Experience With The Condor Distributed Batch System

Mike Litzkow
and
Miron Livny

Computer Sciences Department
University of Wisconsin — Madison
mike@cs.wisc.edu
miron@cs.wisc.edu

1. Introduction

Many organizations now own hundreds of powerful workstations which are connected by local area networks. It is common practice in such organizations to allocate each of these workstations to a single user who exercises full control over the workstation's resources. In such an environment we can find three types of users, *casual* users who seldom utilize the full capacity of their machines, *sporadic* users who for short periods of time fully utilize the capacity of the workstation they own, and *frustrated* users who for long periods of time have computing demands that are beyond the power of their workstations. Unlike the two other groups, the throughput of these frustrated users is limited by the power of their workstations. They often claim that their productivity could be significantly enhanced if they had access to the unutilized computing capacity of workstations owned by casual and sporadic users. Condor is a distributed batch system that was designed to meet the challenge posed by these users, namely to provide convenient access to unutilized workstations while preserving the rights of their owners. The current version of Condor was installed in our department for public use in the summer of 1988. It has since served more than 144,000 jobs that have consumed more than 6,000 days of CPU.

2. Guiding Principles

Several principles have driven the design of Condor. First is the principal that workstation owners should always have the resources of the workstation they own at their disposal. This is to guarantee immediate response, which is the reason most people prefer a dedicated workstation over access to a time sharing system. The second principle is that access to remote capacity must be easy, and should approximate the local execution environment as closely as possible. Portability is the third principle behind the design of Condor. This is essential due to rapid developments in the workstations on which Condor operates. For example, the earliest version of Condor became operational in 1984 when it ran on a network of 20 VAX 11/750s[1]. Six years later we are running Condor on our network of over 200 workstations ranging from MC68000[2] based workstations to MIPS[3] and SPARC[4] workstations running a rainbow of UNIX[5] variants; none of the original machines is still connected.

3. Mechanisms Used

Five mechanisms are basic to the operation of Condor. The first is a mechanism for determining when a workstation is not in use by its owner, and thus should become part of the pool of available machines. This is accomplished by measuring both the CPU load of the machine, and the time since the last keyboard or mouse activity. The default is to consider a machine ‘‘idle’’ when the CPU load average as measured by UNIX is less than 0.3, and the keyboard and mouse have been idle for at least 15 minutes. Individual workstation owners can customize each of these parameters as they deem appropriate.

Second is a mechanism for ‘‘fair’’ allocation of these machines to users who haved queued jobs. This task is handled by a centralized ‘‘machine manager’’. The manager allocates machines to waiting users on the basis of priority. The priority is calculated according to the *up-down* algorithm. This algorithm periodically increases the priority of those users who have been waiting for resources, and reduces the priority of those users who have received resources in the recent past. The purpose of the algorithm is to allow heavy users to do very large amounts of work, but still protect the

---

[1] VAX is a trademark of Digital Equipment Corporation

[2] MC68000 is a trademark of Intel Corporation.

[3] MIPS is a trademark of MIPS Computer Corporation.

[4] SPARC is a trademark of Sun Microsystems.

[5] UNIX is a trademark of AT&T Bell Laboratories

response time for less frequent users.

Thirdly, Condor provides a remote execution mechanism which allows its users to run remotely the same programs that they had been used to running locally after only a re-linking step. File I/O is redirected to the submitting machine, so that users don't need to worry about moving files to and from the machines where execution actually takes place. Also by creating Condor programs in this way, users are able to write in a variety of source languages including C, FORTRAN, and special simulation languages such as DeNet.

The fourth mechanism is responsible for stopping the execution of a Condor job upon the first user activity on the hosting machine. As soon as the keyboard or mouse becomes active, or the CPU load on the remote machine rises above a specified level, a running Condor job is stopped. This provides automatic return of the use of the hosting workstation to its owner.

Finally Condor provides a transparent checkpointing mechanism which allows it to take a checkpoint of a running job, and migrate that job to another workstation when the machine it is currently running on becomes busy with non-Condor activity. This allows Condor to return workstations to their owners promptly, yet provide assurance to Condor users that their jobs will make progress, and eventually complete.

