops something, pushes nothing

- create a new state for every element popped.
- tape 0: read the tape element and write the tape element.

Reads nothing, pops something, pushes nothing

- create a new state for every element popped.
- tape 0: read the tape element and write the tape element.
- tape 1: remove `PDA` rule's pop elements from the tape.
  - in source state read the element on the tape, write a blank, and transition to a new state.
  - move the head left and stay in the same state
  - repeat for all elements in the pop list

Reads nothing, pops something, pushes nothing

- create a new state for every element popped.
- tape 0: read the tape element and write the tape element.
- tape 1: remove `PDA` rule's pop elements from the tape.
  - in source state read the element on the tape, write a blank, and transition to a new state.
  - move the head left and stay in the same state
  - repeat for all elements in the pop list
  - on last popped element move to destination state
- effect: ui remains the same, elements from the pop list are removed from the stack.

Reads something, pops nothing, pushes nothing

Thank you for your attention!

Any questions?