

本体存储技术研究

鲍文¹, 李冠宇²

(1. 大连海事大学 计算机科学与技术学院, 辽宁 大连 116026;

2. 大连理工大学 管理学院, 辽宁 大连 116024)

摘要: Ontology 是对一个特定领域中重要概念的共享的形式化的描述, 由于具有明确性和共享性, 它可以作为领域内不同主体之间进行交流的语义基础; 更进一步的, Ontology 可以帮助机器理解文档表达的语义信息。语义网络是 Ontology 的一个重要应用场景, Ontology 用来描述网络资源的语义, 从而使机器具有自动管理网络信息的能力。那么巨大的数据规模是语义网络环境下 Ontology 数据存储管理面临的一个突出问题, 所以介绍了本体存储的方法、存储模式及几种典型的本体存储管理系统。讨论了当前本体存储模式的问题并展望了未来的发展方向。

关键词: 存储模式; 本体存储管理系统; Sesame

中图分类号: TP18

文献标识码: A

文章编号: 1673-629X(2008)01-0146-05

Ontology Storage Technology Research

BAO Wen¹, LI Guan-yu²

(1. Computer Science and Technology College, Dalian Maritime University, Dalian 116026, China;

2. Management Department, Dalian University of Technology, Dalian 116024, China)

Abstract: Ontology is formal specification of shared conceptualization in certain domain. For it is explicit and shared, ontology can be used as the semantic foundation of communication between different agents. Further, ontology can help machines understand the semantics in documents. Semantic Web is an important application scenario of ontology. Ontology is used to describe the semantics of Web resources, and enable machines do Web information management automatically. The very large volume of data is a significant problem of ontology data storage management in Semantic Web environment. An introduction of ontology storage is given, including methods, schema and several typical ontology storage management system. Finally, the shortage of current ontology storage pattern are discussed and future researches in this field are suggested.

Key words: storage schema; ontology storage management system; Sesame

0 引言

随着语义网络中本体中资源的增加, 本体的规模越来越大, 结构越来越复杂, 此时, 本体查询和管理的效率成为人们普遍关注的问题。如何在本体存储管理系统中合理地存储大规模的本体, 支持高效的本体的管理是一件很有意义且具有挑战性的任务。

1 本体存储的方法

语义网的应用需求促进 Ontology 存储管理工作的发展, 目前已经出现了很多 Ontology 存储管理系

统, 按照存储介质不同可以分为基于主存、基于文件系统、基于关系数据库三类本体管理系统。

1.1 基于主存方法

这一类的 Ontology 数据管理工作的特点是将 Ontology 数据全部导入内存, 按照某种结构进行组织; 在内存结构上执行数据的查询操作。此方法具有很高的运行效率, 但只能处理有限规模的数据。由于是内存数据管理, 不存在磁盘更新的问题。OWLIm^[1]和 OWLJessKB^[2]是两个典型的基于主存的 Ontology 存储管理系统。

1.2 基于文件系统存储方法

基于文件系统的存储: 该方式实现起来比较简单, 很多本体相关工具都支持对文件格式的本体进行存取。但是, 这种方法不仅效率低, 而且很难适应数据量较大的情况。基于文件系统的存储方式一般只适用于规模比较小的本体, 对于规模比较大的本体需要大量

收稿日期: 2007-04-03

基金项目: 国家自然科学基金资助项目(60672031)

作者简介: 鲍文(1982-), 男, 辽宁本溪人, 硕士研究生, 研究方向为智能信息处理; 李冠宇, 博士, 教授, 研究方向为智能信息处理、信息系统与数据库、软件理论与工程。

的内存管理工作,而对于直接以 XML 格式这样子一种树形结构组织的文件来表示的 RDF 数据,当文件很大时,要把握 RDF 模型数据全局的结构,必须通过对文件进行反复的扫描,大量的数据换进换出工作,对系统的效率是一个很大的考验。而且为了保证系统的并发性,必须要建立相关的并发控制和事务管理系统。早期的一些 Ontology 数据管理工作是基于文件系统实现的,它们用简单的文件格式存储 Ontology 数据并支持一些基本的操作。这类工作主要用来编辑和建立 Ontology,并不是为大规模 Ontology 数据的存储和查询管理服务的,例如 OntoEdit^[3],Protégé^[4]。

