Abstract—This paper proposes a dynamic load balancing strategy called DLBEM based on maximum likelihood estimation methods for parallel and distributed applications. A mixture Gaussian model is employed to characterize workload in data-intensive applications. Using a small subset of workload information in systems, the DLBEM strategy reduces considerable communication overheads caused by workload information exchange and job migration. In the meantime, based on the Expectation-Maximization algorithm, DLBEM achieves near accurate estimation of the global system state with significantly less communication overheads and results in efficient workload balancing. Simulation results for some representative cases on a two-dimensional 16*16 grid demonstrate that DLBEM approach achieves even resource utilization and over 90% accuracy in the estimation of the global system state information with over 70% reduction on communication overheads compared to a baseline strategy.

I. INTRODUCTION

With rapid progress in computing, communication, and storage technologies, large-scale computing have gained extensive interests in academia, industry and military [1], [2], [3]. To achieve high throughput and high performance in parallel and distributed systems, dynamic load balancing is one of the fundamental solutions. The increasing complexity of parallel environments characterized by heterogeneity, dynamism, large scale, and high communication frequency presents significant challenges to achieve desired load balancing and calls for lightweight dynamic load balancing strategies with minimum overheads. Load balancing strategies can be classified as centralized, decentralized, or hybrid; decentralized strategies feature good scalability and flexibility in heterogenous parallel computing systems, particularly in multi-site grid systems [2], [3], [4], [5]. However, load balancing approaches designed in a distributed fashion are generally NP-Complete [6] and may require global information to make decisions. Since each processor is “blind” to system’s global status, it requires a large number of communication messages to obtain timely global information, resulting in undesired overheads. For instance, the communication overhead could be significant up to 40% of the total execution time in some applications, e.g. the adaptive mesh refinement applications [7]. Hence, it is beneficial to design a dynamic distributed load balancing strategy with minimum communication overheads.

On the other hand, many numerical algorithms based on statistics models and probability theory have been developed and proved to be effective on approximate information inference. One class of statistical methods, called Maximum Likelihood Estimation (MLE) [8] methods, obtain the best fitting model to sample data and the optimal estimation on current available data. A popular implementation of parameter estimation with maximum likelihood is Expectation-Maximization (EM) [9] algorithm. Its simplicity and widely applicable feature make it useful for parameter inference and Bayesian Network structure learning. When converged, it can achieve optimal parameter estimation based on a small subset of data. Although MLE methods have been applied to many other fields, few are introduced in load scheduling and balancing areas. This study attempts to integrate MLE methods in dynamic load balancing strategy to enhance the performance of parallel and distributed applications with minimum overheads.

We propose the DLBEM (Dynamic Load Balancing Based on EM Algorithm) approach, which uses EM algorithm to make accurate workload migration decisions with reduced number of communications. The DLBEM is designed in tune with grid workload models in a hybrid distributed virtual organization. Unlike earlier methods, this strategy estimates and adjusts workload accurately with a small portion of complete workload information. Our contributions are three-fold: (1) Using the EM algorithm, DLBEM obtains the global workload information quite accurately with much less communication overheads; (2) An efficient job migration policy is designed to achieve balanced utilization of computing resources and avoid oscillation during job migration; (3) The DLBEM strategy unveils the potential of applying maximum likelihood estimation in job scheduling for performance optimization.

The rest of the paper is organized as follows: Section II presents a brief introduction to the EM algorithm. Section III presents the DLBEM strategy. The results of simulation are presented in Section IV. Section V discusses the related work in dynamic workload balancing and relevant statistical methods. Section VI concludes the paper and envisions future work.
II. THE EXPECTATION-MAXIMIZATION ALGORITHM

