Parallel hybrid simulations of block copolymer nanocomposites using Coarray Fortran

Javier Diaz Marco Pinna Andrei V. Zvelindovsky Ignacio Pagonabarraga

Dr Javier Diaz
CECAM, Centre Européen de Calcul Atomique et Moléculaire, École Polytechnique Fédérale de Lausanne, Batochime - Avenue Forel 2, 1015 Lausanne, Switzerland
Prof. Ignacio Pagonabarraga
CECAM, Centre Européen de Calcul Atomique et Moléculaire, École Polytechnique Fédérale de Lausanne, Batochime - Avenue Forel 2, 1015 Lausanne, Switzerland
Departament de Física de la Matèria Condensada, Universitat de Barcelona, Martí i Franquès 1, 08028 Barcelona, Spain
Universitat de Barcelona Institute of Complex Systems (UBICS), Universitat de Barcelona, 08028 Barcelona, Spain
Dr Marco Pinna, Prof. Andrei V. Zvelindovsky
Centre for Computational Physics, University of Lincoln, Brayford Pool, Lincoln, LN6 7TS, UK
Email Address: mpinna@lincoln.ac.uk

Keywords: *nanocomposites, block copolymer, colloids, Coarray Fortran,parallel computing,nanoparticles*

Computer simulations of experimentally comparable system sizes in Soft Matter often require considerable elapsed times. The use of many cores can reduce the needed time, ideally proportionally to the number of processors. In this paper a parallel computational method using Coarray Fortran is implemented and tested for large systems of purely block copolymer melts, as well as block copolymer nanocomposites. A satisfactory strong scaling is shown up to 512 cores while a weak scaling with a drop in performance is achieved up to 4096 cores. The scaling of the parallel cell dynamic simulations scheme displays no drawbacks over MPI and provides an example of the simplicity of the Coarray approach. The code has been tested on several architectures and compilers. The hybrid block copolymer/nanoparticle algorithm can achieve previously unavailable system sizes.

1 Introduction

Block copolymers (BCP) are a particular type of polymer macromolecule in which chemically different monomer blocks are joint into a single chain. In the simplest case, diblock copolymer (DBCP) melts can self-assemble into well-ordered periodic structures due to the chemical incompatibility of the A and B monomers in a A-*b*-B macromolecule. Experiments[1, 2] and theory [3, 4, 5, 6, 7, 8] have described the rich phase diagram of DBCP melts, such as lamellae, hexagonally ordered cylinders and spheres among other phases. Additionally, DBCP thin films are of interest for lithography applications due to their ability to form nanoscale patterns near substrates[9, 10, 11, 12, 13].
The addition of colloidal nanoparticles (NPs) can greatly modify the inherent properties of block copolymer systems, [14, 15, 16, 17, 18, 19, 20] for instance, modifying the shape of block copolymer microparticles[21] in solvents. Block copolymer melts can exhibit changes in the electrical and optical properties[22] in the presence of colloids while nanoparticles can be found to assemble in hierarchical ordered configurations when mixed with BCP[23, 24, 25, 26]. Simulations of BCP/NP composite systems generally involve up to 5 BCP periods[27, 28]. Moreover, many physically relevant mechanisms require taking into account large length scales, such as the global ordering of BCP in the presence of NPs. BCP systems require large simulation box sizes in order to study their properties[29] as small simulation boxes can artificially pin systems in intermediate states[30].
Several microscopic approaches have been used to study BCP nanocomposites systems, namely Discontinuous Molecular Dynamics[31] and Monte Carlo simulations [32, 33, 34]. Self-consistent field theory has been largely used to obtain the equilibrium behaviour of nanocomposites[14, 15], studying, for example, the selectivity of NPs within BCP domains[35]. Additionally, Dissipative Particle Dynamics (DPD) provides a particle-based mesoscopic model to simulate relatively large systems, which has been used to study the aggregation of NPs within BCP domains[36, 37]. While DPD is a coarse-grained method that can use large time-steps to reach long time scales, its particle-based nature limits the ability to reach

more than 10 BCP periods. Cell Dynamic Simulations (CDS) is a highly efficient continuous method that has been used to predict the mesoscopic properties of BCP nanocomposites . It has been coupled with Brownian Dynamics for nanoparticles, resulting in a computationally inexpensive model to study the mesoscopic properties of nanocomposites[27, 38, 39]. The inherent speed of CDS can be further exploited by using parallel computing, allowing to reach larger system sizes while maintaining moderate computational times.
Coarray Fortran (CAF) is a parallelisation technique introduced in recent versions of the Fortran standard, which allows to parallelise computational codes with minimal modifications[40, 41]. CAF is an abstraction of Message Passing Interface (MPI) that simplifies the treatment of scalars or multidimensional arrays by creating several images which execute a program asynchronously[42] . Moreover, CAF uses distributed memory, meaning that each image or processor has its own workspace and communications between processors should be explicitly called. In fact, CAF belongs to the parallel model called partitioned global address space (PGAS). In CAF, a standard variable is purely local to the current image (processor), while a coarray can be accessed from remote images. In recent years there has been a growing number of programs using CAF[43, 44, 45, 46, 47] due to the simplicity of its approach and the popularity of Fortran in the field of High Performance Computing(HPC)[48]. The OpenCoarrays project, for instance, has introduced wrapper compiler and executable launchers among other utilities[42]. CAF performance has been previously compared with MPI, finding no drawback in the use of CAF over MPI [49]. A recent review provides historical context on the development of CAF as well as key features[50].
This work is organised as follows: the model for CDS is presented, followed by its parallel implementation. Secondly, the Brownian Dynamics scheme for colloids is presented, along with the parallel implementation for colloids as well as the coupling between BCP and NPs. The results for CDS are shown, in order to compare with previous MPI implementations, and also as a way of validating the physical results. Finally, the novel results for BCP/NP systems are shown along with simulation snapshots.

2 Model

The evolution of the BCP/colloids system is determined by the excess free energy which can be separated as[51]

$$\mathcal{F}_{tot} = \mathcal{F}_{pol} + \mathcal{F}_{cc} + \mathcal{F}_{cpl} \tag{1}$$

with $\mathcal{F}_{pol}$ being the free energy functional of the BCP melt, $\mathcal{F}_{cc}$ the colloid-colloid interaction and the last contribution being the coupling term between the BCP and the colloids.

2.1 Polymer dynamics: Cell Dynamics Simulations

The diblock copolymer is characterized by the order parameter $\psi(\mathbf{r}, t)$ which represents the differences in the local volume fraction for the copolymer A and B

$$\psi(\mathbf{r}, t) = \phi_A(\mathbf{r}, t) - \phi_B(\mathbf{r}, t) + (1 - 2f_0) \tag{2}$$

with respect to the relative volume fraction of A monomers in the diblock, $f_0 = N_A/(N_A + N_B)$, where N_A and N_B represent the number of repeated units of A or B kind, respectively, in a block copolymer chain.
The dynamics of the order parameter is dictated by the Cahn-Hilliard equation, following mass conservation of $\psi(\mathbf{r}, t)$,

$$\frac{\partial \psi(\mathbf{r}, t)}{\partial t} = M \, \nabla^2 \left(\frac{\delta \mathcal{F}_{tot}[\psi]}{\delta \psi} \right) \tag{3}$$

where M is a phenomenological mobility constant.

