Numerical comparison of two communication algorithms

Thomas Apel Gundolf Haase Arnd Meyer Matthias Pester

February 1, 1995

The paper [1] deals with the application of domain decomposition methods for the parallel solution of boundary value problems for partial differential equations over a domain $\Omega \subset I\!R^d$, $d = 2, 3$. There, the attention is focused on the conception of efficient communication routines for the data exchange which is necessary for example in the preconditioned cg-algorithm for solving the resulting system of algebraic equations. Essentially, two different algorithms are discussed and the results of few computational tests with both algorithms are sketched. In this note, we show the results of all tests, which were done.

Our test example is to solve

$$\begin{aligned} -\Delta u &= 0 \quad \text{in } \Omega = (0,2)^3, \\ u &= xy \quad \text{on } \partial\Omega, \end{aligned}$$

via the finite element method with linear shape functions on tetrahedral meshes. We generated coarse meshes with 48, 96, 192, 384, and 768 elements and refined them uniformly until the memory was exhausted. Note that the coarse meshes with 384 and 768 elements are different from those after one refinement step of the meshes with 48 and 96 elements, respectively.

These examples were solved on a parsytec GC/PP-128 machine using 8 to 128 processors and as variants 1 and 2 processors per node. In Figures 1–10 the times for communication and for accumulation (communication plus some arithmetics) are shown.

We draw the following conclusions:

- For small and medium numbers of processors Algorithm 7 is better than the combination of Algorithms 4 and 6 regarding the pure communication time and the time for accumulation. The time for arithmetics is much higher of course.
- The situation changes for a high processor number and a large number of unknowns (*2D-Kette*s dominate the *1D-Kette*s). It is very likely that in Algorithm 7 a large number of the quite long *2D-Kette*s has to be communicated in several steps. But in Algorithm 4 they are communicated just once. Here, the possibility of direct links under PARIX is very favourable.

  At the moment, the distribution of the elements to the processors is done in the order that the coarse mesh generator produced. An optimization in the direction to minimize the communication and to minimize the subhypercube dimension (see Remark 3 of [1]) is expected to improve the behaviour of Algorithm 7.
- Comparing the same number of processors in the two variants of one or two processors per node, we see that arithmetic time and communication time show controverse behaviour. This influences the comparison of the two algorithms in a very machine dependent way.

References

[1] Th. Apel, G. Haase, A. Meyer, and M. Pester. Parallel solution of finite element equation systems: efficient inter-processor communication. Preprint SPC95_5, TU Chemnitz-Zwickau, 1995.

