Proc. IEEE GLOBECOM, Houston, TX, Dec. 2011

On Characteristics and Modeling of P2P Resources with Correlated Static and Dynamic Attributes

H. M. N. Dilum Bandara and Anura P. Jayasumana
Department of Electrical and Computer Engineering, Colorado State University, Fort Collins, CO 80523, USA.
dilumb@engr.ColoState.edu, Anura.Jayasumana@ColoState.edu

***Abstract*— Modeling and simulation of Peer-to-Peer (P2P) resources with correlated static and dynamic attributes is essential in application design, validation, and performance analysis. A novel mechanism is presented to generate realistic synthetic traces of multivariate static and dynamic attributes of P2P resources. The methodology is demonstrated using characteristics of PlanetLab node traces. First, a multi-attribute resource model is defined using a selected set of static and dynamic attributes. Second, characteristics of resources are presented. We observe that attribute values are correlated, follow a mixture of probability distributions, and time series of some of the dynamic attributes are nonstationary. Third, random vectors of static attributes are generated using empirical copulas that capture the entire dependence structure of multivariate distribution of attributes. Finally, time series of dynamic attributes are randomly drawn from a library of multivariate-time-series segments extracted from PlanetLab traces. These segments are identified by detecting the structural changes in time series corresponding to a selected attribute. Time series corresponding to rest of the attributes are split at the same breakpoints and randomly drawn together to preserve their contemporaneous correlation. Furthermore, a tool is developed to automate the synthetic data generation process and its output is validated using statistical tests.**

Keywords- multivariate resources; peer-to-peer; resource discovery; simulation; synthetic P2P node traces; volunteer computing

I. INTRODUCTION

Models characterizing resources, resource attributes, and demand on resources at computing/storage nodes and end hosts are vital for the design, validation, and performance analysis of many distributed application domains, e.g., cloud computing. Such analysis is of particular interest in collaborative Peer-to-Peer (P2P) systems, desktop grids, and volunteer computing that utilize large numbers of heterogeneous, distributed, and dedicated/voluntary resources. For example, BOINC [1] is a volunteer computing platform that is used to remotely execute jobs using idle computing resources. BOINC schedules jobs based on *static attributes* (e.g., CPU speed, total memory, presence of hardware accelerators, etc.) of nodes as the jobs are expected to run for several hours and the system is optimized for throughput. In contrast, performance, Quality of Service (QoS), and Quality of Experience (QoE) of latency sensitive applications such as Collaborative Adaptive Sensing of the Atmosphere (CASA) [2] and community cloud computing [3] also depend on *dynamic attributes* (e.g., CPU utilization, free memory, and bandwidth). CASA is an emerging heterogeneous network of weather radars, processing nodes, and data fusion algorithms (e.g., tornado tracking and precipitation estimating algorithms) that operates collaboratively to detect hazardous atmospheric conditions. Collaborative P2P data fusion provides an attractive implementation choice for CASA real-time radar data fusion, weather monitoring, and hazard prediction because data is constantly being generated, processed, pushed and pulled among radars, storage, and processing nodes. CASA depends on efficient discovery and utilization of heterogeneous, dynamic, and distributed resources that are characterized by multiple attributes. Therefore, its resource discovery and scheduling algorithms must take into account both the dynamic and static attributes to ensure that data is generated, processed, and delivered to end users within 30 seconds. Community cloud computing aggregates residual computing resources in Internet end hosts to build virtual cloud systems. Given that such systems rely on residual resources, respective job schedulers must take into account the dynamic attributes of hosts while scheduling latency sensitive cloud-based applications (e.g., collaboration tools, multimedia applications, scientific algorithms, etc.) to enhance both QoS and QoE. Therefore, understanding characteristics and modeling of large-scale P2P computing platforms and nodes are essential to correctly design, validate, and analyze the performance of resource discovery solutions, job schedulers, and distributed applications.

Formal characterization of nodes and queries has received attention only recently [4-5]. Characteristics of static attributes of nodes from SETI@home, one of the BOINC deployments, are presented in [4]. In [5], we present both the static and dynamic resource and query characteristics of PlanetLab [6] and SETI@home nodes. It was observed that attribute values are correlated [4-5], skewed [5], follow a mixture of probability distributions [4-5], and some of the dynamic attributes change frequently [5]. Based on the analysis of static attributes, [4] builds a forecasting model for Internet hosts while taking into account their marginal distributions, linear correlation, and time evolution of attributes (e.g., how does ratio between single core to multi-core processors changes with time). While static attributes are useful in evaluating systems such as BOINC, they are insufficient for evaluating latency sensitive systems that are affected by dynamic attributes and their temporal changes. Several other attempts to model computing resources are presented in [7-9]; however, they do not capture dynamic attributes and are applicable only in homogeneous and well-controlled environments such as clusters and grids. In the absence of large datasets and tools to generate synthetic datasets, existing performance studies have either neglected the dynamic attributes [10] or relied on simplifying assumptions such as random attribute values or replication of small data samples

This research is supported in part by the Engineering Research Center program of the National Science Foundation under award number 0313747.

