doi:10.3969/j.issn.1672－0504.2018.05.011

OpenCL 框架下的流域径流模拟分布式调度研究

肖俊文,陈 军*,吕朝阳
(成都信息工程大学资源环境学院,四川 成都 610225)

摘要:基于分布式水文模型的径流模拟以像元为单元进行并行计算并逐小时迭代,计算量大,故将 OpenCL 引入流域径流模拟中,以提高径流模拟效率。由于普通计算机显存有限,面对流域范围大、分辨率高的情况,一般以分块计算方式解决;虽然分块计算能解决显存不足问题,但频繁的显存和内存的数据交换降低了通用计算性能。该文尝试将分布式计算与通用计算相结合,将多台计算机显存资源整合,以避免频繁的显存与内存数据交换。首先将流域数据分为单台计算机能够存储和处理的子块,然后通过网络送至服务器不同的计算机上调用 OpenCL 计算;每执行完一次径流模拟,服务器相关节点通过网络交换计算后的边缘数据。实验证明,分布式计算和通用计算相结合能快速完成大流域的径流汇流模拟,具有一定的应用价值。

关键词:OpenCL;径流模拟;通用计算;分布式计算
中图分类号:TP391;P333.1 **文献标识码**:A **文章编号**:1672－0504(2018)05－0063－05

0 引言

我国是世界上洪水灾害最频繁且严重的国家之一,暴雨洪涝灾害的快速模拟和评估具有重要意义。水文模型是暴雨风险模拟的基础,国内外已经提出了很多成熟的水文模型,如水箱模型、SWMM 和 PRMS 等集总式水文模型,也有 TOPMODEL、SHE 和 SWAT 等半分布式或分布式模型[1]。分布式水文模型将水动力学引入到实际汇流模拟中,研究流域出水口或流域各地的径流汇流[2],在水文模拟的精度上,相较于集总式水文模型,有了进一步的提升。但多数分布式水文模型计算量较大,主要在 CPU 平台上运行,其计算效率成为进一步应用的瓶颈[3]。

GPU 应用于水文数值模拟的研究,国外起步较早。吉田圭介等利用 GPU 技术对自然河流的洪水淹没过程进行了数值模拟,其计算过程考虑了河床、地表植被和河流水体边界的变化,并与 CPU 计算的结果进行了对比,获得了较好的加速效果[4];Vacondio 等使用了极限的有限体积离散技术,并利用通用计算框架 CUDA 精准且快速地模拟了洪水过程,经过严格的测试,其与实际过程相差较小,获得了较好的模拟效果[5];Dullo 等利用 CUDA 对美国密西西比州 Taum Sauk 水坝进行了溃坝洪水演进数值模拟,分别使用单个 GPU 和多个 GPU 两种方式进行了模拟,前者获得了较好的加速效果,后者目前仍处于研究阶段[6]。国内尹灵芝等基于元胞自动机(CA)演进模型,以 CUDA 为 GPU 计算框架,进行了溃坝条件下的洪水演进模拟,其模拟效率与串行的 CPU-CA 模型相比,获得了 15.6 倍的加速比[7];王金宏也进行了基于 GPU-CA 模型的洪水演进模拟,并以四川安县肖家桥堰塞湖为例,进行了溃坝洪水演进模拟和分析实验,获得了较好的加速效果[8]。

传统 GPU 通用计算的数据处理机制是将数据一次加载到显存中计算,当流域数据大于显存存储量时,读取和写入将十分困难[9]。针对此问题,有学者探讨了大数据的分块加载和写入方法,以解决显存不足问题[10];但对于大范围、高精度的流域径流汇流的迭代计算,该方法将导致频繁的显存和内存的数据交换,降低了径流模拟效率。针对上述问题,本文尝试将分布式计算与通用计算相结合,探索 OpenCL(Open Computing Language)框架下的流域径流模拟分布式计算方案。

1 基于 OpenCL 的流域径流模拟

为提高径流汇流模拟精度,采用分布式水文模型进行研究。首先依据一定的空间分辨率将流域划分为面积大小相等的网格集合;然后将径流汇流过程划分为等时间间隔的时间片,每一个时间片以网

