第17卷 第3期
2007年3月

计算机技术与发展
COMPUTER TECHNOLOGY AND DEVELOPMENT

Vol.17 No.3
Mar. 2007

基于语义的Web服务匹配研究

吴 芸[1],鱼 滨[2]

(1.西北大学 软件工程研究所,陕西 西安 710127;
2.西安电子科技大学,陕西 西安 710068)

摘 要:给出了一种基于OWL-S语言的服务描述以及服务的查找方法。该方法依据服务所必须满足的前置条件和后置条件作为判断服务匹配的准则,可以加快服务的匹配速度以及提高匹配精度,并在实际应用中证明了该方法的有效性。

关键词:Web服务;OWL-S;前置条件;后置条件;匹配度

中图分类号:TP301.2;TP311 **文献标识码**:A **文章编号**:1673-629X(2007)03-0016-04

Study on Matching of Web Services Based on Semantics

WU Yun[1], YU Bin[2]

(1.Software Engineering Institute, Northwest University, Xi'an 710127, China;
2.Xidian University, Xi'an 710068, China)

Abstract: A method which used the description of the services based on OWL-S language and service searching is given. The method uses the pre-condition and post condition which the services must be met as a judge matching criteria, and this can speed up the matching of services and improve the matching accuracy, and prove the effectiveness of the method in the practical application.

Key words: Web services; Web ontology language for services; pre-condition; post-condition; matching degree

0 引 言

Web服务[1]是具有自包容、自描述性的应用模块,它可以通过Web来发布、查询和调用。用户可以将在网上发布的Web服务集成到自己的应用程序来完成任务而无需再重复开发,因此,Web服务在Internet环境中提供了一种与平台、语言无关,而在机器与机器之间可以共享数据和服务的模式。

随着Internet中Web服务的数目和种类的增加,如何在诸多服务中找到符合要求的服务,即服务匹配,成为利用Web服务进行资源共享必须解决的问题。服务匹配[2]就是根据用户对服务接口的描述去寻找能够满足要求的服务。服务匹配包含了两个功能:一个是发布服务,一个是从服务的描述数据库中查询服务。服务提供者将服务的描述发布在匹配器的服务注册中心,用户则通过提交查询来获取所需服务的信息。由于网络上的服务种类数目繁多,因此可能和用户需求相关的服务有多个,如何从如此多的信息中过滤出相关信息就是服务匹配急需解决的问题。现有的行业标准UDDI(Universal Description Discovery and Integration,统一描述、发现与集成)提供了一种基于分布式的商业注册中心机制,进行服务描述文件的注册、管理和发现服务。这种方法的主要缺点是对服务的描述缺少灵活性,使得在服务匹配时只能采用简单的关键字搜索方法,显然不能满足服务发现的需要。文中提出一种基于OWL-S的服务描述和服务匹配方法,能够较好地解决服务发现中的服务匹配问题。

1 语义化Web服务

Web服务作为Web技术的最新发展成果,它的出现及推广将变革现有的Web应用模式。但是要想使分布于Internet上的服务器可以通过Web更自动化、更智能化地交互,就必须解决目前Web上广泛存在的信息格式的异构性、信息语义的多重性以及信息关系的匮乏和非统一。

将本体的概念和相应技术引入Web服务技术中,将从根本上解决以上问题。同时,由于本体具有丰富的语义和广泛的关系,它将变革现有的Web服务,使之成为语义Web服务,使Web实现从自动化到智能化

收稿日期:2006-06-20

基金项目:国家自然科学基金资助项目(60373103)

作者简介:吴 芸(1981-),女,陕西人,硕士研究生,研究方向为Web服务发现,分布式应用;鱼 滨,副教授,硕士生导师,研究方向为分布式应用,中间件技术。

的转变成为可能。也就是说,通过合理的设计,可以充分发挥它们各自的优势,既结合语义Web的语义扩展,也结合Web服务的分布特性,最终提供一种基于语义的自动Web服务协作机制。

OWL-S[3]是一个为了描述Web服务的本体语言,它是基于语义Web的OWL语言框架的。它将使用户和软件代理在指定的约束下能够自动地发现、调用、合成和监控Web资源提供的服务。图1给出了OWL-S上层本体的结构[4]。Resource代表提供Web服务的Web资源,类Service作为Web服务本体分类的根节点,代表Web服务自身,OWL-S本体结构是根据Service的三个属性:presents,describedBy和supports来建造的,这三个属性分别涉及Web服务的三个最基本的问题:此Web服务提供什么样的服务;Web服务具体是怎样工作的以及服务调用方具体怎样使用Web服务,这三个问题分别由上述三个属性的值域(range)-ServiceProfile,ServiceModel和ServiceGrounding来回答。