另外一些 Ontology 数据管理工作基于文件系统建立 Native 的 Ontology 数据管理系统:设计专门的存储模式,并基于 Native 存储实现 Ontology 数据查询等管理操作。Kowari^[5]是一个典型的 Native Ontology 数据管理系统,它在存储设计中利用数据冗余提高查询效率。

1.3 基于关系数据库存储方法

用关系数据库存取本体,关系数据库技术发展成熟,关系模式容易建立查询、便于事务处理、便于备份,充分的关系技术支持,大多数现有的 Ontology 数据管理工作使用关系或对象-关系数据库管理系统作为后台存储,代表系统包括 Sesame^[6],Rstar^[7],Jena^[8],3store^[9]等等。

基于关系数据库存储 Ontology 可能有多种模式设计,现有的包括早期的水平表模式、垂直表模式、分解表模式、混合表模式和后来广泛为 Ontology 数据管理系统采用的 Sesame for RDB 存储模式及 Sesame for ORDB 存储模式。

1.3.1 水平表模式

该模式只在数据库中保留一张通用的表,这个对象拥有多少属性,存储它的表就要有多少字段。表中的列是本体中的属性,本体中的每个实例都是该表中的一个记录。文献[10]提到了这种存储模式。这种存储模式比较简单,但是该通用表包含了大量的列,进行查询操作会比较方便,但是也存在如下的问题:一是字段数目太多,目前的数据库系统对一张表拥有的字段数目都加以限制,如 DB2 和 Oracle 中都限制数据库的字段数目最多为 1012 个,这对很多大型 Ontology 来说是远远不够的;二是数据库的表结构很稀疏,本体中类的每个实体,如果只有几个属性,但是要占数据库表的一行,必须把没有用的属性设置成空,这将导致大量的空字段。另外该通用表的模式会不断变化,即随着本体的修改需要不断地增加或删除表中的列,而在当前数据库系统中表模式变化的代价会很大。

1.3.2 垂直表模式

这种模式包含一张三元组表,表中的每个实例都对应于一个 RDF 三元组。在这种模式下需要将本体中的所有信息都使用 RDF 三元组(即 Subject, Predicate, Object)来表示。文献[11,12]使用这种存储模式。这种模式设计简单,并且模式稳定,随着本体的修改只需要修改表中相应的元组。另外,该模式通用性好,因为现有的本体模式都可以转换为 RDF 模型来表示。但是,这种模式的可读性差,设计有关的查询语句比较困难。除此之外,该模式最大的不足在于,对于每个本体查询都必须搜索整个数据库,特别是那些需要进行表连接的查询效率非常低。

1.3.3 分解表模式

该模式与水平模式和垂直模式的一个显著的区别是它使用了若干张表,其基本思想是将数据库进行模式分解。根据分解的对象不同,现有的采用分解模式的方法有两种。

一种分解模式是基于类的分解模式,即为本体中的每个类都创建一张单独的表,表名为类名,表中的列为类的属性。这种模式有点类似于水平模式,只不过具有更小的粒度,每张表就是一个类,这样每张表的列由那些声明了 domain 为该类的属性组成,保证了表的非稀疏性。该模式结构清晰,而且最大的优点是能够高效地查询一个或一组实例的属性值。但是很难适应本体动态变化的情况,因为随着本体中类或者属性的变化,表结构都要随着变化。另外一种分解模式是基于属性的分解模式,即为本体中的每个属性创建一张单独的表,表名为属性名,每张表都只包含两个列,分别代表 RDF 三元组中的 Subject 和 Object。文献[10]采用为每个属性创建一张表的方法,但是,在该模式中对类的隐含实例的查询代价太大。而且,在现有的这两种分解模式的方法中,随着本体的变化都要不断地创建和删除表,而在数据库系统中创建和删除表的效率低、代价大。

1.3.4 混合表模式

该模式通常将上述几种模式进行混合使用。文献[13]提出了一种将基于类的分解模式与基于属性的分解模式混合使用的存储模式,即在本体中定义一个类就为该创建表,在本体中定义一个属性就为该属性创建一个表,表名分别为相应的类名和属性名。然而,与基于类的分解模式不同的是,该模式中在类对应的表中不记录相应实例的所有信息,只记录属于该类的实例的 ID。而实例在各个属性上的取值分别记录在各个属性对应的表中,所以和基于属性的分解模式类似,该模式中在属性对应的表中仍然需要两列:

