XML and Databases
COMP 4317/9317
Session 2, 2007
Final Exam, 14th Nov. 2007

(1) For each of the following, explain why it is not well-formed XML.

a) <author></author><title></title>
b) <author><title></author></title>
c) <info temp='25C'>content</info>
d) <!DOCTYPE greeting [
    <!ELEMENT greeting (#PCDATA)>
    <!ENTITY e1 "&e2; e3">
    <!ENTITY e2 "&e3;">
    <!ENTITY e3 "&e2;">
]>
<greeting> &e1; </greeting>
e) <a at1="blah" at<2="foo"> 1 < 5
f) <a b3="a" b2="b" b1="a" b2="5"/>
g) <a><c><c/></c><c/>ab&e;

(2) Show sequences of Unicode characters for which
a) UTF-8 needs more space than UTF-16
b) UTF-16 needs more space than UTF-8
together with the corresponding UTF codes and their lengths.
c) Explain how to binary sort a sequence of UTF-8 characters. Use pseudo code if appropriate.

(3) Show an element node with mixed content, using the XML Information Set. Assume that for a node M, Type(M) is it's type, i.e., is one of DOC, ELEM, ATTR, or CHAR.
Using the Infoset, show pseudo code that, given a node N,
a) returns all ancestors of the node
b) returns the previous sibling of the node.

(4) Using DOM, give pseudo code that determines the average depth of the XML tree. The average depth of <a/> is 1.

(5) Explain in detail, using an example, why hashing is useful for finding the minimal DAG of a tree. Why are updates more expensive on a DAG than on a tree? Give an example that clearly explains this.

(6) Give the PRE/POST table for the tree

<a><c/><c><d/><d></d></c><d/><c><d/></c><d/>

b) Give pseudo code that computes the POST order of a tree in an iterative way, i.e., without any recursive calls(!). You can use firstChild(n), nextSibling(n), and parent(n) for a node n.

Using the PRE/POST-encoding, explain how to obtain
c) the ancestors of a node
d) the last child of a node
e) the maximal depth of the subtree at a node.

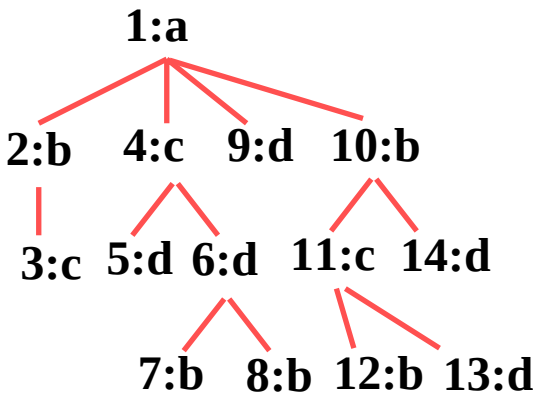
(8) Show the Glushkov automaton for the regular expression
E=(a | b)*a.
Is this expression 1-unambiguous? Explain!
Give a deterministic automaton for the same expression.

Is E2=(b*a(a|b))*a equivalent to E? Is it 1-unambiguous?
Can you find a 1-unambiguous expression that is equivalent to E?

(9) For the tree given in 6, write XPath expressions that

a) select all b nodes
b) select all b nodes that have a c-child
c) select all b nodes that have no c-children
d) select the right most c-node
e) select all nodes that have a c-parent

(10) This a tree corresponding to the XML in (6). Show the sequences of node numbers that are selected by the following queries.



a) //c//d
b) //*[a or b]
c) //b/ancestor::d/following::d
d) //*[not(.//b | ./ancestor::c)]
e) //c//d/preceding::*//d

For query c) show in detail how the Core-XPath evaluation algorithm computes the answer to this query. Do the same for query d).

(11) Show an example of XPath queries q1,q2 such that they are not equivalent, but q1 is included in q2. Show that q1 is included in q2 using one of the methods discussed. Use the homomorphism technique to test whether p=a[.//b[c/*//d]/b[c//d]/b[c/d]] is included in q=a[.//b[c//*//d]/*[c/d]]

(12) Construct a DTD such that 10a) is included in 10e), and another DTD such that 10e) is included in 10a).
[Very easy!!]

(13) Show automata for the queries
a) //n/a/n/o
b) //a/b/a/a//a/a/a
c) //a/b/*/a//*/*/a

(14) Given a PRE/POST/SIZE table, show SQL queries for the following XPath queries
a) /*
b) /a/b/*
c) //a/*//b
d) //a/following-sibling::b