网格环境中节点 CPU 负载的分形预测*)

张 飞 曾国荪

(同济大学计算机科学与技术系 上海 200092)

(国家高性能计算机工程技术中心同济分中心 上海 200092)

摘 要 网格环境下,常常需要知道网格资源在未来某一时刻具有什么样的性能,比如,调度器需要该性能估测以便进行高效的资源调度、提供满足要求的 QoS 以及保证整个网格系统的负载平衡。正如在其他任何计算环境中一样,计算能力是所有网格资源中最为重要的资源,通常用 CPU 负载来刻画节点主机的忙碌程度、衡量节点所能提供的计算能力。已有的研究表明 CPU 负载具有自相似性和长相关性,这启发我们使用本文介绍的分形的方法进行 CPU 负载的预测。实验结果证明该方法具有较高的预测精度,因而具有较好的实用价值。

关键词 网格,资源性能,CPU 负载,分形,预测

Fractal Prediction of CPU Load in Grid Environment

ZHANG Fei ZENG Guo-Sun

(Department of Computer Science and Technology, Tongji University, Shanghai 200092)

(Tongji Branch, National Engineering & Technology Center of High Performance Computer, Shanghai 200092)

Abstract In grid environment, it is in need to know what performance grid resources can afford at some given time in the future. For instance, the scheduler need this performance estimation to make efficient resource scheduling, provide desired QoS and keep the whole grid system optimally load-balanced. Among all kinds of grid resources, computing circles are most valuable in grid system as well as in other computing environments. CPU load is naturally used to depict how busy the host machine is and what performance the host can give to user programs. Former research on CPU load shows that CPU load has self-similarity and long-dependence. This leads to our employing fractal geometry methods to make CPU load prediction described in this paper. The experimental results are encouraging and the accuracy this method yields is also quite satisfying.

Keywords Grid, Resource performance, CPU load, Fractal, Prediction

对于一般用户而言,网格[1]这台超级计算机与传统意义的计算机有着惊人的相似:功能上可以提供计算能力;结构上既有计算、存储、输入输出等"硬件"系统,又有"操作系统"管理这些"硬件"资源;安全上只有具有相应权限才能够使用系统资源;应用上可以通过系统的编程接口编写新的服务实现特定的功能。正如操作系统是传统计算机的核心,网格"操作系统"—网格管理软件是网格之为网格的关键所在,且生来便具有公约的规范:OGSA 及随后的 WSRF 是网格标准化的探索与努力,GlobusToolkit 则是上述规范的具体实现。

网格中存在大量异构的、种类繁多的、动态变化的网格资源,网格资源管理的一个重要方面就是资源状态信息管理,要能够随时提供网格资源当前各种状态信息。然而,我们为什么需要知道资源当前的状态信息呢?不难发现,这里面其实存在这样一个假定:资源当前状态暗示了其在不远的将来所处的状态。这种假定实际上是一种预测行为。对于那些变化极其缓慢的资源信息(如节点操作系统相关信息)来说,这种方法的预测精度几乎总是 100%;而对于那些变化比较频繁,甚至每时每刻都在变化的资源信息(如节点 CPU 负载)来说,这种假定则没有太大的意义。能够根据需要提供资源在未来某一时刻或某段时间内的状态信息,这将更加接近资源信息服务的本意。

除了资源管理之外,网格"操作系统"还负责任务的分配与调度。有效的资源管理使得高效的任务调度成为可能。提交的任务如何分块,分块的子任务如何调度执行,各个子任务大致的执行时间以及整个应用的执行时间等等应该根据实际可用的资源状况来决定,显然,仅仅使用资源当前状态信息如果不是毫无意义,至少也是远远不够的。节点 CPU 负载、可用内存、网络可用带宽等因素在未来时刻的预想值在这里再次占据至关重要的位置。

除对内有效的管理资源和资源调度之外,网格还必须对外提供高质量的服务。随着网格的日益流行,网格掌握的计算资源越来越丰富,网格应用数量与日俱增,应用本身也日趋庞大和复杂,如何有效地调度这些应用以取得上佳的服务质量,引起人们越来越广泛深入的关注与研究。物理上看,网格由地理上相互分布、相互之间用网络连接的计算机组成。网格应用实际上是分块后分配到这些网格终端节点上执行。因此,这些节点计算机的 CPU 负载、内存的可用情况、相互之间的网络负载等资源状况对于保证 QoS 有着极其重要的意义。

