

TDT4136 Logic And Reasoning Systems

Exercise 4

Stian Hvatum (hvatum)
MTDT

24. januar 2012

Innhold

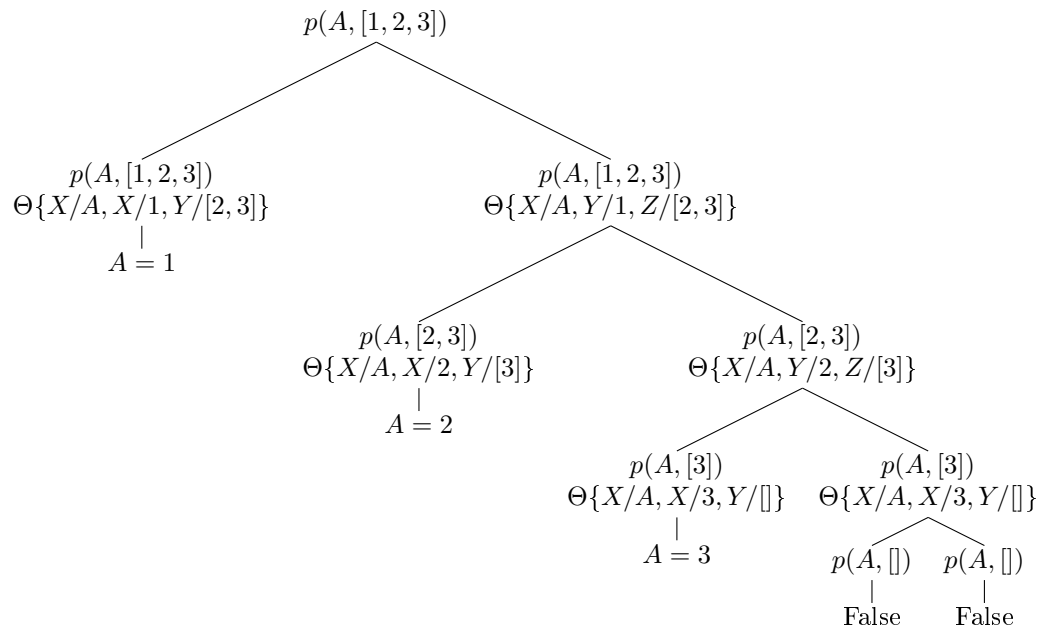
1	Task 1	1
1.1	1
1.2	1
2	Task 2	2
2.1	Kode	2
2.2	Teori	2
2.2.1	Deloppgave 2d)	2

1 Task 1

The following Prolog code defines a predicate p .
 $p(X, [X|Y]).$
 $p(X, [Y|Z]) : \neg p(X, Z).$

1.1

Show the proof trees of the queries
 $? - p(A, [1, 2, 3]).$
 $? - p(2, [1, A, 4]).$



1.2

p representerer $member(element, list)$, funksjonen sjekker om A er element i lista.

2 Task 2

2.1 Kode

Jeg har brukt *insert* fra forrige øving.

```
%a)
sorted([A,B|C]):- (A<B), sorted([B|C]).
sorted([A,B]):- (A<B).

%b)
perm(L,L).
perm(L,M) :- L=[H|L1], perm(L1,L2), insert(H,L2,M).

% fra tidligere øving
insert(X,List1,List2):- List2=[X|List1] |
                        [H|T]=List1,
                        List2=[H|A],
                        insert(X,T,A).

%c)
slow_sort(L,M) :- perm(L,M),
                  sorted(M).

%e)
insert_sort([],[]).
insert_sort(L,M) :- smallest(L,S), delete(S,L,L2),
                  insert_sort(L2,M1), M=[S|M1].

smallest([S],S).
smallest([A,B],S) :- A<B, S=A | B<A, S=B.
smallest([A,B|T],S) :- smallest([A,B],S1),
                      smallest([S1|T],S).

delete(X,L1,L2) :- insert(X,L2,L1).
```

2.2 Teori

2.2.1 Deloppgave 2d)

Slow_sort finner alle permutasjoner av lista, og sjekker om de er sorterte. Å sjekke om en vilkårlig liste er sortert, tar $O(N)$ tid. Å finne permutasjoner alle $N!$ permutasjonene tar $O(N!)$. Vi må for hver permutasjon sjekke om denne er sortert, til sammen gir dette en kompleksitet på $O(N \cdot N!)$.