Adouin, Yoann; Moulinec, Charles; Barber, Robert; Sunderland, Andrew

Preparing TELEMAC-2D for extremely large simulations

Preparing TELEMAC-2D for extremely large simulations

Yoann AUDOUIN, Charles MOULINEC, Robert W. BARBER, Andrew G. SUNDERLAND,
Xiao-Jun GU and David R. EMERSON
STFC Daresbury Laboratory
Warrington, Cheshire, WA4 4AD, UK
charles.moulinec@stfc.ac.uk

Abstract— **This paper describes the latest developments that have been carried out to prepare TELEMAC-2D for simulations using grids composed of hundreds of millions of elements. Even running modest-sized simulations involving around 2 to 10 million grid elements highlights some critical issues concerning both the grid generation and the subsequent grid pre-processing which is currently handled by the PARTEL TELEMAC system tool. A serial accelerated global mesh refinement technique is presented which allows the generation of a 425-million element grid from an existing 106-million element grid in less than an hour on a fat node of an IBM POWER7 cluster. The current version of PARTEL (version 6.0) relies on METIS 4.0 as the partitioner and has two main drawbacks for extremely large simulations; namely, METIS 4.0 is highly memory consuming, and secondly, PARTEL is extremely time-consuming when performing the rest of the pre-processing stage. Four alternative partitioners are tested on large grids, and a new parallel pre-processing tool, PARTEL_P, has been designed with the aim of optimising memory consumption. This new tool allows the pre-processing of a 200-million element grid on up to 32,768 sub-domains and its output has successfully been used to evaluate the scaling performance of TELEMAC-2D on an IBM Blue Gene/P.**

I. INTRODUCTION

The TELEMAC system [1,2] is a multi-scale hydrodynamics free-surface suite that can solve either the two-dimensional shallow water equations (TELEMAC-2D) or the Navier-Stokes equations (TELEMAC-3D) depending on the approximation made in the calculation of the velocity component in the vertical direction. The system relies on the BIEF (Bibliothèque d'Eléments Finis) finite-element library which contains the subroutines to perform the fundamental operations on scalars, vectors and matrices, the iterative solvers, and the discretisation schemes used by the hydrodynamic solvers. The present study has focused specifically on the computational properties of the shallow water equation solver, TELEMAC-2D. The various computational stages necessary to perform a simulation with TELEMAC-2D proceed as follows:

1. Generation of a grid of triangular elements with a mesh generator, and the generation of the bathymetry of the flow domain. This stage is currently performed in serial using the mesh generation tool supplied with the TELEMAC suite. However, third-party mesh generators can also be used to create the finite-element mesh. It is also possible to globally refine an existing mesh to increase the spatial resolution of the simulation; this is also performed in serial using a tool that has recently been optimised.

2. The pre-processing stage: this includes mesh partitioning, calculation of the mesh connectivity, assignment of the boundary conditions, identification of the halo cells, and pre-processing for the method of characteristics for advection. The mesh partitioning and all other pre-processing tasks are currently performed using a serial utility called PARTEL. Serial mesh partitioning is limited by memory availability whereas the rest of the pre-processing tasks are limited by both memory and time constraints.

3. Solution of the shallow water equations using TELEMAC-2D: the equations are solved either in a fully-coupled mode or with the help of a wave equation, depending on the option chosen. The spatial discretisation, in general, is linear and several advection schemes are available depending on the type of flow. Options include the method of characteristics, the streamline-upwind Petrov-Galerkin scheme (SUPG) and residual distributive schemes (such as the N-scheme and PSI-scheme). The matrix-storage in TELEMAC is edge-based and several linear solvers are available in the BIEF library, including conjugate gradient, conjugate residual, CGSTAB and GMRES solvers. TELEMAC-2D is fully parallelised using MPI.

The current work has focused on stages 1 and 2 of the solution procedure. Section II of the paper provides an overview of the computer hardware used in the present study, Section III provides a brief description of the selected test cases and Section IV explains how the global mesh refinement is carried out. Section V then describes the four partitioners considered in the study whilst Section VI details the new parallel pre-preprocessor, PARTEL_P. Finally, Section VII of the paper illustrates the scaling performance of TELEMAC-2D using a 200-million element grid.

II. HARDWARE

A. *IBM POWER7 cluster [3]*

The POWER7 cluster of large memory nodes is composed of four 'POWER 750 Express' nodes, each with

256 GB of memory and 32 processor cores clocked at 3.55 GHz. The cluster is currently linked using an InfiniBand interconnect. A single node has been demonstrated to give 674 Gflop/s Linpack using the new VSX instructions implemented on the POWER7 giving a theoretical peak floating point performance of 8 operations per clock cycle.

B. *IBM Blue Gene/P [4]*

A single rack of the Blue Gene system contains 1024 chips with four processor cores per chip in the P system, giving 4096 cores per rack in total. Memory is provided at 512 MB per core. The BlueGene/P system uses a processor from the Power 450 family running at 850 MHz. A single rack has a theoretical peak floating point performance of 13.9 Tflop/s.

