N2, 2000

Использование структуры универсального конечного элемента при разработке моделей в рамках программы "FEM models"

К.Г.Шашкин

Метод конечных элементов (МКЭ) позволяет эффективно решать самые различные задачи в области расчетов строительных конструкций. При этом он допускает введение расчетных схем сложной геометрической конфигурации, назначение любых граничных условий, введение в расчет элементов с различными характеристиками или даже описываемых разными математическими моделями. Однако применение сложных конечно-элементных моделей на практике сдерживается тем, что в существующих расчетных программах круг моделей достаточно ограничен, а введения новых моделей либо вообще не предусматривается, либо представляет значительную сложность. Отличие нового программного комплекса "FEM models" заключается в том, что он представляет собой не только программу для практических расчетов, но и удобную среду, специально предназначенную для разработки новых конечно-элементных моделей.

При решении задач МКЭ существует большое число процедур, которые являются общими практически для любых конечно-элементных моделей. Такими процедурами являются создание конечно-элементных сеток, формирование разрешающей матрицы системы из матриц отдельных конечных элементов, решение системы линейных алгебраических уравнений, визуализация результатов решения (в виде эпюр, графиков, изолиний и т.д.). Программа "FEM models" предоставляет разработчику возможность сосредоточить свое внимание непосредственно на создании самой модели, используя для решения указанных общих задач уже готовые процедуры.

При разработке конечно-элементных моделей чрезвычайно удобными оказываются приемы объектно-ориентированного программирования. Такие приемы в настоящее время являются наиболее прогрессивными и используются в самых разных языках программирования. Объектом в терминах объектно-ориентированного программирования называется совокупность данных и процедур (методов) для работы с ними. Можно заметить, что конечный элемент удобно укладывается в это понятие: действительно, каждый элемент имеет некоторое количество заданных параметров (физические характеристики, координаты и т.п.) и должен содержать процедуры, выполняющие определенные действия с этими параметрами (например, вычисление матрицы жесткости, построение изображения элемента на экране и т.д.). Однако для обеспечения взаимодействия элементов, созданных различными разработчиками необходим некоторый стандарт конечного элемента, который бы позволял реализовывать самые разные модели и в то же время был достаточно прост для понимания и удобен в использовании. С этой целью при создании программы "FEM models" была разработана *структура универсального конечного элемента*, которая в рамках программы и выполняет роль такого стандарта.

Рассматривая самые различные задачи, решаемые с помощью МКЭ можно заметить, что всякий конечный элемент имеет некоторое количество координат, определяющих его положение в пространстве и времени, и некоторое количество параметров, определяющих его свойства. Элемент может участвовать в расчете, внося изменения как в левую часть разрешающей системы уравнений (добавляя матрицу элемента в разрешающую матрицу системы), так и в правую часть (собственный вес, "фиктивные" силы, используемые для решения нелинейных задач и т.д.). Кроме того, решение многих задач предполагает исключение некоторых степеней свободы (например, перемещений в каком-либо

направлении), поэтому в такой элемент логично ввести закрепление соответствующих степеней свободы в узлах. Учитывая это, представляется нецелесообразным выделять нагрузки или закрепления в отдельные объекты, поскольку они удобно описываются некоторым элементом, который при расчете либо вносит изменение только в правую часть разрешающей системы уравнений (силы), либо определяет заданные значения вектора решения (закрепления, заданные перемещения). *Элементами* в такой интерпретации являются любые объекты, которые могут участвовать в расчете или даже просто быть изображенными на схеме. Последние полезны для создания понятных и легко читаемых схем. Элементы, участвующие в расчете, должны быть также способны по полученному вектору решения вычислять результаты (например напряжения) и выдавать эти результаты для построения графиков и изолиний.

С точки зрения программиста, каждый элемент в рамках программы "FEM models" представляет собой COM-объект. Технология COM (Component Object Model), получившая в своем дальнейшем развитии название ActiveX, была разработана фирмой Microsoft и в настоящее время широко используется в программировании [1]. Это определенный набор стандартов, который позволяет совместно работать частям программы, написанным разными разработчиками на различных языках программирования.. Использование данной технологии позволяет создавать конечно-элементные модели с использованием любого современного языка программирования (C++, Delphi, Visual Basic и др.). Кроме того в программе "FEM models" предусмотрена *интегрированная среда разработчика* конечно-элементных моделей, позволяющая создавать модели непосредственно на математическом языке, при этом навыков в области программирования от разработчика практически не требуется. Знание программирования может понадобиться только при создании достаточно сложных конечных элементов, например, суперэлементов или для решения задач с перестроением конечно-элементной сетки в процессе решения. Поэтому в данной статье мы не будем касаться деталей программирования, а остановимся на основных принципах построения структуры универсального конечного элемента.