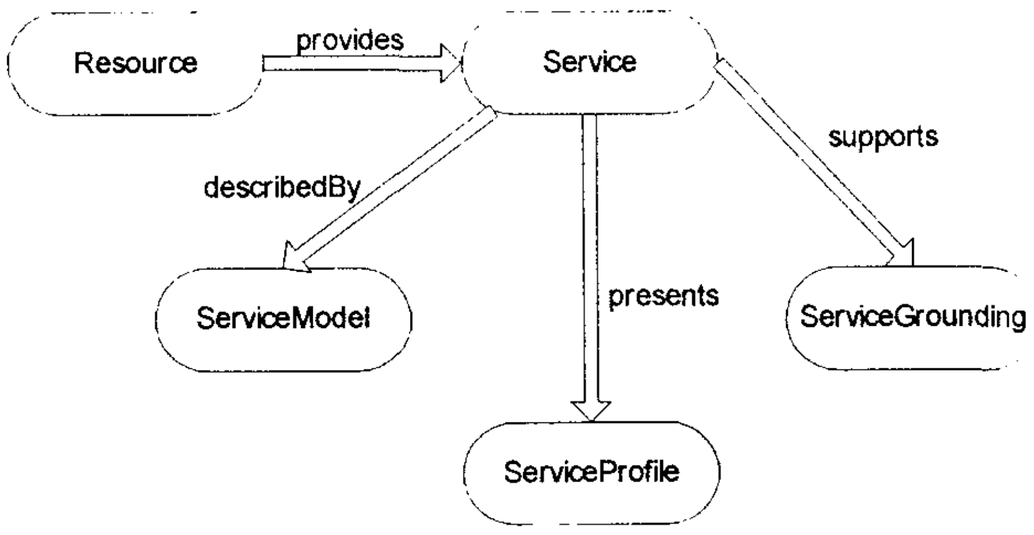

图1　OWL_S的上层结构

ServiceProfile说明了服务干什么,也就是说,它给出了服务搜索Agent决定该服务是否符合自己需要所必需的信息。ServiceProfile并不强制服务的表示法,而是运用OWL子类使得为服务创建专门的表示成为可能,同时这些表达被用作服务的轮廓。OWL-S通过类Profile提供了一种可能的表示法。

OWL-S Profile将服务描述为三类基本信息的函数:服务的提供组织、服务的功能和一组标定服务特点的属性值。提供者信息包括联系信息,它涉及到提供服务的实体。服务的功能描述通过该服务所产生的变换来表示。特别地,它指定服务所要求的输入和产生的输出;此外,既然服务有可能要求额外需满足的条件和改变这些条件带来的影响,Profile描述了服务所要求的前提和执行该服务所期望的结果。最后,Profile要求通过对一组属性的描述来描述服务的特征。第一类信息描述给定服务的种类;第二类信息是服务的质量评价;最后一类信息是无限制的服务参数列表。OWL-S Profile提供了表示这些参数的机制,它可能包含提供对最大响应时间做出估计的参数,以及服务在地理位置上的有效性。

2　服务匹配过程

为了判断服务是否提供了用户需求的功能,基于OWL-S服务描述的服务本体为匹配模型提供了描述信息。服务本体的基本组件所提供的功能规范,包括服务执行的任务、服务提供的质量、服务特征。服务本体的一些属性是服务分类、服务双重性范围和服务参数。一个服务可以被建模为一系列的前置和后置约束。因此,服务的功能匹配就可以通过检查一系列给定的初始约束。

定义1:一个Web服务可用下面的表达式描述:

$WS_i(pre_i, post_i)$

其中,WS_i是Web服务的名字,pre_i是该服务的前置条件,$post_i$是该服务的后置条件。

定义2:一个Web服务请求可以用下面的表达式描述:

$WSR_j(pre_j, post_j)$

其中,WSR_j是Web服务请求的名字,pre_j是该服务的前置条件,$post_j$是该服务的后置条件。服务匹配过程如图2所示。首先利用匹配算法构建相应的匹配器,然后输入用户的需求描述,并从服务中心提取服务描述,通过匹配器将两个描述进行匹配。匹配时,需要Ontology本体库中的相关信息来确定他们之间的关系,最后输出满足用户需求的查询结果。Ontology本体库中存放了在本领域中所有服务的提供者达成共识的领域本体,其中包括:类和属性的分化与它们之间的关系、类的继承关系、类间的二元关系、定义良好的术语集、功能、输入事件签名以及在节点交互时涉及对象的逻辑语义。

