XSAMG

Auto-parallel SAMG
for Clusters

Announcement

Auto-distributive parallel SAMG version for multi-node use

Document version 1.0
July 2011

Service & Technical Support
samg@scai.fraunhofer.de

Fraunhofer Institute SCAI
Schloss Birlinghoven
53754 Sankt Augustin
Germany

What is XSAMG?

XSAMG is a special version of **SAMG**[1] which automatically splits the system(s) of linear equations to be solved and distributes the sub-systems across multiple nodes of a compute cluster, this way drastically reducing solution times. The calling simulation program may itself be either serial or OpenMP parallel, but it is assumed *not* to be prepared for distributed computing across multiple nodes. In any case, however, XSAMG exploits the parallelism offered by both multiple cores and multiple nodes.

XSAMG makes your linear solver phase “cluster ready”

XSAMG makes it easy for code developers to exploit a multi-node cluster computer for the SAMG library interfaced to their code, without them having to acquire the necessary know-how, or indeed needing to be concerned, about the complicated underlying parallel cluster software infrastructure necessary for parallel execution on such a cluster. It lifts their application beyond the performance limits of a single node of such a cluster, which is not possible by the use of a simple paradigm like OpenMP. XSAMG makes the linear solver phase of single-node simulation programs “cluster ready”.

The abovementioned cluster software infrastructure is normally based on a parallel communication interface like “MPI”, as is the case inside XSAMG. In many situations, the time and effort to understand MPI and re-engineer the main application code as an MPI-parallel application is either too high a hurdle or cannot be justified in business terms. XSAMG, however, handles the MPI parallelism via completely internal mechanisms and thus opaque to the user. The user doesn’t need to program anything in MPI but merely has to slightly adapt the way of calling SAMG.

Who should consider using XSAMG?

Suppose that a simulation code calls, as a core section, a linear solver and that the code developer thus already uses (or plans to use) SAMG. Then, using XSAMG instead is a favorable option if the following conditions are met:

- The overall simulation code is either serial or OpenMP parallel[2].
- The code developer does not want to invest the effort to make his or her code MPI-parallel (or is not yet ready to do so), and yet wants to exploit cluster computers.
- The solution of linear systems requires a significant part of the overall simulation time, even with the very efficient SAMG.
- The linear systems to be solved, and resulting compute times, are large enough that the employment of more than one node makes sense.
- The code developer and/or the customers have access to a cluster computer.

Another motivation to use XSAMG may be given by the following scenario:

- Before investing the effort to make their own simulation code MPI parallel, code developers may want to realistically estimate the potential gain through doing this. Assuming that the MPI parallel efficiency of the linear solver is the main limitation for the overall parallel efficiency (which is often the case), then the use of XSAMG allows for a worst-case estimation of the overall parallel performance to be expected.

[1] *SAMG is a very efficient linear solver library based on algebraic multigrid (AMG), specifically developed for industrial applications. SAMG is OpenMP parallel and, hence, supports both serial and multi-core computations on single nodes. For more information on SAMG, please send an email to samg@scai.fraunhofer.de.*

[2] *For MPI parallel simulation codes, XSAMG is not needed. Instead, the MPI parallel analog of SAMG, SAMGp, could be employed directly.*

How XSAMG works: the execution model

XAMG is based on the following flexible execution model: One “head” node performs the user’s simulation tasks, the SAMG solver is spread across additional remote nodes – the more nodes deployed, the faster the overall execution speed (cf. Fig 1a and 1b). Note that XSAMG automatically exploits the parallelism offered by both multiple cores and multiple nodes, *even if the user’s (single-node) program does not exploit multi-core computing at all*.

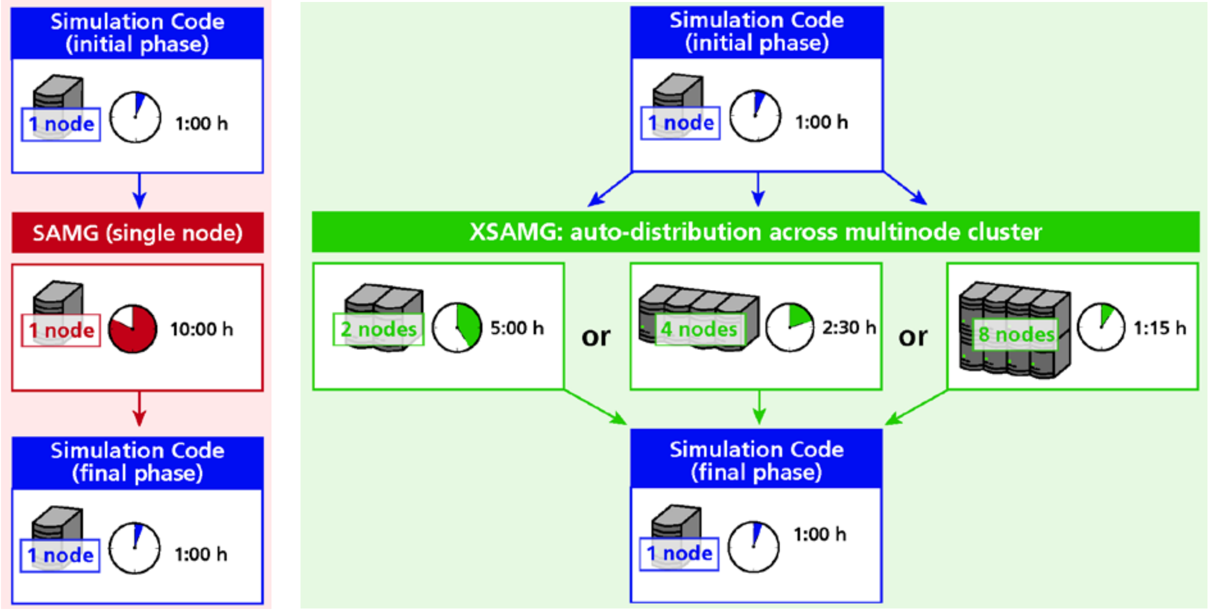
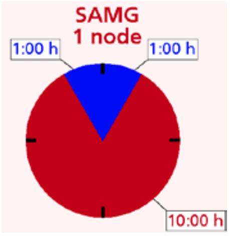

