Materials created at Sandia National Laboratories that I will be using as I continue doing research while going to school at Brigham Young University.

PRESENTATIONS

Smart Refinement of Unstructured Conformal Hexahedral Meshes

August 1st, 2006

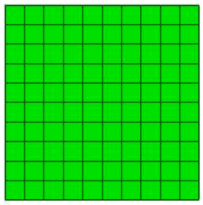
Michael Parrish
Student Intern

Mentors – Matthew Staten and Michael Borden
Org. 1421

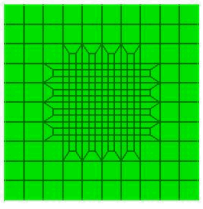
Funding provided by:
Nanoscience, Engineering, and Computation Institute at Sandia (NECIS)and
ASC through the Cubit Project

Benefits of Hex Refinement

- Hex refinement increases the density of the mesh in a specified region.
- Allows greater user control over mesh density and quality.
- More accurate analyses in specified regions because of the increased density.



Unrefined Mesh



Refined Mesh

Sandia National Laboratories

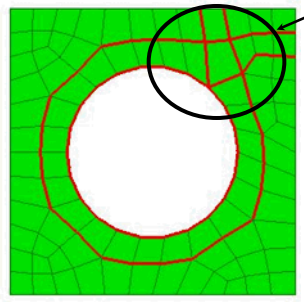
Old Hex Refinement Algorithm

- Implementation
  - The mesh was processed in sheets of hexes rather than on a hex by hex basis.

- Problems
  - Self-Intersecting Sheets
  - Concavities
  - Computationally Expensive

Sandia National Laboratories

Self-Intersecting Sheets

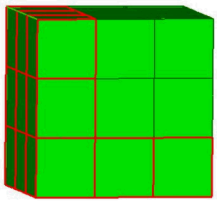


- The old algorithm was designed to handle a single refinement direction at a time.
- With self-intersecting sheets, this is no longer the case.

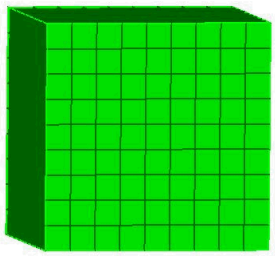
Sandia National Laboratories

Concavities

Desired refined hexes highlighted in red



Before



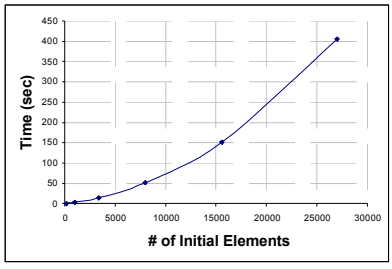
After

- The old algorithm cannot handle any type of concavity in the refinement region

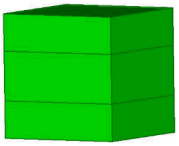
Sandia National Laboratories

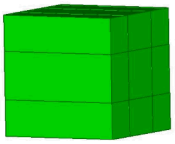
Computationally Expensive

- The old algorithm created and deleted many non-essential hexes which for large meshes made the time required for refinement grow exponentially.



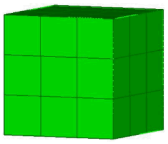
3 hexes are created and one hex is deleted.





9 hexes are created and 3 hexes are deleted.

27 hexes are created and 9 hexes are deleted.



Sandia National Laboratories

Smart Refinement

- Combines old and new theory with old and new techniques to solve the problems with the old algorithm.

- Two different procedures used in Smart Refinement.
  - Total Hex Refinement NEW!
  - Directional Hex Refinement
    - Similar to old algorithm with modifications

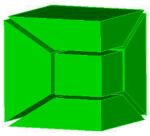
Sandia National Laboratories

Total Hex Refinement

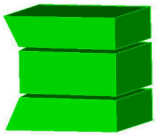
- Total Hex Refinement uses a template to create new hexes
- Only the original hex is deleted and only necessary hexes are created

1 to 27 template

1 to 13 template

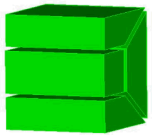


1 to 5 template

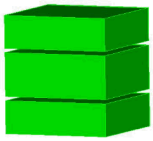


1 to 3 template w/ 1 concavity

1 to 3 template w/ 2 concavities



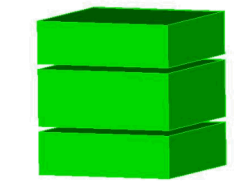
1 to 4 template



1 to 3 template

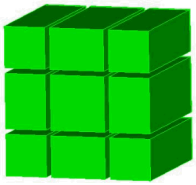
Sandia National Laboratories

Directional Hex Refinement



Step 1

Hexes are created one principle direction at a time.



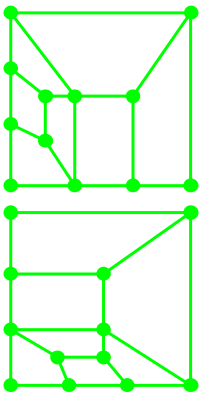
Step 2

Directional Hex Refinement is used when Total Hex Refinement is impossible!

- **Follows same basic principles as old algorithm.**
- **Refines on a hex by hex basis rather than in hex sheets**
- **Conformity is an issue**
- **Ranking System**

Sandia National Laboratories

Conformity

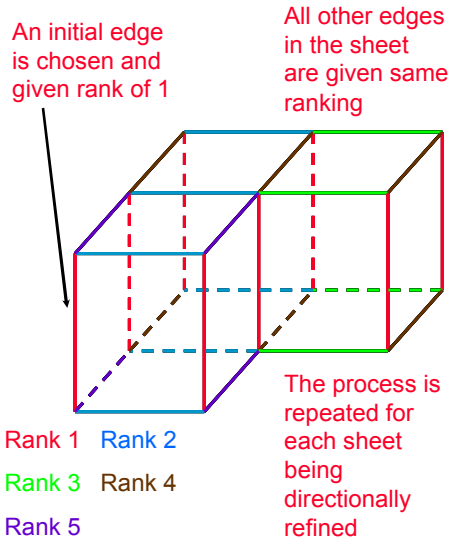


- **Since the old algorithm refined in hex sheets, conformity was not an issue.**
- **Conformity is an issue with smart refinement because refinement takes place on a hex by hex basis.**