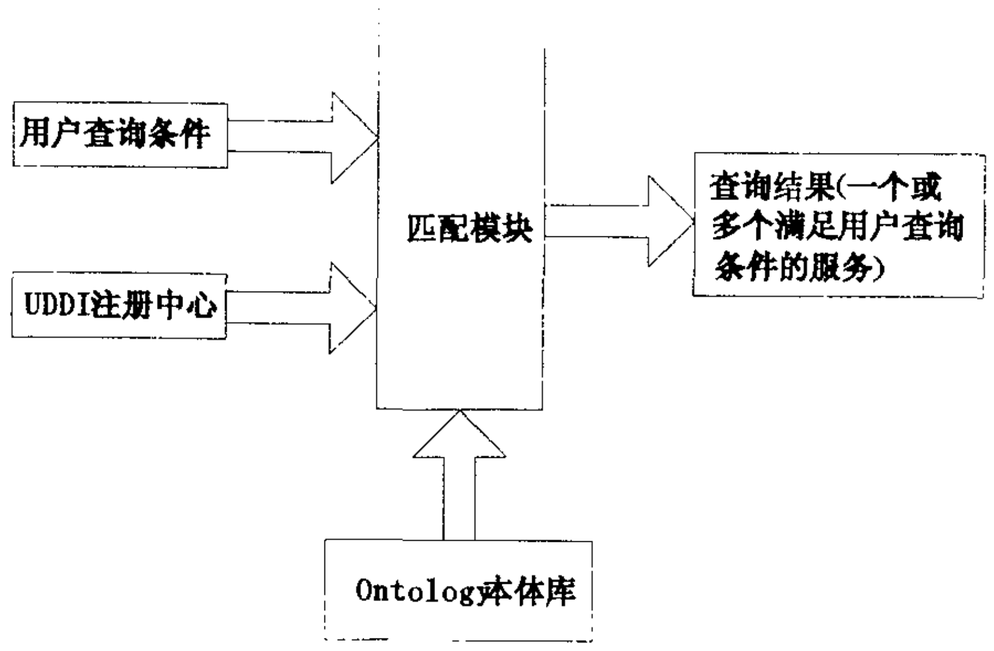

图2　Web服务匹配过程

在Ontology本体库中,根据OWL-S给出如下定义。

定义3:对于两个概念C_i和C_j,在Ontology本体库

中，如果 C_i 被定义成 C_j 的等价类，则称概念 C_i 和概念 C_j 语义相等，记为 $C_i \equiv C_j$；如果 C_i 被定义成 C_j 的子类，则称概念 C_j 语义包含概念 C_i，记为 $C_j \supseteq C_i$；如果 C_i 与 C_j 没有关联，则称概念 C_j 与概念 C_i 没有语义关系，记为 $C_j \neq C_i$。

定义 4：对于两个概念集合 SC_i 和 SC_j，如果对 SC_j 中的任一个概念 C_j，在 SC_i 中都存在一个概念 C_i，满足 $C_i \equiv C_j$ 或 $C_i \supseteq C_j$，则称 SC_i 语义包含 SC_j，记为 $SC_i \supseteq SC_j$；如果有 $SC_i \supseteq SC_j$ 且 $SC_j \supseteq SC_i$，则称 $SC_i \equiv SC_j$，此时称 SC_i 和 SC_j 语义相等。

在服务匹配的过程中，查询语句和每一个存放在注册中心的服务描述进行匹配。每当一个服务和用户需求匹配成功就将其记录到查询结果的列表中，并将它与其他匹配成功的服务进行匹配度的排序。根据 OWL－S 的服务描述，文中将服务描述信息分成两部分：前置条件和后置条件。前置条件是 ServiceProfile 中的输入；后置条件是 ServiceProfile 中的输出和影响属性的集合。在进行匹配时，需求和服务描述的所有前置条件和后置条件都要进行匹配。

3 匹配算法

利用 Ontology 知识从服务提供者的不同表达中抽象出其不同点和相同点，从而辨别出两个服务功能上是否相同。考虑到用户可能输入的属性匹配情况，给出如下定义：

定义 5：对于服务请求 $WSR_i(pre_i, post_i)$ 和服务 $WS_j(pre_j, post_j)$，如果 $pre_i \equiv pre_j$ 以及 $post_i \equiv post_j$，则称 WSR_i 与 WS_j 完全匹配。

