CERN-COMPUTING AND NETWORKS DIVISION
CN/95/6
March 1995

Experience of running PIAF on the CS-2 at CERN

Timo Hakulinen & Fons Rademakers

Presented at the HPCN 95 Conference, Milan, May 1995

Experience of running PIAF on the CS-2 at CERN

Timo Hakulinen, Fons Rademakers

Computing and Networks, CERN
CH-1211 Geneva 23
Switzerland

Abstract. A 32-node Meiko CS-2 MIMD machine has been installed at CERN as part of an ESPRIT project funded by the European Community. It is intended that this machine be used by CERN to run codes developed at CERN and elsewhere and to thereby demonstrate the feasibility of using a CS-2 in a demanding high energy physics environment. One part of the project deals with interactive data analysis using PIAF (Parallel Interactive Analysis Facility), a system developed at CERN for carrying out queries on very large databases of several gigabytes in parallel [1].

1 PIAF software

The PIAF system is based on the PAW (Physics Analysis Workstation) data analysis and visualization package also developed at CERN [2]. Whereas PAW is basically intended to run stand-alone on a variety of platforms ranging from small microcomputers to mainframes, PIAF is designed to interface with PAW to give the user transparent access to high-performance computing facilities which are physically located elsewhere. This enables the user to carry out the most time-consuming and data-intensive computations in parallel on a PIAF server which typically consists of a cluster of fast workstations with large disks and interconnected by a fast network.

PIAF uses a client-server/multiple slave server model in managing the parallel computation. When a user initially connects to PIAF from his or her local PAW session, a new PIAF master process is started by the *inetd* super-server. The master in turn initiates slave processes in all PIAF machines and starts waiting for further commands from the user. All communication between PIAF processes is carried out using TCP/IP on Berkeley sockets.

The user can access and manipulate files on the remote PIAF system and view histograms and graphical plots as if all the work were carried out on his or her local workstation. When for example a command for constructing a histogram from a data set is received by the PIAF master, it subdivides the task to all the slaves running in different machines. Thereafter, each slave goes on processing its portion of the data which is mostly independent of the data of the other slaves. After finishing computation each slave reports back to the master who in turn summarizes the results and sends them back to the client.

PAW and PIAF use powerful data constructs called n-tuples for data storage. An n-tuple is basically a data matrix with rows representing events and columns holding individual variables. Various statistical operations can be easily carried out on n-tuples based on any combination of events or variables. N-tuples are stored columnwise which allows access to an individual column without having to read the complete data base. Column-wise n-tuples are very efficient for analyzing only certain variables over the entire data-set, the task most frequently performed.

Histogramming operations usually require multiple passes over the same data. Together with using column-wise n-tuples this allows one to optimize subsequent accesses to the same data by caching the n-tuple data in memory. PAW and PIAF have an adjustable size n-tuple cache which is always checked when accessing n-tuples in case the required data already resides in the cache. The default maximum size for the cache in PIAF is 54 MB per slave server.

Sizes of the user data-sets analyzed with PIAF vary from a few megabytes to several gigabytes. The usual size for a large n-tuple file is 200 megabytes (due to the size of the standard tape in use at CERN), but many such files can be chained to form a "super-n-tuple" which is then seen as one logical data-set by PAW and PIAF. Analysis of these data sets is usually carried out iteratively requiring multiple passes over the data.

2 The CS-2 at CERN

The CS-2 at CERN has 32 scalar nodes of which 4 are 50 MHz and the rest 40 MHz SuperSparc processors. The fast nodes have 64 MB and the rest 32 MB of memory each. All nodes are connected point to point by a 50 MB/s Meiko Elan communications switch. For outside communications an FDDI and a HiPPI connection exist in one of the nodes. Each node has a 1 GB internal disk for scratch space and there are an additional 52 GB in 18 2.9 GB disks for user home directories and mass data storage. The operating system is Solaris 2.3 with system specific additions from Meiko.

3 PIAF on the CS-2

The PIAF software has been ported to the CS-2 and Solaris. A variety of performance measurements have been carried out to determine the optimum parameters for network and disk I/O as well as optimum number of slaves per session and the effect of the system load to the performance.

For the successful operation of PIAF efficient I/O is of primary importance. This consists mainly of fast local and remote disk I/O. Of secondary concern is peer-to-peer TCP/IP communication, which is utilized for internal communication between the PIAF master and slave servers.

3.1 Basic I/O

To understand the effect of some basic factors to performance, measurements on disk and network I/O were first carried out. The measured parameters were TCP/IP peer-to-peer socket data transfer rate and data transfer rates to and from various filesystems in the CS-2. All tests were carried out on the 40 MHz nodes with 32 MB of memory and 150 MB of swap space.

