Parallel Contact Detection Strategies for Cable and Membrane Structures

Jelle Muylle and Barry H.V. Topping

Department of Mechanical and Chemical Engineering
Heriot-Watt University, Edinburgh, United Kingdom

Abstract. The implementation of a parallel simulation procedure for cable and membrane structures is presented. The procedure includes contact detection and enforcement. Two different domain decompositions are simultaneously used. For the simulation a static partitioning is computed; for the contact detection a dynamically updated parallel RCB decomposition is used. An overview is also given of the strategy for the detailed geometric contact detection procedures. An example of a falling cloth is included to demonstrate.

1 Introduction

The research presented in this paper must be seen in the framework of computational simulation for analysis and design of structures consisting of cables and tensile membranes in a full three dimensional environment [1, 2].

The particular focus in this paper lies on the contact detection procedures required to deal with collapsing and crumpling membrane parts and with entanglement of cables. Considering the scale of the computational simulation which may arise for large structures it comes as no surprise that parallel and distributed processing have been adopted to keep the execution times within reason.

2 Simulation Components

The simulation process developed by the authors' research group for cable and membrane structures consists of several modular components: the simulation loop, the material model, the partitioning, contact detection, and contact handling. Not included in this list is the pre-processing stage of mesh generation, which for cable and membrane structures does involve a whole range of different complications, as discussed in other publications by the authors [3].

2.1 Simulation Loop

In this research, cable and membrane structures are simulated as a network of non-linear springs and masses whose motion is numerically integrated in an explicit transient dynamic analysis. The forces acting upon the nodes of the

network are: gravity; elastic forces due to the elongation of cables or stretching of membrane elements; induced forces over shared cable nodes or shared membrane element edges for simulation of bending stiffness [2]; contact forces when contacts are detected; and user defined external forces.

The equations governing the system are simply Newton's laws written out for each nodal mass m_i :

$$m_i \cdot \ddot{\mathbf{x}}_i = \sum \mathbf{f}_i \quad (1)$$

where $\ddot{\mathbf{x}}_i$ is the acceleration of node i and $\sum \mathbf{f}_i$ is the total of all force components having an influence on the motion of node i .

A variety of numerical integration schemes are available to derive the unknown position $\mathbf{x}_i^{t+\Delta t}$ of node i at time $t + \Delta t$ from the known positions $\mathbf{x}_i^t$ and forces $\sum \mathbf{f}_i^t$ at time t . An overview of the implemented schemes was previously published [1].

2.2 Material Model

A material model is required to calculate the nodal force components from the elongations at each timestep. Once again a variety of models is available, based on elastic [4] or elasto-plastic [5] assumptions or derived directly from energy functions [6]. The chosen implementation simulates cable elements as non-linear springs and decouples each triangular membrane element into a set of three non-linear springs [7].

The model was enhanced by the authors with a pseudo stiffness implementation that introduces the advantages of bending stiffness for a better simulation of natural fabrics without destroying the efficiency of an explicit time integration scheme [2].

2.3 Contact Detection

Contact detection in this framework involves all the necessary procedures to detect geometrical impossibilities caused by simulation of objects not being restrained in their movement by each other. The procedures that will be suggested must be particularly suited to detect the intersection of any two or more cable or membrane elements at the end of a time step or, in a stricter sense, the proximity of any two or more elements within a distance smaller than the thickness of the elements.

These two types of contact (where the first type is actually a special case of the more general second type) will be termed *intersection contact* and *proximity contact*. To avoid the exhaustive geometric checking of any two elements at each timestep a two stage procedure is used consisting of: a global checking phase and a local checking phase. The global checking phase eliminates the bulk of the combinations and selects a series of element pairs, that are to be examined further for contact. The local checking phase performs the detailed geometric check on the candidate element pairs, selected by the local checking phase. The global checking phase is sometimes called *pre-contact search* [8], or

location phase [9] or *neighbourhood identification phase* [10]. Several types of solution procedures can be applied to pair up the candidate elements: Solution procedures may be based on hierarchy building, as described in the work of Volino *et al.* [9, 11]. Solution procedures may also be based on spatial subdivision, where the simulation space is divided into subspaces such as buckets, slices or boxes. Finally solution procedures may be based on sorting algorithms where, by efficiently sorting the coordinate values of the nodes, the proximity of elements can be derived. The sorted coordinate lists must however be maintained when the nodes are continuously moving.