In real world data modeling, the data subject to analysis are often presented in incomplete forms, either due to the limitation of sampling process or due to intentionally missing in purpose of simplifying likelihood function to tractable form. EM algorithm [9], [10], [11], [12] is an effective statistic method to maximize the parameters likelihood estimation in models where unobserved latent variables exist. The algorithm performs in an iterative way until it converges, and each iteration can be divided into two steps. The first step called the E step computes the expectation of complete data set likelihood given the model parameters. The expected likelihood constructed is then used to compute the current maximum likelihood estimation of the model parameters in the second step, called the M step. The beauty of EM algorithm lies in its simplicity, and proved to be effective in Gaussian Mixture models where unobserved latent variables exist. The algorithm method to maximize the parameters likelihood estimation in purpose of simplifying likelihood function to tractable form.

\[ \theta^{n+1} = \arg \max_{\theta} Q(\theta) \] (4)

By alternating between E step and M step, the EM algorithm iterates from initial estimation \( \theta^0 \) and converges to a local maximum of the likelihood function.

III. THE DLBEM STRATEGY

In this section we formally present our load balancing strategy design in two steps. First we use the EM algorithm to estimate the mean workload in a virtual hybrid distributed system. Second jobs are migrated according to this estimated value. The problem formulation and strategy implementation goals are presented before we introduce the strategy.

A. Problem Formulation

Before formally presenting the DLBEM strategy, we make the following assumptions. A parallel computing system consists of a large number of processors, which are scattered in multiple sites connected by local networks. Processors communicate with each other in a point-to-point way. Incoming programs can be segmented into a batch of small jobs. Jobs are assigned to processors and can be migrated freely. Instead of balancing load in every processor, we follow a hybrid system topology for better scheduling performance. All the processors are partitioned into logical zones. Each zone contains geographically nearby processors so that processors communicate more efficiently within zones. In each zone there is a leading processor in charge of collecting information, performing estimation, broadcasting local messages and executing job migration. The processors are organized in this way to reduce computational overheads and number of communication messages. If every processor make balancing decisions with the complete system state information, it requires \( O(n^2) \) messages delivered, causing significant communication overheads. In DLBEM, balancing decisions are made by leading processors when new batch of jobs arrive at system, and each leading processor selects a small subset of processors for workload information collection. In this way the communication overheads are reduced substantially.

In data-intensive applications, a large number of computing jobs is done by decomposing the jobs into "Bags-of-Tasks" (BoT), which stands for a set of independent computational tasks with almost no communication involved. According to a recent research result on grid workload modeling [15], the properties of such tasks make mixture Gaussian model a good choice to fit the probability distributions of workload for real-world parallel computations. In this paper we proposes the DLBEM algorithm based on the assumption of such workload modeling. DLBEM algorithm based on the assumption of such workload modeling. DLBEM first estimates the mean system workload; then each processor selects a small subset of processors for workload information collection. In this way the communication overheads are reduced substantially.

In data-intensive applications, a large number of computing jobs is done by decomposing the jobs into "Bags-of-Tasks" (BoT), which stands for a set of independent computational tasks with almost no communication involved. According to a recent research result on grid workload modeling [15], the properties of such tasks make mixture Gaussian model a good choice to fit the probability distributions of workload for real-world parallel computations. In this paper we propose the DLBEM algorithm based on the assumption of such workload modeling. DLBEM first estimates the mean system workload based on the reported workloads which come from only a small subset of the complete load information. Then each node adjusts its workload according to the mean value. The model intends to maximize the utilization of computing power without incurring large communication overheads. Further, the proposed algorithm considers heterogenous systems and is designed to achieve high scalability and avoid costly oscillation.
in job migrations. These goals are achieved and demonstrated in theoretical analysis and simulation results in the later part of the paper.

B. Workload Estimation Using EM Algorithm

Compared with central control scheduling methods, distributed load balancing exhibits better scalability. However, distributed algorithms may also introduce more communication costs and lead to more computation complexity. The DLBEM strategy follows a hybrid distributed organization to make best use of both decentralized and centralized approaches. The workload information allows for computing power difference, and is recorded as CPU utilization. In this way the algorithm accommodates heterogeneous computing environment, which prevails in today’s parallel systems. Load balancing is activated when a new batch of jobs arrive at system. In one polling period the partial workload information is collected and passed to EM algorithm to obtain optimal parameter estimation iteratively. With the average workload information obtained, each processor then makes job migration decisions accordingly. Figure 1 shows the complete procedure of DLBEM approach:

![Strategy flow chart](image)

1) Information collection and EM initialization: The first step in DLBEM uses a random selection model to decide which processor should report its workload information to the leading processor in the logical zone. This information collection is done in a completely distributed way. In the random selection model, each processor holds a system parameter $\alpha$, which determines the proportion of processors reporting their workloads. Then each processor makes its own decision with probability $\alpha$ to report its workload to leading processor. Leading processors decide whether the procedure is finished and when to start estimation. The scheme is simple to be implemented and achieves desired scalability because of small overheads.

Moreover, we need to consider the initialization of EM algorithm. EM is generally sensitive to initial parameters and inappropriate selection may lead to unsatisfied estimation. There are many solutions to this problem, including highest likelihood [16], using clustering algorithms [17] or tree structure scheme [18]. For example, in practice the observed data set can be clustered by using the k-means clustering algorithm [19], the number of Gaussian distributions $k$ is fixed then and input to the load balancer for optimal parameter estimation.

2) Finding maximum likelihood estimation using EM:

In the next step of DLBEM, the EM algorithm is applied to estimate maximum likelihood of desired parameter. The following notations and definitions are used throughout the iteration process:

- $\chi$: $\chi_{obs}$ and $\chi_{mis}$: $\chi$ denotes for complete data set, $\chi = (\chi_{obs}, \chi_{mis})$, as defined in Section II.
- $m$: the network is divided into $m$ zones, observed data $\chi_{obs}$ are sampled from the $m$ groups in the form of vectors.
- $k$: $k$ Gaussian distributions (or clusters) constitute the Gaussian mixture model. The value of $k$ is given out in initialization.
- $x_{obs}^j$: the particular observed data vector which comes from the $i$-th group. Therefore the observed data set $\chi_{obs} = \{x_{obs}^1, x_{obs}^2, \ldots, x_{obs}^j, \ldots, x_{obs}^m\}$.
- $t$: this variable stands for which cluster a particular observed data vector comes from, $t = 1, 2, \ldots, k$.
- $p(t = i)$: the probability of a particular observed data vector comes from the cluster.
- $\mu_i$: the mean value of the $i$-th Gaussian.
- $\sigma_i^2$: the standard deviation of the $i$-th Gaussian.
- $\theta$: stands for the unknown parameter, specifically $\theta = \{\mu_1, \ldots, \mu_n, \sigma_1^2, \ldots, \sigma_n^2, p(x = 1), \ldots, p(x = n)\}$
- $\theta^n$: current parameter estimation after $n$ times iterations.

The first task is to calculate the probability of a particular observed data set $x_{obs} \in \chi_{obs}$ ($x_{obs}$ is a vector) comes from the $i$-th Gaussian.

$$P(x_{obs}^j | t = i, \theta) = \mathcal{N}(\mu_i, \sigma_i^2)$$

$$= \frac{1}{\sqrt{2\pi\sigma_i}} \exp\left(-\frac{(x_{obs}^j - \mu_i)^2}{2\sigma_i^2}\right)$$  \hspace{1cm} (5)

The expectation step is performed by using parameters from last maximization step and equation (5). The estimation of a particular observed sample comes from the $i$-th Gaussian is calculated by using the Bayes rule as illustrated in Section II:

$$P(t = i | x_{obs}^j, \theta^n) = \frac{P(x_{obs}^j | t = i, \theta^n)P(t = i | \theta^n)}{\sum_{l=1}^k P(x_{obs}^j | t = l, \theta^n)P(t = l | \theta^n)}$$  \hspace{1cm} (6)
The expected log-likelihood function of the joint event is given by:

$$Q(\theta) = \sum_{j=1}^{m} \sum_{i=1}^{k} P(t = i | x^j_{\text{obs}}, \theta^n) \log P(t = i, x^j_{\text{obs}} | \theta)$$ (7)

By expanding the probability density function, and let the partial differential value equals to zero, we find the current maximum parameter for next iteration.