Subject 和 Object。当本体中只包含一百个左右的类时,这种模式对于简单查询运行得很好,效率很高。但是,如果本体中的类比较多,这种方式就会存在一些问题,例如:数据库无法容纳那么多张表,而且查询效率很低。

1.3.5 Sesame for RDB 存储模式及 Sesame for ORDB 存储模式

Sesame 实现了基于关系数据库 MySQL 及基于对象关系数据库 PostgreSQL 的存储方法,对应两种不同的存储模式,分别参见图 1 和图 2。这两种存储模式的区别是:Sesame for RDB 模式用一张表(type 表)存储了所有实例的类型信息,而 Sesame for ORDB 模式为每个类单独建立一张关系表,专门单独存储这个类的实例,并且用关系表之间的 subtable 关系记录类之间的 subClass 关系。实例的属性取值信息的存储方法也有同样的区别。

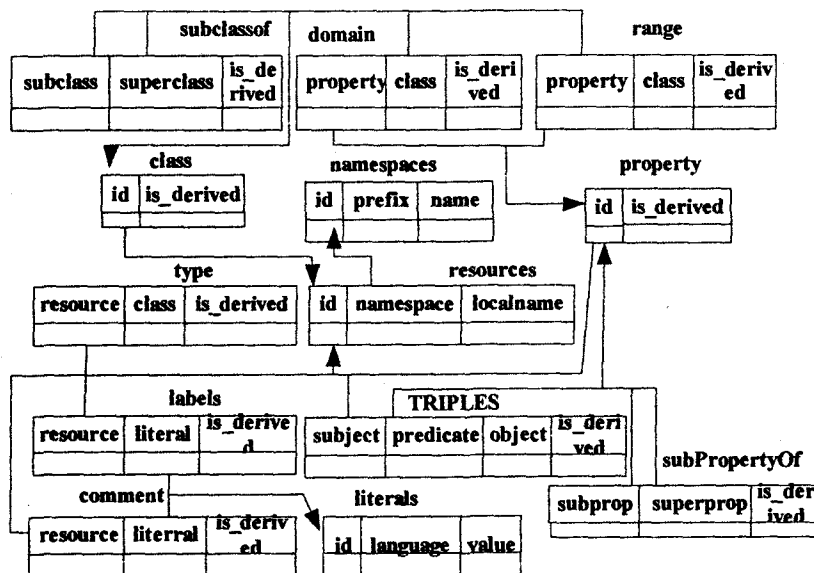


图 1 Sesame for RDB 模式

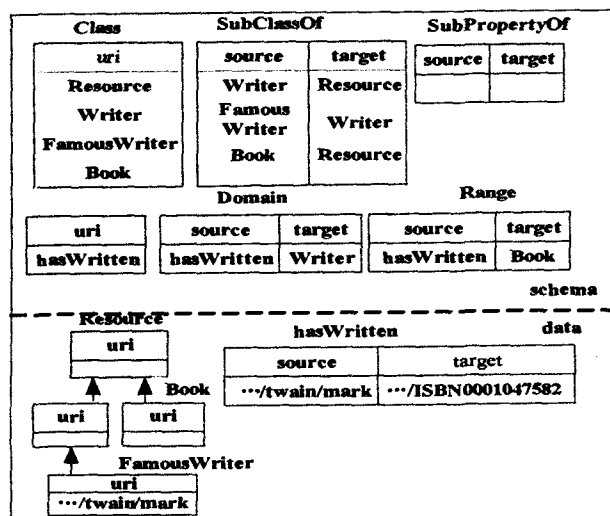


图 2 Sesame for ORDB 模式

2 几种典型的本体存储管理系统

2.1 Sesame

Sesame 是针对 RDF 数据管理提出的一个通用的系统框架,它是一个开源项目,提供了非常开放的 API 接口,使得人们可以很方便地集成不同的存储系统、推理引擎以及查询引擎等。它本身提供了基于关系数据 (MySQL, PostgreSQL, Oracle)、基于文件系统以及基于主存的存储系统的实现,提供了推理算法以及更新算法的实现,支持自定义的查询语言 SeRQL 以及 RDQL。Sesame 旨在提供一个通用的系统框架,它不规定如何设计存储模式,也不规定如何实现推理,而是通过定义一组接口来规定存储模块以及推理模块等应该完成什么样的功能,方便人们可以集成不同的实现模块。

