# Agents Based Hierarchical Parallelization of Complex Algorithms on the Example of *hp* Finite Element Method

M. Paszyński

Department of Computer Science
AGH University of Science and Technology,
Al. Mickiewicza 30, 30-059 Cracow, Poland
paszynsk@agh.edu.pl
http://home.agh.edu.pl/~paszynsk

**Abstract.** The paper presents how application of agents can improve scalability of domain decomposition (DD) based parallel codes, where the optimal load balance for some components of the code cannot be achieved only by partitioning computational domain. The limitation of the DD paradigm, where some highly overloaded pieces of domain cannot be partitioned into smaller sub-domains can be effectively overcome by parallelization of computational algorithm over these pieces. The agents are used to localize such highly loaded unbreakable parts of domain. Multiple agents are then assign to each highly loaded part to execute computational algorithm in parallel. The resulting hierarchical parallelization scheme results in the significant improvement of the scalability. The proposed agent based hierarchical parallelization scheme has been successfully tested on a very complex *hp* Finite Element Method (FEM) parallel code, applied for simulating Step-and-Flash-Imprint Lithography (SFIL), resistance heating of Al-Si billet in steel die for tixoforming process as well as for the Fichera model problem.

**Keywords:** Hierarchical parallelization, Computational agents, Finite Element Method, *hp* adaptivity.

## 1 Introduction

In general, there are two methodologies for parallelization of computational codes [4]. The first one is based on the *domain decomposition* paradigm (DD), where computational domain is partitioned into sub-domains and the same computational algorithm is executed over each sub-domain. The second one based on *functional decomposition* (FD), first decomposing the computation to be performed and then dealing with data.

For some classes of problems the first one is suitable, whilst for some other classes the second one is better. However, for some complex problems mixed approach is necessary. For example let us consider the system which consists of many components working on the same data structure. Most of these components can be efficiently parallelized by utilizing the domain decomposition paradigm. Unfortunately, some other components interpret these data in a different manner, and there are some unbreakable pieces of data which require very expensive computations.

The paper presents agent based hierarchical parallelization of such complex algorithms, where DD paradigm is utilized on the base level, and the functional decomposition is applied to resolve highly overloaded unbreakable pieces of domain. Many agents are applied to localize such highly overloaded pieces of domain, which cannot be broken into smaller subdomains. The agents utilize functional decomposition to reduce computational load on these nodes.

The Finite Element Method (FEM) [1] is the most popular and flexible way of solving engineering problems. The FEM applications are usually parallelized by utilizing the DD paradigm, since its computational mesh can be partitioned into uniformly loaded subdomains. The *hp* Finite Element Method [2] is the most complex version of FEM. The computational meshes for *hp* FEM consist of finite elements with various sizes (thus $h$ stands for finite element diameter) and with polynomial orders of approximation varying locally on finite elements edges, faces and interiors (thus $p$ stands for order of approximation). The parallel version of the *hp* FEM has been developed [5], utilizing the domain decomposition paradigm. Most of the components of the system, including decisions about optimal mesh refinements, parallel mesh refinements and mesh reconciliation, scale very well under domain decomposition paradigm. However, for some other components, like integration and elimination components of the solver, the computational meshes with high polynomial orders of approximation suffer from presence of single nodes, where computational cost can be higher than sum of costs for all other nodes in the mesh [5]. There is no way of partitioning such single nodes, and the domain decomposition paradigm is not sufficient to provide well scalability for these components. The proposed agent based hierarchical parallelization can effectively improve scalability of these components.

## 1.1 Hexahedral 3D *hp* Finite Element

The reference 3D hexahedral *hp* finite element presented in Figure 1 consists of 8 vertices, 12 edges, 6 faces and the interior. It is defined as a triple $\left(\hat{K}, X(\hat{K}), \Pi_{\hat{K}}\right)$ where $\hat{K}$ is a $[0,1]^3$ cube shape geometry of the reference element, $X(K)$ is a space of shape functions, defined as a subspace of $Q^{(p_x,p_y,p_z)}$ polynomials of order $p_x$, $p_y$ and $p_z$ in $\xi_1$, $\xi_2$, $\xi_3$ spatial variables. The polynomial order of approximation at each vertex is set to 1. In order to be able to glue together on single computational mesh finite elements with different orders, we associate a possibly different polynomial order of approximation $p_i = (p_{ih}, p_{iv})$ with each finite element face, where $h$ and $v$ stand for horizontal and vertical orders of approximation for each of six faces $i = 1, ..., 6$. Similarly, we associate with each finite element edges a possibly different polynomial order of approximation $p_j$ for each of twelve edges $j = 1, ..., 12$. Finally, the interior of an element has three polynomial orders of approximation $(p_x, p_y, p_z)$ in every spatial direction. The