$$\mu_i = \frac{\sum_{j=1}^{m} P(t = i | x^j_{\text{obs}}, \theta^n) x^j_{\text{obs}}}{\sum_{j=1}^{m} P(t = i | x^j_{\text{obs}}, \theta^n)}$$ (8)

Using the same method, the EM algorithm can get other parameters including cluster probability and standard deviation.

$$\sigma^2_i = \frac{\sum_{j=1}^{m} P(t = i | x^j_{\text{obs}}, \theta^n) (x^j_{\text{obs}} - \mu_i)^2 (x^j_{\text{obs}} - \mu_i)}{\sum_{j=1}^{m} P(t = i | x^j_{\text{obs}}, \theta^n)}$$ (9)

$$p(t = i) = \frac{\sum_{j=1}^{m} P(t = i | x^j_{\text{obs}}, \theta^n) x^j_{\text{obs}}}{\sum_{i=1}^{n} \sum_{j=1}^{m} P(t = i | x^j_{\text{obs}}, \theta^n)}$$ (10)

When converged, the estimation of parameters calculated in equation (8), (10) yields the mean workload value.

$$\bar{\mu} = \sum_{i=1}^{n} p(t = i) \mu_i$$ (11)

3) Job migration: When a new batch of jobs comes, they are distributed evenly among logical zones to balance inter-zone utilization. Once the global average workload information is estimated, each processor decides whether it is running in an imbalance state. After the decisions are made workload will be redistributed within the system. It is difficult to design job migration policy because each new wave of job migration may cause new imbalance, therefore lead to continuous job migrations among processors, incurring a lot of communication overheads and resulting in the oscillation phenomenon.

To address these issues, we restrict job migration within local zones, which means the intra-zone load balancing is done by migration. The detailed algorithm in pseudo-code for one migration round is presented in Table I. In each round the leading node polls every processor in the zone for match. A processor is lightly loaded if its workload is less than $\bar{\mu}$ (the global average workload), likewise a processor is heavily loaded if its workload is greater than $\bar{\mu}$. A lightly loaded processor and a heavily loaded one will be matched to form a potential migration pair. This process is done by the leading node. When one such pair is found, the leading node will send signals to the pair and the extra load will flow from the heavily loaded processor to the lightly loaded one. If one single processor cannot find a partner, it will wait until the next round of matching. The matching and migration process continues until all the nodes perform migration at least once, or no pair can be found within the zone. An example of how the job migration policy executed in the hybrid organization is illustrated in Section III-B4. Although jobs migrate locally, due to the global average workload information, this policy achieves system wide balance, and involves much less communications. Detailed analysis of our policy based on simulation will be presented in Section IV.

4) An illustrative example: We illustrate the DLBEM strategy with a simple example. Consider the 16 processors network displayed in Figure 2. We use a baseline scheme for comparison. The baseline scheme is fully distributed and requires the complete global system information. Therefore, each processor collects information from all the other 15 nodes for workload balancing. The total number of messages in the system would be $16 \times 15 = 240$. Now using DLBEM, the network is divided into 4 zones numbered from 0 to 3. The information collection phase has resulted processor 1 from zone 0, processor 0, 2 from zone 1, processor 1, 3, 4 from zone 2 (none from zone 3) reporting their workloads. The total number of communications includes both intra-zone and inter-zone communication. Intra-zone communication includes processor reporting workload information to zone leader, and zone leader broadcasting estimation value. Inter-zone communication includes messages passed among zone leaders. In our example with 6 processors selection the number is $6^4 = 36$ messages, a reduction of 85% communication messages involved. By using the EM estimation, we calculate the estimated average is about 0.469. Compared with actual average value of 0.546, the result is about 86% accuracy.

