文章编号:1001 - 2486(2002)06 - 0071 - 06

流域变换的串行与并行策略研究*

周海芳,蒋艳凰,杨学军
(国防科技大学计算机学院,湖南 长沙　410073)

摘　要:流域变换是数字形态学中用于图像分割的一种经典方法,其并行化问题成为近年来研究的重点。首先给出了流域变换的数学模型,并归纳列举了几种离散情况下的形式化定义;其次分类总结了近年来流域变换串行算法研究的新进展,从而在此基础上重点讨论了相应的并行化策略。详细分析了设计并行流域算法需要考虑的几个问题;并比较评价了现有并行算法的性能特点,得出了一些结论;最后提出了有待进一步研究的问题。

关键词:流域变换;图像分割;积水盆;分水岭;并行算法;分布存储;域分解;加速比
中图分类号:TP391.41;TP301.6　　文献标识码:A

Research on Serial and Parallel Strategies of Watershed Transform

ZHOU Hai-fang, JIANG Yan-huang, YANG Xue-jun
(College of Computer, National Univ. of Defense Technology, Changsha 410073, China)

Abstract: Watershed transform is a classical method of image segmentation in mathematical morphology, and its parallelism is an important problem. This paper starts with a mathematical model of watershed transform in topography, followed by its several formal definitions for digital case. Then some new serial watershed algorithms proposed in recent years are classified and analyzed, based on which their related parallel strategies are discussed in detail. Then we analyze some issues, which should be considered when designing parallel watershed algorithm, and give an evaluation of existing parallel algorithms, and draw some conclusions. Finally we point out the problems and challenges of future research.

Key words: watershed; transform; image segmentation; catchment basins; watershed line; parallel algorithms; distributed memory; domain decomposition; speedup

流域变换(又称分水岭变换)是一种有效的图像分割方法,该方法类似于区域生长技术,是在图像内进行连通区域(即集水盆)的标记。该方法可应用于工业、生物医学、计算机视觉及卫星和雷达遥感图像分析等领域。

SIMD 和 MIMD 并行模式下有关二值或灰值图像的连通区域标记及区域生长分割问题已有一些相关的研究。在上述的许多应用领域,大型的图像(如 1024×1024 或更大尺寸)是经常出现的而且要求快速处理,因此研究并行的流域算法具有现实意义和需求。此外,随着多处理和分布式系统的可用性和性价比的不断提高,这方面研究更具有应用价值。并行流域分割算法最初的一些研究工作可参阅文献[1 ~ 6]。然而,模拟泛洪的流域算法的直接并行化并非易事。事实上这些算法具有很强的递归性质。为此,本文有两方面的贡献:首先给出了流域变换的数学物理模型,并归纳列举了几种离散情况下的形式化定义;其次分类总结了近年来流域变换串行算法研究的新进展,并在此基础上重点讨论了相应的并行化策略。文章详细分析了设计并行流域算法需要考虑的几个问题,并比较评价了现有并行算法的性能特点,得出了一些有益的结论,指出值得进一步研究的课题,目的是研究具有良好的并发性、局部性、模块性和可扩展性且执行时间短的并行算法。

* 收稿日期:2002 - 05 - 08
基金项目:国家杰出青年科学基金项目资助(69825104)
作者简介:周海芳(1975—),女,博士生。

1 定义及其串行算法

1.1 定义

描述流域变换的经典形式是将图像看成是自然地貌中的地形表面,每一个像素的灰度级代表其位置的海拔高度,不同的区域称为集水盆,区域间的界限称为分水岭。应用到图像分割中,流域变换就是指将源图像转换为一个标记图像,其中所有属于同一集水盆的点被赋予同一个标记,同时用一个特殊的标记标识分水岭上的点[7]。由此提出了各种描述流域变换的形式化定义,下面我们给出数字空间中两种有代表性的定义。

