Experiences with Industrial Applications on Massively Parallel Computers

Jörg Stadler

debis Systemhaus GmbH
Fasanenweg 11
D 70771 Leinfelden-Echterdingen
E-Mail: jstadler@debis.com

Abstract. HWW (Höchstleistungsrechenzentrum für Wissenschaft und Wirtschaft GmbH) is a large german supercomputing facility, that offers its compute resources to scientific as well as industrial users. This article reports on the applications that industrial users run on the HWW equipment and the computer architectures they use. The current situation is characterized by an increasing demand for compute resources from commercial customers which eventually led to the installation of an additional machine in march 1998. But all of this demand is still exclusively met by vector machines, whereas massively parallel machines (MPPs) are still being evaluated by the industrial user community.

1 Introduction

Supercomputers are no commodity products, but highly specialized and also expensive tools. Still their use can be very advantageous for industry, even if one considers small and medium enterprises. But the purchase of such a machine can only rarely be justified for a single organization, and the operation a world class supercomputing center requires even larger investments. Therefore, the idea was born to join the efforts of academia and industry in the HPC area. This idea was implemented in the HWW (Höchstleistungsrechner für Wissenschaft und Wirtschaft, Supercomputers for Science and Industry), which was founded by the state Baden-Württemberg, Porsche AG and debis Systemhaus. HWW operates several supercomputers that are listed in table 1. Within HWW, the state of Baden-Württemberg gives supercomputer access to university research, the Porsche AG acts as end-user and debis Systemhaus offers high performance computing resources and services to commercial customers. It is the goal of debis Systemhaus to provide the European industry with access to state of the aret supercomputing equipment.

Currently the commercial users at HWW are mainly from the automotive and aerospace industries, their size ranging from major European airplane and car manufacturers to small engineering firms.

In this article we give a brief overview of the applications of HWW's customers and describe in more detail our experiences with industrial applications on MPP systems.

Table 1. Machines available via HWW

Machine	CPUs	Memory [GB]	Type	Location
NEC SX4	32	8	PVP	Universität Stuttgart
NEC SX4/A	4	8	PVP	Universität Stuttgart
Cray T3E	512	64	MPP	debis Untertürkheim
Cray T90	4	1	PVP	debis Untertürkheim
Cray C90	1	2	PVP	debis Untertürkheim
Cray C90	3	2	PVP	debis Untertürkheim
Cray J90	8	2	PVP	debis Untertürkheim
IBM RS6000/SP	256	122	MPP	Universität Karlsruhe

2 Current Customer Applications

The application mix currently consists of roughly ten to twelve different codes from areas such as static and dynamic structure analysis, computational fluid dynamics, and of course crash simulation. Most of these are of-the-shelf programs from ISVs, but a few home grown (source-) codes still survived. While the number of different application codes is to be evenly distributed between fluid dynamics and structural analysis, it must be noted, that by far the largest part of the CPU time is used for crash simulation. Crash simulations are nowadays a proven and reliable tool that is employed in vehicle body design. This, together

with an increasing complexity of the numerical models, has led to an ever increasing demand for compute resources in this area. In fact, HWW installed an additional NEC SX4/A vector supercomputer in march 1998 in response to the increasing demand.

3 The Situation of MPPs

When one considers the type of computers used by industrial users, the current situation can be characterized as follows:

- Vector machines are the true and tried workhorses for production calculations.
- Most applications have been tuned for the SMP vector architectures and typically run on up to four processors.
- MPP versions of some codes are available, but not in production use.
- Industrial users start to evaluate MPPs and run first tests or benchmarks.

HWW offers its users access to two large and powerful MPP machines, namely a Cray T3E with 512 nodes and an IBM SP2 with 256 nodes. However, none of these machines is regularly used by commercial customers, their use by industry is still in what might be called an evaluation stage. During the last months there have been several requests from the user community to run benchmarks and demonstrations on the MPP machines.

All applications used today were originally developed on single processor machines and run on vector machines. Most of them are parallelized for SMP architectures, so that they can exploit more than one CPU on the vector machines. For some of them there are also ports to MPP machines available. One the other hand, there is no application that specifically was designed for an MPP nor one that specifically requires an MPP machine. Given that situation, why should an user leave the proven and well known vector machine and move over to an MPP? The two main arguments in favor of MPPs are: First, the high performance that results, if a given application can use a sufficiently large number of processors. Under favorable conditions, an MPP can easily outperform a vector machine. And second, the large, but distributed memory (compare table 1: The two MPPs at HWW have a main memory of 122 GB and 64 GB, respectively, whereas the largest vector machine has an main memory of 8GB).