Both Quads have the same split edges, however they are not conformal

Sandia National Laboratories

Ranking System

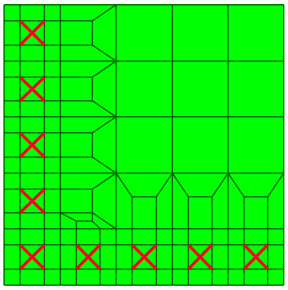


- **Ranking each edge before Directional Refinement ensures that there will be no conformity problems.**
- **Once ranking is complete, directional refinement occurs in ranked order for each hex.**

Sandia National Laboratories

Outline of Algorithm

- **Total Hex Refinement**
  - **1 to 27 template is applied to target hexes**
  - **Other templates are applied to boundary hexes**
- **Ranking System applied to remaining unrefined hexes**
- **Directional Refinement**
  - **Processed in hex by hex fashion**
  - **Direction having the lowest rank is refined first. The process is continued until complete**

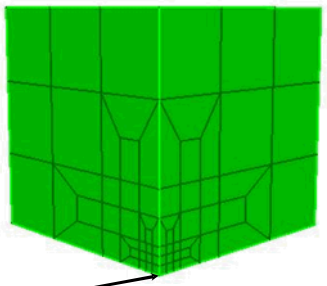


- Indicates target hexes

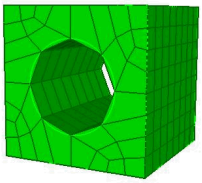
Sandia National Laboratories

Examples of the Smart Refinement

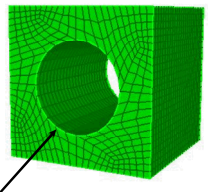
- Smart Refinement allows a higher density of hexes near a vertex while not dramatically increasing the element count



Vertex



Before



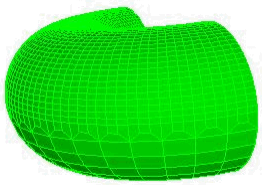
After

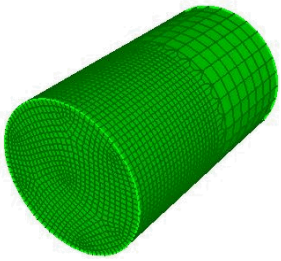
Refinement allows hole to be modeled accurately!

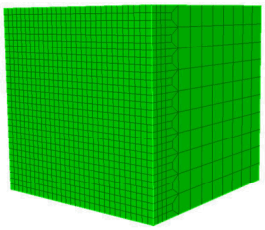
Sandia National Laboratories

Examples of Smart Refinement

- These examples use the Total Hex Refinement capabilities.





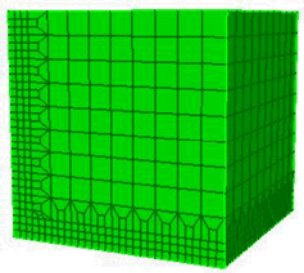


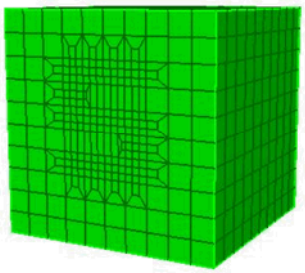
There are no concavities in these examples!

