FCD: Fast-Concurrent-Distributed Load Balancing under Switching Costs and Imperfect Observations

Furong Huang
Dept. Electrical Engineering and Computer Science
University of California, Irvine
Email: furonghATuciDOTedu

Anima Anandkumar
Center for Pervasive Communications and Computing
Dept. Electrical Engineering and Computer Science
University of California, Irvine
Email: a.anandkumarATuciDOTedu

***Abstract*—The problem of distributed load balancing among m agents operating in an n-server slotted system is considered. A randomized local search mechanism, FCD (fast, concurrent and distributed) algorithm, is implemented concurrently by each agent associated with a user. It involves switching to a different server with a certain exploration probability and then backtracking with a probability proportional to the ratio of the measured loads in the two servers (in consecutive time slots). The exploration and backtracking operations are executed concurrently by users in local alternating time slots. To ensure that users do not switch to other servers asymptotically, each user chooses the exploration probability to be decaying polynomially with time for decaying rate $\beta \in [0.5, 1]$. The backtracking decision is then based on an estimate of the server load which is computed based on local information. Thus, FCD algorithm does not require synchronization or coordination with other users. The main contribution of this work, besides the FCD algorithm, is the analysis of the convergence time for the system to be approximately balanced, i.e. to reach an ϵ-Nash equilibrium. We show that the system reaches an ϵ-Nash equilibrium in expected time $O\left(\max\left\{n\log\frac{n}{\epsilon}+n^{\frac{1}{\beta}},\quad\left(\frac{n^3}{m^3}\log\frac{n^2}{\epsilon}\right)^{\frac{1}{\beta}}\right\}\right)$ when $m > n^2$. This implies that the convergence rate is robust with large scale system(large user population), and is not affected by imperfect measurements of the server load. We also extend our analysis to open systems where users arrive and depart from a system with an initial load of m users. We allow for general time-dependent arrival processes (including heavy-tailed processes) and consider a uniform and a load-oblivious routing of the arrivals to the servers. A wide class of departure processes including load-dependent departures from the servers is also allowed. Our analysis demonstrates that it is possible to design fast, concurrent and distributed load balancing mechanisms in large multi-agent systems via randomized local search.**

I. Introduction

The recent years have witnessed an explosive growth in the demand for computational and storage services. The paradigm of cloud computing holds the promise to meet these demands in an efficient and a timely manner. Increasingly, more and more web services are deployed on large-scale distributed cloud systems. However, such systems present huge challenges involving resource allocation, scalability and dynamics. One direction of work has focused on the design and analysis of scalable resource allocation mechanisms, popularly termed as load-balancing mechanisms.

Traditionally, load-balancing mechanisms have relied on the presence of a centralized controller (also known as the dispatcher) with a global view of the operations, which efficiently allocates users to lightly loaded servers upon arrival (e.g., Join-the-Shortest-Queue policy, and the more recent Join-the Idle-Queue policy [13]). However, such dispatchers may not always be present and in many systems, the users may be autonomous agents making decisions. Recent focus has been on the development of distributed load balancing mechanisms, implemented locally and myopically by autonomous agents without a centralized controller, popularly termed as *randomized local search*. Such mechanisms can also effectively model agent behaviors in large distributed systems such as cable or electricity supply market.

Randomized local search can be thought of as a randomized algorithm for in-network computation based on local information at the users with a goal to achieve load balancing among the servers. Under randomized local search, each user randomly selects another server and measures its load. If the alternative load is lower then with some probability, the user switches to the new server. Intuitively, the probability that she switches is higher if the ratio of the current load to the new load is larger. However, the load is sensitive to the number of users and increases under the influx of new users to the server. Thus, it is not always optimal for the users to switch to a server with a lower load and randomization is therefore necessary for efficient load balancing.

We consider randomized local search mechanisms for distributed load balancing, but with additional challenges. First, we consider concurrent load balancing which makes it harder for the users to measure the true load at the server. This means that the users have access to only indirect observations of the server loads since they are simultaneously exploring different servers. This entails the use of imprecise information about the server loads in randomized local search mechanisms. Second, we impose a penalty on switching to new servers and require that users settle down to a load balanced state asymptotically. This entails a careful design of exploration probabilities which decay over time and yet ensure convergence to an (approximate) load balanced state. Third, we extend our convergence time analysis to open systems and demonstrate that convergence time is largely unaffected by user dynamics for a wide class of arrival and departure processes. Our analysis in this paper demonstrates that it is possible to design fast, concurrent and distributed load balancing solutions in large multi-agent

systems through randomized local search without the need for coordination or synchronization among the users.

A. Related Work

The load-balancing game under consideration is a well studied notion in game theory and is popularly known as the *congestion game*. Rosenthal [15] proved that pure Nash equilibria (NE) always exists for congestion games since they are potential games. In our paper, we consider one of the potential functions by [15], modulo a linear rescaling and analyze convergence to (approximate) Nash equilibrium via the decay in the potential function.

The presence of a potential function also establishes that pure NE can be found via a sequence of better-response moves, in which agents repeatedly switch to lower-cost strategies. However, such strategies require global information of all server loads while the randomized local search is only based on the loads in the current and the previous servers used by the agent. Moreover, under the strongly distributed setting considered here, such moves may occur simultaneously leading to stale information and suboptimal responses. These shortcomings are also faced by logit response strategies considered in [16], where convergence time analysis is carried out via elegant Markov chain mixing time analysis. Moreover, the convergence time to approximate Nash equilibrium under logit response is shown to be linear in the number of users while our schemes have logarithmic guarantees. See Table I. Another class of popular mechanisms are based on the so-called replication policies (e.g. [8]), where the agents imitate the actions of other agents. However, such mechanisms require that users be able to observe the actions of other users, while under our setting, the users only need to measure the loads of their servers.

Convergence time for load balancing has also been analyzed through the notion of regret in *multi-armed bandit* systems in [3,12]. However, again, these schemes require the users to keep track of the quality of service at all the servers (assumed to be stochastic) while the schemes considered here only require the current load and the load experienced by the user in the previous time step. Moreover, the regret under the schemes in [3,4,12] scale poorly when the number of users is large, while efficient scaling in large-scale systems forms a core contribution of our work here. Similarly, the work in [11] analyzes the popular multiplicative update algorithm, proposed earlier for non-stochastic bandits, and prove convergence to the Nash equilibrium for congestion games. However, they do not provide convergence rate guarantees.