The copolymer free energy is a functional of the local order parameter which can be expressed in terms of the thermal energy $k_B T$ as[52]

$$\begin{aligned} \mathcal{F}_{pol}[\psi(\mathbf{r})] = \int d\mathbf{r} \left[H(\psi) + \frac{1}{2} D |\nabla\psi|^2 \right] + \\ \frac{1}{2} B \int d\mathbf{r} \int d\mathbf{r}' \; G(\mathbf{r} - \mathbf{r}')\psi(\mathbf{r})\psi(\mathbf{r}') \end{aligned} \tag{4}$$

where the first and second terms are the short and the long-range interaction terms respectively, the coefficient D is a positive constant that accounts for the cost of local polymer concentration inhomogeneities, the Green function $G(\mathbf{r} - \mathbf{r}')$ for the Laplace equation satisfies $\nabla^2 G(\mathbf{r} - \mathbf{r}') = -\delta(\mathbf{r} - \mathbf{r}')$, B is a parameter that introduces a chain-length dependence to the free energy[53] and $H(\psi)$ is the local free energy [53, 54] ,

$$\begin{aligned} H(\psi) = \frac{1}{2} \left[-\tau_0 + A(1 - 2f_0)^2 \right] \psi^2 \\ + \frac{1}{3} v(1 - 2f_0)\psi^3 + \frac{1}{4}\psi^4 \end{aligned} \tag{5}$$

where τ_0, A, v, u are phenomenological parameters[55] which can be related to the block-copolymer molecular specificity.
We can express the time evolution of ψ , Equation 3, using CDS as

$$\begin{aligned} \psi(\mathbf{r}_i, t+1) = \psi(\mathbf{r}_i, t) - \delta t \{ \langle\langle \Gamma(\mathbf{r}_i, t) \rangle\rangle \\ - \Gamma(\mathbf{r}_i, t) + B[1 - P(\mathbf{r}_i, t)\psi(\mathbf{r}_i, t)] \} \end{aligned} \tag{6}$$

$\mathbf{r}_i$ being the position of the node i at a time $t\delta t$, and the isotropic discrete laplacian for a quantity X is given by [56, 57, 58] $\frac{1}{(\delta x)^2}[\langle\langle X \rangle\rangle - X]$. In three dimensions,

$$\langle\langle \psi \rangle\rangle = \frac{6}{80} \sum_{NN} \psi + \frac{3}{80} \sum_{NNN} \psi + \frac{1}{80} \sum_{NNNN} \psi \tag{7}$$

NN, NNN, NNNN meaning nearest neighbours, next-nearest neighbours, and next-next-nearest neighbours, respectively. A scheme of the stencil in 2D can be found in figure S1 in the supporting information.
In Equation 6 we have introduced the auxiliary function

$$\Gamma(\mathbf{r}, t) = g(\psi(\mathbf{r}, t)) - \psi(\mathbf{r}, t) + D \left[\langle\langle \psi(\mathbf{r}, t) \rangle\rangle - \psi(\mathbf{r}, t) \right] \tag{8}$$

and also, the map function [59, 55]

$$g(\psi) = -\tau'\psi - v(1 - 2f)\psi^2 - u\psi^3 \tag{9}$$

2.2 Parallel CDS scheme

Coarray Fortran provides an efficient abstraction of the MPI parallelisation scheme. It naturally allows for a partition of a large system into smaller parts, which are divided into images. Each image performs an execution of the whole program, except from user-specified conditions. While each image has its own workspace, co-dimensions can be used to share values across images.
Thanks to this partition utility, we break-up the system size $V = L_x \times L_y \times L_z$ into $n_{images} = N_x \times N_y \times N_z$ images with one processor per image. Images are accordingly distributed such that each partition spans a region of system space $V_{partition} = \Delta_x \Delta_y \Delta_z$ with $\Delta_\alpha = L_\alpha / N_\alpha$ and $\alpha = x, y$ and z. Computationally, each image possess a local ψ version of the order parameter scalar field. This local image is allocated a size $(\Delta_x + 2) \times (\Delta_y + 2) \times (\Delta_z + 2)$, in order to allow space to read from neighbouring processors. Ghost points are indeed needed only in the exchange of information required to calculate the Laplacian in the

Cahn-Hilliard equation. In this work we make use a single-layer halo exchange method performed just before the calculation of the laplacian of $\nabla^2\psi$ and $\nabla^2\mu$. A flowchart of both the serial and the parallel algorithm is provided in figure S4 in the supporting information. This communication step is considerably simple using CAF, which can be executed in a single line to read the values of ψ boundaries (and μ) from neighboring processors into the local version of the array. An example where processor $[p, q, h]$ communicates with its next neighbor in the X direction $[p-1, q, h]$ is

```
psi(1,2:(DY+1),2:(DZ+1)) =  psi(DX+2,2:(DY+1),2:(DZ+1))[p-1,q,h]
```

where the square brackets indicate the processor index from which data is read. In this example the local processor reads a two-dimensional surface into the local array (the colon in the second and third index indicating a range of values). On the other hand, MPI communications require specifying additional details which are unnecessary in CAF, as CAF is particularly adapted for numeric array communications. Therefore, CAF can be used as a simplified way requiring less modifications of the original code and technical knowledge over parallelisation. CAF is particularly well-suited for scientific computing applications involving arrays. Because the CDS scheme is strongly short ranged, we do not require any long range communication as opposed to Fast Fourier Transforms [60].

2.3 Colloidal dynamics

Contrary to the continuous block copolymer description, colloidal NPs are individually resolved. The presence of a number N_p of colloidal NPs into the BCP melt is introduced via a coupling term in the free energy

$$\mathcal{F}_{cpl} = \sum_{i=1}^{N_p} \sigma \int d\mathbf{r}\ \psi_c(\mathbf{r})\,[\psi(\mathbf{r}) - \psi_0]^2 \tag{10}$$

where ψ_0 represents the colloidal chemical affinity while the tagged function ψ_c relates to the size and shape of the particle[61]. A smoothly decreasing function is chosen to represent the core and corona of the NP

$$\psi_c = \exp\left[1 - \frac{1}{1 - \left(\frac{|\mathbf{r}-\mathbf{R}_i|}{R_{eff}}\right)^2}\right] \tag{11}$$

where the cut-off of the coupling interaction is R_{eff} such that $\psi_c(r \geq R_{eff}) = 0$.
Additionally, the colloid-colloid contribution to the free energy can be expressed as

$$\mathcal{F}_{cc} = \sum_{ij} V(\mathbf{r}_i - \mathbf{r}_j) \tag{12}$$

with V describing the inter-colloidal pairwise additive potential. A purely repulsive soft potential such as a Yukawa-like potential has been previously used to allow to select a large time step[62].
The colloidal dynamics is diffusive and governed by the Langevin equation in the over-damped regime. Each NP centre of mass $\mathbf{R}_i$ follow Brownian Dynamics as

$$\frac{d\mathbf{R}_i}{dt} = \frac{1}{\gamma}\left(\mathbf{f}_i^{cc} + \mathbf{f}_i^{cpl} + \sqrt{2k_BT\gamma}\xi\right) \tag{13}$$

with γ the friction coefficient, k_BT is the NP thermal energy and ξ is a random Gaussian term satisfying the Fluctuation–Dissipation theorem.

