A Multi-host Cluster Load Balancing Scheduling Method

Shu HU, Li LIU, Tingxun LIU
College of computer, Sichuan University, Chengdu 610064, China

ABSTRACT: For some applications system, the demand for high reliability, high stability, high robustness, and discusses how to design and implement a high availability cluster system based on multi-host of the characteristics of such applications. Paper elaborated structure and design method of the cluster communication interface system and focuses on the cluster load balancing algorithm used to make cluster technology to meet the needs of specific applications.
KEYWORD: cluster; process group; load balancing; scheduling algorithm

1 INTRODUCTION

1.1 *Back ground*

In a large distributed system, typically configured with multiple high-end servers, these servers are generally in accordance with the nature of the master-slave configuration of dual-hosts form, in any master-slave servers unit, the two hosts install the same software, which runs all on a single host-based application process.

The existence of this operating mode a few drawbacks: 1. When the host standby units are both fault, the system will appear in the case of major loss of function, but in large systems are generally more than two servers, the fault can not be run on a host of features runs well on other servers, so the form performance of redundancy is not high; 2. Master-slave unit in standby mode process were the main run on a single host, always to cause the host CPU load is too high, Another host because all processes running from relatively low state, the load; host task uneven distribution of high CPU load is also likely to cause the system to slow processing, scheduling, reverse other adverse consequences

1.2 *Related works*

The approach described in this paper allows multiple servers to deploy the same application systems, through the application process grouping, implemented on multiple hosts on the primary process group allocated from the state of equilibrium; Avoid single point of failure in a multi-host cluster, the processes running on the host-based state migrated to run on other hosts; when a primary state application processes on a server exits, its functionality migrated to another host load is relatively low. The algorithm describes how the process is specified from the state primary strategy in a multi-host cluster environment [1].

2 SYSTEM OVERVIEW

2.1 *Cluster system overview*

The multi-host cluster approach be able to support 2-32 hosts, these hosts can be in the hardware configuration, operating system platform inconsistent, but the unanimous request of the deployment process (different operating systems require the same process functions consistent). You can specify the main program from the state to host a different process, and has strong fault tolerance.

2.2 *Multi-host cluster deployment*

The cluster control software runs on each cluster host, the software provides versions for different platforms, exactly the version of the kernel[2] [3].

Cluster system provides a programming interface cluster_client library development, each process requires the use of cluster functions use the library, and in accordance with the interface requirements, call interface to obtain a master-slave status, timing reporting interface status.

Cluster hosts using the same cluster configuration file, specify the host inter-cluster heartbeat interval

and process group division and other information in the configuration file.

2.3 *Process group*

Process according to the way the group is organized to meet: a few inter-process communication is based on the machine's IPC (inter-process communication), if the primary distribution of these processes from different hosts, there will be a primary state transfer process information sent to slave-state process and subsequent treatment of the issue can not be, that is, data supply chain scission. So the process of scheduling granularity should be set when the host all processes as a group, equivalent to switch to the host units. In the multi-host cluster system and multi-process environment, process group master and slave state distribution is shown in Figure 1.

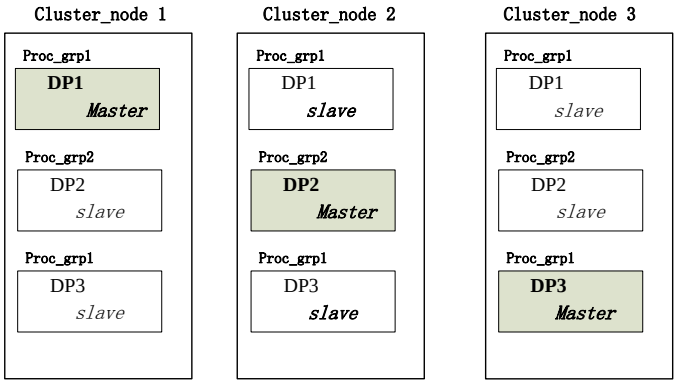

Fig.1. Process group master and slave state distribution in a multi-host cluster system

Process group is scheduling unit in cluster system, the system supports 100 process groups. Each group of processes running when you run properly, you can keep a run priority, if the process exits or functional failure, the priority is zero; each group according to the data of the supply chain process priority order from high to low arrangement, if a process group consisting of n processes, then processes the data supply chain top priority 2n, second priority 2n-1 and so on. Thus, the total priority process group can obtain and compare different priorities can host the same process group.

