A Requirements

A.1 Functional Requirements

1. FE integration: System will be able to interface with a third party finite element application

    1.1. The finite element applications solver must be able to solve a mesh based on its model configuration.

    1.2. The finite element applications solver must be able to execute a model programmatically

    1.3. The finite element applications solver must be able to output stress data at different points on the mesh.

    1.4. The finite element application will provide a graphical representation of the model.

    1.5. It will be possible to manipulate the model that the finite element application uses programmatically.

    1.6. It should be possible to manipulate the model that the finite element application uses from within its graphical user interface.

2. Mesh refinement: System will be able to perform different kinds of finite element mesh refinement

    2.1. The system will be able to refine a finite element mesh using a stress based refinement method.

    2.2. The system will be able to refine a finite element mesh using a non-stress based refinement method.

    2.3. A non stress based refinement method will adapt the mesh using background information about mesh design which has been previously trained.

    2.4. The system will be able to combine the two methods to produce a coherent mesh which the FE application is able to successfully solve in order to obtain results for stress and displacement.

    2.5. The system will be able to combine both methods to varying degrees that will be performed automatically by the system without direct user intervention.

    2.6. The system will re mesh using both stress and non-stress based refinement using quadrilateral elements.

    2.7. System will adapt weighting associated with each method based upon the metrics computed for the mesh in the systems previous iteration.

3. Quality assessment: System will provide the operator with results about the quality of meshes based on metrics obtained from research.

    3.1. An assessment will be conducted automatically for every mesh iteration that occurs.

    3.2. System will assess quality based on a variety of metrics to ensure overall robustness of measurement.

    3.3. The metrics will be computed for both individual element within the model and for the entire mesh.

A.2 Non-Functional Requirements

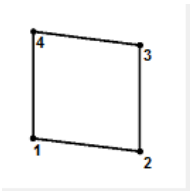
Design: The system architecture will be developed using the object oriented design principals of SOLID to allow for clear interfaces between the different functional components. Functional programming practices will be adopted through use of the .NET Language Integrated Query or LINQ framework. This will help to simplify the code and improve reliability by removing unnecessary state from the program. Where functions and classes are written their length will be kept to a minimum to reduce complexity and allow for reuse wherever possible.

Documentation: The system will be comprehensively documented at both a code level and at an architecture level. At a code level C# doc comments will be written to provide a comprehensive summary of each function. This will allow the tool Doxygen [25] to generate a full set of developer documentation upon completion of the software implementation which will be included as an appendix. Ensuring that the majority of information is present within doc comments will also help to promote a reduction of loose comments within the code and hence function size.

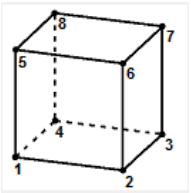
General applicability: In order to demonstrate that hybrid methods are a feasible means of approaching meshing problems the resulting software should be able to successfully execute on a range of models with varying geometry. The range of geometries should be representative of typical structural variation encountered by engineers.

B Element Types within LISA

Here are shown the the visual specifications LISA provides for the ordering and layout of nodes for defining each type of element supported. Each of these element types can be classified using base classes which implement the IElement interface.

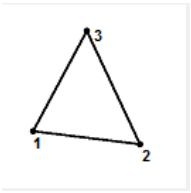


(a) quad4 element

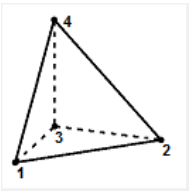


(b) hex8 element

Figure 21: Square based elements

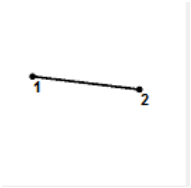


(a) tri3 element

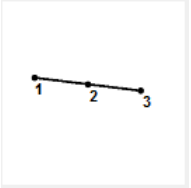


(b) tet4 element

Figure 22: Triangular based elements



(a) line2 element



(b) line3 element

Figure 23: Line based elements

C Unit Testing

Below can be seen the test explorer interface within Visual studio, tests have also been included in the submission with the rest of the code.

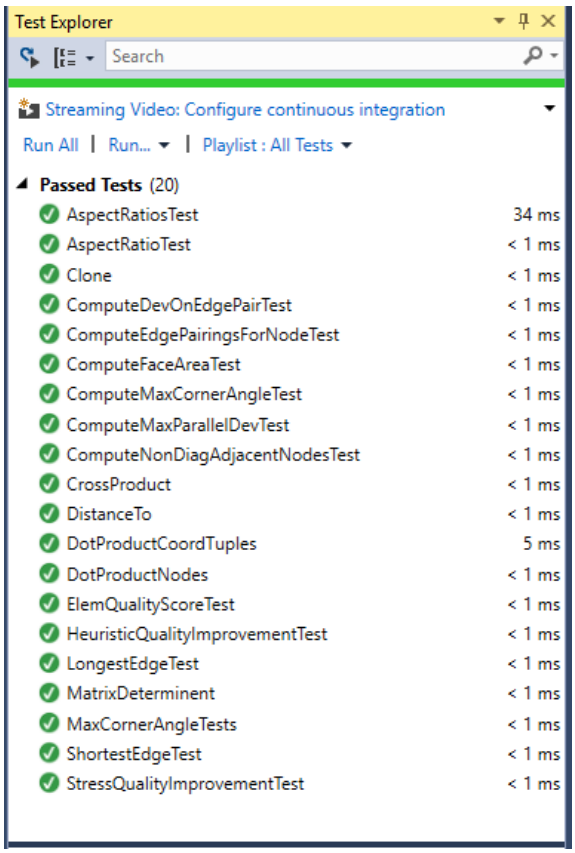


Figure 24: Visual Studio window showing the small suite of twenty tests for validating the core functionalyu of the system.

D Edge Definition Categories

This appendix shows the different values that can be assigned to each of the different edge properties used by the program. Each of these properties has some relevance to the rules and is used to determine how much meshing should occur along a particular edge.

Edge Type

- important long
- important
- important short
- not important
- circuit
- half circuit
- quarter circuit
- short for a hole
- long for a hole
- circuit hole
- half circuit hole
- quarter circuit hole

Boundary Type

- free
- fixed on one side
- fixed on two sides
- fixed completely

Load Type

- no loading
- one side loaded
- two sides loaded
- continuous loading

E Input and Output Files

Below can be seen the format of the input files for the system, a LISA .liml and a .json edge definition file. Examples of both these files have also been included within the submission under the "Experiments" folder.

```
<liml8>
  <analysis type="S3O" />
  <node nid="1" x="0" y="0" z="0" />
  <node nid="2" x="0" y="0" z="25" />
  <node nid="3" x="25" y="0" z="25" />
  <node nid="4" x="25" y="0" z="0" />
  <node nid="5" x="0" y="10" z="0" />
  <node nid="6" x="0" y="10" z="25" />
  <node nid="7" x="25" y="10" z="25" />
  <node nid="8" x="25" y="10" z="0" />
  <node nid="9" x="10" y="30" z="5" />
  <node nid="10" x="10" y="30" z="20" />
  <node nid="11" x="20" y="30" z="5" />
  <node nid="12" x="20" y="30" z="20" />
  <node nid="13" x="10" y="80" z="5" />
  <node nid="14" x="10" y="80" z="20" />
  <elset name="Default" color="-6710887" material="Material">
    <elem eid="1" shape="quad4" nodes="1 4 3 2" />
    <elem eid="2" shape="quad4" nodes="6 7 3 2" />
    <elem eid="3" shape="quad4" nodes="5 6 2 1" />
    <elem eid="4" shape="quad4" nodes="8 5 1 4" />
    <elem eid="5" shape="quad4" nodes="7 8 4 3" />
    <elem eid="6" shape="quad4" nodes="9 10 6 5" />
    <elem eid="7" shape="quad4" nodes="10 12 7 6" />
    <elem eid="8" shape="quad4" nodes="11 9 5 8" />
    <elem eid="9" shape="quad4" nodes="12 11 8 7" />
  </elset>
  <fix selection="BridgeBase" />
  <force selection="Unnamed">
    <x>-1000</x>
    <y>0</y>
    <z>0</z>
  </force>
  <mat mid="1" name="Material">
    <geometric type="Plate" thickness="3" planestrain="0" />
    <mechanical type="Isotropic" youngsmodulus="200000000000" poissonratio="0.3" />
  </mat>
  <faceselection name="BridgeBase">
    <face eid="1" faceid="5" />
    <face eid="27" faceid="5" />
    <face eid="14" faceid="5" />
    <face eid="40" faceid="5" />
  </faceselection>
  <faceselection name="Forces">
    <face eid="110" faceid="6" />
    <face eid="111" faceid="6" />
    <face eid="109" faceid="6" />
    <face eid="26" faceid="6" />
    <face eid="23" faceid="6" />
    <face eid="67" faceid="6" />
    <face eid="57" faceid="6" />
    <face eid="146" faceid="6" />
  </faceselection>
    <analysis type="S3O" />
    <node nid="1" x="0" y="0" z="0" />
    <node nid="2" x="0" y="0" z="2.5" />
    <node nid="3" x="2.5" y="0" z="2.5" />
    <node nid="4" x="2.5" y="0" z="0" />
    <node nid="5" x="0" y="1" z="0" />
    <node nid="6" x="0" y="1" z="2.5" />
    <elset name="Default" color="-6710887" material="Material">
      <elem eid="1" shape="quad4" nodes="1 4 3 2" />
      <elem eid="2" shape="quad4" nodes="6 7 3 2" />
      <elem eid="3" shape="quad4" nodes="5 6 2 1" />
      <elem eid="4" shape="quad4" nodes="8 5 1 4" />
```