收稿日期:2017－09－06; **修回日期**:2017－12－26
基金项目:四川省科技厅项目(2017JY0157);四川省教育厅项目(15ZB0184)
作者简介:肖俊文(1992－),男,硕士研究生,研究方向为 3S 集成与气象应用。*通讯作者 E－mail:494834920@qq.com

格为单元进行汇流计算；最后通过时间片的迭代，完成流域径流汇流的过程模拟。

考虑到分布式水文模型计算量大，CPU 计算性能低，故引入通用计算以提高计算效率。目前，GPU 通用计算框架主要有 CUDA 和 OpenCL[11]。OpenCL 是由 APPLE、Intel、AMD 等公司和团体共同发起的多设备异构计算的跨平台标准，旨在利用 GPU 强大的并行计算能力以及 CPU 对复杂步骤的运算能力，更高效地利用硬件完成大规模的（尤其是并行度高的）计算[12]。由于 OpenCL 具有良好的跨平台特性，选择 OpenCL 框架进行时间片的流域汇流演算。OpenCL 框架下的径流汇流模拟要求在一个时间片上各网格单元的径流汇流结果对其他网格单元没有影响，即网格单元径流汇流计算的可并行性。为此，汇流模拟分两步进行，即水量收支和受力计算，每一步均以网格为单元设计算法。

（1）网格单元水量收支的并行化设计。设任意网格单元为 c，时间片起始时间为 t_n，终止时间为 t_{n+1}，汇流时间为：

$$\Delta t = t_{n+1} - t_n \tag{1}$$

在时间片上，首先计算网格单元的水量收支。设 t_n 时刻的网格水深为 h_c^n，时间片上的降水量、下渗量、蒸发量、流入量和流出量分别为 R_c^n、I_c^n、E_c^n、In_c^n 和 Out_c^n，依据水量平衡原理，t_{n+1} 时刻的网格水深 W_c^{n+1} 为：

$$W_c^{n+1} = W_c^n + R_c^n + In_c^n - I_c^n - E_c^n - Out_c^n \tag{2}$$

在栅格模式下，In_c^n 和 Out_c^n 仅考虑与周围 8 邻域网格单元的水量交换。设网格单元 c 的邻域像元为 b，$\nu_{b\to c}^n$ 为网格 b 水流流入网格 c 的速度分量，ΔL 为网格 b 和网格 c 的中心距离，则流入量 In_c^n 为：

$$In_c^n = \sum_{b=0}^{7} \left(\frac{\nu_{b\to c}^n \Delta t}{\Delta L} W_b^n \right) \tag{3}$$

设 $\nu_{c\to b}^n$ 为网格 c 水流流入网格 b 的速度分量，则流出量 Out_c^n 为：

$$Out_c^n = W_c^n \sum_{b=0}^{7} \frac{\nu_{c\to b}^n \Delta t}{\Delta L} \tag{4}$$

网格单元与 8 邻域水量交换过程中，满足动量守恒定律。设 ν_c^n 和 ν_c^{n+1} 分别表示网格 c 初始水速和终止水速，则有：

$$\nu_c^{n+1} = \frac{\sum_{b=0}^{7} \frac{(\nu_{b\to c}^n)^2 \Delta t}{\Delta L} W_b^n + (h_c^n - Out_c^n)\nu_c^n}{W_c^{n+1}} \tag{5}$$

（2）网格单元受力计算的并行化设计。网格单元的水体受到多种内外力的影响，由于水压不同产生水流内部的压力梯度力；由于地表水流在坡面上或河床上流动，地表径流除受到重力作用外，还受地表摩擦力、相邻水团的作用力等。以中心网格和邻域网格水体高度差来模拟水体在重力和压力梯度作用下产生的速度增量。设网格 c 和邻域网格 b 的地形高度分别为 H_c 和 H_b，则邻域网格相对于中心网格的水体高度差 ΔW_{bc} 为：

$$\Delta W_{bc} = \begin{cases} -W_c^n & \Delta W_{bc} < -W_c^n \\ H_b + W_b^n - H_c - W_c^n & -W_c^n \leqslant \Delta W_{bc} \leqslant W_b^n \\ W_b^n & \Delta W_{bc} > W_b^n \end{cases} \tag{6}$$