1.1.1 基于浸没模拟的算法级定义

设 $D \subseteq Z^2$,设 f 为域 D 上的数字灰度图像,即 $f: D \to N$,$f(p)$ 表示像素 $p \in D$ 的灰度值,且 $h_{\min}$ 和 $h_{\max}$ 分别为 f 中最小和最大灰度值。定义递归过程使灰度级从 $h_{\min}$ 增长到 $h_{\max}$,同时与 f 的极小区相关的集水盆也顺序地扩大。用 X_h 表示在 h 级计算所得的集水盆的集合。一个在 $h+1$ 级属于阈值集 T_{h+1} 的连通域,要么是一个新的极小区,要么是 X_h 中一个集水盆的扩展(其中 $T_h = \{p \in D \mid f(p) \leqslant h\}$)。对于后者计算 X_h 在 T_{h+1} 范围内的测地影响区 IZ[8],并更新为 X_{h+1}。用 MIN_h 表示在 h 级所有极小区的集合。

定义 1 (基于浸没模拟的流域变换)

$$\begin{cases} X_{h_{\min}} = \{p \in D \mid f(p) = h_{\min}\} = T_{h_{\min}} \\ X_{h+1} = MIN_{h+1} \cup IZ_{Th+1}(X_h), \quad h \in [h_{\min}, h_{\max}) \end{cases} \tag{1}$$

分水岭则是 $X_{h_{\max}}$ 在域 D 中的补集:

$$Wshed(f) = D / X_{h_{\min}} \tag{2}$$

1.1.2 基于地形学距离的定义

定义 2 (离散域基于地形学距离的流域变换)

设 $f: D \to N$ 在定义域内存在极小区集合 $\{m_k\}_{k \in I}$,I 为顺序索引标记集合。某一极小区 m_i 的集水盆 $CB(m_i)$ 定义为所有的 $p \in D$ 且 p 到 m_i 的距离比到其它任何极小区 m_j 的距离更短:

$$CB(m_i) = \{p \in D \mid \forall j \in I \setminus \{i\} : f(m_i) + T_f(p, m_i) < f(m_j) + T_f(p, m_j)\} \tag{3}$$

其中 $T_f(p, m_i)$ 表示 p 到 m_i 之间的地形学距离。

变换形成的分水岭是不属于任何集水盆的点的集合:

$$Wshed(f) = D \cap [\bigcup_{i \in I} CB(m_i)]^c \tag{4}$$

1.2 串行算法

基于上述定义,目前存在两类流域变换的实现算法,一类是通过模拟泛洪过程确定集水盆,而另一类是应用矢量技术直接跟踪确定分水岭[1]。但是更多的工作集中在第一类算法的研究上,其中又分为生成分水岭和 0 - 宽度分水岭两类。

文献[10]回顾了历史上基于形态学操作模拟泛洪过程的一些效率较低的算法,并提出了定义 1 及其快速实现算法。该算法首先将源图像所有的像素按灰度级排序,然后逐级将相应的像素划归入比其灰度级低的直接邻域像素所属的集水盆中,至于对高地[10]的处理,是采用一个简单的 FIFO 队列以宽度优先的扫描顺序来模拟泛洪的过程。而在文献[1,2,11]中又提出了一种基于顺序队列(Ordered Queue)的方法。这种方法将像素存储在按灰度值递增的队列组中,即可以按泛洪的先后次序直接访问队列中的像素。另一种算法是从图像重建的角度考虑,对半形态学梯度图(源图像减去其经腐蚀操作后的图像得到的结果)进行积分,并将局部极小区作为有限个临界点[3,4,6]。这种算法的实质是计算极小区和其它所有像素之间的最短路径。由此,基于地形学距离函数的分水岭和集水盆的定义(定义 2)被提出[6,11]。按照这个定义,每个像素被划归入距其最近的极小区所属的集水盆,因此根据泛洪的先后顺序每个像素有一个前驱,即已计算距离值的邻域像素。另外,如果根据距离规定某种优先级的顺序,

那么就可以按照优先级的前后关系自下而上(爬山法)或自上而下(降雨法)地选择出最短路径,从而简化了计算的步骤。再者还可以采用向前或向后固定的扫描方式,相应点的距离值和标记仅基于其已经被扫描过的邻域像素的信息来确定,重复这一过程直到系统稳定为止。图 1 分类总结了串行的流域算法。