To meet the portability requirement, all these five mechanisms were implemented entirely outside the UNIX kernel. We have been able to remove most machine dependent constructs from all of these mechanisms except checkpointing. Even checkpointing is quite portable, but does depend on the specific formats of the ‘‘a.out’’ and ‘‘core’’ files for each system.

4. Experience

The current version of Condor has been in general use by researchers in our department for more than one year. Thousands of CPU days were consumed by Condor jobs over the last year. Condor continuously records the activity of its users and information regarding workstations availability. We have developed tools to extract a wide range of statistics from the recorded data. With the help of these statistics we have profiled the arrival pattern of jobs, the distribution of their processing demands and response time, and the manner in which workstations in our department are used by their owners.

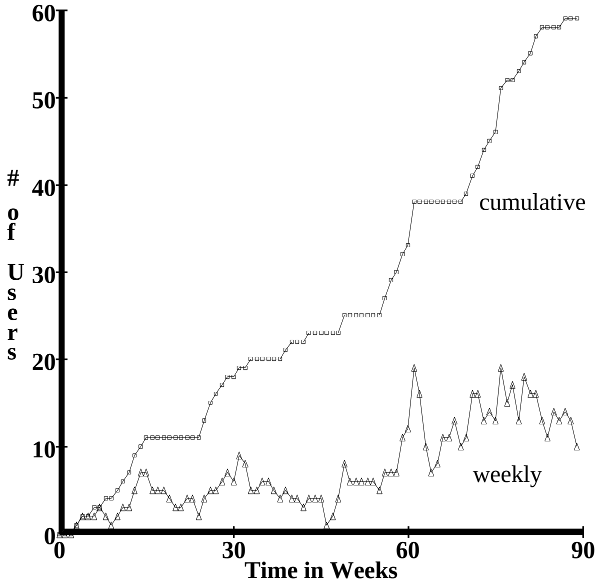
4.1. Support Required

Condor is an experimental system consisting of many pieces of software spread over more than 200 heterogeneous workstations. Also, both the owners of those workstations and the users of Condor vary from neophytes to very sophisticated users. Such a system does not run itself, and a certain amount of human support is required to keep Condor operational and all parties happy. We have been fortunate in that only a few non-trivial bugs have been found since it was released for general use. Due to the complex interaction between the Condor processes and the wide variety of platforms on which Condor runs, however, some of the simple bugs have been fairly difficult to track down. Since new people are trying out Condor all the time, there are frequent questions about how to submit jobs, or what to expect after a job has been submitted. Though such questions generally don't require a great deal of expertise to answer, there are enough of them to require on the average an hour or two a week of support.

The most critical support task is responding to those owners of machines who feel that Condor is in some way interfering with their own use of their machine. Such complaints must be answered both promptly and diplomatically. Workstation owners are not used to the concept of somebody else using their machine while they are away, and are in general suspicious of any new software installed on their systems. The vast majority of the complaints we have received have turned out to be something other than Condor slowing the response time of the machine involved. Still it takes a fairly sophisticated knowledge of both Condor and UNIX to re-construct what happened at the time of the alleged "intrusion", and explain this to the workstation owner. Of course in a few cases Condor was at fault, and we have made several improvements to Condor's "idleness detection" algorithm based on the particular circumstances leading to these complaints.

We have also made the Condor software available to the wider community via anonymous ftp. Over a hundred sites in the US and Europe have picked up Condor so far, though we don't have knowledge of how many sites are actively using it. Time spent helping with installations and answering questions for these sites ranges from a few minutes to several days a month during academic break periods when universities often work on installing new software.

4.2. Condor Users Perspective



Number of Condor Users

Figure 1

Currently we have about 30 regular Condor customers, and on any one day it is likely that 6 to 10 of them will have Condor jobs in the queue. As shown by the Figure 1, these numbers are constantly growing. To these users Condor represents access to ‘‘free’’ machines. Over the past six months, Condor has provided an overall average of three machines to each of its users on demand. This average would be higher, except that there were a large number of requests for only one machine at a time. Naturally, when condor it used to run only a single job at a time, the turnaround time for that job is slightly longer than if the job were run on the submitting workstation in the normal way. The advantage though, is that submittors get their intensive computations done on a remote machine, meanwhile retaining full use of their own machine for interactive work. During ‘‘good’’ periods (nights and weekends), it is not unusual to find 10 or 20 machines working for a single user.

