# 一种快速的 XML 文档验证算法

张 苗[1],惠小强[2]
(1. 西安邮电大学 通信工程学院,陕西 西安 710061;
2. 西安邮电大学 物联网与两化融合研究院,陕西 西安 710061)

**摘 要**:在用 XML(eXtensible Markup Language)Schema 校验来判断 XML 文档合法性的过程中,目前 LIBXML2 所采用的逐层遍历校验法对 3 层及以下嵌套效率低下,对多于 3 层嵌套的校验法失效。针对这一问题,文中提出一种新算法,可有效避免逐层遍历法验证的缺陷。先计算 XML 文档中某待验证元素出现的次数,然后把该次数分解为 XSD(XML Schema Definition)文档中所定义的该元素允许出现次数区间内整数的线性组合,计算出所有线性组合中系数和的最小值和最大值。若计算出的系数和范围与 XSD 文档定义的范围有交集,则验证通过,否则不通过。利用所提算法,对多嵌套和 maxOccurs 较大的 XML 文档,验证效率可提高两个数量级。

**关键词**:XML 文档;复杂类型;LIBXML2;XML Schema 校验
**中图分类号**:TP311 **文献标识码**:A **文章编号**:1673-629X(2015)08-0123-05
doi:10.3969/j.issn.1673-629X.2015.08.026

# A Fast Algorithm of XML Document Verification

ZHANG Miao[1], XI Xiao-qiang[2]
(1. College of Communication Engineering, Xi'an University of Posts and Telecommunications, Xi'an 710061, China;
2. Institute of Internet of Things and IT-based Industrialization, Xi'an University of Posts and Telecommunications, Xi'an 710061, China)

**Abstract**: When using XML (eXtensible Markup Language) Schema to verify the validation of XML document, the present traversal loop method used by LIBXML2 has low efficiency for no more than 3-loop, and will be invalid for more than 3-loop nested document. To solve the problem, propose a new verification algorithm and it can overcome the defect of traversal loop method. First, the times of element that will be validated in the XML document is calculated, then this times is divided into the linear combination of the permitted times section that defined in XSD document, next is to calculate the maximum and minimum of summation of the coefficient in all the linear combination. If the range of the coefficient summation can overlap with the range that defined in XSD document, then the validation passes, otherwise fails. Using this method, the validation efficiency can increase two orders for the XML document with multi-loop nested and bigger maxOccurs.

**Key words**: XML documents; complex type; LIBXML2; XML Schema validation

## 0 引 言

近几年,网络的迅猛发展伴随着海量信息的产生,这些信息一般被划分为两大类,一类是能够用数据库二维逻辑表来表现的结构化数据,而另一类则是无法用数据库二维逻辑表来表现的非结构化数据,如文本、图像、声音等。占绝大部分的非结构化数据的管理一直困扰着设计人员。XML 的出现解决了这一难题,同时,XML 也因其开放性、平台无关性和良好的数据变化能力逐渐得到了业界认可。W3C(World Wide Web Consortium,万维网联盟)于 1998 年 2 月正式发布 XML1.0 版[1]。但 XML 文档只能表示数据的内容,不能表示实例的文档结构、每个元素以及属性的数据类型,因此仅仅只用 XML 来完成文档数据、内容以及结构的表示是不够的。在这种情况下,W3C 于 2001 年

**收稿日期**:2014-07-23 **修回日期**:2014-10-27 **网络出版时间**:2015-07-21
**基金项目**:国家自然科学基金资助项目(11275099)
**作者简介**:张 苗(1990-),女,硕士研究生,研究方向为现代信号处理及应用;惠小强,博士,教授,从事量子信息、图像处理大数据挖掘分析等理论研究。
**网络出版地址**:http://www.cnki.net/kcms/detail/61.1450.TP.20150721.1433.014.html

正式推荐使用 XML Schema[2]。XML Schema 是对 XML 文件的构造进行定义的规范语言之一,也称作 XML Schema 定义(XML Schema Definition,XSD),它是 XML 的一个应用,遵循 XML 语法[3]。