定义 6：对于服务请求 $WSR_i(pre_i, post_i)$ 和服务 $WS_j(pre_j, post_j)$，如果 $pre_i \subseteq pre_j$ 以及 $post_i \subseteq post_j$，则称 WSR_i 与 WS_j 父包含匹配。

定义 7：对于服务请求 $WSR_i(pre_i, post_i)$ 和服务 $WS_j(pre_j, post_j)$，如果 $pre_i \supseteq pre_j$ 以及 $post_i \supseteq post_j$，则称 WSR_i 与 WS_j 子包含匹配。

定义 8：对于服务请求 $WSR_i(pre_i, post_i)$ 和服务 $WS_j(pre_j, post_j)$，如果 $pre_i \neq pre_j$ 或者 $post_i \neq post_j$，则称 WSR_i 与 WS_j 完全不匹配。

3.1 简单条件匹配

表面上来看，一个简单条件 A_1 是否与另一个简单条件 A_2 匹配取决于 A_1 是否包含 A_2。这种约束的包含是决定于它们的含义的，例如，当 $A_1 \Rightarrow A_2$ 时，则 A_1 包含 A_2。正式地说，当 $A_1 \wedge \neg A_2$ 为假的时候，才会有 $A_1 \Rightarrow A_2$。

匹配步骤如图 3 所示。

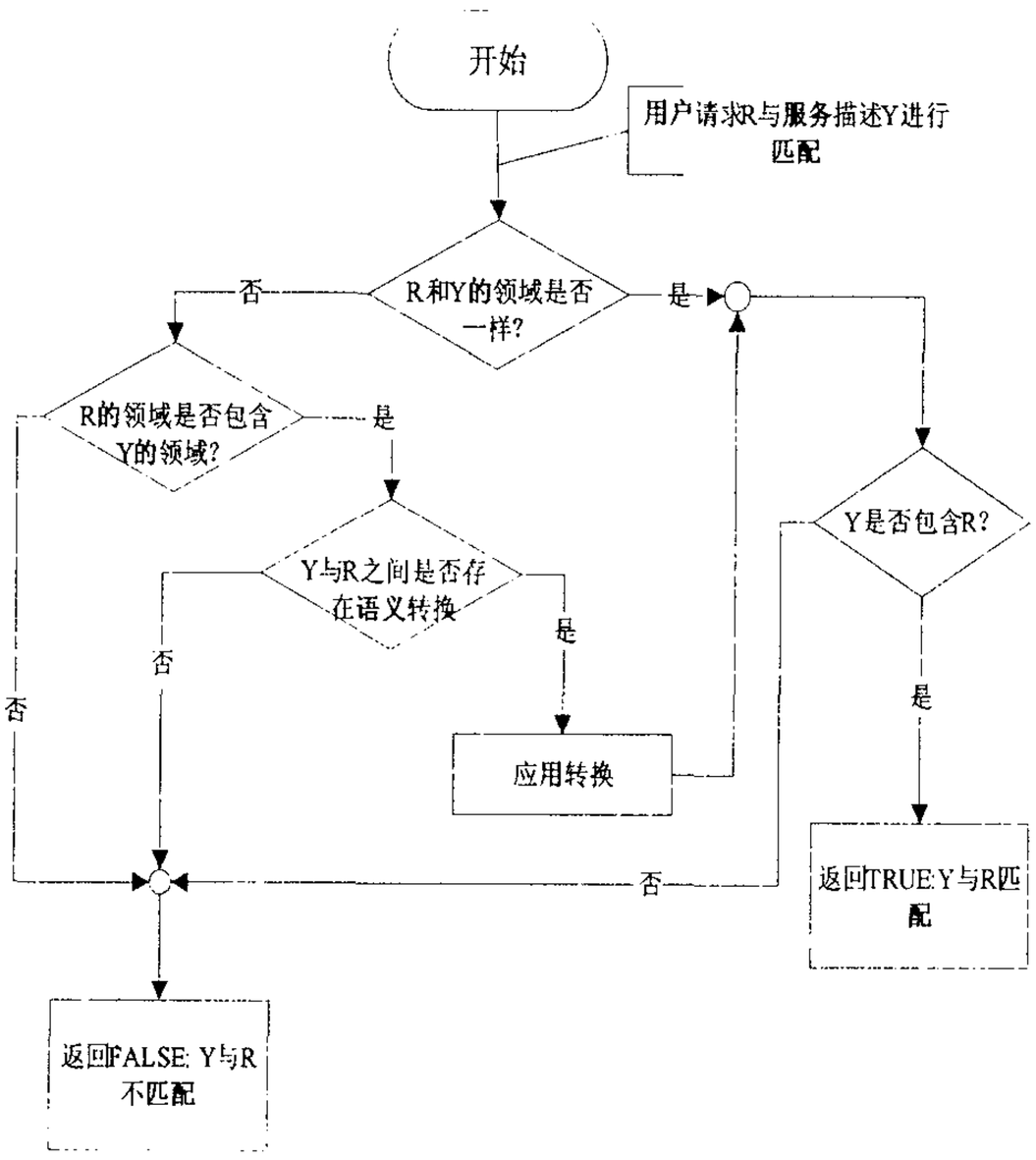

图 3 简单条件匹配过程

3.2 复合条件匹配

复合条件是由简单条件通过与和或连接起来的条件，对 3.1 节中的简单条件的匹配方法进行扩充就可以得到复合条件的匹配方法。复合条件的功能可替换可以通过考虑所涉及的简单条件的可替换性以及 $\wedge$ 和 $\vee$。使用德摩根法则就得到复合条件的匹配结果。

3.3 匹配度计算

由于可能有大部分服务不一定能够完全和用户的需求匹配，因此文中采用了匹配度来进行匹配测度[5]。匹配度是表示用户需求和服务描述的输入输出的匹配程度。

用如下算法计算两词组的相似度：

输入 $phrase_1$ 和 $phrase_2$，输出文本相似度。

(1) 先用 $phrase_1$ 作为外层循环，$phrase_2$ 作为内层循环；

(2) 对于每一个 $phrase_1$ 中的 word 得到与 $phrase_2$ 最大的单词相似度；

(3) 将这些最大相似度相加，得到 sum_1；

(4) 令 $sim_1 = sum_1 / phrase_1.count$；

