Characterizing Heterogeneous Computing Environments using Singular Value Decomposition

Abdulla M. Al-Qawasmeh[1], Anthony A. Maciejewski[1], and Howard Jay Siegel[1,2]
[1]Department of Electrical and Computer Engineering
[2]Department of Computer Science
Colorado State University
Fort Collins, Colorado, USA
{Abdulla.Al-Qawasmeh, aam, hj}@colostate.edu

***Abstract*— We consider a heterogeneous computing environment that consists of a collection of machines and task types. The machines vary in capabilities and different task types are better suited to specific machine architectures. We describe some of the difficulties with the current measures that are used to characterize heterogeneous computing environments and propose two new measures. These measures relate to the aggregate machine performance (relative to the given task types) and the degree of affinity that specific task types have to different machines. The latter measure of task-machine affinity is quantified using singular value decomposition. One motivation for using these new measures is to be able to represent a wider range of heterogeneous environments than is possible with previous techniques. An important application of studying the heterogeneity of heterogeneous systems is predicting the performance of different computing hardware for a given task type mix.**

Keywords- singular value decomposition; heterogeneity; heterogeneous systems

I. INTRODUCTION

The heterogeneity of a system depends on the hardware available to execute a workload and the workload itself [1]. An important research problem in the field of heterogeneous computing is how one can characterize and quantify the heterogeneity of a system [8]. We treat heterogeneous computing (**HC**) systems as those that consist of a collection of machines able to perform task types at different speeds. Different task types may be better suited to specific machine architectures. It is a common practice to arrange the estimated time to compute (**ETC**) of task types on machines in a matrix called an ETC matrix (e.g., [4, 9, 11, 14, 16, 17, 19]). Entry ETC(*i,j*) in the ETC matrix is the ETC of task type *i* on machine *j*.

In previous work (e.g., [2, 3, 5, 15, 18]), machine heterogeneity, task heterogeneity, machine inconsistency, and task inconsistency were either used to describe an HC system or to generate ETC matrices that represent an HC system in simulation studies. Although some of the previous work is concerned with generating ETC matrices with specific values of heterogeneity and consistency, the ETC generation procedure can be reversed to calculate the heterogeneity of an existing ETC matrix. Machine heterogeneity in [3] is the average coefficient of variation (COV) of each row in the ETC matrix. Similarly, task heterogeneity in [3] is the average COV of each column of the ETC matrix. Machine and task heterogeneities in other research are calculated in a similar manner using alternative measures, e.g., skewness [2], kurtosis [2], and the range of the time values [5]. For a **machine consistent** ETC matrix, if machine m_1 is faster than machine m_2 for a given task type, then m_1 must be faster than m_2 for all other task types [5]. Similarly, for a **task consistent** matrix, if task type t_1 runs faster than task type t_2 on a given machine, then t_1 must run faster than t_2 on all machines [18].

The measures used in previous work present one way of describing heterogeneity; however, they may not be appropriate for all situations for the following two reasons. First, the measure of inconsistency is Boolean, i.e., an ETC matrix can either be consistent or inconsistent. The concept of a partially-consistent ETC matrix also was used in previous work [5]; however, a partially-consistent ETC matrix is one that contains a consistent sub-matrix. Thus, the inconsistency measure is still Boolean: consistent for the sub-matrix and inconsistent for the remainder of the matrix. The problem with this measure being Boolean is that we may have two different machine inconsistent ETC matrices where one has only one entry that makes it inconsistent and the other has many entries causing the inconsistency. Those two ETC matrices correspond to HC systems with different characteristics; however, they have the same measure of inconsistency. One goal of this research is to introduce a continuous measure that expresses different levels of inconsistency.

This research was supported by the NSF under grant numbers CNS-0615170 and CNS-0905399, and by the Colorado State University George T. Abell Endowment.

978-1-4244-6534-7/10/$26.00 ©2010 IEEE

The second potential drawback with previous measures is that machine heterogeneity is calculated by computing the variation along rows in the ETC matrix instead of computing the variation of the columns. For example, if machine heterogeneity for the ETC matrix in Figure 1(a) is calculated using most of the methods in the previous work, then the variation of each row in the matrix will be computed first. Because the variation of the values (10, 1) is high, the matrix will have high machine heterogeneity. However, the performance of the two machines is similar in the sense that if either machine was used alone to execute one instance of both task types, both machines would finish at the same time. Thus, intuitively, machines m_1 and m_2 have similar performance with respect to this workload. However, each machine is better suited for a different task type (i.e., the machines are homogeneous in terms of aggregate performance and heterogeneous in the sense that each one is better suited to a different task type). Thus, another goal of this research is to define two measures that match this intuition.