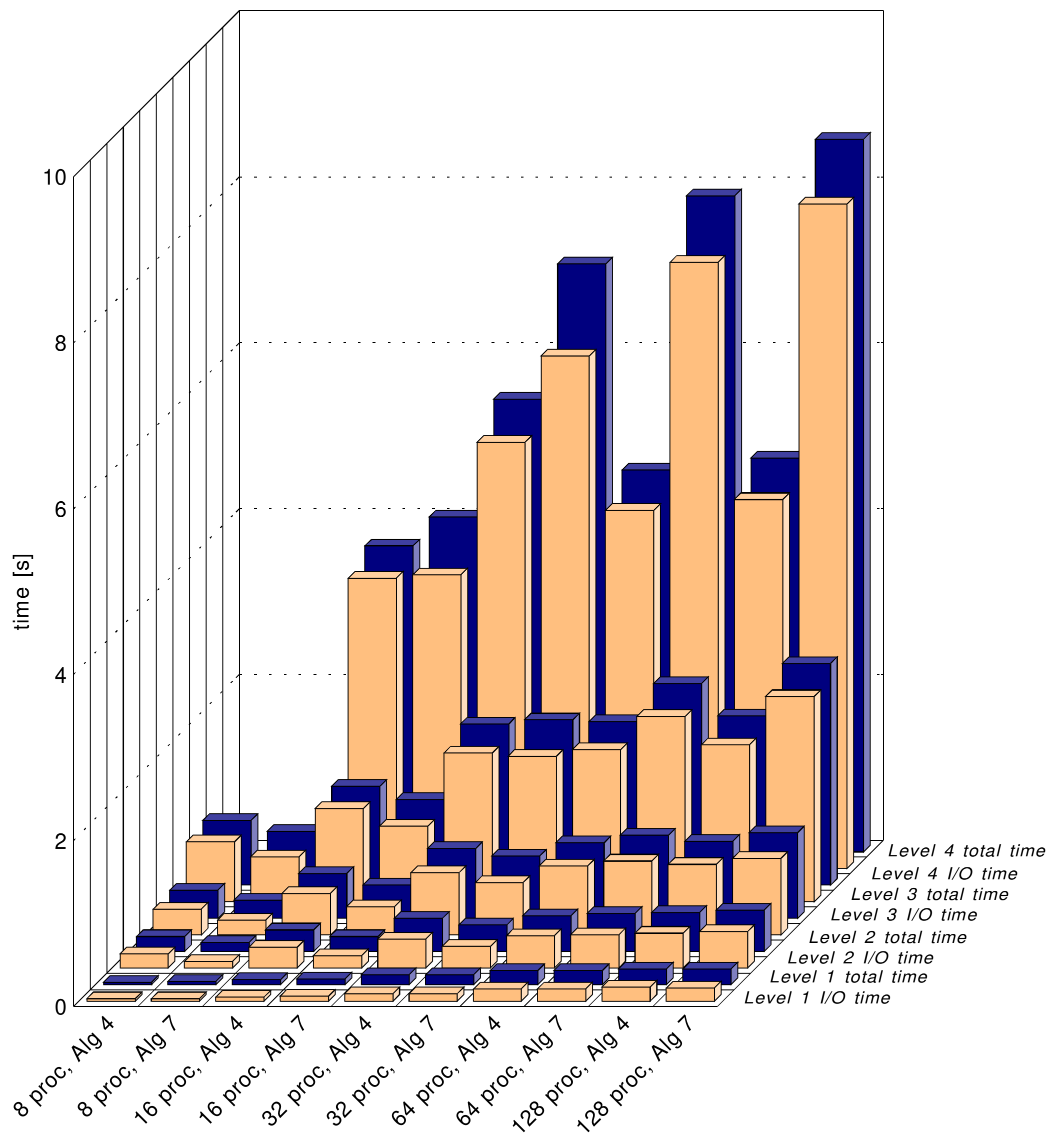

Figure 1: 48 tetrahedra, 2 tasks per node

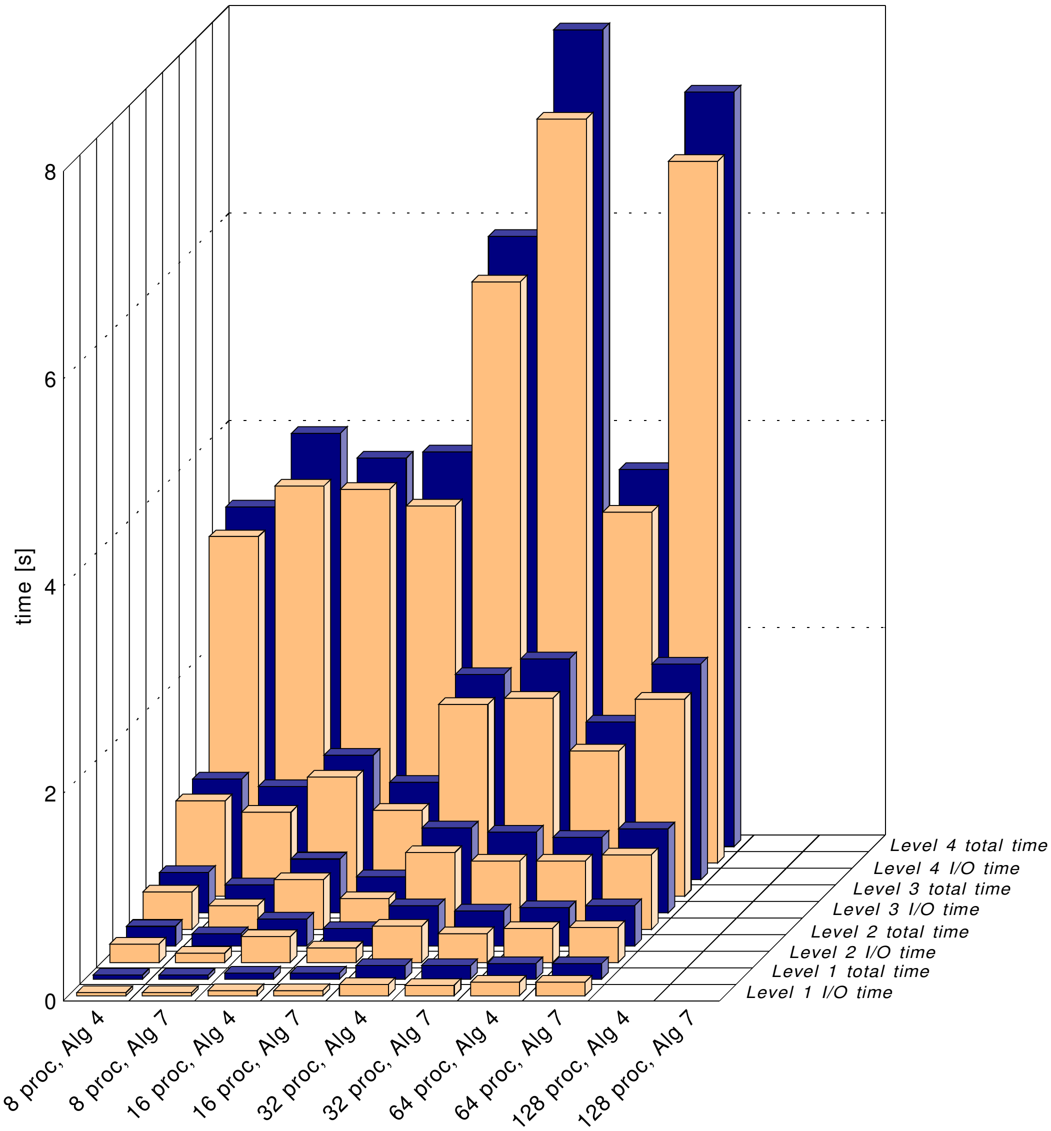

