第 35 卷 第 3 期 计 算 机 工 程 2009 年 2 月
Vol.35 No.3 Computer Engineering February 2009
·博士论文· 文章编号：1000—3428(2009)03—0018—03 文献标识码：A 中图分类号：TP393

基于可用性模型的志愿计算

王 宇，王志坚，黄晓萍，王从明
(河海大学计算机及信息工程学院，南京 210098)

摘 要：对志愿提供空闲计算资源为目的的高性能计算技术来说，资源提供者的可用性在一定范围内具有规律性和周期性。该文介绍用 Hurst 重标度和分形学理论分析 CPU 可用序列的方法，并应用该方法研究了志愿者可用性序列动态变化的分形特性。实验表明，志愿提供计算资源者可以看作一个复杂的非线性动力系统，用分形维数可以从整体上描述志愿计算系统的动态变化特征，用于高性能计算平台性能的测量，并在轻量级计算平台 XtremWeb 上进行实现。

关键词：志愿计算；可用性模型；XtremWeb 平台

Volunteer Computing Based on Availability Model

WANG Yu, WANG Zhi-jian, HUANG Xiao-ping, WANG Cong-ming
(Department of Computer and Information Engineering, Hohai University, Nanjing 210098)

【Abstract】 Volunteer computing is a form of distributed computing in which the availability of general public volunteers is regular and periodic. This paper proposes a novel approach based on Hurst's rescaled range analysis and fractal theory for time series are applied to study the deformation fractal characteristics of the volunteer computing for HPC Project. It is found that the volunteer may be regarded as a non linear dynamic system and the fractal dimension can be used to describe the dynamic variation characteristics of the whole platform and applied to diagnose the probable problem of the availability of node. The approach is implemented and validated on a platform XtremWeb.

【Key words】 volunteer computing; availability model; XtremWeb platform

1 概述

全球计算[1]汇集所有具有交互功能的空闲资源，向不同领域的科学计算或者商业应用的复杂问题提供巨大和便利的计算能力。“志愿”计算的核心思想就是资源提供者主动提供不连续空闲时间，以此来组成一个可共享的、持续稳定的计算资源。不同于传统的分布式系统，此类系统的性能不完全依赖作为中心节点的专用高性能计算机，而主要依赖于分布在不同管理域的、数量庞大的计算节点，也就是计算资源的提供者。波动性是这种主动计算节点的显著特征。任意数量的资源提供者会毫无征兆地突然离开系统或者随时加入到系统中来。即便在联机状态，也不能强迫资源提供者提供持续稳定的计算力，因此不完全可控的非专用资源就无法发挥可靠的作用。SETI@home[2]就利用节点屏保时间进行计算，XtremWeb[3]根据节点可配置的控制策略(如 CPU 负荷等)确定节点是否可用等问题。因此在这样的计算环境中，如何测量志愿者的性能是极具挑战的。

2 研究现状

资源共享环境中，保证这些资源提供者在时间上是可依赖的，关键措施就是对资源可用性建模以及基于可用性的调度。已经有相当多的研究人员关注到此类问题也提出了多种多样的可用性模型、可用性分析方法以及保证系统高可用性技术措施。文献[4]提供了基于 CPU 负荷的运行时刻任务剩余时间的预测方法。Condor 平台部署了 2 类中间件服务，其中关键原理就是利用以 CPU 的“周期窃取”实现计算资源进程级的调度。上述模型研究了如何测量计算资源的可用性建模和分析手段，有的还提供了一定程度的预测能力。在某一范围内，这些资源提供者的可用性具有周期性，例如 Overnet DHT 中的节点可用性呈现出 2 大分支：一类是每天有规律地进入/离开；一类是长期在线。本文拓展了文献[4]对节点可用性的两大类分支，针对志愿参与高性能计算的节点个体，提供了基于统计数据的细粒度的可用性描述手段。同时，设计并实现了基于这种可用性模型的任务运行时刻跟踪检测以及相应的适时调整调度框架。