Load factor in the operation of the system after the host process is running stable set of processes using process group processes and CPU usage increases.

2.4 *Heartbeat Message*

Between the host heartbeat packets through the transmission of information, because the host is generally only equipped with two serial ports, in a multi-host environment, cluster do not use the traditional RS232 interface, heartbeat packets using network packet transfer mode, you can use UDP or data link layer protocol, T1 sent once every so often. Unlike heartbeat information when dealing with general network data and then processed into the message queue after the first, but processing immediately after receiving heartbeat messages, the heartbeat message are defined as follows:

```
typedef struct {
   uint64 pack_count;
   time_t   start_time;
   int32   host_dev_code;
   char ready_status;
   bool bIsManager;
   int32   wManagerDevice;
   char   group_num;
   struct {
   int32 wGroupCode;
   tagRowColBit tagAllRunProcPrior;
   uint8 btStatus;
}group_info[MAX_GROUP];
  }ClusterHeartBeat;
```

Among them: packet_count is a continuous cumulative counter; start_time is the local host start-up time; host_dev_code is number of each host in the cluster that does not repeat; ready_status for the host is ready to join the cluster when the state, when the host discovered cluster already exists that is the main management unit or when their main management machine that the machine is ready to participate in the cluster scheduling; bIsManager identify whether the machine management unit; wManagerDevice the machine where the main cluster management unit; number group_num native process group; local_ready oriented ready information in the initial phase of host initiated, cluster management process will wait for some time, this time under normal circumstances this machine all managed processes are started and begin to stabilize the running time, after this time to enter the main management unit determination work; group_info for each process group operating status of each process, which **tagAllRunProcPrior** for the survival status of a process group process, with a bit of each process, where 0 is the exit, 1 for running, in accordance with the process priority bit by byte represented by a low to high, this approach can make the process of comparing the survival level information can be quickly completed in bytes; btStatus for the main process from the state group.

3 IMPLEMENTATION

3.1 *Management host election*

No special host in the cluster as a manager, management machine is produced by the cluster hosts the elections, as follows: 1.a start-up time (in seconds) to act as the management of small host machine; 2. If the some hosts have same start time,

select the call to the host IP address string values to perform inet_addr get smaller host to act as manager. In accordance with this method is always to elect one and only one manager.

Any cluster host when it starts running at a specified time T2 (usually a multiple of the heartbeat cycle) time waiting for other hosts heartbeat information that already exists at the time of receipt of the packets manager ended prematurely wait for the machine managed machine; while other hosts can not receive heartbeat information for the management of the local machine.

When the cluster is running, the cluster can respond to reassign from the monitoring system for the primary management unit, then the current primary cluster system immediately reduced to general ma host, and then upgrade to the competent authority designated host. When this new master manager has not failed, the master manager does not follow the current selection algorithms.

3.2 *Communication between Cluster mana-gement software and managed processes*

Communication based on local loopback (127.0.0.1) TCP protocol cluster management software and managed the process of inter-cluster management software for server-side TCP, and is located on the client management process， detail described in section4.2

Between cluster management software and managed process includes the following communication message: 1. a timing master and slave status reporting **ProcInfoRepor**t, sent from the management process to manage software; 2 The main-slave state assignment packets **StatusDef** sent from the management process to manage the process. ProcInfoReport message contents is defined as follows:

```
typedef struct    {
   unsigned short proctype;
   int pid;
   unsigned char pristat;
}ProcInfoReport;
```

Among them: proctype managed process is client process type, which is specified by the internal application systems, pid is the process ID, obtained through the operating system, pristat managed process is the current master-slave state, and its value for the following three values: primary state, secondary state, intermediate state; management process was initially reported as an intermediate state runtime, which is equivalent to the management process to apply a master-slave states.