Randomized local search mechanisms are a popular class of distributed load balancing strategies. Goldberg [9] proposed a variant of this strategy, where users select alternative servers at random and migrate if the alternative load is lower. However, this protocol requires that migration events take place one at a time, and that the loads be updated immediately. Even-Dar and Mansour [7] improved the convergence rate to $O(\log\log m + \log n)$ for m users and n servers and allowed for concurrent, independent rerouting decisions. However, they require the users to have a global knowledge on whether they are currently in underloaded or overloaded servers. The work of P. Berenbrink [5] considered a strongly distributed version of randomized local search, where agents migrate selfishly without any centralized control. In each time slot, users select a different server uniformly at random and test its load. If the load is lower than the current server, then users migrate with a certain probability. Guarantees on convergence time are provided in [5]. The work in [17] extends the analysis of randomized local search to asynchronous exploration based on Poisson clock ticks. This assumption leads to a better convergence rate guarantee but requires asynchrony of actions among the agents.

Distributed load balancing mechanisms have also been studied for open systems, where users arrive and depart, typically according to a queueing model. The popular mechanism, known as power of d choices [14], was studied for queueing systems where incoming users randomly choose d out of n servers and waits for service at the servers. Similarly, other works [10,13,17,18] have studied load balancing under queueing processes. For instance, the work in [17] analyzes the stability region under randomized local search and random load oblivious mechanisms but do not consider transient (i.e. convergence rate) analysis.

Our work differs from the analysis in [5,17] in a few key aspects: the works in [5,17] assume that it is feasible to measure the true load of another server before migrating under concurrent moves by different users. However, under our setting, the users can only measure the loads of different servers by committing to spend at least one time slot in those servers. This implies that such load measurements do not include users who are currently exploring other servers and may decide to backtrack. We undertake a careful analysis of the effect of incorrect load information on the convergence rate to an approximately balanced state. We also consider decaying exploration probabilities which discourages switching among the users, while the previous works do not limit exploration even after a load balanced state is reached. Moreover, we extend our analysis to open systems where users arrive and depart from the system. We take a different approach to user dynamics compared to [17]. We assume an initial set of m users in the system and a continuous stream of arrivals and departures, as the users proceed to balance the loads. We provide guarantees on the convergence time to reach and maintain an ϵ-neighborhood of the Nash equilibrium.

We note that load balancing mechanisms have been considered under alternative scenarios which are not directly related to this work. For instance, the work in [13] considers efficient load balancing based on the design of a dispatcher which implements Join-Idle-Queue policy for incoming users. The work in [19] considers green load balancing where energy and geographical constraints are involved. It is envisioned that our proposed distributed load balancing mechanisms achieve better gains when operated in conjunction with these policies.

B. Summary of Contributions

We now summarize the main contributions of this work. Convergence time guarantees for the users to settle down to an ϵ-Nash equilibrium under concurrent play of randomized local search mechanisms are provided. We consider increasingly closed and open systems for analysis. In a closed system, the number of users m is fixed, while in open systems, users arrive and depart from the system in each time slot. The key results on convergence rates are summarized in Table I.

Each user implements the FCD algorithm with a certain exploration probability which decays over time, enabling the users to settle down eventually. A careful choice of the exploration probability is required: if it decays too fast, then we may not be able to achieve convergence to an approximate load balanced state; on the other hand, if it decays slowly, then it encourages more switching among the users. We consider the scenario, where the exploration probability decays polynomially with time as $t^{-\beta}$ for $\beta \in [0.5, 1]$. Our results establish that the convergence rate to an approximately load balanced state has an inverse relationship with β as expected. Moreover, no system synchronization is needed, i.e., different users can explore and backtrack in local alternating time slots without the need for coordination[1]. Since the users are unable to measure the true loads at the servers during the exploration phase, it entails the use of incorrect load information under the randomized local search mechanism. Our convergence time analysis involves a careful manipulation of the concentration bounds on the load estimates, in conjunction with the analysis of the potential function for ϵ-Nash convergence. Conceivably, our work can pave way for more sophisticated estimation algorithms under other noisy load measurement models.

We extend our convergence time analysis to open systems with an initial set of m users, and an arrival and a departure process of users as the load-balancing mechanism proceeds. The results in Table I demonstrate that the convergence rate is hardly affected by the user dynamics, as long as the initial set of users m is large enough and the arrival and departure processes satisfy a general set of conditions. See Theorem 2 for details. This encompasses a large family of arrival and departure processes, including heavy tailed arrivals and load dependent departures. Moreover, our bounds on the convergence rate to an approximately load balanced state are nearly independent of the number of users m when $m \gg n$, where n is the number of servers. Thus, our results demonstrate that it is possible for a large number of autonomous agents to rapidly settle down to an approximately load balanced state through fully distributed, concurrent and myopic mechanisms among the users.

[1]Our setting of a slotted system here is more challenging than a completely asynchronous system considered in [17] since concurrent movement of users results in incorrect load measurements under our setting.

TABLE I: Guarantees on expected convergence time to ϵ-Nash equilibrium with m users, n servers and exploration probability $t^{-\beta}$, where $\beta \in [0.5, 1]$ and $m > n^2$. For details, refer to Theorems 1 and 2.

Settings	Convergence time to ϵ-Nash
Setting in [16][1]	$O(m \log\log m)$
Setting in [5][2]	$\log\log m^2 - \log\log \frac{\epsilon^2 m^2}{n^3}$
Setting in [2][2]	$\log(n)/\epsilon$
Closed System[3]	$\max\left\{ n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}, \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}} \right\}$
Open System[3]	$\max\left\{ n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}, \left(\frac{n^4}{\epsilon m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}, \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}} \right\}$

1: Under logit response strategies.
2: The setting in [5] assumes parallel movements of users. The works in [5] and [2] assume a closed system where users have access to the true load of the servers and impose no switching costs for users to measure loads in other servers.
3: Proposed algorithm in this paper.

II. System Model

A. Notion of Nash Equilibrium for Load Balanced State

There are m users and n servers. Assume that users require identical service rates. An assignment of users to servers at time t is represented as a vector $\mathbf{X}(t) = (X_1(t), X_2(t), \ldots, X_n(t))$ in which $X_i(t)$ denotes the number of users who are assigned to server i, i.e. the load at server i. Let $Y_a(t)$ be the load experienced (measured) by user a. Thus, if user a is served by i at time t, $X_i(t) = Y_a(t)$. Moreover, we assume that each user has one unit of memory. In other words, user a has knowledge of $Y_a(t-1)$ and $Y_a(t)$ at time t. In addition, we assume that all users are active as they are in the closed system.