The TCP/IP test consisted of writing to an open socket in one node and reading from it in another. Standard Unix *write* and *read* system calls were used and the test was repeated using several block sizes as well as TCP send/receive buffer sizes. The early results suggested a possible optimization by changing the maximum transfer unit (MTU) parameter which is by default 69554 bytes (64 KB + overhead) for the internal Elan network to 8232 bytes (8 KB + TCP and IP headers). This value is the optimum for the Solaris TCP/IP implementation giving the maximum throughput of 4.5 MB/s from node to node. System profiling indicates that the TCP/IP performance in this case is basically CPU limited and to further increase performance, system level tuning of TCP/IP is required. It would be possible to get throughputs reaching 40 MB/s using the low level Elan functionality, but it was decided at this time to keep the PIAF code on the CS-2 compatible with other systems.

The disk tests consisted of reading and writing 10 MB and 100 MB files to and from different file systems. The tested file systems were: a local unix file system, a set of three disks striped using the Solaris Disksuite volume manager with striping factor of 64 KB, and Meiko parallel file systems consisting of 4, 8, and 16 nodes also with striping factor of 64 KB. Tests were carried out by using the unix system calls *read* and *write*. Without any specific system level tuning the best reading performance for large 100 MB files was achieved locally from the three striped disk set, around 5 MB/s, with the Meiko parallel file systems coming next with around 4 MB/s transfer rate. Transfer rates from the striped disks over NFS are clearly slower, around 2 MB/s. Writing is fastest to the local disks, around 3.5 MB/s, whereas using NFS should in this case be avoided if possible. For small 10 MB files considerably higher transfer rates, up to 25–30 MB/s, can be seen as file system cache effects start playing a prominent role. The performance depends slightly on the used buffer size with larger buffers which are multiples of the file system block size being more favorable. On the CS-2 it is possible to use Meiko specific ioctl-calls to disable NFS client and server caches on a per-file basis. This opens up a way for substantial optimization in the case of Meiko parallel file systems and when using the optimum buffer size which is the file system stripe size times the number of nodes, per process transfer rates up to 14–15 MB/s are achievable. However, this feature hasn't so far been used in the PIAF tests, since it would require several changes to the low level I/O routines PIAF uses.

3.2 PIAF Scalability

The PIAF tests consisted of a standard n-tuple test suite designed to emulate average usage of the PIAF system. First in the test job is a series of queries

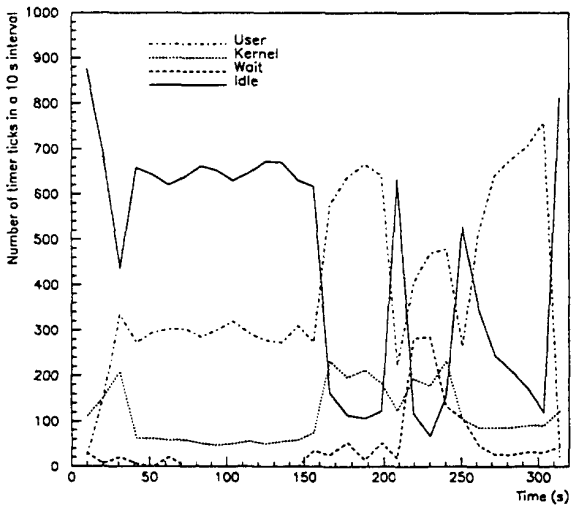

Fig. 1. The CPU utilization profile of a PIAF test with a single job running on 8 nodes with a 20 MB n-tuple cache.

on a small 10 MB n-tuple file followed by a series of queries on a 100 MB chain consisting of ten 10 MB files. The first part of the test is relatively light as far as disk I/O is concerned because the n-tuple generally fits in the memory cache in all cases. However, the ratio of interprocess communication to computation is in this case relatively high. The second part on the other hand mainly stresses the I/O efficiency and CPU processing of the system because in most cases the n-tuple cache gets flushed. During the test a bit more than a half of the data in the actual file needs to be accessed in a random access fashion. In the tests the parameters varied were the number of PIAF nodes (1, 2, 4, 8, 16, 26), the number of concurrent PIAF jobs (1, 2, 4, 8), and the maximum size of the internal n-tuple cache (10 MB, 20 MB, 54 MB). The data files used were located on four striped disk sets physically located on four nodes which are part of the PIAF cluster and mounted everywhere over NFS. The system configuration was similar to the basic I/O tests above. During the tests system and process parameters were recorded in 10 second intervals in all affected nodes. An example of a typical CPU utilization profile is shown in figure 1.