Структура универсального конечного элемента состоит из нескольких групп процедур, определяющих свойства элемента. В терминологии COM-технологии эти группы называются *интерфейсами*. Элемент может поддерживать часть интерфейсов (например, чисто упругий элемент может не поддерживать интерфейсы, обеспечивающие решение нелинейных задач). При решении задач процедуры решения "опрашивают" все элементы схемы, получая от них значения параметров, блоков матриц жесткости и т.д. При этом запрашиваются, естественно, процедуры только тех интерфейсов, которые поддерживает данный элемент.

Базовым интерфейсом, который должны поддерживать все элементы, работающие в программе "FEM models" является интерфейс **IElement** (названия интерфейсов принято начинать с буквы ***I***). В нем определяются параметры, описывающие свойства элемента (цвет изображения на экране, физические параметры и т.д.), а также координаты элемента в пространстве и времени. Время существования элемента может использоваться для решения динамических и нестационарных задач и определяется по общей для всей схемы шкале времени. Каждый элемент может иметь начало и конец существования. Кроме того, в этом интерфейсе содержится возможность доступа ко всей схеме (списку элементов), таким образом становится возможным изменение схемы в процессе решения. В интерфейсе содержатся также процедуры проверки допустимости заданных параметров и координат, что поможет пользователю избегать грубых ошибок (например, для четырехугольного элемента плиты целесообразно проверить, лежат ли все 4 узла в одной плоскости).

Вторым интерфейсом, являющимся обязательным для любого элемента, является **IDraw**, который определяет способ изображения элемента на экране. Изображение может строиться из любого количества отрезков и треугольных граней различного цвета. Грани и отрезки строятся по точкам, которые могут совпадать или не совпадать с координатами элемента. Из треугольных граней можно "собрать" практически любую фигуру, при этом можно отменить

прорисовку ненужных ребер (например, четырехугольник можно составить из двух треугольников, отменив рисование диагонали). Данные о гранях и отрезках, изображающих элемент, используются процедурами, строящими пространственное изображение схемы с учетом собственных теней и удаления невидимых линий.

Приведенные интерфейсы достаточны для описания различных "графических" элементов, не принимающих участия в решении.

Например, описание объемного тетраэдра должно содержать следующую информацию:

Интерфейс IElement:

параметры - цвет элемента;

количество координат - 4;

Интерфейс IDraw:

количество точек для рисования - 4;

координаты точек совпадают с координатами элемента;

количество граней - 4;

номера точек граней (1,2,3), (1,3,4), (1,4,2), (2,3,4);

количество линий 0.

Следующие интерфейсы определяют, каким образом элемент принимает участие в расчете.

Интерфейс **INodes** определяет количество узлов, принадлежащих элементу и количество степеней свободы в каждом узле. Элемент может иметь любое количество узлов. Любой элемент, участвующий в расчете, должен поддерживать данный интерфейс.

Интерфейс **INodeVars** определяет количество степеней свободы в каждом узле. Этот интерфейс используется для элементов, имеющих матрицу жесткости и вносящих изменения в левую часть разрешающей системы уравнений. Каждая степень свободы (перемещение, угол поворота, температура и т.п.) имеет свой уникальный идентификатор. При составлении разрешающей матрицы системы суммируются элементы матриц конечных элементов, соответствующие степеням свободы с одинаковыми идентификаторами. Интерфейс INodeVars используется совместно с описанным далее интерфейсом IMatrix, так как в противном случае на главной диагонали разрешающей матрицы системы могут появиться нулевые элементы.

Интерфейс **IFreeTerms** определяет воздействие на узел сетки, т.е. добавку в правую часть разрешающей системы уравнений в каждом узле элемента. Добавка (например сила, момент) имеет идентификатор, соответствующий степени свободы, в направлении которой действует данное воздействие. Например в методе перемещений сила, действующая в направлении оси *x* имеет тот же идентификатор, что и перемещение по направлению оси *x*. При решении системы уравнений воздействие оказывают только те силы, моменты и т.п., которые действуют в направлении имеющихся в системе степеней свободы, т.е. если данный узел принадлежит хотя бы одному элементу, имеющему описание данной степени свободы в интерфейсе INodeVars.

Интерфейс **IDefinedVars** определяет заданные значения вектора решения системы. Это могут быть заданные перемещения и закрепления (заданные нулевые перемещения). Они имеют те

же идентификаторы, что и степени свободы. Как и в предыдущем интерфейсе, воздействие оказывают только те заданные значения, которые соответствуют имеющимся в системе степеням свободы.