Figure 2: 48 tetrahedra, 1 task per node

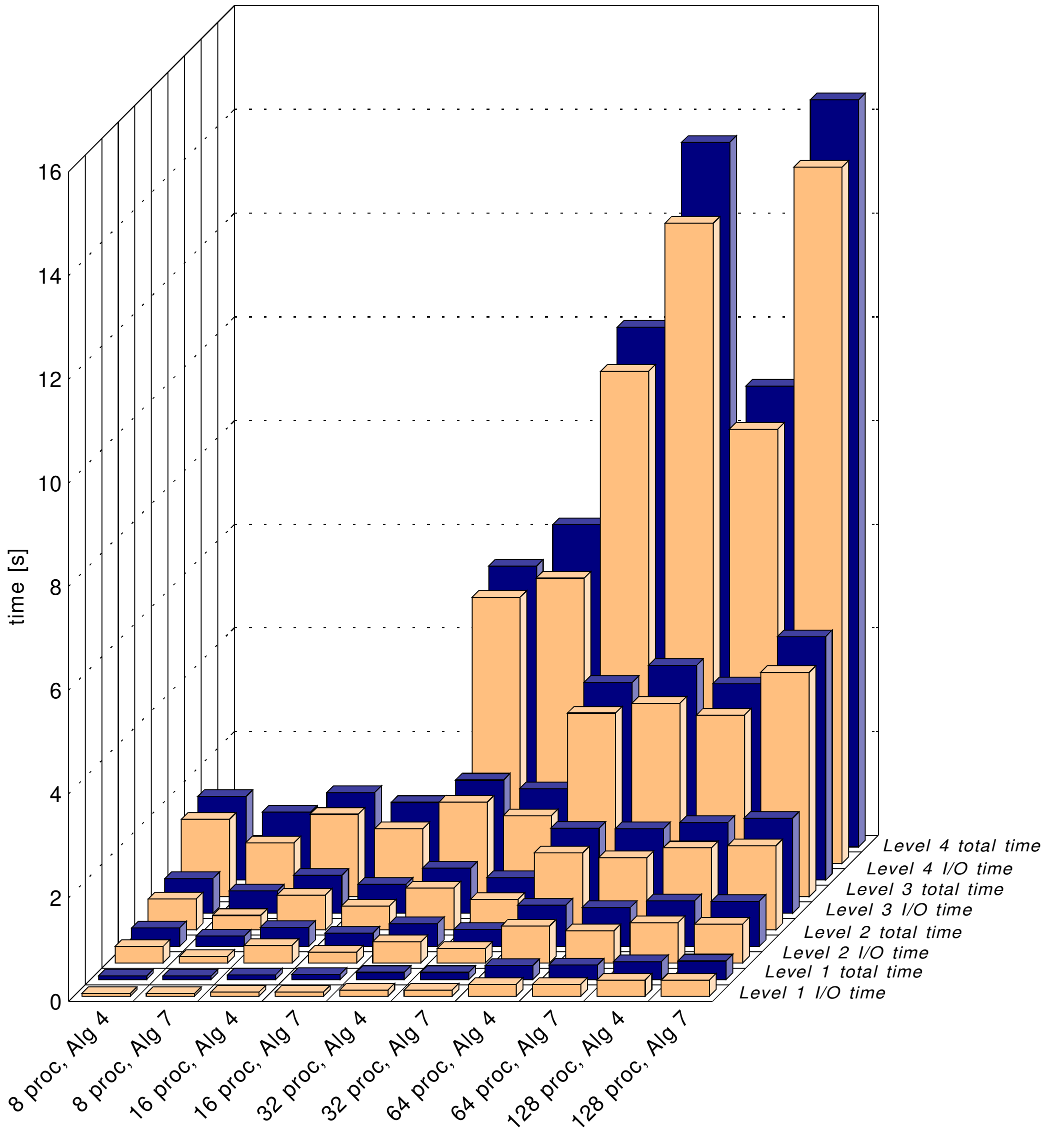

Figure 3: 96 tetrahedra, 2 tasks per node