3 志愿计算节点可用性模型

目前很多研究关注全局计算可用性度量模型，为任务调度、资源预留等提供决策依据以及相应服务。

对于网络可用性的分析，已经有大量理论和实践的积累[5]，本文不再缀述。因此，本文仅围绕节点所有者的使用行为对可用性模型的影响进行分析。

志愿计算环境中，计算节点的可用性有多种定义方式，一般定义如下：

(1)个体的容错性常被用来表示可用性：

可用性＝故障时间/(故障恢复时间＋故障时间)

$$\alpha_v = \frac{MVT}{MVT + MFT}$$

(2)志愿者的可用性由“实时”和“剩余”2 段组成：

基金项目：国家自然科学基金资助项目(60573098)；国家“973”计划基金资助项目(2002CB312002)；江苏省科技基金资助项目(BK 2006 168)

作者简介：王 宇(1979－)，男，工程师、博士研究生，主研方向：分布式计算，软件性能测试；王志坚，教授、博士生导师；黄晓萍、王从明，博士研究生

收稿日期：2008-10-31 **E-mail：**won9805@hhu.edu.com

$$Availability=\frac{probability\ \ forecast\text{-}passed\ time}{passed\ time}$$

(3)因为基于志愿者的行为满足某种随机过程，所以资源的时间和空间分布满足某种随机过程，本文假设每个志愿者的可用性都相似于某一个或多个随机过程：

可用性＝空闲时间/(空闲时间＋正常工作时间)

$$Volunteer\ Availability=\frac{Idle\ time}{Idle\ time+busy\ time}$$

CPU 的可用性用可用时间序列来表示，就是指在节点开机在线的情况下，机器空闲到足以向网格平台提供计算资源的比例。其中，对于空闲程度的理解，不同的网格中间件有不同的定义。可用时间可以用序列的形式表现出来(如图 1 所示)。用户操作 2 个任务之间的空闲时间，即为志愿者能够交给系统的可用时间，志愿计算资源提供者用户的可用性由 T_i 决定。根据前文的描述 T_i 在空间和时间上的分布具有随机性。

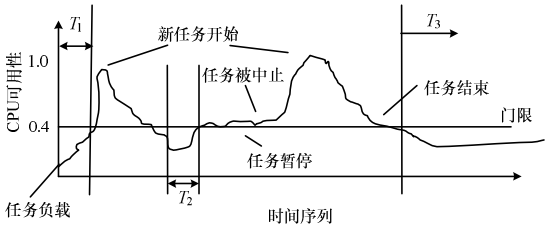

图 1　CPU 可用性序列分布图

随着志愿者涉及的范围扩大，这种局部周期的规律性将不存在。例如移动终端的加入使得随机性更加明显。所以研究这些可用序列在空间和时间上的分布具有的随机性必须通过追踪相应的数据来获得。追踪的方法很多，本文拟采用对 CPU 可用时间序列采样，用标准任务来测试各个志愿者的可用性。

4　可用性和调度框架

4.1　分形理论

1951 年 Hurst 提出的重标度极差分析的时间序列法与分形学理论的结合成功应用于解决许多领域与时间序列有关的问题。传统的时间序列方法应用某些原型观测资料分析时，是针对具体的某一观测量在特定时间段内的变化特征进行分析和拟合，并通过外推方法对其未来的变化进行预测。当观测量和时间范围改变时，模型也要做较大的改变，而且无法发现某一志愿提供者的观测量之间的一些共性。下式用于探索利用分形学理论计算 CPU 可用性序列的分形维数来描述某一志愿者系统的共性，并由此分析志愿者的工作动态：