As mentioned earlier, Condor is primarily designed to meet the needs of those users who are frustrated by the limitations of computing on a single workstation. A look at the remote computing capacity consumed in the past year by the top ten Condor users as shown by Table 1 reveals that indeed, these are *heavy* users who would be frustrated with only a single machine at their disposal. The demands made by these users are generally ‘‘bursty’’.

box; cfI s sw(1.6i) n|l|l. Top Ten Condor Users = Rank User Name CPU Consumed
(days+hours:minutes:seconds) _
1 leut 1327+05:41:44
2 kang 1086+05:05:20
3 kessler 484+20:01:36
4 towell 429+13:28:40
5 hsiao 342+07:56:32
6 wood 319+18:51:20
7 miron 262+01:09:42 8 harit-
sa 238+06:32:12 9 denet 224+11:12:16
10 shavlik 223+17:46:44

Table 1

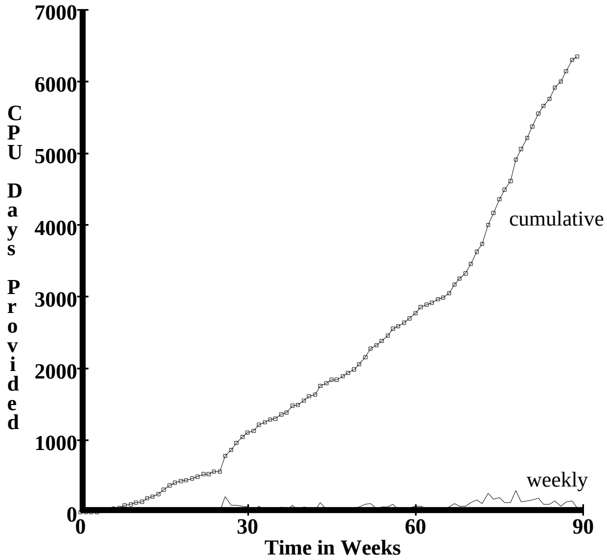
A common pattern is for a user to consume all the computing capacity available for a week or two, then not request any capacity for several weeks. Presumably these users are interpreting results from the last experiments and planning new experiments during their ‘‘quiet’’ periods. Fortunately, it rarely happens that all the Condor users demand cycles at the same time, though the pressure of the academic calendar does tend to cause high demand near the ends of semesters and low demand between semesters.

Condor users very much appreciate the fact that checkpointing guarantees their jobs will complete, even in the face of machine crashes. This is important to users who submit a large number of jobs on Friday afternoon and would like to find them completed on Monday morning, even in the event of a temporary power failure during the weekend. For users who run very long jobs, sometimes taking months to complete, this feature is crucial.

4.3. Machine Donor’s Perspective

Many machine owners who don’t have long running and CPU intensive jobs ‘‘donate’’ their machines to the Condor pool. For these people the reward is in knowing that they are helping out their colleagues by donating unused cycles which would otherwise be wasted. Still, if Condor were interfering with their own use, most people would not want their machine in the pool. Since in our environment membership in the Condor pool is entirely voluntary, the number of people who chose to donate their machines is a good indicator of how well Condor does in avoiding such interference. Out of some 200 workstation owners, only about 5 have prohibited us from running Condor on their machines. Another 5 have noticed some interference, and we have responded by configuring Condor to start jobs on their machines only after the keyboard has been idle for two hours or more, rather than after the default of 15 minutes. This means that 95% of the machine owners have been happily donating their unused cycles without noticing enough interference to even mention it to us.

