# PARALLEL TRUE 3D CAE WITH HYBRID MESHING FLEXIBITY FOR INJECTION MOLDING

Wen-Hsien Yang*, Allen Peng, Louis Liu and David C.Hsu
*CoreTech System Co.,Ltd., HsinChu, Taiwan.*

## Abstract

True 3D mold filling simulation is becoming popular for its capability to providing better accuracy with minimum model simplification. However, such a large-scale non-linear computation places extreme demands on computing power. Moreover, the complex 3D geometry of the injection molded part further challenges the capabilities of the existing mesh generator and computation algorithms. In view of this fact, this paper develops an innovative parallel true 3D mold filling simulation technology, which allows for the adoption of hybrid volume element topologies. The parallel processing capabilities and the hybrid-element-supported solver capabilities of the proposed methodology have allowed the user to perform analyses in much less time on complex model with much larger element number than ever.

## Introduction

The application of true 3D mold filling simulation in the industry is becoming popular in the recent years. The related researches continue being published in the literature [1-4]. Moreover, there are already some commercial mold filling software which can do true 3D computation. The three major concerns of industrial CAE users are accuracy, computation speed and user-friendliness. True 3D simulation not only satisfies those concerns, but also offers more advantages that conventional 2.5D cannot reach, such as CAD integration, accuracy, minimized model simplification…etc.

However, the computation and memory size of true 3D simulation will demand much more than the conventional 2.5D method. Parallel computing technology is expected to be a promising solution to the above challenges. It not only reduces computation time dramatically, but also allows for simulations of much bigger models. Both the computation efficiency and accuracy are improved. We will only need to invest a little bit on the hardware upgrade and be paid back with significant performance enhancement. For optical parts, fiber-reinforced automobile components, connectors, gears...etc., the demand for high accuracy and high-speed computation can never be underestimated. Newer and more powerful CPU can improve computation speed. However, improvement from CPU clock rate alone cannot satisfy industrial users in speed nor accuracy. Utilization of multiple CPU is therefore the most effective approach.

Parallel computing is fast becoming an inexpensive alternative to standard supercomputer for solving large scale problems that arise in scientific and engineering applications. Generally, there are three types of parallel computing platform: (1) Symmetric Multiple Processor, SMP, (2) Distributed Memory Processor, DMP, and (3) Hybrid of SMP and DMP [5]. The CPUs of a SMP platform share the same memory and are controlled by a single operating system. Common SMP computers are supercomputer, dual CPU server, 4-CPU server…etc. A lot of industries have already used SMP computers as servers for their simplicity and easy maintenance. DMP platform consists of multiple computers. Each computer has its own CPU and operating system. The communication and collaboration between computers are done through high-speed network (Myrinet or Gigabit ethernet) and certain message interfaces. Common DMP platforms consist of a cluster of standard PC or high-end workstation. By utilizing standard PCs, PC cluster generally offers the best cost/performance. The cost in operation system can be even lower if Linux rather than Windows is employed. Supercomputer has a lot of advantages, but it's also too expensive to afford for most industies. PC cluster has the advantages of high performance and low-cost, so it's usually referred as "the Poor Man's Supercomputer". Among the 500 most powerful computer systems in the world, many are PC clusters.

This paper adopts the so-called coarse grain parallelization technology [5], that is, the computation domain is divided into several sub domains via Domain Decomposition Method (DDM) [6]. The computation within each subdomain is performed on individual processor, and the inter-communication between processors is necessary in order to exchange the required boundary information, as shown in Figure 1. In this manner, each processor has its own respective subdomain data but executes the same computation process, and hence this kind parallelization is known as SPMD (Single Process Multiple Data). SPMD has the advantages of easy-to-implement and good scalability. Theoretically, massively parallelization is possible via SPMD. As to the message communication during computation, MPI (Message Passing Interface) [7] is used in this paper. Since MPI has good hardware architecture compatibility, the present parallel processing can be applied on the abovementioned SMP, DMP or even hybrid parallel systems.