The global search procedure adopted in this research is based on the *point-in-box* algorithm [12]. The algorithm, as implemented by the authors [1] consists of the following steps:

- All nodal coordinates are sorted or resorted into three lists corresponding to their three spatial coordinates. Rank vectors, containing the dual information of the sorted lists are also kept.
- Around each element a bounding box is constructed, which is extended to a capturing box by a distance dx , related to the maximum distance any node can travel within one timestep. (See Figure 1.)
- With a binary search the extents of the capturing box in the sorted lists can be found. In each of the three lists a sublist will therefore be marked by the corresponding extent of the capturing box.
- Using the rank vectors the three sublists are contracted and all nodes which lie within the capturing box are therefore found.
- The elements that contain any of the captured nodes are said to be in proximity to the element around which the capturing box was constructed. All pairs are then passed on to the local search.

Fig. 1. Construction of the capturing box

The local search or detailed contact check, must then check geometrically if the selected contact pairs did make contact in the last timestep. By relaxing the definition of a contact situation, the detailed contact check can be simplified. For example, in a simulation which involves solid elements, the point-in-box

search described above could be used to report the proximity between nodes and the solid elements. The local check could then verify if the node entered into the volume of the solid element. With this kind of *nodal intersection* as basis for contact detection most of the contacts in a simulation with solid elements could be identified. But this is of no good to a simulation including volume-less triangular membrane elements.

If the local search is made to detect the passing of a node through a triangular element, it becomes of more use to membrane simulations. This checking is geometrically more intensive as both triangle and node are moving during the timestep. It involves the solution of a cubic equation to resolve the intersection between the trajectory of the node and the plane through the transforming triangle [12] and a series of geometrical tests to determine if the intersection lies within the triangle.

The great disadvantage of choosing nodal intersection as basis for contact detection is that an entire class of contacts is ignored. Edge contact, such as the one shown in Figure 2 is not detected. Previously published research by the authors [1] presented three methods to allow detection of edge contact. The method that will be explained in Section 4 is a new look on the problem that will not only detect edge contact but will also provide support for contacts involving cable elements and support for the detection of proximity contact situations rather than intersection contact.

Fig. 2. Edge contact, not detected by nodal intersection

2.4 Contact Handling

Contact handling is a general term for all actions to be taken as a consequence of detected contact situations. It may include:

- correcting geometric impossibilities such as intersections (corrective actions),
- enforcing a constraint placed on the movement of a node,
- or modification of the trajectory of a node that is on a course to intersection by changing its velocity or acceleration components (preventive action).

Two types of contact handling are common: the use of contact forces, or kinematic correction procedures. When a contact is detected and contact forces are introduced, it is hoped that within a few timesteps, due to the action of the forces

the contact situation will be rectified. With the kinematic correction method a direct change to the position, velocity or acceleration components of a contacting node is made to rectify the situation within the same timestep. Both methods were implemented. However, to enable realistic simulation in the future, the contact handling method selected must be able to guarantee that any geometric intersections of the membrane or cable elements must be avoided at all cost. Geometric intersections could cause serious problems if a volume mesh is present around the cable or membrane model, for simulation of fluid flow. Avoiding geometric intersections does however not exclude the use of contact forces. If a proximity contact check is used whereby the proximity distance is taken as n times the maximum distance any node can travel within one timestep, then the contact handling procedure has n timesteps to alter the course of a node from when it is detected as violating the proximity distance. A careful choice for the model behind the calculation of the direction and size of the contact force is however required.

