M2 Data & Knowledge — Exam Assignment 2015–2016
Complex Data and Knowledge – XML and XPath — Duration 3h00
Nicole Bidoit-Tollu, Silviu Maniu, Brigitte Safar

Reminder: slide copies, notes taken during practical labs, books, articles are **not authorized**; the only **authorized** material is **personal** notes with the following restrictions : notes should be on 1 A4 page, one side per part (i.e., one for XML, one for XPath), should not have solutions to exercises done in the practical labs, and can either be handwritten or not. For the purpose of anonymization, as the personal notes pages will be kept with your exam sheet, you should **not** put your name on it.

Strong recommendation: Do not copy the statements of the exercises on your exam sheet and solves the questions in order. Write solutions carefully and clearly and justify your answers.

1 XML data and Schema

Exercise 1

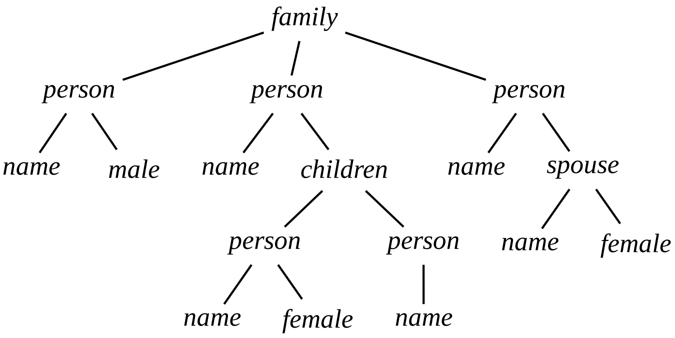

Figure 1: XML document $family$

Consider the document given in Figure 1 and the regular tree grammar (RTG) $\mathcal{G}$ specified below. As usual, a terminal symbol starts with a lower case letter and non terminal symbol with an upper case letter.

Family → family [(MPerson | FPerson)*]
MPerson → person [Name, Mgender?, FSpouse?, Children?]
FPerson → person [Name, Fgender?, MSpouse?, Children?]
Mgender → male [] Fgender → female []
FSpouse → spouse [Name, Fgender] MSpouse → spouse [Name, Mgender]
Children→ children [Person+] Name → name[]

1. Is $\mathcal{G}$ a local RTG ? justify

2. Is $\mathcal{G}$ a single/unique type RTG ? justify

3. Is $\mathcal{G}$ a restrained competition RTG ? justify

4. Give an interpretation of the document t which shows that t conforms to $\mathcal{G}$. Justify

5. Is it possible to exhibit a second interpretation of t with respect to $\mathcal{G}$? if it is the case, give a second interpretation, if not justify.

Algorithm 1: Validation For XML Document againt RTG

```
   Input: XML document $t$, RTG $\mathcal{G}$
   Let $S$ be a stack of lists of sets of non-terminals;
   Let $Y$ be a stack of sets of production rules;
   Let $P$ be a stack of non-terminal sets;
1  Push into $P$ the set of start symbols of $\mathcal{G}$;
2  Traverse $t$ in the depth-first manner
3  for element e visited do
4      Let Rules_e be the set of rules $X \to a\ r$ such that
5          $a$ is the tag of e and $X$ belongs to Top($P$)
6      if Rules_e is empty then
7          return $t$ does not comply to $\mathcal{G}$ ; Exit
8      else
9          Push Rules_e to $Y$ ;
10         Push [ ] to $S$;
11         Push into $P$ the set of non-terminals occurring in the content models
12             of the rules in Rules_e ;
13 for element e exited do
14     % complete this part
15 return Document t is valid ;
```

Exercise 2

The aim of algorithm 1 is to chek whether a document t is valid with respect to a RTG $\mathcal{G}$, in the general case.

1. Complete the algorithm 1 by giving the instructions of block 14
   Please do not copy the beginning of the algorithm on your exam sheet.
2. For the document t of Figure 2 and the RTG of Exercice 1, provide the execution traces of your algorithm.