$$\bar{x}_\tau=\frac{1}{n}\sum_{i=1}^{n}x_n \tag{1}$$

对于观测获得的效应量序列$\{x_i\}$,i=1,2,…, 取某一时间段 $\tau=t_n-t_1$，该时间段内的序列均值 τ 为一个周期，Hurst 指数与分形维数如下：

$$A(\tau,t_j)=\sum_{i=1}^{n}(x_{tj}-\overline{x_\tau}) \tag{2}$$

其中，$A(\tau,t_j)$是在时刻 t_j 观测量的累积偏差。

$$S(\tau)=\left|\frac{1}{\tau}\sum_{i=1}^{n}(x_{tj}-\overline{x_\tau})^2\right|^{1/2} \tag{3}$$

标准差为

$$R/S=(\tau/2)^H \tag{4}$$

其中，$R/S=R(\tau)/S(\tau)$被称为 H 指数。

$$R(\tau)=\max_{1\leqslant i\leqslant n}A(\tau,t_i)-\min_{1\leqslant i\leqslant n}A(\tau,t_i) \tag{5}$$

Hurst 得出许多自然现象呈现的不是随机规律而是与时间序列相关的。研究发现 H>0.5 时，各观测量之间是正相关的，即观测量的值在过去的时间内增大则将来也会增大，在过去时间内减小则将来也会减小；H<0.5 时，各观测量之间是负相关的。

4.2　用户可用性模型建立及应用

若志愿者的工作动态没有发生大的改变，用来描述志愿者可用性的分形维数不随外部环境量和时间跨度的改变而改变，体现了系统的自相似性。因此可以通过计算 CPU 可用性的分形维数，然后选用适当的分形动力学模型，来建立用户可用性的时间序列监控模型。

例如设 F 为 Rn 中的任何子集，s 为一非负数，对任何 δ>0，定义

$$h_\delta^s(F)=\inf\left\{|U_i|^2:F\subset\bigcup_{i=1}^{\infty}U_i,0<|U_i|\leqslant\delta\right\} \tag{6}$$

其中，U_i 为直径不超过δ的 F 一个覆盖，U_i 也是 Rn 中的子集，$|U_i|$表示 U_i 的直径。

$$h_\delta^s(F)=\lim_{\delta\to0}h_\delta^s(F) \tag{7}$$

对 Rn 中的任何子集 F 这个极限都存在，但极限值通常是 0 或∞，则称 $S(F)$为 F 的 S 维 Hausdorff 测度。Hausdorff 维数 D 定义如下：

$$D=\dim_H F=\inf\left\{S:h^s(F)=0\right\}=\sup\left\{S:h^s(F)=\infty\right\} \tag{8}$$

D=1 时，表示曲线；D=2 时，表示曲面；D=3 时，表示几何体，但通常情况 D 值不是整数。根据有关研究资料，Hausdorff 维数 D 与 Hurst 指数 H 之间存在如下关系：

$$D=2-H \tag{9}$$

可用性建模的目的是为了使用户提交的任务能够在最佳效率下执行完毕。正如在概述中所讲，由于计算节点的偶然性，系统很难准确地预测任务的完成时间。CPU 是否可用、空闲的利用率如何、系统和用户级对资源的使用强度和使用条件，是决定志愿者是否可用的充分必要条件。即：主机可用，通信可达，CPU 可控。本文的可用性研究主要以 CPU 的可用性为主线。