2.4 Parallel hybrid CDS/Brownian Dynamics algorithm

Adding NPs into a parallel CDS simulation model requires an efficient parallelisation of the calculation of the BCP/NP coupling (forces and chemical potential) along with the inter-particle forces. Using CAF,

it is essential to have a spatial decomposition of the system. We choose to respect the same spatial decomposition into N_x, N_y, N_z processors, respectively assigned to $\Delta_x, \Delta_y, \Delta_z$ subregion of the system, as described in the previous section. To this end it is desirable to use a Cell List method [63], in which the system size is divided into cells and particles are sorted into linked list. We use a number $N_{cells} = M_x^c M_y^c M_z^c$ number of cells. Therefore, each processor is dedicated to particles in the region occupied by their respective cells. A schematic flowchart of the serial and parallel algorithm can be found in figure S4 in the supporting information, showing a single iteration of the algorithm.
The calculation of the coupling forces and chemical potentials is non-local, involving an spatial integral as shown in equation 10. In order to obtain the forces acting on each particle, first loop through all particles in cells within the local processor. Given the finite size of the NPs, it is crucial to account for the overlapping of particles into neighbouring processors, ie, a particle volume can span over several processors' sub-domains, even if the centre of mass of the particle is within a processor different from the local one. For this reason we perform a two-step calculation: First, a loop over neighboring processors' cells is performed to search for NPs overlapping the local processor domain. Then, the coupling contributions of these particles is added to the chemical potential and forces.
A similar two-step process is used for the calculation of colloid-colloid forces: first the forces for pairs of particles within the local processor are calculated and afterwards the calculation is extended for neighboring cells. Once all forces are determined (coupling and inter-particle), the positions of particles are updated and communicated within processors.

2.5 Analysing the parallel scheme

In order to analyse the efficiency of parallel implementations, it is common to test the scaling of the code with the number of processors for a fixed system size (strong scaling) or for a fixed amount of elements per processor (weak scaling). The speed-up of a parallel algorithm is defined as

$$S(m, n_p) = \frac{T(m,1)}{T(m,n_p)} \tag{14}$$

where m is the size of the system, ie, the number of grid points and the number of processors is denoted by n_p.

3 Results

In this section we will characterize the efficiency of the implementation in various supercomputers: Mare Nostrum, CSCS, ARCHER and the cluster present at the University of Lincoln (UoL). In section 1 of the supplementary information we provide detailed information on each supercomputer specifications as well as the scaling results in these machines.

3.1 Scaling of the CDS model

The correct scaling of the purely polymeric code ($N_p = 0$) should provide the foundation of the more complex hybrid code. An appropriate parallel implementation should ideally reduce the required time linearly with the number of processors in use. In order to test this behaviour, we can explore the effect of parallelisation of a system with size $V = L_x L_y L_z$ into $n_p = N_x N_y N_z$ processors.
The strong scaling of the CDS code is shown in figure 1 for three system sizes $V = 128^3, 256^3$ and 512^3 in the CSCS supercomputer, using 8 cores per node, that is, spanning up to 1024 cores for the largest system size. A considerably linear -close to ideal, in dashed lines- behaviour can be observed up to 16, 32 and 64 nodes for increasingly larger system sizes $V = 128^3$, 256^3 and 512^3, respectively. Larger system sizes lead to larger in-processor calculations versus inter-processor, therefore, the increase in performance with the lateral system size is explained. This is particularly clear in the drop in performance for $V = 256^3$, where the use of 64 nodes (512 cores) leads to a larger computational time than using 32

nodes. This is expected as the number of grid points per core become considerably small $V_{partition} = 32^3$, with the communications between processors dominating the computational time. A direct comparison of the scaling of the reported MPI code and the present CAF scheme is shown in figure S3 in the supporting information. The present CAF scaling displays no drawbacks over MPI, showcasing the efficiency of the Coarray approach. Figure S2 in the supplementary information displays the strong scaling in other supercomputers, finding a good scaling in other architectures and compilers limited to single node.

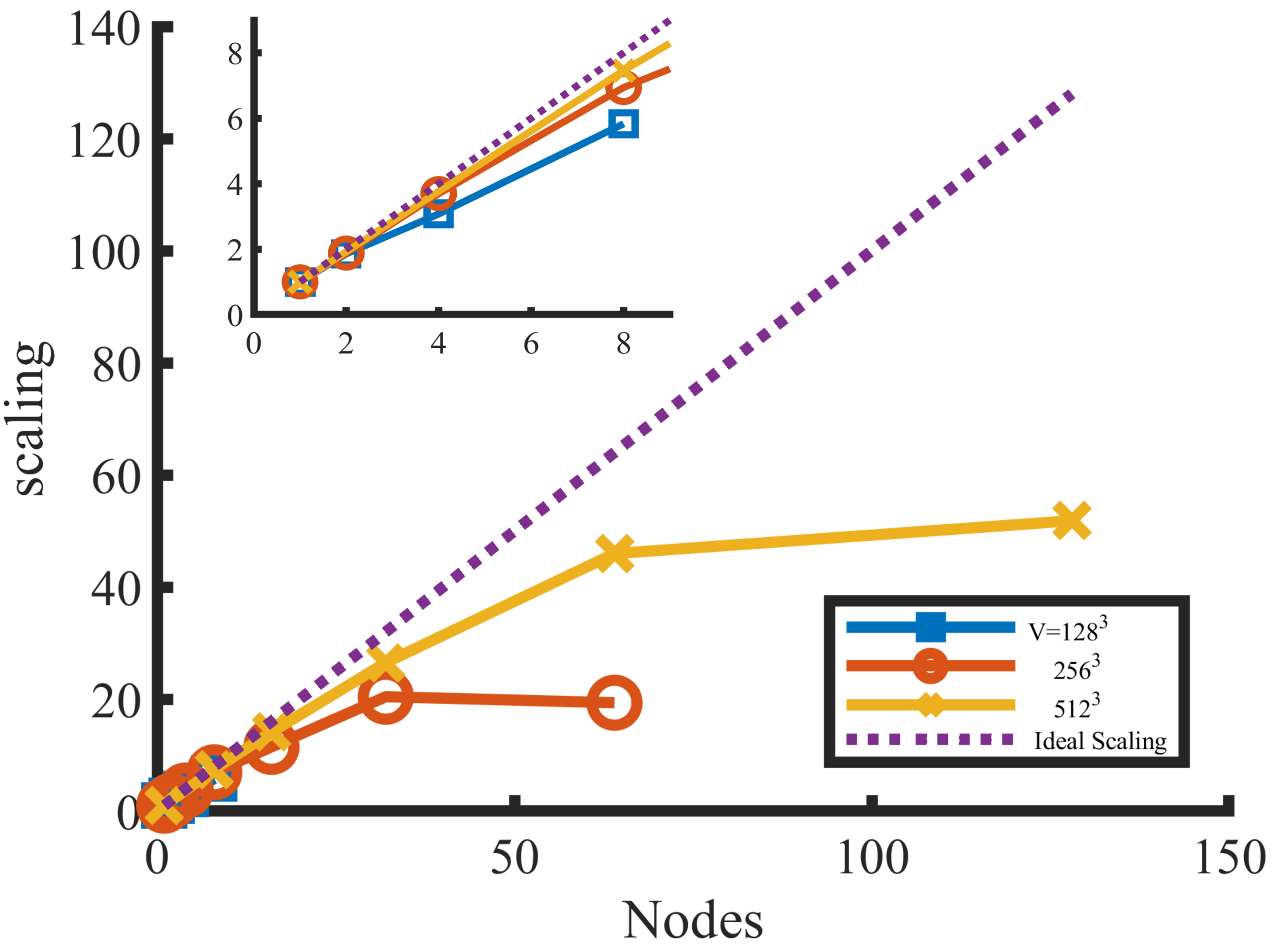


Figure 1: Strong scaling of the purely CDS polymeric part in the CSCS supercomputer for three system sizes $V = L^3$ vs the number of nodes, using 8 cores per node. In the inset we provide a detail of the small number of nodes regime.