C. *Local PC*

A local PC has also been used for some of the test cases. The local PC is a Linux machine with an Intel(R) Core(TM)2 Duo CPU processor clocked at 2.66 GHz. The system has 4 GB of RAM.

III. Description of the Test Cases: The Gironde Estuary and the Malpasset Dam-break

Two separate test cases have been considered in this study; the first considers tidal propagation in the Gironde Estuary [5] while the second simulates the catastrophic Malpasset dam-break flood event [6,7].

A. *The Gironde Estuary*

The first test case considers the simulation of tidal propagation in the Gironde Estuary in the south-west of France. The Gironde is formed by the confluence of the Garonne and Dordogne rivers and is the largest estuary in western Europe with a total surface area of approximately 635 km^2. The Gironde can be categorised as a *macrotidal* estuary with a mean spring tidal range of 4.5 m at the mouth of the estuary. Fig. 1 shows the extent of the computational domain which spans a distance of approximately 170 km. The open seaward boundary is located between 10 and 20 km into the Bay of Biscay whilst the two landward boundaries are located at La Réole on the Garonne and Pessac-sur-Dordogne on the Dordogne.

B. *The Malpasset dam-break*

The second test case involves the simulation of the Malpasset dam-break [6,7] which occurred in the Reyran valley in the south of France on 2nd December 1959, following a period of heavy rain. The sudden and unexpected collapse of almost the entire wall of the 66 m high, 223 m long Malpasset dam, caused a flood wave of 50 million cubic metres of water to flow into the Reyran valley. Fig. 2 shows the computational domain used in the present study.

IV. Mesh Splitting

The SELAFIN format grids used by TELEMAC-2D require knowledge of the 2-D coordinates, the bathymetry, the connectivity between the nodes, the location of each node (whether it is on a physical boundary or not) and the imposed boundary conditions. The boundary conditions are stored in an ASCII file whereas the remaining information is contained in a single binary file. Running in parallel follows the same strategy; each MPI task reads two files, one for the geometry, and the other for the parallel communications and the boundary conditions. Extra files might also be required, for example to set up the mass flow rate or the water level variations at the inlet boundaries, or the specification of meteorological data such as wind speed and direction.

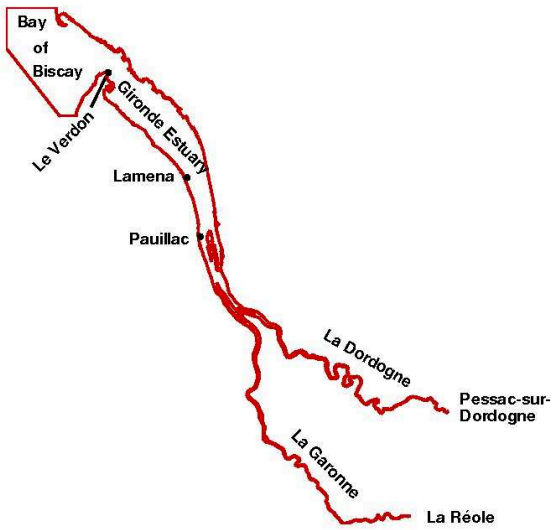

Figure 1. Computational domain for the Gironde study.

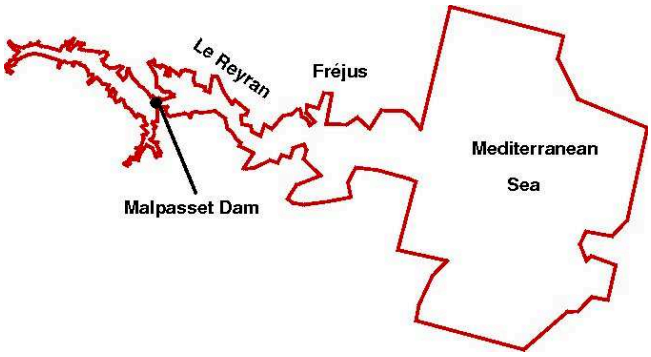

Figure 2. Computational domain for the Malpasset study.

The current hydrodynamic mesh generator in TELEMAC-2D is unable to deal with grids containing more than a few million elements. However, a serial tool, STBTEL, is available within the TELEMAC suite that can be used to globally refine a grid, starting from an initial mesh and its associated bathymetry. Each triangular element is split into four sub-triangles that retain the same aspect ratio as the original element. The tool is suitable for small meshes (i.e. less than 0.2 to 0.5 million elements) but is unable to

handle larger meshes due to memory constraints and the fact that the algorithm loops have not been optimised.

A new serial mesh-splitting tool has therefore been designed to overcome these problems and is described in the following section.