(5) 然后用 $phrase_2$ 作为外层循环，$phrase_1$ 作为内层循环；

(6) 对于每一个 $phrase_2$ 中的 word 得到与 $phrase_1$ 最大的单词相似度；

(7) 将这些最大相似度相加，得到 sum_2；

(8) 令 $sim_2 = sum_2 / phrase_2.count$；

(9) 令 $sim = (sim_1 + sim_2)/2$；

(10) 返回 sim。

其中,sim 表示 similarity,即相似度。

这个算法的基本思想就是将两个关键词列表中能够匹配最好的关键词信息都保留下来进行综合计算,需注意如下几个方面:

1)该算法是一个对称算法,即不论输入为 $phrase_1$ 和 $phrase_2$,还是 $phrase_2$ 和 $phrase_1$,相似度计算结果相同。

2)相似度在[0,1]区间。当两个词组中包含的词完全相同时,相似度为 1;完全不同时,相似度为 0。

3)时间复杂度大概为 n^2。

最后匹配的结果需要根据匹配度进行排序。排序时,首先考虑后置条件匹配度,而前置条件匹配度的值放在第二位,即只有当后置条件匹配度相同时才考虑前置条件匹配度。这是因为服务请求者最希望后置条件匹配度能够尽量地高。

3.4　匹配算法的应用

文中所提出的匹配算法综合考虑了 Web 服务本身所携带的领域信息、基于 OWL－S 的对服务的描述以及 Web 服务所必须满足的前置条件和后置条件,根据用户的请求查找最符合用户需求的服务。在科研管理系统的开发过程中,针对服务的匹配部分使用了上述匹配算法。

结果表明,使用上述算法可以加快服务的匹配速度以及匹配精度。

4　总　结

OWL－S 提出了全新的表示 Web 服务功能的方法。将这种表示应用到服务匹配中,使得服务和请求都是基于服务功能的,这相对于 UDDI 机制进行服务查找增强了灵活性。今后还可考虑将 UDDI 和功能匹配结合起来,使得用户能够更好地进行服务查询。另外,由于大多数的复杂任务都是由多个简单服务合成来完成的,因此只是对 ServiceProfile 进行匹配是远远不够的,还应该将 ServiceModel 的信息考虑进来。这些都是今后的研究重点。

参考文献:

[1] Jaideep R, Anupama R. Understanding web services[C]// IT Professional, Volume 3. Washington: IEEE Computer Society, 2001:77－78.