The ability to reach large number of processors is best shown in the weak scaling analysis, where the system size is increased linearly with the number of nodes, in order to keep the sub-domain per core (referred to as *images* in Coarray Fortran terms) constant. In figure 2 we can observe the scaling $t(1)/t(n_p) \sim 1$ in a semi-logarithmic plot, which can be seen to weakly decrease with the number of nodes. Nonetheless, the scaling never decreases below 0.85, motivating the possibility to simulate considerably large system sizes, for example $V = 2048^3$ for 512 nodes, ie, 4096 cores. The performance drop can be attributed to the large system size and large number of processors. Synchronisation is required after communications, which can decrease the performance of the simulation as the number processors is increased.

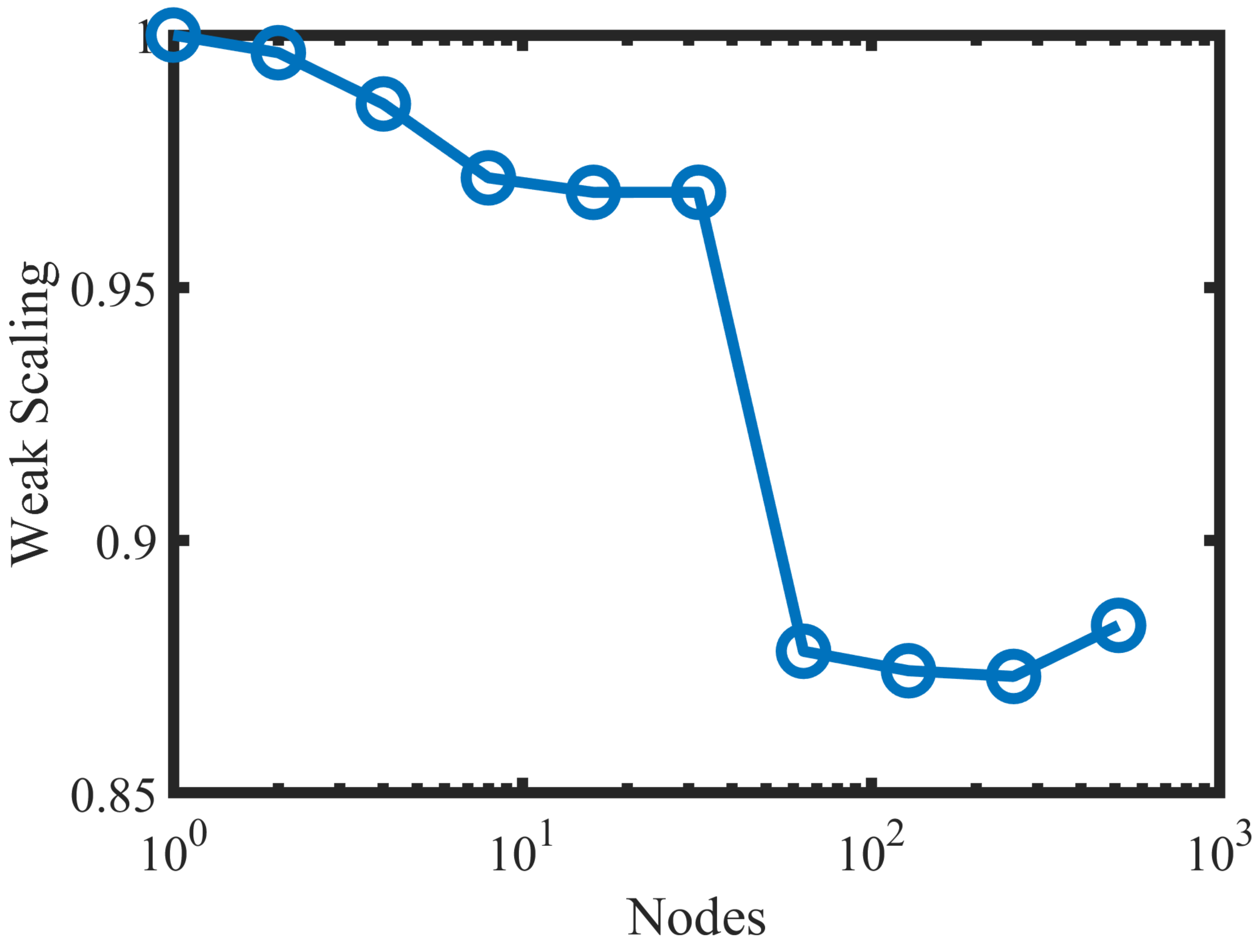


Figure 2: Weak scaling of the purely CDS polymeric part in the CSCS supercomputer, using 8 processors per node. The scaling of the wall clock time is done with respect to the single node simulation.

3.2 Scaling of hybrid CDS/BD

The scaling of the hybrid CDS/Brownian Dynamics scheme is more complex than the purely polymeric program. For this reason, it is useful to separately explore many of the contributions to the total elapsed time. Firstly, we can study how the program scales with the number of particles in the system. Figure 3 shows the elapsed time in function of the number of particles in the system for a system $V = 128^3$ and 8 processors for $N_{steps} = 1000$. The scaling is seen to be strongly linear with the number of particles. This is expected, as the two main colloid-dependent contributions to the time of the program are: colloid-colloid interaction and coupling between BCP/NP. The colloid-colloid calculation of forces naively scale with $(N_p)^2$, but can be reduced to a $(N_p)^1$ scaling by using a cell list scheme [63].