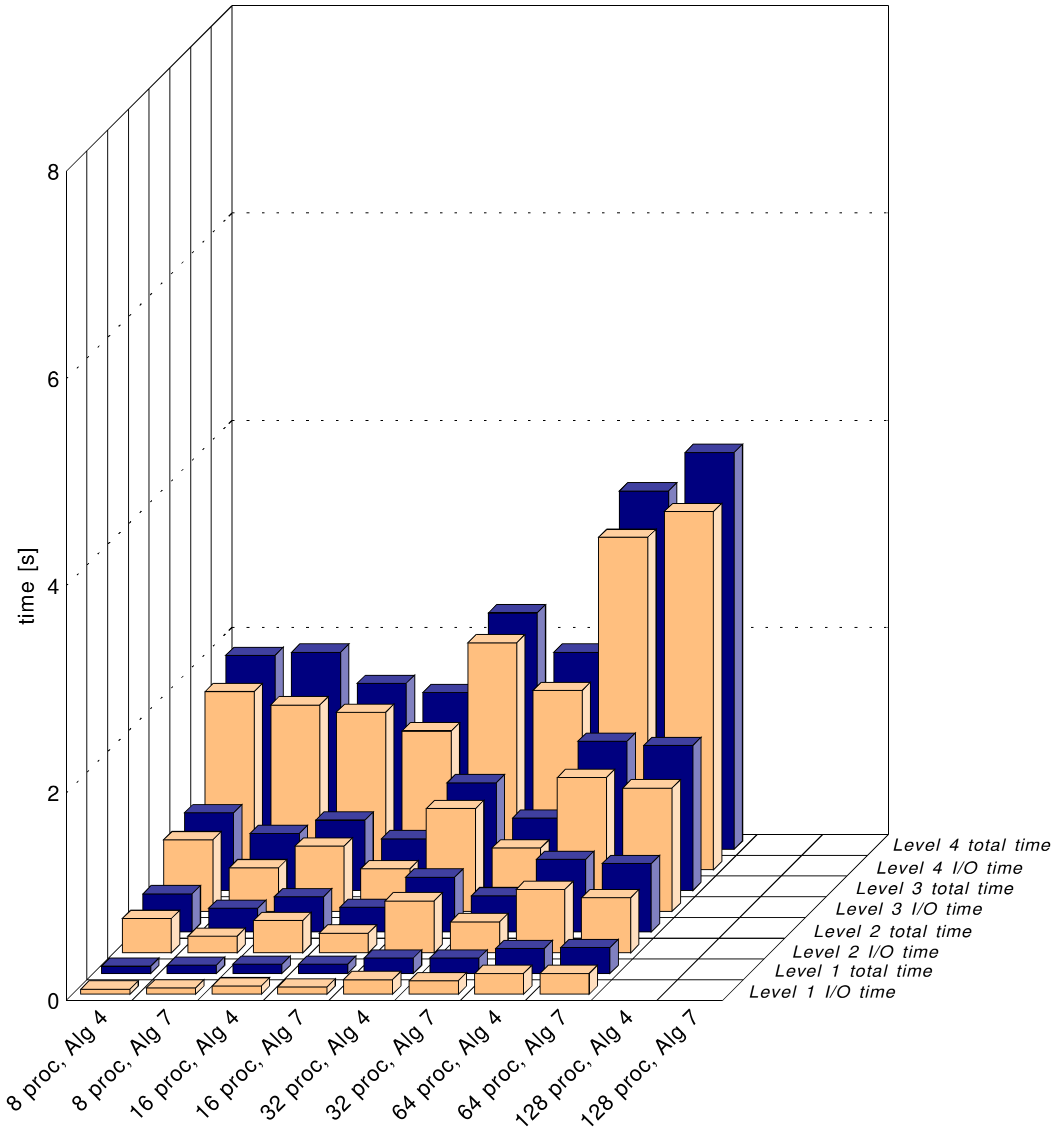

Figure 4: 96 tetrahedra, 1 task per node

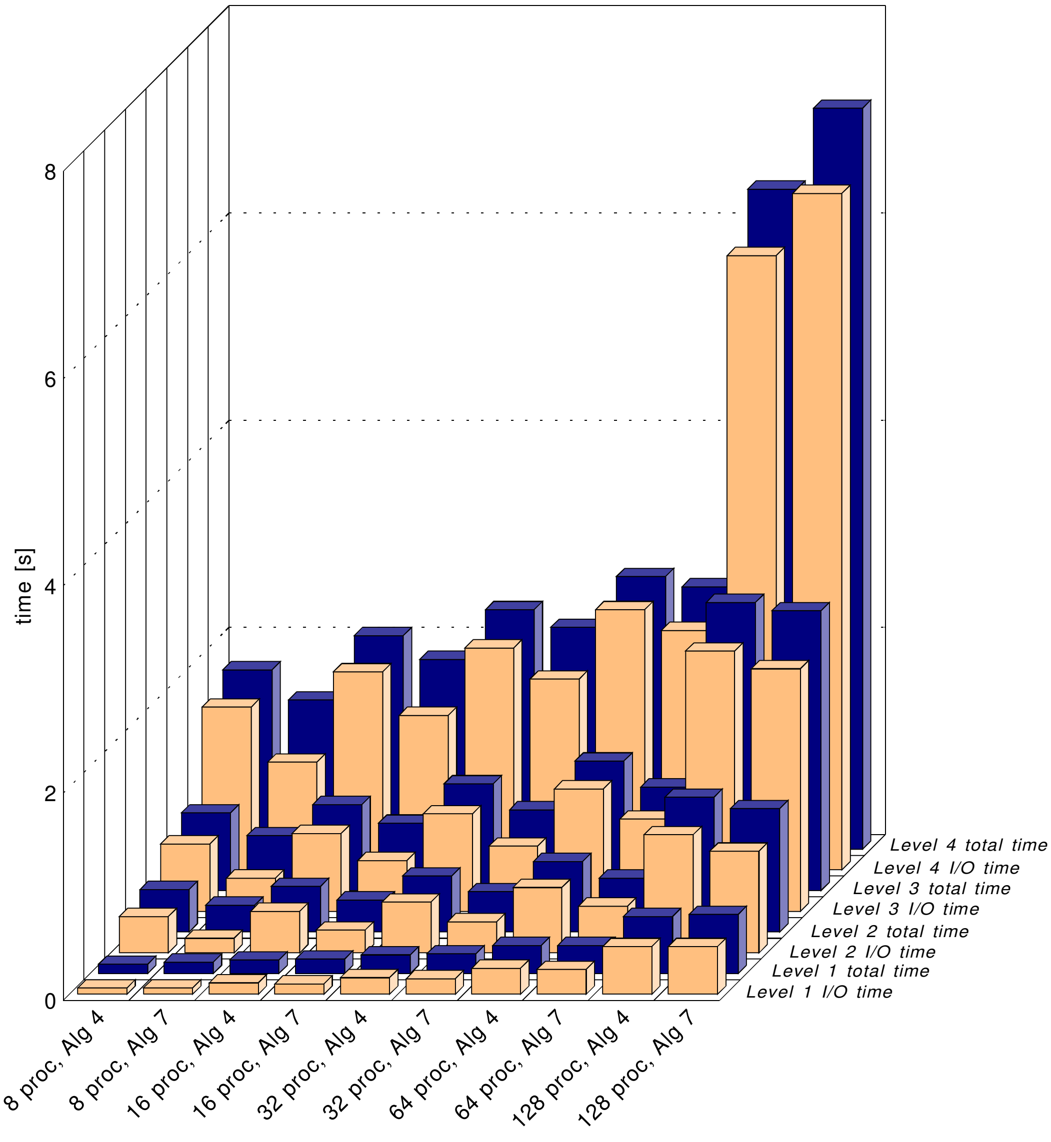