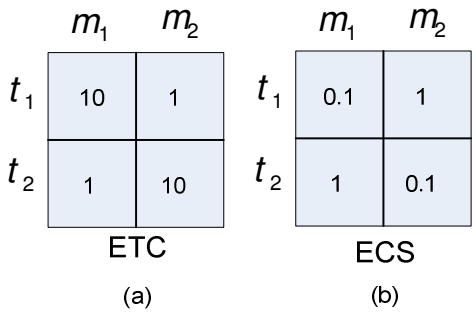

Figure 1. (a) An ETC matrix for which earlier measures will give high machine heterogeneity. (b) An ECS matrix that corresponds to the ETC matrix in (a).

One motivation for using the new measures is to be able to simulate a wider range of heterogeneous environments than is currently possible. In addition, an important application of studying the heterogeneity of heterogeneous systems is predicting the performance of different distributed computing hardware for a given task type.

In summary, the contributions of this paper are: (a) the definition of two new measures to describe an HC system – Machine Performance Homogeneity (MPH) and Task-Machine Affinity (TMA), (b) a demonstration of how singular value decomposition (SVD) can be used to quantify TMA, and (c) an example of how previous measures do not represent the entire range of HC systems that can be described by these two measures.

The rest of this paper is organized as follows. Section II explains singular value decomposition. The proposed heterogeneity measures are given in Section III. Section IV compares the new measures with the COV method. Section V discusses future work. Conclusions are given in Section VI.

II. Singular Value Decomposition

Singular value decomposition (**SVD**) is a standard matrix decomposition that can be performed on any arbitrary matrix [12] that is used widely for determining the rank or condition number of a matrix, computing the best low-rank matrix approximation, and solving a host of pattern recognition problems for a wide range of applications. For an m-by-n real matrix A, there exists a factorization in the form:

$$A = U \sum V^T, \tag{1}$$

where U is an m-by-m orthogonal matrix, Σ is an m-by-n diagonal matrix with non-negative entries in descending order, V is an n-by-n orthogonal matrix, and ^{T} denotes the transpose. The diagonal values of the matrix Σ are the singular values, denoted σ_i, such that

$$\sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n \geq 0 .$$

The magnitudes of the singular values represent the degree of linear dependence among the columns (or rows) of a matrix. The number of singular values is equal to min(m,n). For our application, it is more common to have more task types than machines so that the number of singular values will be equal to n, the number of columns (machines). The SVD of a matrix with $m \geq n$ can be calculated in $O(mn^2)$ operations. Section III.C illustrates how the SVD is used to calculate the TMA. For the purposes of this paper, $m = T$, is the number of task types and $n = M$, is the number of machines.

III. Heterogeneity Measures

A. Estimated Computation Speed Matrices

Large entries in an ETC matrix correspond to task types that take a long time to execute on particular machines. Consider the ETC matrix in Figure 1(a) as an example. Assume that we have only two tasks, task 1 (task type 1) and task 2 (task type 2). The best allocation of tasks to machines is to execute task 1 on machine 2 and task 2 on machine 1. In this case, the two machines will finish in one time unit and the execution times of task 1 on machine 1 and task 2 on machine 2 do not affect the finishing times of the machines. However, if we compute the SVD for an ETC matrix, the higher values will have the biggest effect on the singular values. Therefore, we define the estimated computation speed (**ECS**) matrix to be the matrix obtained by taking the reciprocal of each entry in the ETC matrix. Figure 1(b) shows an example of an ECS matrix. In the ECS matrix, higher values correspond to more powerful machines for a specific task type. The **canonical form** for the ECS matrix that we use in the calculation of the heterogeneities is an ECS matrix with the columns ordered so that the sums of the values (L1 norms) of each column are in ascending order.

B. Machine Performance Homogeneity (MPH)

The value of **MPH** (machine performance homogeneity) represents the performance homogeneity of machines over all task types. Here we are not concerned with the performance of a machine for a task type or a subset of task types; rather, we are concerned with the performance of a machine over all task types. For example, machines 1 and 2 in matrix (b) in Figure 1 show high performance difference for each task type individually. However, if we calculate the performance of the both machines for both task types, then both machines will be identical in performance.

For the sake of simplicity, assume that the rows of the ECS matrix represent a task instance (i.e., we need to execute every task type once) and that all task instances have the same importance. Then, one way to characterize the overall performance of a machine is the sum of the column entries. The MPH would then be related to the ratio of the column sums. For example, for a general 2-by-2 ECS matrix in canonical form (see Figure 2) MPH is given by:

$$\text{MPH} = \left(a_{1,1} + a_{2,1}\right) \Big/ \left(a_{1,2} + a_{2,2}\right). \quad (2)$$

	m_1	m_2
t_1	$a_{1,1}$	$a_{1,2}$
t_2	$a_{2,1}$	$a_{2,2}$

Figure 2. A 2-by-2 ECS matrix in canonical form, where $a_{1,1} + a_{2,1} \le a_{1,2} + a_{2,2}$.