A problem faced when implementing the kinematic correction method is how to deal with multiple contacts involving the same nodes or elements. Correcting the position of a node that is involved in e.g. two contacts according to the constraints posed by the first contact situation may cause under or over correction of the second contact situation. Solutions were offered by Volino *et al.* [11].

3 Implications for Parallel Implementations

The simulation of the cable and membrane structures in motion has been successfully parallelized in a message passing context (MPI). New domain decomposition strategies, based on graph partitioning were developed to take into account the specific nature of cable and membrane structures [13]. A decomposition is therefore used which is optimized for the material simulation steps, in particular the calculation of the nodal forces due to the material behaviour. This decomposition only relies on the topology of the mesh and can be determined beforehand. Elements are assigned to each processor in order to achieve load balance of the computational efforts of calculating the forces and integrating the nodal movements. And at the same time the decomposition minimizes the number of subdomain interfaces to reduce the amount of communication due to nodal force components for nodes shared by more than one partition.

The decomposition scheme used for the material simulation steps is however not ideal for parallelizing the contact detection procedure. If the contact detection is carried out by a global-local scheme as detailed above, then in the parallel equivalent of global search procedure, each of the processors should do some of the global searching. If the point-in-box method is used, ideally each subdomain should consist of those elements and nodes which are geometrically close to each other rather than topologically. Such a decomposition exists: *recursive coordinate bisection* (RCB). Moreover RCB has the advantage that all subdomains will be bound by (almost) non-overlapping axis aligned bounding boxes. The parallel version of the contact procedure will therefore be identical to the serial version

carried out within each subdomain. The details of obtaining and maintaining the RCB decomposition will be described in Section 5. The scheme on which the authors have based their implementation was first suggested by Sandia National Labs implementation of the PRONTO3D code [10].

The implications of working with two distinct decompositions for a single mesh, will be left to the conclusions in Section 7.1. However it must be mentioned that alternatives are available. Parallel tree searching algorithms do exist [14] but the authors are at present unable to comment on their success or applicability.

Once contacts have been detected by a parallel contact detection algorithm, the contact pairs can be communicated to the processors that own the respective elements for the material simulation. The contact forces or kinematic correction procedure can be carried out in this decomposition, requiring communication if necessary with other processors should contact be detected between elements belonging to different decompositions.

4 Geometric Contact Detection

The geometric contact detection procedure is based on a library of geometric test-functions which were implemented with adaptive precision arithmetic. Functions were created for measuring distances, and calculating distance between points, lines and planes. Thanks to a consistent system of tolerances and a limited set of internally consistent predicates [15], a stable geometric platform was developed.

Cable elements are encapsulated in a cylinder of radius equal to the proximity distance with two hemispherical caps over the end nodes of the element. Triangular elements are encapsulated between two planes a distance equal to the proximity distance away from the element, in addition to three cylindrical and three spherical caps. The encapsulated forms are compared for intersections and a system of geometric tests classifies the pairs into a number of categories. Figure 3 shows a series of intersection options for the combination of a cable element with a triangular element.

Only six possible configurations are shown in the figure. This excludes all the cases where the cable element is found to be parallel with the triangular element. The classification at the same time allows the contact handling functions to select the correct nodes and interpolation factors for the calculation of the contact forces or kinematic corrections.

5 Parallel Recursive Coordinate Bisection

Recursive coordinate bisection (RCB) (sometimes called orthogonal recursive bisection (URB)) was first introduced as a static partitioning algorithm [16] for unstructured meshes. Since then it has long been superseded by much better static partitioning algorithms, but as a dynamic partitioning/repartitioning algorithm it is still widely used.

RCB is particularly suited for parallelization of contact detection as it is a geometric partitioning method rather than a topological one. The resulting

Fig. 3. Geometric Contact Cases for Cable-Membrane Contact

partitions are all bound by disjoint orthogonal bounding boxes. As a repartitioning scheme it shows extremely good performance because small changes in the node coordinates produce only small changes in the partitioning, and much of the sorting and searching in the algorithm can be recycled or restarted from near-optimal values.