StatusDef message is very simple, only one byte, there are three kinds of information possible: primary state, secondary state, intermediate state.

3.3 *Load balancing cluster scheduling algorithm*

Sufficient condition for the cluster system switching process groups are: a current primary state process group "survival priority factor" lower (due to client process exit or process-related hardware failure), and its priority factor has been below or equal to the current the process group of other servers priority factor; 2 state process group from the priority factor increased (due to exited process restart running or process-related hardware failure recovery), making it a priority factor has been greater than or equal to the process group of other servers priority factor.

The program cluster load balancing algorithm[4] [5] [6] used is a basic principle of the algorithm, a recursive algorithm is used to switch the primary state process group higher load from the server to the process group with the same priority load factor and the whole sum of the minimum server up until algorithm can no longer switch so far.

The main idea of this algorithm is as follows: Define a maximum load value MAX_LOAD, the value is large enough not appear in the system, when the cluster system meet the conditions of load balancing switch, Cluster managers find the current load is less than the parameter specified load (MAX_LOAD) server in the server cluster highest load on the host system, by which the formation of a "maximum load server list" and then turn to traverse the list each server process is the primary state group. Let n be the number of a process group maximum load on the server is in primary state, so look for survivors in the current state of the server process with the main group n the same process server with the lowest load factor group priority. If the server which has lowest load factor is found eligible, the load difference between the two servers is calculated. If the load is less than the difference between the switched load before switching difference, then after switching has become more balanced load, then switch, and so on, until you have all the primary state traversal process group on the host so far. If primary-secondary state switch in the traversal process, then a recursive algorithm jump out the "maximum load server list" cycle, because once upon time a switch happen, the maximum load on the server in the list of server load aggregate the system may no longer be the highest load, so the re-processing into the next recursion, parameter passing at this time is still MAX_LOAD, until complete traversal process group primary state on all the highest load host. If you traverse completed without switching occurs, then it will be the highest point in the system load is passed to the next recursive worth as a parameter calculation. Conditions of recursion end is: could not find the server in the cluster has a smaller load than the total value of the parameter passed in the. At the moment each host cluster system load reaches the maximum degree of balance. This workflow is shown in Figure 2.