In particular, the MPH for the ECS matrix shown in Figure 1(b) is 1; this is because the ratio of the column norms is 1. More examples of 2-by-2 ECS matrices and their MPH values are given in Section III.D.

In the case where rows of the ECS matrix are task types that can be executed more than once or that may have different importance, we add a weighting factor (w_{t_i}) for each row (task type) of the ECS matrix. Higher weighting factors correspond to task types that are executed more than others or for more important task types. Machines can also have weighting factors (w_{m_j}) that correspond to the machine preference (for example, some machines may be more preferable to execute tasks due to security reasons). The general formula to calculate MPH when T task types and M machines have weighting factors and when we have more than two columns (i.e., more than two machines) in the ECS matrix, is given by:

$$\text{MPH} = \frac{\sum_{j=1}^{M-1} \left(w_{m_j} \cdot \sum_{i=1}^{T} w_{t_i} \cdot a_{i,j} \Big/ w_{m_{j+1}} \cdot \sum_{i=1}^{T} w_{t_i} \cdot a_{i,j+1} \right)}{(M-1)}. \quad (3)$$

It is important to note that including weighting factors for an ECS matrix may require that the order of the columns be changed for an ECS matrix to be in canonical form, i.e., in ascending order of column norm. MPH corresponds to the average of the ratios of pairs of successively higher performance machines. For example, to calculate MPH of the canonical ECS matrix in Figure 3, we calculate the sums of the columns. The sums are (6, 8, 32). The ratios are (0.75, 0.25). MPH is the average ratio which is equal to 0.5.

	m_1	m_2	m_3
t_1	2	3	10
t_2	2	2	5
t_3	1	1	8
t_4	1	2	9

Figure 3. An ECS matrix used in the example to calculate MPH. For simplicity, all weighting factors are assumed to be 1.

C. Task-Machine Affinity (TMA)

To demonstrate how SVD can be used to analyze task-machine affinity, consider a very simple environment that consists of only two machines and two task types. If both of the machines are identical and can compute both task types equally fast, then the ECS matrix will be of the form shown in Figure 4(a). Clearly, this is a completely homogeneous environment and the singular values of this matrix will be 2 and 0. The 0 value for the smallest singular value indicates that the columns are linearly dependent and this would be true even if one column is a scalar multiple of the other one. Even if task 2 has a different execution time than task 1 but is the same on both machines, then the columns will stay linearly dependent and the ratio of the minimum singular value to the maximum one will be 0 (see Figure 4(b)). Analogously, if machine 2 is twice the speed of machine 1 on the two task types (see Figure 4(c)), then the two columns of the ECS matrix will be linearly dependent and the ratio of the minimum singular value to the maximum one will be 0. This would be true for an arbitrary number of task types where for any two task types the ratio of their execution time would be the same for the two machines.

Next, consider an opposite extreme case where we have a very heterogeneous environment in terms of task-machine affinity, where machine 1 can only execute task type 1 and machine 2 can only execute task type 2. The ECS for this environment is shown in Figure 4(d). The singular values for this matrix are both equal to 1 and so their ratio is also equal to 1. These two extreme cases (the ratio of the second singular value to the first one being 0 versus being 1) illustrate the motivation to use the ratio of the minimum singular value to the maximum one as a measure of the heterogeneity with respect to task types that the machines are required to execute (an example between the two extreme cases will be discussed later in this section). In other words, the more similar two machines are in their performance on a given task type mix (i.e., each machine's ratio of execution time for any pair of tasks is the same) the more linearly dependent the columns of the ECS matrix are and the smaller the ratio of the smallest singular value to the maximum one.

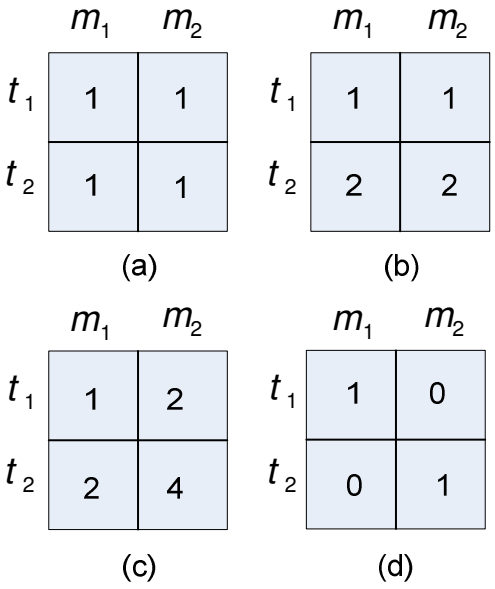

Figure 4. (a) An ECS matrix for a completely homogeneous system. (b) An ECS matrix for systems with homogeneous machines. (c) An ECS matrix with two different machines but with the same affinity for the two tasks. (d) An ECS matrix with an extreme degree of task machine affinity.