A. *Description of the mesh-splitting algorithm*

Globally refining grids by splitting each triangular element of the original mesh (OM) into four sub-triangles of the new mesh (NM) requires the determination of the mid-point of each edge of OM. These mid-points occur twice for internal faces and have different global indices whereas boundary face mid-points only occur once. A temporary new mesh connectivity list is built from all the new nodes, including the ones counted twice. This list is four times larger than OM's connectivity list. The next step consists of merging all OM's mid-points/NM's new nodes and updating the NM connectivity list. This is performed in the following stages:

- From the over-estimated list of NM's nodes, a one-dimensional array is assembled from the two-dimensional nodal coordinates using the following equation: $XY = \alpha X + \beta Y$, where (X,Y) are the nodal coordinates and (α,β) are suitable weighting coefficients. The βY term is used to differentiate nodes that have the same abscissa but different ordinates. In the present study, the weighting coefficients have been set to $(\alpha,\beta) = (10^{10},10^{-10})$. It should be noted that the one-dimensional array, XY, is stored in quadruple precision. Having to handle quadruple precision is a clear limitation since the GNU Fortran compiler is not able to support quadruple precision arithmetic. One way of overcoming this problem is to define a new data type (X,Y) which contains X and Y in double precision, and define a new 'comparison' operator such as $(X,Y) <= (X',Y')$. The piece of code for sorting would have to be modified in order to handle this new data type.
- The one-dimensional array, XY, is then sorted into ascending order so that all the nodes counted twice appear consecutively.
- The nodes are then recounted using a new global index which only counts identical nodes once. The connectivity list of NM is then updated with the new node list of global indices.

The physical boundary condition list also has to be updated accordingly. After each refinement, the bathymetry has to be linearly interpolated from the coarse grid onto the finer grid.

B. *Mesh refinement study*

The new mesh-splitting tool has been tested on both the Gironde and the Malpasset computational domains. Refining the Gironde flow domain is particularly challenging due to the presence of seven islands within the estuary. In contrast, the Malpasset refinement is somewhat simpler since the computational domain does not contain islands.

The mesh refinement process for the Gironde is illustrated in Fig. 3 which shows a section of four separate meshes of increasing spatial resolution within the estuary. The grid refinement computations were performed on a fat node of an IBM POWER7 cluster. The original mesh (0^{th} level), composed of 96,773 nodes and 188,219 (~0.2 million) elements, was successively refined to obtain meshes containing approximately 0.8, 3 and 12 million elements, respectively. The finest mesh illustrated in Fig. 3 is composed of 6,044,358 nodes and 12,046,016 elements.

Table I details the computational time required for the mesh refinement process on an IBM POWER7 cluster and shows the computational efficiency of the numerical algorithm. The mesh refinement process was continued for five levels of refinement, yielding a final mesh composed of 96,453,546 nodes and 192,736,256 elements. The smallest elements are approximately 5 m in size on the coarsest grid and about 0.16 m on the finest grid.

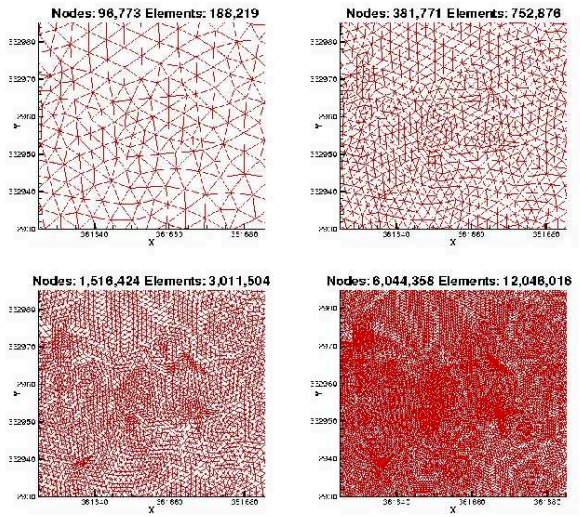

Figure 3. Detail of the most refined part of the Gironde grid for the original (0^{th} level) mesh and the first three levels of refinement.

TABLE I. CPU TIME (S) (IBM POWER7) TO COMPUTE FIVE LEVELS OF GRID REFINEMENT FROM AN ORIGINAL (0^{TH} LEVEL) MESH OF THE GIRONDE COMPOSED OF 0.2 MILLION TRIANGULAR ELEMENTS.

	0^{th} Level	**1^{st} Level**	**2^{nd} Level**
Nodes	96,733	381,771	1,156,424
Elements	188,219	752,876	3,011,504
Time (s)	—	3.26	16.10

	3^{rd} Level	**4^{th} Level**	**5^{th} Level**
Nodes	6,044,358	24,134,738	96,453,546
Elements	12,046,016	48,184,064	192,736,256
Time (s)	92.68	452.86	3090.97

An additional test was performed for the Malpasset computational domain. Table II summarises the times required to reach seven levels of grid refinement and a final mesh of 213,061,121 nodes and 425,984,000 elements. The tests demonstrate that meshes of more than 400 million elements can readily be generated using the proposed serial technique.

TABLE II. CPU TIME (S) (IBM POWER7) TO COMPUTE SEVEN LEVELS OF GRID REFINEMENT FROM AN ORIGINAL (0TH LEVEL) MESH OF THE MALPASSET FLOW DOMAIN COMPOSED OF 0.02 MILLION TRIANGULAR ELEMENTS.