4.4. System Perspective

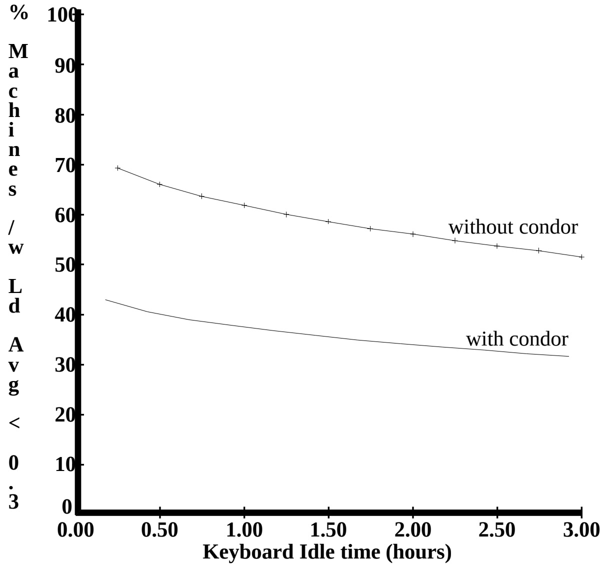


CPU Provided by Condor

Figure 2

From a system administrator's point of view, the effect of Condor is to increase the overall computing capacity of a group of workstations without the purchase of additional hardware. As shown in Figure 2, Condor has provided a total of over 6000 CPU days over the past 90 weeks. To a system administrator, this corresponds to the purchase and maintenance of at least ten additional workstations. Actually if the workstations were put into the hands of individual users, the number would be much larger than that, since as shown in Figure 3, users rarely utilize the full capacity of a workstation over the long run.

Condor calculates whether a machine is idle based on the UNIX load average, and elapsed time since the last keyboard activity. Machines whose load average is below a certain threshold and whose keyboard idle time is above another threshold are considered to be idle. Of course the number of machines which are considered idle at any one time will vary depending on the thresholds used.



System Utilization

Figure 3

Our experience is that a load average of 0.3 and a keyboard idle time of 15 minutes are reasonable choices, and result in a system which is acceptable to most machine donors. Figure 3 summarizes statistics taken over several months on about 70 VAX workstations in our department. For this graph, we fixed the load average requirement at 0.3, and plotted the portion of machines which would be considered idle given various keyboard idle time requirements. The top curve shows shows how many machines would have been considered idle without Condor in the picture, and the bottom curve shows the portion of machines which remained idle even with Condor. Using the 15 minute idleness criteria, we can see that on the average without Condor about 70% of the machines would have been idle. By using Condor, we reduced the portion of idle machines to about 43%. These numbers would be even better, if there were always enough Condor jobs in the queue to use all the available workstations; even the once frustrated users don't keep the system 100% busy in the long run.

5. Future Work

We view Condor as both a "production" tool, and an experimental system. Work on increasing both the power and flexibility of this tool is ongoing. The current version of Condor is limited to running "single process" jobs, though many such jobs may be run in parallel provided they don't need to communicate or synchronize. An example of such a trivially parallel application is a group of simulations all using the same program, but running with different parameters or input files. Still, a much larger field of problems would be open to solution by Condor if the processes could communicate. We are working on a version of Condor which manages such communication using the Linda primitives. Another interesting problem is expanding the Condor pool to even larger groups of machines. Currently all the machines in a Condor pool must communicate with one "central manager", and jobs migrate from machine to machine without regard to the "distance" between various machines. We are looking into ways of decentralizing the control of Condor and expanding to an environment where a wide area network is used to connect "clusters" of workstations.

6. Conclusions

We believe Condor has demonstrated that the challenge of the frustrated users can be met, and has succeeded in meeting a large portion of that need with minimal interference to the workstation's owners. We feel that the need for Condor or a similar system will continue as long as environments where workstations are assigned to ''exclusive'' use by individuals are prevalent. The challenges are to continue to adapt to changes in that environment, and expand the kinds of jobs which Condor can run effectively.

7. Acknowledgements

This research is supported in part by the National Science Foundation under grant DCR-8521228 and by a Digital Equipment Corporation External Research Grant.

The authors gratefully acknowledge the work of Anthony Dayao without whom the statistics and graphs presented here would not have been possible.