Practical Lab 2

Web Data Models

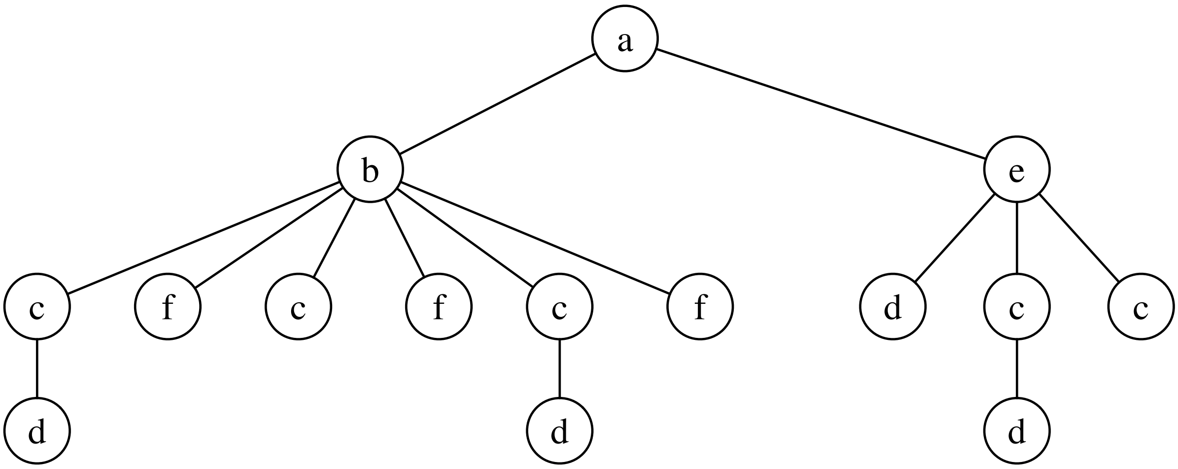
Silviu Maniu

October 3rd, 2016

The goal of this lab sessions is to exercise the validation of XML in regards to XML and DTD, and to evaluate the differences between the two schema languages.

1 DTD Validation

Consider the following XML document t, given in tree form:



1. Provide a DTD D such that the document t is valid with regard to D. Validate it using the command `xmllint`.
2. Write D as a regular grammar G.
3. Show that t is in $L(G)$. Hint: explain how a unranked tree automaton works on t and G.
4. Generate another document that is in $L(G)$.

2 DTD and RTG

Consider the following DTD D:

```
<!ELEMENT files (file*, person*) >
<!ELEMENT file (consultation)* >
<!ELEMENT consultation (symptom+, prescription?) >
<!ELEMENT symptom (#PCDATA) >
<!ELEMENT prescription (medication)* >
<!ELEMENT medication (#PCDATA) >
<!ELEMENT person (fname, lname, telnum?) >
<!ELEMENT fname (#PCDATA) >
<!ELEMENT lname (#PCDATA) >
<!ELEMENT telnum (#PCDATA) >
```

Moreover, consider the following regular tree grammar G:

$$\begin{aligned}
files &\rightarrow files[file*, Patient*, Doctor*]\\
file &\rightarrow file[Consultation*]\\
Patient &\rightarrow person[FName, LName]\\
Doctor &\rightarrow person[FName, LName, Tel]\\
Consultation &\rightarrow consultation[Symptom+, Prescription?]\\
FName &\rightarrow fname[Pcdata]\\
LName &\rightarrow lname[Pcdata]\\
Tel &\rightarrow telnum[Pcdata]\\
Symptom &\rightarrow symptom[Pcdata]
\end{aligned}$$

1. Does every document that is valid w.r.t. the DTD D is generated by the RTG G? Justify why or give a counter-example if applicable.
2. Is every document generated by G valid w.r.t. the DTD D? Justify why or give an counter-example if applicable.

3 XML Schema

Consider the XML schema specified by:

```
<?xml version="1.0" encoding="UTF-8" ?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="shiporder">
   <xs:complexType>
       <xs:sequence>
        <xs:element name="orderperson" type="xs:string"/>
        <xs:element name="shipto">
          <xs:complexType> <xs:sequence>
            <xs:element name="name" type="xs:string"/>
            <xs:element name="address" type="xs:string"/>
```

```
      <xs:element name="city" type="xs:string"/>
      <xs:element name="country" type="xs:string"/>
     </xs:sequence> </xs:complexType>
    </xs:element>
    <xs:element name="item" maxOccurs="unbounded">
     <xs:complexType> <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="note" type="xs:string" minOccurs="0"/>
      <xs:element name="quantity" type="xs:positiveInteger"/>
      <xs:element name="price" type="xs:decimal"/>
     </xs:sequence></xs:complexType>
    </xs:element>
   </xs:sequence>
   <xs:attribute name="orderid" type="xs:string" use="required"/>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

Give a corresponding regular tree grammar, a document valid w.r.t. the above schema, and validate using `xmllint`.

4 Properties of Regular Tree Grammars

Consider the following regular tree grammars:

G_1	G_2
$Dir \rightarrow directory[Pers+]$	
$Pers \rightarrow student[StudContact]$	$Dir \rightarrow directory[Pers+]$
$Pers \rightarrow teacher[TeachContact]$	$Pers \rightarrow student[PerContact]$
$StudContact \rightarrow contact[NameNumAdr?]$	$Pers \rightarrow teacher[PerContact]$
$TeachContact \rightarrow contact[NameAdrTel?]$	$PerContact \rightarrow contact[NameAdr?]$
$Name \rightarrow name[PCdata]$	$Name \rightarrow name[PCdata]$
$Num \rightarrow num[PCdata]$	$Adr \rightarrow address[PCdata]$
$Adr \rightarrow address[PCdata]$	
$Tel \rightarrow telnum[PCdata]$	

1. Is every document generated by G_2 also generated by G_1, i.e., $L(G_2) \subseteq L(G_1)$?
2. Is there a DTD equivalent to G_1 and/or G_2? If possible, provide the DTD(s).
3. Give an XML document generated by both regular tree grammars.

5 (Optional) Properties of Regular Tree Grammars

Consider two regular tree grammars $G_1 = (N_1, T, S_1, P_1)$ and $G_2 = (N_2, T, S_2, P_2)$, having the same terminal symbols T. Can you build the regular tree grammar $G_3 = (N_3, T, S_3, P_3)$ that captures the intersection of $L(G_1)$ and $L(G_2)$.