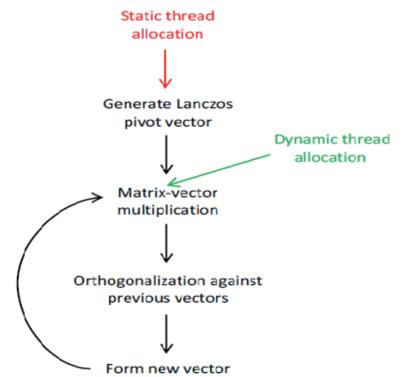


MFDn using Hybrid MPI/OpenMP

- Multi-threading using OpenMP in the most computationally intensive parts.
 - Take advantage of the multiple cores per node on modern supercomputers – mask memory access and communication with computation.
- Multi-threaded Lanczos iterative process is adapted at run-time.
- Two ways of allocating the number of threads.
 - Static – At the start of the run.
 - Dynamic – At regular intervals during the iterative process.

Iterative model for the Lanczos procedure (The Lanczos pivot vector is the initial working vector required for first MATVEC)



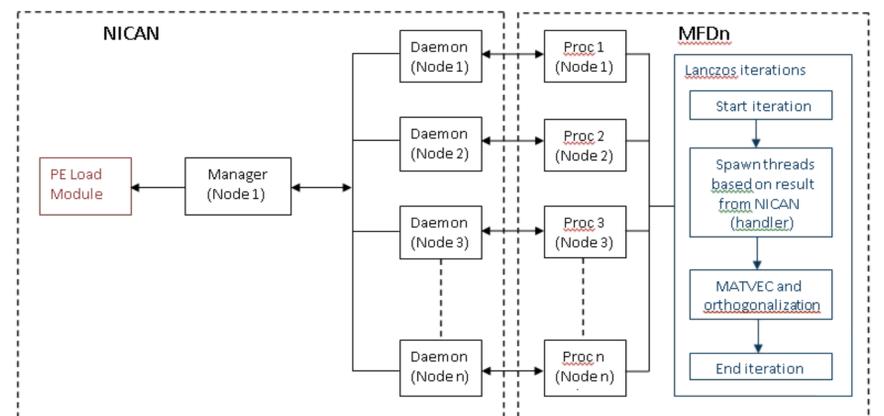
NICAN middleware

- NICAN is a middleware tool integrated with high performance parallel and distributed applications.
- Monitors the system resources during run-time and invokes necessary application adaptations to preserve parallel performance.
- Salient features of NICAN:
 - Decouples system related monitoring and decision making from application execution.
 - Can turn its actions ON or OFF with ease on demand by the application – ease of integration with application.
 - Does not hinder application performance – lightweight.
- Main components of the NICAN engine:
 - Manager – Controls functional modules and invokes application adaptations.
 - Module – Monitors system resources such as PE, I/O, Memory.
 - Daemon – Serves as an interface between the Manager and the distributed processes of the application.

NICAN integration with MFDn

- Problem:** In the presence of competing applications, cores get oversubscribed causing context switching to occur – degradation of performance for multi-threaded MFDn.
- Goal:** set the number of threads dynamically during the Lanczos iterations to prevent oversubscription.
- NICAN monitors the number of threads in the system during the run-time of MFDn to detect the presence of competing applications, if any.

Architecture of the MFDn-NICAN integration

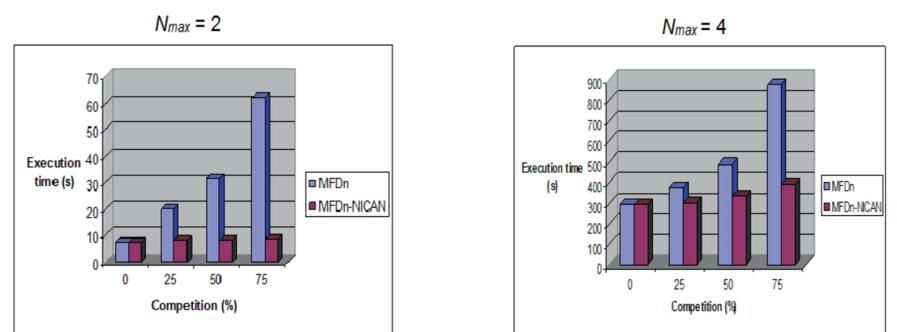


NICAN manager determines the number of threads to be spawned for an iteration of the Lanczos process and invokes appropriate adaptations in MFDn. NICAN daemons gather system information gathered from daemons and separate modules.

Parallel performance evaluation

- Testbed** – “Dynamo” cluster – 34 SMP nodes, 2 quad-core 3.0 GHz Intel Xeon processors and 16 GB RAM per node.
- Input data:** ¹²C nucleus for 2 problem sizes.
- Varying degrees of external competition.
- Significant performance gains are obtained with the MFDn-NICAN integrated model.
 - More than twofold for the larger problem size $N_{max} = 4$.
 - More than sevenfold for the smaller problem size $N_{max} = 2$.
- Difference in performance increases with increase in competition.

Comparison of the execution times of MFDn and MFDn-NICAN for ¹²C nucleus



Conclusions and future work

- Main contribution of this work – integration of middleware NICAN with MFDn.
- Dynamic thread adaptations invoked in MFDn using system information (regarding presence of competing applications) gathered at run-time by NICAN.
- Generic monitoring and decision making strategies employed by NICAN – can be used with any application having a computationally intensive iterative calculation.
- Possibility of incorporating global monitoring strategies left as future work.