

Administrative Issues

- HW3, MP3, Essay
- Essay: ABET requirement
you must get a passing score, otherwise you will receive Incomplete (I) for this course
- Guest lectures (Bonus credit)
Nov. 29th and Dec. 4th
- Final Review: Dec 6th
- Final exam: open notes and open book
Thursday, Dec. 13th, 2-4:30pm, NH 110

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2007 1

CSE 3302


Programming Languages

Logic Programming: Prolog (III)

Chengkai Li
Fall 2007


Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2007 2

Where to cut exactly?



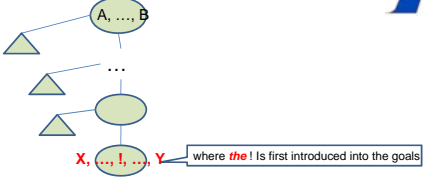
Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2007 3

Where to cut exactly?



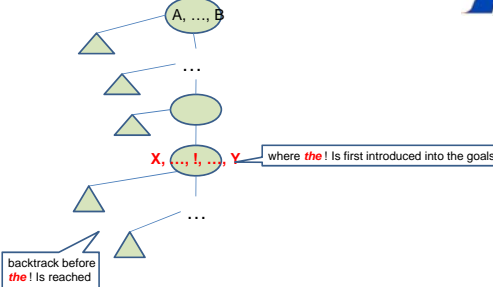
Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2007 4

Where to cut exactly?



Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2007 5

Where to cut exactly?



Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington
©Chengkai Li, 2007 6

Where to cut exactly?

The diagram shows a goal tree starting with A, \dots, B . A node $X, \dots, !, \dots, Y$ is highlighted in red, with a callout: "where *the !* is first introduced into the goals". Below it, a node $!, \dots, Y$ is highlighted in red, with a callout: "The first time when all the subgoals before *the !* are solved". A callout "backtrack before *the !* is reached" points to the left side of the tree. A blue star logo is in the top right.

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington ©Chengkal Li, 2007 7

Where to cut exactly?

The diagram is identical to slide 7, but with an additional callout "Solving the goals after *the !*" pointing to the right side of the tree. A blue star logo is in the top right.

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington ©Chengkal Li, 2007 8

Where to cut exactly?

The diagram is identical to slide 7, but with red 'X' marks on the branches to the right of the cut point. A callout "Backtracking doesn't come here" points to these branches. A blue star logo is in the top right.

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington ©Chengkal Li, 2007 9

Where to cut exactly?

The diagram is identical to slide 9, but with a callout "Backtracking does go to these subtrees" pointing to the subtrees to the left of the cut point. A blue star logo is in the top right.

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington ©Chengkal Li, 2007 10

Where to cut exactly?

The diagram is identical to slide 9, but with dashed arrows showing backtracking paths from the cut point back to the root. A callout "Other ! in other subtrees are handled using the same strategy." points to other parts of the tree. A blue star logo is in the top right.

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington ©Chengkal Li, 2007 11

Where to cut?

- *The cut (!) goal always succeeds.*
- *All the choices made before ! are frozen.*
- *Along the path from the node where ! is introduced into the goals till the node where ! is reached (all previous goals satisfied), all the siblings of these nodes are pruned.*

Lecture 22 – Prolog (III), Fall 2007 CSE3302 Programming Languages, UT-Arlington ©Chengkal Li, 2007 12

Example of !



If A then B else C:

D :- A, !, B.
D :- C.

What if ?

D :- A, B.
D :- C.

Exercise 1



- Duplicate the elements in a list, using the numbers of duplicates specified in another list.

- E.g.,
?- duplicate([1,5,3], [2,1,4], Result).

Result = [1,1,5,3,3,3,3].

Exercise 1



```
duplicate_single(H, 0, []).
duplicate_single(H, N, [H|U]) :- N>0,
                                M is N-1,
                                duplicate_single(H,M,U).

duplicate_list([], _, []).
duplicate_list([H|T], [N|Ns], Result) :- duplicate_single(H,N,U),
                                         duplicate_list(T,Ns,V),
                                         append(U,V,Result).
```

Exercise 2



- An example with predicates containing predicates
- Binary Tree (MP2)
e.g.,
tree(1,tree(2,tree(4,tree(8,void,void),void),tree(5,void,tree(9,void,void))),tree(3,tree(6,void,tree(10,void,void)),tree(7,tree(11,void,void),void))).



Exercise 2



- ordered(T): true if T is ordered
(For each node n, all nodes in n's left subtree are smaller than n, and all nodes in n's right subtree are larger than n. Assuming no duplicates).

```
ordered(T) :- ordered(T, Min, Max).
ordered(tree(X, void, void), X, X).
ordered(tree(X, L, void), Min, X) :- ordered(L, Min, Max), X>Max.
ordered(tree(X, void, R), X, Max) :- ordered(R, Min, Max), X<Min.
ordered(tree(X, L, R), Min, Max) :- ordered(L, Min, Max1), ordered(R, Min2, Max), X>Max1, X<Min2.
```

Exercise 3



```
student(Amy).
student(Bob).
take(Amy, CSE3302).
take(Amy, CSE3303).
take(Amy, CSE3304).
take(Bob, CSE3301).
take(Bob, CSE3303).
credit(CSE3301, 3).
credit(CSE3302, 3).
credit(CSE3303, 4).
credit(CSE3304, 2).

?- student(X), take(X, Y), credit(Y,Z).
?- !, student(X), take(X, Y), credit(Y,Z).
?- student(X), take(X, Y), !, credit(Y,Z).
?- student(X), take(X, Y), credit(Y,Z), !.
```