Effects of Mapping Strategies on Average Duration and Throughput of Colocated HPC Applications

***Ioannis Vardas*, Sascha Hunold, Philippe Swartvagher, and Jesper Larsson Träff**

Faculty of Informatics, TU Wien, Austria

Due to resource contention like memory bandwidth, as well as limited scalability, applications are not always able to efficiently utilize the high core count of modern, deeply hierarchical HPC systems, like VSC-5. In such cases, colocating multiple applications to share compute nodes can increase the job system throughput. However, colocation can adversely impact individual application performance due to resource contention. Past research suggests that good mappings that take the system's topology into account can improve the performance of parallel applications that do not share resources [1].

We implement eight application-oblivious and topology-aware mapping strategies that produce process-to-core mappings and support colocation. We evaluate them using eight MPI applications on VSC-5. Our results show that colocation combined with specific mappings outperforms the exclusive allocation that is widely adopted by HPC clusters, both in average application duration and makespan.

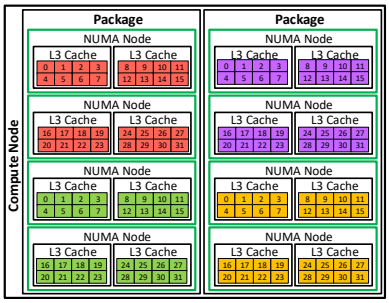

Fig. 1: Four applications sharing a node with `bbb` mapping.

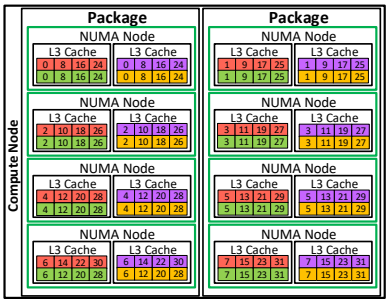

Fig. 2: Four applications sharing a node with `bcc` mapping.

Our mapping strategies consider three levels of resource hierarchy, the node, the package or socket and the NUMA node. The processes at each level are allocated either in a block or a cyclic way. Block allocates all cores within a resource before allocating from the next resource in the same level of hierarchy whereas cyclic performs the opposite. We denote the allocation (block or cyclic), for every level of the resource hierarchy with a single character, b or c. Figs. 1 and 2 show two mappings, `bbb` and `bcc`, of four colocated applications in a single VSC-5 node. We use different colors to depict the processes that belong to different applications. The `bbb` mapping (Fig. 1) increases the communication locality, whereas `bcc` (Fig. 2) increases resource utilization. Mappings also affect the number of shared resources among different applications.

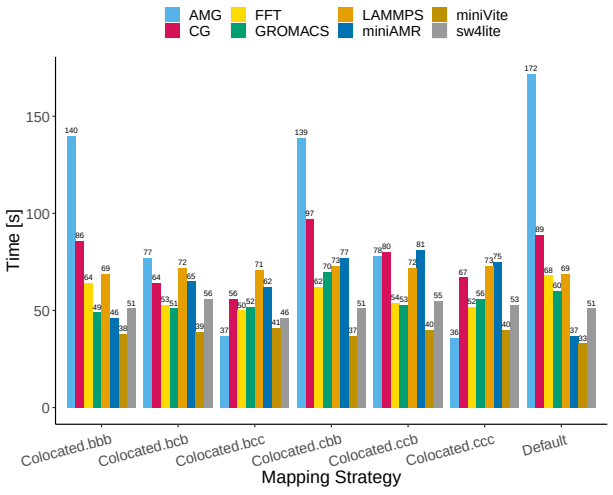

Fig. 3: Performance of eight colocated applications with six different mappings on eight VSC-5 nodes.

Fig. 3 shows the performance of eight colocated applications, each with 128 processes, that share eight nodes in VSC-5 and use six different mappings. We compare them against the default policy that allocates one node to each application where it runs in isolation. We notice that miniAMR and miniVite benefit more from mappings that increase communication locality, such as `bbb` or the default. Overall, `colocated.bcc` outperforms the default by 1.5× and 2.4× in terms of average duration and makespan respectively.

References

[1] von Kirchbach, K., Lehr, M., Hunold, S., Schulz, C., Träff, J.L.: Efficient Process-to-Node Mapping Algorithms for Stencil Computations. In: Proceedings of IEEE CLUSTER, 1 (2020)