Subject: Alloy vs XML Schema – infinite loops

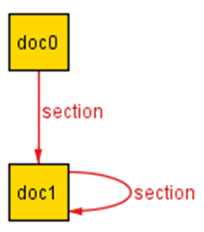
Here is an Alloy model of documents that have a section which contains a document.

**sig** doc {  
 section: doc  
}

That appears to be creating an infinite loop:

A doc has a section field which must have a doc  
 which has a section field which must have a doc which …

and should be illegal. It seems like such a signature would trigger the Alloy Analyzer to go into an infinite loop. But that’s not so. The signature simply says that there are a set of doc atoms and the section relation maps one doc atom to another doc atom. To construct a section relation, it’s as simple as looking at the doc set and choosing, “I’ll pair that doc with that doc.” The two docs that are paired could be the same doc. Here is an instance of the model:



The section relation pairs doc0 with doc1 and doc1 with itself. In the graphic, the self-loop on doc1 appears to be an infinite loop, but it’s not. It is simply doc1 paired to itself, nothing more than that.

|  |  |
| --- | --- |
| **section** | |
| doc0 | doc1 |
| doc1 | doc1 |

Compare with XML Schemas. Here is an XML Schema definition that looks strikingly similar to the Alloy signature:

<xs:complexType name="doc">  
 <xs:sequence>  
 <xs:element name="section" type="doc" />  
 </xs:sequence>  
</xs:complexType>

It says that each doc type has a section element which must contain a doc type. In fact, this *does* create an infinite loop:

A doc type has a section element which must have a doc  
 type which has a section element which must have a doc type which …

and there are no valid instances.