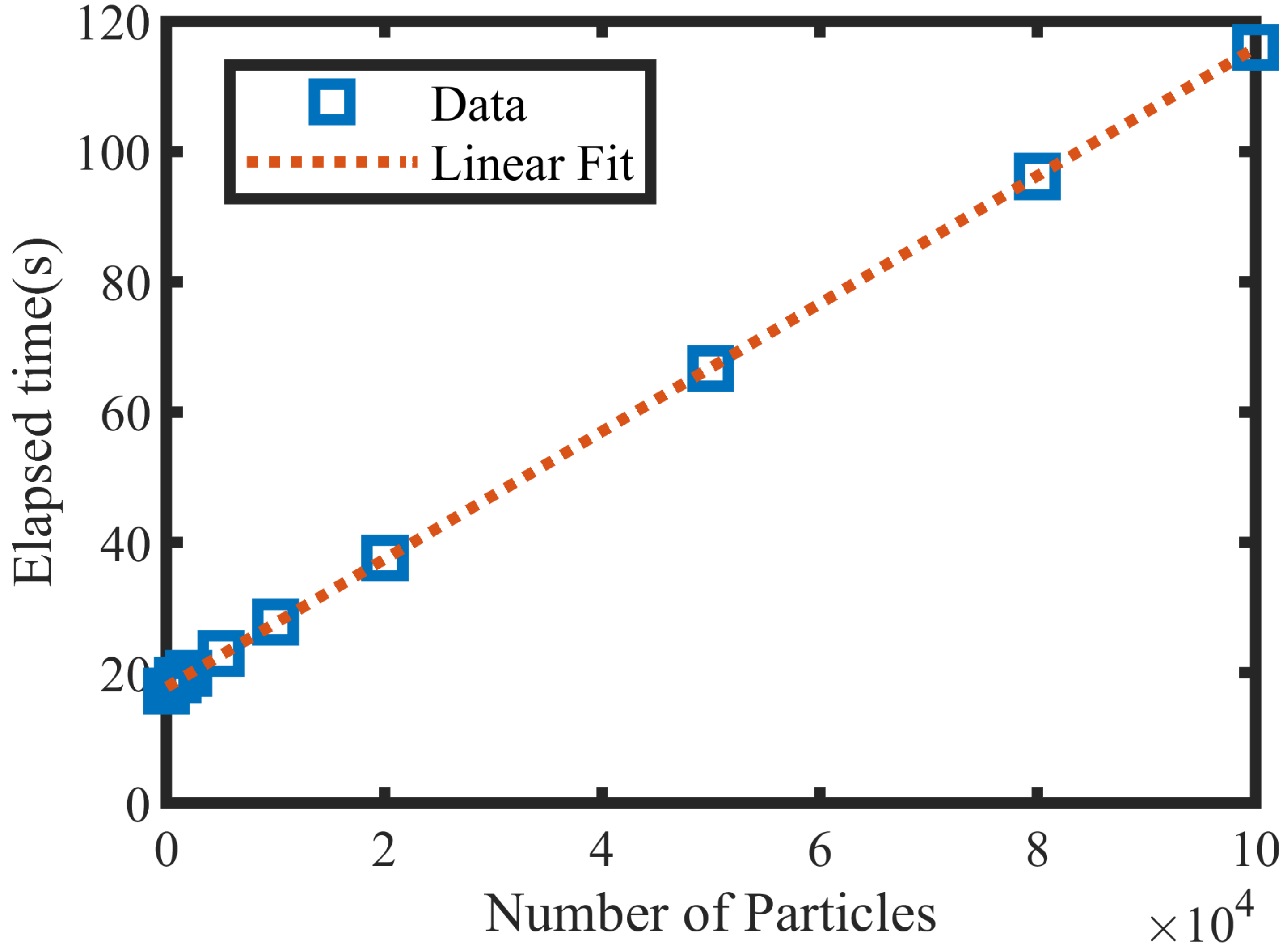


Figure 3: Scaling of the elapsed time with respect to the number of particles N_p. The number of processors and system sizes is fixed: $n_p = 8, V = 128^3$. The particles size is set to $R_{eff} = 1.56$ grid points. These simulations are performed on ARCHER.

Figure 3 suggests that the NP contribution to the computational time is not negligible for relevant number of particles, as the elapsed time more than doubles for $N_p \sim 10^4$ compared with $N_p = 0$. For this reason, it is essential to use an efficient NP algorithm. In a second step, we can differentiate between the coupling and the colloid-colloid contributions to the computational time. Figure 4 shows the elapsed time with respect to N_p, differentiating two values of the coupling constant introduced in equation 10: $\sigma = 1$ and 0. In the present code, the program skips the BCP-NP coupling calculations if the coupling constant σ is set to zero. Therefore, it is an easy way to separate different contribution to the elapsed time. We can observe that both the total and the colloid-colloid times are strongly linear with N_p, while the coupling is clearly the most important contribution. This suggests that the used cell-list scheme is considerably efficient as it is sub-dominant over the more computationally-heavy coupling calculations. Additionally, the linearity of the coupling contribution suggest that the neighbour cell search algorithm described in section 2.4 is efficiently implemented. Further details on the effect of the different steps (see flowchart in figure S4 in the supporting information) can be found in the bar chart in figure S7 in the supporting information where the total time is broken-down into in-processor and communication steps for polymeric, coupling and colloid parts of the algorithm. The colloid-colloid force calculation, colloid position update and colloid communications are found to be always negligible compared to computational time of the coupling forces. Nonetheless, the weight of the coupling itself compared to the polymeric part is strongly dependent on the concentration of particles, as already demonstrated in figure 3 and 4. These results suggest that the largest contribution to the elapsed time depends on the concentration of particles in the system: low concentration simulations spend the biggest share of the simulation time calculating the laplacians of ψ and μ, while for high concentration simulations the bottleneck of the simulation is the coupling step. Finally, the coupling force calculation clearly scales as N_pR^3, as the force calculation involves triple loops around the particle centre of mass.

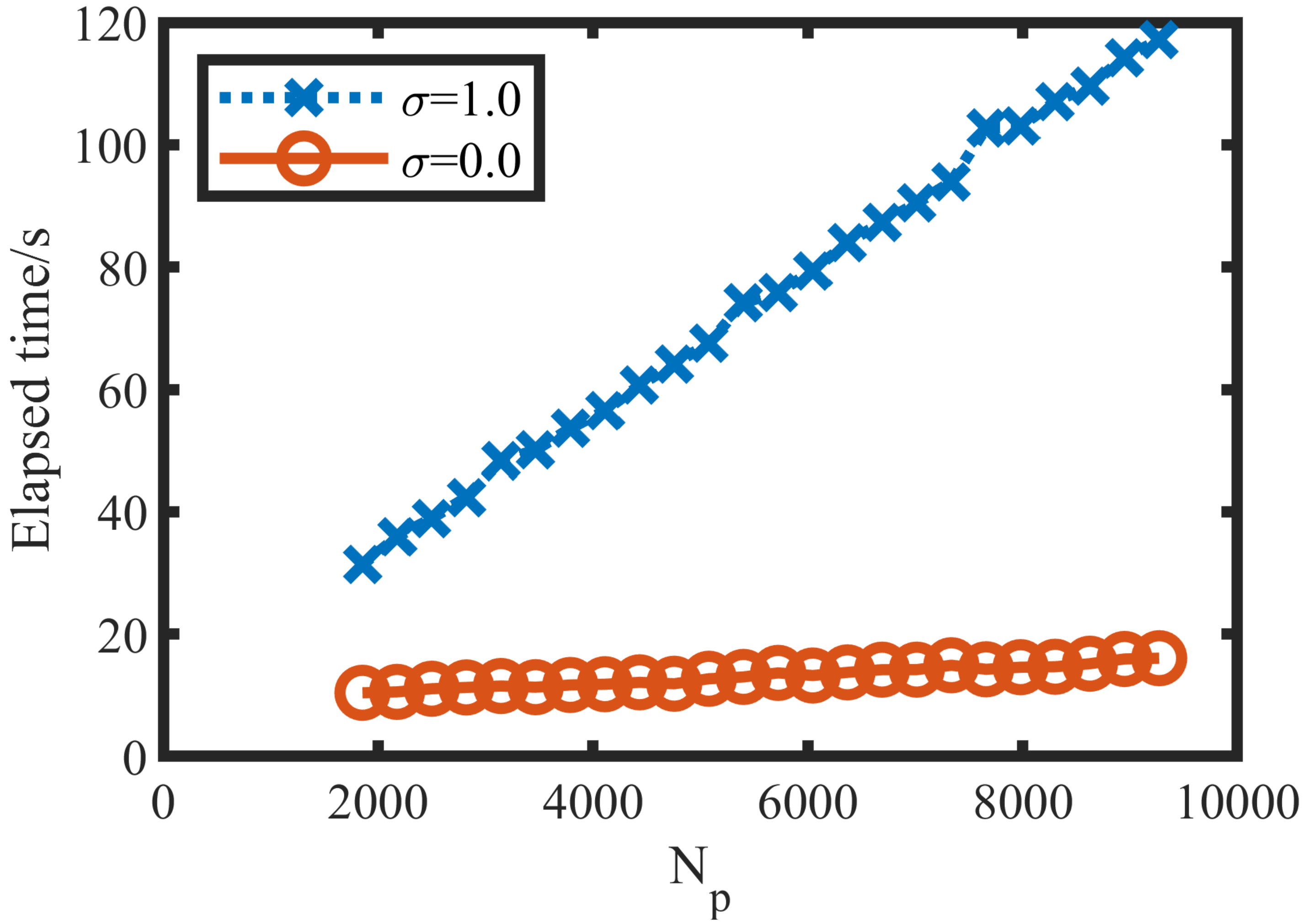


Figure 4: Simulation time (8 cores, UoL cluster) dependence with the number of particles N_p for colloids with a size $R_{eff} = 2.34$. Two sets are produced: for $\sigma = 1$ and $\sigma = 0$. In the latest the coupling calculations are ignored, which can be seen to lead to a considerable reduction in the elapsed time.