本文尝试采用分形方法根据已有的 CPU 负载监测数据预测未来的系统 CPU 负载,并通过实验数据对该预测算法的精确度进行了检验。

1 CPU 负载预测相关研究

在如何预测 CPU 负载方面已经有不少的相关研究。文

*)基金项目:国家自然科学基金资助项目(60173026),上海科委重大项目(03DZ15029),上海高校网格技术 E-研究院资助(200301-1)。张 飞 硕士研究生,研究方向为网格计算。曾国荪 教授,博导,主要研究领域为异构计算,计算网络。

[2]提出两种截然不同的预测方法。一种认为CPU负载的变化是一个稳定的过程，即总是倾向于向其稳定值-平均值变化，当前值大于均值时，预测值取当前值减去某个值；当前值小于均值时，预测值取当前值加上某个值。显然，在负载高于均值且持续上升阶段或低于均值且持续下降阶段，预测误差越来越大。另一种方法换为跟着趋势走，即如果当前值较之前是增长的，则认为下一时刻也是增长的，反之，当前值是减小的，则未来值也服从该下降趋势。该方法的缺点是在变化趋势由增长转入下降或从下降转而上扬时，误差太大，趋势变化越频繁，误差越大。

让我们再来看看当前在网格预测中应用的较多的NWS[3]，其用来进行预测的模型较为简单，如简单的取均值、取中值、传统的线性回归模型等等，这些简单的数学模型显然无法精确刻画复杂多变的CPU负载变化曲线。

那么，有没有什么描述功能更强大的数学工具可供我们选择呢？幸运的是，答案是肯定的。

已有的对于网络数据包流量[4]、万维网流量[5]、网络协议性能[6]、网络视频传输[7]以及网络互连文件系统[8]的研究均表明，自相似性是现代分布式系统的固有特征。我们自然而然地会考虑这样一个问题：系统的CPU负载是否也具有自相似性呢？

Dinda和O'Halloran[9]专门对Unix系统的CPU负载进行了深入的研究。他们在一年中不同时段先后两次以1Hz的频率(对于用户态运行的程序来说，1Hz的采样频率提供了足够的负载变化的动态信息)对包括计算机集群、服务器、桌面工作站三大类在内的38种不同机器的CPU负载进行长达数周的采样分析。两份数据的统计分析结果大体一致，其中令我们感兴趣的一个重要结论就是：CPU负载同样具有自相似性，其Hurst指数大致在0.63到0.97之间浮动且明显倾向于峰值。而我们知道，当 $H\neq 1/2$ 时，意味着持久性，即所研究的时间序列不是相互独立的，而是有相关性；当 $H<1/2$ 时，过去的增量与未来的增量负相关，过程具有反持久性；而当 $H>1/2$ 时，用平均的观点看，过去的增长意味着将来的一个增长趋势，反之亦然，即过程具有持久性，并且H愈趋近1.0，其持久性愈强，运动轨道愈平滑。

可见，CPU负载具有很强的自相似性和长相关性。这为我们用分形方法预测CPU负载提供了依据。

2 分形预测原理及模型

2.1 分形拼贴定理

分形拼贴是分形理论的基本定理之一。设(X,d)是一个完备距离空间，令 $L\in H(X)$，且 $\varepsilon\geqslant 0$ 给定，选取一个压缩因子 $0\leqslant s<1$ 的迭代函数系$\{X;w_1,w_2,\cdots,w_n\}$，使得：

$$h(L,\bigcup_{i=0}^{n} w_i(L))\leqslant\varepsilon \tag{1}$$

则有：

$$h(L,A)\leqslant\frac{\varepsilon}{1-s} \tag{2}$$

式中 $h(L,A)$ 为Hausdorff距离；A 为迭代函数系$\{X;w_1,w_2,\cdots,w_n\}$的吸引子。

分形拼贴定理说明存在一个迭代函数系$\{X;w_1,w_2,\cdots,w_n\}$，其吸引子 A 近似于或相似于一个给定的集合 L。

2.2 分形插值

Barnsley[10]给出了利用插值方法构造上述迭代函数系统的过程，称为分形插值方法。分形插值方法实际就是构造出一个IFS，使其吸引子 A 为插值函数 f 的图形。分形插值方法以给定集合为分形插值函数 $f(x)$，所构造的迭代函数系$\{R^2;w_1,w_2,\cdots,w_n\}$将使其吸引子趋近于分形插值函数 $f(x)$ 的图像。该迭代函数系$\{R^2;w_1,w_2,\cdots,w_n\}$中的每个函数 w_i 是仿射变换，其构造为

$$w_i\begin{bmatrix}x\\F\end{bmatrix}=\begin{bmatrix}a_i & 0\\c_i & d_i\end{bmatrix}\begin{bmatrix}x\\F\end{bmatrix}+\begin{bmatrix}e_i\\f_i\end{bmatrix} \tag{3}$$