Figure 5: 192 tetrahedra, 2 tasks per node

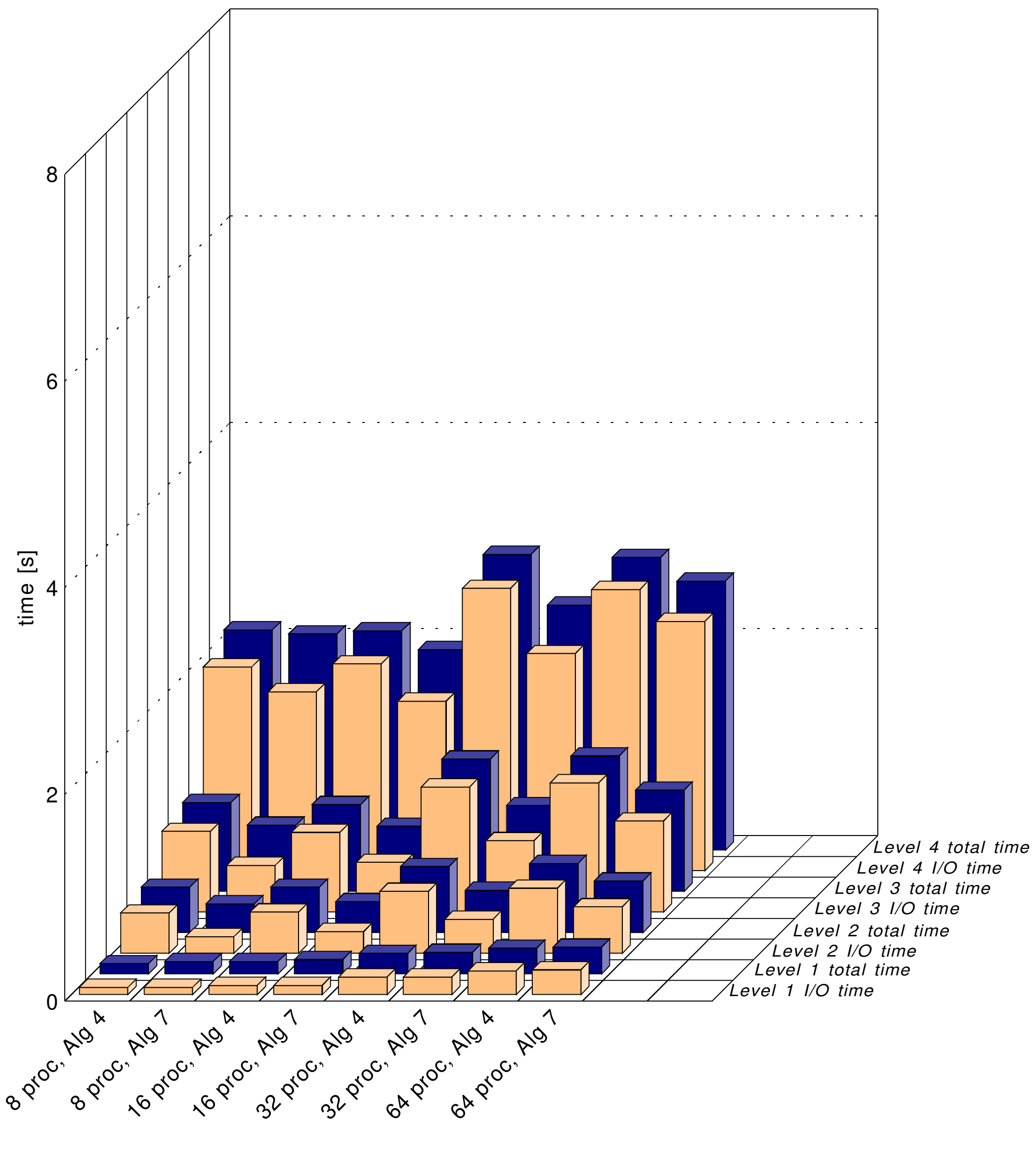

Figure 6: 192 