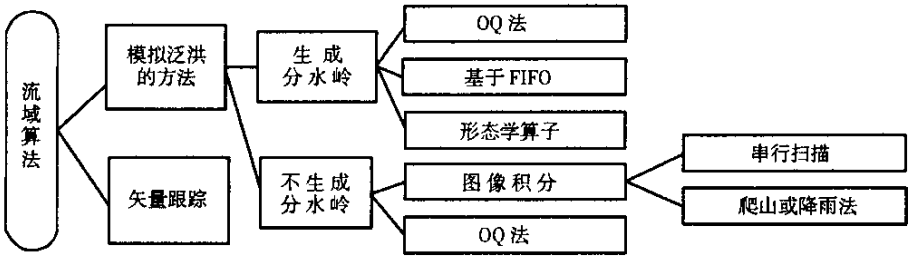

图 1 串行流域算法分类

Fig.1 The classification of serial watershed algorithm

2 并行化策略

2.1 分布存储实现

2.1.1 基于顺序队列 OQ 的爬山法

基于顺序队列的流域变换的并行化在文献[14]中讨论得比较详细。该算法没有构造分水岭,同步时以消息传递的方式与主进程通信,图像按块分布。计算步骤如下:

(1) 极小区检测

在每一个子图中使用一个 FIFO 队列进行宽度优先扫描,如果某一极小区跨越了不同的子域,在合并时处理机间要进行通信,这一过程可能会反复执行直到稳定为止。

(2) 基于局部 OQ 的泛洪操作

每个处理机在子域内部执行基于 OQ 的串行泛洪算法。为了能使泛洪结果传播到相邻的子域,可采用两种方法:一是处理机间在每一灰度级都保持同步[15],即分析那些与邻域相关的边界像素,它们的下降最快的或灰值相同的邻域像素可能存在于该子域的扩展区域中。当一个处理机到达同步级 h 时,要与其相邻的处理机交换扩展区域中像素的标记和灰值。通信和再泛洪要反复执行直到标记不再改变为止,这个稳定状态由主处理机判断。由于处理机一般不会在近似相同的时间内执行相同的代码,所以这种严格同步会引入相当多的空闲时间。而另一种方法是先在子域中执行所有灰度级的局部泛洪过程[14],然后再进行通信和重复泛洪的工作以处理边界问题,这种方法将大量减少重复泛洪过程中的通信时间。

2.1.2 完全下降化后的爬山法和降雨法

文献[16]讨论了经完全下降化的爬山法,另外还讨论了降雨法的实现。两者都要经历下列步骤:(i)极小区检测;(ii)完全下降化[7];(iii)基于爬山法或降雨法的泛洪。

第(i)、(ii)步经局部操作后都需要反复交换边界像素的信息直到稳定为止,以保证全局的一致性。而泛洪步骤则是在完全下降图的基础上给每个像素打上某一极小区的标记,使得每个像素到相应极小区有一条路径相通。从某个给定像素下降最快的邻域点中(可能有多个)任意选出一个传播标记,这个过程将 DAG 图转化为不相交的集合组成的森林,优点是减弱了非局部性,但结果与扫描次序相关。

对于降雨算法,森林的标记也在子域内进行,并用一个 FIFO 队列存放尚未解析的路径的根结点,只要子域内还存在尚未解析的路径,处理机间就要进行通信。但是由于算法可以局部地决定什么时候终止子域内的计算,所以不需要全局的归约操作,也不需要路径之间的重标记或同步。而在爬山法中,每个处理机通过一次光栅扫描初始化一个 FIFO 队列,放入子域中极小区的边缘像素,一个像素 p 从队列中移出并向其所有的上游邻域像素 q 传播标记。相邻处理机之间反复交换着其扩展区域中像素的