## Three-Dimensional Flow Analysis

**Governing Equations:**

The polymer melt is assumed to behave as Generalized Newtonian Fluid (GNF). Hence the non-isothermal 3D flow motion can be mathematically described by the followings:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot \rho \mathbf{u} = 0 \quad (1)$$

$$\frac{\partial}{\partial t}(\rho \mathbf{u}) + \nabla \cdot (\rho \mathbf{uu} - \boldsymbol{\sigma}) = \rho \mathbf{g} \quad (2)$$

$$\boldsymbol{\sigma} = -p\mathbf{I} + \eta\left(\nabla \mathbf{u} + \nabla \mathbf{u}^T\right) \quad (3)$$

$$\rho C_P\left(\frac{\partial T}{\partial t} + \mathbf{u} \cdot \nabla T\right) = \nabla(\mathbf{k}\nabla T) + \eta\dot{\gamma}^2 \quad (4)$$

where **u** is the velocity vector, *T* the temperature, t the time, *p* the pressure, $\sigma$ the total stress tensor, $\rho$ the density, $\eta$ the viscosity, k the thermal conductivity, $C_p$ the specific heat and $\dot{\gamma}$ the shear rate. In this work, the modified-Cross model with Arrhenius temperature dependence is employed to describe the viscosity of polymer melt:

$$\eta(T, \dot{\gamma}) = \frac{\eta_o(T)}{1 + \left(\eta_o \dot{\gamma} / \tau^*\right)^{1-n}} \quad (5)$$

with

$$\eta_o(T) = BExp\left(\frac{T_b}{T}\right) \quad (6)$$

where n is the power law index, $\eta_o$ the zero shear viscosity, $\tau^*$ is the parameter that describes the transition region between zero shear rate and the power law region of the viscosity curve. A volume fraction function f is introduced to track the evolution of the melt front. Here, f=0 is defined as the air phase, f=1 as the polymer melt phase, and then the melt front is located within cells with 0<f<1. The advancement of f over time is governed by the following transport equation:

$$\frac{\partial f}{\partial t} + \nabla \cdot (\mathbf{u}f) = 0 \quad (7)$$

The flow rate or injection pressure is prescribed at mold inlet. No slip is assumed at mold wall. Note that only inlet boundary condition is necessary for the hyperbolic transport equation of volume fraction function.

**Numerical Method**

The collocated cell-centered FVM (Finite Volume Method)-based 3D numerical approach developed in our previous work is applied in this paper [1]. The numerical method is basically a SIMPLE-like FVM with improved numerical stability. Furthermore, the volume-tracking method based on a fixed framework is incorporated in the flow solver to track the evolutions of melt front during molding.

**Mesh generation with hybrid elements**

Since the injected part geometry is usually quite complicated, it is not easy to mesh the whole model including cavity and mold base by the volumetric elements of the same topology. Hexahedral elements are preferred and commonly used in the conventional CFD analysis for their abilities to provide high resolution of viscous boundary layers. However, automatic generation of hexahedral elements to mesh the whole computation domain is almost impossible due to the complex geometry of injection-molded part. Therefore, user’s intervention is often required during the mesh generation process. Moreover, the thin and irregular part shape could cause the generated structured grid to be highly skewed. On the other hand, unstructured tetrahedral grids offer geometrical flexibility and allow fully automatic generation. However, tetrahedral grids are unfortunately not suited for the simulations of mold filling in thin part, since isotropic refinement in the thickness direction leads to an unacceptably large number of elements.

Therefore, in this paper, we develop a quite flexible numerical method that combination of different element topologies, including hexahedron, prism, tetrahedron and pyramid, are allowed to mesh the model as shown in the Figure 2. Several example models meshed by the hybrid element are shown in Figure 3.

## Domain Decomposition Method

This paper applies DDM to divide the whole model including part model, runner system, cooling channels and mold base, into a few smaller subdomains of equal size, then distributes each subdomain to each CPU for calculation. Therefore, maximum degree of parallelization can be guaranteed. Figure 4 illustrates the case of a block being decomposed into four subdomains of similar sizes. A good load-balancing algorithm is necessary for minimizing the difference of loading of different CPU. Otherwise, one or more of CPUs may be idled during computation.