图 3 中的 RDF Model 是指不同的存储系统应该提供的 RDF 数据模型 Sail (Storage And Inference Layer)

API 提供在 RDF Model 层上 RDF 数据的存储以及推理功能,同时为查询引擎提供数据存取接口, Sesame 的推理基于用户定义的规则,用户可自由定义规则以及规则之间的触发关系;Rio 表示 RDF I/O,它包含许多的 RDF 文档解析器 (Parser) 以及生成器 (Writer),解析器将原始的 RDF 文档解析成 RDF 语句 (Statement),然后由 RDF Model 层负责语句的存储,生成器将 RDF 语句重新转换为文档。Repository API 是用于封装底层的 Query API 以及 Rio API,对用户提供一个统一的接口。用户的应用程序可通过本地的 Repository

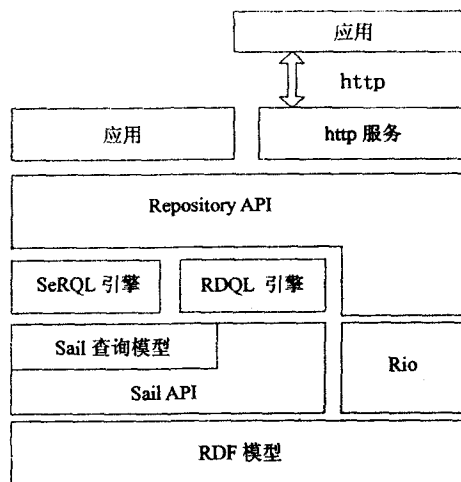


图 3 Sesame 系统体系结构

ry API 或是通过 Http 协议访问来访问 Sesame。

2.2 Jena

Jena 是由 HP 实验室开发的综合系统,它的系统框架见图 4。它包括了一个易于面向对象使用操作 RDF 的 API,一个 APR RDF/XML 解析器,一个 RDF/XML 输入器,可以使用 RDQL 查询语言,支持 DAML,即有持久稳定的存储能力。Jena 的存储功能包含有 API 的三种执行方式:第一是存储它的数据在主要的存储器中;第二种是把数据存储到关系数据库中;第三种是用 Sleepycat 软件的开源的植入式数据库 Berkeley DB。关系数据库的执行方式可以用任何支持 JDBC 的数据库。配置表允许对一个特殊的数据库进行特殊化的处理。如同关系数据库一样,Berkeley 数据库也是持久稳固的,尽管它缺乏关系数据库的事物处理支持能力,但它比关系数据库可以快一个数量级。

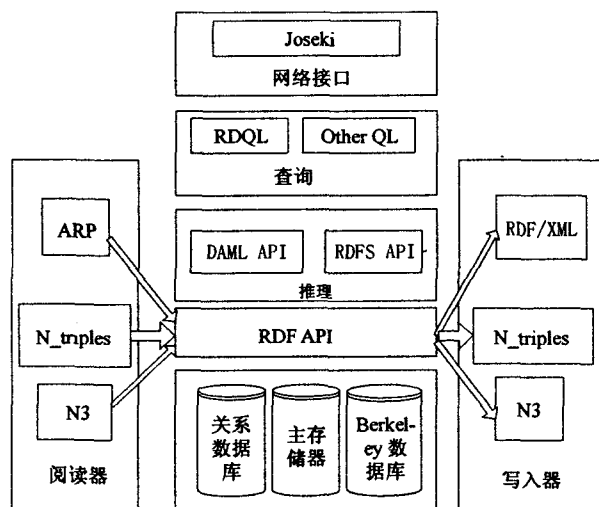


图 4 Jena 系统体系结构

2.3 KAON

KAON^[14] (The Karlsruhe Ontology and Semantic Web Infrastructure) 是德国 Karlsruhe 大学的一个科研项目。该项目致力于为语义 Web 提供所需的基础本体系统和相关工具。它针对基于本体的上层商业应用的需求提供了一个开放的本体管理软件,为本体的存储、创建和标识提供了一个全面的支撑平台。图 5 是 KAON 平台的体系结构。