Figure 25: Cut down .liml file to show general content which largely defined the schema for the systems data model

```
{
    "Edges": [{
        "Id": 1,
        "edgeType": "circuit",
        "loadType": "oneSideLoaded",
        "boundaryType": "free",
        "nodePath": [42, 198, 197, 41]
    },
    {
        "Id": 2,
        "edgeType": "importantLong",
        "loadType": "oneSideLoaded",
        "boundaryType": "fixedTwoSides",
        "nodePath": [70, 62, 61, 46, 45, 67]
    },
    {
        "Id": 3,
        "edgeType": "importantShort",
        "loadType": "oneSideLoaded",
        "boundaryType": "fixedTwoSides",
        "nodePath": [41, 197]
    },
    {
        "Id": 4,
        "edgeType": "notImportant",
        "loadType": "notLoaded",
        "boundaryType": "fixedCompletely",
        "nodePath": [197, 207]
    },
    {
        "Id": 5,
        "edgeType": "circuit",
        "loadType": "continuiousLoading",
        "boundaryType": "fixedOneSide",
        "nodePath": [88, 142, 148, 147, 140, 141, 135, 85]
    }
    ]
}
```


Figure 26: A json file containing the edges of interest specified by an engineer, this is parsed and the rules are applied to determine the models meshing based on the input

F Project Layout in Solution Explorer

Below show the Visual Studio Solution Explorer which provides a general idea of the layout of the project with namespace hierarchies from within an IDE.

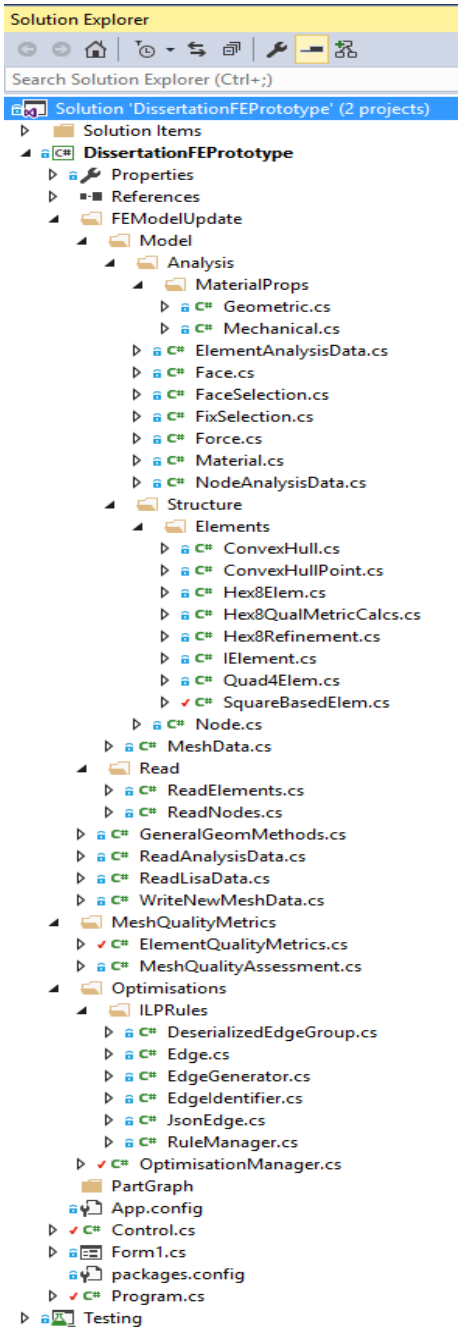


Figure 27: The metrics calculated by visual studio for all high level modules in the system

G Doxygen Documentation

Here can be seen two screenshots of the documentation that was automatically generated using the tool , the full documentation has been included as part of the supporting material for the dissertation.

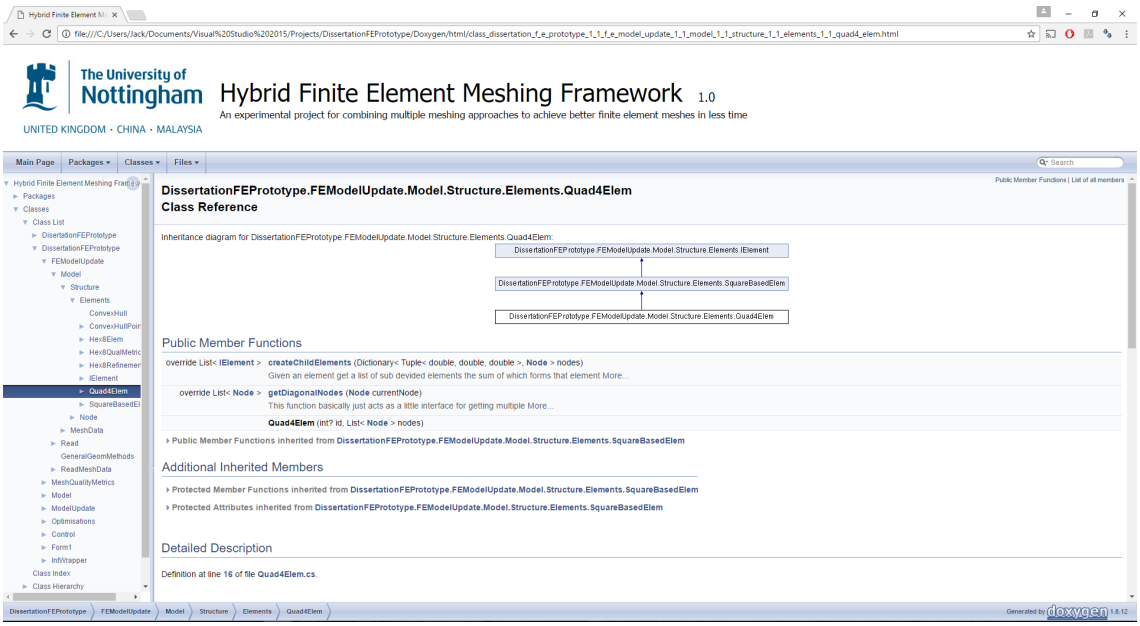


Figure 28: The manual page for the Quad4 element class with the class hierarchy and specific public methods