[11]. Therefore, it is important to develop new algorithms and tools for generating large synthetic datasets while preserving the statistical properties of attributes of real-life systems.

To evaluate applications and protocols for scalability beyond what is available, it becomes necessary to consider node configurations with higher number of nodes and attributes. Yet, it is still necessary to adhere to statistical characteristics, dependencies, and temporal patterns exhibited by real-world systems. It is impractical to gather traces with sufficient resolution and duration even for existing systems. Therefore, our idea is to gather representative information about the traces and generate synthetic trace arrays of larger dimensionality in number and time, to meet the required goals.

We present a novel mechanism to generate random nodes with both static and dynamic attributes that are useful in evaluating the performance of large-scale P2P resource discovery schemes and job schedules. The presented methodology is applicable to any multivariate resource dataset, and PlanetLab node traces are utilized as an example. First, a multi-attribute resource model is defined using a selected set of static and dynamic attributes that are essential to characterize a node. Second, characteristics of nodes are presented. Our findings show that attribute values are skewed, follow a mixture of probability distributions, complex correlation patterns among attributes, and time series of dynamic attributes are nonstationary. These characteristics make it nontrivial to generate random nodes with multiple attributes. Third, vectors of static attributes are generated using empirical copulas that capture the entire dependence structure of multivariate distribution of attributes. Finally, time series of dynamic attributes are randomly drawn from a library of multivariate-time-series segments extracted from PlanetLab traces. These segments are determined by identifying the changes in the regression coefficients of time series corresponding to a selected attribute. Time series corresponding to rest of the attributes are split at the same breakpoints and randomly drawn together to preserve their contemporaneous correlation. Furthermore, a tool is developed to automate the synthetic data generation process and its output is validated using statistical tests. The tool generates n random nodes with a_s static and a_d dynamic attributes. Dynamic attribute values can be generated up to a given time t (ranging from several hours to weeks) with sampling interval s.

Section II presents the node model and node characteristics. Static attribute generation is presented in Section III while dynamic attribute generation is presented in Section IV. Section V presents the design of the tool that generates synthetic data and its validation. Concluding remarks are presented in Section VI.

II. Node Model and Characteristics

PlanetLab [6] is a global research network that supports the development of new network services and applications. It provides a versatile platform for users to run their distributed applications, protocols, etc., by aggregating a globally distributed set of heterogeneous nodes. Node characteristics are analyzed using data from CoMon [12] which is a node and slice monitoring system for PlanetLab. CoMon tracks nodes using 46 attributes (12 static and 34 dynamic) that are useful in discovering resources that match user requirements. These attribute values are measured every five minutes. CoMon keeps track of ~1,000 nodes and 500-700 nodes are typically active at any given time instance. Though SETI@home has more than 300,000 active hosts [5], BOINC tracks only four dynamic attributes. Moreover, their temporal changes are not tracked. PlanetLab reflects many characteristics of Internet-based distributed systems such as heterogeneity, multiple end users, dynamic nodes, and global presence, and hence is being used to evaluate many preliminary P2P protocols and applications. Therefore, PlanetLab dataset is more appropriate for capturing the temporal behavior of dynamic attributes. Not all attributes tracked by CoMon are equally important in describing a node. Hence, our analysis focuses only on a selected subset of attributes that are essential to characterize a typical node useful for collaborative P2P and volunteer computing systems. Next, characteristics of those attributes are analyzed using CoMon datasets collected between 2010/11/01-15.