	0th Level	**1st Level**	**2nd Level**
Nodes	13,541	53,081	210,161
Elements	26,000	104,000	416,000
Time (s)	—	0.27	1.17

	3rd Level	**4th Level**	**5th Level**
Nodes	836,321	3,336,641	13,329,281
Elements	1,664,000	6,656,000	26,624,000
Time (s)	5.48	27.37	127.76

	6th Level	**7th Level**
Nodes	53,282,561	213,061,121
Elements	106,496,000	425,984,000
Time (s)	598.44	3382.31

V. TESTS ON VARIOUS PARTITIONERS

The current serial version of PARTEL (version 6.0) uses METIS 4.0 as the partitioner. However, METIS 4.0 is known to be very memory consuming. Instead, four other partitioners have been tested including, METIS 5.0, ParMETIS 4.0, SCOTCH 5.1.12 and PT-SCOTCH 5.1.12. The Gironde Estuary and the Malpasset dam-break test cases have been run on a PC and an IBM POWER7 cluster. Unfortunately, PT-SCOTCH returns errors on the IBM cluster and these are currently being investigated by the main developer of the software.

A. *METIS/ParMETIS [8]*

METIS is a set of serial programs for partitioning graphs, partitioning finite-element meshes, and producing fill reducing orderings for sparse matrices. The algorithms implemented in METIS are based on the multilevel recursive-bisection, multilevel *k*–way, and multi-constraint partitioning schemes.

ParMETIS is an MPI-based parallel library that implements a variety of algorithms for partitioning unstructured graphs, meshes, and for computing fill-reducing orderings of sparse matrices. ParMETIS extends the functionality provided by METIS and includes routines that are especially suited for parallel AMR computations and large scale numerical simulations. The algorithms implemented in ParMETIS are based on the parallel multilevel *k*–way graph-partitioning, adaptive repartitioning, and parallel multi-constrained partitioning schemes developed at the Karypis Laboratory [8].

Version 5.0 of METIS and version 4.0 of ParMETIS are used in the present study. A huge effort has been made to optimise memory consumption in the latest releases of these codes. As both software codes are graph-based, a good strategy is to build a dual mesh of the existing grid and to transform it into a graph before partitioning since partitioning by elements has been shown to provide better quality results than partitioning by nodes. This option is provided in both METIS and ParMETIS.

B. *SCOTCH/PT-SCOTCH [9]*

SCOTCH is a project carried out by the Satanas team of the Laboratoire Bordelais de Recherche en Informatique (LaBRI) in France. It is part of the ScALApplix project of INRIA Bordeaux-Sud-Ouest. Its purpose is to apply graph theory, with a *divide and conquer* approach, to scientific computing problems such as graph and mesh partitioning, static mapping, and sparse matrix ordering, in application areas ranging from structural mechanics to operating systems or bio-chemistry.

The SCOTCH distribution is a set of programs and libraries which implement the static mapping and sparse matrix reordering algorithms developed within the SCOTCH project. PT-SCOTCH is the parallel version of SCOTCH.

Version 5.1.12 of SCOTCH has been used in this study. Unlike METIS and ParMETIS, SCOTCH/PT-SCOTCH do not provide tools for building dual meshes since they only deal with graphs. In contrast, a function has been built into METIS/ParMETIS to handle meshes directly. The function to perform the grid-to-dual-graph operation in METIS/ ParMETIS avoids using a less optimised Fortran function. SCOTCH and PT-SCOTCH require a strategy to define how to perform the partitioning. The default strategy is used here.

C. *The Gironde Estuary*

Each of the partitioners was used to process the 0th level Gironde mesh (~0.2 million elements) into 8 sub-domains; this initial study was performed using a local PC. Fig. 4 shows that each partitioner creates substantially different arrangements of sub-domains, as can be seen in the Garonne and Dordogne rivers.

A more elaborate series of tests were then performed on the IBM POWER7 cluster using METIS, SCOTCH and ParMETIS. These tests considered the time required to complete the partitioning process. Fig. 5 shows the time spent by the partitioners for various levels of grid refinement; all three partitioners exhibit a linear behaviour with the level of grid refinement, with ParMETIS being the fastest and SCOTCH the slowest. The number of partitions for each grid was selected so as to obtain an average of 3250 elements per sub-domain since previous tests [5] have demonstrated that TELEMAC-2D offers very good speed-up on a variety of computer platforms when reducing the number of elements per sub-domain from 6500 to 3250.

Fig. 6 compares the time spent by ParMETIS to partition the Gironde grids as a function of the refinement level, for different numbers of cores. In general, the 16-core simulations are the fastest.