TMA (task-machine affinity) is the degree to which various sets of task types are better suited to run on different sets of machines. We would like this property of TMA to be as independent a measure as possible from the MPH. Therefore, we normalize the columns of the ECS matrix using the L1 norm before computing the TMA[†]. The normalized ECS will have an MPH equal to one, because all of the columns have a sum of one.

For a 2-by-2 ECS matrix, we define the TMA as the ratio of the second singular value to the first singular value of the normalized ECS matrix[‡]. Figure 5 shows a 2-by-2 ECS matrix and the corresponding normalized ECS matrix. The singular values obtained by computing the SVD of the normalized matrix are (1, 0.55) so that the TMA is equal to 0.55.

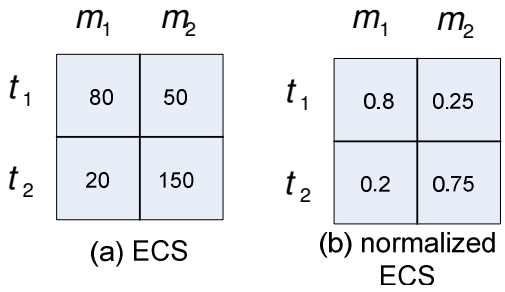

Figure 5. (a) A 2-by-2 ECS matrix used as an example to compute the TMA. (b) The corresponding normalized 2-by-2 ECS matrix for the matrix in (a).

When we have more than two machines (more than two columns in the ECS matrix), we also will have more than two singular values. Therefore, we generalize the procedure to calculate the TMA of an arbitrary *T*-by-*M* ECS matrix. First, we normalize the columns of the ECS matrix, and then we compute the SVD for the normalized ECS matrix. The TMA is then given by:

$$\text{TMA} = \left(\sum_{i=2}^{M} \sigma_i \Big/ (M-1) \right) \Big/ \sigma_1 . \qquad (4)$$

This corresponds to the average of the non-maximum singular values ($\sigma_i : 2 \le i \le M$) divided by the first (largest) singular value, i.e., σ_1. The maximum singular value provides a measure of the correlation (linear dependence) of the columns of the ECS matrix. The higher the maximum singular value the more the columns of the ECS matrix are correlated. Further, the higher the values of the non-maximum singular values, the less the columns of the ECS matrix are correlated. The less the columns of the ECS are correlated, the more each machine is suited to execute one set of task types compared to the other machines. Two matrices with the same TMA will have the same average correlation among their columns. Clearly, knowing all of the singular values (and their associated singular vectors) provides a more complete description of the affinity between sets of task types and sets of machines for a given ECS matrix. However, the TMA is proposed as a single value to

† The L2 norm, or any Lp norm can be used to normalize the columns. We used the L1 norm to remove the effect of MPH and thus make TMA independent of MPH.

‡ For the 2-by-2 case, the TMA is equivalent to the reciprocal of the condition number of the ECS matrix.

make it easier to compare and classify different ECS matrices. Analogous to the MPH, there are other ways of combining the singular values into a single number that may be more appropriate for different applications.

As an example of how to calculate the TMA of an ECS matrix, consider the matrix shown in Figure 6(a). One first normalizes the columns, with the resulting normalized ECS given in Figure 6(b). Computing the SVD for the normalized ECS matrix results in the following singular values (0.91, 0.38, 0.18), so that the TMA is given by:

$$\text{TMA} = ((0.38 + 0.18)/2)/0.91 = 0.31.$$

	m_1	m_2	m_3
t_1	2	1	4
t_2	1	3	7
t_3	3	4	1
t_4	1	1	2

(a) ECS

	m_1	m_2	m_3
t_1	0.29	0.11	0.29
t_2	0.14	0.33	0.5
t_3	0.43	0.45	0.07
t_4	0.14	0.11	0.14

(b) normalized ECS

Figure 6. (a) An ECS matrix used as an example for computing the TMA. (b) The corresponding normalized ECS matrix for the matrix in (a).

There are two extreme cases that correspond to maximum and minimum values of TMA. The first is when all the columns are multiples of each other; in this case when computing the SVD for the ECS matrix we will have only one non-zero singular value, i.e., σ_1. Therefore, the TMA will be 0, indicating that the ratio of the speed for any task type *a* to any task type *b* is the same for all machines. The second case is when the columns of the ECS matrix are orthogonal. In this case, the SVD of the ECS matrix will have all equal singular values and the TMA will be equal to 1. This corresponds to the case where task types are especially suited for specific machines. The two cases are depicted by matrices G and B in Figure 7, respectively. For two ECS matrices with the same MPH, the one with a larger TMA will, in general, result in a higher performance system.