A. Node Model

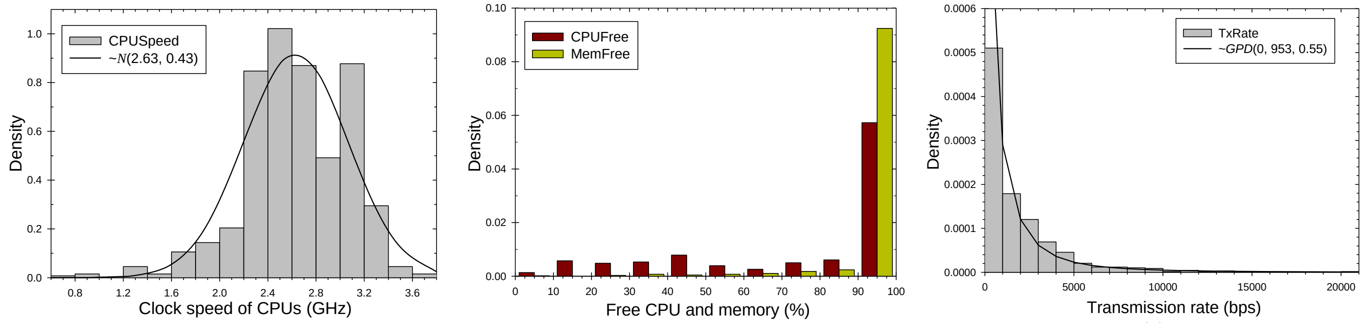
Following nine attributes are selected to describe a node:

1. *CPUSpeed* – Processor clock speed in GHz. Provides insight on relative computing power of a node.
2. *NumCores* – Number of processor cores. Indicates how much parallelism in processing is possible.
3. *CPUFree* – (100 – *CPU utilization*)%. Indicates to what extent the CPU(s) is available for processing. If multiple cores are available, the average value is given.
4. *1MinLoad* – One minute exponentially weighted moving average of number of active processes competing or waiting for CPU. Indicates how long a user process has to wait. Both CPUFree and 1MinLoad are complementary to each other as a large CPU load does not necessarily mean high CPU utilization (e.g., processes could be blocked for I/O).
5. *MemSize* – Size of volatile memory in GB.
6. *MemFree* – Free user-level memory as a percentage. Indicates how much memory is available for user processes.
7. *DiskFree* – Free disk space in GB.
8. *TxRate* – Average transmission rate in bps. In conjunction with bandwidth limit specified by most nodes, it provides insight on amount of available bandwidth.
9. *RxRate* – Average receive rate in bps.

CPUSpeed, NumCores, and MemSize are static attributes (number of static attributes $a_s = 3$) while the rest are dynamic (number of dynamic attributes $a_d = 6$). Though rest of the discussion is based on these attributes, analysis and the proposed methodology is applicable to other systems, e.g., SETI@home [4]. Resource discovery solutions and scheduling algorithms for latency sensitive applications are typically interested only in short-term trends. Therefore, we capture statistical characteristics that are valid for several minutes to few weeks.

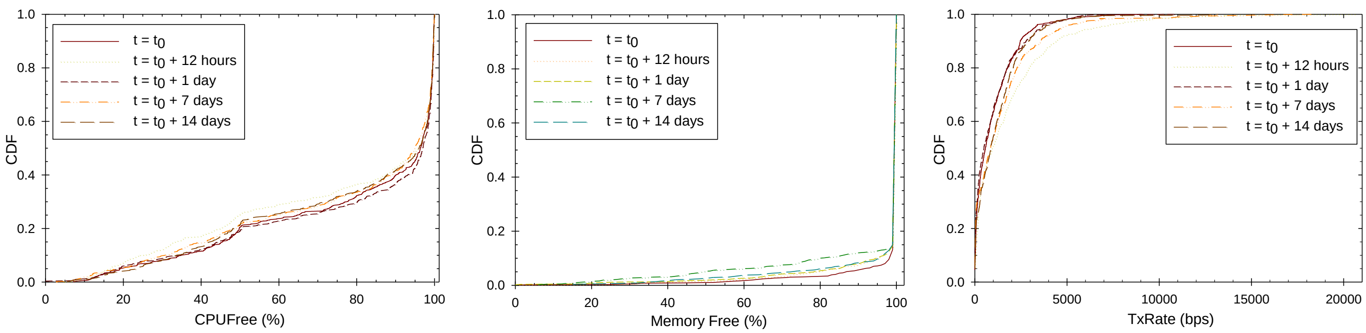
B. PlanetLab Node Characteristics

Fig. 1 illustrates the distribution of different attribute values of PlanetLab nodes. While ~*Normal*(2.63, 0.43) captures the distribution of CPUSpeed rest of the attributes are highly skewed. We further observed that MemSize and DiskFree also approximate Gaussian-like distributions. However, few nodes with very high memory and disk space prevented a good fit. Instantaneous values of both TxRate and RxRate of nodes fit a Generalized Pareto Distribution (GPD) with ~*GPD*(0, 953, 0.55) and ~*GPD*(0, 1055.7, 0.34), respectively.



(a) (b) (c)

Figure 1. Distribution of attribute values of PlanetLab nodes as of 2010/11/01 16:00 UTC.



(a) (b) (c)

Figure 2. Cumulative distribution of dynamic attributes sampled at different times. Starting time t_0 = 2010/11/01 4:00 UTC.