During computation, each CPU will only calculate the subdomain assigned to it. At the same time, necessary data exchange across interfaces are required to transmit results of one subdomain to its surrounding subdomains. Data communication will inevitably further slow down computation. The bigger the area, the more data communications among CPUs are required. In each time step, data communications among CPUs are done with MPICH. Finally, the results from subdomains are collected into the whole results.

## Results and Discussion

In order to measure wall clock performance, two parameters can be defined:

$$S_n = \frac{T_s}{T_n} \tag{8}$$

$$E_n = \frac{T_s}{nT_n} \tag{9}$$

where $T_s$ and $nT_n$ refer to the sequential and parallel wall clock execution times respectively, and n is the number of partitions. $S_n$ is called speed-up ratio and $E_n$ is the speed-up performance.

First, we perform a filling simulation with simple part geometry to verify the numerical model. This case is analyzed on single, two and four processors, respectively. Figure 5 shows the respective mesh partition and compares the melt front advancements. We can see that both the sequential and parallel computations give the same prediction results.

Table 1 and Figure 6 demonstrate the parallel computation performance of the proposed methodology with various models. The testing platform is a SMP system with DualIntel Xeon 2.4GHz CPU and Windows XP Professional. Although speedup performance partially depends on the geometry, we can typically have 60% to 75% acceleration. Flow analysis is usually the most time-consuming step in all analyses. This improvement implies a big reduction in overall analysis time. Table 2 and Figure 7 show the parallel computing performance of warpage analysis. Generally, we can have 70% - 88% acceleration.

Using the latest AMD 64-bit Opteron CPU, the proposed approach can also show excellent parallel computing efficiency. Concluded from the table 3, the speedup of 4-Opteron-CPU (2.2 GHz) is about 3.2 times of single Pentium 4 CPU (3.0 GHz).

Another big advantage of parallel processing is the capability to deal with bigger models. Current 32-bit CPU can only address up to 4GB RAM. Excluding the memory reserved for Windows itself, an application program can only access up to 3GB RAM. Therefore, memory may not be enough for some big scale problems. Since the whole domain is divided with DDM, it's therefore possible for 32-bit CPU to calculate bigger model than ever. Table 4 demonstrates the application of Dual Xeon CPU to a big model with 3.2 million elements. Although the machine is equipped with 8GB RAM, memory is not enough when using single CPU due to the limitation of CPU itself. However, calculation can be successfully finished with dual CPU.

## Conclusions

An industry-strength parallel 3D mold filling simulation is developed in this paper. A highly flexible meshing approach comprising hybrid element shapes is also proposed to mesh the complex geometry of injection-molded part. Through the DDM and MPI, the current parallel processing technology not only enhances the computation efficiency, but also allows for simulation of much bigger models. This paper opens a brand-new age of CAE for injection molding in the history of CAE analysis technology.

## Reference

1. R.Y.Chang and W.H.Yang, Int. J. Numer. Methods Fluids, 37, 125-148 (2001).
2. W.H.Yang, A.Peng, L.Liu and D.C.Hsu, SPE ANTEC proceedings, 1003-1007 (2004).
3. W.H.Yang, A.Peng, L.Liu and D.C.Hsu, SPE ANTEC proceedings, 1434-1438 (2004).
4. S.J.Han, F.Costa, S.Ray and S.Willmorth, SPE ANTEC proceedings, 1684-1688 (2004).
5. A.Grama, V.Kumar, A.Gupta and G.Karpis, An Introduction to Parallel Computing: Design and Analysis of Algorithms, Pearson Addison Wesley (2003).
6. L.F.Pavarino and A.Toselli, Recent Developments in Domain Decomposition Methods, Springer Verlag (2002)
7. W.Gropp, E.Lusk and A.Skjellum, Using MPI, MIT Press (1999).