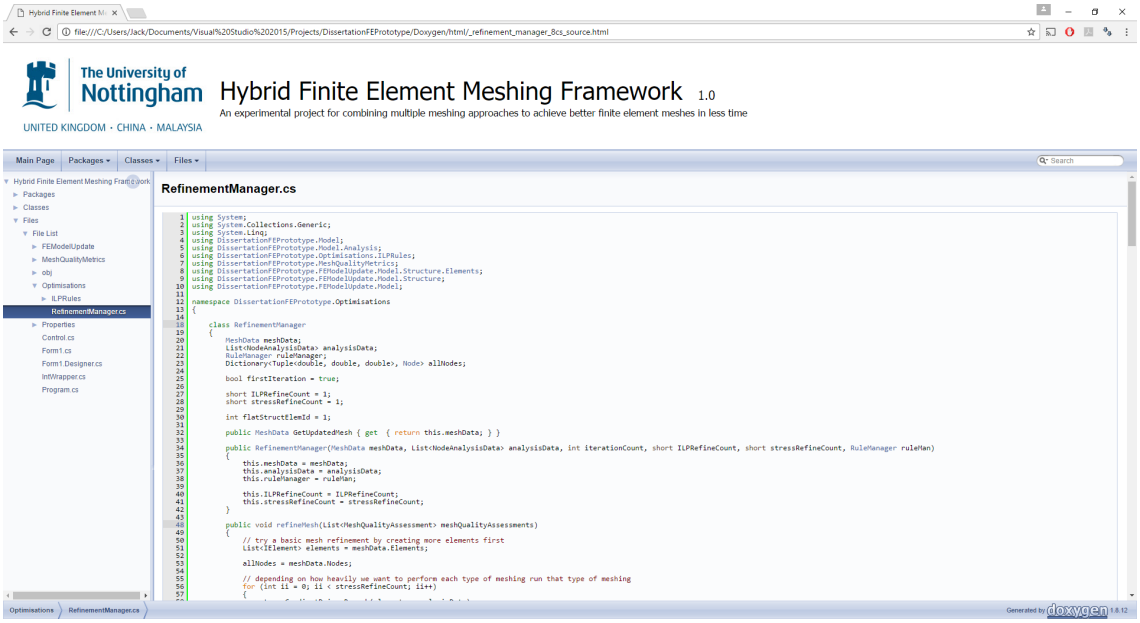


Figure 29: Code for the RefinementManager class viewed within the Doxygen UI

H Software Quality Metrics

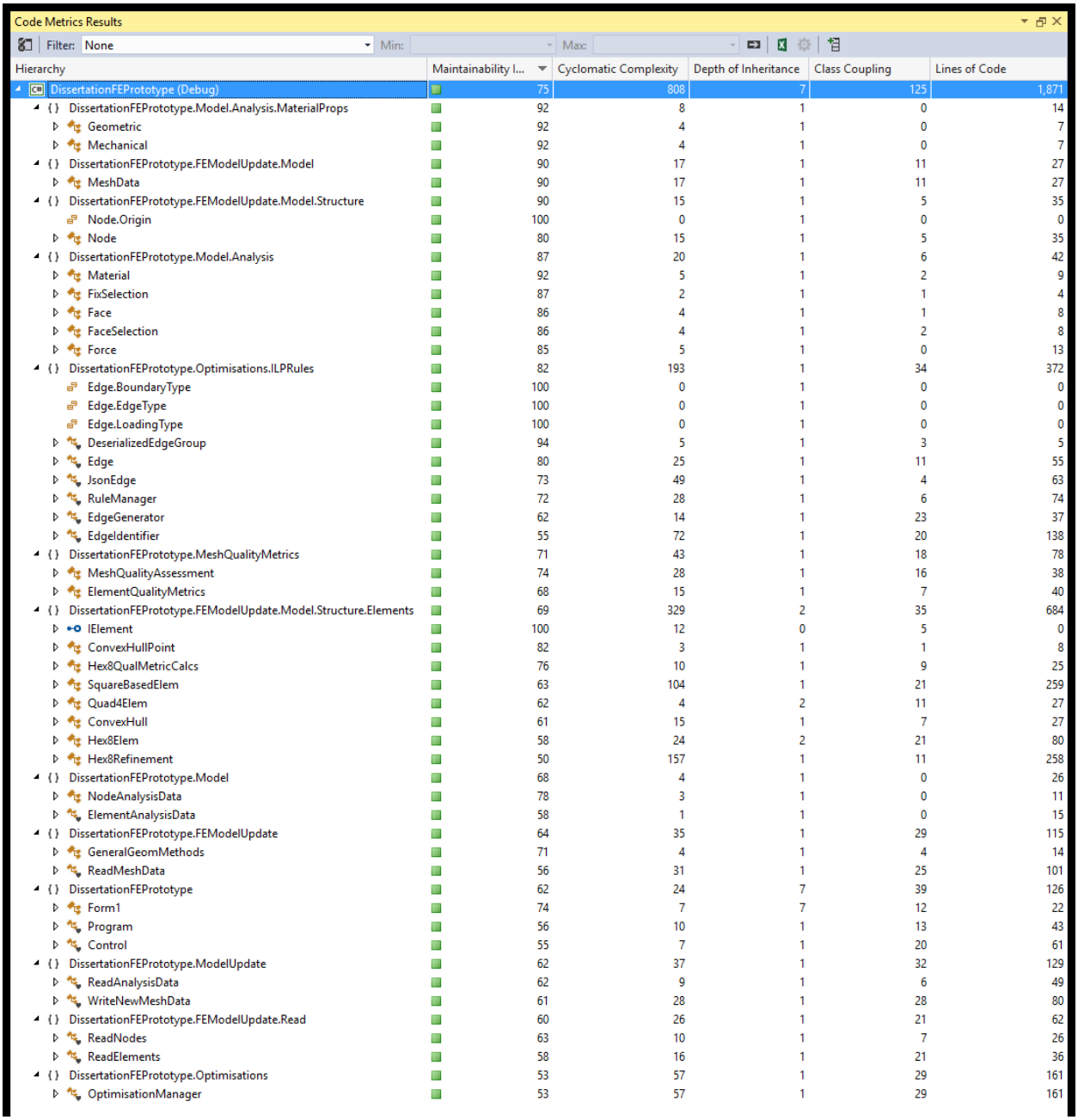
This appendix shows some of the code metric results calculated by visual studio for the final software implementation.

Code Metrics Results

Filter: None Min: Max:

Hierarchy	Maintainability Index	Cyclomatic Complexity	Depth of Inheritance	Class Coupling	Lines of Code
DissertationFEPrototype (Debug)	75	808	7	125	1,871
{ } DissertationFEPrototype	62	24	7	39	126
{ } DissertationFEPrototype.FEModelUpdate	64	35	1	29	115
{ } DissertationFEPrototype.FEModelUpdate.Model	90	17	1	11	27
{ } DissertationFEPrototype.FEModelUpdate.Model.Structure	90	15	1	5	35
{ } DissertationFEPrototype.FEModelUpdate.Model.Structure.Elements	69	329	2	35	684
{ } DissertationFEPrototype.FEModelUpdate.Read	60	26	1	21	62
{ } DissertationFEPrototype.MeshQualityMetrics	71	43	1	18	78
{ } DissertationFEPrototype.Model	68	4	1	0	26
{ } DissertationFEPrototype.Model.Analysis	87	20	1	6	42
{ } DissertationFEPrototype.Model.Analysis.MaterialProps	92	8	1	0	14
{ } DissertationFEPrototype.ModelUpdate	62	37	1	32	129
{ } DissertationFEPrototype.Optimisations	53	57	1	29	161
{ } DissertationFEPrototype.Optimisations.ILPRules	82	193	1	34	372

Figure 30: The metrics calculated by visual studio for all high level modules in the system



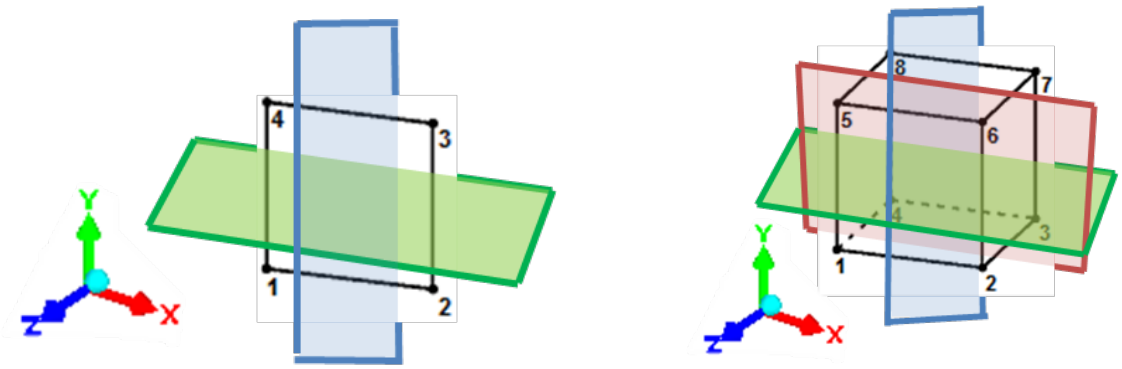
Hierarchy	Maintainability I...	Cyclomatic Complexity	Depth of Inheritance	Class Coupling	Lines of Code
DissertationFEPrototype (Debug)	75	808	7	125	1,871
DissertationFEPrototype.Model.Analysis.MaterialProps	92	8	1	0	14
Geometric	92	4	1	0	7
Mechanical	92	4	1	0	7
DissertationFEPrototype.FEModelUpdate.Model	90	17	1	11	27
MeshData	90	17	1	11	27
DissertationFEPrototype.FEModelUpdate.Model.Structure	90	15	1	5	35
Node.Origin	100	0	1	0	0
Node	80	15	1	5	35
DissertationFEPrototype.Model.Analysis	87	20	1	6	42
Material	92	5	1	2	9
FixSelection	87	2	1	1	4
Face	86	4	1	1	8
FaceSelection	86	4	1	2	8
Force	85	5	1	0	13
DissertationFEPrototype.Optimisations.ILPRules	82	193	1	34	372
Edge.BoundaryType	100	0	1	0	0
Edge.EdgeType	100	0	1	0	0
Edge.LoadingType	100	0	1	0	0
DeserializedEdgeGroup	94	5	1	3	5
Edge	80	25	1	11	55
JsonEdge	73	49	1	4	63
RuleManager	72	28	1	6	74
EdgeGenerator	62	14	1	23	37
EdgeIdentifier	55	72	1	20	138
DissertationFEPrototype.MeshQualityMetrics	71	43	1	18	78
MeshQualityAssessment	74	28	1	16	38
ElementQualityMetrics	68	15	1	7	40
DissertationFEPrototype.FEModelUpdate.Model.Structure.Elements	69	329	2	35	684
IElement	100	12	0	5	0
ConvexHullPoint	82	3	1	1	8
Hex8QualMetricCalcs	76	10	1	9	25
SquareBasedElem	63	104	1	21	259
Quad4Elem	62	4	2	11	27
ConvexHull	61	15	1	7	27
Hex8Elem	58	24	2	21	80
Hex8Refinement	50	157	1	11	258
DissertationFEPrototype.Model	68	4	1	0	26
NodeAnalysisData	78	3	1	0	11
ElementAnalysisData	58	1	1	0	15
DissertationFEPrototype.FEModelUpdate	64	35	1	29	115
GeneralGeomMethods	71	4	1	4	14
ReadMeshData	56	31	1	25	101
DissertationFEPrototype	62	24	7	39	126
Form1	74	7	7	12	22
Program	56	10	1	13	43
Control	55	7	1	20	61
DissertationFEPrototype.ModelUpdate	62	37	1	32	129
ReadAnalysisData	62	9	1	6	49
WriteNewMeshData	61	28	1	28	80
DissertationFEPrototype.FEModelUpdate.Read	60	26	1	21	62
ReadNodes	63	10	1	7	26
ReadElements	58	16	1	21	36
DissertationFEPrototype.Optimisations	53	57	1	29	161
OptimisationManager	53	57	1	29	161

Figure 31: The metrics calculated by visual studio for the all classes in the final system

I Sorting Nodes by Splitting With Planes

Attempting to arrive at a more general solution for sorting nodes in 3D space focus was directed towards the more complex Hex8 element type as this represented a more complete example of a node sorting problem that may have to be solved. Consideration of Hex8 elements led to the realisation that the most important task in sorting nodes for an arbitrary type is to simply establish the corner nodes relative to that type. Having established corners correctly sorting then simply required adding them to a list in the order specified by LISA.