满足

$$w_i\begin{bmatrix}x_0\\F_0\end{bmatrix}=\begin{bmatrix}x_{i-1}\\F_{i-1}\end{bmatrix} \tag{4}$$

$$w_i\begin{bmatrix}x_N\\F_N\end{bmatrix}=\begin{bmatrix}x_i\\F_i\end{bmatrix} \tag{5}$$

得方程组

$$\begin{cases}a_ix_0+e_i=x_{i-1}\\a_ix_N+e_i=x_i\\c_ix_0+d_iF_0+f_i=F_{i-1}\\c_ix_N+d_iF_N+f_i=F_N\end{cases} \tag{6}$$

取定 d_i，则可以解出

$$\begin{cases}a_i=\dfrac{x_i-x_{i-1}}{x_N-x_0}\\e_i=\dfrac{x_Nx_{i-1}-x_0x_i}{x_N-x_0}\\c_i=\dfrac{F_i-F_{i-1}}{x_N-x_0}-d_i\times\dfrac{F_N-F_0}{x_N-x_0}\\f_i=\dfrac{x_NF_{i-1}-x_0F_i}{x_N-x_0}-d_i\times\dfrac{x_NF_0-x_0F_N}{x_N-x_0}\end{cases} \tag{7}$$

到此，即确定了迭代函数系中的第 i 个变换。

2.3 负载预测模型

根据式(1)，负载预测模型的一般形式为

$$\bar{L}=\bigcup_{i=1}^{P}\overline{w}_i(L) \tag{8}$$

其中，$\bar{L}$ 为负载预测值集合，L 为负载历史纪录集合，$\overline{w}_i$ $(i=1,2,\cdots,P)$为从负载历史记录中确定的统计意义上的IFS的第 i 个仿射变换。

3 CPU负载预测

3.1 实际预测方法

Wolski等[11]关于CPU负载预测的研究发现，尽管CPU负载具有很强的长相关性，通常最近的采样数据对下一时刻的负载值却具有较大的影响。因此，进行负载预测时，我们既要考虑长相关性对于负载走向的作用，也要注重将最近时期数据的较大的影响反映在我们的预测模型中。

已有的利用分形插值方法进行预测的研究[13,14]将历史数据分成等长的若干样本，对每个样本分别构造一个IFS，然后从这若干个IFS中构造出一个新的统计意义上的IFS用于预测。这无疑对样本长度的选择具有较严格的要求，只有选择合适的样本长度，才能使在该标度区间上样本具有较好的相似性，从而做出较为精确的预测，而如何确定该样本长度有待于进一步的研究。另外，这种方法是对将来的一个区间内的特征值进行预测，如果单单观察该预测区间内某个时刻的预测值，则会发现该预测值与用来预测该值的历史数据间存在一定的"延迟"，也就是说离该预测值最近的特征值没有反映在预测过程中。

基于以上考虑，我们使用以下预测方法进行CPU负载的预测。

设$\{L_n, L_{n-1}, L_{n-2}, \cdots, L_1, L_0\}$为历史记录序列，$L_0$为当前记录，预测迭代函数系为$\{R^2; w_0, w_1, w_2, \cdots, w_m\}$，则预测值

$$\hat{L}=\sum_{i=0}^{m} t_i w_i L_i \tag{9}$$

其中t_i为映射w_i的权重，$i=0,1,2,\cdots,m$，满足$0\leqslant t_i<1$，且$\sum_{i=1}^{m} t_i=1$。本文取时间相隔越近，权重越大，因此有$t_0\geqslant t_1\geqslant t_2\geqslant\cdots\geqslant t_m$。上式中$w_i$的构造过程如下：