The popular notion of Nash-equilibrium of a game denotes a state where no player has the incentive to change her current decision. The setting under consideration belongs to a class of games termed as *congestion games* or *balls and bins*. In the setting where the users have uniform loads, there is a unique Nash equilibrium corresponding to the perfectly load balanced state [5]. Recall that the Nash equilibrium is reach at time slot t if $\max_{ij} |X_i(t) - X_j(t)| \leq 1$, we now define the notion of convergence to the ϵ-Nash equilibrium.

Definition 1: An ϵ-Nash equilibrium is reached at time slot t, if

$$\max_{ij} |X_i(t) - X_j(t)| \leq \epsilon \frac{m}{n}.$$

The system state $\mathbf{X}(t)$ is called approximately load balanced[2]. $T := \min t$ is defined as the convergence time.

Thus, the notion of an ϵ-Nash equilibrium relaxes the notion of the Nash equilibrium. In an ϵ-Nash equilibrium, no user can unilaterally change the experienced load by a multiplicative factor of less than $1 - \epsilon$ [5].

[2]The term "approximately load balanced state" is replaced by "load balanced state" in the sequel.

Algorithm 1 FCD algorithm with static number of users. Note that no system synchronization is needed, i.e., different users explore and backtrack in local alternating times in slotted distributed system.

Exploration probabilities $\gamma_t(a) \propto t^{-\beta(a)}$ at time t.
if $\gamma_t(a) > 0$ **then**
  System state is exploring;
else
  System state is inactive;
end if
for $k = 0 : T$ **do**
  if $t = 2k$ **then**
    $i \leftarrow$ the server a resides in;
    if exploring **then**
      Explore to another server uniformly with probability $\gamma_{2k}(a)$;
      $\mathbb{I}[\text{Explore}] = 1$;
    end if
  else
    $j \leftarrow$ the server a resides in;
    if $\mathbb{I}[\text{Explore}] = 1$ **then**
      Switch back with probability $f_{ji}(Y_a(2k), Y_a(2k+1), \gamma_{2k}(a))$;
    end if
    $\mathbb{I}[\text{Explore}] = 0$;
  end if
end for

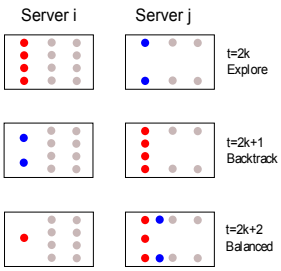


Fig. 1: A single round of the iterative algorithm. The red users in server i and the blue users in server j decide to explore, while the grey users in both the servers stay. In the backtracking slot, each user decides whether to backtrack to the original server. This completes one round of the algorithm.

B. Randomized Local Search Mechanism

The randomized local search algorithm, FCD, for load balancing in slotted systems and is given in Algorithm 1. FCD algorithm proceeds as follows: a user a explores to another server randomly in even time slots $2k$ with probability γ_{2k}. In odd time slots, the user decides whether to stay in the new server or switch back to the previous server with a certain backtracking probability $f_{ji}(2k)\,(Y_a(2k), Y_a(2k+1), \gamma_{2k})$. We term a cycle of exploration and backtracking as one round(one super time slot) of the algorithm. By carefully selecting the backtracking probability, we can expect a rapid load-balancing of the system.

FCD algorithm is simple, completely local, myopic, and only depends on the current load experienced by each user. An example of the randomized local search mechanism is provided in figure 1. A key design parameter in the algorithm is the exploration probability γ_{2k}. Faster exploration may help the system converge fast but result in system instability with frequent switching (for instance, when $\gamma_{2k} \equiv 1$, the users are constantly switching). Thus, there is a tradeoff between convergence rate and stability. We require $\gamma_{2k} = 2k^{-\beta}$ and $\beta \in [0.5, 1]$. Proofs refer to Lemma 1 and Remark 5. Besides exploration probability, backtracking probability also needs to be carefully selected. We derive the backtracking probability in proposition 1. Notations are explained in Table II.

TABLE II: Frequently Used Notation table 1

Notation	Explanation
$\mathcal{H}_i^j(t)$ [1]	$\{X_j(t) \geq X_i(t)\}$
$\mathcal{M}_i^j(t)$	$\left\{X_i(t)\left(1-\frac{n\gamma_t}{n-1}\right) < X_j(t) < X_i(t)\right\}$
$\mathcal{L}_i^j(t)$	$\left\{X_j(t) \leq X_i(t)\left(1-\frac{n\gamma_t}{n-1}\right)\right\}$

1: $\widehat{\mathcal{H}}_i^j(t)$ denotes $\left\{\widehat{X}_j(t) \geq X_i(t)\right\}$ and so forth.

Proposition 1: At backtracking time slots $t = 2k+1$, the backtracking probability that user a switches back from server j to server i, $f_{ji}\left(X_i(2k), X_j(2k)\right)$, is

$$f_{ji} = \begin{cases} 1, & \mathcal{H}_i^j(2k) \\ 1 - \dfrac{n-1}{n\gamma_{2k}}\left(1 - \dfrac{X_j(2k)}{X_i(2k)}\right), & \mathcal{M}_i^j(2k) \\ 0, & \mathcal{L}_i^j(2k) \end{cases} \quad (1)$$

for $0 \leq \gamma_{2k} \leq \frac{n-1}{n}$.

Proof Ideas: Given that user explores with probability γ_{2k} at $t = 2k$, expected load of each server at $t = 2k+1$ can be calculated. Users choose to backtrack with probability f_{ji} so that $\mathbb{E}[X_i(2k+2)] = \mathbb{E}[X_j(2k+2)]$ for any $i, j \in 1, 2, \ldots, n$. Note that upon finding a worse load, it is always efficient to switch back. Refer to Appendix I [1] for detailed proof.

Estimation of $\mathbf{X_j(2k)}$ *:* The backtracking probability f_{ji} derived in Eqn. (1) is a function of $\{X_i(2k), X_j(2k), \gamma_{2k}\}$ but the real observation is $\{X_i(2k), X_j(2k+1), \gamma_{2k}\}$. Estimating load $X_j(2k)$ and plugging in $\widehat{X}_j(2k)$ are needed to get the real backtracking probabilities.