The subsequent method which was used to successfully establish corners for both Quad4 and Hex8 elements during the earlier prototyping stages of the project was to split nodes for each element using planes running along the x, y and z axis as can be seen in Figure 9 below.



(a) Dividing a Quad4 along planes to establish each node as a corner point

(b) Dividing a Hex8 along planes to establish each node as a corner point

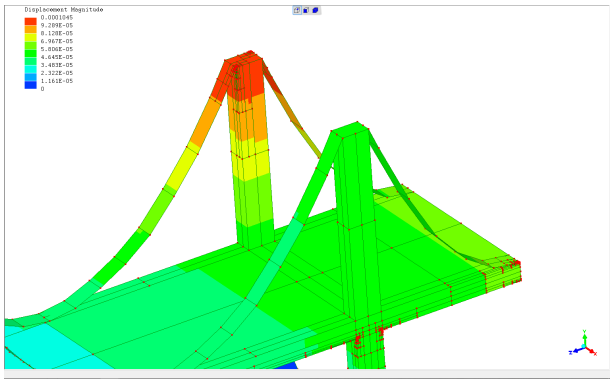
Figure 32: Splitting Element points using x, y and z planes in order to perform ordering for LISA

Although this approach resolved the initial problems resulting from simply trying to traverse the nodes it did not offer a strong general case solution to the problem with the code for a Hex8 element needing to be significantly different and more complex than that of a Quad4 and with the potential for the most complex FE element types such as wedge15, hex20, and pyr13 there would again be an even greater numbers of plane divisions yet again.

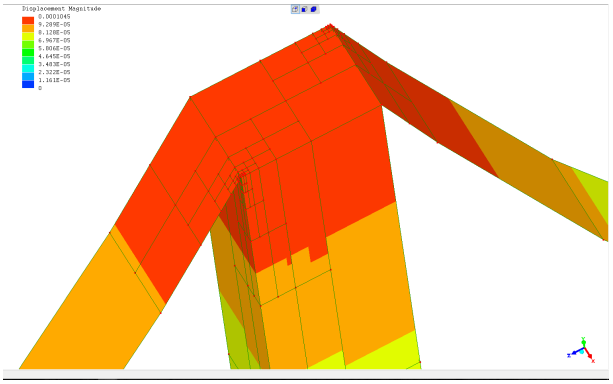
J Mesh Refinements

This appendix item attempt too show the general mesh that are formed using the heuristic and stress based refinement strategy with examples of where a heuristic has been placed well and poorly and where there is also variation in the threshold used to decide whether elements are meshed with the stress variable. Although in the rest of the models we are looking for stress since this is the primary variable of interest displacement has been selected as the analysis variable for displacement due to it producing a clearer gradient than stress.

J.1 Heuristic Refinement



(a) Important edges specified effectively to facilitate pre-emptive meshing of area which undergoes high stress



(b) Close up view of refinement for high displacement areas

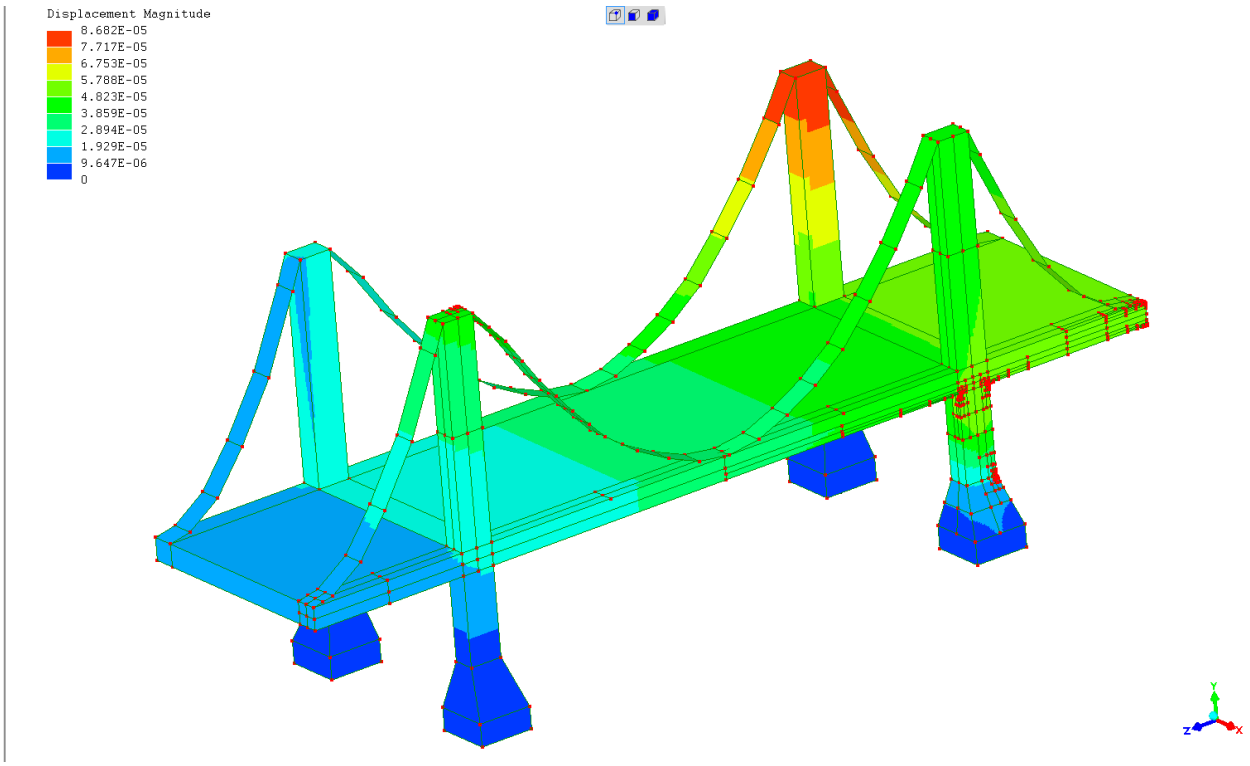


Figure 34: Important edges more poorly specified missing high displacement region on top of furthest suspension bridge tower

J.2 Stress Refinement

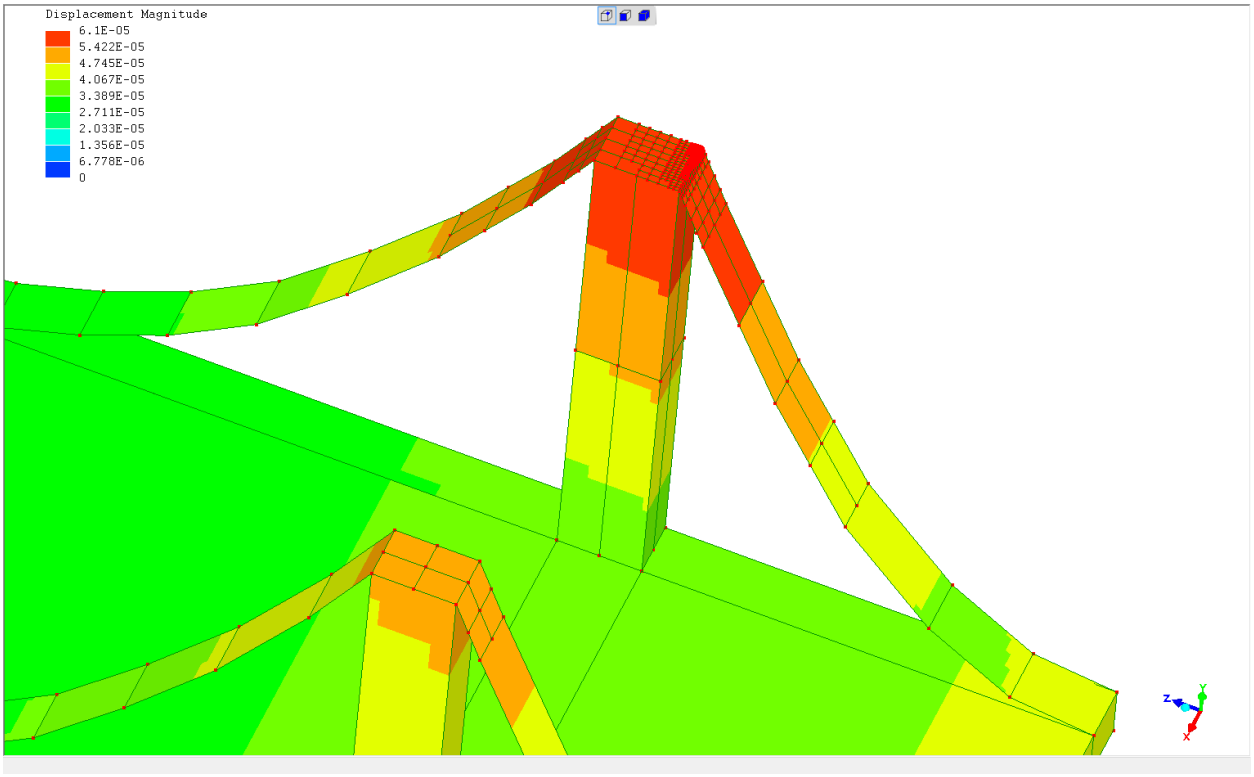


Figure 35: Iterative stress/ displacement refinement method used to focus meshing on the top 6% most displaced region of model