Because of column normalization, TMA is not affected by having different weighting factors for different machines. Task type weights, however, do affect the value of TMA. Therefore, to calculate the TMA for an ECS that includes task types with different weights, one must multiply each row of the ECS matrix with its corresponding weight before computing the SVD.

D. Illustrative Examples of the MPH and TMA

In this section, we show examples of 2-by-2 ECS matrices and where they fall within the range of all possible values of the MPH and TMA measures. The example ECS matrices are shown in Figure 7.

The minimum limit for the MPH value of an ECS matrix is 0. This corresponds to the case where one machine performs significantly better than other ones in the HC system for all task types. For MPH to be exactly 0, we must have all machines except one not being able to execute any task type in the HC system. However, this is not a realistic case because we can simply remove such machine(s) from the HC system without affecting the performance. In addition, the TMA measure is not defined for an ECS matrix that has a MPH of 0 because a 0 column cannot be normalized.

IV. COMPARISON WITH THE COV METHOD

To illustrate one possible use of the MPH and TMA measures, we studied ETC matrices generated using the COV method [3] and compared their heterogeneities with the measures proposed in this paper. The COV method has been used by many researchers (e.g., [6, 7, 10, 13, 15]) to characterize the heterogeneity of systems for simulation studies. We have also used a method (described later in this section) to generate ETC matrices with varying MPH. Four different combinations of machine and task heterogeneity ETC matrices were generated using the COV method: (1) low machine, low task heterogeneity (L-L), (2) medium machine, medium task heterogeneity (M-M), (3) high machine, low task heterogeneity (H-L), and (4) high machine, high task (H-H) heterogeneity. COV values of 0.1, 0.5, and 0.9, correspond to low, medium, and high heterogeneity, respectively. Twenty ETC matrices were randomly generated for each combination. The COV generation method does not use weightings for machines or tasks, therefore, it is assumed that both the task types' and machines' weighting factors are 1.

Figure 8 shows the TMA and MPH values of ETC matrices that were generated. All L-L matrices had a low TMA value around 0.04, and a high MPH value around 0.95. M-M matrices had higher TMA values than L-L matrices.

M-M had high MPH (high MPH corresponds to lower machine performance heterogeneity). However, they should have medium MPH values because machine heterogeneity is medium. H-H matrices have high TMA and high MPH. However, they should have low MPH values because machine heterogeneity is high. H-L matrices have, on average, lower MPH than H-H matrices and low TMA. In the cases of M-M, H-H and H-L, MPH was high (i.e., the machines are not as heterogeneous as they should be). This shows that the COV method did not generate ETC matrices that have low MPH and that cover the entire range of possible heterogeneities. To study a wider range of HC systems that are represented by ETC and ECS matrices, it is important to generate matrices that span the entire range of possible heterogeneities. Any real world ECS matrix would be represented by a single point on the MPH and TMA plane.

To decrease the MPH of an ECS, we normalize the columns of the ECS matrix; hence, the sum of each of the columns will become one. Then, we select the desired column sums that will yield the desired MPH. Finally, we multiply each entry in each column with the changed sum. In Figure 9, for each of the M-M ETC matrices we have decreased its MPH to a random number between 0.4 and 0.6 and for each of the H-H ETC matrices we have decreased its MPH to a random number between 0.2 and 0.4. We can also increase or decrease the value of MPH for any ETC matrix and can cover the entire space of possible heterogeneities.

