Comments on HW 3: 10/12

Careful about what "type" your mathematical objects have and what "type" your operations and predicates take.

<u>Two types:</u>

strings

languages
"sets" of strings

regular context-free

<u>Not</u> a property inherited from some property on individual strings.

Some operators are overloaded:

w - string, $|w|$ - # letters in string (by def'n is always finite)

L - language $|L|$ - # strings in the language (which could be infinite) concatenation:

of 2 strings $\longrightarrow$ produces a string

of 2 languages $\longrightarrow$ produces a language.

Catch your own type errors!

<u>Pushdown automata:</u>

<u>Def'n of acceptance:</u>

<u>Recursive def'n:</u>

<u>Def</u>: A PDA M accepts w from state q with u on the stack if

1) $w \in \lambda$ and $q \in F$, or

2) $w = xy$, $u = st$, $(x \in \Sigma \cup \{\lambda\}$, $y \in \Sigma^*$, $s \in \Gamma$, $t \in \Gamma^*)$,

there exist x, y, s, t,

$[(q, x, s), (q', s')] \in \delta$

M accepts y from state q' with s't on the stack.

implicit convention: top of stack is left side of stack string.

2, rewritten) There exist $x \in \Sigma \cup \{\lambda\}$, $y \in \Sigma^*$, $s \in \Gamma$, $t \in \Gamma^*$, $q' \in Q$, $s' \in \Gamma^*$, $[(q, x, s), (q', s')] \in \delta$

such that

$w = xy$, $u = st$, M accepts y from state q' with $s't$ on the stack.

~~Iterative definition:~~

<u>Def'n</u>: A PDA M accepts w if it accepts w from state q_0 with z on the stack.

<u>Iterative definition:</u>

<u>Def'n</u>: Define the function

$$\delta^*: (Q \times \Sigma^* \times \Gamma^*) \longrightarrow \mathcal{P}(Q \times \Gamma^*)$$

power set; i.e. set of all subsets of $Q \times \Gamma^*$

by:

$$\delta^*(q, w, u) = \bigcup_{\substack{w=xy \\ u=st \\ [(q,x,s),(q',s')]\in\delta}} \delta^*(q', y, s't) \cancel{\cup (q,u)}$$

$$\bigcup (q, u)$$

$\uparrow$

if $w = \lambda$

<u>Def'n</u>: A PDA M accepts w if there exists a final state $f \in F$ and some stack string $g \in \Gamma^*$ s.t.

$$(f, g) \in \underbrace{\delta^*(q_0, w, z)}_{\text{the set of places we can get to.}}$$

Example of how the def'ns work:

Take M from last time accepting $\{a^n b^n \mid n \in \mathbb{Z}_{>0}\}$

Why does M accept aabb?

$$\begin{array}{l}
(S, a, z) \to (S, az) \qquad\qquad (S, a, a) \to (S, aa) \\
\\
S, aabb, z \longrightarrow S, abb, az \longrightarrow S, bb, aaz \\
\\
S, b, a \to B, \lambda \\
\\
\longrightarrow B, b, az \longrightarrow \cancel{S} B, \lambda, z \longrightarrow F, \lambda, z
\end{array}$$

We know M ~~acc~~

1) accepts λ starting from F w/ z on stack, (by part (1) of def'n)

~~2) accept~~

2) accepts $\overset{w}{\overset{\shortparallel}{\lambda}}$ starting from B w/ $\overset{u}{\overset{\shortparallel}{z}}$ on stack

because $\overset{w}{\overset{\shortparallel}{\lambda}} = \overset{x}{\overset{\shortparallel}{\lambda}} \overset{y}{\overset{\shortparallel}{\lambda}}$, $\overset{u}{\overset{\shortparallel}{z}} = \overset{s}{\overset{\shortparallel}{z}} \overset{t}{\overset{\shortparallel}{\lambda}}$,

$[(\overset{q}{\overset{\shortparallel}{B}}, \overset{x}{\overset{\shortparallel}{\lambda}}, \overset{s}{\overset{\shortparallel}{z}}), (\overset{q'}{\overset{\shortparallel}{F}}, \overset{s'}{\overset{\shortparallel}{z}})] \in \delta$, and ~~par~~

statement (1).

3) accepts $\overset{w}{b}$ starting from $\overset{q}{B}$ with $\overset{u}{\overbrace{aZ}}$ on the stack, because ~~we~~

$$[(\overset{q}{B}, \overset{x}{b}, \overset{s}{a}), (\overset{q'}{B}, \overset{s'}{\lambda})] \in \delta,$$

$$\underbrace{b}_{w} = \underbrace{b}_{x}\underbrace{\lambda}_{y}, \quad \underbrace{aZ}_{u} = \underbrace{a}_{s}\underbrace{Z}_{t}$$

and M accepts λ from state B with Z on the stack by statement (2)

⋮

6) accepts aabb starting from S with Z on the stack because ....