CSc 4330 Programming Language Concepts
Solutions to Practice for Exam 3, Spring 2018

1. Consider the following substitutions:

   $\sigma = \{X \leftarrow g(U), Y \leftarrow f(Z), V \leftarrow W, Z \leftarrow c\}$

   $\theta = \{Z \leftarrow f(U), W \leftarrow V, U \leftarrow b\}$

   Compute $\sigma\theta$ and $\theta\sigma$

   $\sigma\theta = \{X \leftarrow g(b), Y \leftarrow f(f(U)), Z \leftarrow c, W \leftarrow V, U \leftarrow b\}$

   $\theta\sigma = \{Z \leftarrow f(U), U \leftarrow b, X \leftarrow g(U), Y \leftarrow f(Z), V \leftarrow W\}$

2. Find the mgu, if any, for the following sets:

   `{ p(X, f(X)), p(Y, f(a)) }`

k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{p(X,f(X)),p(Y,f(a))}	{X,Y}
1	{X ← Y}	{p(Y,f(Y)),p(Y,f(a))}	{Y,a}
2	{X ← a, Y ← a }	{p(a,f(a))}	

   `{ p(a, X), p(X, f(X)) }`

k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{p(a,X),p(X,f(X))}	{a,X}
1	{X ← a}	{p(a,a),p(a,f(a))}	{a, f(a)}
2	NO UNIFIER		

{p(a,X,f(g(Y))), p(Z,f(Z),f(W))}

k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{p(a,X,f(g(Y))), p(Z,f(Z),f(W))}	{Z, a}
1	{Z ← a}	{p(a,X,f(g(Y))), p(a,f(a),f(W))}	{X, f(a)}
2	{Z ← a, X ← f(a)}	{p(a,f(a),f(g(Y))), p(a,f(a),f(W))}	{W, g(Y)}
3	{Z ← a, X ← f(a), W ← g(Y)}	{p(a,f(a),f(g(Y)))}	

{p(f(g(X,a)),X), p(Z,b)}

k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{p(f(g(X,a)),X), p(Z,b) }	{ f(g(X,a)), Z}
1	{Z ← f(g(X,a))}	{p(f(g(X,a)),X), p(f(g(X,a)),b)}	{X, b}
2	{Z ← f(g(b,a)), X ← b}	{p(f(g(b,a)),b)}	

{[[the,Y]|Z], [[A,hare],[is,here]]}

k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{ [[the, Y]|Z], [[A,hare],[is,here]] }	{the, A}
1	{A ← the}	{ [[the, Y]|Z], [[the,hare],[is,here]]}	{Y, hare}
2	{A ← the, Y ← hare}	{ [[the, hare]|Z], [[the,hare],[is,here]]}	{Z, [[is,here]]}
3	{X ← the, Y ← hare Z ← [[is,here]]}	{ [[the, hare],[is,here] }	

{[X,Y|Z], [mary,likes,wine]}

k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{[X,Y|Z], [mary,likes,wine]}	{mary, X}
1	{X ← mary}	{[mary,Y|Z], [mary,likes,wine]}	{Y, likes}
2	{X ← mary, Y ← likes}	{[mary,likes|Z], [mary,likes,wine]}	{Z, [wine]}
3	{X ← mary, Y ← likes Z ← [wine]}	{[mary,likes,wine]}	