Noting that in large server loads regime, i.e., $X_i(2k), X_j(2k) \to \infty$, we can accurately estimate the loads (in the sense of MMSE). Estimator

$$\hat{X}_j(2k) = X_j(2k+1) \quad (2)$$

is used. Details refer to Appendix II [1]. As a result, the real backtracking probability $f_{ji}(X_i(2k), X_j(2k+1))$ is obtained:

$$f_{ji} = \begin{cases} 1, & \widehat{\mathcal{H}}_i^j(2k) \\ 1 - \dfrac{n-1}{n\gamma_{2k}}\left(1 - \dfrac{X_j(2k+1)}{X_i(2k)}\right), & \widehat{\mathcal{M}}_i^j(2k) \\ 0, & \widehat{\mathcal{L}}_i^j(2k) \end{cases} \quad (3)$$

for $0 \leq \gamma_{2k} \leq \frac{n-1}{n}$. Notations are explained in Table II.

We make this approximation precise in following analysis by alluding to concentration bounds for the estimates. Recall that the notion of ϵ-Nash equilibrium is described in Definition 1. We use the standard potential function from the load balancing game [5], given by

$$\Phi(\mathbf{X}(t)) := \sum_{i=1}^{n}(X_i(t) - \bar{X})^2, \tag{4}$$

to assess the state of the system, where $\bar{X}$ is the average load of the servers $\bar{X} := \frac{1}{n}\sum_{i=1}^{n} X_i(t)$.

III. Convergence Time Results for Closed Systems

We now state some of the main results of the paper. We provide bounds of the convergence time to reach ϵ-Nash equilibrium under FCD algorithm.

Theorem 1 (Closed System Convergence Time): **Let T be the number of rounds needed to reach an ϵ-Nash equilibrium for the first time, the upper bound of $\mathbb{E}[T]$ is**

$$\mathbb{E}[T] \leq \max\left\{ n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}, \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}} \right\}$$

when $\beta \in [\frac{1}{2}, 1]$ and $m > n^2$. We also have the following concentration bound for the potential function

$$\Pr\left\{ \left| \frac{\Phi\left(X(T)\right)}{m^2} - \frac{\epsilon^2}{4n^2} \right| \geq \theta \right\} \leq e^{-2\theta^2}.$$

Proofs are provided in Section VI.

Remark 1:

1) We derive a non-asymptotic upper bound on the expected convergence time to ϵ-Nash equilibrium (in contrast to asymptotic bounds derived in [5]). The actual convergence time concentrates around the expected value, and the probability that the deviation is at most d decays exponentially with d. Approximate load balancing can be achieved when each user employs an unbiased estimator of the explored server load. To obtain accurate estimates, we require concentration bounds to hold, as analyzed below. This leads to an additional term in the convergence time, given by $\left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}$. In the regime with large number of users $(m \gg n)$, this term is negligible and the convergence rate coincides with the scenario where perfect information about the system load are available to the users.
2) The use of estimated server loads in place of the true server loads leads to small perturbations in the transition probabilities of the Markov chain. We can limit its effect on the convergence time using concentration bounds in the large-m regime.
3) If we set $\epsilon = \frac{n}{m}$, potential function scales as $\mathbb{E}[\Phi(X(T))] = O(n)$,

$$\mathbb{E}[T] \leq \max\left\{ n\log 4m^2 + \left(\tfrac{n}{2}\right)^{\frac{1}{\beta}}, \left(\tfrac{n^3}{m^3}\log 4n^2m^2\right)^{\frac{1}{\beta}} \right\}.$$

IV. Convergence Time Results in Open Systems

We now extend our analysis to open systems where users can arrive and depart from the system. We first describe the system model and then undertake analysis of the convergence time to ϵ-Nash equilibrium. We establish that our mechanism is fairly robust to user dynamics.

A. System Model

a) Arrival Process: The basic model consists of parallel servers and a single stream of arrivals. We consider a simple *load-oblivious* routing scheme where arriving users choose servers uniformly at random. Our convergence time guarantees in the next section demonstrates that the design of routing schemes is not crucial for achieving rapid convergence to an approximately load balanced state.

A slotted system with an initial load of m users is considered. Let $A(t)$ denote the number of arrival users in time slot t and let $B_i(t)$ denote the number of arrival users routed to server i. Due to uniform random access, it's obvious that $\mathbb{E}[B_i(t)] = \frac{A(t)}{n}$. Denote the load at server i at slot t as $X_i^+(t) = X_i(t) + B_i(t)$. The arrival process $A(t)$ can be arbitrary and only needs to satisfy some condition (as a function of t) in order to have rapid convergence to an approximately load balanced state. It is reasonable to assume that new comers mimic the behaviors of existing users and switch at similar rates with existing ones.

b) Departure Process: In real systems, users frequently depart the servers after their tasks are completed. We consider a load-dependent departure process, where the departure rate increases as the server load becomes heavier. This model is consistent with many systems with autonomous agents who are discouraged to stay in servers with heavy loads. Let $C_i\left(t\right)$ denote the number of departure users at server i. Three constraints are imposed on $C_i\left(t\right)$:

- Constraint 1: $C_i\left(t\right) \leq X_i^+(t)$.
- Constraint 2: If $X_i^+(t) \leq X_j^+(t)$, then $C_i\left(t\right) \leq C_j\left(t\right)$.
- Constraint 3: If $X_i^+(t) \leq X_j^+(t)$, then $C_{ij}\left(\lambda X_i^+(t) + (1-\lambda)X_j^+(t)\right) \leq \lambda C_i\left(t\right) + (1-\lambda)C_j\left(t\right)$, for $0 < \lambda < 1$.

As a result, the new load in a server i can be achieved and is denoted as $X_i^F(t)$. Note that for simplicity, arrival and departure processes are in the beginning of the super time slot without loss of generality. Notation $Z_i(2k) = \sum_{l=0}^{k}\left(B_i(2l) - C_i(2l)\right)$ is used.

An open system with dynamic users is thus considered. We assume that all users implement the randomized local search mechanism in Algorithm 1. Let $M(t)$ denote the number of users in the system at time t. We have

$$\mathbb{E}\left[M(t)\right] = \left[M(t-1) + A(t-1)\right] - \sum_{i=1}^{n} C_i\left(X_i^+(t)\right).$$

We analyze the load information of each server. For server i and j, it can be seen that

$$\frac{\mathbb{E}[X_j^F(2k+1)]}{X_i^F(2k)}\Bigg| \{X_i(2k), X_j(2k+1), Z_i(2k)\} = \frac{X_i(2k+1) + Z_i(2k)\left(1-\gamma_{2k}\right) + \left(M(t) - Z_i(2k)\right)\frac{\gamma_{2k}}{n-1}}{X_i(2k) + Z_i(2k)}. \tag{5}$$

c) Notion of ϵ-Nash equilibrium : We use a similar notion of ϵ-Nash equilibrium for open systems, m is the initial number of users in the system.