Fig. 2 shows the cumulative distribution (CDF) of three dynamic attributes at different time instances. It can be seen that CDFs for different samples (taken at different times relative to a given starting time t_0) are somewhat similar. Thus, distributions derived for a particular sample remains valid for several weeks. Our goal is to generate nodes with similar overall characteristics to evaluate impact of dynamic attributes over a moderate time span ranging from several minutes to few weeks. Therefore, long-term trends are not considered. If desired, an analysis similar to [4] can be used to determine such trends using CoMon data archives. When dynamic attributes change rapidly, resource discovery schemes and schedules have to rely on many update messages to represent them correctly. Thus, number of changes in dynamic attributes over a 24-hour period is observed for seven consecutive days starting from 2010/11/01. Fig. 3 shows the CDF of number of changes for selected set of attribute values. A fixed threshold is applied to ignore minor variations. Changes in dynamic attributes such as DiskFree and MemFree were less significant. During 24-hours, 89% of the nodes had less than 50 changes in TxRate and RxRate. 66% and 72% of the nodes had less than 100 changes in CPUFree and 1MinLoad, respectively. 5% of the nodes changed their attribute values over 200 times (out of 288 samples). Such rapid changes arise either due to the variability in applications' resource usage or execution of small jobs. Thus, there is a wide variation in how frequently the attribute values change. It was realized that the rate of change in dynamic attribute values can be approximated using a GPD [5].

Though distribution of instantaneous values of dynamic attributes are stable over moderate periods (Fig. 2) and GPD can capture the number of changes within a given time period, static and time varying dynamic attribute values cannot be drawn

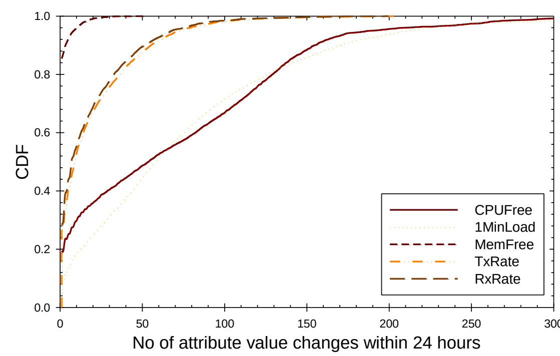


Figure 3. Cumulative distribution of number of significant changes in attribute values within 24-hours. Thresholds: *CPUFree* = *MemFree* = ± 10%, *1MinLoad* = ± 2, *TxRate* = *RxRate* = ± 1 Kbps.

randomly from those respective distributions. This is due to the correlations that exist among attributes, as well as the specific structure in time series. Table 1(a) shows the Pearson's correlation coefficient (i.e., linear association) among attributes sampled at a specific time instance. It can be seen that attribute pairs (NumCores, CPUFree), (MemSize, DiskFree), and (TxRate, RxRate) are positively correlated. Analysis of correlation among static attributes in [4] was limited to Pearson's correlation. However, further analysis of Spearman's ranked correlation coefficient ρ (see Table 1(b)) indicates higher correlation among many attributes. Spearman's ρ measures how well the correlation between two attributes can be described using a monotonic function. Correlation between (NumCores, DiskFree), (MemSize, MemFree), and (CPUFree, RxRate) have increased. Moreover, (CPUFree, 1MinLoad) pair is negatively correlated as CPU is typically not free while load is high. Also note the correlation between static-static, static-dynamic,

TABLE I. CORRELATION AMONG ATTRIBUTES (2010/11/01 14:00 UTC).

(a) – Pearson's correlation coefficient.

	CPUSpeed	NumCores	CPUFree	1MinLoad	MemSize	MemFree	DiskFree	TxRate
NumCores	-0.09							
CPUFree	0.02	0.48						
1MinLoad	0.03	-0.31	-0.57					
MemSize	0.06	0.28	0.26	-0.25				
MemFree	0.13	0.21	0.31	-0.35	0.25			
DiskFree	-0.09	0.46	0.37	-0.29	0.54	0.23		
TxRate	0.08	-0.23	-0.26	0.24	-0.12	-0.17	-0.12	
RxRate	0.10	-0.23	-0.30	0.35	-0.13	-0.20	-0.16	0.85

(b) – Spearman's ranked correlation coefficient.