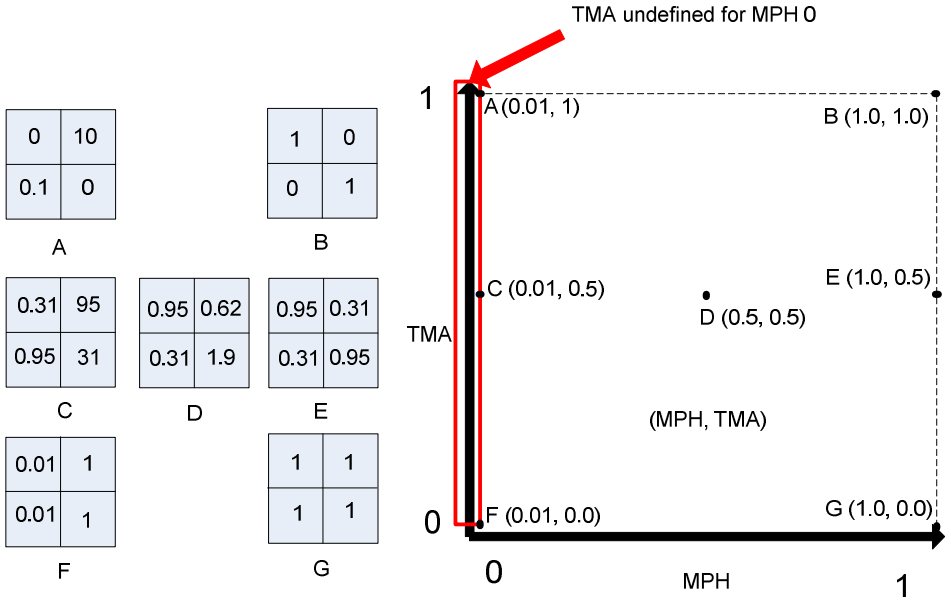

Figure 7. Shown are some example 2-by-2 ECS matrices. As we move vertically, we change TMA with the value of MPH fixed. For example, matrices A, C, and F all have the same MPH value (0.01), however, matrix F has the minimum possible TMA value (0) because its columns are multiples of each other, and matrix A has the maximum possible TMA value (1) because its columns are orthogonal to each other. As we move horizontally, we change MPH with the value of TMA fixed. For example, matrices A and B have the same TMA value (1), however, matrix B has the maximum possible MPH value because its machines have similar performance, and matrix A has a TMA value close to the minimum limit of 0. (Note that TMA is not defined for a matrix with MPH value of zero.)

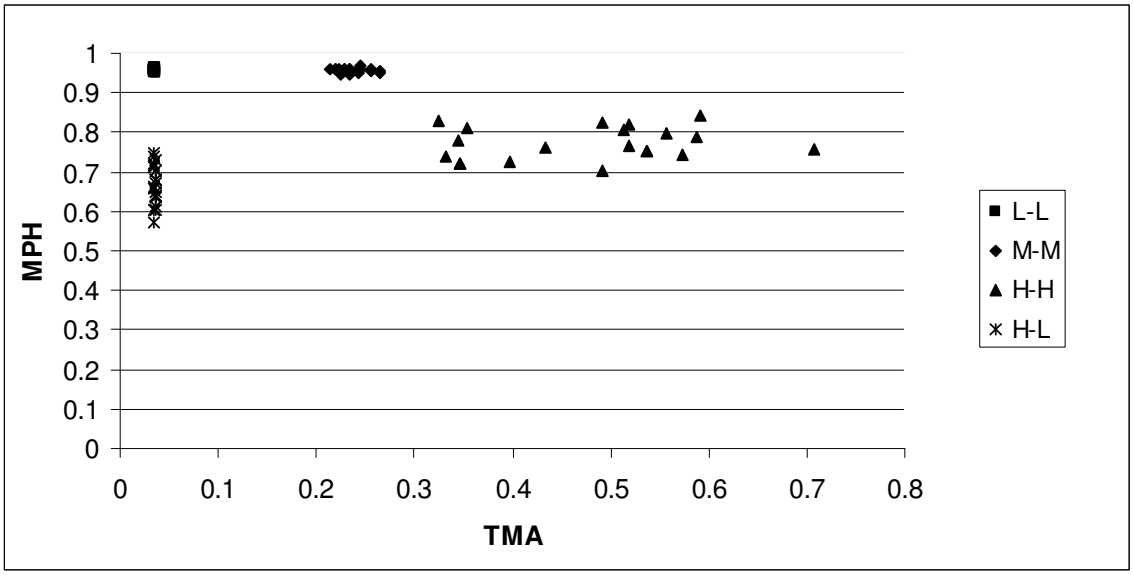

Figure 8. The TMA and MPH values of ETC matrices that were generated using the COV method. The L-L matrices have very close TMA and MPH values that is why they appear as one point in the plot.