取Δt_i为L_i与L_0的时间间隔，以$L_{\Delta i-1}$为起点，Δt_i为时间间隔，依次向后取N_i个样本点，连续取M_i个这样的样本，然后在这M_i个样本上用第三部分介绍的方法，d_{ij}取样本均值与样本最大值的比值，得到一个统计意义上的$IFS_i=\{R^2; w_{i1}, w_{i2}, \cdots, w_{ip}\}$，根据式(4)，取$w_i=w_{i2}$。

构造出预测IFS后，则根据式(9)得到下一时刻CPU负载的预测值。

3.2 历史数据采集

本文采用分形的方法对网格环境中的重要资源—系统平均负载进行预测。Unix系统任一时刻的负载指的是系统中当前正在运行或处于就绪状态的进程数，也即由操作系统scheduler维护的就绪队列的长度。内核以一定的频率采样该就绪队列的长度，用前几次的采样结果按照一定的算法平均得到系统的平均负载(load average)，用户进程在权限许可的条件下可以直接访问该统计数据。本文用系统的平均负载作为刻画CPU负载的衡量指标。

本文用于预测的历史数据来自于作者所在实验室的曙光3000高性能计算机。该计算机安装的操作系统为AIX 4.3。我们专门编写了一个工具以每10秒一次的频率对系统在过去1分钟的平均负载进行采样，并将所得的时序记录以文件的形式保存下来以供分析预测之用。

我们将所得的数据以文本方式保存在文件中，如图1所示。

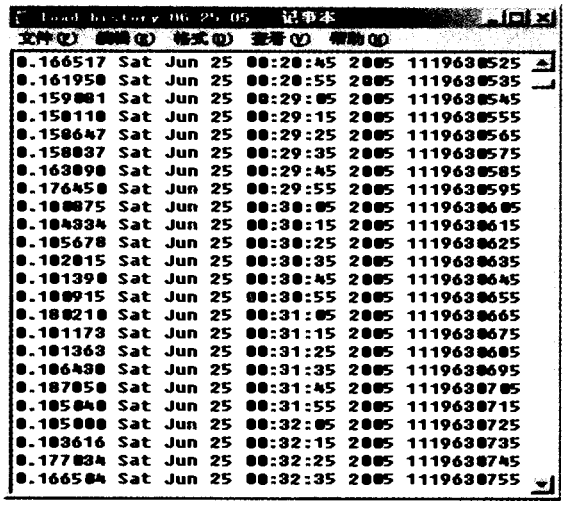

图1 CPU负载历史数据

其中包括3个字段，依次为平均负载值、时间标签以及一个整数，该整数表示自1970年1月1日以来经过的秒数。

3.3 预测分析

3.3.1 历史数据R/S分析

因为利用分形插值方法进行预测，则历史数据的自相似性对预测精确度有着至关重要的影响，这里我们采用Hurst指数作为衡量时间序列自相似性的指标。

我们用文[12]中介绍的R/S分析方法，取y轴为$\lg[R(T)/S(T)]$，x轴为$\lg T$，则理论上有：

$$\lg[R(T)/S(T)]=H\times\lg T+H\times\lg\alpha$$

其中α为常数，$R(T)$为样本极差，$S(T)$为样本标准差，T为时间，则H为Hurst指数，对时间序列而言，分维数$D=2-H$。我们对连续10天的采样数据进行分析，所得结果如图2所示。

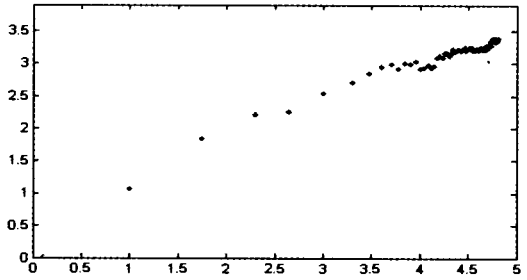

图2 CPU负载Hurst指数分析

可见，CPU负载的确具有很强的自相似性，H在0.66～1之间变动，大致估计为0.75。

3.3.2 预测误差分析

我们取$m=3, t_0=0.5, t_1=0.25, t_2=t_3=0.125, N_i=M_i=6(i=0,1,2,3)$，对一段时间(2分钟，间隔为10秒)的CPU负载进行预测并与实际记录值进行比较，结果如表1所示。

