|  |  |  |
| --- | --- | --- |
| **ACTIES OP HET SCHERM** | **VOICE-OVER** | **DUUR** |
|  | OpenMP allows you to leverage the full computational power of a computer, but what if that is not sufficient? What if you need more processing power, or if your data doesn’t fit in the memory of a single computer? In that case your application can be designed to run on multiple computers. The processes of this application communicate with one another to exchange state information or data. The de facto standard for distributed computing for scientific applications is MPI, the Message Passing Interface. Just like for OpenMP, you’ll just get a flavor here. PRACE has a number of courses specifically on parallel programming with MPI, as well as on hybrid programming by combining shared memory programming with OpenMP and distributed programming with MPI. |  |
| 1. Show compute\_pi.f90 | 1. Point out use statement 2. Mpi\_init, mpi\_finalize 3. Get rank, number of processes 4. Flow based on rank, print example 5. Type definition 6. Explain reduce |  |
| 1. Compile and run timed |  |  |
| 1. Show speedup, parallel efficiency | 1. Discuss parallel efficiency |  |
|  | Writing an efficient parallel application using MPI is far from trivial. However, it has the potential to scale your application to thousands of compute nodes. |  |
| **TOTALE DUUR** | | *Maak je screencast niet langer dan ca. 6 minuten.* |