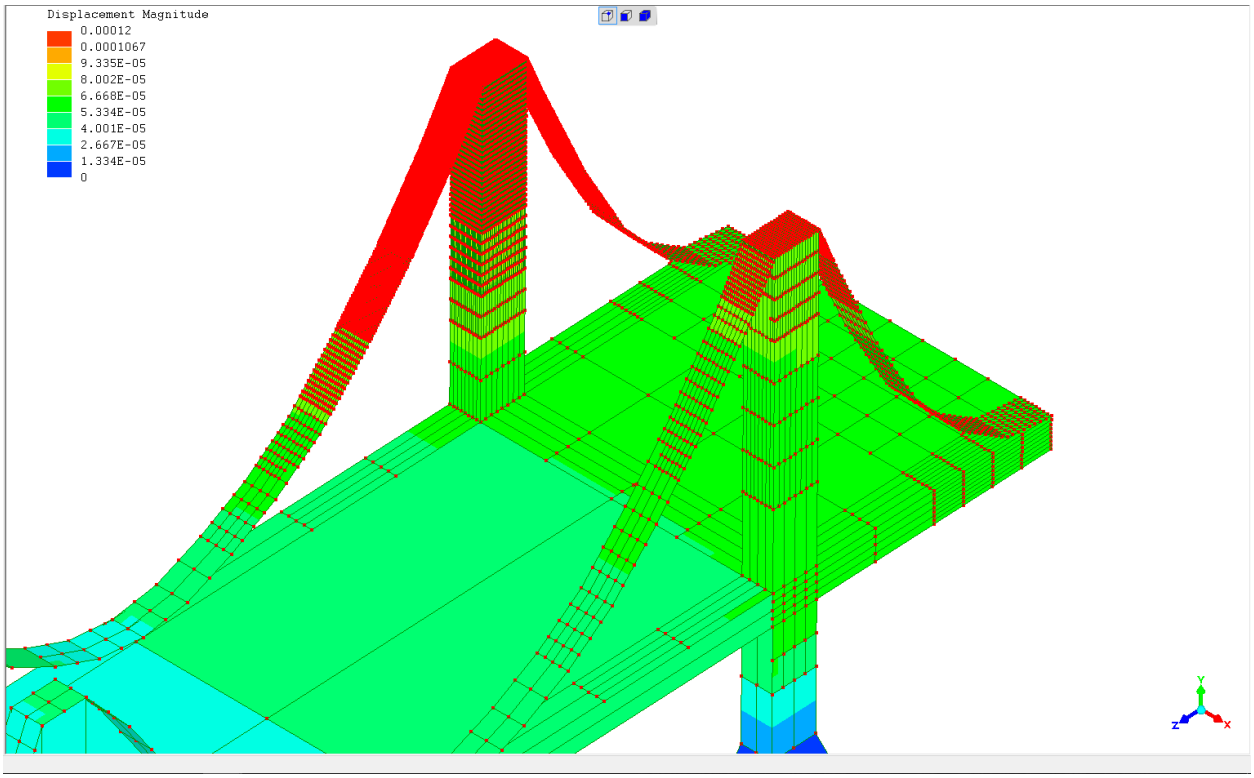


Figure 36: Iterative refinement of high displacement but with the remesh threshold specified as the average displacement across the whole model. A consequence of this is the gradient of refinement fidelity that can be seen corresponding to the importance of that part of the structure

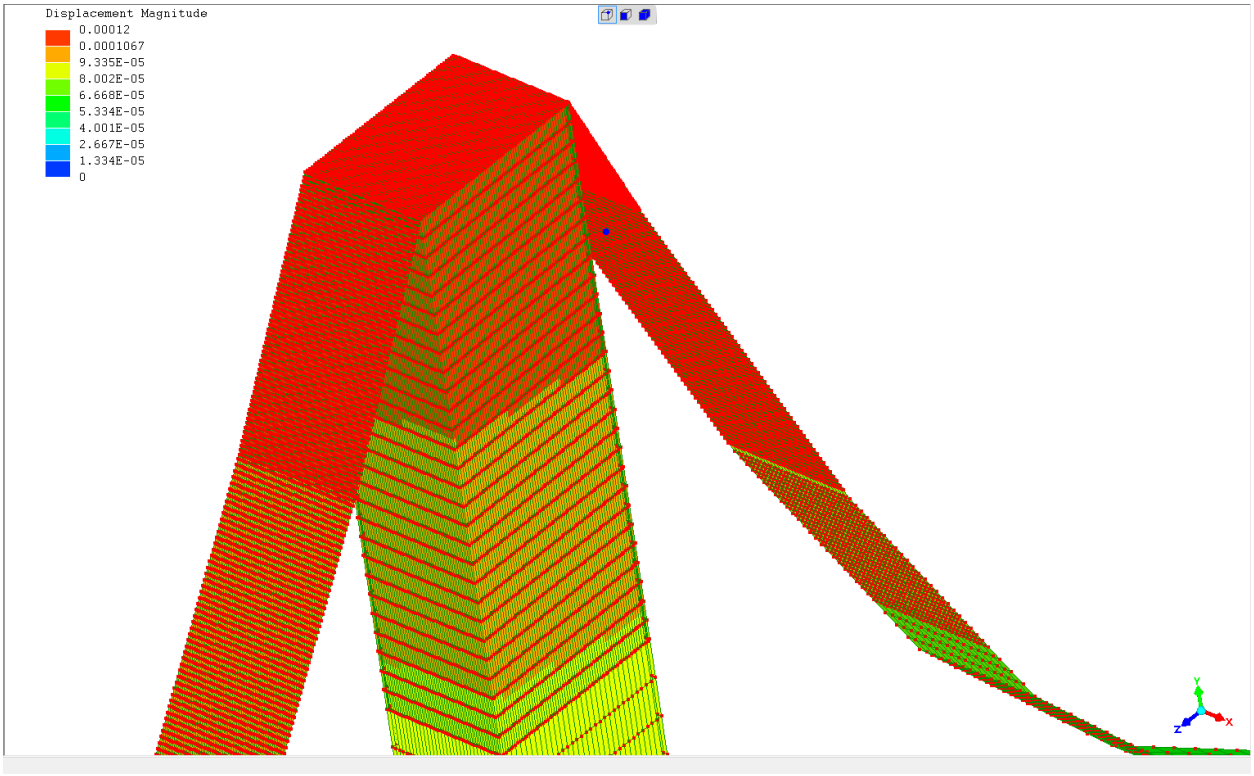


Figure 37: Closer view of very high meshing intensity

K Paper Mill Simulation Results

For the paper mill simulation angular forces were set up around the outside of the disk so as to simulate the effect of the disk rotating at high speed, with it also being pulled outwards in the axial direction. This generated some interesting patches of stress across the main body of the structure which could easily be specified as edge rules. Looking at figure 34 it is possible to see very high range of stress values for stress observable within the model as a consequence of stress concentrating at particular points.

analyzed. The first example was the cylinder in Fig. 2 [4], the second example was the hook in Fig. 4 and the last example was the cross-section of a paper mill in Fig. 5 [5].

Golem needs three types of input files to build the rules:

- foreground examples,
- negative examples,
- background facts.

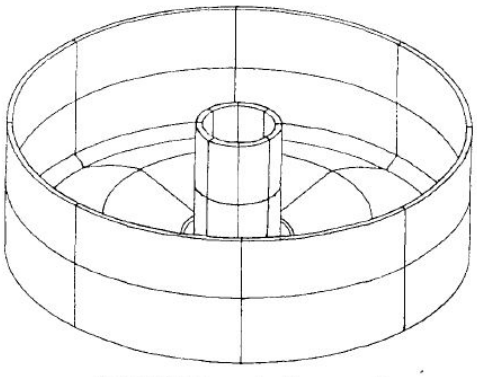
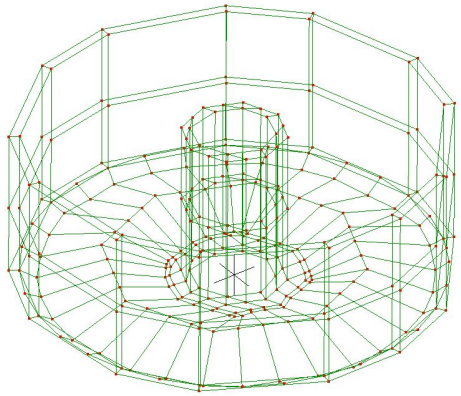


Fig. 5. Training example: the paper mill.