Definition 2: An ϵ-Nash equilibrium of the network is reached at time slot t, if

$$\max_{ij} \left|X_i^F(t) - X_j^F(t)\right| < \max\left\{\epsilon\frac{m}{n}, \epsilon\frac{M(t)}{n}\right\}.$$

Hence, under an ϵ-Nash equilibrium, the potential function satisfies

$$\Phi\left(X^F(t)\right) \leq \max\left\{\left(\frac{\epsilon m}{2n}\right)^2, \left(\frac{\epsilon M(t)}{2n}\right)^2\right\}.$$

B. Convergence Time Results

***Theorem 2 (Open System Convergence Time):* In open system with initial load m and real time load $M(t)$, if**

$$\Pr\left\{\left|M(t) - \left(1 - \frac{\gamma_t}{n-1}\right)m\right| > n^{-\frac{1}{2}}m^2\sqrt{\gamma_t}\right\} \leq \frac{\epsilon^2}{4n^2} \tag{6}$$

the upper bound of expected convergence time $\mathbb{E}[T]$ is

$$\mathbb{E}[T] \leq \max\left\{n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}, \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}, \left(\frac{n^4}{\epsilon m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}\right\} \tag{7}$$

when $\beta \in [\frac{1}{2}, 1]$ and $m > n^2$.

Remark 2:

1) We demonstrate that rapid approximate load balancing is feasible in open systems under suitable conditions on the arrival and departure process and with a large number of initial users in the system (i.e. $m > n^2$) even under a load-oblivious allocation of the arriving users to the servers. We allow for time-dependent arrivals and departures as long as no tremendous gap exists between $M(t)$ and m.
2) The constraint in (6) on the arrival process implies that the numbers of arrivals and departures are more stringently constrained as time proceeds.

Corollary 1:

- For uniform and independent arrivals with fixed support $[0, \ldots, K]$ and uniform independent departures at a uniform rate of ν at all the servers in each time slot, we can have rapid convergence if $K \leq m^2\sqrt{\gamma_t n}$ and $\nu \leq 1 - \frac{1}{1+\sqrt{\gamma_t n}\frac{m}{2}}$.
- When the arrival process $A(t)$ is discrete Weibull distribution [6], the reliability function is given by $A(t) = \Pr\{A(t) > a\} = q^{a^k}$, where $q \in [0, 1]$ and k is the shape parameter. We observe that we can allow for heavy-tailed processes ($k < 1$) and yet achieve rapid convergence to approximate Nash equilibrium as long as the departure process is also heavy tailed[3].

An overview of the proof techniques follow in section VII. In a nutshell, we demonstrate that the arrival and departure processes lead to perturbations in the transition probabilities between the closed and the open system. We prove that the perturbations are small under the above conditions, so the dominant term of the potential function is the same as it is in the closed system with an initial number of m users. In other words, the convergence time is not tremendously affected by the presence of arrival and departure processes. Rapid convergence is feasible through distributed load balancing.

V. Experiments

A. Setup

Our objective is to simulate and analyze the excepted convergence time $\mathbb{E}[T]$ under different n, m, β settings. So the FCD algorithm is executed 120 times for each setting and averaged. For each execution, initial system loads are generated randomly. The potential function is monitored to identify system convergence: If the potential function is under the threshold n ($\epsilon = \frac{n}{m}$), the system is known as load balanced. As describe above, m and n are large enough and $m > n^2$. The exploration probability is set to decay polynomial with time, i.e. $\gamma_t = t^{-\beta}$. For the open system, we assume a uniform arrival process whose support is $[0, K]$ and a departure rate of $\nu \leq 1 - \frac{1}{1+\sqrt{\gamma_t n}\frac{m}{2}}$. Thus, $K \leq m^2\sqrt{\gamma_t n}$. In this section, we assess the average performance of our distributed algorithm via computer simulation in MATLAB.

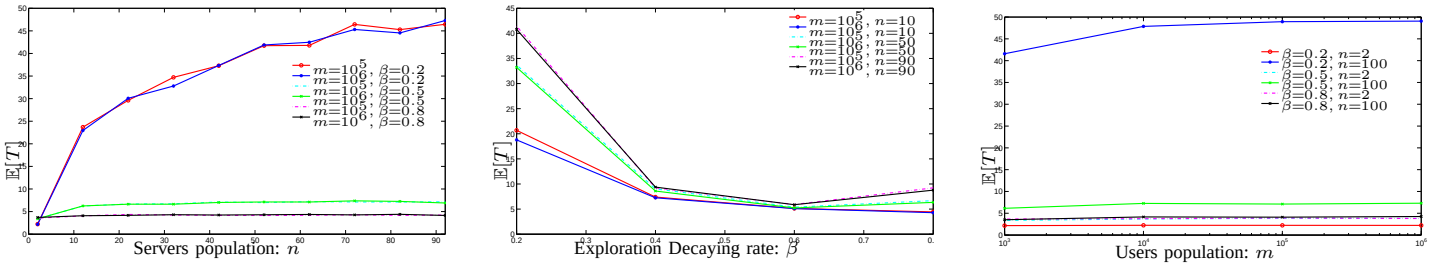
B. Results

First, how the expected convergence time $\mathbb{E}[T]$ of FCD algorithm varies with different server numbers n is evaluated in Fig 2a. We observe that the expected convergence time $\mathbb{E}[T]$ grows linearly with the number of servers n. In other words, convergence time can be limited to an acceptable quantity as long as n being small, which is promising in real systems.

The relationship between decaying rate of exploration probability β and expected convergence time $\mathbb{E}[T]$ is estimated in Fig 2b. One can conclude that the slower the exploration probability decays, the slower the system converges. This means that the switching cost is rather high since users have to switch over to observe and predict the load ratio. Rapid and frequent exploration would make the prediction $\hat{X}_j(2k)$ more involved.

Robustness of the algorithm is observed in Fig 2c. Obviously, the expected convergence time $\mathbb{E}[T]$ is not sensitive to user population m, which is good news to large scale user

[3]Probability density function for Weibull distribution could be accessed from Wikipedia http://en.wikipedia.org/wiki/Weibull_distribution



(a) Converge linearly with server population. (b) Converge slower with smaller decay rate. (c) Convergence time is robust with m.

Fig. 2: Expected convergence time evaluation under different settings.

systems. However, since the expected convergence time $\mathbb{E}[T]$ is bounded by Eqn. (7), the convergence progress would be slow when user population m becomes extremely large.

