On efficiency of fire simulation realization: parallelization with greater number of computational meshes

Lukas Valasek, Jan Glasa
Slovak Academy of Sciences/Institute of Informatics, Dubravska cesta 9, 845 07 Bratislava, Slovakia

E-mail: lukas.valasek@savba.sk

Abstract. Current fire simulation systems are capable to utilize advantages of high-performance computer (HPC) platforms available and to model fires efficiently in parallel. In this paper, efficiency of a corridor fire simulation on a HPC computer cluster is discussed. The parallel MPI version of Fire Dynamics Simulator is used for testing efficiency of selected strategies of allocation of computational resources of the cluster using a greater number of computational cores. Simulation results indicate that if the number of cores used is not equal to a multiple of the total number of cluster node cores there are allocation strategies which provide more efficient calculations.

1. Introduction
Simulation of fires belongs to computationally demanding tasks requiring efficient realization on a HPC (high-performance computer) system. Contemporary fire simulation systems are capable to incorporate many physical and chemical processes related to fire such as combustion, pyrolysis, thermal radiation, heat transfer, turbulence, gas spread dynamics, etc. Fire Dynamics Simulator (FDS) [1] is an advanced fire simulator utilizing the knowledge of CFD (computational fluid dynamics). It numerically solves a form of Navier-Stokes equations for low-speed fire induced flows with emphasis on transport of smoke and heat from the fire. FDS has been tested and validated for various types of fires and environments. It supports four programming models: sequential model designed for sequential computers, parallel MPI (Message Passing Interface) model designed for distributed memory systems, multithreading OpenMP (Open Multi-Processing) model designed for shared memory systems and hybrid MPI&OpenMP model designed for distributed shared memory systems. In our previous research we studied the impact of parallelization on efficiency and accuracy of a corridor fire simulation [2, 3] using a single computational node of the HPC cluster at the Institute of Informatics of Slovak Academy of Sciences in Bratislava (Slovakia). The research focused on the parallel MPI model of FDS and on testing the main strategies of allocation of available computational resources to given parallel calculations. In this paper we study efficiency of the parallel MPI model for simulation of the same type of fire using greater numbers of computational meshes and nodes.

2. Corridor fire scenario
Similarly to [2, 3] we use a rectangular 100 kW fire source in a corridor with dimensions of 7.2 m x 1.2 m x 2.7 m. The fire source with dimensions of 0.1 m x 1.0 m is placed on the floor at the distance of 3.64 m from the left corridor end and about 20 cm away from the corridor walls. We consider initial 10 s of fire (Figure 1).
In this paper we illustrate the efficiency of two strategies for allocation of computational resources: parallelization by the parallel MPI model using a single computational node of the HPC cluster. We illustrated efficiency of two strategies for allocation of computational resources: mapping, ranking and binding [4]. The exact choice on how MPI processes will be mapped by particular computational nodes and subsequently bound to particular sockets or cores is provided by the mpiexec script which is a part of Open MPI.

In [2, 3], we studied the sequential simulation described in the previous section as well as 9 parallel simulations parallelized by the parallel MPI model using a single computational node of the HPC cluster. We illustrated efficiency of two strategies for allocation of computational resources: --map-by core, --bind-to core and --map-by socket, --bind-to socket for parallel simulations with ms12 [3].

In this paper we illustrate the efficiency of two allocation strategies: --map-by node, --bind-to core (denoted by NC) and --map-by node, --bind-to socket (denoted by NS). The NC strategy binds particular MPI processes to particular computational cores of a cluster such that it binds particular MPI processes alternately node-by-node to corresponding particular cores of the socket 0 and consequently to corresponding particular cores of the socket 1. The NS strategy binds particular MPI processes to groups of cores of a socket such that it binds particular MPI processes alternately node-by-node to corresponding groups of cores alternately in socket 0 and socket 1. For testing purposes, the cluster nodes IBM dx360 M3 consisting of two 6-core processors Intel E5645, 2.4GHz with 48 GB RAM were available. The nodes were connected by the InfiniBand interconnection network with the bandwidth of 40 Gbit/s per link and direction. In order to avoid interference of the tested simulation with other tasks being solved on the cluster, we reserved the whole computational nodes for the tested simulation exclusively.