表1

时间	实际值	预测值	相对误差
1	0.166517	0.176008	5.7%
2	0.161958	0.167627	3.5%
3	0.159081	0.169898	6.8%
4	0.158118	0.153058	−3.2%
5	0.158647	0.148970	−6.1%
6	0.158837	0.151848	−4.4%
7	0.163898	0.162587	−0.8%
8	0.176450	0.187213	6.1%
9	0.180875	0.187748	3.8%
10	0.184334	0.185071	0.4%
11	0.185678	0.195519	5.3%
12	0.182815	0.191773	4.9%

这里我们采用固定的权重，若进一步对参数进行训练，则有望提高预测精度。下一步我们考虑采用神经网络等方法对各映射权重及样本长度、样本个数进行训练。

结论 研究发现，不同时间间隔的采样特征值对于未来的特征值有着程度不同的影响，时间距离越近，影响越显著。据此，本文所用的预测方法将CPU负载在下一时刻的值视为不同时间尺度上各种趋势综合作用的结果，作用程度的大小通过各映射的权重系数来调节，通过对这些系数的选择以求更为全面、精确地反映CPU负载曲线的变化趋势，得到更为精确的预测精度。

从实验结果误差来看，采用该方法进行预测具有较高的准确性，且计算速度快、无收敛性问题，具有比较大的实用价值。

我们下一步考虑使用神经网络等方法对各参数进行训练

(下转第79页)

于不同的 LDPC 码和在不同的信噪比的条件下，α 的最优值往往是不相同的，而且最优值也只能通过仿真运算得到。图 1 给出了加权参数对于 Type-I 2-D(1023,781) EG-LDPC 码性能的影响。从仿真结果可以看出，本文所提出的解码算法的性能随着加权参数 α 的变化而变化，并且在 α 值的仿真范围中，存在 α 的最优值。另外，我们还注意到，在 α 的某一特定范围内，解码算法的性能对于 α 的变化并不是十分的敏感。也就是说，当 α 的取值为这一范围中任意值时，解码算法仍然可以取得接近最优值的性能。

在图 2 中，我们给出了本文所提出的解码算法在加权参数 α 取最优值条件下解码 Type-I 2-D(1023,781) EG-LDPC 所需要的平均迭代次数。另外，图中还给出了加权 BF 算法所需的平均迭代次数，作为比较。从图中可以看出，与加权 BF 算法相比，本文所提出的算法在不同的信噪比条件下，均需要较少的迭代运算次数，从而说明本文所提出的解码算法有着更快的收敛速度。

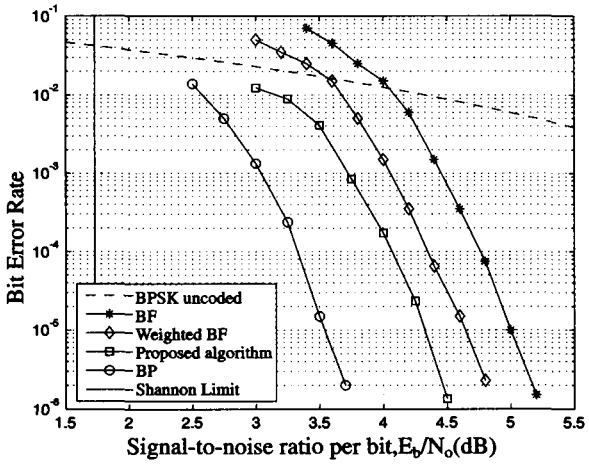

图 3　Type-I 2-D (1023,781) EG-LDPC 码在不同解码算法下的性能

最后，我们采用 Monte Carlo 法仿真了当采用这 4 种不同的解码算法时 Type-I 2-D(1023,781) EG-LDPC 码在 AWGN 信道下的性能，如图 3 所示。同时，在图中我们还标出了没有采用纠错码而仅采用 BPSK 调制时系统的性能和 AWGN 信道的 Shannon 限作为研究系统性能的参考。从仿真结果可以看出，当误码率 BER$<10^{-5}$时，本文所提出的算法与加权 BF 算法相比，额外提供了 0.35 dB 的编码增益。如果与 Gallager 提出的 BF 算法相比，则额外提供了 1.4 dB 的编码增益。而且，本文所提出的算法与标准的置信传播算法相比，仅仅有 0.7 dB 的距离。从而说明，本文所提出的解码算法作为一种比特翻转算法，具有良好的性能，在解码复杂度与纠错性能两者之间提供了一种很好的均衡。