VI. Performance Analysis for Closed Systems

A. Transition Probability

TABLE III: Frequently Used Notation table 2

Notation	Explanation
$j \in \mathcal{A}_i(t)$[1]	$\left\{j : X_j(t) \leq X_i(t)\left(1-\frac{n\gamma_t}{n-1}\right)\right\}$
$j \in \mathcal{B}_i(t)$	$\left\{j : X_i(t)\left(1-\frac{n\gamma_t}{n-1}\right) < X_j(t) < X_i(t)\right\}$
$j \in \mathcal{C}_i(t)$	$\left\{j : X_i(t) \leq X_j(t) < \frac{X_i(t)}{1-\frac{n\gamma_t}{n-1}}\right\}$
$j \in \mathcal{D}_i(t)$	$\left\{j : X_j(t) \geq \frac{X_i(t)}{1-\frac{n\gamma_t}{n-1}}\right\}$

1: For any servers i, j.

Proposition 2: In the super time slot $\{2k \to 2k+1\}$, the transition probability $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ is given by

$$p_{ij} = \begin{cases} 0, & \widehat{\mathcal{H}}_i^j(2k), j \neq i \\ \frac{1}{n}\left(1-\frac{X_j(2k+1)}{X_i(2k)}\right), & \widehat{\mathcal{M}}_i^j(2k), j \neq i \\ \frac{\gamma_{2k}}{n-1}, & \widehat{\mathcal{L}}_i^j(2k), j \neq i \\ 1-\sum\limits_{j\in\widehat{\mathcal{B}}_i(2k)} \frac{1}{n}\left(1-\frac{X_j(2k+1)}{X_i(2k)}\right) & \\ \quad -\sum\limits_{j\in\widehat{\mathcal{A}}_i(2k), j\neq i} \frac{\gamma_{2k}}{n-1}, & j = i. \end{cases} \tag{8}$$

where $\widehat{\mathcal{H}}_i^j(2k)$, $\widehat{\mathcal{M}}_i^j(2k)$, $\widehat{\mathcal{L}}_i^j(2k)$, $\widehat{\mathcal{A}}_i(2k)$, $\widehat{\mathcal{B}}_i(2k)$ mean that these terms are based on the estimation, $\hat{X}_j(2k) = X_j(2k+1)$, instead of the true value, $X_j(2k)$.

Proof: The transition probability that user a switches from server i to server j in a super time slot is nothing but

$$p_{ij} = \begin{cases} \frac{\gamma_{2k}}{n-1}\left(1-f_{ji}\right), & i \neq j \\ \left(1-\gamma_{2k}\right)+\gamma_{2k}f_{ji}, & i = j \end{cases}.$$

So equation (8) is obtained by plugging in backtracking probability in Equation (3).

$p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ is the probability that user a who resides in server i switches to server j during $2k \to 2k+1$ slots. For users in server i, there are users whose local exploration and backtracking time slots are not synchronized. However, $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ is user-oriented probability based on the observations of user a. More precisely, $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ is equivalent to $p_{ij}\left(Y_a(2k), Y_a(2k+1)\right)$. Obviously, synchronization is not necessary in this distributed system. □

Remark 3: No global clock is needed to synchronize the system. Users need no global knowledge of user number in the system as well.

B. Potential Function Threshold

The potential function $\Phi(\mathbf{X}(t))$ is nothing but un-normalized variance at t, however transition probability is selected by comparing $X_i(t-1)$ $(= X_i(2k))$ and $X_j(t)(= X_j(2k+1))$. To manipulate this contradiction, we first estimate algorithm performance assuming knowledge of $X_j(t-1)(= X_j(2k))$, and then derive the perturbation in convergence time T with knowledge of $X_j(t)(= X_j(2k+1))$ only. Notations are explained in Table III.

Observation 1: Expected load on server i under the condition that the last state of the network is x is

$$\mathbb{E}[X_i(2k+2)|X(2k) = x] \leq \sum_{\substack{l=1 \\ l\neq i}}^{n} \frac{\gamma_{2k}}{n-1}x_l - \sum_{l\in\mathcal{A}_i(2k)} \frac{2\gamma_{2k}}{n-1}x_l + x_i.$$

Proof Ideas: It is known that $\mathbb{E}[X_i(2k+2)|X(2k) = x] = \sum_{l=1}^{n} x_l p_{l,i}(x)$. So by simply substituting $p_{li}, l \in 1, 2, \ldots, n$, the observation is proved. For detailed proof, refer to Appendix III [1].

Observation 2: Variance of load on server i under the condition that the last state of the network is x is

$$\mathrm{Var}\left[X_i(2k+2)|X(2k) = x\right] \leq \sum_{l\in\mathcal{B}_i(t)\bigcup\mathcal{C}_i(t)} \frac{\gamma_{2k}}{n-1}x_l + \sum_{l\in\mathcal{D}_i(t)} \frac{\gamma_{2k}}{n-1}x_l - \sum_{l\in\mathcal{A}_i(t)} \frac{\gamma_{2k}}{n-1}x_l + x_i$$

Proof Ideas: It is known that $\mathrm{Var}\left[X_i(2k+2)|X(2k) = x\right] = \sum_{l=1}^{n} x_l p_{li}(x)\left(1-p_{li}(x)\right)$. Utilizing the facts that $p_{li} \leq 1$ and $1-p_{li} \leq 1$, we get the upper bound. For detailed proof, refer to Appendix III [1].

According to the observations, a recursive relationship between the potential functions at consecutive time slots is achieved in Proposition 3.

Proposition 3: Given $\gamma_{2k} = \frac{1}{m}\sum_{a=1}^{m}\gamma_{2k}(a)$, potential function satisfies

$$\mathbb{E}[\Phi(X(2k+2))] \leq \left(\frac{\gamma_{2k}}{n-1}\right)^2 \left[\left(\frac{n-1}{\gamma_{2k}}-1\right)^2 + \left(1-\frac{n\gamma_{2k}}{n-1}\right)^2 n^2 - 4\left(\frac{n-1}{\gamma_{2k}}-1\right)\left(1-\frac{n\gamma_{2k}}{n-1}\right)\right]\Phi(X(2k)),$$

if perturbation between $\min_a \gamma_{2k}(a)$ and $\max_a \gamma_{2k}(a)$ is small.