In the past few month there were several requests that addressed the above points. For example, we ran a fluid dynamics model of a size of 8GB on the Cray T3E.

In the area of crash simulations we experience the fastest growth in demand: The models get more and more complex and also the number of simulations increases dramatically. More specifically, the models grow faster with time than the performance of the vector systems increases. The introduction of SMP software pushed the vector systems to higher levels of performance, but it could well be that the crash models even outgrow these. Therefore we decided to explore the possibility of running crash simulations on MPP systems. The LS-Dyna code had already been ported to MPP systems, but to our knowledge nobody was actually using it for that application.

4 Case study: crash simulations

When we started to try crash simulations on MPPs, the first thing we did was to run several benchmarks in order to assess the feasibility of the project. The results, as table 2 shows, were indeed very promising: They showed that roughly eight processors of an MPP system delivered roughly the performance of one vector processor and they also showed an excellent scalability of the MPP systems.

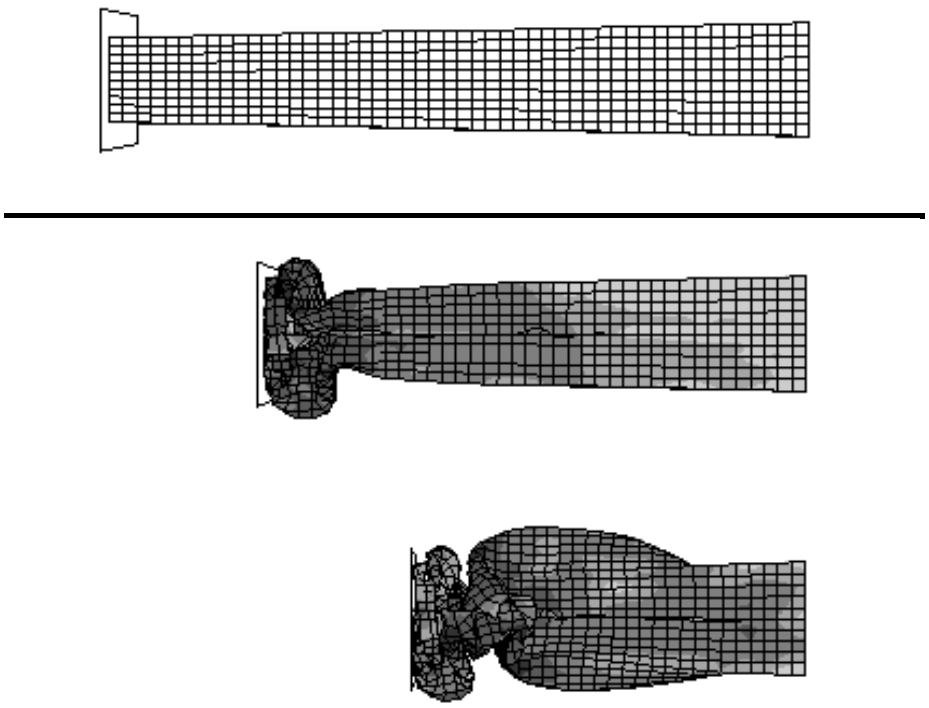
Table 2: Benchmark example for crash simulation

Machine	Processors	Runtime [s]
SGI Origin	4	103.090
SGI Origin	8	53.652
SGI Origin	16	25.997
NEC SX4	1	51.608
NEC SX4	4	25.420

We encountered, however, two major numerical problems: First, we found that the results depend on the number of processors used, but second and worse, we found that we couldn't reproduce our results even when we held the number of

CPUs constant. In fact, we had a situation, where, much to our surprise, the code gave us considerably different results from the same input data on the same machine and with the same number of processors. What had happened? Could it be that MPPs are non-deterministic machines? To study this strange behavior further, we used the small model of a tubular beam shown in Figure 1. The beam fixed on its right end and deformed by a moving rigid wall that hits the beam from the left. Figure 1 shows three consecutive stages from the corresponding calculation.