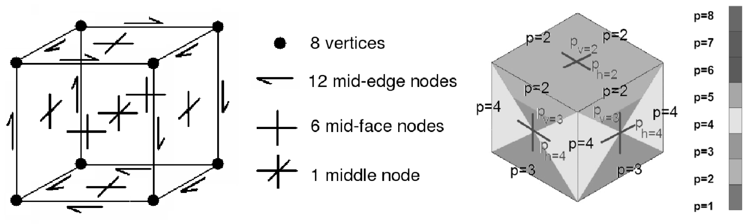

**Fig. 1. Left picture:** 3D hexahedral *hp* finite element. **Right picture:** Graphical notation for various polynomial orders of approximation on element edges and faces.

graphical notation for denoting different polynomial orders of approximation by different colors is introduced in Fig. 1. $\Pi_{\hat{K}}$ is the projection operator $\Pi_{\hat{K}} : X \rightarrow X(\hat{K})$, see [2]. The integration performed by FEM over any arbitrary geometry finite element $K$ called *physical element* are transferred into the *reference element* $\hat{K}$ by performing change of variables.

## 2 Computational Problems

The hierarchical parallelization will be discussed on three engineering problems

- The 3D model Fichera problem, described in details [5].
- The Step-and-Flash Imprint Lithography (SFIL), the patterning process utilizing photopolimerization to replicate microchip pattern from the template into the substrate [6]. The goal of *hp* FEM simulation is to compute volumetric shrinkage of the feature modeled by linear elasticity with thermal expansion coefficient.
- The resistance heating of the Al-Si billet in steel die for tixoforming process [7]. The goal of *hp* FEM simulation is to compute heat distribution during the resistance heating process, modeled by the Poisson equation with Fourier boundary condition of the third type.

To solve each of these problems in an accurate way, a sequence of *hp* FE meshes are generated by the fully automatic parallel *hp*-adaptive FEM code [5]. These sequences of meshes generated in automatic mode (without any user interaction) deliver exponential convergence of the numerical error with respect to the mesh size (number of degrees of freedom).

When the computational problem contains singularities related with either jumps in material data, jumps of prescribed boundary conditions, or complicated geometry, the generated *hp* meshes are irregular, and may contain very small finite elements with high polynomial orders of approximation, especially in areas close to these singularities.

In the first problem, there is one singularity at the center of the domain [5]. The generated *hp* mesh contains a single finite element with interior node with polynomial orders of approximation set to 7 in all three directions, as well as three finite elements with interior nodes with polynomial orders of approximation set to 6 in two directions and 7 in the third direction. The load representing computational cost over these elements for integration and elimination components is much higher then load over all other elements, see Fig. 2. In the second