In Figure 5 the strong scaling for a system sized $V = 256^3$ with $N_p = 10^4$ NPs is shown, with simulations performed in the CSCS cluster using 8 processors per node. The NP size is set to $R_{eff} = 1.56$. The scaling behaviour is considerably close to ideal for 8 nodes ($n_p = 32$ processors) but the curve drops in efficiency faster than the purely polymeric scaling shown in figure 1 for 16 nodes. A similar behaviour is shown in figure S6 in the Supporting Information for the Mare Nostrum cluster, where the efficiency similarly drops for a larger system.

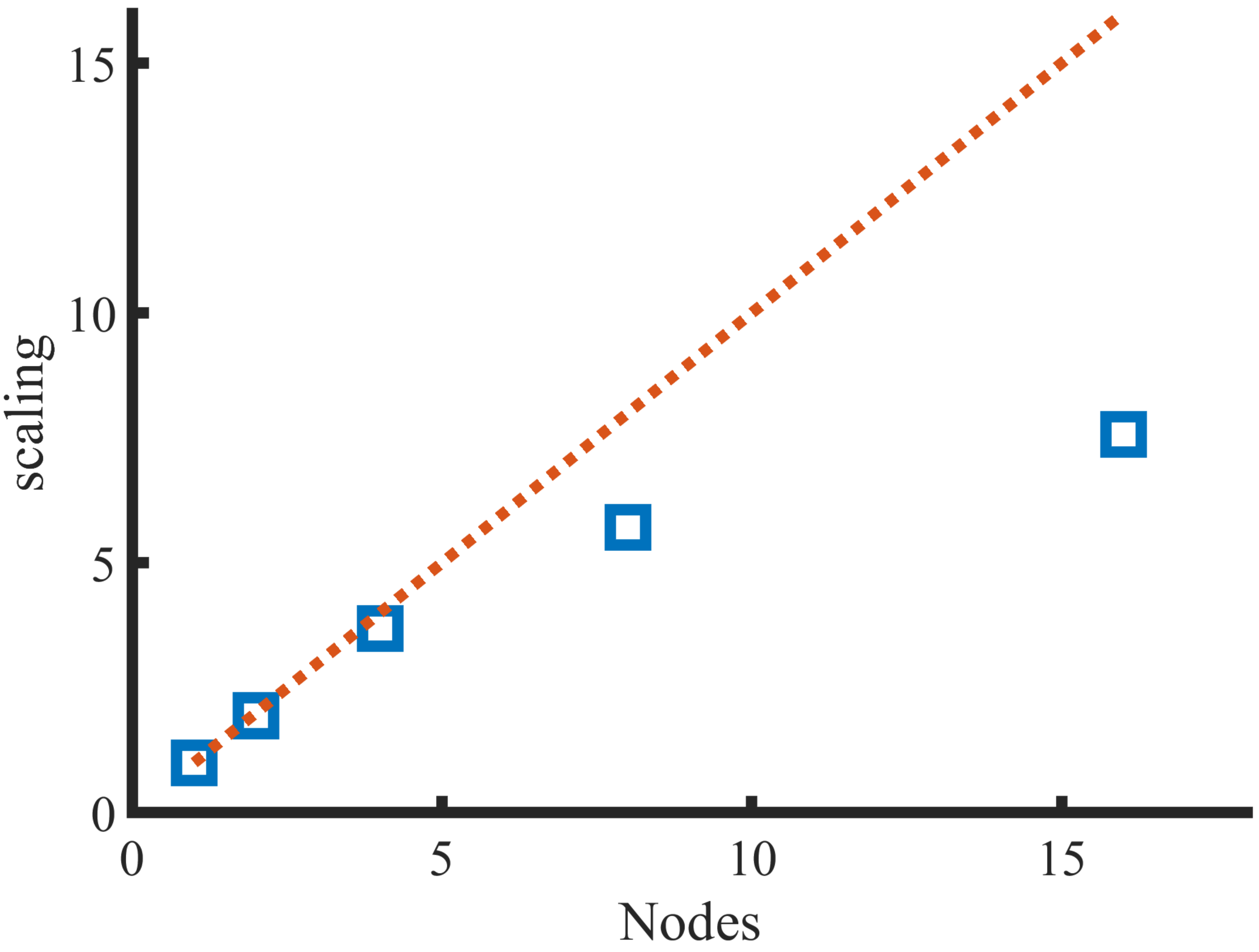


Figure 5: Strong scaling of the hybrid BCP/NP code vs the number of nodes. The speed-up S is compared with the ideal, linear scaling(dotted line). The number of particles is $N_p = 10^4$ in a $V = 256^3$ system performed in CSCS, using 8 cores per node. The coupling constant is set to $\sigma = 1$.

Due to the parallel implementation that has been described, we can achieve previously unavailable system sizes. In Figure 6 a large simulation box $V = 400 \times 400 \times 300$ with $N_p = 100$ is shown. The NP size is relatively large with a radius $R_{eff} = 13.26$ grid points immersed in a symmetric BCP mixture. The parallel code allows to simulate a large number of lamellar periods along with colloidal NPs.

4 Conclusions

Coarray Fortran has been used to develop a parallel code of the well-established CDS scheme for BCP melts and BCP nanocomposites systems. This relatively simple approach based on spatial decomposition shows no drawbacks when compared with a more elaborate method using MPI [60]. The scaling of the pure CDS code is highly linear for relatively large system sizes and improves the previous implementation using MPI. It allows to scale up large system sizes while maintaining reasonable computational times: the weak scaling test has shown the ability to simulate systems with 2048^3 grid points using 512 nodes, with a small drop in the performance. This code has been tested in various national and academic-wide supercomputers. The best scaling behaviour has been found using the CRAY Fortran

