XML proposal for automata description

The VAUCANSON group

June 27, 2005
The Vaucanson group XML proposal for automata description
Objectives:

- provide an unified communication tool,
- represent various kinds of automata and transducers,
- keep declarations simple for widely used structure,
- provide ability to specify complex types.

This proposal is an evolution of the one we made at CIAA’04
Objectives:

- provide an unified communication tool,
- represent various kinds of automata and transducers,
- keep declarations simple for widely used structure,
- provide ability to specify complex types.

This proposal is an evolution of the one we made at CIAA’04
About the format definition:

- need to provide default types,
- default types must be context-sensitive,
- some attributes must be context-sensitive.

→ Can be achievable with an XSD.
Overview of the proposal

About the format definition:

- need to provide default types,
- default types must be context-sensitive,
- some attributes must be context-sensitive.

→ Can be achievable with an XSD.
The content tag

Example

```xml
<automaton>
  <content>
    <states>
      <state name="s0" label="p"/>
      <state name="s1" label="q"/>
    </states>
    <transitions>
      <transition src="s0" dst="s0" label="a"/>
      <transition src="s0" dst="s0" label="b"/>
      <transition src="s0" dst="s1" label="b"/>
      <transition src="s0" dst="s1" label="a"/>
      <transition src="s0" dst="s1" label="b"/>
      <initial state="s0"/>
      <final state="s1"/>
    </transitions>
  </content>
</automaton>
```
The content tag

- label attribute is a non-restricted string,
- `<transition>`, `<initial>` and `<final>` are context sensitive.

Example

Transducer declaration:

```xml
<transducer>
  <content>
    <states>
      <state name="s0"/>
    </states>
    <transitions>
      <transition src="s0" dst="s0" in="a" out="b"/>
      <initial state="s0" out="a"/>
    </transitions>
  </content>
</transducer>
```
The type tag

Example

Set automaton type to $\mathbb{Z}$:

```xml
<automaton>
  <type>
    <semiring set="Z"/>
  </type>
  <content>
    ...
  </content>
</automaton>
```

- Need to override only the inappropriate tag,
- can define weight and alphabet structure.
The geometry tag

Example

Set a global offset, then a state position:

```xml
<transducer>
  <geometry x="-5" y="0"/>
  <content>
    <states>
      <state name="s0">
        <geometry x="10" y="10"/>
      </state>
    </states>
  </content>
  ...
</transducer>
```

- Contains embedding informations of the graph,
- can be defined at any level of the document,
- is context-sensitive.
The drawing tag

Example

Set some drawing properties:

```xml
<transducer>
    <geometry x="-5" y="0"/>
    <drawing stateFillColor="black" edgeStyle="dashed"/>
    <content>
        <states>
            <state name="s0">
                <drawing stateFillColor="red"/>
            </state>
            ...
        </states>
    </content>
</transducer>
```

- Contains drawing informations,
- can be defined at any level of the document,
- can be extented thanks to anyAttribute.
The complexity of the type tag

Some features:

- can be recursively defined,
- can express a wide set of structures (various alphabets, various weight set and related operations, etc.),
- can be declared only where needed.

→ The type tag offers a full description of the automaton type.
The complexity of the type tag

Some features:

- can be recursively defined,
- can express a wide set of structures (various alphabets, various weight set and related operations, etc.),
- can be declared only where needed.

→ The type tag offers a full description of the automaton type.
The complexity of the type tag

Some features:

- can be recursively defined,
- can express a wide set of structures (various alphabets, various weight set and related operations, etc.),
- can be declared only where needed.

→ The type tag offers a full description of the automaton type.
The complexity of the type tag

Some features:

- can be recursively defined,
- can express a wide set of structures (various alphabets, various weight set and related operations, etc.),
- can be declared only where needed.

→ The type tag offers a full description of the automaton type.
Default types are:

- for `<automaton>`, Boolean automaton,
- for `<transducer>`, automaton with a direct product of free monoids.
The session tag

Example

Combine many automata in a single document:

```xml
<session>
  <automaton name="a1">...</automaton>
  <transducer name="t1">...</transducer>
  <transducer name="t2">...</transducer>
</session>
```
The proposal offers:

- full description of the content of the automaton,
- full description of the type of the automaton,
- geometry and drawing facilities,
- a set of predefined types to ease declaration.
The proposal offers:

- full description of the content of the automaton,
- full description of the type of the automaton,
- geometry and drawing facilities,
- a set of predefined types to ease declaration.
The proposal offers:

- full description of the content of the automaton,
- full description of the type of the automaton,
- geometry and drawing facilities,
- a set of predefined types to ease declaration.
Conclusion

The proposal offers:

- full description of the content of the automaton,
- full description of the type of the automaton,
- geometry and drawing facilities,
- a set of predefined types to ease declaration.
Questions