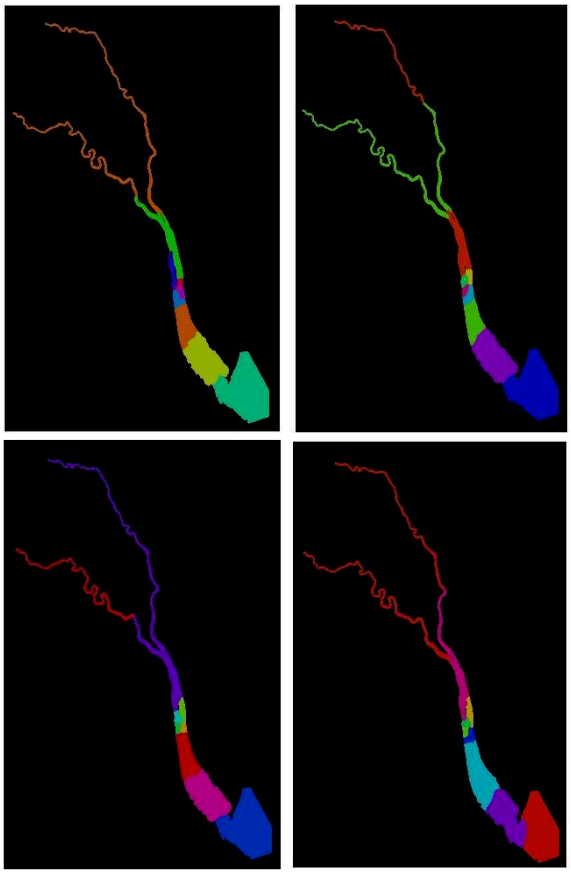

Figure 4. Sub-domains generated by each partitioner for the Gironde study; METIS (upper left), SCOTCH (upper right), ParMETIS (lower left), PT-SCOTCH (lower right).

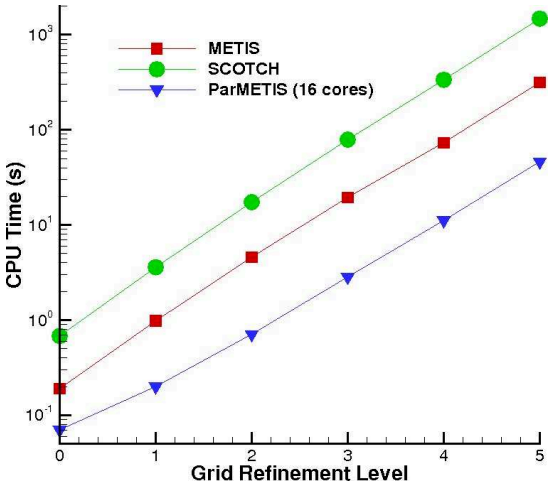

Figure 5. Partitioner times for each of the Gironde grid refinement levels.

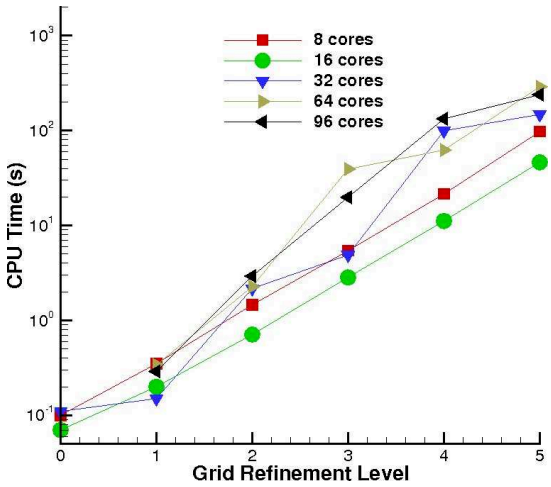

Figure 6. Time spent by ParMETIS as a function of the grid refinement level for the Gironde study.

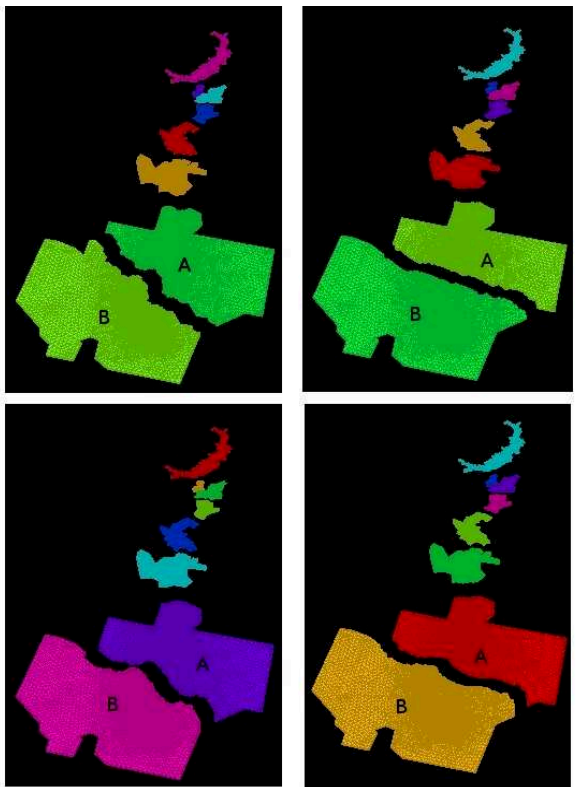

Figure 7. Sub-domains generated by each partitioner for the Malpasset study; METIS (upper left), SCOTCH (upper right), ParMETIS (lower left), PT-SCOTCH (lower right).