Sandia National Laboratories

Examples of Smart Refinement

- These examples use full Smart Refinement capability





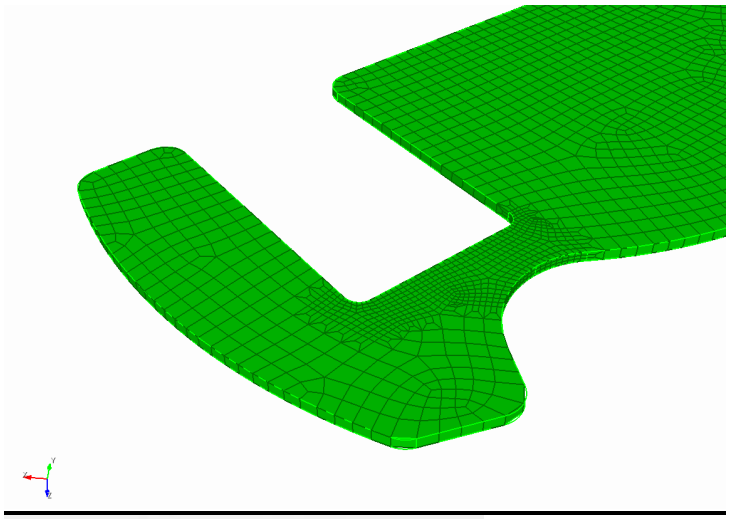
Sandia National Laboratories

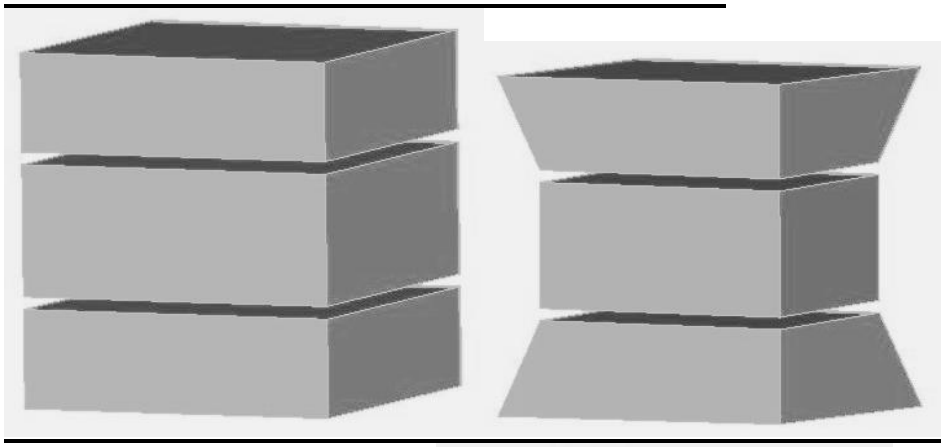
Future Work

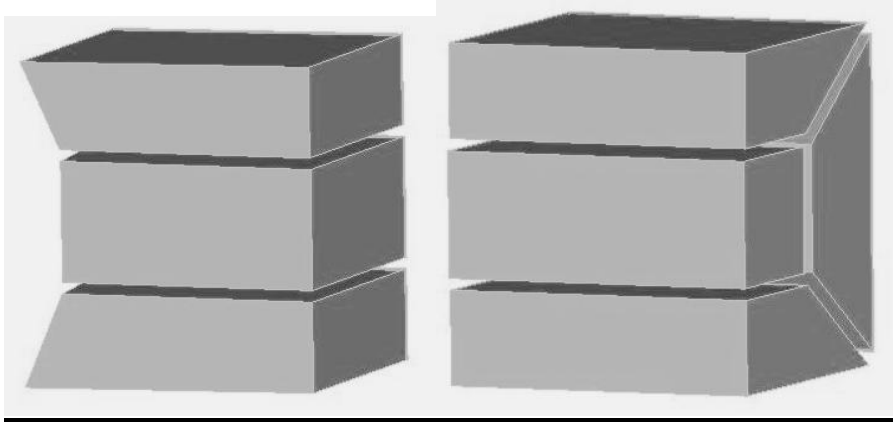
- Speed
  - Currently, the speed of Smart Refinement is on the same order of magnitude as old algorithm.
  - Next few months will be spent on speeding up bottle-neck areas while maintaining functionality.
- Testing