进行合法性校验时需要 XSD 文件以及对应的 XML 文件,并将它们的路径输入到指定入口。先把文件解析成树结构[4](关于 XML 解析技术可参考文献[5]),然后进行合法性校验,看给定的 XML 文件是否符合 XSD 的定义[6]。该过程的实质是对相应的两棵树进行同步遍历[7-8]。一般情况下采用微软的. NET 或者 LIBXML2 来进行 Schema 校验。. NET 平台就是微软用来实现 XML,Web Services,SOA(Service-Oriented Architecture,面向服务的体系结构)等技术的平台。微软在这方面做了一定的优化,使得校验效率得到了较大提升,但其代码不是开源的。LIBXML2 是一个 XML C 语言版的解析器,可以满足一般用户的需求。LIBXML2 采用的是状态机,要求遍历所有状态,才知道校验是否正确。但是在 choice 和 sequence、choice 和 choice 多层嵌套的情况下效率很低,且嵌套层数过多时校验失效。

文中没有通过遍历元素所有出现的可能性来判断元素的实际出现次数是否正确,而是通过数学方法计算出元素可能出现的最大最小次数区间,看该区间是否和 XSD 定义的范围区间有交集。因此,文档校验的问题转为设计算法求出元素可能出现的最大和最小次数。

## 1 相关理论描述

### 1.1 XML 文档

最基本的 XML 文档由声明和元素构成。声明放在 XML 文档的第一行,如<? xml version="1.0" encoding="UTF-8"? >。元素是 XML 文档中最基本的单元,每个文档有且仅有一个根元素,其他元素需包含在该根元素中[9]。

对基于 XML 的应用而言,文档的模式验证是有效处理数据的前提和保证。例如向某个 Web 服务器发送基于 XML 的 SOAP 协议时,防火墙便通过对 XML 文档的模式验证来判断该请求是否合法,并把不合法的请求过滤掉以减轻服务器的负载[10]。

XML 文档的常用模式为 XML Schema 和 DTD(Document Type Definition,文档类型定义)。

### 1.2 XML Schema 和 DTD 比较

一个有效文档既遵守 XML 语法规则,也遵守 DTD 或 XML Schema 中定义的规则。XML Schema 和 DTD 之间的区别有下面几点:

(1)Schema 本身也是 XML 文档,DTD 的定义跟 XML 没有什么关系,Schema 在理解和实际应用中有很多便利。

(2)DTD 文档的结构是“平铺型”的,若定义复杂的 XML 文档,各元素之间的嵌套关系就很难把握;Schema 文档结构性强,各元素之间的嵌套关系很直观。

(3)DTD 只能指定元素含有文本,不能定义元素文本的具体类型,如字符型、整型、日期型、自定义类型等。Schema 在这些方面的功能比 DTD 强大。

(4)Schema 支持元素节点顺序的描述,DTD 没有提供无序情况的描述,要定义无序必须穷举所有排列。Schema 可以利用 xs:all 来表示无序的情况。

(5)对命名空间的支持。DTD 无法利用 XML 的命名空间,Schema 很好满足命名空间的规范。并且,Schema 还提供了 include 和 import 两种引用命名空间的方法[11]。

由于 XML Schema 的优势明显,取代 DTD 已成大势所趋。国际上一些知名企业和组织已在战略上向 XML Schema 倾斜,提供相应的技术支持。其中最为典型的要数微软 BizTalk 和 xml. org 组织的注册/资源库[12]。文中采用 XML Schema。

## 2 基于 Schema 的模式验证

### 2.1 算法应用环境描述

在 XML Schema 中,有几种比较重要的指示器:

(1)Order 指示器。

Choice:规定可出现某个子元素或另外一个子元素(非此即彼)。

Sequence:规定子元素必须按照特定的顺序出现[13]。

(2)Occurrence 指示器:用于定义某个元素出现的频率。

maxOccurs:规定某元素可出现的最大次数。

minOccurs:规定某元素可出现的最小次数。

对于 all,choice,sequence,其中的 maxOccurs 以及 minOccurs 的默认值均为 1。

每个 XML 文档的元素或属性都有一个数据类型,它限定了元素的内容和属性值的验证,可以是简单类型或是复杂类型。只有在复杂类型中才可以出现 choice 和 sequence 的嵌套情况,在 XML Schema 中用 complexType 元素可定义一个复杂类型[14],下面给出一个具体例子:

```
<xs:element name="number">
<xs:complexType >
<xs:choice>
<xs:choice minOccurs="2" maxOccurs="3">
```

```
<xs:sequence minOccurs="1" maxOccurs="3">
<xs:element name="number1" type="xs:string"
minOccurs="4" maxOccurs="9" >
<xs:element name="number2" type="xs:string"
minOccurs="0" maxOccurs="3" >
</xs:sequence>
</xs:choice>
</xs:choice>
</xs:complexType >
</xs:element>
```