D. The Malpasset dam-break

The first test for the Malpasset dam-break study consisted of running each of the partitioners on a PC to create 8 sub-domains for the 1st level of refinement mesh (~0.1 million elements). Fig. 7 again shows that the various partitioners

produce slightly different results (see for example, the boundary between sub-domains, A and B, in Fig. 7). A second series of tests were then performed on the IBM POWER7 cluster using METIS, SCOTCH and ParMETIS to assess the time required to complete the partitioning process.

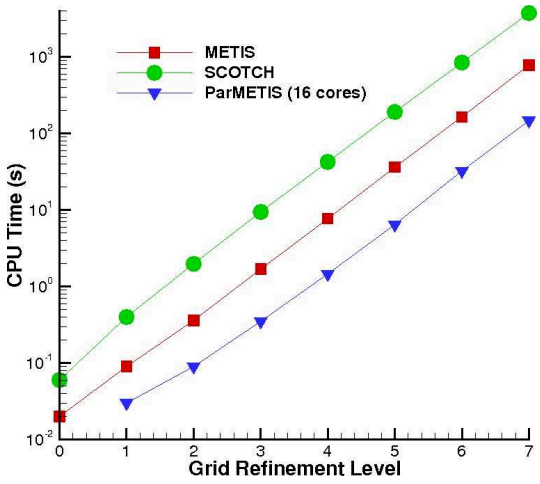

Figure 8. Partitioner times for each of the Malpasset dam-break refinement levels.

Fig. 8 shows the execution time of METIS, SCOTCH and ParMETIS for all seven levels of grid refinement for the Malpasset study. The same criterion of 3250 elements per sub-domain was used when selecting the number of partitions. ParMETIS was run on 16 cores except for the original mesh, where only 8 cores were used because the number of partitions to satisfy the 3250 element criterion is only 8, and the number of cores should always be lower than or equal to the number of sub-domains, otherwise the communication is too costly. The results show that ParMETIS is the fastest of the partitioners whereas SCOTCH is the slowest.

Fig. 9 compares the time spent by ParMETIS to partition the Malpasset grids as a function of the refinement level, for different numbers of cores. A speed-up was still observed for 96 cores but additional tests should be conducted to see if the partitioning time decreases with larger core counts. The 7[th] level of grid refinement requires at least 32 GB of RAM per ParMETIS task and therefore the tests on 32, 64 and 96 cores could not be performed because only 3 fat nodes are available on the POWER7 cluster, the fourth being a stand-alone node.

An additional test has shown that SCOTCH can partition the 7[th] level of refinement for the Malpasset study into 294,912 subdomains which represents the total number of cores on the IBM BlueGene/P located at Juelich [10].

VI. Parallel Pre-Processing (PARTEL_P)

To overcome the 2-10 million element grid limit of the serial pre-processor, PARTEL, a parallel version has been developed within the PRACE-1IP project [11], called PARTEL_P, which runs on NPROCS cores and partitions grids into NSUBS sub-domains. It should be noted that the current version of PARTEL_P does not support parallel IOs nor the method of characteristics for the pre-processing stage.

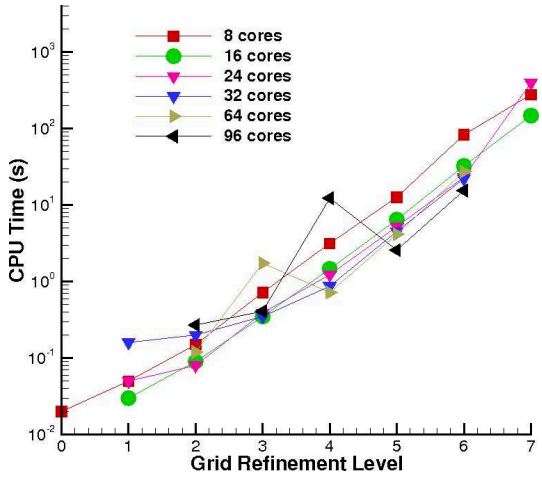

Figure 9. Time spent by ParMETIS as a function of the grid refinement level for the Malpasset study.

A. *Format of the files output by PARTEL*

Two files per sub-domain are output by PARTEL; a geometry file in SELAFIN format and a boundary file in ASCII format following the TELEMAC-2D standard. The geometry file contains a header, and the number of elements, nodes, physical boundaries and interfaces for a given sub-domain. It also contains the local connectivities of the nodes, the local-to-global node table and finally the coordinates and/or other quantities known at each node. The boundary file contains the information for the physical boundaries, the number of interfaces with other sub-domains, and the information required to handle the interfaces. Each physical boundary requires knowledge of the neighbouring nodes located in a different sub-domain. The treatment of the interfaces is more complex as the number of contiguous sub-domains has to be known, as well as their partition index. The interfaces also have to be sorted into ascending order to comply with the TELEMAC-2D standard.

B. *Description of PARTEL_P*

PARTEL_P is actually split into two parts; PARTEL_P_1 is used to generate NPROCS files to be read by PARTEL_P_2 so as to reduce memory consumption. These two programs will be merged in the future since both PARTEL_P_1 and PARTEL_P_2 are run on the same number of cores.