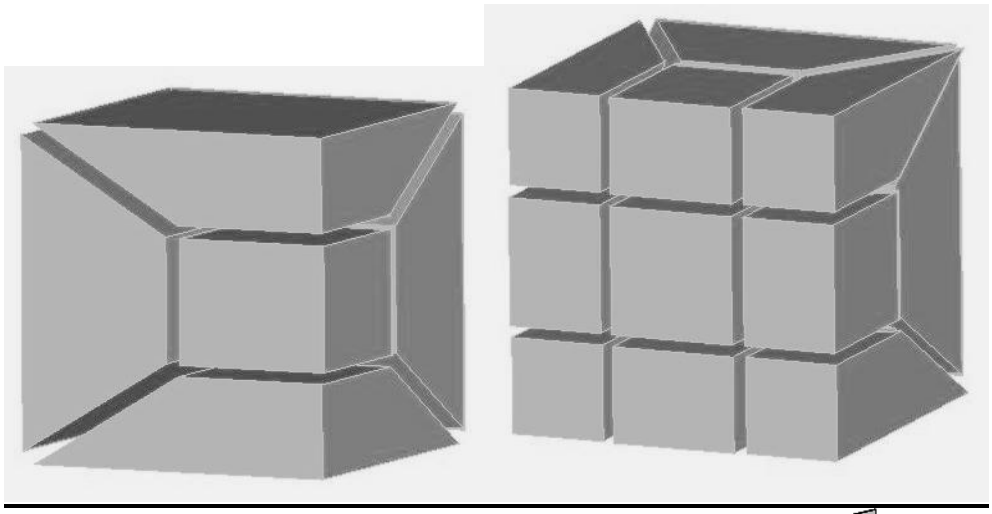
Sandia National Laboratories

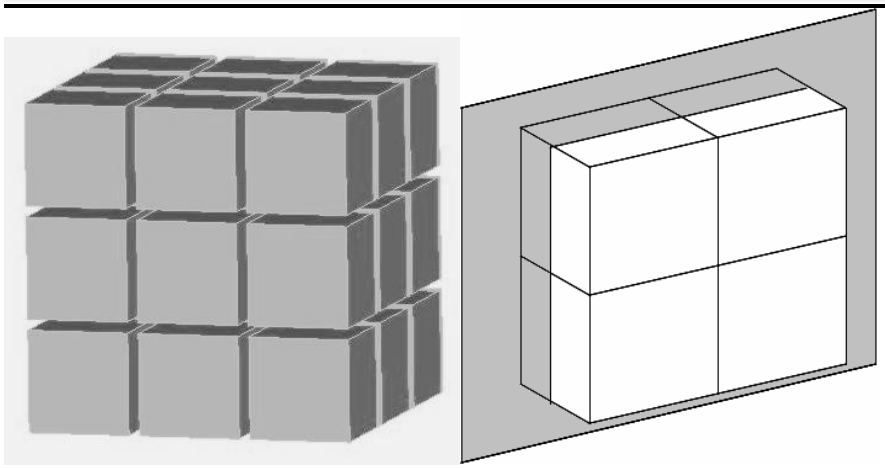
PICTURES

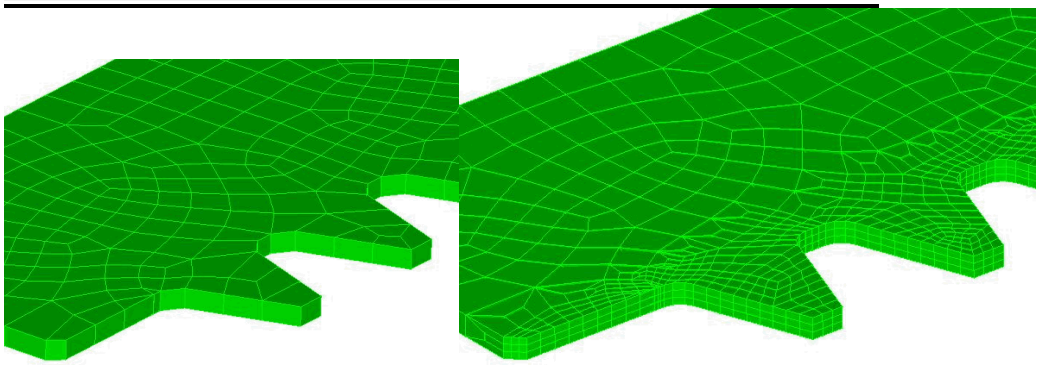








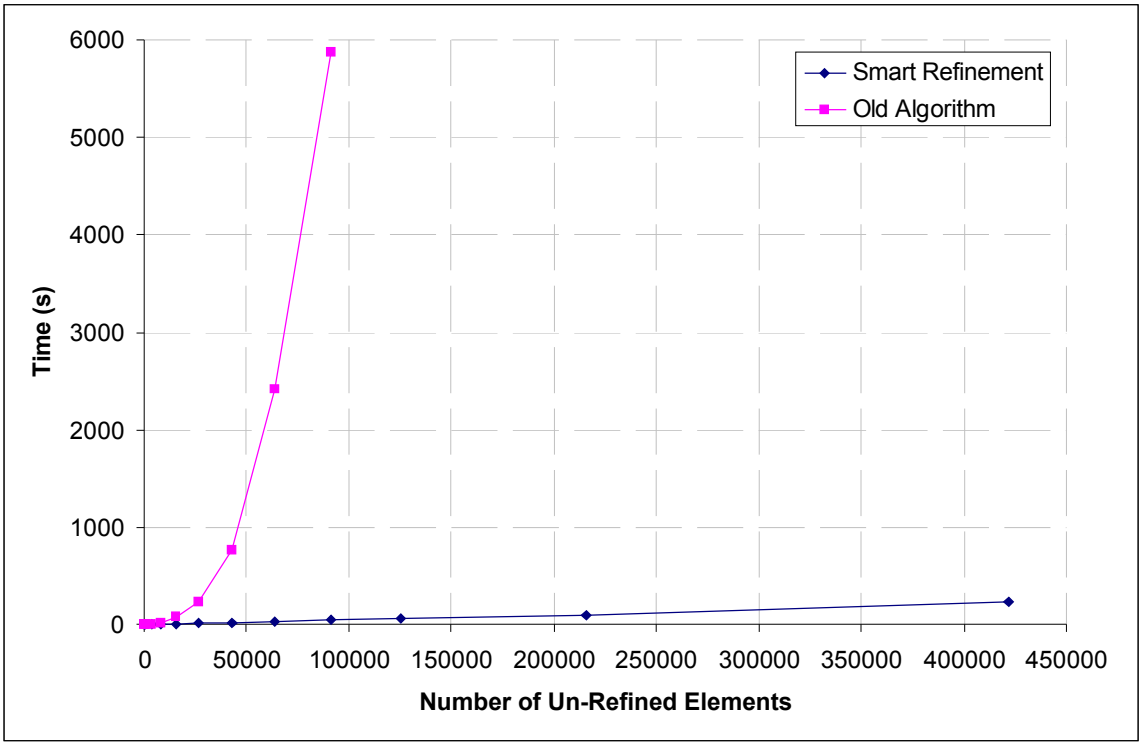




GRAPHS AND REPORTS

Interval	Element Count	New Refinement	Old Refinement
0	0	0	0
5	125	0.05	0.12
10	1000	0.38	1
15	3375	1.19	5.2
20	8000	3.05	21.4
25	15625	6.29	75.55

30	27000	10.51	230.39
35	42875	19.63	770.39
40	64000	28.21	2418
45	91125	48.56	5868
50	125000	62.52	
60	216000	95.51	
75	421875	236.75	



April Report

- Continued working on Mesh Cutting bugs. I have not been able to resolve them I believe they will require some big changes to the mesh cutting code. I have been toying with the idea to use a more optimized approach in selecting which side of a hex to move to the cutting plane.

May Goals

- Move down to Albuquerque to start my new internship for the summer at Sandia National Laboratories!! I will continue the research here that I started at BYU.

May 2006 Progress Report for Mike Parrish

Hex Refinement

- Created alternate data structure to perform the refinement so that the mesh will not be changed until after the refinement is preformed.
- Created the 1 to 27, 1 to 13, 1 to 5, and 1 to 1 templates used in total hex refinement.
- I created a PowerPoint presentation that outlines my plans for this summer.
- The current state of hex refinement is that is can be used on meshes that contain no concavities.

Goals for June

- Finish the other 4 templates used in total hex refinement
- Create faces in cubit once refinement is done so that one will not need to type "draw hex all" to see the refinement
- Go through code I have written and add comments and make things easier to understand for others.
- Spend some time on the C++ courses in which I am enrolled
- Do some testing of the current code that I have implemented.

July Report

- Prepared presentation and abstract for SIP Symposium
- Worked on bugs found in code
- Worked on improving the scalability of hex refinement. The scalability is now linear where as before it was cubic

August Goals

- Work on cleaning up code
- Continue working on speed
- Write paper for NECIS
- Prepare to go back to BYU for Fall Semester

SMART REFINEMENT OF UNSTRUCTURED CONFORMAL HEXAHEDRAL MESHES

Michael Parrish
Brigham Young University
Teddy Blacker, Manager (Org. 1421 – Computational Modeling Sciences)
Matthew Staten and Michael Borden, Technical Advisors
July 12th 2006

Localized hexahedral (hex) modification algorithms modify unstructured conformal meshes allowing greater user control over mesh density and quality. Hex refinement increases the density of the mesh in a specified region proving extremely useful in many analyses. Previous research using solely Directional Refinement made the implementation computationally expensive and unable to handle concave refinement

regions and self-intersecting hex sheets. Smart Refinement is a new procedure that combines old and new theory to create an efficient and robust algorithm able to handle the above stated problems. The Smart Refinement procedure uses two different methodologies in the refinement process. These are: 1) Total Hex Refinement and 2) Directional Hex Refinement. Total Hex Refinement refines a hex in one step using one of seven templates. As each hex is processed, the correct template is selected and replaces the processed hex. Directional Refinement which was previously used for all refinement is only used when the more efficient Total Hex Refinement algorithm is impossible. A ranking system allows Directional Refinement to be done in the proper order thus eliminating conformity problems.

\begin{abstract}
Localized hexahedral (hex) modification algorithms modify unstructured conformal meshes allowing greater user control over mesh density and quality. Hex refinement increases the density of the mesh in a specified region proving extremely useful in many analysis. Previous research using solely directional refinement theory made the implementation computationally expensive and unable to handle concave refinement regions and self-intersecting hex sheets. Smart Refinement is a new procedure that combines old and new theory to create an efficient and robust algorithm able to handle the above stated problems. The Smart Refinement procedure uses two different methodologies in the refinement process. These are: 1) Total Hex Refinement and 2) Directional Hex Refinement. Total Hex Refinement refines a hex in one step using one of seven templates. As each hex is processed, the correct template is selected and replaces the processed hex. Directional Refinement which was previously used for all refinement is only used when the more efficient Total Hex Refinement algorithm is impossible. A ranking system allows Directional Refinement to be done in the proper order thus eliminating conformity problems. \end{abstract}

\section{Introduction}

Hex refinement is a mesh modification procedure which increases element density in a localized region. A refined mesh can increase the accuracy of the analysis within the refined region because of the increased element density as shown in Figure \ref{fig:comp1} and Figure \ref{fig:comp2}. While hex refinement can be an invaluable tool, previous theory and implementation were unable to provide needed capabilities and made hex refinement computationally expensive. The work presented in this paper therefore, attempts to solve these issues.