<table>
<thead>
<tr>
<th>TABLE I</th>
<th>JOB MIGRATION ALGORITHM</th>
</tr>
</thead>
<tbody>
<tr>
<td>n: number of processors in zone, mod(n, 2) = 0</td>
<td>Load: workload values vector. Load = [L₁, L₂,...,Lₙ]</td>
</tr>
<tr>
<td>Status: processor status vector. Status = [S₁, S₂,...,Sₙ]</td>
<td>IF $L₁ &lt; \bar{\mu}$, Mark $S₁ = L$ (-1)</td>
</tr>
<tr>
<td>ELSE IF $L₁ &gt; \bar{\mu}$, Mark $S₁ = H$ (1)</td>
<td>ELSE Mark $S₁ = B$ (0)</td>
</tr>
<tr>
<td>Match: matching times record vector. Match = [0,0,...,0]</td>
<td></td>
</tr>
<tr>
<td>Sub Procedure: Initialization</td>
<td></td>
</tr>
<tr>
<td>SET 1 → Terminate</td>
<td></td>
</tr>
<tr>
<td>CALL Sub Procedure: Initialization</td>
<td></td>
</tr>
<tr>
<td>WHILE ( Search( Status[ ] ) AND Terminate)</td>
<td></td>
</tr>
<tr>
<td>{</td>
<td></td>
</tr>
<tr>
<td>Avg(i, j)</td>
<td></td>
</tr>
<tr>
<td>Match[i]++</td>
<td></td>
</tr>
<tr>
<td>Match[j]++</td>
<td></td>
</tr>
<tr>
<td>r°Check status of vector Match°</td>
<td></td>
</tr>
<tr>
<td>IF ALL Match[k ∈ n] ≠ 0</td>
<td></td>
</tr>
<tr>
<td>UPDATE 0 → Terminate</td>
<td></td>
</tr>
</tbody>
</table>
The sample of this example is small, in larger systems, the statistical property of big number will achieve more accuracy, which will be shown in simulation of 256 processors in Section IV.

Fig. 2. A simple example of 16 processors

The job migration policy works as follows. Take the third zone as an example, the estimated average workload is $0.469$, the workload vector of zone 3 is $(0.502, 0.541, 0.673, 0.353)$. There are three heavily loaded processors and one lightly loaded processor. Processor $P_0$ will first match with processor $P_3$ to produce workload of 0.428 after job migration from $P_0$ to $P_3$. Then the unmatched two heavily processors will match the two newly balanced processor to produce a balancing zone two with workload 0.485 and the other two with workload 0.55. Compared to the actual system average workload of 0.546, the obtained result is satisfying and jobs only migrates three times within the zone, which incurs little overheads on system performance.

IV. Evaluation and Discussion

In this section we present three different simulations to evaluate the performance of our strategy. A workload generator program is designed and implemented in C language on a $16 \times 16$-two-dimensional grid. In order to perform load balancing among processors, we need to find out the average workload with less communication overheads. We also assume a multiprogramming environment, which means several small jobs run simultaneously on a single processor. EM estimation and job migration policy are also implemented in C. At each time instant, the probability function of workload distributed in the whole system follows mixture Gaussian model. With this assumption, We measure the prior workload distribution of oncoming jobs in two sites in different time instants. The results are shown in figure 3(a).

The DLBEM strategy approximates average workload with acceptable accuracy, as well as reducing overhead of communication messages as many as possible. The first two simulations explore influential factors of estimated accuracy. For the initialization, since we use a random mixture model generator as input data, the cluster number is known beforehand and the initialization parameter $k$ (number of Gaussian distributions in mixture model) is fixed (in these simulations $k = 2$). The parameter $\alpha$ chosen in the node selection phase is based on how many processors are designated to contribute their workload information for sampling. For example, if 96 out of
processors are selected, the initial parameter \( \alpha \) is set to 0.375.