PARTEL_P_1 is used to distribute the information of NSUBS/NPROCS sub-domains over the NPROCS cores. The initial stage is mainly serial as no attempt has yet been made to improve the IO operations. The input parameters are read, i.e. the name of the geometry file, the name of the boundary file, NSUBS and the library used to partition the grid. Each processor reads the geometry and the boundary

files, and calls the subroutines VOISIN_PARTEL, ELEBD_PARTEL and FRONT2_PARTEL. The partitioning is performed using either METIS or SCOTCH on the master node (ParMETIS and PT-SCOTCH have yet to been tested), and its output is broadcast to the other cores. NSUBS/NPROCS sub-domains are gathered over the NPROCS cores in order to reduce the array sizes in PARTEL_P_2. Two files per core are written; one for the geometry, and the other for the boundary conditions with some additional information compared to a regular boundary condition file. PARTEL_P_1 transmits the information about the adjacent neighbouring nodes located in a different sub-domain but on a different core to PARTEL_P_2. Interfaces are not dealt with at this stage.

PARTEL_P_2 is run on NPROCS cores. It first reads the input parameters, i.e. the name of the original geometry file, the name of the original boundary file, NSUBS and NPROCS. Each core then reads the files output by PARTEL_P_1 which contains information for NSUBS/NPROCS sub-domains. The number of elements, nodes, physical boundaries, and interfaces per sub-domain are easily computed. This information, together with the knowledge of the sub-domain local connectivity and coordinates, helps build the NSUBS geometry files that are read by TELEMAC-2D. The local-to-global node table is also easily accessible.

The main issues arise when building the physical boundary information for the neighbouring nodes located in a different sub-domain and on a different core. The information relative to the interfaces on all NSUBS sub-domains also has to be computed.

The neighbouring nodes located on the same core but in a different sub-domain have to be identified. Working on a given core, all the physical boundaries are gathered in an array containing the global index. This array is sorted in ascending order and global indices that occur twice, or more, indicate that the corresponding nodes belong to several sub-domains. Their neighbours are easily identified and the array is sorted back to its original structure to comply with the TELEMAC-2D standard.

The interfaces are treated globally. A loop over all the NSUBS/NPROCS sub-domains allows the code to gather the interfaces of all the NPROCS cores before using MPI_Allgatherv to get their global index, as well as the index of the sub-domain they belong to. This array is sorted by global indices in ascending order. The number of consecutive occurrences, NINTERF, of a given global index indicates that the same interface belongs to NINTERF sub-domains and these partition indices have to be saved. To comply with the TELEMAC-2D standard, the information per interface has also to be sorted. All this information is then distributed in two stages, first onto the NPROCS, using an MPI_Scatterv command, and then to the NSUBS sub-domains.

The information relating to the physical boundaries and the interfaces is finally copied into the boundary files which are read by TELEMAC-2D.

C. *Timings for PARTEL_P_1 and PARTEL_P_2*

PARTEL_P_1 and PARTEL_P_2 have been run to pre-process the 5[th] level of refinement for the Gironde Estuary test case, and the output will be used in the next section to test TELEMAC-2D.

TABLE III. CPU TIME (S) (IBM POWER7) FOR PARTEL_P_1 AND PARTEL_P_2 TO PRE-PROCESS THE 5TH LEVEL OF REFINEMENT FOR THE GIRONDE ESTUARY STUDY, USING METIS AS THE PARTITIONER.

	PARTEL_P_1	**PARTEL_P_2**	**Total**
4096	2073	189	2262
8192	2421	284	2705
16384	2876	413	3289
32768	3941	748	4689

TABLE IV. CPU TIME (S) (IBM POWER7) FOR PARTEL_P_1 AND PARTEL_P_2 TO PRE-PROCESS THE 5TH LEVEL OF REFINEMENT FOR THE GIRONDE ESTUARY STUDY, USING SCOTCH AS THE PARTITIONER.

	PARTEL_P_1	**PARTEL_P_2**	**Total**
4096	2859	201	3060
8192	3021	262	3283
16384	3671	364	4035
32768	4878	587	5465

Tables III and IV indicate the total time spent by PARTEL_P to pre-process the grid into 4096, 8192, 16384 and 32768 sub-domains respectively, using METIS and SCOTCH as the partitioner. Overall, partitioning by METIS allows a faster pre-processing.

All PARTEL_P_1 simulations are faster when METIS rather than SCOTCH is used as the partitioner. However, PARTEL_P_2 is normally faster when SCOTCH is used. A more thorough study should be able to confirm whether this is due to the fact that the edge-cut should be smaller with SCOTCH, which has a direct impact on the global communications used in PARTEL_P_2.

VII. SCALING PERFORMANCE OF TELEMAC-2D

The 5[th] level of refinement for the Gironde Estuary test case (~200 million elements) has been used to evaluate the performance of TELEMAC-2D on 32,768 cores of Argonne's IBM Blue Gene/P [12]. PARTEL_P was used to perform the pre-processing with both METIS and SCOTCH being used as the partitioner.