其速度增量 $\Delta\nu_{bc}$ 在水平 x 和 y 方向的分量为：

$$\begin{cases} \Delta\nu_{bc}^x = -\Delta W_{bc} \times K \times \mathrm{d}x / \Delta L \\ \Delta\nu_{bc}^y = -\Delta W_{bc} \times K \times \mathrm{d}y / \Delta L \end{cases} \tag{7}$$

式中：K 为综合压力梯度、重力及相邻水团作用等因素的水力加速度常数；$\mathrm{d}x$ 和 $\mathrm{d}y$ 分别为网格 b 和网格 c 在 x 和 y 方向的坐标之差。

地表水流由于受到河床摩擦力的影响，水流速度发生衰减。设在单元网格上摩擦力作用的上界深度为 $W_{\max}$，下界深度为 $W_{\min}$，摩擦比例常数为 σ，由具体流域特征确定，则衰减系数 ε 近似定义为：

$$\varepsilon = \begin{cases} \sigma \times \left(\frac{W_c^n - W_{\min}}{W_{\max} - W_{\min}} \right)^2 & W_c^n \leqslant W_{\min} \\ \sigma & W_c^n > W_{\max} \end{cases} \tag{8}$$

2 流域径流模拟的分布式调度思路

2.1 单台计算机流域径流模拟缺陷

为实现 OpenCL 径流汇流计算，水深和流速设计为纹理数组，纹理的大小由流域范围和空间分辨率确定。由于显存的限制，OpenCL 计算所接受的纹理尺寸有限，当流域栅格尺寸超出显存所能表示的范围，将不能完成径流汇流模拟。在单台计算机上，为解决显存限制问题，将纹理按相等大小划分为一定数量的小纹理块，每次仅从内存读入一个纹理块的数据。径流汇流计算完成后，将分块数据再读回内存。频繁的显存和内存的数据交换降低了计算性能。

以相同流域的径流模拟（测试数据未超过单台计算机显存）为例，分别测试 CPU、GPU 分块策略和 GPU 不分块策略下模拟 100 个时间片的计算性能，其中，GPU 分块策略将流域分为大小相等的 4 个子块。由测试结果（表 1）可见，GPU 分块策略相对于 CPU 计算，其性能提升了 10 倍以上，但与不分块的 GPU 计算相比，其性能仅为后者的 1/24 左右。因此，要解决大流域、高分辨率的径流汇流实时模拟，必须寻求新的实现方式。

表 1　相同流域汇流模型 CPU 和 GPU 模拟耗时对比
Table 1　The comparison of simulated time between CPU and GPU computing for the same watershed

迭代时间片	CPU	GPU 分块		GPU 不分块	
	累计耗时(s)	累计耗时(s)	加速比	累计耗时(s)	加速比
100	2 396.24	221.1	10.84	9.07	264.19

2.2　流域径流模拟的分布式调度思路

为解决单台计算机模拟大流域径流汇流时频繁的显存和内存的数据交换问题，提出通用计算与分布式计算相结合的思路[13]。首先将流域数据分为单台计算机能够存储和处理的子块，然后通过网络送至服务器不同的计算机上进行通用计算。为减少服务器之间的大块数据交换，服务器接收到子块处理任务后，一直负责对该数据块的径流模拟作迭代处理。在径流模型中，每一个网格水流和水深计算均需要周围 8 邻域网格参与计算。为保证分块计算结果的正确性，子块在每一次迭代后，与相邻子块交换边缘数据，以保持相邻数据块在任意时间片上的数据一致性。流域径流模拟分布式调度思路如图 1 所示。

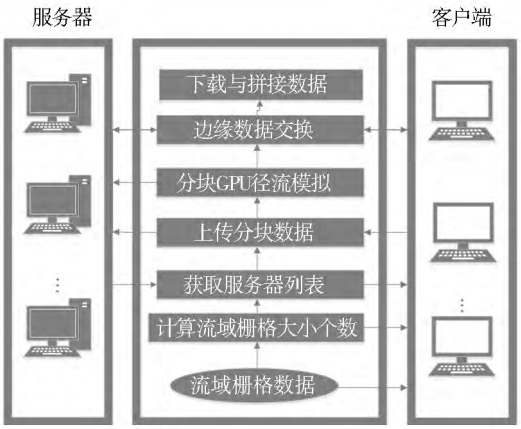

图 1　流域径流模拟分布式调度思路
Fig. 1　The method of distributed scheduling of runoff simulation