The RCB algorithm divides the simulation space up in Cartesian subspaces attached to each processor with roughly a balanced number of nodes. It does so by cutting a partition into two sub-partitions with an equal or almost equal number of nodes. The sub-partitions are bisected recursively until the required number of partitions is reached. In its most basic form only 2^n partitions can be created. However by appropriately choosing the ratio of nodes on either side of the cutting plane, any number of subdomains can be obtained. The basic cutting step, which is recursively applied, consists of three parts:

1. Selection of the coordinate (x , y or z) upon which the nodes will be judged for bisecting.
2. Choosing the cutting point which splits the population of nodes into the appropriate ratio ($\frac{1}{2}$ in case of 2^n partitions).
3. Redistribution of the nodes into either one of the subpartitions, depending on which side of the cutting point they fall.

A parallel version of the RCB algorithm (pRCB) was implemented by the authors, based on the ideas of Nakhimovski [17]. As such it becomes a repartitioning algorithm where all processors already own a series of nodes and cooperate in a parallel fashion to determine the RCB of the global set of nodes. The authors' implementation postpones the actual redistribution of locally owned nodes to the end and performs the bisection on destination tags for each node. Once these have been determined the redistribution can be organized using the

unstructured communication patterns, suggested in references [10, 18] to avoid *all-to-all* communication.

6 Case Study: A Falling Cloth

The example in Figure 4 shows the simulation of a falling cloth suspended from one corner on a four processor machine. The cloth is represented by a unstructured triangular mesh containing 682 nodes and 1262 elements. Image a) of the figure displays the domain decomposition used for the material simulation. A elastic material model representing cotton ($\rho = 409.8\text{kg/m}^3$, thickness= 0.47mm, $E = 10^7\text{N/m}^2$, $\nu = 0.2$) was used with a timestep of 1msec and a central difference integration scheme.

Fig. 4. Case Study: A falling cloth. a) The domain decomposition used for the material simulation. b)-f) The domain decomposition used for contact detection, dynamically updated during the fall of the cloth

It was decided to update the contact detection decomposition only every 50 timesteps. This number was found as an engineering trade-off between the cost of recomputing the RCB decomposition and the extra communication obtained due to overlapping imperfect decompositions. Further efforts are being made to automate the finding of this threshold value.

7 Concluding Notes

7.1 The Implications of Using Two Decompositions

Using a different domain decomposition for the material simulation and the contact detection has quite a number of implications:

- At the end of the calculation of the new position of each node in the material simulation communication has to take place between the processor that owns the node in the material simulation and the processor that owns the node for the contact detection. The communication involves the new coordinates and velocities of the node as both are required for the contact detection.
- As suggested earlier the contact handling is carried out in the material simulation decomposition and not in the contact detection decomposition. All reported contacts therefore will have to be communicated to the material simulation decomposition. The reason for this is that computing the contact response involves far more variables, such as mass and material parameters, which are not known in the contact detection decomposition. Making them all available to the contact detection decomposition throughout the simulation would require far too much extra communication.
- Complications will also occur because of the differences between a decomposition of nodes (with halo elements) and a decomposition of elements (with halo nodes). The halo elements or nodes will always cause unavoidable communication.

It is however suggested that the communication disadvantages of using two decompositions are a reasonable price to pay in order to obtain overall load balance of the entire simulation process [10, 19].

7.2 Looking towards the Future

As a result of the computational intensity of the geometric procedures in the detailed contact detection phase, the authors' current implementation is still relatively slow compared to other less complicated schemes. The guarantee however that all element intersections can be avoided is of paramount importance when considering realistic simulations.

Research considering the optimization of the communication patterns between the two decompositions is still in progress. The implementation of a robust fully parallel simulation of cable and membrane structures with contact detection and handling is gradually taking shape.