标记信息,并触发新的标记过程。当子域中所有的像素已被标记,则相应的处理机变为不活跃状态。总的来说,处理高地问题时采用以宽度优先的次序,而沿路径打标记时,降雨法是深度优先的搜索,爬山法则是宽度优先的搜索。

2.1.3 基于顺序队列的爬山法与连通区域算子相结合

Bieniek 等人在文献[9]中讨论了利用局部条件的并行实现方法,另外 Moga 等人在文献[14,17]中也讨论了这种方法的并行化。其并行化的主要思想是在所有的子域中独立地解决流域问题而无需同步。事实上,是先给像素分配临时的标记,然后再根据相邻子域的情况进行修正,边界像素的信息被存储在图或等价表中。通过归约算子计算全局的标记,具体步骤类似于 UNION-FIND 算法中的解析步。如果处理机数为 M,那么计算全局标记需要 $\log_2 M$ 步,这个步骤与数据的复杂度无关,但借鉴了连通区域标记问题的解决方法。一个类似的算法在文献[17]中讨论,与文献[9]不同的是跨子域的非极小区高地的全局下降距离信息是在泛洪前进行计算和修正,而不是在泛洪过程中进行,因此避免了因错误标记而执行的重标记过程。

2.1.4 基于串行扫描的并行流域变换

最后我们讨论一种基于串行扫描的流域算法[18]。虽然这种方法的串行执行时间较长,但在并行实现时能获得很好的可扩展性。实现的步骤包括在子域内进行的光栅扫描和反光栅扫描以及处理机之间的消息传递,这几步反复执行直到达到稳定状态为止。

并行实现有以下几步:(i)检测极小区;(ii)图像的完全下降化;(iii)标记极小区;(iv)图像积分。处理机之间必须交换子域边界像素的标记信息,即使子域内的扫描已达到稳定状态,但由于其它处理机可能引发边界标记的变化,因此仍需要触发新的扫描过程,全局的稳定状态由一个主处理进程检测。因为所有的子域大小相同,且每个处理机在光栅扫描模式下均执行简单的操作,所以各处理机到达同步点的时机相差较小,从而这种方法的负载较为平衡。

2.1.5 性能评价与结论

文献[9~18]分别给出了上述几种并行算法的实验数据,实验环境主要是两种:一是 128 个结点的 Parsytec Supercluster 系统[12],软件平台是 PIPS;另一种是 256 个结点的 CrayT3D MIMD 分布存储的并行系统,使用 MPI 软件平台。图 2、图 3 是我们综合分析和比较结果,图 2 是各种算法分别在 1 个结点和 4 个结点的系统上对一幅自然图像进行流域分割的执行时间的比较;图 3 是随处理结点的增多,各算法加速比的变化情况。

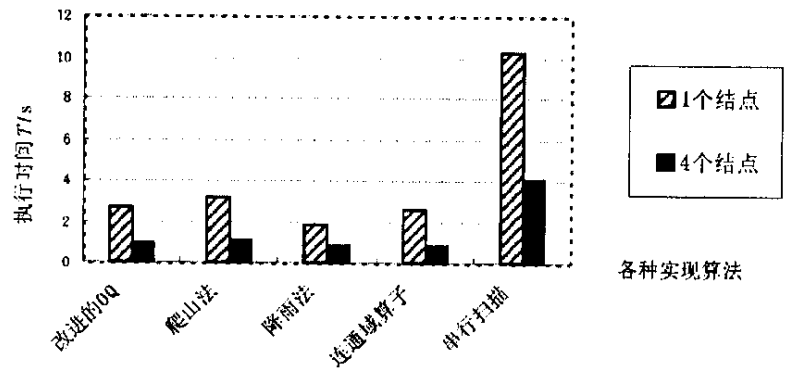

图 2 各流域并行算法执行时间的比较

Fig.2 The comparison of executing time of parallel algorithms

具体的分析如下(主要参考了在第二种平台上得到的数据):严格同步的 OQ 法不仅执行时间长,而且可扩展性很有限,效率和加速比在某一临界值后,随着处理机数的增加迅速降低,而改进后对大部分图像的分割处理加速比随着处理机数目的增长有适度提高,而且加速比的大小几乎是改进前的两倍。降雨法和爬山法的加速比曲线形状十分相似,区别在于降雨法的加速比的值相对较小,但算法的执行时