2.3　分布式调度框架设计

分布式调度框架(图 2)分为客户端和服务器端，服务器端由主服务器、节点服务器和分布式文件系统三部分组成。主服务器负责接收客户端的命令请求，以分布式文件系统的形式存储、管理流域数据[14]，节点服务器承担子块的径流汇流计算。在分布式计算框架下，完整的大数据径流模拟流程为：首先，客户端将要计算的流域地形数据上传到分布式文件系统中，然后向主服务器发送径流模拟任务请求；主服务器返回所有可用节点服务器的信息后，客户端筛选出可用节点，并将任务提交给节点服务器；节点服务器根据客户端提交的任务计算某一子块的径流汇流；当节点计算完成后，与相关节点服务器交换边缘数据；所有节点服务器边缘交换完成后，继续下一个时间片的迭代计算，直至所有时间片计算完成；最后，客户端将节点计算结果下载后拼合得到最终汇流结果。

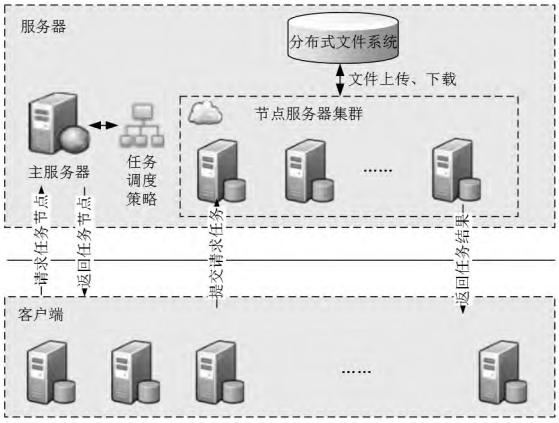

图 2　流域径流模拟的分布式框架
Fig. 2　The distributed framework of runoff simulation

3　分布式调度的边缘数据交换

3.1　边缘数据交换的必要性

(1)分布式计算的分块方案。在流域汇流计算时，网格单元为基本的并行计算单元，其水深和流速预测值由其与相邻 8 邻域网格的当前水深和流速计算得到。在 OpenCL 中，纹理的一个像元为汇流计算的网格单元[15]。为保证流域分块计算结果的正确性，每个分块纹理需在上、下、左、右 4 边多读取一行或一列网格像元。图 3 为分块内部像元和边缘像元示意图。假定分块大小为 5 行 5 列，则左上角第一块实际大小为 6 行 6 列。其中，右侧和下侧均从地形、水深或水速栅格文件中多读取一层边缘数据。在分块径流模拟时，实际计算的像元为内部像元，边缘像元仅用于计算其相邻内部像元的水深或水速。

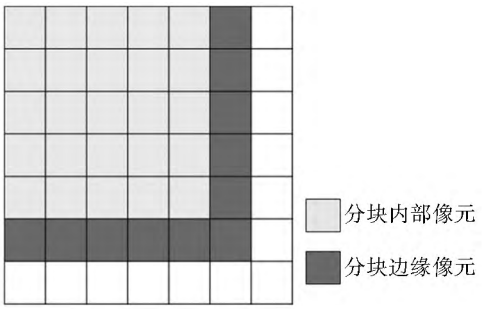

图 3　分块内部像元与分块边缘像元
Fig. 3　The internal pixels and marginal pixels in blocks

(2)边缘数据交换的必要性。如果仅需对流域做一次全图计算，则直接读取节点的内部像元并在客户端拼接即可得到完整的流域模拟结果。但径流

汇流是一个在时间上不断迭代的过程，上一个时间片的水深和水速成为下一时间片水深和水速生成的依据。在迭代计算情况下，每一个时间片的分块计算完成后必须先交换边缘数据再执行下一时间片的汇流计算，否则不能得到正确的结果。这是因为，每个分块只负责内部像元的计算，其边缘像元由相邻分块计算得到。