References

- [1] Muylle, J., Topping, B.H.V.: Contact detection and enforcement techniques for the simulation of membrane structures in motion. In Topping, B.H.V., ed.: *Computational Mechanics: Techniques and Developments*, Edinburgh, U.K., Civil-Comp Press (2000) 243–256

- [2] Muylle, J., Topping, B.H.V.: A cable and mebrane structure pseudo stiffness implementation. In Topping, B.H.V., ed.: *Proceedings of the Eighth International Conference on Civil and Structural Engineering Computing*, Stirling, UK, Civil-Comp Press (2001) paper 24.
- [3] Topping, B.H.V., Muylle, J., Putanowicz, R., Iványi, P., Cheng, B.: *Finite Element Mesh Generation*. Saxe-Coburg Publications, Stirling, UK (2002)
- [4] Volino, P., Courchesne, M., Magnenat Thalmann, N.: Developing simulation techniques for an interactive clothing system. In: *Proceedings VSM 97*, IEEE Computer Society (1997) 109–118
- [5] Wu, Y., Thalmann, D., Thalmann, N.M.: Deformable surfaces using physically-based particle systems. In: *Proceedings Computer Graphics International 95*, Academic Press (1995) 205–216
- [6] Baraff, D., Witkin, A.: Large steps in cloth simulation. In: *Proceedings Siggraph 1998*, New York, ACM Press (1998) 43–54
- [7] Topping, B.H.V., Iványi, P.: *Computer Aided Design of Cable-Membrane Structures*. Saxe-Coburg Publications, Stirling, UK (2002)
- [8] Zhong, Z.H.: *Finite Element Procedures for Contact-Impact Problems*. Oxford University Press, Oxford (1993)
- [9] Volino, P., Magnenat Thalmann, N.: Collision and self-collision detection: Efficient and robust solutions for highly deformable surfaces. In Terzopoulos, D., Thalmann, D., eds.: *Computer Animation and Simulation '95*, Springer-Verlag (1995) 55–65
- [10] Attaway, S.W., Hendrickson, B.A.e.a.: A parallel contact detection algorithm for transient solid dynamics simulations using pronto3d. *Computational Mechanics* **22** (1998) 143–159
- [11] Volino, P., Magnenat Thalmann, N.: Accurate collision response on polygonal meshes. In: *Proceedings Computer Animation Conference*, Philadelphia, IEEE Computer Society (2000)
- [12] Heinstein, M.W., Attaway, S.W., Swegle, J.W., Mello, F.J.: A general-purpose contact detection algorithm for nonlinear structural analysis codes. Technical Report SAND92-2141, Sandia National Laboratories, Albuquerque, NM (1993)
- [13] Iványi, P., Topping, B., Muylle, J.: Towards a CAD design of cable-membrane structures on parallel platforms. In Bathe, K., ed.: *Computational Fluid and Solid Mechanics*, Elsevier Science Ltd. (2001) 652–654
- [14] Al-furaih, I., Aluru, S., Goil, S., Ranka, S.: Parallel construction of multidimensional binary search trees. *IEEE Transactions on Parallel and Distributed Systems* **11** (2000) 136–148
- [15] Shewchuk, J.R.: *Delaunay Refinement Mesh Generation*. PhD thesis, Carnegie Mellon University, Pittsburgh, PA (1997)
- [16] Berger, M.J., Bokhari, S.H.: A partitioning strategy for nonuniform problems on multiprocessors. *IEEE Transactions on Computers* **36** (1987) 570–580
- [17] Nakhimovski, I.: Bucket-based modification of the parallel recursive coordinate bisection algorithm. *Linköping Electronic Articles in Computer and Information Science* **2** (1997)
- [18] Plimpton, S., Attaway, S., Hendrickson, B., Swegle, J., Vaughan, C., Gardner, D.: Parallel transient dynamics simulations: Algorithms for contact detection and smoothed particle hydrodynamics. *Journal of Parallel and Distributed Computing* **50** (1998) 104–122
- [19] Hendrickson, B., Devine, K.: Dynamic load balancing in computational mechanics. *Computer Methods in Applied Mechanics and Engineering* **184** (2000) 485–500