	CPUSpeed	NumCores	CPUFree	1MinLoad	MemSize	MemFree	DiskFree	TxRate
NumCores	0.04							
CPUFree	-0.07	0.67						
1MinLoad	0.10	-0.42	-0.72					
MemSize	0.03	0.37	0.37	-0.33				
MemFree	-0.07	0.37	0.37	-0.38	0.53			
DiskFree	-0.20	0.60	0.52	-0.41	0.44	0.44		
TxRate	0.06	-0.35	-0.39	0.30	-0.07	-0.20	-0.29	
RxRate	0.07	-0.33	-0.42	0.41	-0.11	-0.21	-0.29	0.86

and dynamic-dynamic attributes. These findings suggest that a complex correlation exists among attributes that is not captured by linear correlation.

Fig. 4 shows example time series corresponding to dynamic attributes of a selected node. It can be seen that attribute values tend to change together indicating contemporaneous correlation, e.g., CPUFree reduces while 1MinLoad increases, and TxRate and RxRate change together. This behavior is called *contemporaneous correlation* [13] where observations of one time series are correlated with the observations of another time series during the same time interval. Note the distinct pattern in MemFree time series and its structural changes. Such temporal patterns need to be preserved to accurately represent the behavior of a node. Some of the time series were nonstationary, e.g., 29.3% of the time series corresponding to MemFree had at least one significant structural change within 7 days.

III. RANDOM VECTORS OF STATIC ATTRIBUTES

Because of the strong correlation among some of the attributes as well as specific structures in time series, attribute values of random nodes cannot be drawn from independent distributions. Therefore, we have to rely on the joint distribution of attributes. Static and dynamic attributes are handled separately as the time series of dynamic attributes are nonstationary and have specific temporal structures, as exemplified by Fig. 4.

As the correlation among attributes is nonlinear and complex, it is insufficient to use the matrix of Pearson's correlation coefficients to establish the dependence among random variables. Alternatively, copulas [14] can be used to capture the entire dependence structure of multivariate distributions. Copulas are functions that couple multivariate distribution functions to their marginal distributions. A copula $C(u)$ is a multivariate joint distribution defined on the d-dimensional unit cube $[0, 1]^d$, $(u_1, \ldots, u_d) \in [0, 1]^d$, such that every marginal distribution u_i is uniform on the interval [0, 1]. Let F denote the d-dimensional distribution function (CDF) with marginals $F_1, \ldots, F_d$ then a copula C exists such that for all real $u = (u_1, \ldots, u_d)$:

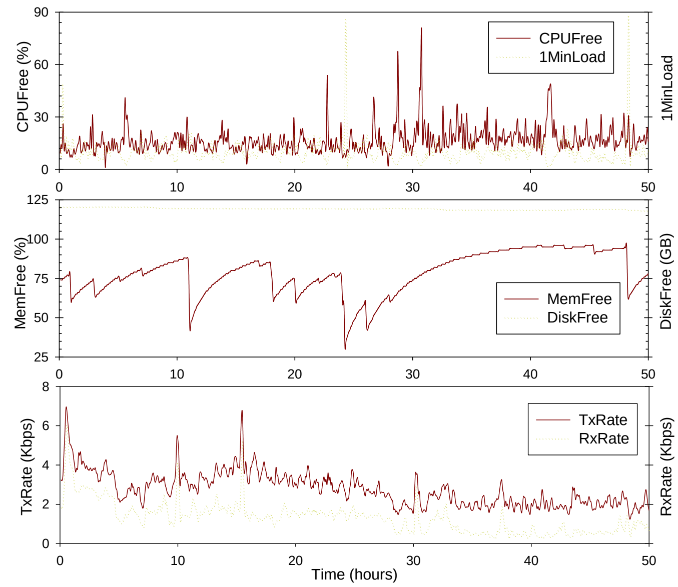


Figure 4. Time series of dynamic attributes.

$$F(u) = C\big(F_1(u_1), \ldots, F_d(u_d)\big) \tag{1}$$

Several well-known copula families are available, e.g., Gaussian and Archimedean copulas. However, these copulas tend to be symmetric along the axis of correlation. Alternatively, empirical copulas are useful while analyzing data with complex and/or unknown underlying distributions. Empirical copula also supports any number of dimensions, and its bivariate function is given by:

$$C_n\left(\frac{i}{n}, \frac{j}{n}\right) = \frac{\text{No of pairs } (x, y) \text{ s.t. } x \le x_{(i)} \text{ and } y \le y_{(j)}}{n} \tag{2}$$

where $1 \le i, j \le n$, $x_{(i)}$ is the ordered statistics of x, and n is the number of data points. It is proven that empirical copula converges uniformly to the underlying copula. After deriving the copula, dependent random numbers can be generated. Those numbers can then be transformed into original marginal distributions using inverse transforms.

