Appendix C

Theorems and Lemmas

**Theorem 4.1**: Given a cluster and a parallel application submitted to cluster, time complexity of MALB algorithm is $O(nm)$, where $n$ is the number of hosts in the cluster and $m$ is the number of tasks in the application and values of $n$ and $m$ should be larger than 2.

**Proof**: It takes $O(1)$ time to compute the response time of a task on a host. The time complexity to determine that a local host is overloaded is $O(n)$. Since there are $n$ hosts in cluster, it takes $O(n)$ time to choose most appropriate host with minimum load. Hence time complexity of balancing I/O resources is $O(2+2n)$. Similarly time complexity of balancing memory and CPU resources are $O(2+2n)$. Since there are $m$ tasks in the applications, hence time complexity of MALB algorithm is $O(2(1+n)m)$. The value of $n$ and $m$ is much larger than 2, so time complexity of the algorithm is $O(nm)$.

**Theorem 4.2**: Consider a network of $N$ servers with same mean arrival $\lambda$ and service time $\mu$ at each server, $N$ is a power of 2, then average number
Proof: We will prove the result by induction on number of levels

\[ K = 1, 2, \ldots, \log_2 N - 1 \]

**Step 1: Basis of Induction**

For \( K = 1 \), we have only two servers one of which is leader (say host 0). Whenever arrival or departure of tasks occur at leader no message is sent otherwise as soon as some event takes place appropriate messages are sent to leader and the corresponding hosts. Probability of an event occurring at either host in small interval is 0.5. So average number of messages after the occurrence of an event is

\[
\frac{\log_2 2^1}{2} = \frac{1}{2}.
\]

**Step 2: Induction hypothesis and proof**

Let the result is true for \( K = U \). Consider two sub trees \( A \) and \( B \), each having \( U \) levels and \( 2^U \) leaves. Occurrence of event at leaf of \( A \) or \( B \) requires exchanging of \( \frac{\log_2 2^U}{2} = \frac{U}{2} \) messages at the most. Now join \( A \) and \( B \) sub tree to one containing \( 2^{U+1} \) leaves, then number of levels increases to \( U+1 \). Again one of two conjugate hosts at level \( U \) is leader say root of sub tree \( A \). Then every leaf of sub tree \( A \) still has to send \( \frac{U}{2} \) messages after each event, but every leaf of sub tree \( B \) has to send one extra message to leader of new sub
tree. Hence probability of occurrence of an event in each sub tree is

\[ \frac{0.5}{2} U + 0.5 \left( \frac{U}{2} + 1 \right) = \frac{U + 1}{2} = \frac{\log_2 2^{U+1}}{2} \]

Hence the proof of the theorem. Using this theorem we can conclude that a task transfer time demands in total \( \log_2 N \) update messages.

**Theorem 6.1:** The traffic for a host located at a distance \( r \) from the centre of the disk can be expressed by the following expression:

\[
\left( \pi R^2 \delta - 1 \right) \lambda + \frac{\pi \left( R^2 - r^2 \right) \delta^2 \beta}{2}
\]

**Proof:** Let us denote by \( A \), a host located at a distance \( r \) from the centre of the disk. Let us define the following notation- \( x(i) \) is a point on the edge of the disk such that angle between \( Ax(i) \) and the axis \( OA \) is equal to \( i \). Consider \( S_\alpha d(\alpha) \), the portion of disk centre around \( x(\alpha) \) with a aperture of \( d\alpha \). Our goal is to determine the amount of traffic generated by source host \( S_\alpha d(\alpha) \) and going through host \( A \). We use shortest path routing mechanism to prove the result{ Figure 1}. Then the problem reduces to determine the portion of the disk containing all possible destination hosts corresponding with source hosts in \( S_\alpha d(\alpha) \) through host \( A \). If the routes were perfect straight lines then destination area would be the portion of the
disk centered on $A$, $x(\alpha + \pi)$ with aperture $d\alpha$. But the routes are not straight lines, so the destination area would be larger than this i.e. $S_{\alpha+\pi}(d\alpha + \beta)$ with $\beta$ being the small positive real number, independent of $\alpha$ and $d\alpha$.

The value of $\beta$ is depending upon network density and host distribution.