Figure 1. Smoke spread at the 2- and 6- s to the fire (left and middle), and the fire HRR (heat release rate) curve (right)

3. Sequential simulation

For sequential simulation a regular cube computational mesh with the 2 cm resolution (360 cells in the x-direction, 60 cells in the y-direction and 135 in the z-direction) was used. The total number of cells equals to 2916000. For simplicity, the INERT and OPEN boundary conditions were used on corridor walls and ends, respectively. The fire source is represented by a VENT with the 1000 kW/m2 HRRPUA (heat release rate per unit area) boundary condition. The default values of ambient and initial temperatures (20°C) and the default dominant chemical reaction (ETHYLENE) were considered. In regard of the mesh resolution, LES (Large Eddy Simulation) was used to resolve turbulence.

The mesh sensitivity study performed in [2, 3] indicates that the chosen mesh can be considered to be fine enough already since the 2- s to the fire. The total execution time of the sequential simulation (denoted by 1M) performed on a single computational core of the HPC cluster was about 17.45 hours. The fire course is illustrated in Figure 1. At first, the smoke spread upwards from the fire source. Then it hit the ceiling at the 3- s to the fire and began to spread under the ceiling in all directions. At the 6- s to the fire, the smoke reached both corridor ends. The HRR of the fire increased to the value of 100 kW at the 2- s to the fire and then fluctuated around this value till the end of the simulation.

4. Parallel simulations

In the parallel MPI model of FDS, the domain is divided into several computational meshes, the computation on each mesh is considered as a single MPI process assigned to one computational core. Communication between MPI processes is provided by MPI. We used the 64-bit MPI version 6.3.2 of FDS for Linux and the open-source Open MPI version 1.10.0 of MPI. The process of computational resources allocation for a given simulation represented by m-MPI processes consists of three phases: mapping, ranking and binding [4]. The exact choice on how MPI processes will be mapped by particular computational nodes and subsequently bound to particular sockets or cores is provided by the mpiexec script which is a part of Open MPI.
Similarly to [2, 3], we decomposed the computational domain in the x-direction into \( m = 18, 24, 30, 36 \) cube computational meshes (denoted by 18M, 24M, 30M, 36M) with the 2 cm mesh resolution (Table 1 and Figure 3). As the chosen mesh resolution and the simulated fire scenario were the same as in [2, 3], the mesh sensitivity study performed in [3] for both the sequential and parallel simulations indicates that the chosen mesh resolution can be considered as fine enough.

**Table 1.** Series of simulations: \( N_P \) is the number of MPI processes, \( N_M \) is the number of computational meshes, \( N_C \) is the total mesh cells, \( N_{cm} \) is the number of cells per mesh, \( N_C \) is the number of required computational cores and \( N_N \) is the number of required computational nodes.

| \( N_P \) | \( N_M \) | \( N_C \) | \( N_{cm} \) | \( N_C \) | \( N_N \) |
|---|---|---|---|---|---|
| 1M | - | 1 | 2916000 | 1 | 1 |
| 6M | 6 | 6 | 2916000 | 6 | 1 |
| 12M | 12 | 12 | 2916000 | 12 | 1 |
| 18M | 18 | 18 | 2916000 | 18 | 2 |
| 24M | 24 | 24 | 2916000 | 24 | 2 |
| 30M | 30 | 30 | 2916000 | 30 | 3 |
| 36M | 36 | 36 | 2916000 | 36 | 3 |

**Figure 2.** Total execution times of the sequential simulation (denoted by ●), parallel simulations carried out by CC and NC (denoted by ■) and parallel simulations carried out by SS and NS (denoted by ♦), where the values relating to those carried out by CC and NC are coloured grey (left), and their differences in % (right).