The following section discusses previous hex refinement theory and highlights problems addressed by the current work. Smart Refinement will then be introduced accompanied with an abbreviated outline of the algorithm. A comparison between Smart Refinement and the old hex refinement algorithm will follow showing that Smart Refinement is a more robust and efficient way to preform hex refinement. Finally, application to the field of nanoscience and nanotechnology will be discussed.

\begin{figure}[htb]

```
\begin{center}
\subfigure[Before Refinement]{\includegraphics[scale=.3]{test1.png}\label{fig:comp1}}
\subfigure[After Refinement]{\includegraphics[scale=.3]{test2.png}\label{fig:comp2}}
\caption{Purpose of Hex Refinement}
\end{center}\end{figure}

\section{Background}
Conformal refinement of unstructured hexahedral meshes was previously done utilizing the dual of the mesh also known as the spatial twist continuum (STC)\cite{bib2}. The dual is a geometric representation that explicitly defines the connectivity characteristics of a mesh \cite{dual}. A component of the dual is a hex sheet as shown in Figure \ref{fig:Mentor05}. Hexes were refined in hex sheets thus making the refinement conformal because of the connectivity properties associated with the dual of the mesh. While refinement using hex sheets has the advantage of conformity, three problems have arisen that limit both capability and speed of this theory. These three critical issues are: 1) Self-intersecting hex sheets, 2) Concavities, and 3) Scalability. Smart Refinement is an attempt to solve these three issues while maintaining a conformal mesh.

\begin{figure}[htb]
\begin{center}
\includegraphics[scale=.2]{hex_sheet.png}
\caption{Hex Sheet(Shown in Gray) of 4 Hexes}\label{fig:Mentor05}
\end{center}\end{figure}

\subsection{Self-Intersecting Hex Sheets}
A hex sheet starts at a boundary and ends at a boundary or it will form a closed loop. Sometimes meshing algorithms will create self-intersecting hex sheets. This means that before reaching the end, the hex sheet will include hexes more than one time. The old algorithm was unable to handle multiple refinement directions in the same hex sheet. In fact, it would create a non-conformal meshes which make an accurate analysis impossible. A two-dimensional example shown below in Figure \ref{fig:self} highlights the self-intersecting hex sheet.

\begin{figure}[htb]
\begin{center}
\includegraphics[scale=.2]{self_intersection.png}
\caption{Example of Self-Intersecting Hex Sheet}\label{fig:self}
\end{center}\end{figure}

\subsection{Concavities}
Concavities present a new set of problems for hex refinement. The old algorithm could not handle concavities so it would add hexes to to hex sheet until no concavities remained. While this solves the concavities problem, it leads to another problem which is excessive refinement. Excessive refinement can cause unwanted modification of a mesh and slow down the analysis process and accuracy. The templates to handle concavities existed but they were not implemented in the old algorithm\cite{bib1}.
```

\subsection{Scalability}
Refining a mesh using hex sheets ensures a conformal mesh, however the scalability of using such a process is on the order of n^3. The reason for this is the creation and deletion of intermediate hexes. The process occurs in the following manner. First as shown in Figure \ref{fig:method1}, the original hex is deleted and three new hexes are created in its place. Second, the three intermediate hexes are deleted and nine more intermediate hexes are created as in Figure \ref{fig:method2}. Finally, the nine intermediate hexes are deleted and 27 final hexes are created as in Figure \ref{fig:method3}. In this example, 13 hexes were deleted and 40 hexes were created. This requires significant computational power and time.

\begin{figure}[htb]
\begin{center}
\subfigure[3 Hexes]{\includegraphics[scale=.2]{oldmethod1.jpg}\label{fig:method1}}
\subfigure[9 Hexes]{\includegraphics[scale=.2]{oldmethod2.jpg}\label{fig:method2}}
\subfigure[27 Hexes]{\includegraphics[scale=.2]{oldmethod3.jpg}\label{fig:method3}}
\caption{Three Phase Process of Old Algorithm}
\end{center}\end{figure}

\section{Smart Refinement}
Smart Refinement is a new algorithm that attempts to make hex refinement more robust while reducing computational cost. This new algorithm changes the fundamental way in which hexes are processed. Rather than being processed in hex sheets, hexes are processed one at a time. This means that once a hex is processed it is completely refined and removed from the refinement algorithm. Immediately, this methodology solves the self-intersecting sheet problem and it will be shown later that Smart Refinement also solves the other problems stated in the previous section. It is also hoped that by processing hexes individually, future problems that may arise can be solved with more ease.

Smart Refinement can be broken down into two distinct processes. These are: 1) Total Hex Refinement and 2) Directional Hex Refinement. The remainder of this section will discuss how these two types of refinement work and how they function together to form the Smart Refinement algorithm.

