

网格作业完工时间与作业分割粒度的关系

李荣胜¹,赵文峰²,徐惠民¹

(1. 北京邮电大学 信息与通信工程学院,北京 100876; 2. 北京邮电大学 网络与交换技术国家重点实验室,北京 100876)

(lrsbnu@sohu.com)

摘要:对可分割的计算密集型大型作业在并行且不间断运行情况下的完工时间与作业分割粒度之间的关系进行研究。首先分析了子作业之间无通信和有通信两种情况下可分割计算密集型大型作业的完工时间和分割粒度的关系,然后对可分割计算密集型大型作业在专用网格资源上的完工时间与分割粒度的关系进行仿真。仿真结果显示,大型作业的完工时间随着分割粒度的增大先减小后增大;当单个子作业的计算时间和通信时间之比增大时,作业的分割粒度可以更细,作业完工时间的最小值减小。因此完工时间最优的作业分割粒度不能过粗或过细。

关键词:完工时间;分割粒度;计算密集型;作业调度;网格计算

中图分类号: TP393; TP316.4 **文献标志码:** A

Relationship between makespan of grid job and granularity of job partitioning

LI Rong-sheng¹, ZHAO Wen-feng², XU Hui-min¹

(1. School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China;

2. State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China)

Abstract: There have been many divisible compute-intensive grand-challenge jobs running on volunteer grid. The relationship between makespans of such grid jobs and granularities of jobs' partitioning was studied. Firstly, the relationship between makespan and partitioning granularity of compute-intensive jobs with and without communications between sub-jobs were analyzed theoretically. Then, the relationship between makespan and partitioning granularity of a job with and without communications between sub-jobs running on dedicated grid resources in parallel mode were simulated. The simulation results show that grand-challenge job's makespan decreases at first and then increases when granularity increases. Granularity can be more grain, and the best makespan will decrease when the ratio of computation time to the communication time of a sub-job increases. To optimize job's makespan, the job's partitioning granularity should not be too coarse or too fine.

Key words: makespan; partitioning granularity; compute-intensive; job scheduling; grid computing

0 引言

计算密集型大型作业的完工时间的优化是研究的焦点之一。迄今为止,已有许多计算密集型大型科学研究项目运行在以个人计算机为主的网格平台上^[1]。这些网格项目包括英国牛津大学开展的预测21世纪气候实验的Climateprediction.net、IBM公司开展的医药、生物、环境等方面研究的World Community Grid、欧洲原子能研究组织CERN研究高速粒子束在大型强子对撞机轨道中运行稳定性的LHC@home、美国威斯康星——密尔沃基大学寻找引力波存在的证据的Einstein@home、美国加州大学伯克利分校搜寻地外文明的SETI@home、美国华盛顿大学开展的蛋白质结构预测、蛋白质—蛋白质对接和蛋白质设计等研究的Rosetta@home等。这些项目由大学、研究实验室、公司、科学家、国际研究组织等发起,研究涉及天文学、物理学、化学、生物学、医学、认知科学、地球与环境科学、数学等领域。

这些网格项目的资源由一个项目服务器和成千上万的个人计算机组成。项目服务器把计算作业分割成若干个工作单元(Work Unit, WU),这些工作单元被分配给世界各地的个人计算机处理,处理结果上传到服务器,服务器对处理结果进行

合并处理^[2]。

个人计算机的可用性具有易变的特点,即只在空闲(没有鼠标点击或键盘按键事件)时或是个人设定的时间段内运行。所以,项目服务器端的作业调度系统无法预知和控制作业的完工时间。

随着欧洲原子能研究组织CERN的大型强子对撞机LHC的7万亿电子伏特的两个束流对撞实验的成功^[3],WLCG(the Worldwide LHC Computing Grid)网格将要为分析这些实验数据进行计算。LHC每年将产生15 PB(15×10^6 GB)数据,这些数据由位于CERN总部的WLCG的计算中心Tier-0整理备份后,分发给分别位于加拿大、法国、德国、意大利、荷兰、北欧、西班牙、台北、英国、美国(有两个计算中心参与)的11个Tier-1计算中心,这11个Tier-1计算中心将为34个国家的130个Tier-2计算中心提供数据,研究者(可称为Tier-3)将与Tier-2计算中心连接进行数据分析研究^[4-5]。