**Key Words**

**Three-Dimensional, CAE, Parallel computation, Mold Filling, Hybrid Mesh.**

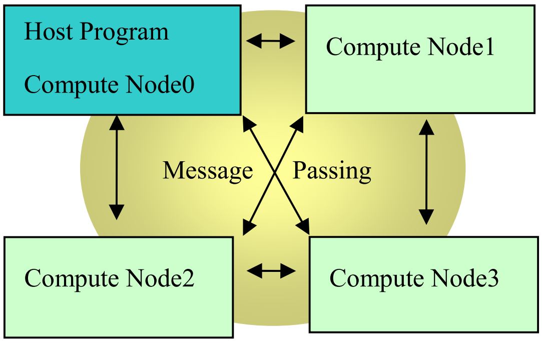


**Figure 1.** Parallel computation Architecture

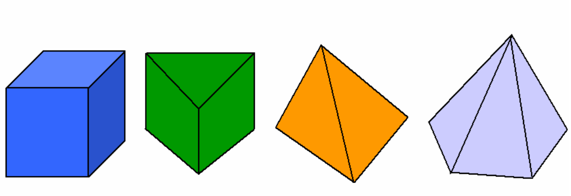
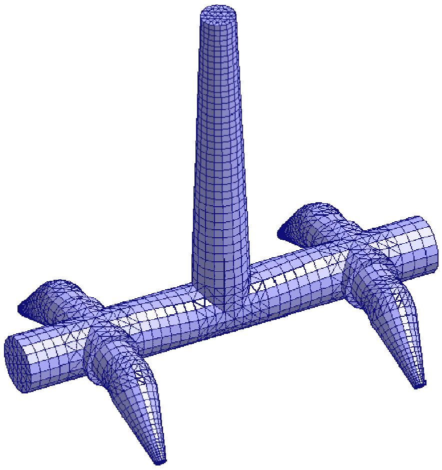