tetrahedra, 1 task per node

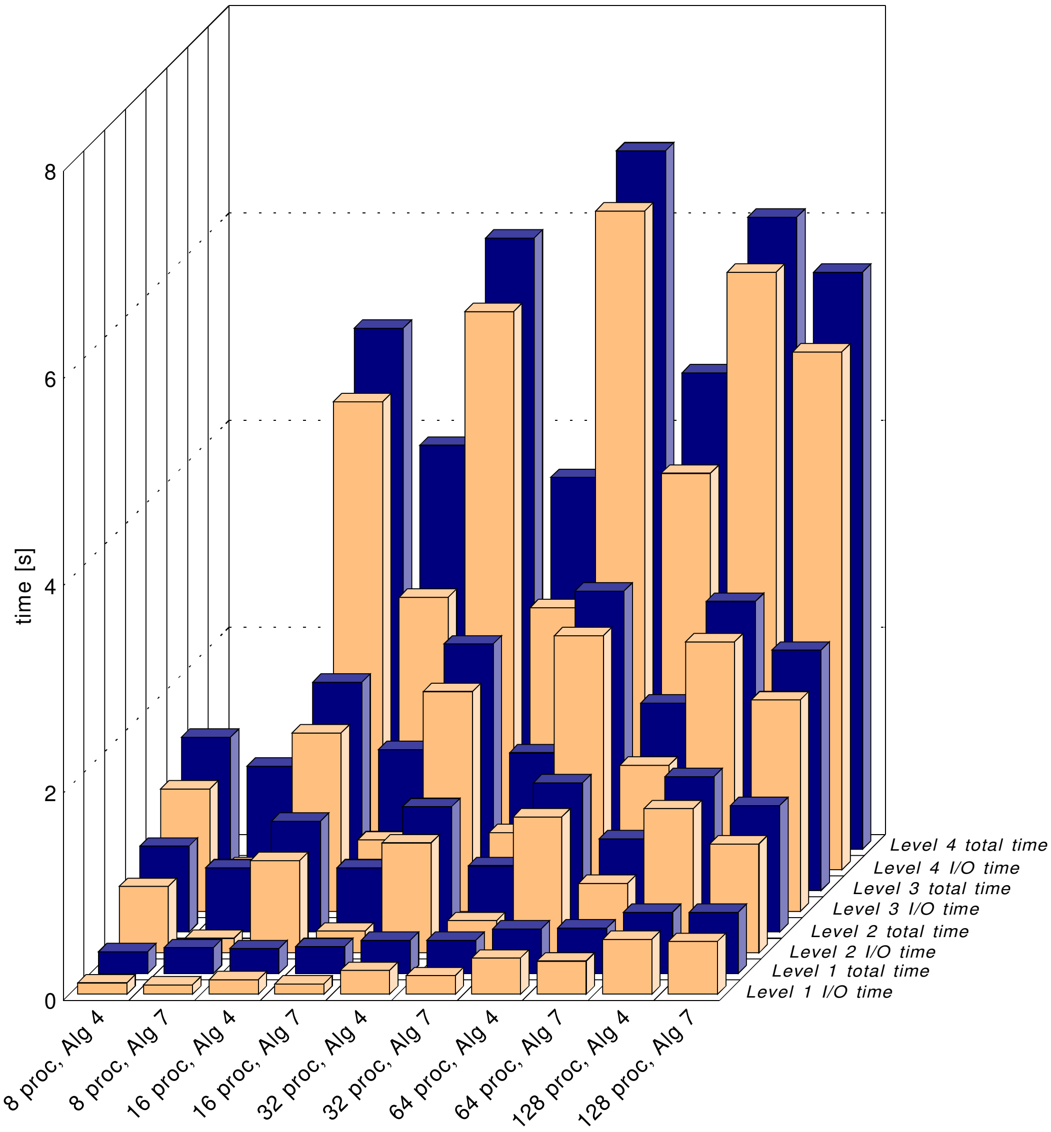

Figure 7: 384 tetrahedra, 2 tasks per node