\subsection{Templates}
Seven templates are used in Smart Refinement and are shown in the following figure. Figure \ref{fig:27}, Figure \ref{fig:13}, and Figure \ref{fig:5} are the most common templates used in the refinement process. In fact, any refinement not requiring concavities can be refined using these three templates. All seven templates are used in Total Hex Refinement while Figure \ref{fig:27} and Figure \ref{fig:13} are not used in Directional Hex Refinement. Figure \ref{fig:3conc} and Figure \ref{fig:3conc2} are used to handle concavities in the refinement process.

\begin{figure}[htb]

```
\begin{center}
\subfigure[1 to 27]{\includegraphics[scale=.25]{1to27.jpg}\label{fig:27}}
\subfigure[1 to 13]{\includegraphics[scale=.25]{1to13.jpg}\label{fig:13}}
\subfigure[1 to 5]{\includegraphics[scale=.25]{1to5.jpg}\label{fig:5}}
\subfigure[1 to 4]{\includegraphics[scale=.25]{1to4.jpg}\label{fig:4}}
\subfigure[1 to 3]{\includegraphics[scale=.25]{1to3.jpg}\label{fig:3}}
\subfigure[1 to 3 with 1
concavity]{\includegraphics[scale=.25]{1to3concave.jpg}\label{fig:3conc}}
\subfigure[1 to 3 with 2
concavities]{\includegraphics[scale=.25]{1to3concave2.jpg}\label{fig:3conc2}}
\caption{Templates Used in Total Hex Refinement}
\end{center}\end{figure}

\subsection{Total Hex Refinement}
Total Hex Refinement refines a hex in one step.  The initial hex is deleted and the final
hexes are created using one of the seven templates described in the previous subsection.
No intermediate hexes are created or deleted thus increasing the efficiency.  This type of
refinement is preferred and is used as the primary refinement method.

\subsection{Directional Hex Refinement}
If all meshes were refined using solely Total Hex Refinement, dozens of templates would
be required.  Most of these required templates are currently unknown or create low
quality elements.  Therefore, to reduce the number of templates required, Directional Hex
Refinement is used.  Directional Hex Refinement only requires five templates which are
shown above in Figures \ref{fig:5} to \ref{fig:3conc2}.

Directional Hex Refinement refines along the three principle axes in series.  Since hexes
are processed individually in Smart Refinement, new techniques had to be developed to
maintain a conformal mesh and ensure that refinement occurred correctly.  The remainder
of this section discusses the advancements made to refine hexes directionally on a hex by
hex basis.

\subsubsection{The Conformity Problem}

\begin{figure}[htb]
\begin{center}
\subfigure[Refinement
1]{\includegraphics[scale=.4]{conformity1.png}\label{fig:conf1}}
\subfigure[Refinement
2]{\includegraphics[scale=.4]{conformity2.png}\label{fig:conf2}}
\caption{Conformity Issues}
\end{center}\end{figure}

Conformity becomes a real problem for Directional Hex Refinement.  An example of the
conformity problem is shown in the above figure.  Both Figure \ref{fig:conf1} and Figure
\ref{fig:conf2} are split on the left and bottom edges.  Both contain valid refinement
```

schemes yet they are not exactly identical. This problem can occur frequently because each hex is refined independently of its neighbors.

\subsubsection{Ranking System}
To solve the conformity problem, we used the concept of the dual of the mesh discussed previously. This concept was implemented as a ranking system. The ranking system works in the following manner. First, an initial arbitrary edge is selected and given a rank of 1. Next, all other edges that are in the same hex sheet and need to be directionally refined are given the same rank. This process continues until all edges that need to be directionally refined are ranked. Refinement then occurs on a hex by hex basis starting in the direction with the lowest rank and continuing in ranked order until the hex is completely refined.

\subsubsection{Projection Scheme}
After a hex is refined in a principle direction, new edges exist that may need to be split in order that the refinement works properly. To handle this, we developed a projection scheme to determine which edges need to be split. Figure \ref{fig:proj} shows a 2D example of a hex that has been refined in one direction. After the refinement has occurred, the projection scheme checks edges 1 and 4. If both edges are split then it will split edges 2 and 3. If only one or no edges are split then edges 2 and 3 will not be split. This process occurs on the four faces that are not parallel to the initial direction of refinement.

\begin{figure}[htb]
\begin{center}
\includegraphics[scale=.3]{project.png}
\caption{Projection Scheme}\label{fig:proj}
\end{center}\end{figure}

\subsection{Algorithm}
The Smart Refinement algorithm starts by applying the the 1 to 27 template to the target hexes as shown in step \algref{Smart}{target}. The boundary hexes are all that remain after this step. Because Total Hex Refinement is more efficient, it is applied first in step \algref{Smart}{boundary}. The remaining hexes are then ranked as shown in algorithm step \algref{Smart}{rank}. Finally, the remaining hexes are refined directionally from lowest to highest rank.

\begin{algorithm}
\caption{Smart Refinement} \label{Smart}
\begin{algorithmic} [1]
  \Loop{ target hexes} \Comment{Total Hex Refinement}
    \State apply 1 to 27 template to target hex \label{target}
  \EndLoop
        \Loop{ boundary hexes} \label{boundary}
          \If{template applies}
            \State refine hex using template

```
            \Else
              \State add to directional hex list
            \EndIf
          \EndLoop
          \Loop{ directional hex list} \label{rank}
            \State apply ranking system
          \EndLoop
          \Loop{ directional hex list} \Comment{Directional Hex Refinement}
            \State directionally refine hexes
          \EndLoop
\end{algorithmic}
\end{algorithm}

\section{Comparison between Smart Refinement and Old Algorithm}

\begin{figure}[htb]
\begin{center}
\subfigure[Smart
Refinement]{\includegraphics[scale=.3]{concavitynew.png}\label{concnew}}
\subfigure[Old
Algorithm]{\includegraphics[scale=.3]{concavityold.png}\label{concold}}
\caption{Comparison of Concavity Capability}
\end{center}\end{figure}
```

Smart Refinement, because it refines individual hexes, solves the self-intersecting sheet problem. Smart Refinement also is able to handle concavities. Figure \ref{concnew} and Figure \ref{concold} show the results of both algorithms while trying to refine the bottom and left surfaces of a cube. Smart refinement works correctly while the old algorithm excessively refines the whole cube.

Probably the greatest advantage of Smart Refinement over the old algorithm is scalability. It has already been stated that the scalability of the old algorithm was on the order of n^3. The scalability of Smart Refinement is nearly linear. Figure \ref{fig:scale} decisively shows that Smart Refinement out-preforms the old algorithm. The old algorithm was unable to refine anything with an initial element count larger than 100000 elements. Contrast this with Smart Refinement which was able to go up to almost half a million initial elements and would go further if more data was taken.

```
\begin{figure}[htb]
\begin{center}
\includegraphics[scale=.5]{scalability.png}
\caption{Scalability comparison}\label{fig:scale}
\end{center}\end{figure}
```

Smart Refinement has become powerful mesh modification tool. While further testing is needed, the data given in this section supports this conclusion. Smart Refinement can

handle self-intersecting hex sheets and concavities. The scalability is nearly linear and by modifications to directional refinement theory, Smart Refinement is able to maintain conformity. For these reasons, Smart Refinement should replace the old refinement algorithm.