**Figure 2.** Element shapes supported in this paper



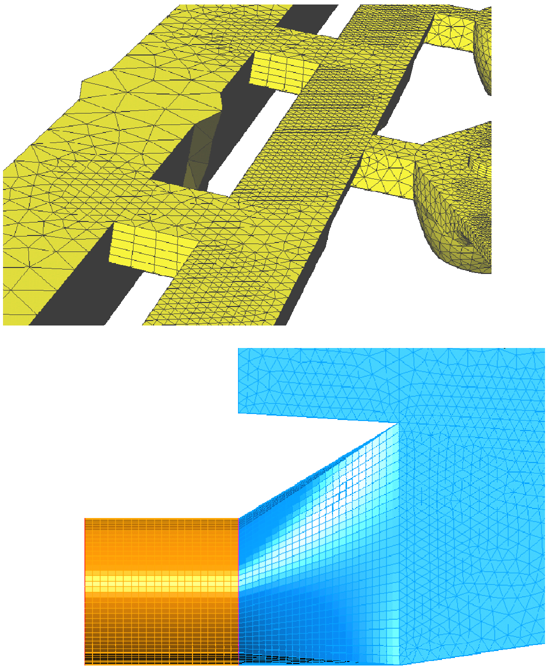


**Figure 3.** Example models meshed with hybrid elements

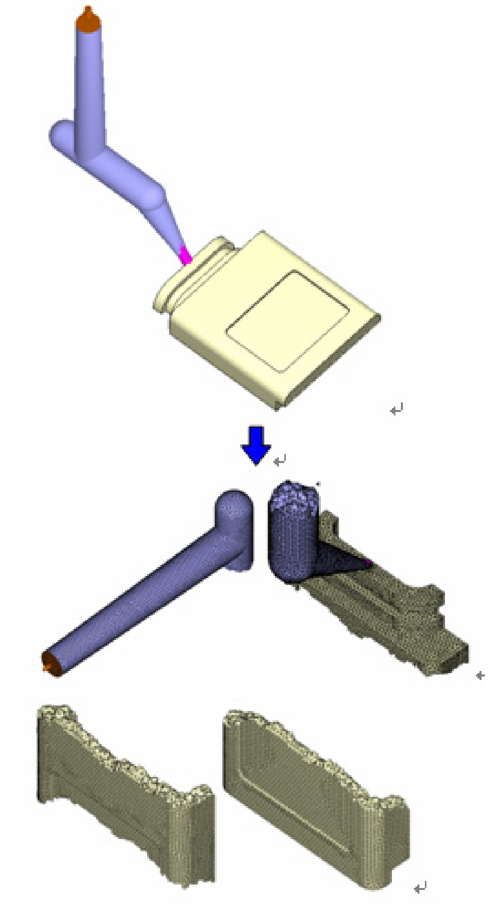
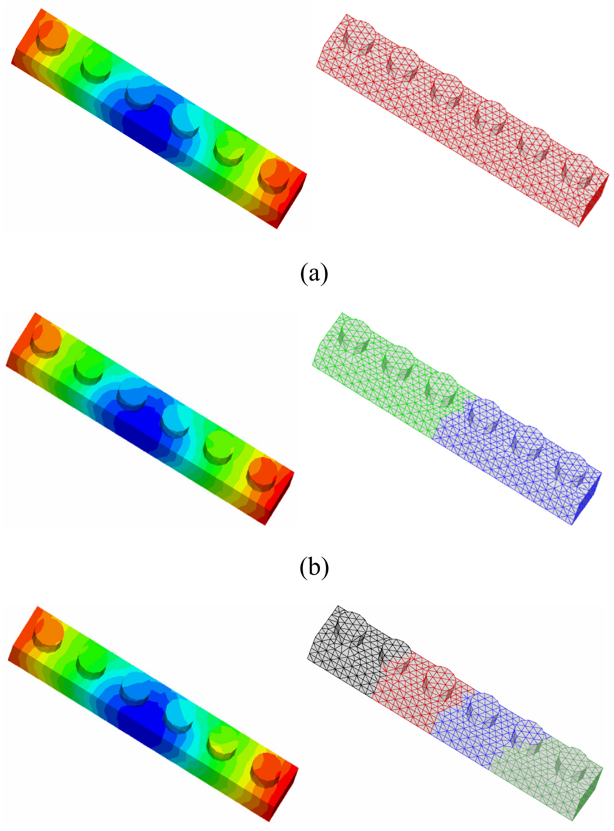


**Figure 4.** The partitioned FE mesh



(a)

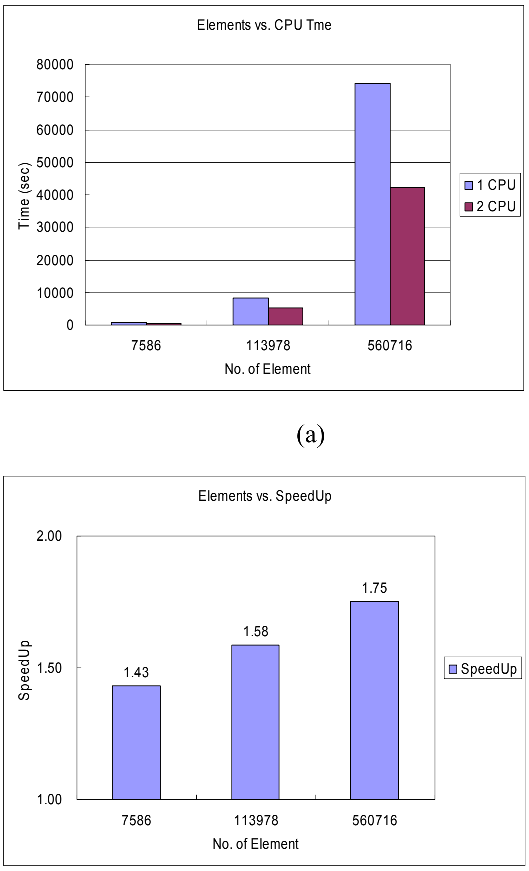
(b)

(c)

**Figure 5.** Comparison of melt advancements predicted by parallel computation (a) 1 CPU (b) 2 CPU (c) 3CPU

<table>
<tr><td colspan="5"><b>Platform: Dual Intel Xeon 2.4 GHz CPU with 2 GB RAM</b><br><b>System: Windows XP Professional</b><br><b>Mold Filling Simulation</b></td></tr>
<tr><td><b>Case #</b></td><td><b>Elements</b></td><td><b>1 CPU (sec)</b></td><td><b>2 CPU (sec)</b></td><td><b>Speed Up</b></td></tr>
<tr><td><b>Case 1</b></td><td>7,586</td><td>33</td><td>23</td><td>1.43</td></tr>
<tr><td><b>Case 2</b></td><td>113,978</td><td>8,400</td><td>5,300</td><td>1.58</td></tr>
<tr><td><b>Case 3</b></td><td>560,716</td><td>74,100</td><td>42,300</td><td>1.75</td></tr>
</table>