We use empirical copulas to generate vectors of static attributes as the joint distribution is unknown. First, all the PlanetLab nodes active at a given time instance are sampled for their static attributes. Marginal distribution of each attribute is then transformed to a uniform random variable $\sim U(0, 1)$, e.g., using Kernel smoothing density estimation. Third, empirical copula is calculated using multivariate version of Eq. (2). Fourth, dependent random numbers are then generated from the multivariate copula. Finally, random numbers are transformed back to desired marginal distributions using inverse transformation techniques, e.g., using estimated empirical distribution functions. If an attribute value is continuous, then linear interpolation can be used to generate in-between values while performing inverse transformation. Empirical cumulative distribution function (e.g., Kaplan-Meier estimator) can be used for discrete valued attributes. We generate CPUSpeed and MemSize values using linear interpolation and NumCores are generated using the empirical distribution. Fig. 5 shows the actual and generated random data obtained using *pwlCopula* [14] tool. The generated random values closely match the actual data. We will statistically quantify the similarity between actual and generated attributes in Section V. If only the instantaneous values of dynamic attributes are of interest, empirical copula can be simultaneously applied for both static and dynamic attributes. Moreover, we do not need to fit the attribute values to a specific distribution(s) as the random vector generation process is based only on empirical data and distributions.

IV. Generating Dynamic Attributes

Time varying dynamic attribute values cannot be drawn randomly from marginal distributions due to the contemporaneous correlation and specific structure (Fig. 4) in time series. Failing to capture such behavior could result in over or under estimating the number of changes in attribute values over a given period and traces not relevant to practical systems (e.g., higher FreeCPU values are typically associated with lower 1MinLoad). Therefore, one time series cannot be generated independently from rest of the attributes. Furthermore, many structural changes in these multivariate time series (Fig. 4) make it nontrivial to model them using regression. Though it may be possible to fit a model for piecewise stationary time series, such an approach provides only a minor enhancement as our goal is not to predict the future behavior of nodes but to generate nodes with similar overall characteristics. Moreover, such a model will not be valid over long time durations. Instead, it is better to build a library of time-series segments corresponding to distinct temporal patterns. This is sufficient as our goal is to preserve the temporal variation of an attribute and its contemporaneous correlation.

We selected MemFree as the attribute based on which to partition the time series because it has the most distinguishable pattern (Fig. 4). Consider the standard linear regression model:

$$y_i = x_i^T \beta_i + u_i \quad i = 1, \ldots, n \tag{3}$$

where at time i, y_i is the dependent variable, x_i is the vector of regressors, β_i is the vector of regression coefficients, and u_i is an i.i.d. error term. We determine the structural changes by testing the null hypothesis that regression coefficients remain constant (i.e., H_0: $\beta_i = \beta_0$, i = 1, …, n). Each time series of a node is split using the *strucchange* package [15], which checks H_0 to determine the optimal number of structural breakpoints using a dynamic programming algorithm. Fig. 6 illustrates the breakpoints obtained for the MemFree time series. Time series corresponding to rest of the attributes are split at the same breakpoints. Multivariate-time-series segments are then added to the library. If desired, stationary time series can also be split after a specific duration to increase the number of segments in the library. However, one needs to be careful not to introduce unnecessary variability by splitting the time series after a short duration as time series are concatenated randomly during the time series generation. For generating dynamic attribute values, multivariate-time-series segments are drawn randomly from the library. Longer sequences are generated by concatenating one randomly drawn segment with another. Breaking all the time series of a node at the same point and replaying them together preserve the contemporaneous correlation among attributes. As

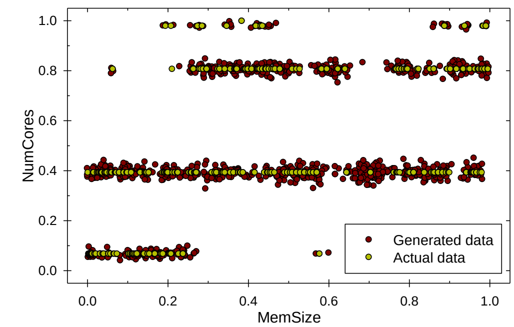


Figure 5. Random data generated from copula (data in unit scale).

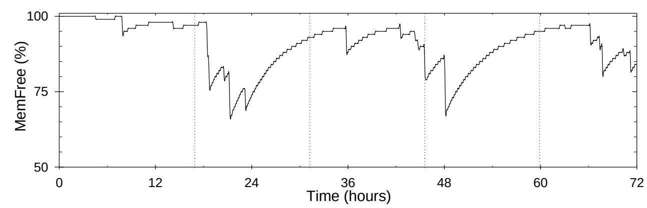


Figure 6. Breakpoints identified for MemFree time series.

the static and dynamic attributes are correlated, it is essential to establish the dependency between them. For example, a node with large NumCores typically has higher CPUFree values (Table 1). Therefore, time-series segments in the library are grouped according to the NumCores of the corresponding node. Consequently, given the NumCores generated from empirical copula, the dependency between static and dynamic attributes can be established by randomly drawing time-series segments from the corresponding group of segments. This is sufficient to establish the correlation as correlation between CPUSpeed and MemFree is not strong (Table 1).