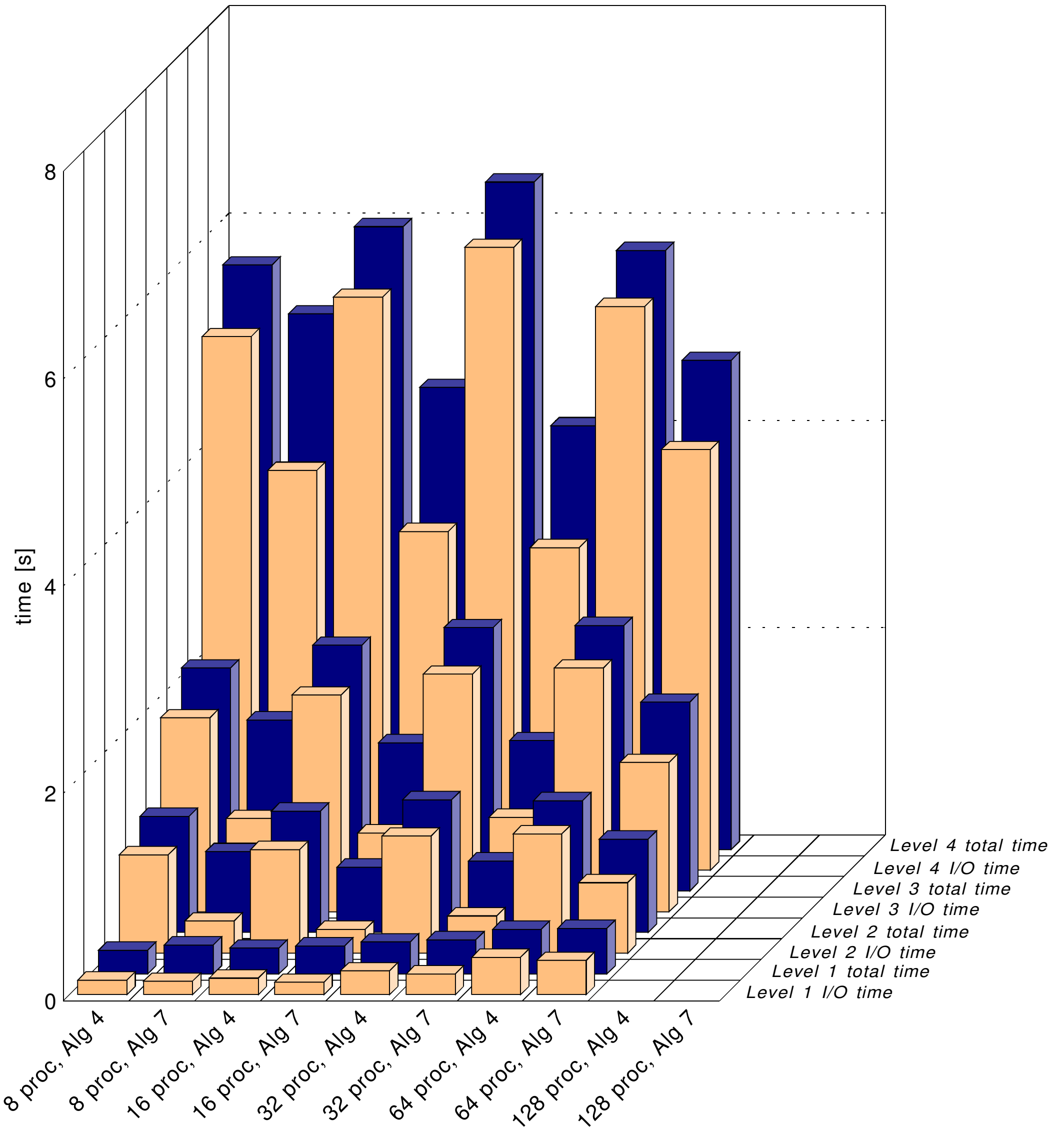

Figure 8: 384 tetrahedra, 1 task per node

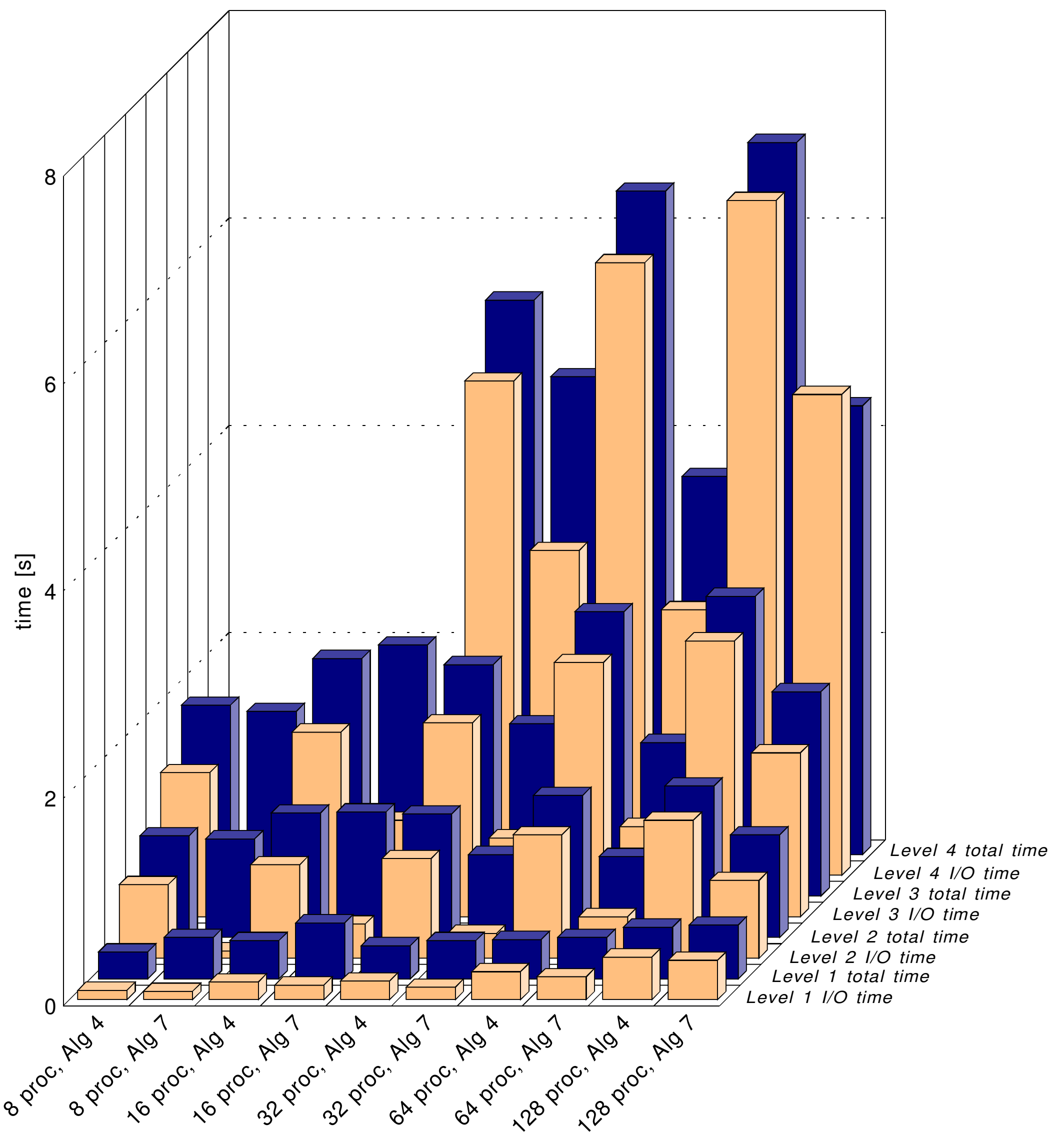