结论　本文对于规则 LDPC 码提出了一种两阶段的比特翻转解码算法。在解码算法的每一次迭代过程中，解码算法的翻转比特选择过程可以分为两个阶段。首先，解码算法从校验节点的角度选择候选翻转比特。所有满足可靠性要求的校验节点从与它们邻接的信息节点中各自选择一个最有可能错误的信息比特作为候选翻转比特。在第一阶段的选择过程中，我们综合考虑信息比特向量的纠错伴随式和对数似然比，提出了新的翻转比特选择标准。另外，由于只有小部分校验节点进行比特选择运算，因此校验节点选择运算的运算复杂度不会太高。然后，解码算法对于从第一阶段所选择的候选翻转比特进一步细化，从而进一步降低了误翻的概率。具体而言，解码算法对于候选翻转比特采用投票的方法，从中选择出最有可能是错误的信息比特最为本次迭代运算最终的选择结果。仿真结果表明，本文所提出的解码算法在解码运算复杂度和纠错性能之间提供了一种很好的均衡，为工程人员在设计通信系统时提供了另外一个选择。

参 考 文 献

1　Gallager R G. Low density parity check codes. IRE Trans Info Theory, 1962, IT-8: 21～28
2　——. Low Density Parity Check Codes. Research monograph series. Cambridge, Mass.: MIT Press, 1963
3　Tanner R M. A recursive approach to low complexity codes. IEEE Transactions on Information Theory, 1981, 27(5): 533～547
4　MacKay D J C, Neal R M. Near Shannon limit performance of low density parity check codes. Electronics Letters, 1996, 32(18): 1645～1646
5　MacKay D J C. Good error correcting codes based on very sparse matrices. IEEE Transactions on Information Theory, 1999, 45(2): 399～431
6　Chung S Y, Forney G D, Richardson T, et al. On the design of low-density parity-check codes within 0.0045 dB of the Shannon limit. IEEE Commun Lett Feb. 2001, 5: 58～60
7　Kou Y, Lin S, Fossorier M P C. Low density parity check codes based on finite geometries: A rediscovery and new results. IEEE Transactions on Information Theory, 2001. 2711～2736

(上接第 63 页)

以期进一步提高预测精度。

参 考 文 献

1　Foster I, Kesselman C. The Grid: Blueprint for a New Computing Infrastructure. Morgan-Kaufmann, 1999
2　Yang L, Foster I, Schopf J. Homeostatic and Tendency-based CPU Load Predictions. In: Proc. of IPDPS 2003, April 2003
3　Wolski R, Spring N, Hayes J. The Network Weather Service: A Distributed Resource Performance Forecasting Service for Metacomputing. Journal of Future Generation Computing Systems, 1998
4　Leland W, Taqqu M, Willinger W, Wilson D. On the self-similar nature of Ethernet traffic. IEEE/ACM Transactions on Networking, Feb. 1994
5　Croyelia M, Bestavros A. Self-similarity in world wide web traffic: Evidence and possible causes, IEEE/ACM
6　Park K, Kim G, Crovella M. On the effect of traffic self-similarity on network performance. In: Proc. of the SPIE International Conference on Performance and Control of Network Systems, Nov. 1997
7　Bernan M, Serman R, Taqqu M. Long-range dependence in variable-bit-rate video traffic. IEEE Transactions on Communications, March 1995
8　Gribble S, Mandu S, Roselli D. Brewer Self-similarity in file systems. available from http://www.cs.berkeley.edu/~gribble
9　Dinda P, O'Halloran D. The statistical properties of host load, In to appear in the Fourth Workshop on Languages, Compilers, and Run-time Systems for Scalable Computers(LCR98) and CMU Tech, report CMU-CS-98-143, 1998, available from http://reports-archive.adm.cs.cmu.edu/anon/1998/CMU-CS-98-143.ps
10　Barnsley M. Fractals Everywhere, Academic Press Inc, 1998
11　Wolski R, Spring N, Hayes J. Predicting the CPU availability of Time-shared Unix Systems. In: Proc. of 8th IEEE High Performance Distributed Computing Conference(HPDC8), 1999
12　陈凌，等编著. 分形几何学. 北京：地震出版社，1998
13　唐立春，李光熹，熊曼丽. 基于分形的电力系统负荷预测. 电力系统及其自动化学报，1999，11(4)
14　梁平，樊福梅，吕玉坤. 电力系统负荷分形预测及 R/S 分析. 华北电力大学学报，2004，31(4)