Proof Ideas: It is known that

$$\mathbb{E}[\Phi(X(2k+2))|X(2k)] + n\overline{X}^2 \leq \sum_{i=1}^{n}(\mathbb{E}[X_i(2k+2)|X(2k)])^2 + \sum_{i=1}^{n}\mathrm{Var}[X_i(2k+2)|X(2k)],$$

so $\mathbb{E}[\Phi(X(2k+2))|X(2k)=x]$ can be obtained by plugging in above observations. Knowing that $\Phi(X(2k) = x) = \sum_{i=1}^{n}(x_i - \bar{x})^2$, the relation between $\mathbb{E}[\Phi(X(2k+2))]$ and $\Phi(X(2k))$ is achieved. Note that even when $\gamma_{2k}(a)$ are different for different users $a \in 1, 2, \ldots, m$, the proposition still holds if $\gamma_{2k} = \frac{1}{m}\sum_{a=1}^{m}\gamma_{2k}(a)$ and perturbation between $\min_a \gamma_{2k}(a)$ and $\max_a \gamma_{2k}(a)$ is small. Detailed proof can be found in Appendix IV [1].

We now apply the inductive hypothesis to get an upper bound of convergence time for our algorithm.

Lemma 1: A sufficient condition on the parameter β for system convergence is $\beta \in [\frac{1}{2}, 1]$. In this region, for all $t' < t$, $\mathbb{E}[\Phi(X(2t))] \leq (1-\frac{1}{n})^t \mathbb{E}[\Phi(X(0))]$ where $t > \left(\frac{n}{2}\right)^{\frac{1}{\beta}}$.

Proof: Refer to appendix V [1]. □

Remark 4: Lemma 1 explains why we need polynomial decaying exploration rate. For instance, the potential function will not be shrinking with time if we set the exploration probability as e^t.

We can now provide an upper bound on convergence time.

Lemma 2: There is a $\tau \leq \left\{n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}\right\}$ such that $\mathbb{E}\left[\Phi\left(X\left(2\tau\right)\right)\right] \leq \frac{\epsilon^2 m^2}{4n^2}$ in the sufficient feasible region $\beta \in [\frac{1}{2}, 1]$.

Proof: It is important to see that the system is in ϵ-Nash equilibrium if potential function is smaller than or equal to $\frac{\epsilon^2 m^2}{4n^2}$. This results from the definition of ϵ-Nash equilibrium.

Obviously, for $\tau \leq \max\left\{n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}\right\}$, either there is a $\mathbb{E}[\Phi(X(2\tau))] \leq \frac{\epsilon^2 m^2}{4n^2}$ or there is a $\mathbb{E}\left[\Phi\left(X\left(2\tau\right)\right)\right] > \frac{\epsilon^2 m^2}{4n^2}$.

In the case of the potential function being smaller than $\frac{\epsilon^2 m^2}{4n^2}$, we already reached an ϵ-Nash equilibrium based on Definition 1.

In the other case, we take $t = \tau$, then

$$\mathbb{E}[\Phi(X(2t))] \leq \left(1-\frac{1}{n}\right)^t \Phi(X(0)) \ .$$

Even in the worst case that $\Phi(X(0)) = m^2$, i.e. $\mathbb{E}[\Phi(X(2t))] = \left(1-\frac{1}{n}\right)^t m^2$,

$$t = \left(\frac{n}{2}\right)^{\frac{1}{\beta}} + \frac{\log\frac{4n^2}{\epsilon^2}}{\log\frac{n}{n-1}}$$

can guarantee $\mathbb{E}\left[\Phi\left(X\left(2t\right)\right)\right] \leq \frac{\epsilon^2 m^2}{4n^2}$.

So there must be a τ satisfying $\tau \leq \max\left\{\left(\frac{n}{2}\right)^{\frac{1}{\beta}} + n\log\frac{4n^2}{\epsilon^2}\right\}$ such that $\mathbb{E}\left[\Phi\left(X\left(2\tau+1\right)\right)\right] \leq \frac{\epsilon^2 m^2}{4n^2}$.

The lemma is proved. □

Remark 5: As long as user a's exploration decaying rate β is in the feasible region, lemma 2 holds. We allow for different β's for different users.

Overall, we have proved that, according to our proposed algorithm an ϵ-Nash equilibrium is reached within an expected convergence time $\left\{n\log\frac{4n^2}{\epsilon^2} + \left(\frac{n}{2}\right)^{\frac{1}{\beta}}\right\}$. We can see that as ϵ decreases, i.e., a more stringent load balancing criterion is imposed, the convergence is slower.

C. Perturbation Guarantees

Analysis in Section VI-B are based on $p_{ij}(X_i(2k), X_j(2k))$, but we only have access to $X_i(2k)$ and $X_j(2k+1)$. We now provide guarantees on the perturbation between $p_{ij}(X_i(2k), X_j(2k+1))$ and $p_{ij}(X_i(2k), X_j(2k))$.

Proposition 4: Perturbations in transition probability decays exponentially with m:

$$\Pr\left\{\left|p_{ij}\left(X_i(2k), X_j(2k+1)\right) - p_{ij}\left(X_i(2k), X_j(2k)\right)\right| > \delta\right\} \doteq e^{-2m\delta^2}$$

Proof Ideas: Chernoff-Hoeffding bound.

Let us denote $\widehat{\Phi}(X(2k+2))$ as the potential function with knowledge of $X_i(2k)$ and $X_j(2k+1)$. Similarly, $\Phi(X(2k+2))$ is the potential function with knowledge of $X_i(2k)$ and $X_j(2k)$. The perturbation between the two potential functions is analyzed below.

Lemma 3: If perturbation in transition probability $\delta \leq \sqrt{\frac{\gamma_{2k}m^2}{n^3}}$, then the potential function has negligible perturbation and satisfies proposition 3, lemma 1 and lemma 2. However, when $\delta = \sqrt{\frac{\gamma_{2k}m^2}{n^3}}$, $t \leq \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}$ is required for ϵ-Nash equilibrium.

Proof Ideas: On one hand, if the perturbation between $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ and $p_{ij}\left(X_i(2k), X_j(2k)\right)$ is small(which is of high probability), the dominant term in potential function has not changed. As a result, the expected convergence time is not changed. On the other hand, if the perturbation between $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ and $p_{ij}\left(X_i(2k), X_j(2k)\right)$ is large(which is high unlikely), some extra time $t \leq \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}$ for convergence is needed. Detailed proof can be found in Appendix VI [1].

With lemma 2 and 3 at hand, theorem 1 is proved.

VII. Performance Analysis for Open Systems

In this section, we prove theorem 2. Firstly, we prove that there is only negligible perturbation between $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ and $p_{ij}^F\left(X_i^F(2k), X_j^F(2k+1)\right)$ under certain constraints on $A(t)$ and $\sum_{i=1}^{n} C_i(t)$. Secondly, it is shown that convergence time perturbation between open and closed system is small.