该结构首先定义了一个元素 number,在该元素内部,又定义了四个元素。这几个元素出现的次数和位置受 choice,sequence,minOccurs 和 maxOccurs 的限制。这是 choice 和 sequence 以及 choice 和 choice 多层嵌套的情况,sequence 要求子元素必须按顺序出现,否则报错,相对比较简单,不在文中讨论之列。下面重点讨论 choice 多层嵌套下的算法优化问题。

### 2.2　算法设计与有效性证明

先将 2.1 节中的例子转化为数学问题,定义如下:

```
<xs:element name="number1" type="xs:string" minOccurs=
"4" maxOccurs="9">
<xs:element name="number2" type="xs:string" minOccurs=
"0" maxOccurs="3">
```

上述语句的意义:$number_1$ 元素的类型为 xs:string,可以出现的最小次数为 4 次,最大次数为 9 次。即在 XML 文档中,$number_1$ 元素可以出现 4 到 9 次。$number_2$ 元素类似。

但是这两条语句的上面还有:

```
<xs:choice  minOccurs="1"  maxOccurs="3">
```

这是一层嵌套,属最简单的情况。由 choice 规定可知,它可以选择 1 到 3 次,每次只能选择一个元素,要么 $number_1$ 要么 $number_2$。由于不能确定下一个元素,所以 $number_1$ 和 $number_2$ 出现的范围也不能确定。

本算法要完成的是:在给定 $number_1$ 出现次数(从 XML 文档中获取)的基础上,求出它可能满足 $number_1$ 元素中定义的 minOccurs 和 maxOccurs 多少次,即求出类似 choice 中定义的 minOccurs 和 maxOccurs,看是否和 choice 中定义的 minOccurs、maxOccurs 范围有交集。如果有交集,说明它可以落在定义的范围内,因此是合法的。算法和证明如下:

假设 XSD 中定义的 $number_1$ 元素的 minOccurs = "$a_1$",maxOccurs = "$a_n$",则 $number_1$ 元素可以出现的次数依次为 $(a_1,a_2,\cdots a_i,\cdots,a_n)$,即 $a_i$ 是公差为 1 的等差数列,$a_1 > 0$。设 $number_1$ 各次数可以出现的次数为 $(x_1,x_2,\cdots,x_i,\cdots,x_n)$,即 $a_1$ 出现 $x_1$ 次,$a_2$ 出现 $x_2$ 次,依次类推。由 XML 文档计算出 $number_1$ 出现的次数为 $y$,假设该次数 $y$ 可分解为

$$y = \sum_{i=1}^{n} x_i a_i = x_1 a_1 + x_2 a_2 + \cdots + x_i a_i + \cdots x_n a_n \tag{1}$$

总次数之和为 $z = \sum_{i=1}^{n} x_i$,现在需要确定 $z$ 的最大值 $z_{max}$ 和最小值 $z_{min}$。显然,$z_{max}$ 决定于尽可能多的 $a_1$ 系数;$z_{min}$ 则决定于尽可能多的 $a_n$ 系数。

由式(1)可知,如果 $y/a_1$ 余 0 的话,最大次数 $z_{max} = y/a_1$;如果 $y$ 不能整除 $a_1$ 的话,$a_1$ 的系数减 1,增加一个 $a_2$,依次类推,直到 $y$ 被完全分解。

当 $y$ 不能整除 $a_1$ 时,根据等差数列的性质可知 $a_1,a_2,\cdots,a_i,\cdots,a_n$ 中的任意一个元素 $a_i$ 都可以由相邻的两个元素 $a_{i+1},a_{i+2}$ ($a_{n+i} = a_i$) 线性表示,结合式(1)知,$y$ 可由 $a_1,a_2,\cdots,a_i,\cdots,a_n$ 中任意两相邻的元素线性表示,但有可能产生负系数;而在文中环境下需保证系数为正,故需要条件判断和相应的循环。设 $y/a_i$ 的商为 $d_i$,余数为 $r_i$ (其中 $1 \leqslant i \leqslant n$),则

$$y = a_i d_i + r_i = a_i(d_i - r_i) + (a_i + 1)r_i = a_i(d_i - r_i) + a_{i+1}r_i \tag{2}$$