\section{Application to Nanoscience and Nanotechnology}
Continuum modeling is very important to nanoscience. With the current computational power available, it is impossible to model a system atomistically for more than a few moments. New areas of research and modeling are attempting to model on the continuum and atomistic levels simultaneously to increase modeling time. To use both effectively, a intimate relationship must exist between the two levels to ensure that the analysis is accurate. Usually in the continuum-atomistic interface, a continuum node must also represent an atom. Smart Refinement would effectively allow the interface between continuum and atomistic to be sufficiently small to represent both a node on the continuum level and an atom on the atomistic level. Other applications are also highly probable once the capabilities of Smart Refinement are known.

\section{Conclusion}
Smart Refinement was developed to solve problems with old theory and implementation. It can handle self-intersecting hex sheets and concavities. The scalability of Smart Refinement is linear which is a dramatic improvement over cubic scalability found in the old algorithm. With it's increased capability and scalability, Smart Refinement is a powerful mesh modification tool. Smart Refinement will allow increased accuracy in the analysis of target areas.

\bibliographystyle{siam}
\bibliographystyle{siam}
\begin{thebibliography}{1}

\bibitem{bib2}
{\sc N.~Harris}, {\em Conformal Refinement of
All-Hexahedral Finite Element Meshes}, M.S. thesis,
Brigham Young University, Provo, UT, 2004.

\bibitem{dual}
{\sc P.~Murdoch, and S.~Benzley}, {\em The
spatial twist continuum}, in Proceedings, 4th
International Meshing Roundtable, pp.~243--252.

\bibitem{bib1}
{\sc S.~Benzley, N.~Harris, M.~Scott, M.~Borden, and S.~Owen}, {\em Conformal
Refinement and Coarsening of Unstructured Hexahedral Meshes}, Journal of
Computing
and Information Science in Engineering, 5 (2005), pp.~330--337.