**Table 1.** Performance comparison of filling analysis on SMP with two CPUs

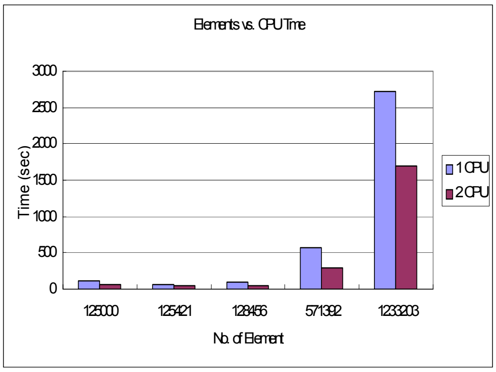


(b)

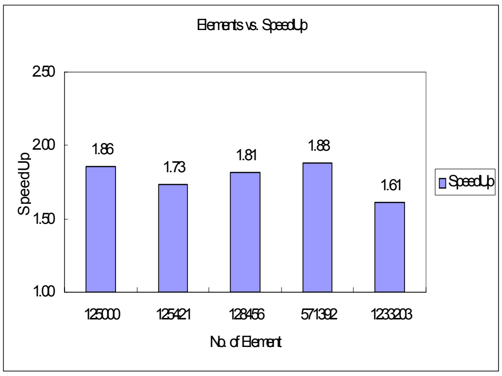
**Figure 6.** Performance plots of filling analysis on SMP with two CPUs (a) computation time (b) speedup ratio

<table>
<tr><td colspan="5"><b>Platform: Dual Intel Xeon 2.4 GHz CPU with 2 GB RAM</b><br><b>System: Windows XP Professional</b><br><b>Part Warpage Simulation</b></td></tr>
<tr><td><b>Case #</b></td><td><b>Elements</b></td><td><b>1 CPU (sec)</b></td><td><b>2 CPU (sec)</b></td><td><b>Speed Up</b></td></tr>
<tr><td><b>Case 1</b></td><td>125,000</td><td>119</td><td>64</td><td>1.86</td></tr>
<tr><td><b>Case 2</b></td><td>125,421</td><td>71</td><td>41</td><td>1.73</td></tr>
<tr><td><b>Case 3</b></td><td>128,456</td><td>98</td><td>54</td><td>1.81</td></tr>
<tr><td><b>Case 4</b></td><td>571,392</td><td>521</td><td>277</td><td>1.88</td></tr>
<tr><td><b>Case 5</b></td><td>1,233,203</td><td>2,721</td><td>1,688</td><td>1.61</td></tr>
</table>

**Table 2.** Performance comparison of warpage analysis on SMP with two CPUs



(a)



(b)

**Figure 7.** Comparison plots of warpage analysis on SMP with two CPUs (a) computation time (b) speedup ratio

| **CPU** | **Pentium 4** | **Opteron** | **Opteron** |
|---|---|---|---|
| **Clock (GHz)** | 3.0 | 2.2 | 2.2 |
| **CPU number** | 1 | 2 | 4 |
| **RAM (GB)** | 3 | 2 | 8 |
| **Operating System** | Windows XP | Windows XP | Windows 2003 Server (32bit) |
| **Flow Analysis** | 1.0 | 1.5 | 3.1 |
| **Pack Analysis** | 1.0 | 1.5 | 3.4 |
| **Warp Analysis** | 1.0 | 2.1 | 3.2 |

**Table 3.** Comparison of speed up on SMP with different CPU numbers

| **CPU** | Xeon | |
|---|---|---|
| **Clock (GHz)** | 2.8 | |
| **CPU number** | 1 | 2 |
| **RAM (GB)** | 8 | |
| **Operating System** | Windows 2003 Server Enterprise 32bit | |
| **Solid-Flow** | Memory Not Enough | 24.4 hour |

**Table 4.** Parallel computation of huge model size with 3.2 million elements