间较短。而结合连通域算子的算法比爬山法或降雨法能获得更好的可扩展性和更短的执行时间,处理机数不超过 64 时都可获得接近线性的加速比,处理机数为 128 时效率保持在 10% ~ 50%之间。串行扫描方法执行时间较长,但在多数情况下都能得到接近理想线性加速比的结果,而且对于同样数量的处理机,图像规模越大,加速比越大。从上述分析中我们得到以下一些结论:

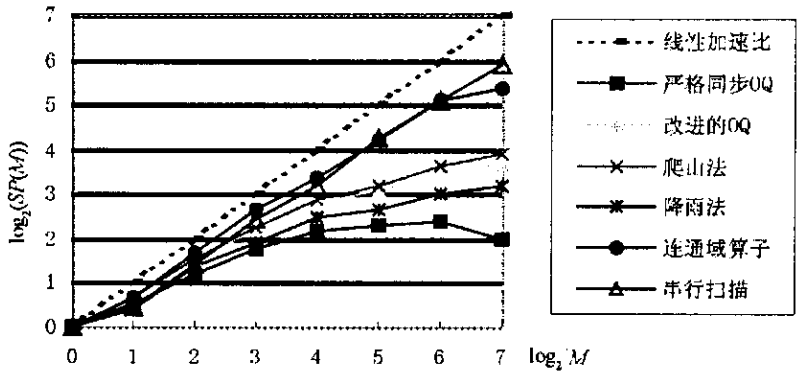

图 3 各流域并行算法加速比曲线(M:处理机数;$SP(M)$:加速比函数)

Fig.3 Speedup of algorithms(M: number of processors; $SP(M)$: speedup)

低效的算法(如串行扫描)却具有很好的可扩展性。基于顺序队列的方法相对较快,但可扩展性最差,因为在分布存储系统中局部的顺序队列需要通过适当的同步来保持全局的次序。而爬山法和降雨法相对顺序队列的方法在性能上并没有显著的提高。获得最好结果的是顺序队列与连通区域算子相结合的方法,该方法的数据相关性较弱,因为连通区域算子的使用使得标记修正工作具有固定的时间复杂度而与内容无关。所有算法的性能都与数据相关,如在处理含有较大范围的蛇形图案的图像时都有不同程度的性能衰减,这导致负载的不平衡,而且由于过渡的重复标记而致使加速比很小甚至随着处理机数的增加而降低;而且算法并不总是随着问题规模的增大而获得更好的性能。

2.2 共享存储实现

设计共享存储的并行实现的主要目的是减少由于数据相关而带来的大量同步通信。

2.2.1 区域关系图算法

在区域关系图[19]中,所有属于同一级连通区域的像素聚合为一个节点,因此高地不再存在。由于共享主存,并行程序与串行程序十分相似,仅仅是在同时访问同一个存储单元时需要用相应的同步原语保证互斥访问,具体步骤如下:

(1) 逐级连通区域的标记:由一个处理机对整个图像进行连通区域的标记,并将输入图像和标记后的图像分配到各个处理机上去。每个处理机分派大小近似的连续扫描行。

(2) 并行的关系图流域变换:每个处理机根据各自的图像分块建立一个局部的区域关系图,由于某些灰度级的连通区域被多个处理机共享,所以各处理机上的关系图可能相交。然后,每个处理机执行相应的泛洪过程,其中特别处理共享的节点。

(3) 还原变换:每个处理机的泛洪过程结束后,将各自的局部区域关系图再还原为源图像的形式,最后拼接成完整的输出图像。

2.2.2 基于顺序队列的地形学距离算法

算法的第一步是检测局部极小区,计算下降坡度和代价函数都是局部操作,因此易于并行化。实际上,极小区检测和完全下降化能一步完成。流域变换的步骤是基于最短路径的图像积分,每个处理机计算近似相等数量的集水盆,但这一要求与数据十分相关,因为集水盆实际上是大小不一的,因此为了获得较好的性能,负载的动态分配和映射是十分必要的。