由于系数不能为负数,所以式(2) 必须满足 $d_i - r_i \geqslant 0$。$d_i - r_i > 0$ 表示 $y$ 由相邻两个元素表示,$d_i - r_i = 0$ 表示 $y$ 由某个元素 $a_i$ 表示。

最小值算法求解过程和 $z_{max}$ 求解类似。若 $y/a_n$ 余 0,最小次数为 $z_{min} = y/a_n$;如果 $y$ 不能整除 $a_n$ 的话,$a_n$ 的系数减 1,增加一个 $a_{n-1}$,依此类推,直到 $y$ 被完全分解。

同理,当 $y$ 不能整除 $a_n$ 时,为迅速找到 $z_{min}$,可用下面推出的条件判断。设 $y/a_i$ 的商为 $d_i'$,余数为 $r_i$,$d_i = d_i' + 1$。定义 $d_i = d_i' + 1$ 是为了得到负的余数,使得搜索方向向下,将 $y$ 用 $a_n$ 和 $a_{n-1}$ 线性表示。$y$ 可表示如下。

$$y = a_i d_i - a_i + r_i = (a_i - r_i)a_{i-1} + (d_i + r_i - a_i)a_i \tag{3}$$

由式(3)知,当 $d_i + r_i - a_i \geqslant 0$ 时,即 $a_i - r_i \leqslant d_i$ 时,可以推出 $y$ 由 $a_i$ 和 $a_{i-1}$ 的线性组合表示。

### 2.3　算法实现流程图

根据上述的分析,可写出具体的算法流程,前提是 $y \geqslant a_1$,流程图如图 1 和图 2 所示。

### 2.4　优缺点分析

LIBXML2 采用的是状态机,先把 XML 文档解析成 DOM[15-16](Document Object Model,文档对象模型)树,再用 XSD 把元素所有可能出现的情况编译成一张图,然后根据 DOM 树中的节点遍历整个图,如果 XML 文档中出现的元素正确就继续遍历,否则就回溯遍历其他情况。当 maxOccurs 很大,嵌套层数很多时,XSD 编译后的图将非常复杂,若出现元素错误,回溯将十分