\end{thebibliography}

CUBIT FILES

```
reset
bri x 10
mesh vert all
create node location 5 -2 5 owner vol 1
create node location 5 2 5 owner vol 1
create node location -5 -2 5 owner vol 1
create node location -5 2 5 owner vol 1
create node location 5 -2 -5 owner vol 1
create node location 5 2 -5 owner vol 1
create node location -5 -2 -5 owner vol 1
create node location -5 2 -5 owner vol 1

create hex node 4 1 6 7 11 9 13 15 owner volume 1
create hex node 11 9 13 15 12 10 14 16 owner volume 1
create hex node 12 10 14 16 3 2 5 8 owner volume 1

draw hex all

reset
bri x 10
mesh vert all
create node location 4 -2 4 owner vol 1
create node location 4 2 4 owner vol 1
create node location -5 -2 5 owner vol 1
create node location -5 2 5 owner vol 1
create node location 5 -2 -5 owner vol 1
create node location 5 2 -5 owner vol 1
create node location -4 -2 -4 owner vol 1
create node location -4 2 -4 owner vol 1

create hex node 4 1 6 7 11 9 13 15 owner volume 1
create hex node 11 9 13 15 12 10 14 16 owner volume 1
create hex node 12 10 14 16 3 2 5 8 owner volume 1

draw hex all

reset
bri x 10
mesh vert all
create node location 5 -2 5 owner vol 1
create node location 5 2 5 owner vol 1
create node location 0 -2 5 owner vol 1
create node location 0 2 5 owner vol 1
create node location 5 -2 0 owner vol 1
create node location 5 2 0 owner vol 1
create node location 0 -2 0 owner vol 1
create node location 0 2 0 owner vol 1
create hex node 9  13  15  11  10  14  16  12 owner volume 1
create hex node 10  14  16  12  2  5  8 3 owner volume 1
create hex node 1  6  7  4  9  13  15  11 owner volume 1
create hex node 4  11  15  7  3  12  16  8 owner volume 1
create hex node 6  7  15  13  5  8  16  14 owner volume 1
draw hex all
```

```
reset

create cylinder height 500  radius 150
create cylinder height 100  radius 50
move surface 5 location surface 2 midpoint nomerge
imprint all
delete volume 2
move surface 8 location x 0 y 0 z 0 nomerge
rotate volume 1  about X angle -90

webcut volume 1 with plane yplane offset 100 noimprint nomerge
webcut volume 3 with plane yplane offset 200 noimprint nomerge
imprint all
merge all

surface 8 size 8
surface 8 scheme circle fraction 0.5
mesh surface 8
surface 8  smooth scheme centroid area pull
smooth surface 8

surface 7 size 15
surface 7 scheme hole rad_intervals 15 bias -1.5
mesh surface 7


Volume 1  size 5
Volume 3  size 15
Volume 4  size 30
Volume 1 3 4 scheme Sweep
mesh Volume 1
mesh volume 3
mesh volume 4

# block, sidesets
block 1 volume 1 3 4
Sideset 1 surface 8
Sideset 2 surface 7
Sideset 3 surface 16 14 10 3

export mesh "./1m_coarse.g" overwrite

group "entry" add hex with x_coord < 3 and x_coord > -3 and y_coord <
70 and z_coord < 3 and z_coord > -3
refine hex in entry radius 60 no_smooth
export mesh "./1m_refine1.g" overwrite

delete group entry
group "entry" add hex with x_coord < 2 and x_coord > -2 and y_coord <
20  and z_coord < 2 and z_coord > -2
refine hex in entry radius 55
export mesh "./1m_refine2.g" overwrite

##I had to create a brick because I couldn't create
##nodes without assigning an owner.
brick x 10
```

```
mesh vert all
create node location -2 0 5 owner Surface 1
create node location 2 0 5 owner Surface 1
create node location 2 0 -5 owner Surface 2
create node location -2 0 -5 owner Surface 2
create hex node 2 10 9 3 5 11 12 8 owner volume 1
create node location 5 0 2 owner Surface 6
create node location 5 0 -2 owner Surface 6
create node location -5 0 -2 owner Surface 4
create node location -5 0 2 owner Surface 4
create node location 2 -2 2 owner Volume 1
create node location 2 -2 -2 owner Volume 1
create node location -2 -2 -2 owner Volume 1
create node location -2 -2 2 owner Volume 1
create hex node 5  14  13  2  11  18  17  10  owner volume 1
create hex node 11  18  17  10  12  19  20  9  owner volume 1
create hex node 3  16  15  8  9  20  19  12  owner volume 1
create node location 2 -5 2 owner Surface 3
create node location 2 -5 -2 owner Surface 3
create node location -2 -5 -2 owner Surface 3
create node location -2 -5 2 owner Surface 3
create node location -2 5 5 owner Curve 4
create node location 2 5 5 owner Curve 4
create node location -2 -5 5 owner Curve 4
create node location 2 -5 5 owner Curve 4
create node location 5 -5 -2 owner Curve 10
create node location 5 -5 2 owner Curve 10
create node location 2 -5 -5 owner Curve 6
create node location -2 -5 -5 owner Curve 6
create node location -5 -5 -2 owner Curve 9
create node location -5 -5 2 owner Curve 9
create hex node 12  32  31  11  19  23  22  18  owner volume 1
create hex node 17  21  24  20  18  22  23  19  owner volume 1
create hex node 10  28  27  9  17  21  24  20  owner volume 1
create hex node 14  29  30  13  18  22  21  17  owner volume 1
create hex node 16  34  33  15  20  24  23  19  owner volume 1
create hex node 1  30  21  28  2  13  17  10  owner volume 1
create hex node 6  31  22  29  5  11  18  14  owner volume 1
create hex node 7  33  23  32  8  15  19  12  owner volume 1
create hex node 4  27  24  34  3  9  20  16  owner volume 1

#This command changes the way hexes are drawn.  A shrink value of 0 is
the
#default and will draw the hexes at their actual size.  Using a larger
value
#shrinks the size of the hex when it is drawn and allows you to see all
the
#hexes and how they fit together.
graph shrink .5
draw hex all

reset
bri x 10
mesh vert all
create node location 5 -2 5 owner vol 1
create node location 5 2 5 owner vol 1
create node location -5 -2 5 owner vol 1
```

```
create node location -5 2 5 owner vol 1
create node location 5 -2 -5 owner vol 1
create node location 5 2 -5 owner vol 1
create node location -4 -2 -4 owner vol 1
create node location -4 2 -4 owner vol 1

create hex node 4 1 6 7 11 9 13 15 owner volume 1
create hex node 11 9 13 15 12 10 14 16 owner volume 1
create hex node 12 10 14 16 3 2 5 8 owner volume 1

draw hex all

reset
bri x 10
mesh vert all
create node location 5 -2 5 owner vol 1
create node location 5 2 5 owner vol 1
create node location -5 -2 5 owner vol 1
create node location -5 2 5 owner vol 1
create node location 5 -2 0 owner vol 1
create node location 5 2 0 owner vol 1
create node location -5 -2 0 owner vol 1
create node location -5 2 0 owner vol 1

create hex node 4 1 6 7 11 9 13 15 owner volume 1
create hex node 11 9 13 15 12 10 14 16 owner volume 1
create hex node 12 10 14 16 3 2 5 8 owner volume 1
create hex node 15 13 6 7 16 14 5 8 owner volume 1

draw hex all

reset
bri x 10
mesh vert all
create node location 5 -2 5 owner vol 1
create node location 5 2 5 owner vol 1
create node location -5 -2 5 owner vol 1
create node location -5 2 5 owner vol 1
create node location 5 -2 -5 owner vol 1
create node location 5 2 -5 owner vol 1
create node location -5 -2 -5 owner vol 1
create node location -5 2 -5 owner vol 1

create node location 2 5 5 owner vol 1
create node location -2 5 5 owner vol 1
create node location 2 -5 5 owner vol 1
create node location -2 -5 5 owner vol 1
create node location 2 5 -5 owner vol 1
create node location -2 5 -5 owner vol 1
create node location 2 -5 -5 owner vol 1
create node location -2 -5 -5 owner vol 1

create node location 2 2 5 owner vol 1
create node location 2 -2 5 owner vol 1
create node location -2 2 5 owner vol 1
create node location -2 -2 5 owner vol 1
create node location 2 2 -5 owner vol 1
```

```
create node location 2 -2 -5 owner vol 1
create node location -2 2 -5 owner vol 1
create node location -2 -2 -5 owner vol 1

create hex node 3 18 27 12 8 22 31 16 owner volume 1
create hex node 18 17 25 27 22 21 29 31 owner volume 1
create hex node 17 2 10 25 21 5 14 29 owner volume 1
create hex node 12 27 28 11 16 31 32 15 owner volume 1
create hex node 27 25 26 28 31 29 30 32 owner volume 1
create hex node 25 10 9 26 29 14 13 30 owner volume 1
create hex node 11 28 20 4 15 32 24 7 owner volume 1
create hex node 28 26 19 20 32 30 23 24 owner volume 1
create hex node 26 9 1 19 30 13 6 23 owner volume 1


reset
bri x 10
mesh vert all
create node location 5 -2 5 owner vol 1
create node location 5 2 5 owner vol 1
create node location -5 -2 5 owner vol 1
create node location -5 2 5 owner vol 1
create node location 5 -2 -5 owner vol 1
create node location 5 2 -5 owner vol 1
create node location -4 -2 -4 owner vol 1
create node location -4 2 -4 owner vol 1

create node location 0 -2 -5 owner vol 1
create node location 0 2 -5 owner vol 1
create node location -5 -2 0 owner vol 1
create node location -5 2 0 owner vol 1

create node location -2 2 -2 owner vol 1
create node location 2 2 -2 owner vol 1
create node location 2 2 2 owner vol 1
create node location -2 2 2 owner vol 1

create node location -2 -2 -2 owner vol 1
create node location 2 -2 -2 owner vol 1
create node location 2 -2 2 owner vol 1
create node location -2 -2 2 owner vol 1


create hex node 4 1 6 7 11 9 13 15 owner volume 1
create hex node 11 9 13 15 12 10 14 16 owner volume 1
create hex node 12 10 14 16 3 2 5 8 owner volume 1

draw hex all
```