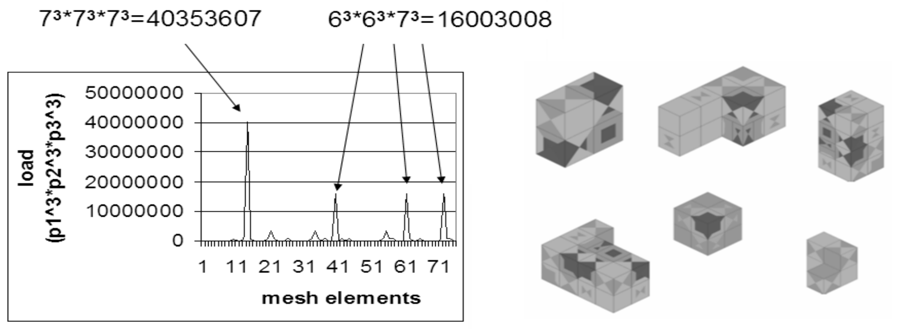

**Fig. 2.** The load inbalance over the optimal *hp* mesh for the Fichera model problem

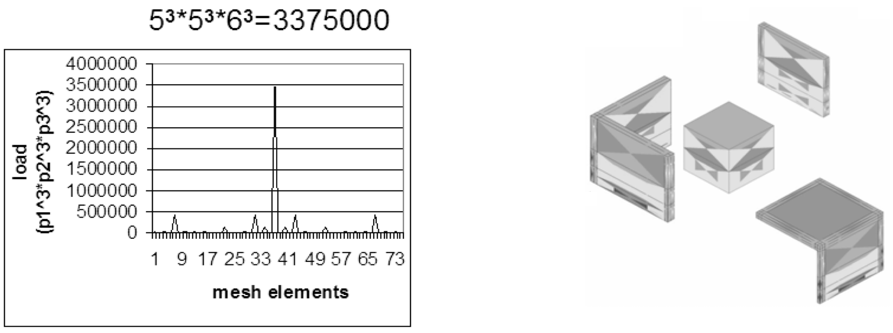

**Fig. 3.** The load inbalance over the optimal *hp* mesh for the SFIL problem

problem, there is one central finite element with polynomial order of approximation higher then orders in all other finite elements. The load over this element is higher then load over all other finite elements, and it is equal to 5 in two directions and 6 in the third direction, see Figure 3. In the third problem, there are many singularities related to jumps in material data. There are many finite elements with high polynomial orders of approximation and the load distribution is quite uniform, see Figure 4.

## 3 Load Balancing Problem

This section presents an overview of FEM computations performed by integration and direct solve components over *hp* finite element meshes. The goal of this presentation is to derive estimation of the computational cost over a single *hp* finite element. All problems presented in the previous section fit into abstract variational formulation

$$\begin{cases} u \in u_0 + V \\ b(u,v) = l(v) \, \forall v \in V \end{cases} . \tag{1}$$

We seek a solution $u$ from a functional space $u_0 + V$ where $u_0$ is the lift of the Dirichlet boundary conditions [3]. Here $b$ and $l$ are problem dependent functionals defined as integrals over the entire domain. The variational formulations are derived from partial differential equations describing the considered physical phenomena, and is satisfied for all test functions $v \in V$. The problem (1) is

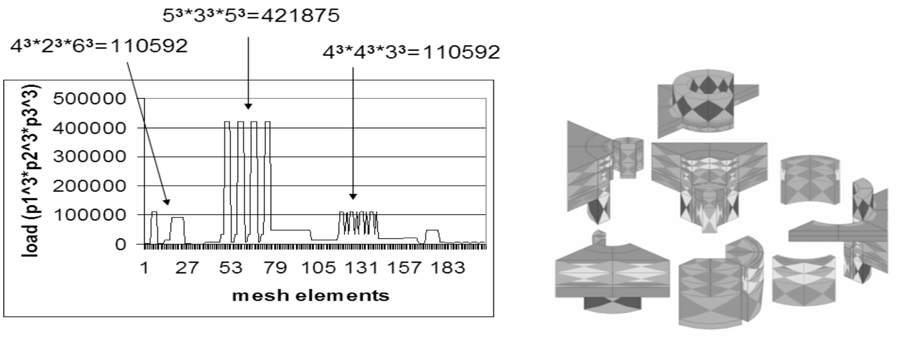

**Fig. 4.** The load balance over the optimal *hp* mesh for the resistance heating problem

solved over a finite dimensional subspace $V_{h,p} \subset V$ [3]

$$\begin{cases} u_{h,p} \in u_0 + V_{h,p} \\ b(u_{h,p}, v_{h,p}) = l(v_{h,p}) \, \forall v_{h,p} \in V_{h,p} \end{cases} . \tag{2}$$

The finite dimensional basis of $V_{hp}$ is constructed from polynomials of the first order in all directions at all finite elements vertices, from polynomials of order $\{p_{j,K}\}_{j=1,..,12,K}$ over all 12 edges of all finite elements $K$, from polynomials of order $\{(p_{ih,K}, p_{iv,K}\}_{i=1,...,6,K}$ over all 6 faces $j = 1, ..., 6$ of all finite elements $K$ and from polynomials of order $\{(p_{x,K}, p_{y,K}, p_{z,K})\}_K$ over interiors of all finite elements $K$. These polynomials are called *finite element shape-functions* and their support is spread over adjacent elements *only*, see [3] for more details. The *global shape functions* $\{e^i_{h,p}\}_{i=1,...,N}$ are obtain by collecting all interior finite elements shape functions, and by glueing together edge and face finite elements shape functions, respectively. The total number of global shape functions is denoted by $N$. Thus, the approximated solution $u_{h,p}$ is represented as a linear combination of global shape functions