耗时。

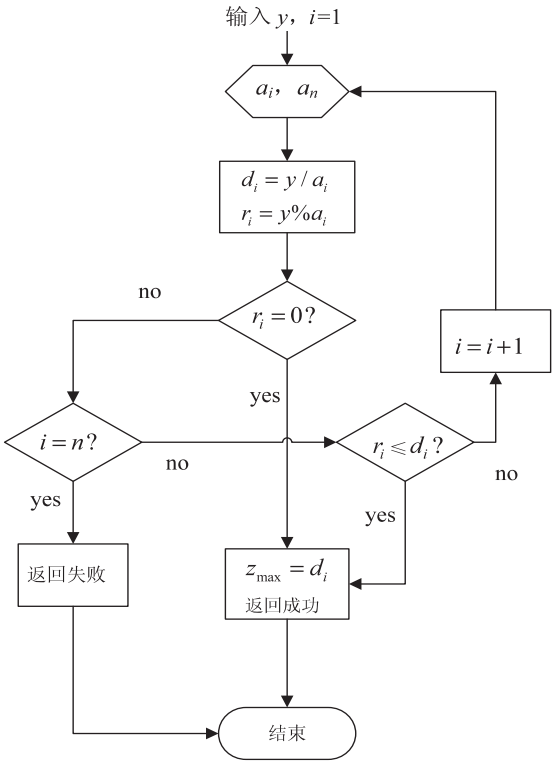


图 1 求解最大值流程图

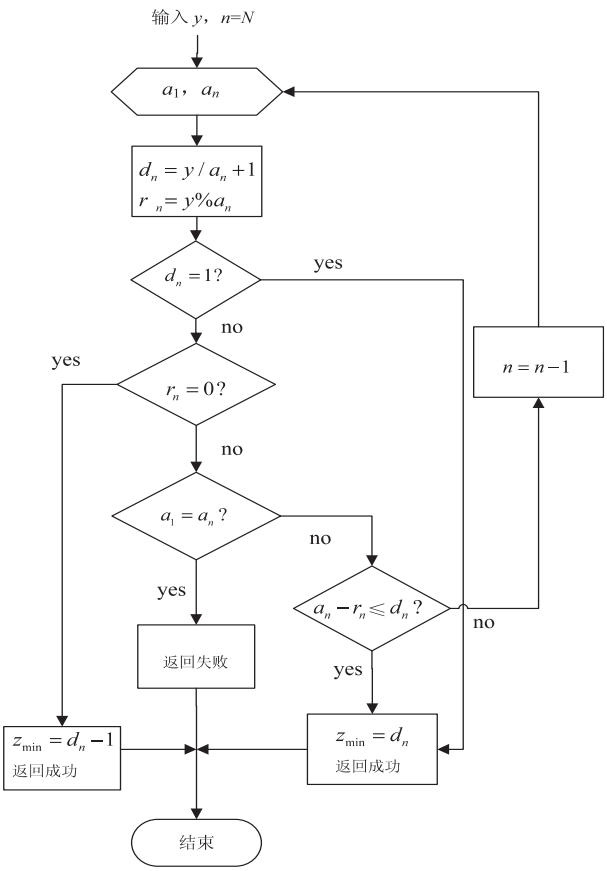


图 2 求解最小值流程图

文中算法是直接检查，避免了不必要的回溯。校验器先遍历 DOM 树找到对应的 XML 元素，然后计算出该元素出现的次数，即 2.3 节中的 $y$，由 $y$ 和 minOccurs，maxOccurs 计算出 $z_{min}$ 和 $z_{max}$，若它们和 minOccurs，maxOccurs 有交集，继续检查下一个元素；否则向上检查，直到达到最上层。因为最坏的情况才需要检查所有嵌套，所以能较大提升校验速度。本算法在 maxOccurs 大、嵌套层数多时优势明显，当 maxOccurs = 2 000 时，校验速度会是 LIBXML2 的 100 多倍。缺点是只能判定某个地方出现的元素合法或者不合法，不能告诉下一个正确的元素是什么。

## 3 实 验

本节以实例说明文中算法的优势，选取 maxOccurs = 2 000，XSD 文件中元素 number 定义如下（二层嵌套，多层的话增加相应的 choice 层数）：

```
<xs:element name="number">
<xs:complexType >
<xs:choice minOccurs="10" maxOccurs="2000">
<xs:choice minOccurs="10" maxOccurs="2000">
<xs:element name="number1" type="xs:string" minOccurs="1000" maxOccurs="2000">
<xs:element name="number2" type="xs:string" minOccurs="1000" maxOccurs="2000">
</xs:choice>
</xs:choice>
</xs:complexType >
</xs:element>
```

当 maxOccurs 比较大时，两种算法比较如表 1 所示。

表 1 maxOccurs = 2 000 时算法耗时对比 s

| 嵌套层数 | LIBXML2-2.9.0 | 文中算法 |
|---|---|---|
| 1 | 8 | 0.083 |
| 2 | 11 | 0.115 |
| 3 | 无效 | 0.161 |
| 4 | 无效 | 0.213 |
| 5 | 无效 | 0.241 |
| 6 | 无效 | 0.269 |

当 maxOccurs 比较小时，两种算法比较如表 2 所示（执行 600 个相同的测试用例）。

表 2 maxOccurs = 10 时算法耗时对比 s

| 嵌套层数 | LIBXML2-2.9.0 | 文中算法 |
|---|---|---|
| 1 | 1 | 0.91 |
| 2 | 1.9 | 0.928 |
| 3 | 3.0 | 0.941 |
| 4 | 无效 | 0.967 |
| 5 | 无效 | 0.982 |
| 6 | 无效 | 1.100 |

由表 1 和表 2 可知，对于一般较简单的 XML 文

档,两者执行速度相差不大。但是对 maxOccurs 大、嵌套层数多的情况,LIBXML2 执行很慢,当嵌套层数大于 2 时,会栈溢出导致算法无效。以上数据是在 XML 文档中元素个数确定时得到的,可以看出嵌套层数对 LIBXML2 的执行影响很大。文中算法受嵌套层数影响不大,而且支持的最多嵌套层数为 6,满足实际需求,目前已投入使用。

## 4　结束语

文中针对在用 XML Schema 校验来判断 XML 文档合法性的过程中,目前 LIBXML2 所采用的逐层遍历校验法对 3 层及以下嵌套效率低下,对多于 3 层嵌套的校验法失效的问题,提出一种新算法,可有效避免逐层遍历法验证的缺陷。实验结果表明,该算法稳定有效。

**参考文献:**


[1] 彭　涛,孙连英. XML 技术与应用[M]. 北京:清华大学出版社,2012.