![Graph](image)

(a) Estimated accuracy affected by number of nodes

![Graph](image)

(b) Estimated accuracy with communication overhead reduction

Fig. 4. Simulations of influence factors on estimation accuracy

For the purpose of comparison, we use a baseline strategy, in which node collects all the workload information of other nodes for balancing. In both simulations, the accuracy of estimations is given by percentages of estimated quotient (estimated mean load dividing actual mean load), and used as the Y-axis of both figures. In figure 4(a), X-axis represents the relative communication overheads reduction compared with baseline strategy in estimation process, while in figure 4(b), X-axis represents the number of processors used. Simulation results demonstrate that DLBEM algorithm can successfully reduce communication overheads among nodes while maintains satisfactory results with over 90% accuracy. Another interesting observation is that when relatively less nodes are selected, the estimation accuracy exhibits more randomness. The reason can be explained as follows: a small number of data can’t model distribution model accurately enough. To overcome this problem, we should select more processors for estimation. In figure 4(b), 20 processors or more selected by using our information collection scheme make stable estimation accuracy increasing. As a example of modeling from sample data, the Gaussian mixture model estimated by the DLBEM strategy with workload information of 80 processors is shown in figure 3(b).

Finally, the effectiveness of the proposed job migration policy is displayed in figure 5. The procedure is simulated as follows. At each time instant we measure the load information from each processor. The EM estimation is performed with 90 nodes selection to estimate the average workload. The 256 processors are divided into 16 zones. We integrate our job migration policy into the simulation program. Here we use standard deviation to show the effects of load balancing because it is a good criteria for displaying workload departure from the mean utilization value. The result displayed intuitively indicates that our policy greatly decrease standard deviation at each time instant, results in more evenly-distributed workload among processors.

V. RELATED WORK

Dynamic load balancing algorithms in parallel and distributed systems have been studied intensively with the help of both theoretical and practical investigations. Iterative load balancing approaches such as diffusive load balancing [20], [21] exchange workload information between neighbor processors, thus to avoid communication messages flooding in the whole system. However, although locally balanced, the algorithms were proved to be globally imbalance [22]. Innovative distributed methods were introduced to estimate workload for global balance, including AI techniques [23], P2P techniques [24], and bit vector approach specifically designed for publisher/subscribe systems [25]. Similar to our DLBEM strategy, most of these methods deal with a collection of independent tasks, known as BoT (Bags-of-Tasks). M. Ardler, Y. Gong and A. L. Rosenberg [26] investigated optimal sharing of BoT in heterogenous clusters by designing a FIFO work sharing protocol. More recent work done by A. Benoit et al. [27] solved the problem of finding the optimal scheduling of concurrent BoT for the off-line setting, and the method was adopted to a online scenario which was proved to be near-optimal. Our DLBEM algorithm considers practical aspects of balancing process by using mixture Gaussian models of BoT and aims at achieving near-optimal balancing result.

To reduce communication overheads for better performance, the particles approach [28] places intensively communicating tasks close to each other to minimize communication delays. The MOSIX [29] system implements load balancing methods at the system level, at each polling round every processor randomly selects a subset of all the other processors for workload information update. Gu et al. [30] suggested a predictive method which updates workload information less by making decisions after certain rounds, the estimation is achieved by
introducing a linear regression $L_2E$ predictive filtering model based on historical workload data. Our DLBEM strategy differs from the previous methods by combining their merits. The strategy generates more accurate workload estimation based on frequently updated current information.

Recently some studies have started to apply statistical models in various computer science research areas, e.g. aiding wireless sensor network data collection process [31], modeling trends analysis [32] and grid resource scheduling [33]. Bayesian decision networks were used to handle uncertainty in system activities in [34]. The simulation results showed the effectiveness of such methods.

VI. CONCLUSION

We proposed the DLBEM dynamic load balancing strategy using Expectation-Maximization algorithms in a hybrid distributed system topology. DLBEM uses EM algorithm to make accurate estimations for workload repartition decisions based on partial workload information, and effectively performs local workload migration according to the estimated workload information. Our simulation results showed that the DLBEM strategy reduces communication overheads and workload migration overheads substantially, as well as making accurate migration decisions and achieving even utilization of computing resources.

In our future work, we will conduct trace-based simulations and real-world experiments in large-scale systems such as TerraGrid and PlanetLab and further refine the DLBEM strategy to accommodate node failures to achieve better robustness. More studies on using various stochastic methods for dynamic load balancing will be investigated.

REFERENCES