3.2 边缘数据更新策略

边缘数据更新是指在每个时间片的径流汇流计算完成后，每一分块从相邻块读取更新后的边缘数据。设分块编码为(x,y)，则该分块需要从相邻分块$(x-1,y)$、$(x,y-1)$、$(x+1,y)$、$(x,y+1)$、$(x-1,y-1)$、$(x+1,y-1)$、$(x-1,y+1)$、$(x+1,y+1)$读取对应的边缘数据。边缘数据更新策略如表2所示，如果内部分块在4个方向上同时存在边缘数据，则需从相邻的8个分块中读取边缘并更新。其中，上、下、左、右相邻分块读取一行或一列，其余相邻分块则需读取一个像元。

表2 分块(x,y)的边缘数据更新策略
Table 2 The updated strategy of marginal data of (x,y)

边缘位置	读取的分块数据源	读取的分块数据
左边缘	$(x-1,y)$	相邻块倒数第2列
上边缘	$(x,y-1)$	相邻块倒数第2行
右边缘	$(x+1,y)$	相邻块第2列
下边缘	$(x,y+1)$	相邻块第2行
左上角	$(x-1,y-1)$	相邻块倒数第2列、倒数第2行
右上角	$(x+1,y-1)$	相邻块第2列、倒数第2行
左下角	$(x-1,y+1)$	相邻块倒数第2列、第2行
右下角	$(x+1,y+1)$	相邻块第2列、第2行

虽然每个时间片均要求交换边缘数据，但相对于单台计算机GPU分块策略交换全部分块数据而言，其数据交换量非常小。以分块大小4 096×4 096为例，并假定4个边缘均存在边缘更新，其数据交换量仅为分块全部数据量的1/1 024。理论上，如果局域网网速较快，分布式径流模拟效率相对于单台计算机GPU分块，具有更好的性能。

3.3 分布式框架下的边缘数据更新

分布式计算旨在将大流域划分成多块数据，使计算机集群分而算之。通常情况下，客户端向服务器发送径流模拟请求任务，服务器计算完成后将结果返回客户端汇总。在分布式径流模拟迭代计算场景中，边缘数据更新有两个关键环节：1)下一个时间片径流模拟计算时，边缘数据需要从相邻块读取，而相邻块的计算在不同的节点服务器上，因而相邻块之间的边缘交换转变为不同计算节点之间的数据交换；2)节点的计算在显卡中完成，节点内部显存和内存的边缘数据交换是边缘数据更新的另一个环节。

3.3.1 计算节点的边缘数据交换 计算节点的边缘数据交换通过网络数据传输完成。为深入研究节点边缘数据交换的最佳方式，尝试设计两种不同的方案：1)以节点为中心的数据交换方案。在该方案中，每个节点不仅记录分块位置，还建立了相邻分块的网络连接。节点完成单时间片径流汇流后，按表2所示分别读取各边缘，并将边缘数据发送给相邻分块的计算节点；每个节点接收到所有边缘的更新数据后，进入下一个时间片的汇流计算。以节点为中心的数据交换方案的优点在于，每个节点单时间片计算网络流量与分块大小有关，与流域大小无关，缺点是该方式的网络交换次数较多。2)以客户端为中心的数据交换方案。该方案是将客户端作为边缘数据聚合中心，节点完成径流汇流后，直接读取4个边缘的行列数据，一次性传输给客户端；客户端接收到所有节点的边缘数据后，将各节点所需的新边缘数据一次性发送给节点服务器。以客户端为中心的数据交换方案在一个时间片的径流汇流过程中，需要的网络通讯总次数等于分块数的2倍，相对于第一种方案减少了总的网络通讯次数，但该方案的客户端网络传输压力较大。

3.3.2 显存和内存的边缘数据读取和更新 数据节点通过内存的方式在网络上交换边缘数据。当节点的分块数据计算完成后，将边缘数据从显存读回内存，并通过网络传输到相邻计算节点或客户端；当节点接收到新边缘数据后，数据存储在内存，将其更新到显存后才能进行下一个时间片的径流汇流。

4 性能测试

为验证OpenCL框架下流域径流模拟分布式调度方法的效率，在局域网环境下(5 M/s)进行测试。测试时选择8台配置相近的计算机，其中使用INTEL的酷睿i5系列CPU和NVIDIA的GTX 950系列GPU。选取两种规模的流域，其大小分别为3 225×5 325(小流域)、6 450×10 646(大流域)；在分块策略中，2节点和4节点分块数为4，8节点分块数为8。观测迭代次数分别为100、200、400情况下的累积耗时，测试结果如表3所示。