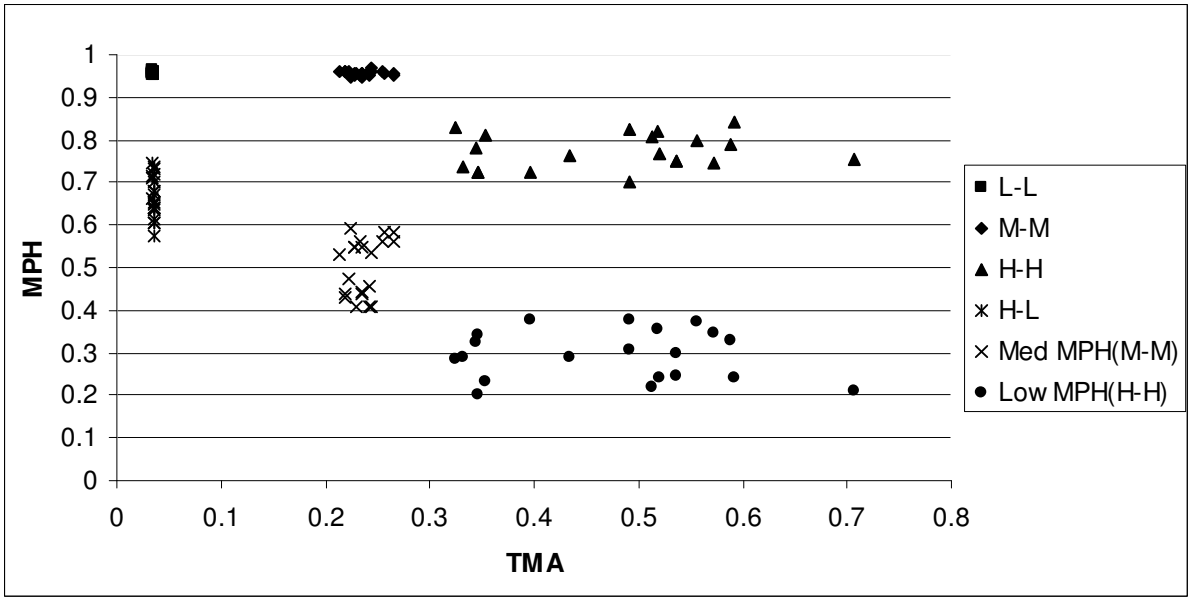

Figure 9. The TMA and MPH values of ETC matrices with medium and low MPH along with ETC matrices generated using the COV method.

V. FUTURE WORK

We have demonstrated how to calculate the MPH and TMA of existing ETC and ECS matrices. However, we currently do not have a way to generate ETC matrices with a desired TMA value. TMA is calculated based on the singular values. Therefore, we can set the singular values to obtain a specific TMA. The singular values are the diagonal values of the matrix $\sum$ in Equation 1. Matrices U and V are orthogonal matrices. Multiplying U, $\sum$, and V (see Equation 1) will result in an ECS matrix with the desired TMA. However, this matrix may have negative values are not physically meaningful.

VI. CONCLUSIONS

We have demonstrated how the SVD and the sum of the columns of the ECS matrix can be used to characterize heterogeneous systems. We proposed two measures, MPH and TMA, to quantify the heterogeneity of an HC system. These may provide a better characterization of the system than earlier methods. The detailed procedures to calculate MPH and TMA have been described. We have also generated ETC matrices using the COV method and compared their heterogeneity in terms of the new measures. We showed that the ETC matrices generated do not cover the entire range of possible heterogeneities as described by our new measures. We also present a procedure to generate a wider range of HC systems for simulation studies.

ACKNOWLEDGMENTS

The authors thank Paul Maxwell, Luis Briceño, Jerry Potter, and the anonymous reviewers for their valuable comments on this work.

REFERENCES

[1] S. Ali, T. D. Braun, H. J. Siegel, A. A. Maciejewski, N. Beck, L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, and B. Yao, "Characterizing resource allocation heuristics for heterogeneous computing systems," in *Advances in Computers*, Vol. 63, 2005, pp. 91-128.

[2] A. M. Al-Qawasmeh, A. A. Maciejewski, H. J. Siegel, J. Smith, and J. Potter, "Task and machine heterogeneities: Higher moments matter," *2009 International Conference on Parallel and Distributed Processing Techniques and Applications (PDPTA'09)*, Jul. 2009, pp. 3-9.

[3] S. Ali, H. J. Siegel, M. Maheswaran, D. Hensgen, and S. Ali., "Representing task and machine heterogeneities for heterogeneous computing systems," *Tamkang Journal of Science and Engineering*, Special 50th Anniversary Issue, invited, Vol. 3, No.3, Nov. 2000, pp.195–207.

[4] H. Barada, S. M. Sait, and N. Baig, "Task matching and scheduling in heterogeneous systems using simulated evolution," 10th Heterogeneous Computing Workshop (HCW 2001), in the proceedings of the *15th IEEE International Parallel and Distributed Processing Symposium (IPDPS 2001)*, Apr. 2001.

[5] T. D. Braun, H. J. Siegel, N. Beck, L. Bölöni, R. F. Freund, D. Hensgen, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, and B. Yao, "A comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems," *Journal of Parallel and Distributed Computing*, Vol. 61, No. 6, Jun. 2001, pp. 810–837.

[6] L. Canon and E. Jeannot, *Precise evaluation of the efficiency and the robustness of stochastic DAG schedules*, Research Report 6895, INRIA, April 2009.

[7] L. Canon and E. Jeannot, "Evaluation and optimization of the robustness of DAG schedules in heterogeneous environments," *IEEE Transactions on Parallel and Distributed Systems*, to appear.

[8] R. C. Chiang, A. A. Maciejewski, A. L. Rosenberg, and H. J. Siegel, "Statistical predictors of computing power in heterogeneous clusters," 19th Heterogeneous Computing Workshop (HCW 2010), in the proceedings of the *24th International Parallel and Distributed Processing Symposium (IPDPS 2010)*, Apr. 2010.

[9] M. K. Dhodhi, I. Ahmad, and A. Yatama, "An integrated technique for task matching and scheduling onto distributed heterogeneous computing systems," *Journal of Parallel and Distributed Computing*, Vol. 62, Sep. 2002, pp. 1338–1361.