V. Tool for Random Node Generation

A tool has been built to automate the synthetic data generation process, by combining the empirical-copula-based static attribute generation and time-series-library-based dynamic attribute generation. Fig. 7 is a flowchart illustrating the technique. It can generate synthetic traces corresponding to a set of nodes, e.g., n random nodes with a_s static and a_d dynamic attributes. As the distribution of dynamic attributes is stable over few weeks, the technique can be used to generate data from

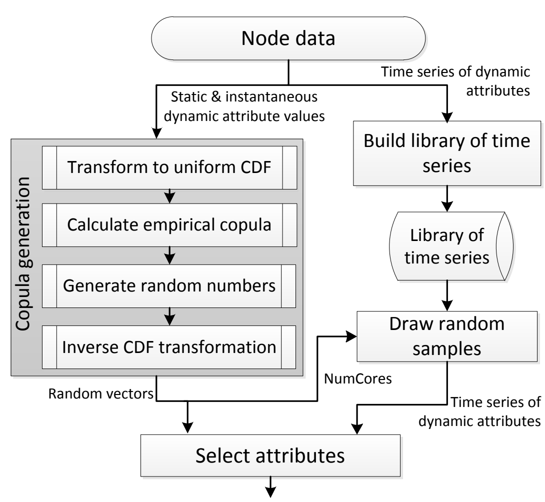


Figure 7. Flowchart of random node generation tool.

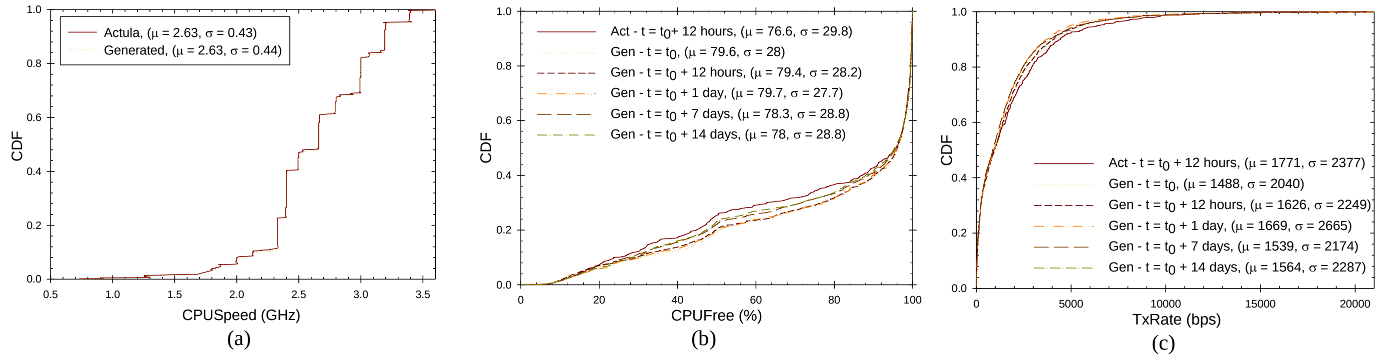


Figure 8. Comparision of actual nodes and nodes generated by the tool. t_0 = 2010/11/01 4:00 UTC. *Act* – Actual, *Gen* – Generated.

several minutes to few weeks. Instantaneous values of dynamic attributes are also fed to the copula generator to generate random vectors with instantaneous dynamic attributes that may be useful in evaluating scheduling algorithms. NumCores from copula is fed to *draw random samples* module to establish the dependence between static and dynamic attributes. If desired, a user may use only a subset of the attributes supported by the tool. Several additional attributes (e.g., 5MinLoad, DiskSize, and Location) are included in the MATLAB-based tool that is downloadable from [16].