如图 2 所示，RP 表示资源提供者，RC 表示资源控制节点，记录系统中所有志愿者的信息，CT 表示客户端，用于提交计算任务并把结果返回用户。

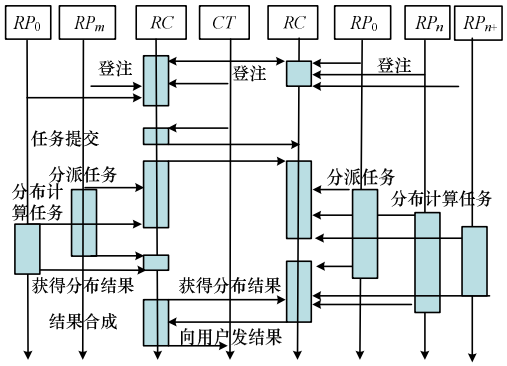

图 2　志愿计算交互时序

任务分派和资源调度算法如下：

(1)任务 J 到达 RC，RC 检查 RP 可用性，收集志愿提供 R 的节点的可用性信息，计算 H 值。

(2)for (k = 1 ; k < m ; k ++)than compute the T until T_k >

T_k+1

按照资源可用性队列预测结果并分割任务指派到各个计算节点。

(3)计算各个计算节点的 *D* 值，分别比对 *D* 值与 1, 2, 3 的距离，选择拟合曲线(最近 1, 2, 3 分别选择线性拟合，曲面拟合，几何体拟合)，周期性监视任务完成情况，直到任务完成。

(4)如果灾难性的事件发生，例如 RC 迁移和变化，则根据 *D* 值选择重新分配和调度策略。

(5)if original scheduling (time) > rescheduling (time) then rescheduling

(6)*RC* 主动询问各个 *RP* 是否完成任务。

调度算法通过预测任务在资源提供者上的完成时间，用随机分形动力学的方法预测资源“时间”和“空间”上资源提供的有效区间。进行任务分配和资源调度，任务分割，再把子任务分派给适合的资源提供者，根据可用性监控任务的执行过程，直到任务完成。并且周期性地考虑遇到特殊事件时如何处理任务。

5 框架实现与验证

XtremWeb 的周期窃取策略是可配置的，*worker* 即前面提到的 RP 资源提供者。在其宿主机 CPU 负荷降低到一定阈值后，向 *master* 即 *RC* 资源控制节点请求任务，进行计算。

对 XtremWeb 进行了改造。在其平台上实现了对 *worker* 的可用性测试和计算，这是基于构件的灵活设计，还可以支持 *RC* 与 *worker* 之间 1：*n* 的检测模式。试验平台包括 1 个 *master* 节点和 64 个 *worker* 节点，*worker* 节点中 52 台属于实验室中的学生用机，其余 12 台为教职员工的办公用机。所选 *worker* 节点都为 PC，硬件软件配置大致相同。经过近 4 周的运行，对其中 5 个节点的可用性进行了统计分析。

以 2 天为一个周期 τ，固定时长间隔 *T* 为单位，将其分隔成若干独立的时间片断 t_i。该计算节点的可用时间模型为：$(t_1, t_2,\cdots, t_i,\cdots, t_n)$。其中，$n=7\times24/T$，*T* 是时间片间隔单位长度。基于统计观测的某个时间片断的可用时间又可定义为：$S(\tau)$，平均可用时间的样本标准方差。根据式(1)~式(5)分别把 t_i 带入，可求出 *H* 指数。由 *R*/*S* 计算 *H* 时 200 个左右样本数据点计算得的 *H* 精度约为 90%，2 000 个左右样本数据点计算得的 *H* 精度约为 95%。

(1)由 *R*/*S* 求出 Hurst 指数。图 3 为实验机器计算所得的 *H* 值曲线。篇幅所限，表 1 仅列出了图 3 中几个实验者 CPU 可用性的 *H* 值的变化范围、平均值、方差等。

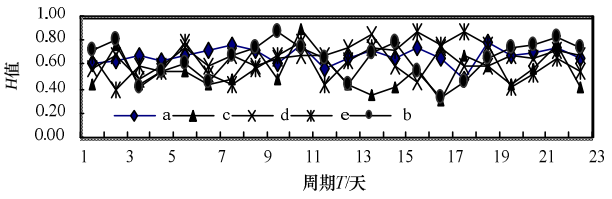

图 3　部分志愿者 *H* 曲线

表 1　*H* 值和维数 *D* 实验结果

编号	*H* 平均	*H* 方差	*D* 平均	*D* 方差
a	0.674 500	0.005 376	1.458 182	0.022 682
b	0.541 818	0.022 682	1.357 273	0.008 126
c	0.642 727	0.008 126	1.325 500	0.005 376
d	0.632 927	0.020 450	1.367 073	0.020 450
e	0.634 091	0.028 578	1.365 909	0.028 578