Results show that the speedup of PIAF performance as a function of processing nodes is highly nonlinear as shown in figure 2. The superlinear growth in the initial portion of the curves is a direct consequence of the effective increase in the amount of n-tuple cache over nodes, which allows the complete n-tuples to be cached in memory. This can also be seen in the system profiles as a decrease in I/O wait time due to a more even distribution of disk accesses over several nodes. As the number of nodes is increased further, the limiting factor becomes the interprocess communication between the master and the many slaves. This can be seen clearly as a substantial increase in CPU idle time on the

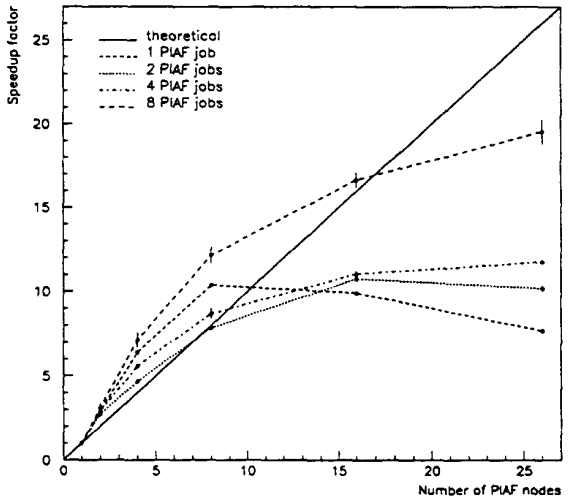

Fig. 2. The speedup of the aggregate PIAF system throughput as a function of PIAF nodes. N-tuple cache size is 20 MB.

slaves while the master is processing the results. The point where the aggregate limit of scalability is reached depends on the number of concurrent PIAF jobs on the system. Since other jobs are able to utilize the leftover CPU idle time, it is in principle more efficient to keep a higher load in the machine. However, since PIAF is basically an interactive service, high system loads often directly translate to poor performance of the system as perceived by the users. This is illustrated in figure 3, which shows the slowdown factor as more concurrent jobs are started on the system. It can be seen that on a small number of PIAF nodes (1–8) the slowdown goes superlinear after 4 jobs. This is mainly due to the relatively small amount of memory in each node which increases paging and thereby the I/O wait time in system. Larger PIAF systems (16–26 nodes) are much more resistant to performance degradation in this case.

4 Conclusions

Based on the above results a few conclusions can be drawn:

- For a PIAF system processing large data sets not fitting in cache I/O and NFS performance as well as the amount of memory and CPU power become limiting factors. This can be helped considerably by enhancing the overall I/O rates as well as using faster CPUs with more memory.
- For small data sets which essentially fit in cache, the bottleneck lies in master-slaves communications. The solution in this case involves enhancing both the message passing efficiency and CPU power especially on the nodes where the masters are to be run.

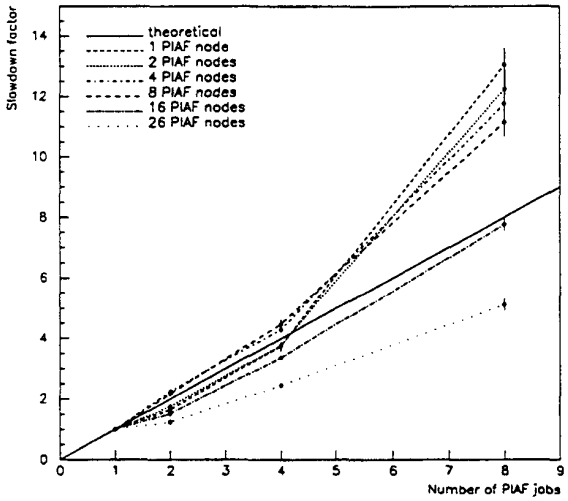

Fig. 3. The slowdown of a single PIAF job as a function of the total number of concurrent jobs in the system. N-tuple cache size is 20 MB.

- The larger the number of PIAF nodes, the higher are the required data rates over NFS. Using the parallel file system should also help considerably here.
- Depending on the expected usage different PIAF configurations can be used. For jobs using small data files and straightforward queries a smaller system is adequate. For large I/O limited jobs or jobs with complex queries a bigger configuration should be used.
- The number of concurrent users in the system is mainly limited by the available memory. In real use PIAF process sizes can grow all the way to the n-tuple cache size which is of the same order as the available memory on one CS-2 node. Although the low system load associated with interactive work would allow more concurrent jobs, increased paging due to small memory limits the performance.

After improving the performance in the above mentioned areas, the CS-2 should prove to be well suited for running PIAF. The coming system upgrade in April '95 where, per node, the CPUs are upgraded to twin 100 MHz SuperSparcs and the main memory increased to 64 MB, should improve the situation considerably. Also the anticipated improvements in TCP/IP performance are very welcome.

References

1. F. Rademakers, *The Parallel Interactive Analysis Facility.* CERN Report, CERN/CN/94/9 (1994).
2. *PAW, Physics Analysis Workstation, The Complete Reference.* CERN Program Library Long Writeup Q121. October 1993.