2.2.3 基于改进的 UNION-FIND 过程[7]的地形学距离算法

这个方法相对来说更易于在共享存储的结构上进行并行化。首先第一步应用 FIFO 队列将输入图

像完全下降化,在并行情况下图像被分为大小近似的子域,每个处理机维护各自的 FIFO 队列,并在各私有子域内初始化一些种子点(如高地的下降沿像素)。然后各处理机在其 dist 图像中传播距离值,当所有处理机完成这一步后,所有 dist 图像的最小化就是希望的完全下降图像。计算这个最小值可以通过所谓的归约算子来高效地完成。完全下降化后,经一次图像扫描即可建立 DAG 图,而且在扫描过程中对每个像素而言只有其邻域像素被访问,只要计算次序确定后,像素之间没有相关性,因此这一操作很容易并行。而接下来的解析步也易于并行化,只需用每个处理机分配的子域替代相应串行算法中的域 D 就可以了。值得注意的是,这种方法中的映射内容是变化的:开始一个处理机只访问各自子域中的像素,但在解析步则将访问到其它处理域中的像素。

2.2.4 评价

文献[20]对 2.2.2 节讨论的算法进行的实验是在有 16 个处理结点、共享存储结构的系统 Cray-J932 上完成的,使用的是静态分配方法。算法的效率从 20%(对含有少量极小区的图像)变化到 60%(对含有较大数量的极小区的图像),但当图像含有极小区数量非常大时,效率反而会降低。计算下降坡度和代价函数的加速比几乎是线性的。尽管在检测极小区的过程中,地址访问冲突在多机情况下(特别是 8 个结点以上)对算法的效率有影响,但仍然能达到接近线性的加速比。如果极小区数目小于处理机数 M,则用更多的处理机不会获得性能增益,但实际上极小区的数量通常远远大于 M。另外,对 2.2.3 节讨论的方法所展开的实验工作正在进行之中。

3 结语及有待进一步研究的问题

效率低一直是困扰流域计算发展的问题,以至于 1998 年后国内在这方面的研究遭到冷遇,但随着应用需求的增长和并行化手段的发展,流域算法的并行化成为近几年国际上研究的热点。虽然已取得一些研究成果,但仍然存在亟待解决的问题:

- 很多流域算法的并行性开发得并不充分,在前期和后期都需要较复杂的变换还原计算影响性能,处理结点的利用率较低。
- 当研究同时确定分水岭和集水盆的并行算法时,发现会出现一些算法副作用,比如很宽的分水岭和孤立区[2,11]。已有一些针对性的解决方案,但它们可能会在区域生长的过程中引起结果不一致问题。
- 算法效率和可扩展性的折衷,目前仍没有提出能有效解决性能与数据相关问题的并行方案。
- 过渡分割问题的并行解决方案。
- 基于记号的并行区域生长算法和多尺度、多分辨率[21]解决方法是近期的热点。

参考文献:

[1] Beucher S, Meyer F. The Morphological Approach to Segmentation: The Watershed Transformation [C]. In E.R. Dougherty, Editor, Mathematical Morphology in Image Processing, Marcel Dekker Inc., N.Y., 1993: 433 – 481.

[2] Dobrin B.P, Viero T, et al. Fast Watershed Algorithms: Analysis and Extensions [C]. In Proceedings SPIE Nonlinear Image Processing Ⅴ, San Jose, California, 1994: 209 – 220.

[3] Meyer F. Integrals and Gradients of Images [C]. In Proceedings SPIE, Image Algebra and Morphological Image Processing Ⅲ, San Diego, California, 1992: 200 – 211.

[4] Meyer F. Integrals, Gradients and Watershed Lines [C]. Proc. Mathematical Morphology and Its Applications to Signal Processing, Barcelona, May 1993: 70 – 75.