**Table 2.** Simulations efficiency: total execution times of the sequential and parallel simulations (in hours), where for the simulations carried out by CC and SS are coloured grey (columns 3-4) and for those carried out by NC and NS are coloured black (columns 5-9).

| | 1M | 6M | 12M | 18M | 24M | 30M | 36M |
|---|---|---|---|---|---|---|---|
| CC / NC | - | 17.45 | - | 6.393 | 3.744 | 2.551 | 2.073 | 1.740 | 1.576 |
| SS / NS | - | - | 4.778 | 3.748 | 2.249 | 2.075 | 1.591 | 1.581 |

**Table 3.** Simulations efficiency: differences between the total execution times from Table 2 in hours (row 2) and in % (row 3).

| | 6M | 12M | 18M | 24M | 30M | 36M |
|---|---|---|---|---|---|---|
| CC,SS / NC,NS | 1.615 | 0.004 | 0.302 | 0.002 | 0.149 | 0.005 |
| CC,SS / NC,NS | 25.262 | 0.107 | 11.838 | 0.096 | 8.563 | 0.317 |
In Table 2 and Figure 3, the total execution times (in hours) of the considered simulations 1M, 6M, 12M, 18M, 24M, 30M and 36M are shown. The simulations requiring a single computational node are coloured grey. In Table 3, the differences between the total execution times of the simulations listed in Table 2 are shown. One can make a similar observation about the allocation strategies NC and NS as it was described in [2, 3] about the CC and SS strategies. The total execution times of the simulations, where m equals to a multiple of the total number of cluster node cores (a multiple of 12), are very similar (see the execution times of the simulations 12M, 24M and 36M and their differences in Figure 3). In the case, where m does not equal to a multiple of the total cluster node cores number, the execution times of the simulations carried out by the NS strategy are smaller than of the ones carried out by the NS strategy (see the execution times of the simulations 18M and 30M and their differences).

5. Conclusion
In this paper, the efficiency of corridor fire simulation performed on the HPC cluster in Bratislava (Slovakia) is studied using the parallel MPI model of FDS and greater numbers of computational meshes. The strategies NC (\texttt{--map-by node, --bind-to core}) and NS (\texttt{--map-by node, --bind-to socket}) for allocation of the cluster computational resources were tested. The simulation results indicate that the NS strategy is more efficient than the NC strategy in cases where the number of computational meshes is not equal to a multiple of the total number of cluster node cores. As a fire simulation in large complex environments is computationally demanding task requiring greater number of computational meshes [5, 6, 7], further research of the efficiency of MPI parallelization using greater number of computational cores will be necessary.

Acknowledgment
This paper was partially supported by the Slovak Science Foundation (project VEGA 2/0165/17) and the Slovak Research and Development Agency (project APVV-15-0340).

6. References
[1] McGrattan K, Hostikka S, McDermott R, Floyd J, Weinschenk C and Overholt K 2015 Fire Dynamics Simulator (Version 6): User’s Guide, \textit{NIST Special Publication 1019-6}, NIST, Washington
[2] Valasek L and Glasa J 2016 A note on parallel efficiency of fire simulation on cluster \textit{J. Phys.: Con. Ser.} 738 012129
[3] Valasek L and Glasa J 2016 Note on fire simulation efficiency realized on computer cluster In Proceedings of the EUROSIM Congress on Modelling and Simulation, Danvers, IEEE Computer Society, 671-676
[4] Documentation of OpenMPI, version 1.10.0, August 25, 2015, https://www.openmpi.org/doc/v1.10/
[5] McGrattan K, Hostikka S, McDermott R, Floyd J, Weinschenk C and Overholt K 2017 Fire Dynamics Simulator Technical Reference Guide Volume 3: Validation, \textit{NIST Special Publication 1018-3}, NIST, Washington.
[6] Valasek L and Glasa J 2017 On realization of cinema hall fire simulation using Fire Dynamics Simulator Computing and Informatics 35 4 in press
[7] Weisenpacher P, Glasa J, Halada L, Valasek L and Sipkova V 2014 Parallel computer simulation of fire in road tunnel and people evacuation \textit{Computing and Informatics} 33 6 1237-1268