RDF API 采用的是斯坦福大学的 RDF API,但做了相应的重写和扩展,为上层应用或 KAON API 提供了本体的内存存储机制。目前,RDF API 不但包括了一个 RDF Parser 可解析 RDF 文件,还包括了 RDF Serializer 可以将本体序列化到关系型数据库和文件中去。

KAON API 为应用屏蔽了底层的存储机制,但实际上它也可以通过多种方式访问 KAON 本体,一种是

通过 RDF API(然后通过 RDF Server),另一种是直接通过 Engineering Server。KAON API 的定义有其合理性,例如它有 Observable 这个设计范式,可以让应用自动得到本体修改或升级的消息。

RDF Server 和 Engineering Server 都基于关系型数据库,可以提供并发控制和交易机制,它们还可以直接支持 EJB(可选),提供 Entity Java Beans 接口。不同的是 RDF Server 面向 RDF,Engineering Server 面向 KAON 自己的本体标准。

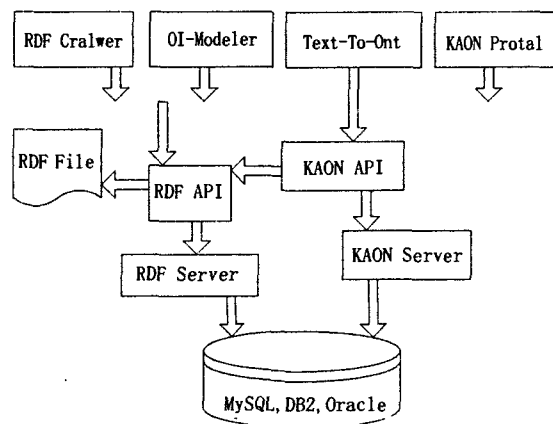


图 5 KAON 平台体系结构

KAON 的 RDF Crawler 用于 crawling,并综合 Web 上的 RDF 信息。可以把 Crawling 的深度、指定范围等这样的参数放到配置文件中,并把结果存于本地文件。KAON Portal 用于建立一个多语种的、基于本体的门户网站。需要先将网站内容进行本体标识。在网站上可以基于本体进行可视化的浏览导航。它把显示与内容做了严格的分离,有很好的可配置性。KAON 的 OI-Modeler 是一个本体的建模工具,用于可视化地建立文件并维护它。

3 结语与展望

总之,现有的本体存储方法解决了本体存储管理的应用,大大提高了本体高效的管理。但是目前现有的方法还存在很多问题:

(1) 现有的大多数研究工作尝试基于关系数据库的方式解决问题。但是,关系数据库并不是针对本体数据的特点设计的,本体数据复杂的图形结构和关系数据简单的扁平结构存在很大差异。基于关系模型管理本体数据,需要把复杂的本体图拆分成简单的关系存储,并把基于图上的查询转换成大量关系查询的连接。模式上的不匹配从根本上制约了基于关系数据库的方法处理大规模本体数据的能力。

(2) 由于查询推理隐含的本体数据耗费大量时间,基于关系数据库的本体管理方法往往预先将隐含数据

转化为显式数据物化在存储中。这种做法虽然可以保证查询效率,但是增加了更新的代价。在更新原有数据时,如何动态维护物化的隐式数据,使二者保持推理的一致性,是一个代价昂贵的难题。

因此在以后的研究中,设计一种专门的存储模式,它基于纯 XML 数据库系统,借鉴 XML 树形模式,保持用户模式的层次,为高效的查询处理和更新提供有力支持。

参考文献:

- [1] Kiryakov A, Ognyanov D, Manov D. OWLIM - a Pragmatic Semantic Repository for OWL[C]//In Proc. of Int. Workshop on Scalable Semantic Web Knowledge Base Systems (SSWS). New York, USA:[s. n.],2005:182-192.
- [2] Kopena J, Regli W C. DAMLJessKB: A Tool for Reasoning with the Semantic Web[C]//In Proc. of International Semantic Web Conference (ISWC) 2003. Sanibel Island, FL, USA:[s. n.],2003:628-643.
- [3] Sure Y, Erdmann M, Angele J, et al. OntoEdit: Collaborative Ontology Engineering for the Semantic Web[C]//In Proceedings of the first International Semantic Web Conference 2002 (ISWC 2002). Sardinia, Italy:[s. n.],2002.
- [4] Gennari J H, Musen M A. The Evolution of Portege - 2000: An Environment for Knowledge - based Systems Development [J]. International Journal of Human - Computer Studies, 2003, 58(1): 89-123.
- [5] Wood D, Gearon P, Adams T. Kowari: A Platform for Semantic Web Storage and Analysis[C]//In Proc. of WWW 2005. Chiba, Japan:[s. n.],2005.
- [6] Broekstra J, Kampman A, Harmelen F. Sesame: A Generic Architecture for Storing and Querying RDF and RDF Schema [C]//In Proc. of the 1st International Semantic Web Conference. Sardinia, Italy:[s. n.],2002:54-68.
- [7] Ma Li, Su Zhong, Pan Yue, et al. RStar: An RDF Storage and Query System for Enterprise Resource Management[C]//In Proc of CIKM 2004. New York, NY, USA:[s. n.],2004:484-491.
- [8] Carroll J J, Reynolds D, Dickinson I, et al. Jena: Implementing the Semantic Web Recommendations[C]//In Proc. of WWW 2004. New York, USA:[s. n.],2004:74-83.
- [9] Harris S, Gibbins N. 3store: Efficient Bulk RDF Storage [C]//In Proc. of the 1st International Workshop on Practical and Scalable Semantic Systems. Sanibel Island, Florida, USA:[s. n.],2003:1-15.
- [10] Agrawal R, Somani A, Xu Y. Storage and querying of e-commerce data[C]//In: Peter M G A, Paolo A, eds. Proc of the 27th VLDB. Roma: Morgan Kaufman Publishers Inc, 2001:149-158.
- [11] McBride B. Jena: implementing the Rdfmodeland syntax specification[R]. Bristol: Hewlett Packard Laboratories, 2000.
- [12] Alexaki S, Christophides V, Karvounarakis G, et al. On storing voluminous RDF description: the case of Web portal catalogs [C]//In: Mecca G, Simeon J, eds. Proc of the 4th WebDB in conjunction with ACM SIGMOD'01. Bristol: Hewlett Packard Laboratories, 2001:43-48.
- [13] Pan Z X, Heflin J. DLDB: extending relational database to support semantic Web queries [C]//In: Vclz P, Decker S, eds. Proc of the 1st PASS. Santa Barbara: Informal Proceedings, 2003:43-48.
- [14] Volz R, Oberle D, Staab S, et al. KAON SERVER - A Semantic Web Management System [C]//In: Proceedings of the WWW - 2003 Alternate Track on Practice and Experience. Budapest, Hungary:[s. n.],2003.

(上接第 145 页)

成正比,而改进的滤波算法只与窗口中像素点数成正比。可见改进的算法相对于矢量中值滤波算法而言,计算量显著下降,滤波效果好,算法简单且易于实现。对于不同噪声比例的彩色图像,文中改进算法用时均明显优于传统矢量中值滤波算法。但是从峰值信噪比结果比较来看,文中算法在噪声比例比较小时有一定效果,当噪声比例变大时,与矢量中值滤波算法比较,改进效果不是很明显,同时在滤波过程中也丢失了相关的颜色信息。在算法实现过程中使用了 3×3 的方形窗口以及包含它的十字形窗口作为双滤波器的叠加组合,噪声比例较大时也可考虑增大滤波器窗口的大小^[6](如 $5 \times 5, 7 \times 7$),以此来改善滤波效果,这有待于今后的研究和进一步改进。

参考文献:

- [1] Lukac R. Adaptive vector median filtering[M]. Pattern Recognition Letters, 2003, 24(12): 1889-1899.
- [2] 曲延锋. 有效去除图像中脉冲噪声的新型滤波算法[J]. 计算机辅助设计与图形学学报, 2003, 15(4): 397-401.
- [3] Qu Yan - Feng, Xu Jian, Li Wei - Jun, et al. New effective filtering algorithm for the removal of Impulse noise from images [J]. Journal of Computer aid design & Computer graphics, 2003, 28(2): 361-364.
- [4] 朱其刚. 具有细节保护特性的多级中值滤波算法[J]. 山东科技大学学报: 自然科学版, 2005, 24(3): 73-75.
- [5] 章毓晋. 图像工程(上册)[M]. 北京: 清华大学出版社, 2002.
- [6] Zoican S. Improved median filter for impulse noise removal [J]. TELSIKS Serbia and Motenegra, Nii, 2003, 10: 681-684.