A. Small perturbation of transition probability

Guarantees on the perturbation between $p_{ij}\left(X_i(2k), X_j(2k+1)\right)$ and $p_{ij}^F\left(X_i^F(2k), X_j^F(2k+1)\right)$ is given in this subsection.

Proposition 5: Perturbations in transition probability decays exponentially in M:

$$\begin{aligned}&\Pr\{\bigcup_{ij}|p_{ij}\left(X_i(2k), X_j(2k+1)\right)\\&-p_{ij}^F\left(X_i^F(2k), X_j^F(2k+1)\right)\big| > \delta\}\\&\leq \Pr\left\{\bigcup_{ij}\left|\frac{Z_j(2k)+\frac{m\gamma_{2k}}{n-1}}{X_i(2k)+Z_i(2k)}\right| > n\delta\right\} + n^2 e^{-2M(2k)\delta^2}.\end{aligned}$$

Proof Ideas: Chernoff-Hoeffding bound. Detailed proof can be found in Appendix VII [1].

B. Small perturbation of potential function

Let us denote $\widehat{\Phi}(X^F(2k+2))$ as the potential function with knowledge of $X_i^F(2k)$ and $X_j^F(2k+1)$. Similarly, $\widehat{\Phi}(X(2k+2))$ is the potential function with knowledge of $X_i(2k)$ and $X_j(2k+1)$. The perturbation between the two potential functions is analyzed below.

Lemma 4: If perturbation in transition probability $\delta \leq \sqrt{\frac{\gamma_{2k}m^2}{n^3}}$, potential function has negligible perturbation and satisfies proposition 3, lemma 1 and lemma 2. However, when $\delta = \sqrt{\frac{\gamma_{2k}m^2}{n^3}}$, $t \leq \left(\frac{n^3}{m^3}\log\frac{4n^4}{\epsilon^2}\right)^{\frac{1}{\beta}}$ is required for ϵ-Nash equilibrium.

Proof Ideas: Proof idea is the same as it is in lemma 3. Detailed proof can be found in Appendix VIII [1].

Therefore, we can conclude that our algorithm is robust with the small amount of arrival and departure users in the system.

VIII. Conclusion

In this paper, we undertook a careful analysis of the convergence rate of randomized local search mechanism for load balancing in large closed and open systems. Our results demonstrate that simple mechanisms can achieve rapid convergence to an approximate load balanced state. Moreover, we demonstrate the robustness of randomized local search to inaccurate server load measurements, user dynamics and decaying explorations by the users. Our study paves way to a number of interesting questions. To what extent, can we have noise in the server load measurements? Can we extend our analysis to the case with variable server quality of service, possibly in a user-dependent manner? In this case, there are multiple Nash equilibria. Is it possible to design mechanisms which choose an efficient equilibrium and can we provide convergence rate guarantees? The answers to these questions will not only deepen our understanding but also have impact on designing efficient resource allocation mechanisms in real systems.

References

[1] Appendix for INFOCOM '13 submission. http://newport.eecs.uci.edu/anandkumar/pubs/Infocom13-supp.pdf.

[2] D. M. A. P. A. Ganesh, S. Lilienthal and F. Simatos. Load balancing via random local search in closed and open systems. In *Proceedings of ACM Sigmetrics*, 2010.

[3] A. Anandkumar, N. Michael, and A. Tang. Opportunistic Spectrum Access with Multiple Users: Learning under Competition. In *Proc. of IEEE INFOCOM*, San Deigo, USA, March 2010.

[4] A. Anandkumar, N. Michael, A. Tang, and A. Swami. Distributed algorithms for learning and cognitive medium access with logarithmic regret. *Selected Areas in Communications, IEEE Journal on*, 29(4):731–745, 2011.

[5] P. Berenbrink, T. Friedetzky, L. Goldberg, P. Goldberg, Z. Hu, and R. Martin. Distributed selfish load balancing. In *Proc. of ACM-SIAM symposium on Discrete algorithm*, pages 354–363, 2006.

[6] C. Bracquemond and O. Gaudoin. A survey on discrete lifetime distributions. In *Int. J. Reliability, Quality, Safety Engineering*, 2003.

[7] E. Even-Dar and Y. Mansour. Fast convergence of selfish rerouting. In *Proc. of the 16th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2005.

[8] S. Fischer, H. Räcke, and B. Vöcking. Fast convergence to wardrop equilibria by adaptive sampling methods. In *Proc. of ACM symposium on Theory of computing*, pages 653–662, 2006.

[9] P. Goldberg. Bounds for the convergence rate of randomized local search in a multiplayer load-balancing game. In *Proc. of the 23rd Annual ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing (PODC)*, pages 131–140, 2004.

[10] J. M. Ho-Lin Chen and A. Wierman. On the impact of heterogeneity and back-end scheduling in load balancing designs. In *Proceedings of IEEE Infocom*, 2009.

[11] R. Kleinberg, G. Piliouras, and E. Tardos. Multiplicative Updates Outperform Generic No-regret Learning in Congestion Games. In *Proc. of ACM Symp. on theory of computing (STOC)*, Bethesda, MD, May-June 2009.

[12] K. Liu and Q. Zhao. Decentralized Multi-Armed Bandit with Multiple Distributed Players. *submitted to IEEE Transactions on Signal Processing*, Oct. 2009.

[13] B. P. M. Bramson, Y. Lu. Randomized load balancing with general service time distributions. In *ACM Sigmetrics*, 2010.

[14] M. Mitzenmacher. The power of two choices in randomized load balancing. *IEEE Transactions on Parallel and Distributed Systems*, pages 1094–1104, 2001.

[15] R. Rosenthal. A class of games possessing pure-strategy nash equilibria. In *International Journal of Games Theory 2*, 1973.

[16] D. Shah and J. Shin. Dynamics in congestion games. *ACM SIGMETRICS Performance Evaluation Review*, 38(1):107–118, 2010.

[17] A. T. Bonald, M. Jonckheere. Insensitive load balancing. In *ACM Sigmetrics*, 2004.

[18] G. K. A. G. J. L. A. G. Y. Lu, Q. Xie. Join-idle-queue: A novel load balancing algorithm for dynamically scalable web services. In *29th International Symposium on Computer Performance, Modeling, Measurements, and Evaluation*, Oct. 2011.

[19] A. W. S. L. Zhenhua Liu, Minghong Lin and L. L. H. Andrew. Greening geographical load balancing. In *Proceedings of ACM Sigmetrics*, 2011.