本文研究可分割的计算密集型作业在计算中心、研究组织、大学等专用网格资源上并行且不间断执行时,作业完工时间与作业分割粒度之间的关系。首先描述可分割的计算密集型网格作业和网格环境,然后对作业完工时间与分割粒度的关系进行理论分析,最后对作业完工时间和分割粒度的关系

收稿日期:2010-07-19;修回日期:2010-09-17。

基金项目:国家973计划项目(2007CB307103);贵州省重大科技专项计划项目(黔科合重大专项字[2007]6017)。

作者简介:李荣胜(1975-),男,广西大化人,博士研究生,主要研究方向:网格作业调度; 赵文峰(1980-),男,山西夏县人,博士研究生,主要研究方向:语义Web、Web服务; 徐惠民(1941-),男,上海人,教授,博士生导师,主要研究方向:网格计算。

进行仿真。

1 可分割的计算密集型网络作业及网络环境描述

1.1 作业的分割

作业的分割是一个难题。哪些作业能分割以及如何分割,分割后的子作业之间在执行过程中有无通信、有通信时如何通信等,这些问题可能需要作业所属领域的专家和计算机领域的专家合作才能解决。

以 SETI@home 项目为例来看看作业是如何被分割的^[6-7]。SETI@home 项目使用志愿者的计算机来分析来自位于波多黎哥的 Arecibo 天文望远镜采集的无线电信号,以寻找地外文明存在的证据。SETI@home 项目要分析的数据是以 1420 MHz 为中心,从 1418.75 MHz 到 1421.25 MHz 波段内 2.5 MHz 的数据。2.5 MHz 的波段被等分为 256 子波段,每一个子波段是 9766 Hz,约为 10 kHz,采样频率为 20 Kbps (奈奎斯特频率)。把包含 107s 上述每个子波段的数据作为一个工作单元 WU 分配给一台个人计算机处理,每一个 WU 的大小约为 0.25 MB,加上封装信息,约为 340 KB。每一个 WU 执行结果的输出数据量为 1 KB。每一个 WU 在一台 Pentium II 500 MHz 的计算机上不间断执行约需要 10 h。

已经有很多类似 SETI@home 这样可分割的大型作业正在由个人计算机组成的网络环境中运行^[1],本文研究可分割的大型作业的完工时间与分割粒度的关系,而不考虑作业如何分割。

1.2 可分割的计算密集型网络作业描述

设有可分割的计算密集型网络作业 J ,用五元组表示为:

$$J = \langle id, a, w, in, out \rangle \quad (1)$$

其中: id 表示作业 J 的编号, a 表示网格系统接收 J 的时间, $w (> 0)$ 表示 J 的计算量, in 表示 J 的输入数据量, out 表示 J 的输出数据量。

作业 J 可分割为 n 个子作业,即:

$$J = \{J_1, J_2, \dots, J_j, \dots, J_{n-1}, J_n\} \quad (2)$$

其中 n 称为作业 J 的分割粒度。子作业 J_j 表示为:

$$J_j = \langle id_j, a_j, w_j, in_j, out_j \rangle \quad (3)$$

其中: id_j 表示子作业 J_j 的编号, a_j 表示 J_j 被分割生成时间, $w_j (> 0)$ 表示 J_j 的计算量, in_j 表示 J_j 的输入数据量, out_j 表示 J_j 的输出数据量。子作业 J_j 的参数和作业 J 的参数有式 (4)~(7) 的关系:

$$a_j > a \quad (4)$$

$$\sum_{j=1}^n w_j = w \quad (5)$$

因为输入输出的数据有封装开销,所以有:

$$\sum_{j=1}^n in_j > in \quad (6)$$

$$\sum_{j=1}^n out_j > out \quad (7)$$

1.3 网络环境

网络环境由多个资源组成,每个资源包含一到多台机器,每台机器包含若干个相同的 CPU。假设资源的存储能力以及内存足够大,不影响作业的执行,则网络环境可以描述为:

$$\begin{cases} Grid = \{Grid_resource_\xi \mid \xi \in \mathbf{N}_+\} \\ Grid_resource_\xi = \{Computer_\eta, \\ \quad bandwidth_\eta \mid \eta \in \mathbf{N}_+\} \\ Computer_\eta = \{CPU_\zeta \mid \zeta \in \mathbf{N}_+\} \end{cases} \quad (8)$$

