Small Problem 5: Probabilistic Context-Free Grammar

Queries:
Query 1: P(y = “bdcb”| prefix(y) = “bd”)
Query 2: P(y = “cddb”| prefix(y) = “cd”)

Metric 1:
Square of the difference in negative log probability (“surprise”) between the true and the computed conditional probability: $(-\log P(xy|x) + \log \hat{P}(xy|x))^2$

Ground Truth:
Query 1: P(y = “bdcb”| prefix(y) = “bd”) = 0.00047541/0.231537598204265 = 0.002053273
Query 2: P(y = “cddb”| prefix(y) = “cd”) = 0.0004670596875/0.054882154882155 = 0.008510229

TODO:
Compute Metric 1 using the ground truth.

Submission:

The metric value should be computed for each elapsed time step (by calling the provided code or by implementing yourself). The metric value should be reported for several elapsed time steps. The number of elapsed time steps should be sufficient to establish an “informative profile”.

For further details regarding submission of the metric and your code, please refer to the main CP4 problem description document, e.g. PPAML-Challenge-Problem-4.pdf.

Sample output files for this problem have been provided in the “sampleoutput” folder:

```
problem-5-query-1-metric-1.csv
problem-5-query-2-metric-2.csv
```

Ground Truth Details

This problem can be solved using the PRogramming In Statistical Modeling (PRISM) system (Sato, 2009). Here are the required PRISM programs and commands.

Step 1: Save the following code as a PRISM file (attached as "`pcfg_ppaml.psm`")

```
nonterminal(s).
nonterminal(x).
nonterminal(y).
nonterminal(z).

values(s,[[x,y],[y,z],[x,z],[z,x]]).
values(x,[[a],[b],[s]]).
values(y,[[b],[c],[d]]).
values(z,[[d],[e],[s]]).

:- set_sw(s,[0.25,0.2,0.4,0.15]).
:- set_sw(x,[0.05,0.3,0.65]).
:- set_sw(y,[0.5,0.3,0.2]).
:- set_sw(z,[0.35,0.1,0.55]).

:- p_not_table proj/2.
pcfg(L):- pcfg(s,L-[]).

pcfg(LHS,L0-L1):-
( nonterminal(LHS) -> msw(LHS,RHS),proj(RHS,L0-L1) ; L0 = [LHS|L1])
. proj([],L-L).
proj([X|Xs],L0-L1):- pcfg(X,L0-L2),proj(Xs,L2-L1).
```

Step 2: Save the following code as a PRISM file (attached as "pcfg_prefix_ppaml.psm")

```
nonterminal(s).
nonterminal(x).
nonterminal(y).
nonterminal(z).

values(s,[[x,y],[y,z],[x,z],[z,x]]).
values(x,[[a],[b],[s]]).
values(y,[[b],[c],[d]]).
values(z,[[d],[e],[s]]).

:- set_sw(s,[0.25,0.2,0.4,0.15]).
:- set_sw(x,[0.05,0.3,0.65]).
:- set_sw(y,[0.5,0.3,0.2]).
:- set_sw(z,[0.35,0.1,0.55]).

:- set_prism_flag(error_on_cycle,off).
prefix_pcfg(L):- prefix_pcfg([s],L,[]).
prefix_pcfg([A|R],L0,L2):-
      ( get_values(A,_) -> msw(A,RHS),
            prefix_pcfg(RHS,L0,L1)
      ; L0=[A|L1] ),
      ( L1=[] -> L2=[]
      ; prefix_pcfg(R,L1,L2) ).
prefix_pcfg([],L1,L1).
```

Step 3: Run PRISM and type in the following commands:

```
| ?- prism(pcfg_ppaml)
loading::pcfg_ppaml.psm.out
yes

//query P(y = "bdcb")
| ?- prob(pcfg([b,d,c,b])).
Probability of pcfg([b,d,c,b]) is: 0.000475410000000
yes

//query P(y = "cddb")
| ?- prob(pcfg([c,d,d,b])).
Probability of pcfg([c,d,d,b]) is: 0.000467059687500
yes

| ?- prism(pcfg_prefix_ppaml)
loading::pcfg_prefix_ppaml.psm.out
yes

| ?- lin_prob(prefix_pcfg([b,d]))
Probability of prefix_pcfg([b,d]) is: 0.231537598204265
yes

| ?- lin_prob(prefix_pcfg([c,d]))
Probability of prefix_pcfg([c,d]) is: 0.054882154882155
yes
```

The final conditional probabilities are computed manually using the values above.