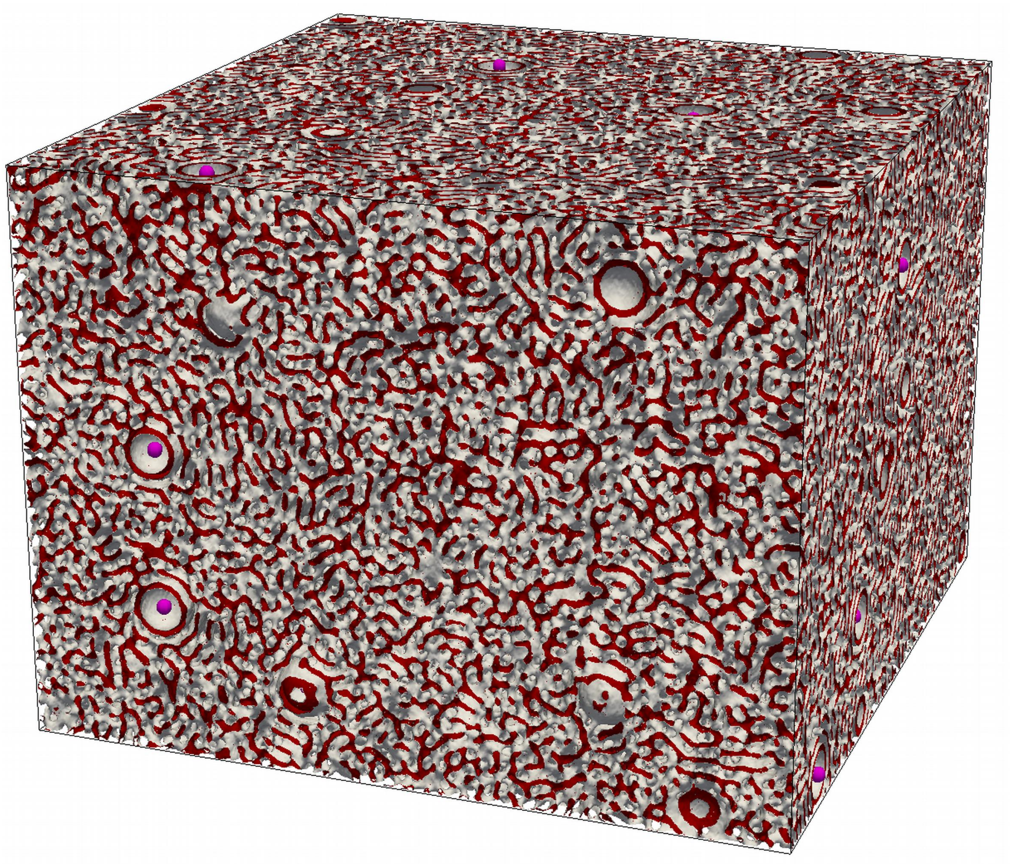


Figure 6: Large scale simulation of relatively large NPs with radius $R_{eff} = 13.26$ selective towards one of the block $\psi_0 = -1$ of a lamellar-forming BCP $f_0 = 1/2$.

compiler in the CSCS supercomputer.
The ability of this algorithm to scale up to a large number of computer nodes allow us to reach previously unavailable system sizes. For instance, the largest system size of lateral size 2048 grid points can model 128 BCP periods. This would correspond to approximately $V \sim (5\mu m)^3$ for typical lamellar periodicities.
Secondly, a scheme for a parallel hybrid BCP/NP mesoscopic model has been presented. Similarly as the purely polymeric CAF implementation, this code has the ability to scale linearly with the number of processors in use, although the range of linearity is smaller than for the purely polymeric algorithm. The actual scaling of the code depends heavily on colloidal size -due to its influence in the cell size- but the code has been found to perform efficiently. This parallel code opens the possibility to study more complex systems involving block copolymer nanocomposites at considerably large length scales[62].

Supporting Information
Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgments

The work has been performed under the project HPC-EUROPA3 (INFRAIA-2016-1-730897), with the support of the EC Research Innovation Action under the H2020 Programme; in particular, the authors gratefully acknowledges the computer resources and technical support provided by Barcelona Supercomputing Center (BSC). I. P. acknowledges support from MINECO (Grant No. PGC2018-098373-B-100), DURSI (Grant No. 2017 SGR 884) and SNF Project No. 200021-175719. This work used the ARCHER UK National Supercomputing Service (http://www.archer.ac.uk). The authors are grateful to Dr Sergey Chulkov for HPC support.

References

[1] A. K. Khandpur, S. Foerster, F. S. Bates, I. W. Hamley, A. J. Ryan, W. Bras, K. Almdal, K. Mortensen, *Macromolecules* **1995**, *28*, 26 8796.

[2] F. S. Bates, M. F. Schulz, A. K. Khandpur, S. Förster, J. H. Rosedale, K. Almdal, K. Mortensen, *Faraday Discussions* **1994**, *98*, 0 7.

[3] M. W. Matsen, M. Schick, *Physical Review Letters* **1994**, *72*, 16 2660.

[4] M. W. Matsen, F. S. Bates, *Macromolecules* **1996**, *29*, 4 1091.

[5] M. W. Matsen, F. S. Bates, *Macromolecules* **1996**, *29*, 23 7641.

[6] A. V. Zvelindovsky, G. J. A. Sevink, B. A. C. van Vlimmeren, N. M. Maurits, J. G. E. M. Fraaije, *Physical Review E* **1998**, *57*, 5 R4879.

[7] Q. Zhang, J. Lin, L. Wang, Z. Xu, *Progress in Polymer Science* **2017**, *75* 1.

[8] C. W. Pester, C. Liedel, M. Ruppel, A. Böker, *Progress in Polymer Science* **2017**, *64* 182.

[9] G. Krausch, R. Magerle, *Advanced Materials* **2002**, *14*, 21 1579.

[10] M. Serral, M. Pinna, A. V. Zvelindovsky, J. B. Avalos, *Macromolecules* **2016**, *49*, 3 1079.

[11] D. Q. Ly, M. Pinna, T. Honda, T. Kawakatsu, A. V. M. Zvelindovsky, *The Journal of Chemical Physics* **2013**, *138*, 7 074904.

[12] M. Pinna, A. V. M. Zvelindovsky, X. Guo, C. L. Stokes, *Soft Matter* **2011**, *7*, 15 6991.

[13] C. Xu, K. Ohno, V. Ladmiral, D. E. Milkie, J. M. Kikkawa, R. J. Composto, *Macromolecules* **2009**, *42*, 4 1219.

[14] R. B. Thompson, *Science* **2001**, *292*, 5526 2469.

[15] M. W. Matsen, R. B. Thompson, *Macromolecules* **2008**, *41*, 5 1853.

[16] C.-T. Lo, Y.-C. Chang, S.-C. Wu, C.-L. Lee, *Colloids and Surfaces A: Physicochemical and Engineering Aspects* **2010**, *368*, 1-3 6.

[17] Y. Wang, Y. Han, J. Cui, W. Jiang, Y. Sun, *Langmuir* **2016**, *32*, 33 8484, publisher: American Chemical Society.

[18] T. N. Hoheisel, K. Hur, U. B. Wiesner, *Progress in Polymer Science* **2015**, *40* 3.

[19] B. Sarkar, P. Alexandridis, *Progress in Polymer Science* **2015**, *40* 33.

[20] L.-T. Yan, X.-M. Xie, *Progress in Polymer Science* **2013**, *38*, 2 369.

[21] Z. Hou, M. Ren, K. Wang, Y. Yang, J. Xu, J. Zhu, *Macromolecules* **2019**.

[22] K. Tsuchiya, S. Nagayasu, S. Okamoto, T. Hayakawa, T. Hihara, K. Yamamoto, I. Takumi, S. Hara, H. Hasegawa, S. Akasaka, N. Kosikawa, *Optics Express* **2008**, *16*, 8 5362.

[23] E. Ploshnik, A. Salant, U. Banin, R. Shenhar, *Advanced Materials* **2010**, *22*, 25 2774.