从表3可见，分布式/GPU方法在节点数量为4时计算效率最高，是单机分块计算效率的3倍以上。这是因为，当节点数量不足时，节点同时执行多个分

块的径流汇流，未能充分体现任务的并行性；当节点数量过多，分块划分更细时，需要交换的边缘数据量也在不断增大，从而导致性能下降。因此，分布式径流汇流最佳分块策略是将流域划分为单台计算机能接受的最大分块尺寸并交于相等节点计算，每一个节点只负责其中一个分块的计算。通过表 3 的两种分布式调度方法，发现服务器交互方法的效率均高于客户端交互，可见，数据交换次数相比交换数据量大小，对径流模拟效率影响更小，服务器交互方法更适合分布式框架下的径流汇流模拟。

表 3　不同方式下 OpenCL 流域径流模拟迭代执行时间对比

Table 3　The comparison of iterative execution time of runoff simulation in different ways

时间片数	流域规模	单台计算机(s)		分布式/GPU 客户端交互(s)			分布式/GPU 服务器交互(s)		
		不分块	分块	2 节点	4 节点	8 节点	2 节点	4 节点	8 节点
100	小	9.07	221	59	60	60	58	55	56
	大	\	445	98	87	87	95	83	83
200	小	18.1	423	122	91	93	118	96	97
	大	\	830	205	164	168	203	160	165
400	小	36.5	\	250	188	191	225	180	182
	大	\	\	405	365	372	395	333	341

注：\表示无法计算。

从流域规模上分析，分布式调度不适合小范围流域径流模拟。主要原因在于，小范围径流模拟在单台计算机下足以完成，而在分布式框架下，额外增加了网络通信、网络传输的时间。当计算量超过单台计算机承受范围时，分布式/GPU 方法不仅能计算大范围流域径流模拟，而且随着数据量的增大，计算效率还有一定的提升。

综上所述，OpenCL 框架下径流模拟分布式调度方法很好地解决了径流汇流模型的计算量大与 CPU 性能瓶颈间的矛盾，也突破了 GPU 分块计算中频繁的显存和内存的数据交换的瓶颈，提高了径流汇流模拟的效率。

5　结论

OpenCL 是一套对高性能处理芯片构建并行程序的强大工具，它除了具有可移植性，还在向量处理和并行编程方面有着天然的优势，有助于软件开发人员为高性能计算服务器、桌面计算系统、手持设备编写高效代码。本文提出的 OpenCL 框架下径流模拟分布式调度方法，充分利用现有的单台计算机 GPU 资源，将其结合为计算机集群，实现了单台计算机无法完成的大范围、高精度的流域径流模拟。实验结果证明，分布式调度方法不仅能完成大范围流域径流模拟，而且模拟效率相对于传统的 CPU 径流模拟和单机 GPU 分块模拟有一定的提升。

参考文献：

[1] 徐宗学. 水文模型[M]. 北京：科学出版社，2009. 9－10.

[2] 徐宗学，程磊. 分布式水文模型研究与应用进展[J]. 水利学报，2010，41(9)：1009－1017.

[3] 郑贵元，李建贵，孙伟，等. 分布式水文模型研究进展[J]. 安徽农学通报，2017，23(10)：27－33.

[4] 吉田圭介，田中龍二，前野詩朗. GPUによる分流を含む洪水流計算の高速化[A]. 土木学会水工学委員会. 水工学論文集[C]. 2016. 59.

[5] VACONDIO R，PALÙA D，MIGNOSA P. GPU-enhanced finite volume shallow water solver for fast flood simulations[J]. Environmental Modelling & Software，2014，57：60－75.

[6] DULLO T，KALYANAPU A，GHAFOOR S，et al. Computational performance of a two-dimensional flood model in single and multiple GPU frameworks[A]. EGU General Assembly 2015[C]. 2015. 14739.

[7] 尹灵芝，朱军，王金宏，等. GPU-CA 模型下的溃坝洪水演进实时模拟与分析[J]. 武汉大学学报(信息科学版)，2015，40(8)：1123－1129.