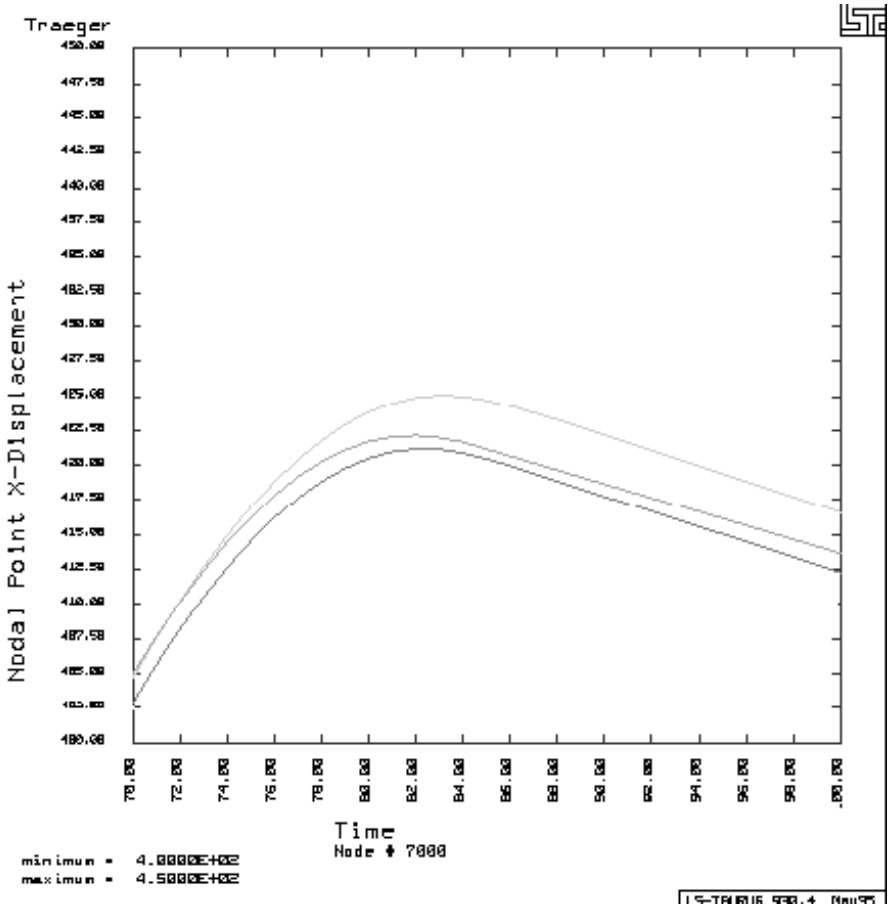
Figure 1: Small test calculation: Deformation of a thin walled tubular beam



The beam deforms until the energy of the impacting wall has been absorbed. The beam shortens by the amount by which the wall is displaced during the calculation. Naturally, this quantity is of vital interest for crash simulations. Figure 2 shows

plots of this displacement during the last phase of the calculation. In figure 2 we have plotted the results of three calculations that we ran consecutively with a constant number of processors. The difference in the results is clearly visible and in the order of roughly 1%. They are of course unacceptable for a calculation that could possibly influence the safety of a car passengers.

Figure 2: Displacement of the wall from three consecutive simulations



These non-reproducible results puzzled us for quite some time. It must be noted, however, that crash calculations are numerically extremely sensitive. One can

easily imagine, that there are several numerically instable situations in a calculation like the one shown in Figure 1. In these situations, even the smallest numerical differences can easily grow to a noticeable size. Now it is well known that in contrast to their mathematical counterparts numerical operations on computers are not associative. So two such simulations have identical results only when they process the same input data in exactly the same order.

The LS Dyna code uses geometric domain decomposition to distribute the data to the available processors. It uses the MPI message passing library for interprocessor communication. From the point of view of a single processor a simulation can be seen as a sequence of calculation and communication steps, which depends on the data that lives on that processor. When we now change the number of processors, the data assigned to a given processor will change and so will its sequence of operations. So the results will change, when one uses a different number of processors. This fact somewhat limits the usability of MPPs: One can't just use some more processors when in a hurry, and some less, when not. Instead, on has to stick to a given number if one needs to compare the results of different runs.

But at least we can understand why the results change with the number of CPUs. But why did the results change from one run to the other, with the same number of CPUs? We checked every possibility we could think of, but couldn't find a solution until the we talked directly to the maintainer of the code: We found out, that he had implemented dynamic load balancing in one part of the program. This is of course, considered good programming practice on MPP systems, but also dynamically changes the sequence of operations from one run to another.

This problem was identified and is now fixed. At this time it is still being worked on, as it lacks some features that are available in the vector version. It will also undergo some additional testing, but we do hope that we can soon use MPP machines for production calculations and we sure have learned, that it takes some time and effort to get production quality results from MPP systems.

5 Conclusion

We have reported on the usage of the supercomputer resources of HWW by industrial customers. The overall acceptance of HWW's supercomputing services by commercial customers is promising: HWW purchased and installed an additional NEC SX4/A in March 1998 to meet their demand for compute

resources. But all of this demand is still exclusively met by vector machines. MPPs are only just evaluated by industrial users for some special cases. The example from the area of crash simulation shows that it still takes some effort to move production applications to MPPs, even after the initial porting of the application was finished.