(a) Half of Cylinder structure described by dolsak in his papers for training ILP system



(b) Replication of mesh structure specified by Dolsak within his paper [3]

Figure 38: Execution time increase compared to the amount of information revealed for the different approaches

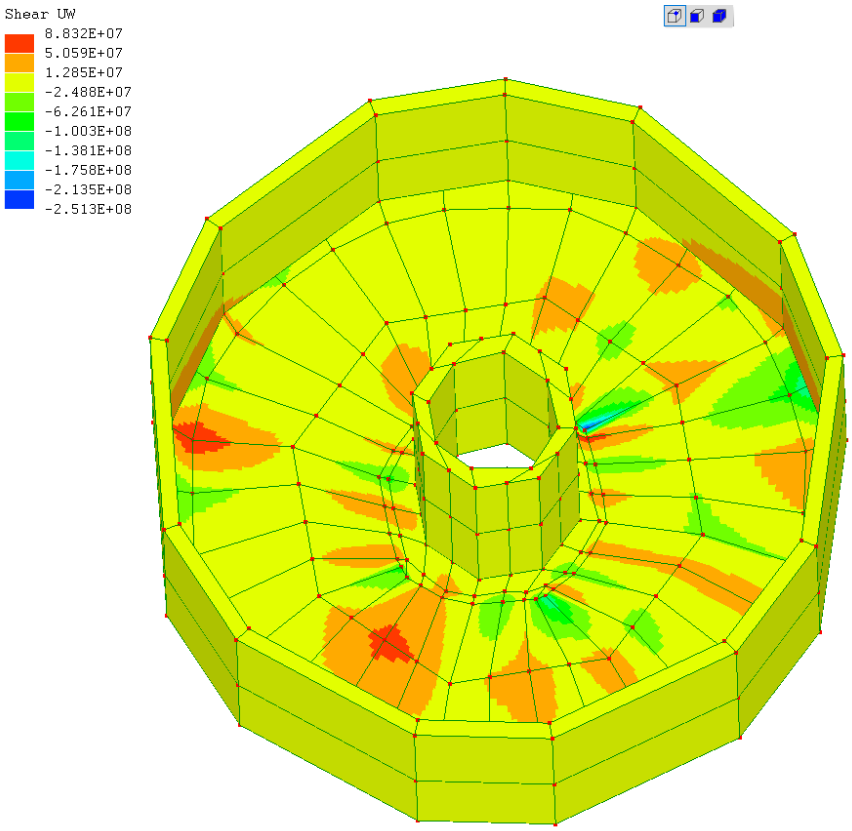


Figure 39: The initially stressed paper mill part used to define edge sets for further meshing, stress concentrations can be observed in red with colour coding at the top indicating showing a rapidly exponential increase concentrated at those points

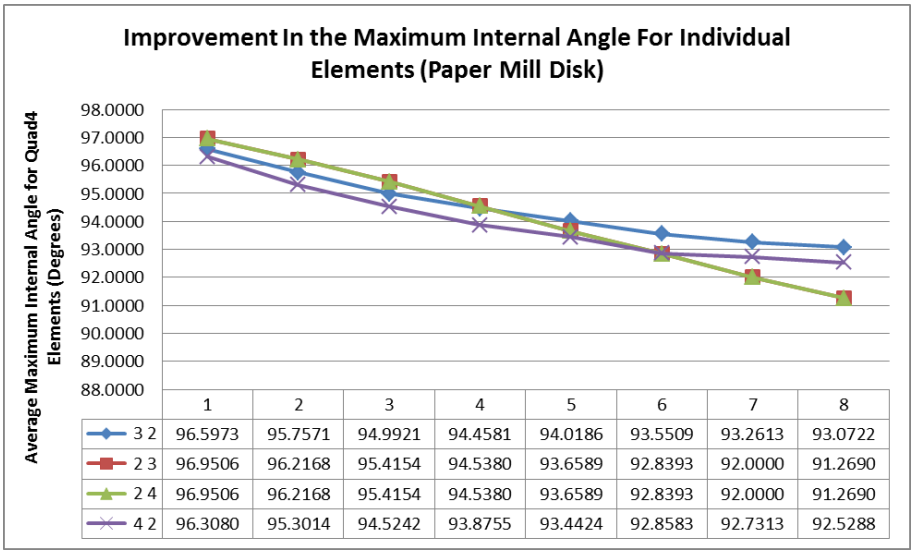
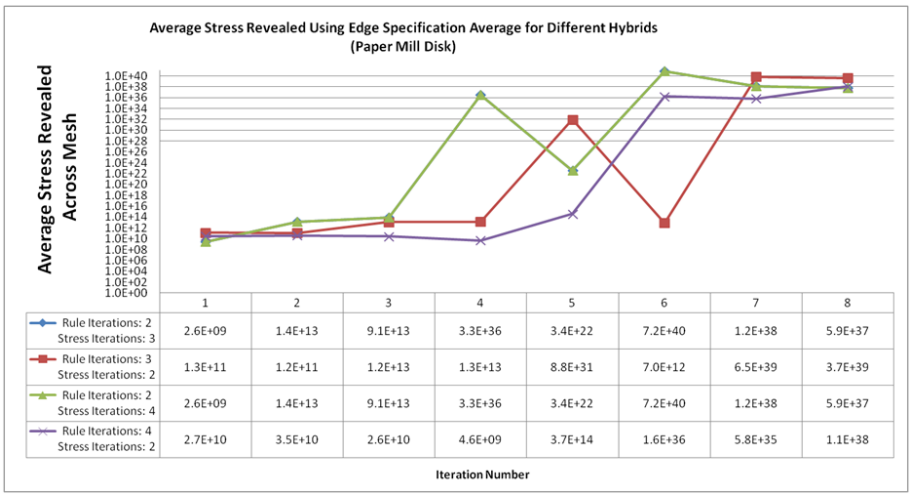
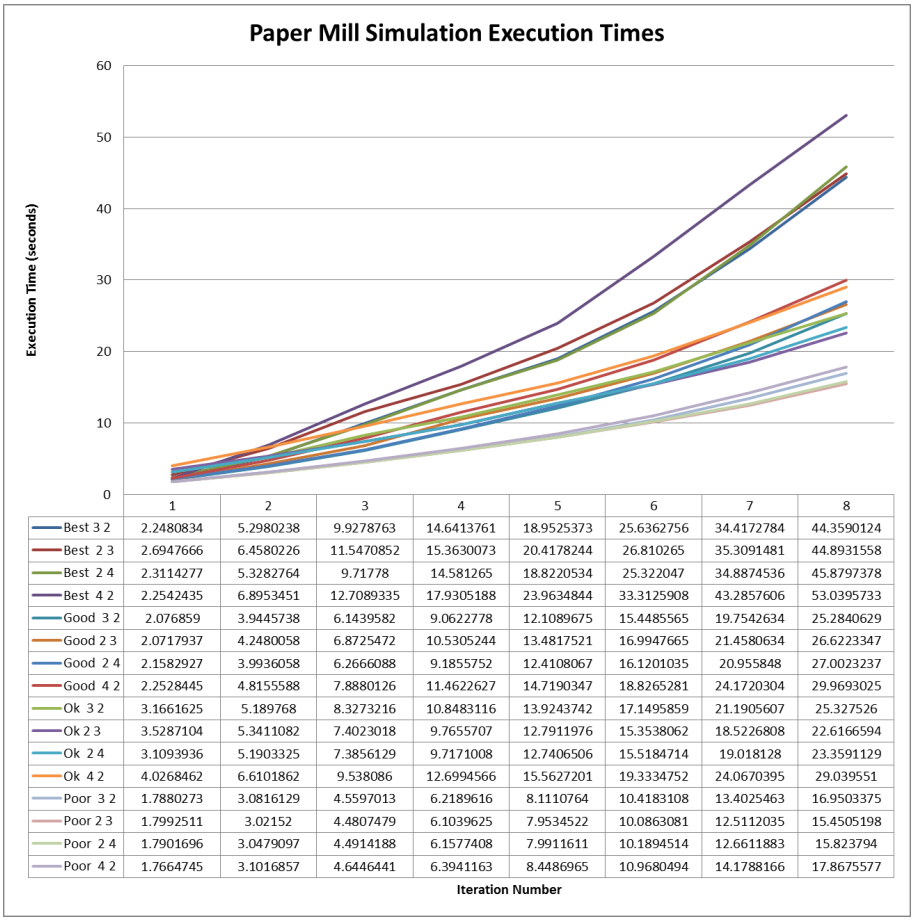


Figure 40: Improvement in the maximum internal Angle for Quad4 Elements



	1	2	3	4	5	6	7	8
Rule Iterations: 2 Stress Iterations: 3	2.6E+09	1.4E+13	9.1E+13	3.3E+36	3.4E+22	7.2E+40	1.2E+38	5.9E+37
Rule Iterations: 3 Stress Iterations: 2	1.3E+11	1.2E+11	1.2E+13	1.3E+13	8.8E+31	7.0E+12	6.5E+39	3.7E+39
Rule Iterations: 2 Stress Iterations: 4	2.6E+09	1.4E+13	9.1E+13	3.3E+36	3.4E+22	7.2E+40	1.2E+38	5.9E+37
Rule Iterations: 4 Stress Iterations: 2	2.7E+10	3.5E+10	2.6E+10	4.6E+09	3.7E+14	1.6E+36	5.8E+35	1.1E+38

Figure 41: Improvement on average for detecting across all nodes within the model over multiple iterations, results for each weighting with different edge heuristics also averaged