The positive stream-wise implicit (PSI) advection scheme was selected since PARTEL_P does not yet support the method of characteristics. The scaling performance of TELEMAC-2D was evaluated using simulations of 60 seconds (1200 time steps). The CPU time is reported as the time for the executable to complete (T_{TOTAL}), as well as the time difference between the end and the beginning of the main program, *homere_telemac2d.f* (T_{SOLVER}).

Fig. 10 shows that T_{SOLVER} decreases linearly as a function of the number of cores, whether METIS or SCOTCH is used as the partitioner. Good performances are observed with about 6100 elements per core. A 65,536 sub-domain simulation would help assess the performance of TELEMAC-2D with about 3000 elements per core. However, T_{TOTAL} shows a different behaviour, with no real speed-up for the 32,768-core simulations. This might be explained by the time spent opening files, and the way the system manages the simulations.

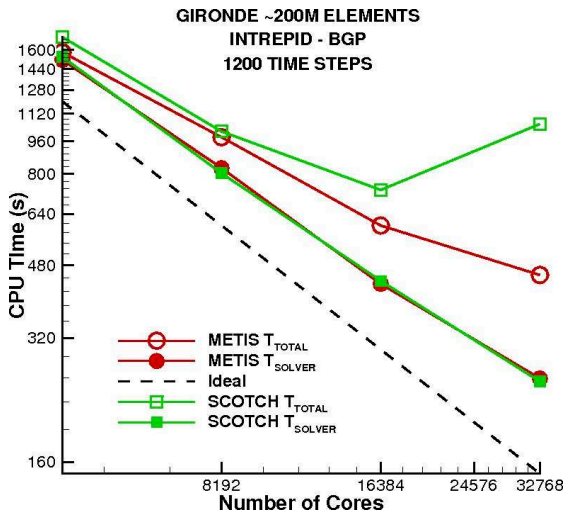

Figure 10. Scaling performance of TELEMAC-2D for the 5[th] level of grid refinement for the Gironde Estuary simulation.

VIII. Conclusions and Future Work

This paper has described the latest developments for running TELEMAC-2D on massively parallel computer architectures. An efficient serial global mesh refinement technique has been developed that can readily create meshes containing 400 to 500 million elements.

To overcome the mesh-size limitations of the existing serial pre-processor, PARTEL, a parallel version of the system tool has been developed called PARTEL_P. This new tool has been shown to be capable of pre-processing a 200-million element grid on up to 32,768 sub-domains and its output has successfully been used to demonstrate good scaling performance of TELEMAC-2D on Argonne National Laboratory's IBM Blue Gene/P.

The next stage is to be able to run TELEMAC-2D on more than 32,768 cores. This should be possible by implementing some form of memory optimisation, for example using compressed sparse row (CSR) format for some of the arrays. There is also a need to test the new pre-processing tool, PARTEL_P, with ParMETIS and PT-SCOTCH in order to complete the performance comparisons for different partitioners.

Acknowledgements

The authors acknowledge that the developments outlined in this paper have been achieved with the assistance of the high-performance computing resources (Tier-0) provided by PRACE on Jugene, based in Germany. This research also used resources of the Argonne Leadership Computing Facility at Argonne National Laboratory which is supported by the Office of Science of the U.S. Department of Energy under contract DE-AC02-06CH11357.

The authors would also like to thank the UK Engineering and Physical Sciences Research Council (EPSRC) for their support of Collaborative Computational Project 12 (CCP12) and the Distributed Computing Group at STFC Daresbury Laboratory. In addition, the authors would like to thank Nathalie Durand from EDF R&D for providing the data for the Gironde study.

References

[1] J.-M. Hervouet, Hydrodynamics of Free Surface Flows: Modelling with the Finite Element Method. Chichester: John Wiley & Sons, 2007.

[2] http://www.telemacsystem.com

[3] http://www.cse.scitech.ac.uk/disco/power7.shtml

[4] http://www-03.ibm.com/systems/deepcomputing/bluegene

[5] C. Moulinec, C. Denis, N. Durand, R.W. Barber, D.R. Emerson, X.J. Gu, E. Razafindrakoto, R. Issa and J.-M. Hervouet, "Coupling HPC and numerical validation: accurate and efficient simulation of large-scale hydrodynamic events," Proc. 2nd Int. Conf. on Parallel, Distributed, Grid and Cloud Computing for Engineering, B.H.V. Topping and P. Iványi (Eds.), Civil-Comp Press, Stirlingshire, Scotland, 2011. Paper 77.

[6] J.-M. Hervouet and A. Petitjean, "Malpasset dam-break revisited with two-dimensional computations," J. Hydraulic Research, vol. 37(6), pp. 777-788, 1999.

[7] J.-M. Hervouet, "A high resolution 2-D dam-break model using parallelization," Hydrological Processes, vol. 14(13), pp. 2211-2230, 2000.

[8] http://glaros.dtc.umn.edu/gkhome/views/metis

[9] http://www.labri.fr/perso/pelegrin/scotch/

[10] http://www2.fz-juelich.de/jsc/jugene

[11] http://www.prace-project.eu/

[12] http://www.alcf.anl.gov