![Diagram](image)

**Figure 1:** Traffic Analysis of the disk for shortest path

Let us now evaluate $S_{\alpha}d(\alpha)$ and $S_{\alpha+\pi}(d\alpha + \beta)$. Since $d\alpha$ is very small, the following expressions holds-

\[
\sin(d(\alpha)) = d(\alpha)
\]

\[
Ax(\alpha - d(\alpha)) = Ax(\alpha), \ Ax(\alpha + d(\alpha)) = Ax(\alpha)
\]

\[
S_{\alpha}d(\alpha) = \frac{Ax(\alpha - d(\alpha)) \times Ax(\alpha + d(\alpha)) \sin(d(\alpha))}{2}
\]
From the above equations we can conclude that 

\[ S_{\alpha} d(\alpha) = \frac{Ax(\alpha)^2 d(\alpha)}{2} \]

Similarly we can get 

\[ S_{\alpha + \pi} (d\alpha + \beta) = \frac{Ax(\alpha + \pi)^2 (d\alpha + \beta)}{2} \]

Assuming a uniform distribution of hosts in the dish then, then the number of hosts in \( S_{\alpha} d(\alpha) / 2 \) and \( S_{\alpha + \pi} (d\alpha + \beta) \) will be respectively \( S_{\alpha} d(\alpha) \delta \) and \( S_{\alpha + \pi} (d\alpha + \beta) \delta \), so the number of routes going through host \( A \) would be:

\[
N = S_{\alpha} d(\alpha) \delta \times S_{\alpha + \pi} (d\alpha + \beta) \delta \\
= \frac{Ax(\alpha)^2 d\alpha \times Ax(\alpha + \pi)^2 (d\alpha + \beta) \delta^2 (d\alpha^2 + d\alpha\beta)}{4}
\]

When \( d\alpha \) is very small, then \((d\alpha^2 + d\alpha\beta) = \beta d\alpha\)

Hence 
\[
N = \frac{Ax(\alpha)^2 d\alpha \times Ax(\alpha + \pi)^2 (d\alpha + \beta) \delta^2 \beta d\alpha}{4} \]

…………………..(1)

In order to solve the problem we have to prove the following:

For any line \( B1C1 \) going through host \( A \), we have:

\[ AC \times AB = AC1 \times AB! = (R^2 - r^2) \]
Figure 2: Analysis of a lie going through host A

Also from the Figure 2, we can see that triangle $ABB_1$ is similar to $ACC_1$ so

$$\frac{AB_1}{AC} = \frac{AB}{AC_1}$$

Hence $AB \times AC_1 = AB \times AC = R^2 - r^2$. Since $Ax(\alpha)$ and $Ax(\alpha + \pi)$ are on the same straight line it can be seen from equation (1)
\[ N = \frac{\pi(R^2 - r^2)\delta^2 \beta d\alpha}{4} \]

Since the traffic is bidirectional, taking integration for \( d\alpha \) from 0 to \( \pi \), the relay traffic going thorough host \( A \) will be:

\[ \text{Relay traffic} = \frac{\pi(R^2 - r^2)\delta^2 \beta \lambda}{2} \]

The traffic on each host includes the following traffic, traffic from other hosts and relay traffic. Since the circle is of radius \( R \), the area is \( \pi R^2 \). So the numbers of host in the circle are: \( \pi R^2 * \delta \). Hence, there are \((\pi R^2 \delta - 1)\) hosts communicating with the current host with traffic rate \( \lambda \). The traffic on a host with distance \( r \) from the centre is:

\[ \text{traffic} = \text{common traffic} + \text{relay traffic} = \left(\pi R^2 \delta - 1\right)\lambda + \frac{\pi(R^2 - r^2)\delta^2 \lambda \beta}{2} \]

Hence the proof of the theorem.

**Lemma 8.1:** At the end of first step, every process has one temporary checkpoint with same \( \text{chk\_no} \) and this set is consistent.

**Proof:** Every process has one temporary checkpoint. Since, all the processes had the same \( \text{chk\_no} \) at the beginning of the check pointing process, all the agents will carry the identical value of \( \text{chk\_no} \). So, the \( \text{chk\_no} \) f the new set of checkpoints will be same too.
Lemma 8.1: In the first step, if the number of concurrent initiations is $k$, the total number of moves by all the agents is $O(kn)$.

Proof: Since an agent carries the list of visited processes, the agent travels along the edges of the tree only, and an edge of the tree is traversed at most twice. So, for each agent, the number of hops traversed is $O(n)$. Thus, the total number of hops traversed by all the agents is $O(kn)$. It may be noted that the total number of hops cannot be claimed to be $O(n)$ as the processes visited by the different agents are not exclusive.

Lemma 8.3: In the second step, the total number of moves for the agents is $O(n)$.

Proof: All but one agent travel along disjoint parts of the network graph. The total number of moves of these agents is $O(n)$. The other agent travels also travels along path on the reduced graph and hence has $O(n)$ moves.

Theorem 8.1: At the end of second step, every process has one permanent checkpoint with same $chk\_no$ and this set establishes a CGS of the system.

Proof: we know that every process has one temporary checkpoint with the same $chk\_no$ after the first phase. Also all initiators are visited by an agent from the minimum id initiator and guarantees all processes in the system confirm their checkpoints. So at the end of second phase, every process has confirmed checkpoint generated in the first phase. Since processes take checkpoints induced by application messages, the system will not have any orphan message.