	1	2	3	4	5	6	7	8
Best 3 2	2.2480834	5.2980238	9.9278763	14.6413761	18.9525373	25.6362756	34.4172784	44.3590124
Best 2 3	2.6947666	6.4580226	11.5470852	15.3630073	20.4178244	26.810265	35.3091481	44.8931558
Best 2 4	2.3114277	5.3282764	9.71778	14.581265	18.8220534	25.322047	34.8874536	45.8797378
Best 4 2	2.2542435	6.8953451	12.7089335	17.9305188	23.9634844	33.3125908	43.2857606	53.0395733
Good 3 2	2.076859	3.9445738	6.1439582	9.0622778	12.1089675	15.4485565	19.7542634	25.2840629
Good 2 3	2.0717937	4.2480058	6.8725472	10.5305244	13.4817521	16.9947665	21.4580634	26.6223347
Good 2 4	2.1582927	3.9936058	6.2666088	9.1855752	12.4108067	16.1201035	20.955848	27.0023237
Good 4 2	2.2528445	4.8155588	7.8880126	11.4622627	14.7190347	18.8265281	24.1720304	29.9693025
Ok 3 2	3.1661625	5.189768	8.3273216	10.8483116	13.9243742	17.1495859	21.1905607	25.327526
Ok 2 3	3.5287104	5.3411082	7.4023018	9.7655707	12.7911976	15.3538062	18.5226808	22.6166594
Ok 2 4	3.1093936	5.1903325	7.3856129	9.7171008	12.7406506	15.5184714	19.018128	23.3591129
Ok 4 2	4.0268462	6.6101862	9.538086	12.6994566	15.5627201	19.3334752	24.0670395	29.039551
Poor 3 2	1.7880273	3.0816129	4.5597013	6.2189616	8.1110764	10.4183108	13.4025463	16.9503375
Poor 2 3	1.7992511	3.02152	4.4807479	6.1039625	7.9534522	10.0863081	12.5112035	15.4505198
Poor 2 4	1.7901696	3.0479097	4.4914188	6.1577408	7.9911611	10.1894514	12.6611883	15.823794
Poor 4 2	1.7664745	3.1016857	4.6446441	6.3941163	8.4486965	10.9680494	14.1788166	17.8675577

Figure 42: Time taken per iteration using the different hybrid weightings with varying edge quality specifications

L Half Cylinder Simulation Results

The half cylinder was the third model used to test the system.

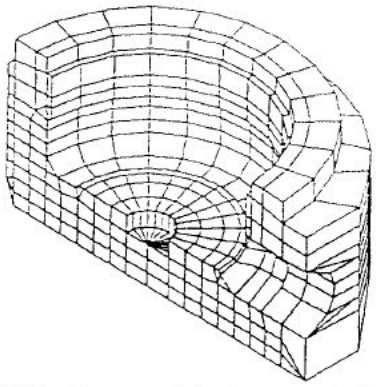
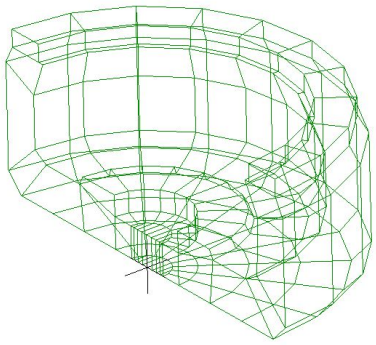


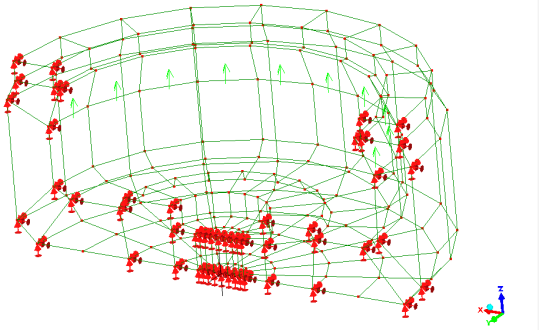
Fig. 2. Finite element mesh for the structure in Fig. 1.

(a) Half of Cylinder structure described by dolsak in his papers for training ILP system

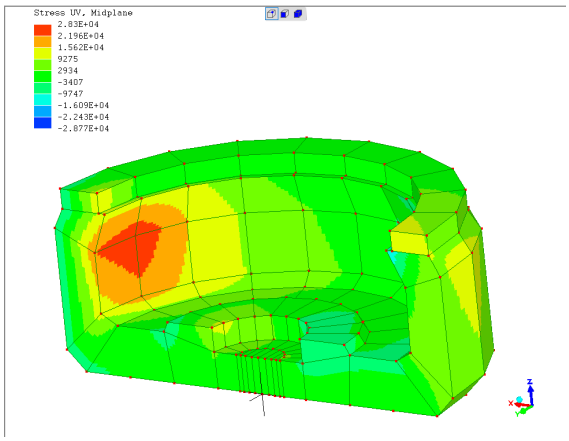


(b) Replication of mesh structure specified by Dolsak within his paper [3]

Figure 43: Execution time increase compared to the amount of information revealed for the different approaches

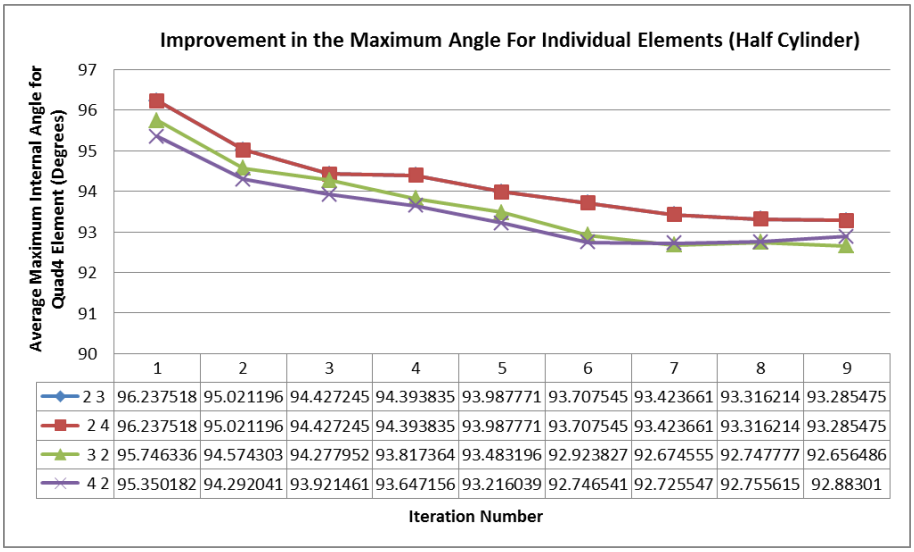


(a) Cylinder Constrained by its adjacent half with forces applied up and outwards on its inner rim



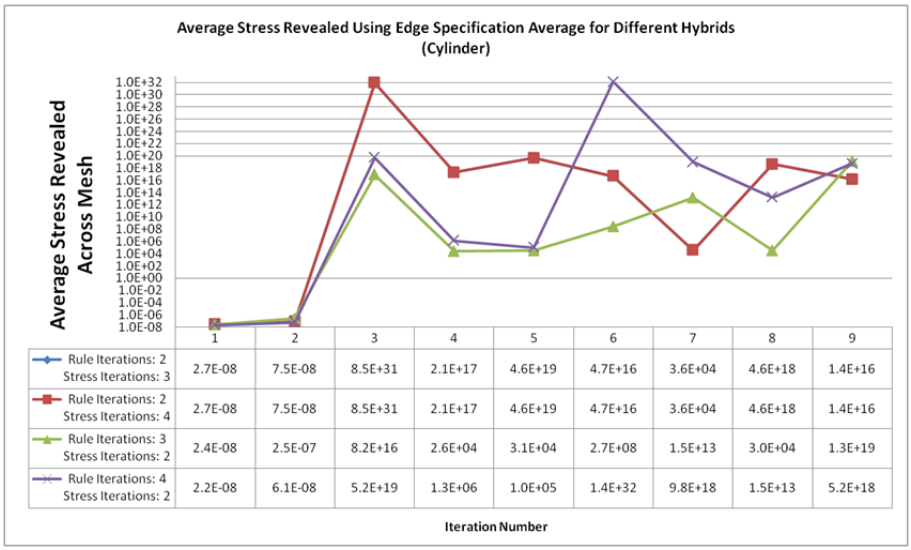
(b) Half cylinder with an initial stress concentration having performed a simple execution of the model.

Figure 44: Initial configuration for the half cylinder and some stresses revealed on the structure



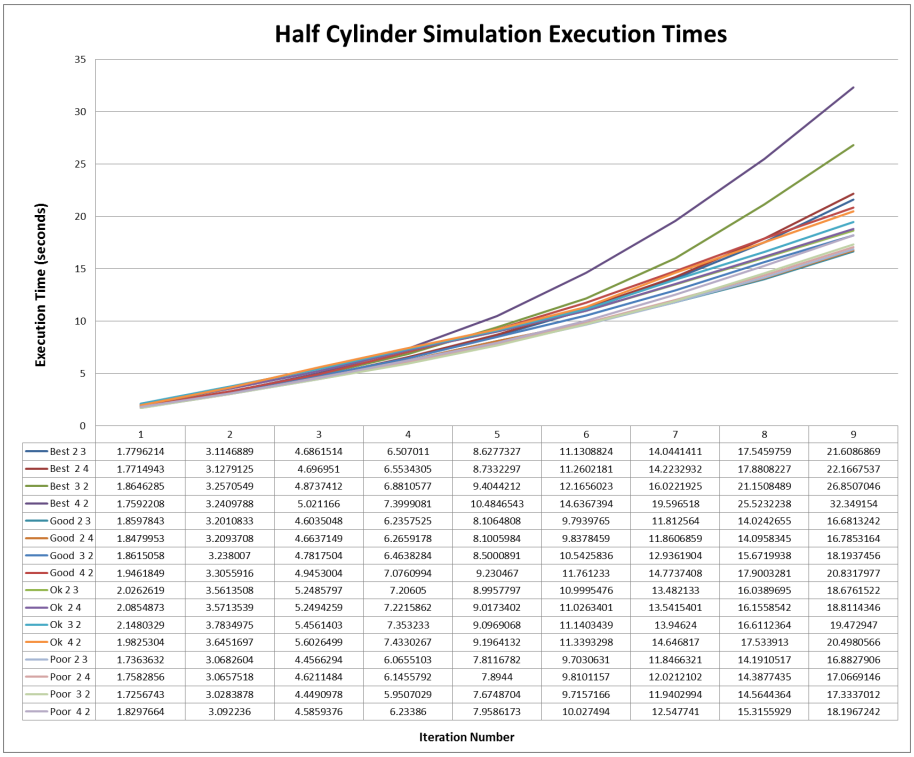
	1	2	3	4	5	6	7	8	9
2 3	96.237518	95.021196	94.427245	94.393835	93.987771	93.707545	93.423661	93.316214	93.285475
2 4	96.237518	95.021196	94.427245	94.393835	93.987771	93.707545	93.423661	93.316214	93.285475
3 2	95.746336	94.574303	94.277952	93.817364	93.483196	92.923827	92.674555	92.747777	92.656486
4 2	95.350182	94.292041	93.921461	93.647156	93.216039	92.746541	92.725547	92.755615	92.88301