Simulation Code (initial phase)
1 node
1:00 h
SAMG (single node)
1 node
10:00 h
Simulation Code (final phase)
1 node
1:00 h
Simulation Code (initial phase)
1 node
1:00 h
XSAMG: auto-distribution across multinode cluster
2 nodes
5:00 h
or
4 nodes
2:30 h
or
8 nodes
1:15 h
Simulation Code (final phase)
1 node
1:00 h

***Fig 1a**: Single-node multi-core use of SAMG* ***Fig 1b**: Multi-node multi-core use of XSAMG*

Remark: The replacement of SAMG by XSAMG is straightforward: While merely two simple calls for initialization and finalization of XSAMG are required, the interfaces of SAMG and XSAMG are identical. The distribution of data and compute tasks across the nodes of a cluster is automatic.

What is the potential benefit?

Suppose a single-node application uses (serial or OpenMP parallel) SAMG as a linear solver. Just as an example, let us assume that the complete application needs 12h from which the linear solver takes up 10h (shown in red, cf. Fig 2a). Fig 2b shows how, ideally, the run time of the overall application could benefit from the cluster readiness of XSAMG: In principle, the pure solution time (shown in green) gets smaller as more nodes are employed. In-line with Amdahl’s law, the asymptotic minimum time required in this example is 2h.



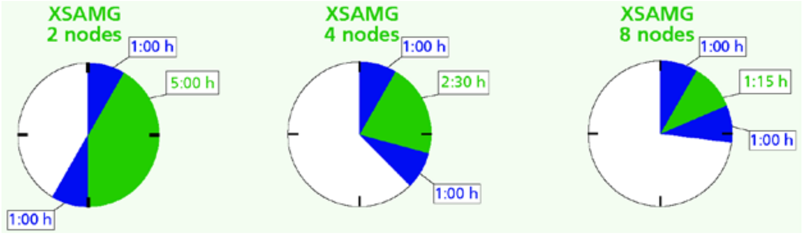


***Fig 2a**: Single-node multi-core use of SAMG: total run time (12h)* ***Fig 2b**: Multi-node multi-core use of XSAMG: Ideal total time for increasing number of nodes*

Limitations of XSAMG

The performance suggested by the above figures assumes that every doubling of the number of nodes speeds up XSAMG by a factor of 2. Clearly, this is an ideal assumption which will not be met in practice for the following reasons:

- Just as in any other MPI-parallel program, there is an MPI communication overhead involved which limits the MPI-parallel performance of XSAMG. As ever, this communication overhead is negligible relative to the computing cost only if the number of variables per node is large enough.
- Some additional overhead is required for distributing the solver data to the different cluster nodes. Although this distribution has been carefully optimized in XSAMG, the corresponding work cannot be neglected.

Hence, the performance figures in the example above are admittedly rather simplistic but certainly give a good intuitive feel for what can be gained in the ideal case.

Pre-requisites for using XSAMG

Essentially two things are pre-requisites for using XSAMG efficiently:

- The availability of a cluster with some brand of MPI installed on it.
- This cluster should be equipped with a reasonably fast network (e.g., Infiniband).

If both are true, SAMG may immediately be replaced by XSAMG. In addition, in the user’s simulation program, only a couple of very simple calls need to be introduced, whose purpose is only to initialize and terminate the MPI-parallel part of the code.

Remark: For mere testing, not even a real cluster is needed. In fact, the functionality can be verified also with a single-node usage of MPI (MPI can also use a shared memory, as OpenMP does).

Questionnaire

We would highly appreciate your help in planning the next steps of our development work. In particular, we would encourage you to fill in and return the **questionnaire** which comes with this announcement. Even if you think that the XSAMG software announced here is of no relevance for you, please let us know why this is so and, may be, whether there is something which could be done to make it more interesting for you.

Thank you very much for your help!

Your SAMG team

SAMG Development Group
Fraunhofer-Institute for Algorithms and Scientific Computing (FhG-SCAI)
Schloss Birlinghoven, 53754 Sankt Augustin, Germany
Fax: +49 (0) 2241 14-2102
E-Mail: samg@scai.fraunhofer.de
http://www.scai.fraunhofer.de/samg