[2] Sycara K, Klusch M. Brokering and matchmaking for coordination of agent societies: ni, ed. Coordination of internet agents [M]. London, UK: Springer－Verlag, 2000.

[3] W3C. OWL Web Ontology Language Guide[EB/OL]. 2004. http://www.w3.org/TR/owl－guide.

[4] OWL－S Home Page[EB/OL]. 2003. http://www.daml.org/services.

[5] Sycara K, Klusch M. Dynamic service matchmaking among agents in open information environments[J]. ACMSIGMOD Record, 1999, 28(1):47－53.

(上接第 15 页)

成结果缓冲区边界多边形的内环。如果是逆时针方向,面积(绝对值)最大的多边形是缓冲区边界的最终有效组成部分,构成结果缓冲区边界多边形的外环。其它逆时针方向的多边形为岛屿多边形,不是缓冲区最终有效边界的组成部分。图 6 是线缓冲区最终有效边界图。

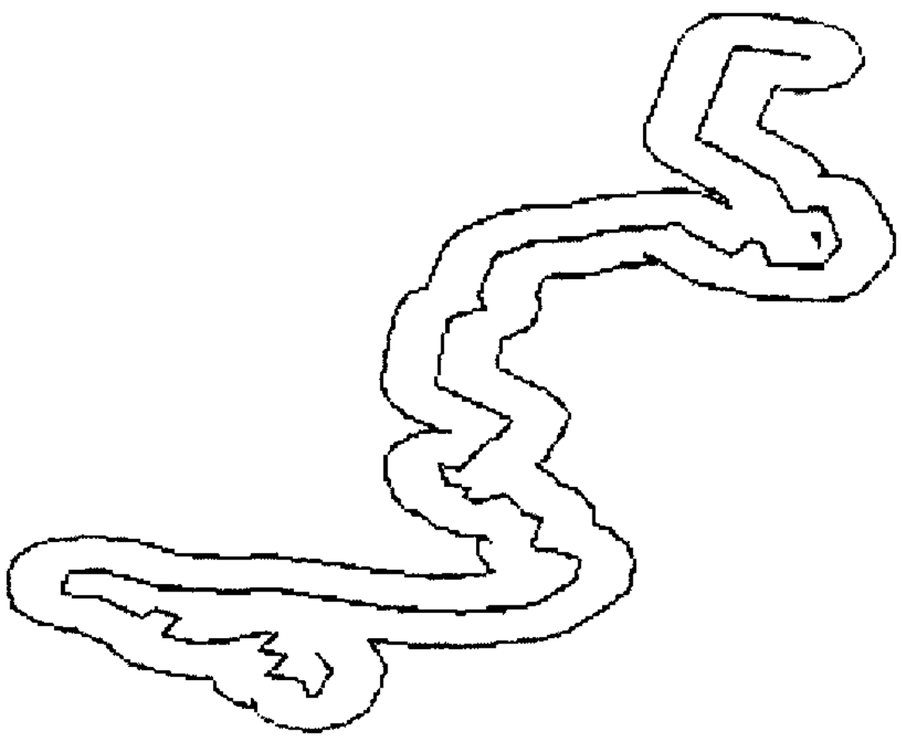

图 6　线缓冲区最终有效边界图

5　结　语

文中在缓冲区凸角圆弧法的思想基础上,提出并运用半径旋转法,简化了求平行线和尖锐角的光滑矫正过程;对缓冲区生成过程中出现的特殊问题进行了处理,最大限度地保证了目标缓冲区生成的有效性;并引用递归方法,很好地管理和存储了自相交多边形,进一步形成了一套新的完整有效的缓冲区矢量生成算法。算法已编程实现,具有较好的效果。

参考文献:

[1] 黄杏元, 汤　勤. 地理信息系统概论[M]. 北京:高等教育出版社,1990:130－133.

[2] 吴立新,史文中. 地理信息系统原理与算法[M]. 北京:科技出版社,2003.

[3] 程鹏根,龚健雅. 机助制图中平行线的绘制方法及其特殊问题的处理[J]. 武汉测绘科技大学学报,1994(1):43－52.

[4] 毋河海. 关于 GIS 中缓冲区的建立问题[J]. 武汉测绘科技大学学报,1997,22(4):358－364.

[5] 王家耀. 空间信息系统原理[M]. 北京:科技出版社,2004.