Figure 9: 768 tetrahedra, 2 tasks per node

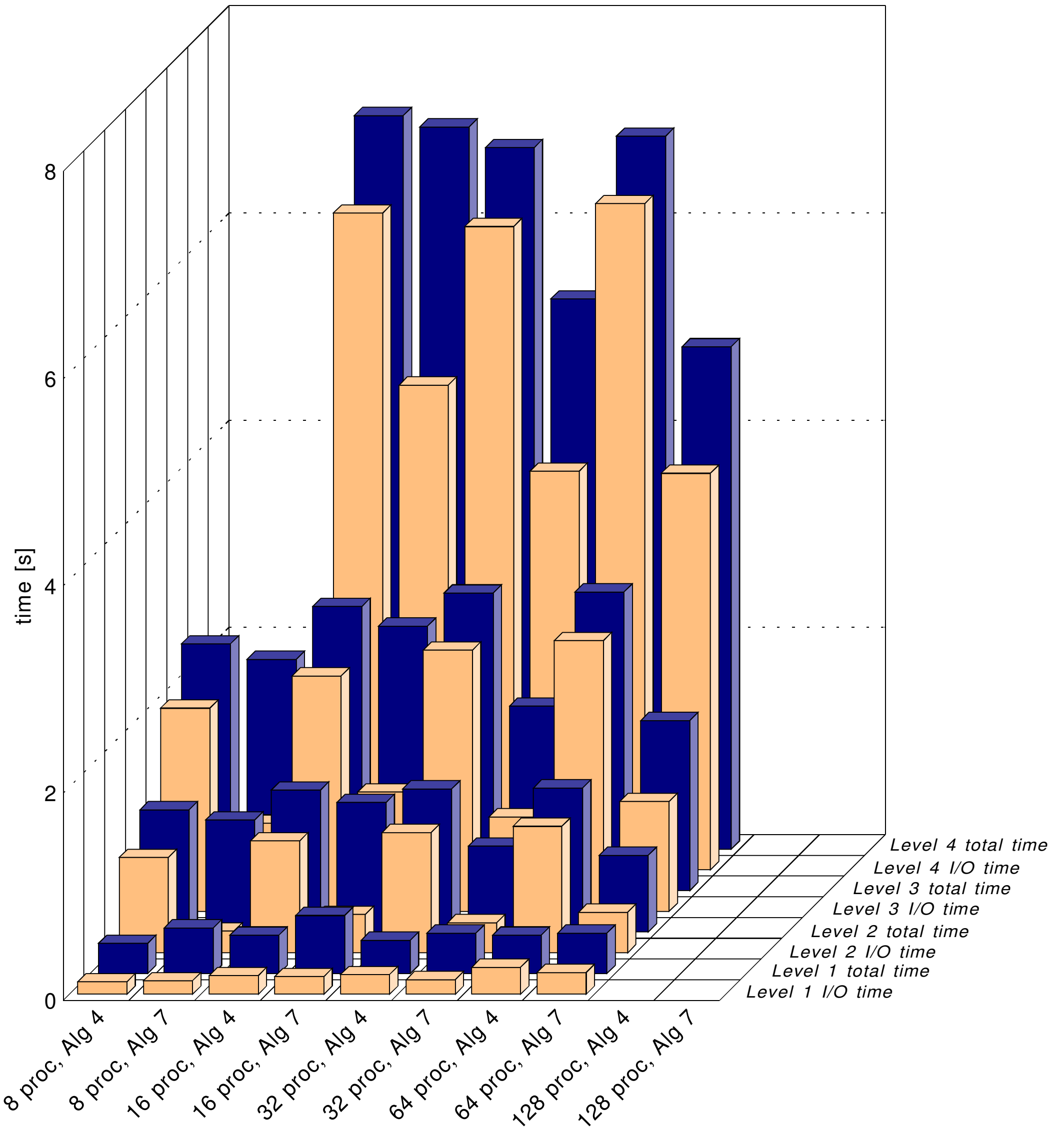

Figure 10: 768 tetrahedra, 1 task per node