$$u_{h,p} = u_0 + \sum_{i,...,N} u^i_{h,p} e^i_{h,p}. \tag{3}$$

The coefficients $\{u^i_{h,p}\}_{i=1,\dots,N}$ are called *global degrees of freedom.* and the discretized problem (2) can be rewritten as

$$\begin{cases} u_{h,p} \in u_0 + V_{h,p} \\ \sum_{i=1,\dots,N} u^i_{h,p} b\left(e^i_{h,p}, e^j_{h,p}\right) = l\left(e^j_{h,p}\right) \forall j = 1, \dots, N \end{cases} . \tag{4}$$

To solve problem (4) it is necessary to build and solve the global stiffness matrix $\{b\left(e^i_{h,p}, e^j_{h,p}\right)\}_{i,j=1,\dots,N}$, and the right-hand-side (rhs) vector $\{l\left(e^j_{h,p}\right)\}_{j=1,\dots,N}$. The global stiffness matrix and the rhs vector are aggregated from local vectors and matrices, related to restrictions of integrals $\{b\left(e^i_{h,p}|_K, e^j_{h,p}|_K\right)\}_{i,j=1,\dots,N}$, and $\{l\left(e^j_{h,p}|_K\right)\}_{j=1,\dots,N}$ over particular finite elements $K$.

For *hp* meshes with high polynomial orders of approximation the construction of these local contributions, performed by the integration component, is very expensive.

Let us consider a single finite element with polynomial orders of approximation in its interior equal to $(p_1, p_2, p_3)$. The size of the local matrix associated with such a finite element is equal to the number of degrees of freedom *nrdof* of the shape function of order $(p_1, p_2, p_3)$.

$$nrdof = (p_1 + 1)(p_2 + 1)(p_3 + 1). \tag{5}$$

Here is the algorithm building the local matrix over such a finite element:

```
for i=1,nint1
  for j=1,nint2
    for k=1,nint3
      for m=1,nrdof
        for n=1,nrdof
          aggregate local contribution to the stiffness matrix,
            associated with m-th and n-th degrees of freedom
        aggregate local contribution to the rhs vector
         associated with m-th degrees of freedom
```

where $nint1 = p_1 + 1$, $nint2 = p_2 + 1$ and $nint3 = p_3 + 1$ stand for number of Gaussian quadrature interpolation points necessary to compute integral of polynomial of orders $(p_1, p_2, p_3)$. The computational complexity of the integration algorithm is then $nint1 \times nint2 \times nint3 \times nrdof \times nrdof = (p_1+1) \times (p_2+1) \times (p_3+1) \times (p_1+1)(p_2+1)(p_3+1)(p_1+1)(p_2+1)(p_3+1) = (p_1+1)^3(p_2+1)^3(p_3+1)^3$. The load balancing is performed based on the computational cost estimation

$$load = (p_1 + 1)^3(p_2 + 1)^3(p_3 + 1)^3. \tag{6}$$

When the *hp* mesh contains finite elements with load much higher than sum of loads of all other elements, the optimal load balance is such that each expensive finite element is assigned to a separate processor, and some number of other processors is responsible for all other finite elements. For example in Figures 2 and 3 there are *hp* meshes balanced in optimal way into 6 or 4 processors, whilst total number of available processors is 8 (2 or 4 processors are idle).

## 4 Agents Based Hierarchical Parallelization

To overcome the problem with expensive integration over finite elements with very high $p$, the following agent based strategy is proposed.

The multiple agents are executed to localize finite elements with highest polynomial orders of approximation. Multiple agents are then assigned to each high order finite element. All agents assigned to a single high order element execute parallel version of the integration algorithm. The following line is added in the integration algorithm after $i, j, k$ loops:

```
4 if((k+(j-1)*nint3+(i-1)*nint2*nint3) modulo NRA != RANK)continue
```

where $NRA$ is the total number of agents assigned to high order node, and $RANK$ is the identificator of current agent. In other words, the Gaussian quadrature integration loops are cut into parts, and each agent executes some portion of loops. There are three phases of communication in this algorithm

- To localize finite elements with highest order agents must exchange orders of approximations of interiors of *hp* finite elements.
- Agents must exchange data necessary to perform integration over Gaussian quadrature points assigned to each agent. This involves *only* material data for a finite element and geometry of an element represented as *single double precision* Jacobian from change of variables from a physical finite element $K$ into the reference $\hat{K} = [0,1]^3$ element.
- At the end of loops, agents must sum up resulting local matrices of size $nrdof^2$ and vectors of size $nrdof$.