其中 \mathbf{N}_+ 表示正整数集。

本文研究由式 (1)~(3) 描述的作业 J 在式 (8) 所描述的网络环境中并行执行时完工时间 $Makespan_J$ 与作业分割粒度 n 之间的关系。

2 作业完工时间与分割粒度关系的分析

2.1 子作业之间无通信时的作业完工时间

各个子作业之间相互独立,在计算过程中不需要相互通信时,作业 J 的执行时间中包括网格调度系统把作业 J 分割成子作业消耗的时间 C^g 、各个子作业的输入数据下载的开始时间 t_j^{in} 、各个子作业计算结果上传的结束时间 t_j^{out} 、网格调度系统把各个子作业计算结果合并处理消耗的时间 C^m 及子作业的计算时间。所有子作业的输入文件下载、计算、计算结果的上传是并行发生的。

令 t^{begin} 为子作业输入文件下载开始时间的最小值,即:

$$t^{begin} = \min(t_j^{in} \mid 1 \leq j \leq n) \quad (9)$$

令 t^{end} 为子作业计算结果上传结束时间的最大值,即:

$$t^{end} = \max(t_j^{out} \mid 1 \leq j \leq n) \quad (10)$$

因为任意子作业计算的开始时间晚于 t^{begin} ,结束时间早于 t^{end} ,所以所有子作业从输入文件下载开始到计算结果上传结束总共消耗的时间 C_{exe}^{com} 为:

$$C_{exe}^{com} = t^{end} - t^{begin} \quad (11)$$

综上所述,作业 J 的完工时间 $Makespan_J$ 为:

$$Makespan_J = C^g + C_{exe}^{com} + C^m \quad (12)$$

2.2 子作业之间有通信时的作业完工时间

当子作业在计算过程中需要相互通信时,作业 J 的执行时间中除了包括 C^g 、 t_j^{in} 、 t_j^{out} 、 C^m 、子作业的计算时间外,还有子作业计算过程中相互通信的时间。设相邻的子作业在计算过程中需要异步通信,即 $\langle J_1, J_2 \rangle$ 、 $\langle J_j, J_{j+1} \rangle$ ($n \geq 4$, $2 \leq j \leq n-2$)、 $\langle J_{n-1}, J_n \rangle$ 在计算过程中需要互相通信,且子作业 J_j 传输给与其相邻的子作业的数据量 $comm_j = out_j$,即, J_1 向 J_2 传送数据量 out_1 , J_n 向 J_{n-1} 传送数据量 out_n , J_j 向 J_{j-1} 和 J_{j+1} 传送数据量 out_j 。则当 $n \geq 2$ 时,上述式 (3) 描述的子作业变为式 (13):

$$\begin{cases} J_1 = \langle id_1, a_1, w_1, in_1 + out_2, out_1 \times 2 \rangle \\ J_j = \langle id_j, a_j, w_j, in_j + out_{j-1} + out_{j+1}, out_j \times 3 \rangle, \\ \quad n \geq 3, 2 \leq j \leq n-1 \\ J_n = \langle id_n, a_n, w_n, in_n + out_{n-1}, out_n \times 2 \rangle \end{cases} \quad (13)$$

此时,作业 J 的完工时间 $Makespan_J$ 的表达式还是式 (12),但等式右边第二项 C_{exe}^{com} 的值已发生变化。

2.3 作业完工时间与分割粒度的关系

作业 J 的计算量 w 可分为串行分量 w_1 和并行分量 w_n 两部分,即 $w = w_1 + w_n$ 。此外,把作业 J 分割为 n 个子作业在网格资源上并行执行时还有分割生成子作业、子作业结果合并、通信等额外开销,把这些额外开销记为 w_o 。作业 J 在单个

处理器上的串行执行时间记为 T_1 , 在 n 个处理器上的并行执行时间记为 T_n , 加速比记为 S_n , 则有式(14)、(15)^[8]。

$$S_n = \frac{T_1}{T_n} \quad (14)$$

关于加速比 S_n , 有 Amdahl 定律:

$$S_n = \frac{w_1 + w_n}{w_1 + \frac{w_n}{n} + w_o} = \frac{w}{w_1 + \frac{w_n}{n} + w_o} \quad (15)$$