[10] B. Eslamnour, S. Ali, "Measuring robustness of computing systems," *Simulation Modelling Practice and Theory*, Vol. 17, No. 9, Oct. 2009, pp. 1457-1467

[11] A. Ghafoor and J. Yang, "A distributed heterogeneous supercomputing management system," *IEEE Computer*, Vol. 26, No. 6, Jun. 1993, pp. 78–86.

[12] G. H. Golub and C. F. Van Loan, *Matrix Computations, Second Edition,* The John Hopkins University Press, Baltimore, MD, 1989.

[13] D. Huang, Y. Yuan, L. Zhang, and K. Zhao, "Research on tasks scheduling algorithms for dynamic and uncertain computing grid based on a+bi connection number of SPA," *Journal of Software*, Vol. 4, No 10, Dec 2009, pp. 1102-1109.

[14] M. Kafil and I. Ahmad, "Optimal task assignment in heterogeneous distributed computing systems," *IEEE Concurrency*, Vol. 6, No. 3, Jul. 1998, pp. 42–51.

[15] S. U. Khan and I Ahmad, "Non-cooperative, semi-cooperative, and cooperative games-based grid resource allocation," *20th International Parallel and Distributed Processing Symposium*, Apr. 2006.

[16] A. Khokhar, V. K. Prasanna, M. E. Shaaban, and C. Wang, "Heterogeneous computing: Challenges and opportunities," *IEEE Computer*, Vol. 26, No. 6, Jun. 1993, pp. 18–27.

[17] D. Xu, K. Nahrstedt, and D. Wichadakul, "QoS and contention-aware multi-resource reservation," *Cluster Computing*, Vol. 4, No. 2, Apr. 2001, pp. 95–107.

[18] V. Shestak, E. Chong, A. A. Maciejewski, and H. J. Siegel, "Robust sequential resource allocation in heterogeneous distributed systems with random compute node failures," 18th Heterogeneity in Computing Workshop (HCW 2009), in the proceedings of the *23rd IEEE International Parallel and Distributed Processing Symposium (IPDPS 2009)*, May 2009.

[19] H. Singh and A. Youssef, "Mapping and scheduling heterogeneous task graphs using genetic algorithms," *5th IEEE Heterogeneous Computing Workshop (HCW '96)*, 1996, pp. 86–97.

BIOGRAPHIES

Abdulla M. Al-Qawasmeh received his B.S. in Computer Information Systems from the Jordan University of Science and Technology in 2005. He received his M.S. in Computer Science from the University of Houston Clear Lake in 2008. Since August, 2008 he has been a Ph.D. student in Computer Engineering and a Graduate Research Assistant at Colorado State University. His research interests include robust heterogeneous computing systems and resource management in heterogeneous computing systems.

Anthony A. Maciejewski received the B.S., M.S., and Ph.D. degrees in Electrical Engineering in 1982, 1984, and 1987, respectively, all from The Ohio State University. From 1988 to 2001, he was a Professor of Electrical and Computer Engineering at Purdue University. In 2001, he joined Colorado State University where he is currently the Head of the Department of Electrical and Computer Engineering. He is a Fellow of IEEE. A complete vita is available at www.engr.colostate.edu/~aam.

Howard Jay Siegel was appointed the Abell Endowed Chair Distinguished Professor of Electrical and Computer Engineering at Colorado State University (CSU) in 2001, where he is also a Professor of Computer Science. He is the Director of the CSU Information Science and Technology Center (ISTeC), a university-wide organization for promoting, facilitating, and enhancing CSU's research, education, and outreach activities pertaining to the design and innovative application of computer, communication, and information systems. From 1976 to 2001, he was a professor at Purdue University. Prof. Siegel is a Fellow of the IEEE and a Fellow of the ACM. He received a B.S. degree in electrical engineering and a B.S. degree in management from the Massachusetts Institute of Technology (MIT), and the M.A., M.S.E., and Ph.D. degrees from the Department of Electrical Engineering and Computer Science at Princeton University. He has co-authored over 370 technical papers. His research interests include robust computing systems, resource allocation in computing systems, heterogeneous parallel and distributed computing and communications, parallel algorithms, and parallel machine interconnection networks. He was a Coeditor-in-Chief of the *Journal of Parallel and Distributed Computing*, and was on the Editorial Boards of both the *IEEE Transactions on Parallel and Distributed Systems* and the *IEEE Transactions on Computers*. He was Program Chair/Co-Chair of three major international conferences, General Chair/Co-Chair of seven international conferences, and Chair/Co-Chair of five workshops. He is a member of the Eta Kappa Nu electrical engineering honor society, the Sigma Xi science honor society, and the Upsilon Pi Epsilon computing sciences honor society. He has been an international keynote speaker and tutorial lecturer, and has consulted for industry and government. For more information, please see www.engr.colostate.edu/~hj.