[24] E. Ploshnik, A. Salant, U. Banin, R. Shenhar, *Physical Chemistry Chemical Physics* **2010**, *12*, 38 11885.

[25] E. Ploshnik, K. M. Langner, A. Halevi, M. Ben-Lulu, A. H. E. Müller, J. G. E. M. Fraaije, G. J. Agur Sevink, R. Shenhar, *Advanced Functional Materials* **2013**, *23*, 34 4215.

[26] J. Diaz, M. Pinna, A. V. Zvelindovsky, I. Pagonabarraga, R. Shenhar, *Macromolecules* **2020**.

[27] V. V. Ginzburg, F. Qiu, A. C. Balazs, *Polymer* **2002**, *43*, 2 461.

[28] H. Chen, E. Ruckenstein, *The Journal of Chemical Physics* **2009**, *131*, 24 244904.

[29] G. J. A. Sevink, A. V. Zvelindovsky, *Macromolecules* **2005**, *38*, 17 7502.

[30] T. Xu, A. V. Zvelindovsky, G. J. A. Sevink, K. S. Lyakhova, H. Jinnai, T. P. Russell, *Macromolecules* **2005**, *38*, 26 10788.

[31] A. J. Schultz, C. K. Hall, J. Genzer, *Macromolecules* **2005**, *38*, 7 3007.

[32] F. A. Detcheverry, H. Kang, K. C. Daoulas, M. Müller, P. F. Nealey, J. J. de Pablo, *Macromolecules* **2008**, *41*, 13 4989.

[33] H. Kang, F. A. Detcheverry, A. N. Mangham, M. P. Stoykovich, K. C. Daoulas, R. J. Hamers, M. Müller, J. J. de Pablo, P. F. Nealey, *Physical Review Letters* **2008**, *100*, 14.

[34] J. Huh, V. V. Ginzburg, A. C. Balazs, *Macromolecules* **2000**, *33*, 21 8085.

[35] M. R. Bockstaller, Y. Lapetnikov, S. Margel, E. L. Thomas, *Journal of the American Chemical Society* **2003**, *125*, 18 5276.

[36] H. Chen, E. Ruckenstein, *Journal of Colloid and Interface Science* **2011**, *363*, 2 573.

[37] P. Posocco, Z. Posel, M. Fermeglia, M. Lísal, S. Pricl, *Journal of Materials Chemistry* **2010**, *20*, 46 10511.

[38] M. Pinna, L. Schreier, A. V. Zvelindovsky, *Soft Matter* **2009**, *5*, 5 970.

[39] J. Diaz, M. Pinna, A. V. Zvelindovsky, I. Pagonabarraga, *Macromolecules* **2019**, *52*, 21 8285.

[40] M. Metcalf, J. Reid, M. Cohen, *Modern Fortran Explained*, Oxford University Press, Oxford ; New York, edición: 4 edition, **2011**.

[41] R. W. Numrich, J. Reid, *ACM SIGPLAN Fortran Forum* **1998**, *17*, 2 1.

[42] A. Fanfarillo, T. Burnus, V. Cardellini, S. Filippone, D. Nagle, D. Rouson, In *Proceedings of the 8th International Conference on Partitioned Global Address Space Programming Models*. **2014** 1–11.

[43] A. Shterenlikht, L. Margetts, J. D. Arregui-Mena, L. Cebamanos, In *2016 PGAS Applications Workshop (PAW)*. IEEE, **2016** 1–8.

[44] A. Shterenlikht, L. Cebamanos, In *Proceedings of the 25th European MPI Users' Group Meeting*. **2018** 1–10.

[45] N. T. Weeks, G. R. Luecke, B. M. Groth, M. Kraeva, L. Ma, L. M. Kramer, J. E. Koltes, J. M. Reecy, *The International Journal of High Performance Computing Applications* **2018**, *32*, 3 321, publisher: SAGE Publications Ltd STM.

[46] M. Curcic, *ACM SIGPLAN Fortran Forum* **2019**, *38*, 1 4.

[47] G. Mozdzynski, M. Hamrud, N. Wedi, J. Doleschal, H. Richardson, In *2012 SC Companion: High Performance Computing, Networking Storage and Analysis*. IEEE, **2012** 652–661.

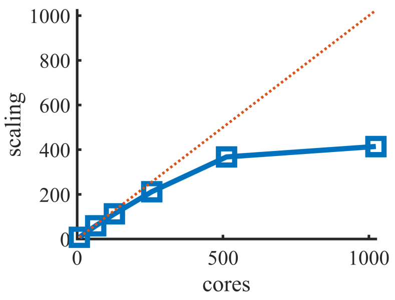
[48] A. Fanfarillo, S. K. Garain, D. Balsara, D. Nagle, *Parallel Computing* **2019**, *81* 58.

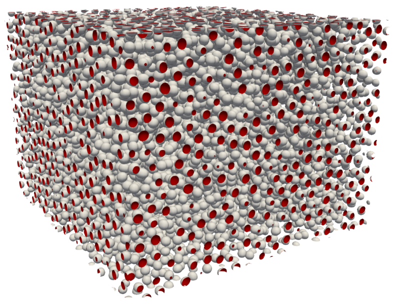
[49] S. Garain, D. S. Balsara, J. Reid, *Journal of Computational Physics* **2015**, *297* 237.

[50] J. Reid, B. Long, J. Steidel, *Proceedings of the ACM on Programming Languages* **2020**, *4*, HOPL 72:1.

[51] M. Pinna, I. Pagonabarraga, A. V. Zvelindovsky, *Macromolecular Theory and Simulations* **2011**, *20*, 8 769.

Table of Contents





A parallel Cell Dynamic Simulation algorithm allows to reach previously unavailable system sizes which are essential to capture the long-range behaviour of block copolymer nanocomposite systems. Coarray Fortran displays no drawbacks over previous MPI implementations allowing to scale up to thousands of processors. A hybrid in-grid/out-of-grid model is parallelised showing a good scaling up to hundreds of processors.

[52] T. Ohta, K. Kawasaki, *Macromolecules* **1986**, *19*, 10 2621, publisher: American Chemical Society.

[53] I. W. Hamley, *Macromolecular Theory and Simulations* **2000**, *9*, 7 363.

[54] S. R. Ren, I. W. Hamley, G. J. A. Sevink, A. V. Zvelindovsky, J. G. E. M. Fraaije, *Macromolecular Theory and Simulations* **2002**, *11*, 2 123.

[55] S. R. Ren, I. W. Hamley, *Macromolecules* **2001**, *34*, 1 116.

[56] Y. Oono, S. Puri, *Physical Review A* **1988**, *38*, 1 434.

[57] Y. Oono, S. Puri, *Physical Review Letters* **1987**, *58*, 8 836.

[58] S. Puri, Y. Oono, *Physical Review A* **1988**, *38*, 3 1542.

[59] M. Bahiana, Y. Oono, *Physical Review A* **1990**, *41*, 12 6763.

[60] X. Guo, M. Pinna, A. V. Zvelindovsky, *Macromolecular Theory and Simulations* **2007**, *16*, 9 779.

[61] H. Tanaka, T. Araki, *Physical Review Letters* **2000**, *85*, 6 1338.

[62] J. Diaz, M. Pinna, A. V. Zvelindovsky, I. Pagonabarraga, *Soft Matter* **2019**, *15*, 45 9325.

[63] M. P. Allen, D. J. Tildesley, *Computer simulation of liquids*, Oxford university press, **2017**.