由式(14)、(15)得:

$$T_n = \frac{w_1 + \frac{w_n}{n} + w_o}{w} \times T_1 \quad (16)$$

其中, T_n 即为作业 J 的完工时间 $Makespan_J$ 。对于一个指定的作业 J , w 和 w_1 是一定的; 对于给定的处理器速度, T_1 也是一定的。因此, T_n 的值取决于 $w_n/n + w_o$ 的值。当分割粒度 n 增大时, w_n/n 的值减小, w_o 的值增大; 当 $w_n/n + w_o$ 取得最小值时, 完工时间 T_n 最优。

3 作业完工时间与作业分割粒度关系的仿真

3.1 仿真设计

在下文中, 数据量的单位为 KB, 带宽的单位为 KBps, 作业计算量的单位为 MI(百万指令), CPU 速度单位为 MIPS, 时间单位为 s。

1) 关于作业的假设。子作业计算结果的合并时间与作业分割时间相等, 即, $C^m = C^s$; 每个工作单元 WU 输入的数据量为 256, 输出的数据量为 8; 每个子作业(包含一或多个 WU)输入数据量的封装开销为 64, 输出数据量的封装开销为 192^[9-10]。

2) 关于资源的假设。在仿真过程中资源不发生故障; 网格中有 $m(m \geq n)$ 个计算资源, 每个计算资源只有一个 CPU, 且所有 CPU 的处理速度相等; m 个计算资源的带宽相等, 为 BW_c ; 网格调度系统的带宽为 BW_s , 且 $BW_s > BW_c$, 仿真中设 $BW_s = 10^6$ 。

设可分割计算密集型作业 J 包含 1000 个工作单元 WU, 分割粒度 n 分别为 1、5、10、20、50、100、200、500、1000。每个 WU 在单个资源上计算完成的时间为 T_{WU} 。此处, 作业 J 并不是无限可分的, 每一个子作业必须包含整数个 WU。

仿真使用了 GridSim^[11] 和 ALEA^[12] 软件包。仿真时, 调度 n 个子作业在 n 个资源上并行计算。

3.2 仿真结果

当 $T_{WU} = 10$ 、 $BW_c = 1000$ 时, 仿真结果如图 1 所示。

当 $T_{WU} = 10$ 、 $BW_c = 10000$ 时, 仿真结果如图 2 所示。

当 $T_{WU} = 100$ 、 $BW_c = 1000$ 时, 仿真结果如图 3 所示。

由图 1~图 3 可知, 作业的完工时间随着作业分割粒度的增大先减小后增大, 在某一分割粒度处有最小值。

由图 1 可知, 在子作业之间无通信情况下, 作业分割粒度为 100 左右时, 完工时间最小; 在子作业之间有通信情况下, 作业分割粒度为 50 左右时, 作业完工时间最小。

由图 2 可知, 分割粒度在 200 左右时完工时间最小; 子作业之间有通信和无通信两种情况下的作业完工时间差别很小。与图 1 相比, 图 2 中的资源带宽变大了, 即通信时间变小了。

图 3 中, 在子作业之间无通信情况下, 作业分割粒度为 500 左右时, 完工时间最小; 在子作业之间有通信情况下, 作业分割粒度为 200 左右时, 作业完工时间最小。与图 1 相比, 图 3 中的每个子作业在单个资源上计算完成的时间变大了。

对比图 2 和图 1、图 3 和图 1 可见, 当单个子作业的计算时间和通信时间之比增大后, 子作业之间有通信时的作业完工时间更接近于子作业之间无通信时的作业完工时间, 作业完工时间取得最小值时对应的作业分割粒度更细, 作业完工时间的最小值减小。