[8] 王金宏. 基于 GPU-CA 模型的溃坝洪水实时模拟与分析[D]. 成都：西南交通大学，2014.

[9] SANDERS J，KANDROT E. CUDA by Example：An Introduction to General-Purpose GPU Programming[M]. US：Addison-Wesley Professional，2010.

[10] 靳华中，柯敏毅. 一种基于集群的栅格图像分割算法的研究[J]. 湖北工业大学学报，2006，21(4)：104－107.

[11] DU P，WEBER R，LUSZCZEK P，et al. From CUDA to OpenCL：Towards a performance-portable solution for multi-platform GPU programming[J]. Parallel Computing，2012，38(8)：391－407.

[12] MUNSHI A，GASTER B，MATTSON T G，et al. OpenCL Programming Guide[M]. US：Addison-Wesley Professional，2012.

[13] 刘永和，冯锦明，徐文鹏. 分布式水文模型的 GPU 并行化及快速模拟技术[J]. 水文，2015，60(4)：20－26.

[14] 阴桂梅，刘耀军，郭广行. 云计算在分布式地理信息系统中的应用[J]. 山西电子技术，2010，38(6)：28－29.

[15] DO H T，LIMET S，MELIN E. Parallel computing flow accumulation in large digital elevation models[J]. Procedia Computer Science，2011，4(4)：2277－2286.

(下转第 91 页)

[25] 李强，严金明. 武汉市城市边缘区村镇发展模式与土地利用政策研究[J]. 地理与地理信息科学，2012，28(2)：76－79.

[26] 曾赞荣，王连生. “绿隔”政策实施下北京市城乡结合部土地利用问题及开发模式研究[J]. 城市发展研究，2014，21(7)：24－28.

Study on the Implementation Path and Model of Construction Land Decrement Development in Beijing

LI Qiang，WANG Zi－xin，WANG Hong－yue，SUN Qian－qing

(*School of Urban Economics and Public Administration，Capital University of Economics and Business，Beijing 100070，China*)

Abstract：The development strategy of construction land decrement is a new mechanism of producing land resources to ensure the capital city Beijing develop rationally. The scientific design of the implementation path and the construction of the model are the premises to guarantee the successful reduction of construction land. Based on the policy requirements of reducing construction land and the realistic demands of Beijing social and economic development，the decrement development trends，the existing implementation model and the related practice of construction land reduction management were analyzed in this paper. The paper presents a framework of the implementation path to address the key issues of " how much，where，when，how，and how to use land" in the construction land decrement development，to propose the main contents and implementation requirements of each stage，and to develop five types of construction land decrement models：town planning as a whole，land purchase and storage，cooperation between village and enterprises，comprehensive land renovation，and relation among people，land and capital. Furthermore，the implementation mechanism of construction land decrement development was established by endogenous dynamic cultivation，land use index coordination and development right compensation，and so on. This study aims to provide support for the formulation and policy of construction land decrement development in Beijing，so as to promote its scientific and orderly implementation.

Key words：construction land；decrement development；path；model；Beijing

(**上接第 67 页**)

Study on Distributed Scheduling of Runoff Simulation under OpenCL Framework

XIAO Jun－wen，CHEN Jun，LV Zhao－yang

(*Institute of Resources and Environment，Chengdu University of Information Technology，Chengdu 610225，China*)

Abstract：The runoff simulation based on distributed hydrological model is computed parallelly based on pixel units and iterated hourly，the amount of calculation is very huge，so OpenCL is introduced into runoff simulation to improve the efficiency. Because the video memory of common computer is limited，in the face of large watershed area and high resolution，the general solution is the block calculation. Although the block calculation can solve the problem of insufficient video memory，the frequent data exchange of video memory and internal memory reduces the performance of general-purpose computing. This paper attempts to combine distributed computing with general-purpose computing，combining the video memory of multiple computers to avoid frequent data exchange between video memory and internal memory. First，watershed data is divided into sub blocks that can be stored and processed by a single computer，then they are sent to different computers to call OpenCL to compute by network. After each execution of runoff simulation，the related nodes of server exchange will calculate marginal data by network. The experiment proves that the combination of distributed computing and general-purpose computing can complete the runoff simulation of large watershed quickly. It has definite application value.

Key words：OpenCL；runoff simulation；general-purpose computing；distributed computing