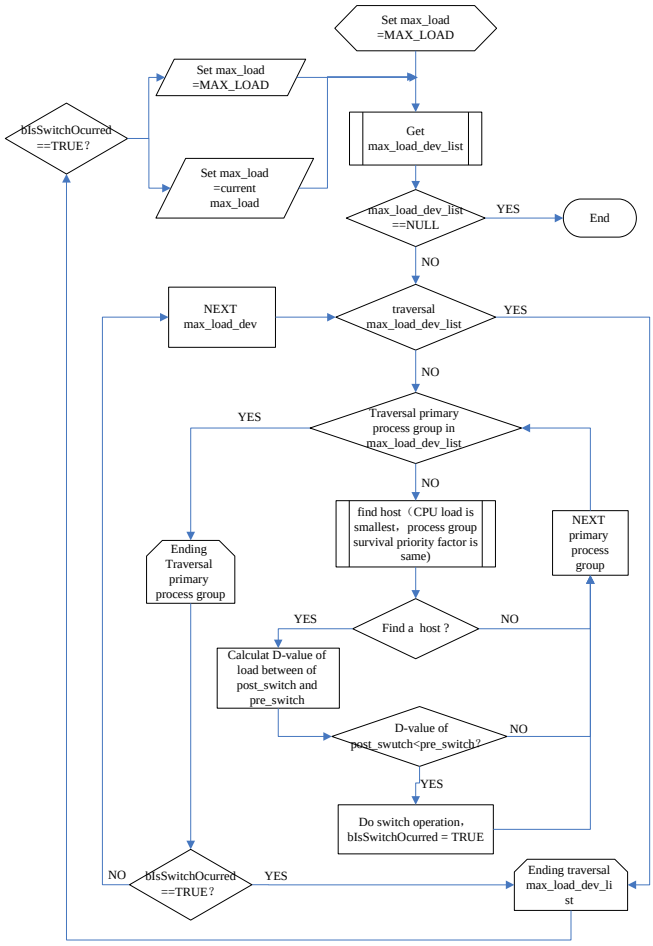

Fig.2. Flowchart of cluster load balancing scheduling algorithm

Scheduling algorithm flowchart is a replica of the current cluster host and process group information (here referred to as S2) operation, which is the process of implementing switched as follows: one by one copy and the system compares the current cluster host and process group information (here referred to as S1), a copy of the record is the result of last switched on by the primary state to which the process of state from the first issue of the group switch command, and then enter the wait, after the completion of the first round switch (in this case the state for S3) re-introduction of the first Second round switch, switch from S1 to S2 when needed from the state switched from the primary state process group, but the conversion process is completed, the two-phase switching system in order to make the problem does not appear the two primary state process group. In switching from S1 to S2 process, cluster management unit of the main cluster of external control commands are not processed, for heartbeat packets received recording time only, purpose of doing so is to avoid from S1 to S2 switching process caused by the new scheduler.

4 CONCLUSION

The algorithm is different from general image processing and other mathematical algorithms. On the implementation of the algorithm at the cluster management software, at the same time the algorithm appeal recursive algorithm, but also include the reasons for triggering process, processes running, abnormal exit examination, abnormal checking etc.

In the implement of the cluster software, in addition we also offer the following algorithm design: 1.Cluster management peer need to wait for a configurable time period, in this time period to wait for all client processes establish a TCP connection to clsuter management software, this time is a test value from these client processes usually able to run and connection; 2. After the cluster management software through after this time, began to enter the selection of the main management host, then issued consecutive heartbeat information, as requested manager machine messages, when the main management unit receives the message, found that the host only native state of readiness and the cluster is not ready (ie, waiting for the main management unit gives answer) then immediately send heartbeat messages to the aircraft, ending their wait and call the load scheduling algorithm specified master-slave state. 3.During operation, no heartbeat if we find a host of information, a process exit / restart, or receive a mandatory requirement for manual intervention will be called load balancing is load balancing algorithm, but after the implementation of the two-step algorithm calls as previously describe.

REFERENCES

[1] Cluster manager, http://en.wikipedia.org/wiki /Cluster_ manager, July 2012

[2] R. Davis and A. Burns. A Survey of Hard Real-Time Scheduling Algorithms and Schedulability Analysis Techniques for Multiprocessor Systems. Technical Report YCS-2009-443, Dept. of Computer Science, University of York, 2009.

[3] Y.Etsion and D. Tsafrir, A Short Survey of Commercial Cluster Batch Schedulers. Technical Report 2005-13, The Hebrew University of Jerusalem, May 2005.

[4] BARUAH, S.K., Techniques for Multiprocessor Global Schedulability Analysis. In proceedings of 28th IEEE Real-Time Systems Symposium, 2007, pp. 119-128..

[5] LIN, X., LU, Y., DEOGUN, J., AND GODDARD, S., Real-time divisible load scheduling for cluster computing. In proceedings of 13th IEEE Real-Time and Embedded Technology and Applications Symposium, 2007,pp. 303–314.

[6] LUNDBERG, L., Analyzing Fixed-Priority Global Multiprocessor Scheduling. In proceedings of 8th IEEE Real-Time and Embedded Technology and Applications Symposium, 2002, pp. 145–153 .