[2] 董亚娟,卑小贤,刘西洋,等. 一种 XML 文档模式有效性验证算法[J]. 计算机工程与应用,2005,41(16):86-89.

[3] 李亚佳. 基于 Schema 验证的 XML 解析器中编辑子系统的设计与实现[D]. 西安:西安电子科技大学,2005.

[4] 王　仲,陈晓鸥. 基于 XML 的数据交换与存取技术研究[J]. 计算机工程与应用,2001,37(24):108-111.

[5] 冯　进,丁　博,史殿习,等. XML 解析技术研究[J]. 计算机工程与科学,2009,31(2):120-124.

[6] 俞　斌,熊齐邦. 基于 XML 的网络配置管理的研究与实现方案[J]. 计算机技术与发展,2007,17(2):168-171.

[7] 王　霜. 基于 Schema 文档的 XML 文档验证系统的设计[J]. 沈阳师范大学学报:自然科学版,2010,28(2):229-232.

[8] 李　浩,沈　琦. XML Schema 中的面向对象思想[J]. 计算机系统应用,2004,14(2):36-39.

[9] Elmasri R, Li Qing, Fu J, et al. Conceptual modeling for customized XML schemas[J]. Data & Knowledge Engineering, 2005,54:57-76.

[10] 王伟良,施　佺,曹渠江. 基于 XML Schema 抽象模型的 XML 模式验证方法[J]. 计算机应用与软件,2004,24(3):41-43.

[11] 李　华,刘修国. 对 XML 的模式 DTD 和 Schema 的探讨[J]. 计算机与现代化,2003(2):12-14.

[12] 张　伟,苑迎春,王克俭. DTD 与 Schema 简介[J]. 现代电子技术,2001(6):75-79.

[13] Walmsley P. XML 模式权威教程[M]. 北京:清华大学出版社,2003.

[14] 胡亚明. 基于 Schema 验证的 XML 解析器中验证子系统的设计与实现[D]. 西安:西安电子科技大学,2005.

[15] Ma Jianliang, Zhang Shaobin, Hu Tongsen. Parallel Speculative dom-based XML parser[C]//Proc of the 14th international conf on high performance computing and communications. Hangzhou:[s. n.],2012:33-40.

[16] Lam T C, Ding J J, Liu J C. XML document parsing: operational and performance characteristics[J]. IEEE Computer, 2008, 41(9):30-37.


(上接第 122 页)


for task assignment problem[J]. Microprocessors and Microsystems,2002,26(8):363-371.

[10] Shi Y, Eberhart R C. Empirical study of particle swarm optimization[C]//Proceedings of the 1999 congress on evolutionary computation. Piscataway, NJ: IEEE, 1999:1945-1950.

[11] 王铁君,邬月春. 基于混沌粒子群算法的物流配送路径优化[J]. 计算机工程与应用,2011,47(29):218-221.

[12] 高立群,任　苹,李　楠. 基于混沌粒子群算法的高速旅客列车优化调度[J]. 东北大学学报:自然科学版,2007,28(2):176-179.

[13] Bosco G L. PGAC: a parallel genetic algorithm for data clustering[C]//Proceedings of the seventh international workshop on computer architecture for machine perception. [s. l.]:[s. n.],2005.

[14] 刘　莉,王长林. 城市轨道交通列车运行调整的粒子群算法研究[J]. 铁路计算机应用,2013,22(6):62-64.

[15] Tsai S J, Sun T Y, Liu C C, et al. An improved multi-objective particle swarm optimizer for multi-objective problems[J]. Expert Systems with Applications, 2010, 37(8):5872-5886.

[16] Liu B, Wang L, Jin Y H. An effective PSO-based memetic algorithm for flow shop scheduling[J]. IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics, 2007, 37(1):18-27.

[17] Parsopoulos K E, Vrahatis M N. On the computation of all global minimizers through particle swarm optimization[J]. IEEE Transactions on Evolutionary Computation, 2004, 8(3): 211-224.

# 一种快速的XML文档验证算法



作者： 张苗， 惠小强， ZHANG Miao， XI Xiao-qiang
作者单位： 张苗,ZHANG Miao(西安邮电大学 通信工程学院,陕西 西安,710061)， 惠小强,XI Xiao-qiang(西安邮电大学 物联网与两化融合研究院,陕西 西安,710061)
刊名： 计算机技术与发展 ISTIC
英文刊名： Computer Technology and Development
年，卷(期)： 2015(8)