{append([b],[c,d],L), append([X|Xs],Ys,[X|Zs])}

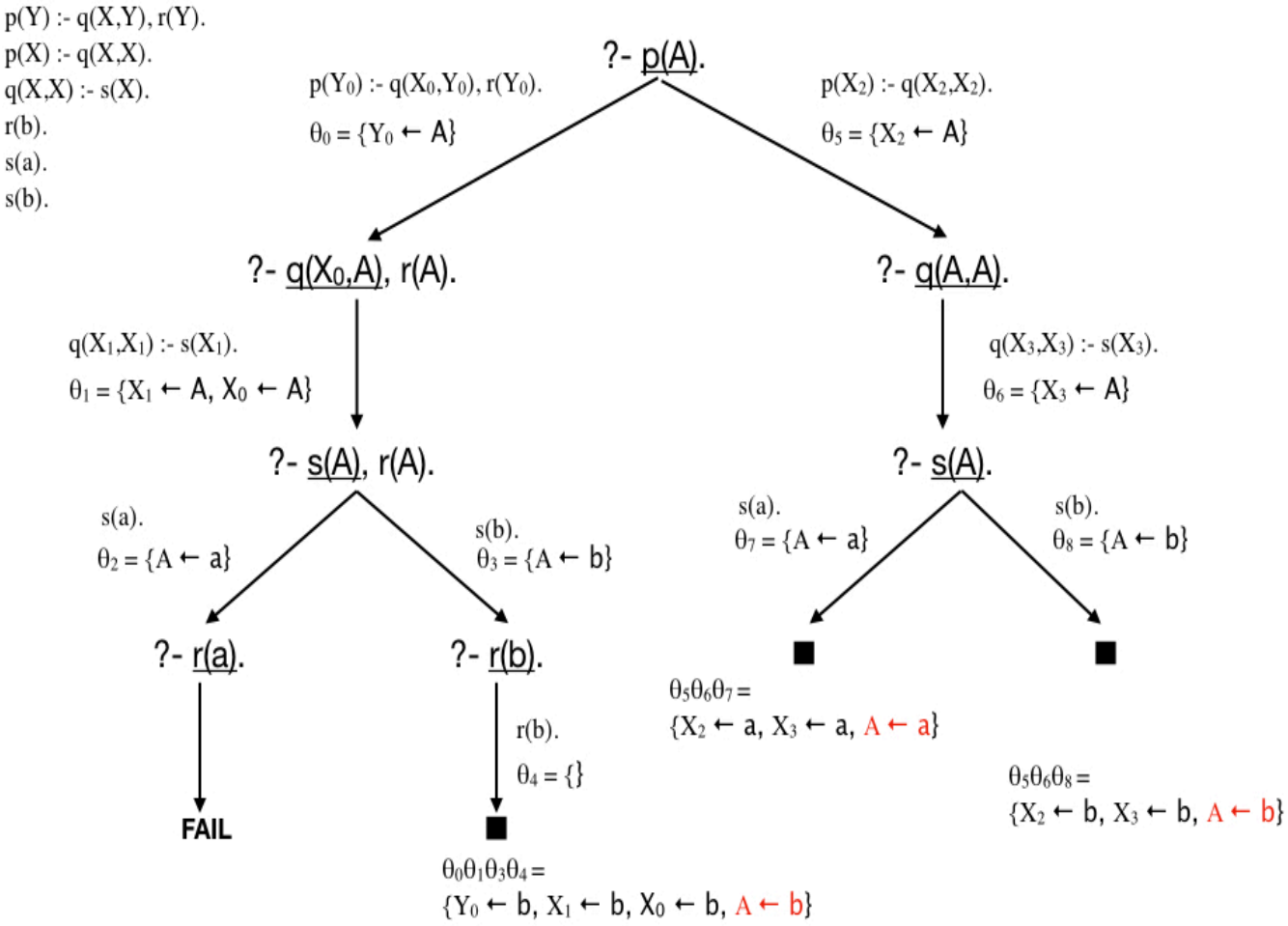
k	σ_k	$S\sigma_k$	Disagreement($S\sigma_k$)
0	{}	{ append([b],[c,d],L), append([X|Xs],Ys,[X|Zs]) }	{X, b}
1	{X ← b}	{ append([b],[c,d],L), append([b|Xs],Ys,[b|Zs]) }	{[], Xs}
2	{X ← b, Xs ← []}	{ append([b],[c,d],L), append([b],Ys,[b|Zs]) }	{[c,d], Ys}
3	{X ← b, Xs ← [] Ys ← [c,d]}	{ append([b],[c,d],L), append([b],[c,d],[b|Zs]) }	{L,[b|Zs]}
4	{X ← b, Xs ← [] Ys ← [c,d],L ← [b|Zs]}	{ append([b],[c,d],[b|Zs]) }	

3. Consider the following program:

```
p(Y) :- q(X,Y), r(Y).
p(X) :- q(X,X).
q(X,X) :- s(X).
r(b).
s(a).
s(b).
```

Draw the complete SLD-refutation tree for the goal:

```
?- p(A).
```



4. Write a Prolog program to pack consecutive elements in a list into lists. For example,

```
?- pack([a,a,a,a,b,c,c,a,a,d,e,e,e,e],X).
X = [[a,a,a,a],[b],[c,c],[a,a],[d],[e,e,e,e]]

pack([X],[[X]]).
pack([X|L],[[X|S]|T]) :-
  pack(L,[S|T]), member(X,S).
pack([X|L],[[X],S|T]) :-
  pack(L,[S|T]), \+ member(X,S).
```

and then use this program to find the run-lengths as follows:

```
?- encode([a,a,a,a,b,c,c,a,a,d,e,e,e,e],X).
X = [[4,a],[1,b],[2,c],[2,a],[1,d][4,e]]

encode(L,M) :- pack(L,R), runLength(R,M).

runLength([],[]).
runLength([[A|R]|T],[[A,N]|U]) :-
  length([A|R],N), runLength(T,U).
```

5. Write a Prolog program to split a list into two lists of positive and negative numbers. For example:

```
?- split([20,-10,30,22,45,0,-15,0,12], L1, L2).
L1 = [20,30,22,45,12]
L2 = [-10,-15]

split([],[],[]).
split([A|L],L1,[A|L2]) :- A < 0, split(L,L1,L2).
split([A|L],[A|L1],L2) :- A > 0, split(L,L1,L2).
split([A|L],L1,L2) :- A = 0, split(L,L1,L2).
```