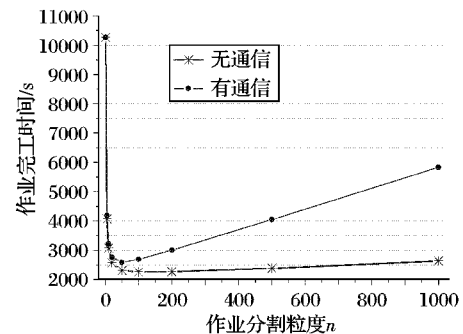


图 1 $T_{WU} = 10$ 、 $BW_c = 1000$ 时完工时间和分割粒度的关系

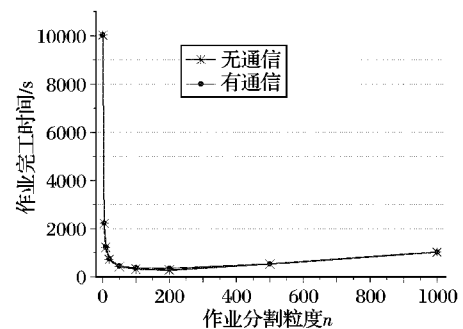


图 2 $T_{WU} = 10$ 、 $BW_c = 10000$ 时完工时间和分割粒度的关系

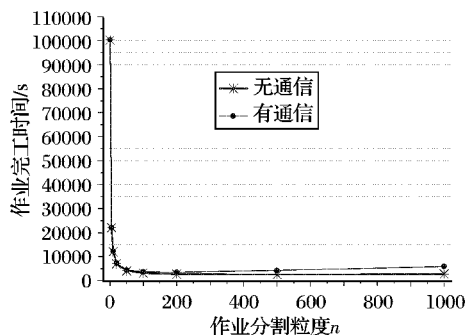


图 3 $T_{WU} = 100$ 、 $BW_c = 1000$ 时完工时间和分割粒度的关系

4 结语

本文研究了可分割计算密集型作业的完工时间与作业分割粒度之间的关系, 对子作业之间无通信和有通信两种情况进行了分析。当各个子作业在相同的资源上不间断并行执行时, 仿真结果表明: 无论子作业之间是否相互通信, 作业的完工时间随着分割粒度的增大先减小后增大, 要使作业的完工时间最短, 作业的分割粒度不能过粗或过细; 当每个子作业在单个资源上计算完成的时间以及资源的带宽一定时, 子作业之间无通信情况下作业完工时间取得最小值的分割粒度比子作业之间有通信情况下作业完工时间取得最小值的分割粒度细, 无通信时完工时间的最小值比有通信时的小; 单个子作业

(下转第 547 页)

度。本文研究旨在为进一步的辨识算法设计提供参考,以提高辨识精度。

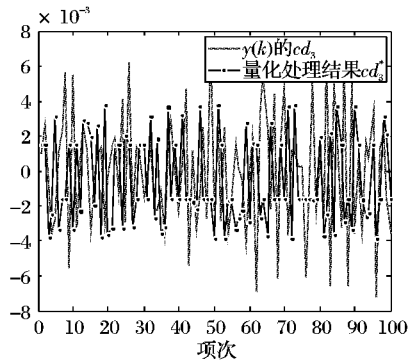


图13 cd_3 处理效果对比($SNR = 0.2$)

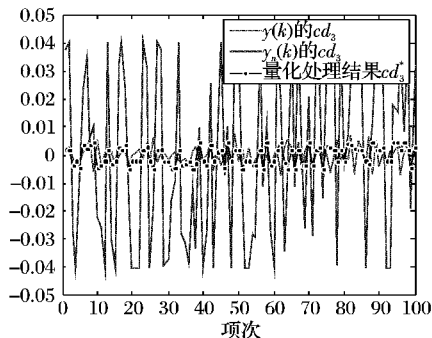


图14 cd_3 及其处理结果($SNR = 0.3$)

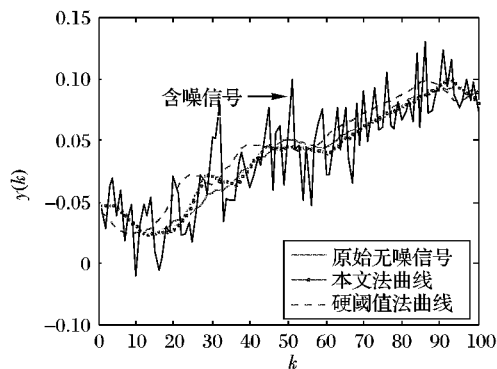


图15 输出信号降噪效果对比($SNR = 0.15$)

在小波分析理论与应用中,如何选择小波目前还没有很好地解决,这也是本文基于小波进行辨识信号降噪的一个关键环节,对此仍有待进一步研究。

参考文献:

- [1] TORVIK P J, BAGLEY R L. On the appearance of the fractional derivative in the behavior of real materials[J]. Journal of Applied Mechanics, 1984, 51(2): 294-298.
- [2] 方崇智, 萧德云. 过程辨识[M]. 北京: 清华大学出版社, 1988.
- [3] 王守觉, 李兆洲, 王柏南, 等. 用前馈神经网络进行带噪声信号的去噪声建模[J]. 电路与系统学报, 2000, 5(4): 21-26.
- [4] 姚宏伟, 梅晓榕, 庄显义. 模糊神经网络在噪声消除中的应用[J]. 电机与控制学报, 1999, 3(1): 50-52.
- [5] 刘福才. 非线性系统的模糊模型辨识及其应用[M]. 北京: 国防工业出版社, 2006.
- [6] 余世明, 冯浩, 王守觉. 基于小波和最小绝对误差的去噪抗扰动辨识方法[J]. 电子学报, 2003, 31(2): 192-195.
- [7] 程正兴, 杨守志, 冯晓霞. 小波分析的理论算法进展和应用[M]. 北京: 国防工业出版社, 2007.
- [8] MALLAT S. 信号处理的小波引导[M]. 杨力华, 译. 北京: 机械工业出版社, 2002.
- [9] DONOHO D L, JOHNSTONE I, KERGACHARIAN G, et al. Density estimation by wavelet thresholding[J]. The Annals of Statistics, 1996, 24(2): 508-538.
- [10] PODLUBNY I. Fractional differential equations[M]. San Diego: Academic Press, 1999.
- [11] 朱呈祥, 邹云. 改进的基于PSE和Tustin变换的分数阶系统求解递推算法[J]. 系统工程与电子技术, 2009, 31(11): 2736-2741.
- [12] COHEN A, DAUBECHIES I. Orthonormal bases of compactly supported wavelets III better frequency resolution[J]. SIAM Journal on Mathematical Analysis, 1993, 24(2): 520-527.
- [13] DAUBECHIES I. Orthonormal bases of compactly supported wavelets[J]. Communications on Pure and Applied Mathematics, 1988, 41(7): 909-996.

(上接第532页)

的计算时间和通信时间之比越大,则子作业之间有通信情况和无通信情况下的完工时间越接近,且完工时间取得最小值的分割粒度越细,完工时间的最小值越小。下一步将研究更复杂的子作业之间相互通信情况下,作业完工时间与作业分割粒度之间的关系。

参考文献:

- [1] University of California at Berkeley. Project list - BOINC[EB/OL]. [2010-04-19]. http://boinc.berkeley.edu/wiki/Project_list.
- [2] ANDERSON D P. BOINC: A system for public-resource computing and storage[C]// Fifth IEEE/ACM International Workshop on Grid Computing. Washington, DC: IEEE Computer Society, 2004: 4-10.
- [3] CERN. CMS-media[EB/OL]. [2010-07-18]. <http://cms.web.cern.ch/cms/News/2010/7TeVCollisions.html>.
- [4] BIRD I, ROBERTSON L, SHIERS J. Deploying the LHC computing grid - the LCG service challenges[C]// Local to Global Data Interoperability - Challenges and Technologies. Washington, DC: IEEE Computer Society, 2005: 160-165.
- [5] CERN. WLCG worldwide LHC computing grid[EB/OL]. [2010-

04-20]. <http://leg.web.cern.ch/leg/public/>.

- [6] ANDERSON D P, COBB J, KORPELA E, et al. SETI@home: An experiment in public-resource computing[J]. Communications of the ACM, 2002, 45(11): 56-61.
- [7] University of California at Berkeley. How SETI@home works[EB/OL]. [2010-07-12]. http://seticlassic.ssl.berkeley.edu/about_seti/about_seti_at_home_2.html.
- [8] 陈国良. 并行计算——结构·算法·编程[M]. 北京: 高等教育出版社, 2003: 77-84.
- [9] University of California at Berkeley. JobIn-BOINC-Trac[EB/OL]. [2010-04-19]. <http://boinc.berkeley.edu/trac/wiki/JobIn>.
- [10] YI S, KONDO D. RT-BOINC data format compaction[EB/OL]. [2010-04-19]. <http://rt-boinc.sourceforge.net/dataformat.html>.
- [11] BUYYA R, MURSHED M. GridSim: A toolkit for the modeling and simulation of distributed resource management and scheduling for grid computing[J]. The Journal of Concurrency and Computation: Practice and Experience, 2002, 14(13/15): 1175-1220.
- [12] KLUSACEK D. Alea - GridSim based grid scheduling simulator[EB/OL]. [2010-04-03]. <http://www.fi.muni.cz/~xklusac/alea/>.