Statistical properties of synthetic data generated by the tool are validated as follows. CoMon data for a week, starting from Nov 1, 2010 4:00 UTC (same as in Section 2), is used as the input to the tool. Static attributes are sampled on Nov 1st at 16:00 UTC and dynamic attributes are extracted from 300 nodes that were active during the entire week. Tool is then used to generate 5,000 random nodes with static and dynamic attributes over a two weeks period. Stationary time series are split after 12 hours to create more segments, and it did not significantly vary the distribution of number of attribute changes within 24-hours. We compare synthetic data with actual data as there are no other comparable models that capture the correlation among dynamic attributes. Fig. 8 plots the distribution of both the actual and generated attributes. It can be seen that the generated attributes closely match the distributions observed in Section 2. Mean (μ) and standard deviation (σ) of CPUSpeed (included in figure) derived using copula is identical to actual data. Even for the CPUFree and TxRate error in μ and σ is 3-16% which is expected as the distribution of time series vary among samples (see Fig. 2). Kolmogorov-Smirnov test (KS-test) with a significance level of 0.05 further confirmed that synthetic data satisfy the distributions of original data. In addition to meeting μ and σ, synthetic traces also mimics the true variations/patterns inherent in time series, e.g., Fig 6. These findings indicate our approach can generate static and dynamic attributes of nodes while preserving their statistical properties.

VI. Conclusions and Future Work

A technique is presented to generate vectors of static attributes and multivariate time series of dynamic attributes while preserving correlations observed in operational systems. Such attributes are useful in collaborative P2P and cloud computing for evaluating scalability of applications, resource discovery solutions, and job schedulers, far beyond that is possible with existing test beds. Data from any other platform may be used as the basis for trace statistics. Synthetic data from the tool are being used to evaluate the performance of P2P resource discovery solutions. Future collaborative P2P systems will include multitude of heterogeneous resources such as special hardware, sensors/actuators, middleware, and algorithms. In future, proposed tool will be extended to support such diverse set of resources, resource failures, and new resource datasets.

References

[1] D. P. Anderson and K. Reed, "Celebrating diversity in volunteer computing," In Proc. Hawaii Int. Conf. on System Sciences, Jan. 2009.

[2] D. McLaughlin et al., "Short-wavelength technology and the potential for distributed networks of small radar systems," Bull. Amer. Meteor. Soc., vol. 90, Dec. 2009, pp. 1797–1817.

[3] G. Briscoe and A. Marinos, "Digital ecosystems in the clouds: towards community cloud computing," In Proc. 3rd Int. Conf. on Digital Ecosystems and Technologies, June 2009, pp. 103–108.

[4] E. Heien, D. Kondo, and D. P. Anderson. "Correlated resource models of Internet end hosts," In Proc. 31st Int. Conf. on Distributed Computing Systems (ICDCS '11), June 2011.

[5] H. M. N. D. Bandara and A. P. Jayasumana, "Characteristics of multi-attribute resources/queries and implications on P2P resource discovery," In Proc. 9th IEEE Int. Conf. on Computer Systems and Applications, Dec. 2011.

[6] A. Bavier et al,, "Operating system support for planetary-scale network services," In Proc. 1st Conf. on Networked Systems Design and Implementation, May 2004.

[7] Y. S. Kee, H. Casanova, and A. Chien, "Realistic modeling and synthesis of resources for computational grids," In Proc. ACM/IEEE Conf. on Supercomputing, Nov. 2004.

[8] A. Sulistio et al., "A toolkit for modelling and simulating data grids: an extension to gridsim," Concurrency and Computation: Practice and Experience, vol. 20, no. 13, Sep. 2008.

[9] D. Lu and P. A. Dinda, "Synthesizing realistic computational grids," In Proc. ACM/IEEE Conf. on Supercomputing, Nov. 2003.

[10] H. Shen, A. Apon, and C. Xu, "LORM: Supporting low-overhead P2P-based range-query and multi-attribute resource management in grids," In Proc. 13th Int. Conf. on Parallel and Distributed Systems, Dec. 2007.

[11] M. Cai et al., "MAAN: A multi-attribute addressable network for grid information services," J. Grid Comput., Jan. 2004.

[12] K. Park and V. S. Pai, "CoMon: A mostly-scalable monitoring system for PlanetLab," In Proc. ACM SIGOPS OSR, vol. 40, no. 1, Jan. 2006.

[13] J. L. Worrall and T. C. Pratt, "Estimation issues associated with time-series - cross-section analysis in criminology," Western Criminology Review, vol. 5, no. 1, 2004, pp. 35–49.

[14] J. C. Strelen, "Tools for dependent simulation input with copulas," In Proc. 2nd Int. Conf. on Simulation Tools and Techniques, Mar. 2009.

[15] A. Zeileisa et al., "Testing and dating of structural changes in practice," J. Comput. Stat. and Data Anal., vol. 44, no. 1-2, Oct. 2003, pp 109–123.

[16] CMPRG – Correlated Multi-attribute P2P Resource Generator, available: http://www.cnrl.colostate.edu/Projects/CP2P/