从图 3、表 1 中可以看出，当每 2 天计算所得的 *H* 值非常相近，即 lg(*R*/*S*)与 lgτ 呈线性关系。大部分 *H* 值都大于 0.5，即各观测量之间正相关，测值过程线也呈现了相似的规律。

(2)计算 Hausdorff 维数 *D*。图 4 为实验节点计算所得的 *D* 值曲线。*D* 值的变化范围、平均值等见表 1。对同一被测机每个时段的 *D* 值非常接近，因此本文实验范围中的志愿者可以看作一个非线性动力系统，汇总得到系统的分形维数 *D* 大约在 1.37。*D* 的大小是衡量系统复杂程度的重要指标，*D* 与 1 的距离比较接近，说明实验用志愿计算机的 CPU 可用性比较接近单边随机游动占主要部分，同时 *D* 大于 1 说明系统同时存在非线性变形。

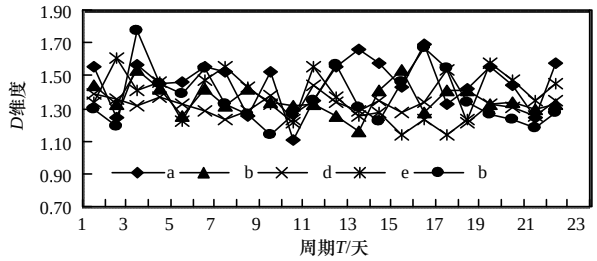

图 4　部分志愿者 *D* 维数曲线

6 结束语

本文介绍了基于志愿者可用性模型的任务监控框架。其中分形可用性模型不同于传统的二分模式，采取了基于统计的细粒度的建模方式。鉴于志愿者行为的高波动性，很难准确地预测给出量化的指标用于评估当前节点可提供高性能计算的能力。该框架已经在 XtremWeb 上得以实现。大量的试验数据表明基于该可用性模型进行志愿者计算任务监控是合理、可行、有效的。

过于简单的模型包含的信息量往往是不够的。根据现阶段的研究结果，推断出基于志愿者节点可用性时间模型的评估和决策机制可以采用分形维数来调度，再采用线性和非线性曲线来做拟合。因此，需要设计相关的仿真平台和工具，以进一步测试系统可用性以及对志愿者系统任务的影响。而突发性既要考虑长时间尺度的突发，也要考虑短时间尺度的突发。因此，寻找一个具有恰当精度并具有可分析性的原型，依然是一个有待深入研究的问题。

参考文献

[1] Anderson D P, McLeod J. Local Scheduling for Volunteer Computing[C]//Proc. of Parallel and Distributed Processing Symposium. Long Beach, CA, USA: [s. n.], 2007.

[2] Foster I, Kesselman C, Nick J, et al. The Physiology of the Grid: An Open Grid Service Architecture for Distributed Systems Intergration[EB/OL]. (2002-06-22). http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.14.8105.

[3] James P, Karim D, Dew P M. Grid-based SLA Management[M]. Heidelberg, Germany: Springer. 2005.

[4] Li Chunjiang, Li Dongsheng, Xiaonong, et al. A Measurement Model for the Availability of Applications in Computational Grid[J]. Journal of Computer Research and Development, 2003, 40(12): 1705-1709.

[5] Bhagwan R, Savage S, Voelker G M. Understanding Availability[C]//Proc. of the 2nd International Workshop on Peer-to-Peer Systems. Berkeley, CA, USA: [s. n.], 2003: 1-11.