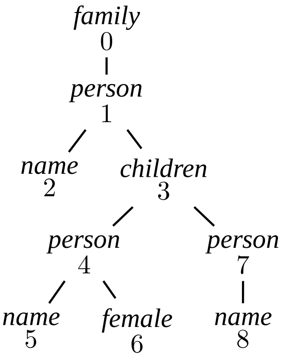

Figure 2: XML document t

Correction de la simulation :

Initial step : Put {Family} in P.
Visiting node 0:

		{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Visiting node 1:

Stack Y	Stack S	Stack P
		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[ ]	{MP, FP }
{ r1}	[ ]	{Family}

Visiting node 2:

Stack Y	Stack S	Stack P
		{ }
{r9 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[ ]	{MP, FP }
{ r1}	[ ]	{Family}

Existing node 2:

Stack Y	Stack S	Stack P
		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}

Visiting node 3:

Stack Y	Stack S	Stack P
		{MP, FP }
{r8 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}

Visiting node 4:

Stack Y	Stack S	Stack P
		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[ ]	{MP, FP }
{r8 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}

Visiting node 5:

Stack Y	Stack S	Stack P
		{}
{r9 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[ ]	{MP, FP }
{r8 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}

Exiting node 5:

Stack Y	Stack S	Stack P
		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{r8 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}

Visiting node 6:

Stack Y	Stack S	Stack P
		{ }
{r5 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{r8 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}

Exiting node 6:

		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name } , { FG }]	{MP, FP }
{r8 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Exiting node 4:

		{MP, FP }
{r8 }	[{ FP }]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Visiting node 7:

		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[ ]	{MP, FP }
{r8 }	[{ FP }]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Visiting node 8:

		{}
{r9 }	[ ]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[ ]	{MP, FP }
{r8 }	[{ FP }]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Exiting node 8:

		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{r8 }	[{ FP }]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Exiting node 7:

		{MP, FP }
{r8 }	[{ FP } , { FP, MP }]	{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Exiting node 3:

		{ Name, MG, FG, FSp, MSp, Ch}
{r2, r3 }	[{ Name } , { Ch }]	{MP, FP }
{ r1}	[ ]	{Family}
Stack Y	Stack S	Stack P

Exiting node 1:

		{MP, FP }
{ r1}	[{FP, MP }]	{Family}
Stack Y	Stack S	Stack P

Exiting node 0:

		{Family}
Stack Y	Stack S	Stack P

2 XPath

Exercise 3

Consider the XML document below:

```
<a>
  <b>
    <a >
      <c />
      <a>
        <b />
        <b />
      </a>
      <a />
    </a >
    <d>
      <a />
      <b />
    </d>
  </b>
  <b />
</a>
```

In the above document, the *context* node is the node labelled a, uniquely identified by the XPath query /a/b/a.

For each of the below XPath axes, give the set of corresponding XML nodes, as node IDs, where each ID is the *pre-order* ID, considering that the root of the document is the node having ID 0:

1. preceding
2. following
3. descendant-or-self
4. preceding-sibling
5. child

For each axis, give the list of nodes as their *pre-order node ID*, considering that the root of the document has ID 0.

Exercise 4

Consider again the XML document in Problem A. For each of the XPath queries below, give the list of the XML document nodes satisfying the query, as a sequence of node IDs, where each ID is the *pre-order* ID, considering that the root of the document is the node having ID 0:

1. //a/a
2. //a[//a]
3. //*[not(b)]
4. //a[parent::a and preceding-sibling::a]
5. //b[parent::a[parent::a] and child::a]

For each query, give a list of XML nodes satisfying the query, using their *pre-order node ID*, considering that the root of the document has ID 0.

Exercise 5

For each pair P, Q of XPath queries below, indicate whether $P \subseteq_0 Q$, i.e., P is 0-contained in Q. Give a justification why.

1. P=/a[//b]/c/d and Q=/a[/b]//d
2. P=/a[/b]/c[/a]/*/d and Q=/a[/b]/c//d