[5] Meyer F, Beucher S. Morphological Segmentation [J]. Journal of Visual Communication and Image Representation, 1990, 1(1): 21 – 46.

[6] Meyer F. Topographic Distance and Watershed Lines [J]. Signal Processing, 1994, 38: 113 – 125.

[7] Roerdink J B, Meijster A. The Watershed Transform: Definitions, Algorithms and Strategies [J]. Fundamental Information, 2000, 41: 187 – 228.

[8] 崔屹. 图像处理与分析——数学形态学方法及应用 [M]. 北京: 科学出版社, 2000: 75 – 83.

[9] Bieniek A, Burkhardt H, et al. A Parallel Watershed Algorithm [C]. In Proc. 10th Scandinavian Conference on Image Analysis (SCIA '97), Lappeenranta, Finland, 1997: 237 – 244.

(下转 90 页)

图2 返回结果及用户操作界面

Fig.2 Result and user interface

参考文献:

[1] Foote Kenneth E, Kirvan Anthony P. WebGIS[R]. NCGIA Core Curriculum in Geographic Information Science, http://www.ncgia.ucsb.edu/giscc/units/u133/u133 f.html.

[2] Zhong Ren Pong. An Assessment of the Development of Internet GIS[C]. Proceedings of the ESRI User Conference, 1997.

[3] ESRI[R]. http://www.esri.com.

[4] 张立,龚健雅.地理空间元数据管理的研究与实现[J].武汉测绘科技大学学报,2000,25(2).

(上接76页)

[10] Vincent L, Soille P. Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations [J]. IEEE Trans. Patt. Anal. Mach. Intell. 1991, 13(6): 583 – 598.

[11] Viero T. Algorithms for Image Sequence Filtering, Coding and Image Segmentation [D]. Ph D Thesis, Tampere University of Technology, Tampere, Finland, January 1996.

[12] Olle M N, Schreiber G, Schulz Mirbach H. PIPS—A General Purpose Parallel Image Processing System [C]. In G. Kropatsch, Editor, Proceedings of 16. DAGM-Symposium" Mustererkennung", Wien, September 1994.

[13] PUL-RD Research Group. PUL-RD Prototype User Guide [R]. Parallel Computer Centre, University of Edinburgh, 1995.

[14] Moga A. Parallel Watershed Algorithms for Image Segmentation [D]. PhD Thesis, Tampere University of Technology, Finland, February, 1997.

[15] Moga A N, Viero T. Implementation of a Distributed Watershed Algorithm [R]. In Mathematical Morphology and Its Applications to Image Processing, Serra J, Soille P, Eds. Kluwer Acad. Publ., Dordrecht, 1994: 281 – 288.

[16] Moga A N, Cramariuc B, Gabbouj M. Parallel Watershed Transformation Algorithms for Image Segmentation [J]. Parallel Computing 1998, 24: 1981 – 2001.

[17] Moga A N, Gabbouj M. Parallel Image Component Labeling with Watershed Transformation [J]. IEEE Trans. Patt. Anal. Mach. Intell., 1997, 19(5): 441 – 450.

[18] Moga A N, Viero T, et al. Parallel Watershed Algorithm Based on Sequential Scanning [J]. In Proc. IEEE Workshop on Nonlinear Signal and Image Processing, Neos Marmaras, Halkidiki, Greece, 1995: 991 – 994.

[19] Meijster A, Roerdink J B. A Proposal for the Implementation of a Parallel Watershed Algorithm [R]. In Computer Analysis of Images and Patterns, Hlavac V Sara R, Eds. New York-Heidelberg-Berlin, 1995: 790 – 795.

[20] Meijster A, Roerdink J B. Computation of Watersheds Based on Parallel Graph Algorithms [R]. In Mathematical Morphology and Its Applications to Image and Signal Processing, P. Maragos, Eds., Dordrecht, 1996: 305 – 312.

[21] Moga A N. Parallel Multiresolution Image Segmentation with Watershed Transformation [C]. ACPC'99, 1999: 226 – 235.