Figure 45: Improvement in corner angles for Quad4 elements using the different hybrid methods on the cylinder



	1	2	3	4	5	6	7	8	9
Rule Iterations: 2 Stress Iterations: 3	2.7E-08	7.5E-08	8.5E+31	2.1E+17	4.6E+19	4.7E+16	3.6E+04	4.6E+18	1.4E+16
Rule Iterations: 2 Stress Iterations: 4	2.7E-08	7.5E-08	8.5E+31	2.1E+17	4.6E+19	4.7E+16	3.6E+04	4.6E+18	1.4E+16
Rule Iterations: 3 Stress Iterations: 2	2.4E-08	2.5E-07	8.2E+16	2.6E+04	3.1E+04	2.7E+08	1.5E+13	3.0E+04	1.3E+19
Rule Iterations: 4 Stress Iterations: 2	2.2E-08	6.1E-08	5.2E+19	1.3E+06	1.0E+05	1.4E+32	9.8E+18	1.5E+13	5.2E+18

Figure 46: Stress revealed for each iteration using the different hybrid methods



	1	2	3	4	5	6	7	8	9
Best 2 3	1.7796214	3.1146889	4.6861514	6.507011	8.6277327	11.1308824	14.0441411	17.5459759	21.6086869
Best 2 4	1.7714943	3.1279125	4.696951	6.5534305	8.7332297	11.2602181	14.2232932	17.8808227	22.1667537
Best 3 2	1.8646285	3.2570549	4.8737412	6.8810577	9.4044212	12.1656023	16.0221925	21.1508489	26.8507046
Best 4 2	1.7592208	3.2409788	5.021166	7.3999081	10.4846543	14.6367394	19.596518	25.5232238	32.349154
Good 2 3	1.8597843	3.2010833	4.6035048	6.2357525	8.1064808	9.7939765	11.812564	14.0242655	16.6813242
Good 2 4	1.8479953	3.2093708	4.6637149	6.2659178	8.1005984	9.8378459	11.8606859	14.0958345	16.7853164
Good 3 2	1.8615058	3.238007	4.7817504	6.4638284	8.5000891	10.5425836	12.9361904	15.6719938	18.1937456
Good 4 2	1.9461849	3.3055916	4.9453004	7.0760994	9.230467	11.761233	14.7737408	17.9003281	20.8317977
Ok 2 3	2.0262619	3.5613508	5.2485797	7.20605	8.9957797	10.9995476	13.482133	16.0389695	18.6761522
Ok 2 4	2.0854873	3.5713539	5.2494259	7.2215862	9.0173402	11.0263401	13.5415401	16.1558542	18.8114346
Ok 3 2	2.1480329	3.7834975	5.4561403	7.353233	9.0969068	11.1403439	13.94624	16.6112364	19.472947
Ok 4 2	1.9825304	3.6451697	5.6026499	7.4330267	9.1964132	11.3393298	14.646817	17.533913	20.4980566
Poor 2 3	1.7363632	3.0682604	4.4566294	6.0655103	7.8116782	9.7030631	11.8466321	14.1910517	16.8827906
Poor 2 4	1.7582856	3.0657518	4.6211484	6.1455792	7.8944	9.8101157	12.0212102	14.3877435	17.0669146
Poor 3 2	1.7256743	3.0283878	4.4490978	5.9507029	7.6748704	9.7157166	11.9402994	14.5644364	17.3337012
Poor 4 2	1.8297664	3.092236	4.5859376	6.23386	7.9586173	10.027494	12.547741	15.3155929	18.1967242

Figure 47: Time taken per iteration using the different hybrid weightings with varying edge quality specifications

M Gantt Chart for Project Time Management

This appendix lists the main tasks that were undertaken for the project and how they fit into the time plan that was initially formulated at the start of the project.
Tasks:

A Write supervisor project Proposal.

B Review currently available FE tools that I can use as the basis for conducting my work.

C Conduct research on current applications of AI and machine learning for FEA.

D Given feasibility of work begin to build basic interfaces with FE tool and consider architecture for tool.

E Conduct research on mesh assessment metrics allowing for automatic quality check of generated meshes.
- research metrics
- investigate building them into prototype of tool

F Time allocation to catch up with University coursework.

G Write interim report for dissertation (deadline 08 December).

H Research possible methods for combining both stress based refinement and AI refinement
- explore research methods
- implement prototype of composition functions and run iteratively on basic model.

I Take Christmas time off/revise other modules.

J Research feasibility of distribution or concurrent execution of problem.

K Develop/obtain a series of finite element models of increasing complexity to test current prototype on.

L Having run the developed models look at fixing any problems that may have arisen as a result of increased complexity in geometry.

M Analyse results from running prototype on range of models and analyse/cross reference results against those reported in papers.

N Review and update methods before rerunning them on the models to hopefully achieve improved results.

O Begin to structure the layout/arguments I want to make for the final dissertation.

P Write final dissertation (Deadline 6th April 2017).

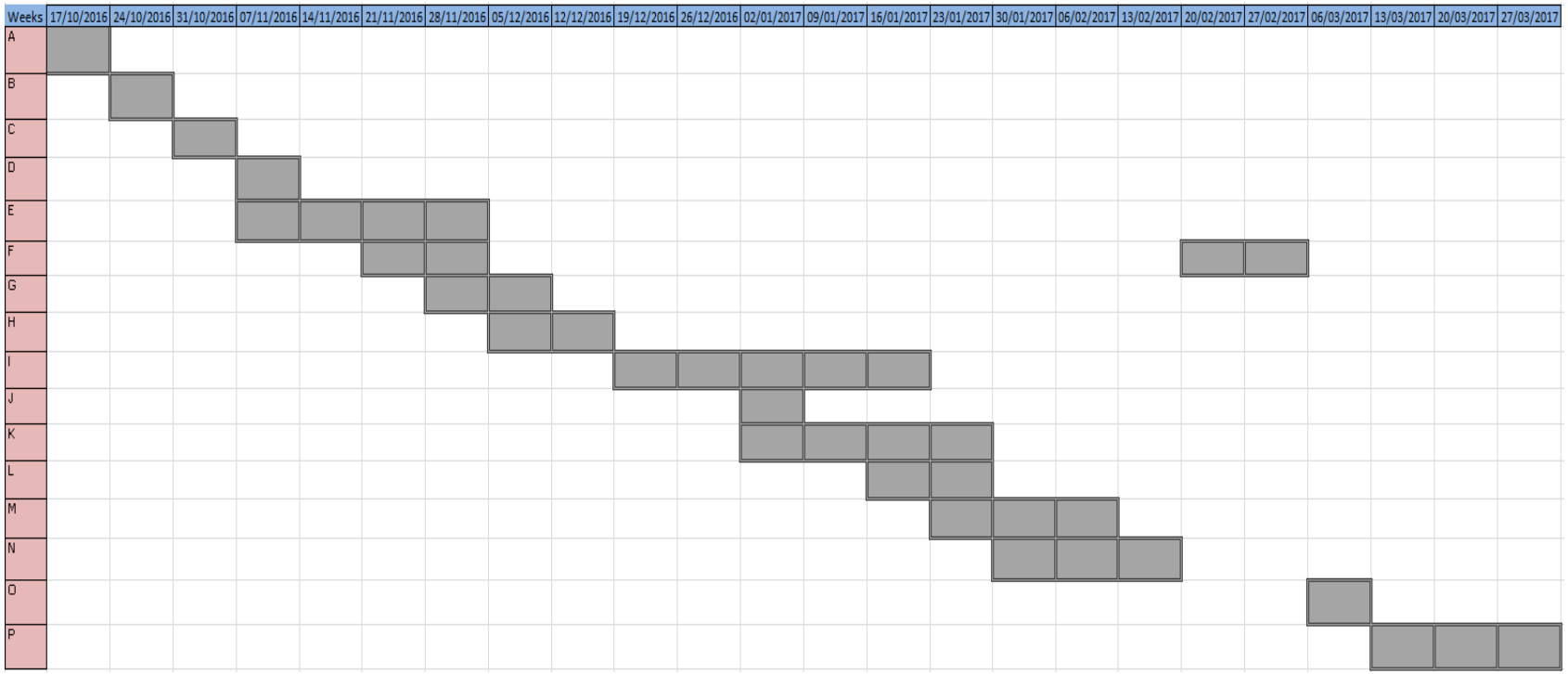


Figure 48: Gantt chart showing the initial plan for the projects progress