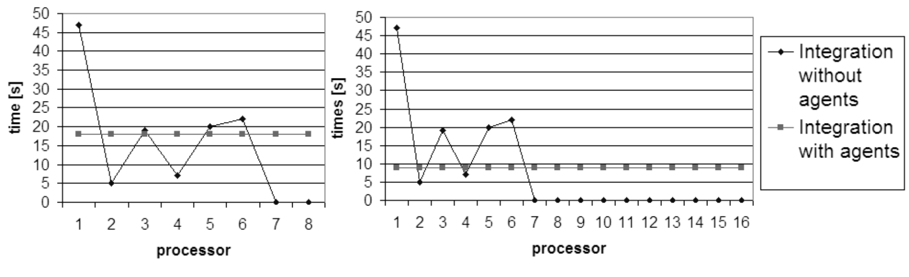

**Fig. 5.** Execution time of the integration algorithm with and without agents for 8 and 16 processors, for the Fichera model problem

The agent based hierarchical parallelization scheme was utilized to improve efficiency of the integration component for the presented problems. Exemplary results for the Fichera model problem are presented in Figure 5. There is 1 node with $(p_x, p_y, p_z) = (7,7,7)$ and 3 nodes with $(p_x, p_y, p_z) = (6,6,7)$, compare Figure 2. The domain decomposition, optimal with respect to load defined by (6), is put each of these nodes into separate sub-domain, and all other nodes into two other sub-domains. For 8 processors execution there are 2 processors idle, whilst for 16 processors execution there are 10 processors idle. The application of agents to localize highly loaded nodes and eliminate them by performing parallelization of loops gives ideal load balance, see Figure 5. The integration is followed by forward elimination phase of the parallel multi-frontal direct solver [5]. The cost of elimination over highly loaded element is $nrdof^3$ and the same load inbalances as for the integration component occurs. It means that we were able to improve efficiency of the integration and elimination components by 50%, since the elimination is still not well balanced. The further improvement of the efficiency can be achieved by switching to iterative solvers.

## 5 Conclusions and Future Work

The agent based hierarchical parallelization with DD paradigm on the base level and functional decomposition on highly loaded unbreakable pieces of data was proposed. The computational agents were utilized to localize such highly loaded unbreakable parts of domain, and to utilize functional decomposition by executing computational algorithm with parallelization of loops. The methodology has been successfully applied to improve efficiency of parallel *hp* FEM computations.

**Acknowledgments.** The work reported in this paper was supported by Polish MNiSW grant no. 3 TO 8B 055 29.

## References

1. Ciarlet P., The Finite Element Method for Elliptic Problems. North Holland, New York (1994)
2. Rachowicz, W., Pardo D., Demkowicz, L., Fully Automatic hp-Adaptivity in Three Dimensions. ICES Report 04-22 (2004) 1-52
3. Demkowicz L., Computing with hp-Adaptive Finite Elements, Vol. I. Chapman & Hall/Crc Applied Mathematics & Nonlinear Science, Taylor & Francis Group, Boca Raton London New York (2006)
4. Foster I., Desiging and Building Parallel Programs. www-unix.mcs.aml.gov/dbpp
5. Paszyński, M., Demkowicz, L., Parallel Fully Automatic hp-Adaptive 3D Finite Element Package. Engineering with Computers (2006) in press.
6. Paszyński, M., Romkes, A., Collister, E., Meiring, J., Demkowicz, L., Willson, C. G., On the Modeling of Step-and-Flash Imprint Lithography using Molecular Statics Models. ICES Report 05-38 (2005) 1-26
7. Paszyński, M., Macioł, P., Application of Fully Automatic 3D hp Adaptive Code to Orthotropic Heat Transfer in Structurally Graded Materials. Journal of Materials Processing Technology **177** 1-3 (2006) 68-71