Некоторое усложнение структуры конечного элемента при введении идентификаторов степеней свободы позволяет совмещать в одной схеме самые разные модели, например, чисто упругие с моделями термоупругости, упругие задачи с реологическими задачами и т.д.

Интерфейс **IMatrix** обеспечивает одну единственную функцию - выдачу матрицы элемента. Матрица выдается по блокам, соответствующим узлам элемента. Матрица может быть и несимметричной (что иногда встречается, например, при использовании смешанного метода расчета конструкций). В этом случае устанавливается соответствующий признак.

Интерфейс **IIterations** используется для решения нелинейных задач. Его единственная функция LookResults вызывается после решения системы и должна по полученным результатам определять, требуются ли дальнейшие итерации для обеспечения точности решения, при этом на следующей итерации может быть изменена правая или левая часть разрешающей системы. Возможно также перестроение схемы в процессе решения.

Результаты решения, которые выдает каждый элемент определены в интерфейсе **IResults**. При этом элемент имеет доступ к вектору решения системы и может выдавать результаты непосредственно из него, а также вычислять по вектору решения и выдавать дополнительные результаты (например, напряжения в элементе). Поскольку результаты должны изображаться на экране, они выдаются в тех точках, которые определены в интерфейсе IDraw. Кроме того, в интерфейсе IResults определяются те значения, которые не могут быть вычислены по вектору решения и которые необходимо хранить в файле (например, напряжения на данном шаге решения при решении нелинейных задач).

Рассмотрим пример описания треугольного конечного элемента для решения задачи плоской упругой деформации.

Интерфейс IElement:

количество параметров - 3;

параметры - цвет изображения на экране, модуль упругости, коэффициент Пуассона, собственный вес;

количество координат - 3;

Интерфейс IDraw:

количество точек - 3;

координаты точек соответствуют координатам элемента;

количество граней - 1;

номера точек грани - (1,2,3);

количество линий - 0;

Интерфейс INodes:

количество узлов - 3;

координаты узлов соответствуют координатам элемента;

Интерфейс INodeVars:

количество степеней свободы во всех узлах - 2;

степени свободы в узлах - перемещение по x и по y;

Интерфейс IFreeTerms:

количество добавок в вектор сил - по 1 силе в каждом узле (собственный вес элемента)

направление силы - по оси y;

величина силы в каждом узле $F_y = \frac{\gamma\Delta}{3}$, *где* γ *- собственный вес,* Δ *- площадь элемента.*

Собственный вес прикладывется на шаге решения, соответствующем началу существования элемента. В конце существования элемента с обратным знаком должны быть приложены узловые силы, вычисляемые по формуле $\{F\} = \Delta[B]^T\{\sigma\}$, *где* $[B]$ *- матрица производных от функций формы,* Δ *- площадь элемента,* $\{\sigma\}$ *- напряжения в элементе[2].*

Интерфейс IMatrix:

блоки матрицы жесткости элемента;

Матрица жесткости вычисляется по формуле $K = \Delta[B]^T[D][B]$, *где* $[D]$ *- матрица упругих характеристик.*

Интерфейс IResults:

количество результатов в каждом узле - 5;

результаты - перемещения по x и по y, напряжения $\sigma_x, \sigma_y, \tau_{xy}$.

Величины перемещений берутся из вектора решения, напряжения вычисляются по формуле $\{\sigma\} = [D][B]\{\delta\}$, *где* $\{\delta\}$ *- перемещения узлов элемента. Напряжения считаются постоянными в пределах данного конечного элемента. Сохранять какие-либо данные в файле для этого элемента не нужно, так как напряжения легко вычисляются по вектору решения.*

Модель упруго-хрупкого элемента может быть легко создана из описанной модели путем добавления интерфейса IIterations. Пусть при приложении нагрузок по шагам на некотором шаге в элементе возникли напряжения, превышающие предел прочности. Тогда элемент на данном шаге прекращает свое существование, т.е. разрушается, а в его узлы прикладываются силы, вычисляемые в интерфейсе IFreeTerms при окончании существования элемента, после

чего производится новая итерация.

Аналогичным образом с помощью структуры универсального конечного элемента можно достаточно просто и быстро создавать конечно-элементные модели любой степени сложности. Кроме облегчения труда разработчика моделей структура позволяет также упростить изучение созданных моделей, поскольку четко определяет место описания каждого конкретного свойства модели в тексте программы. Все это, на наш взгляд, будет способствовать созданию новых эффективных конечно-элементных моделей и успешному применению их в практических расчетах.

Литература

1. Дэвид Чеппел. Технологии ActiveX и OLE/ пер. с англ. - М.: Издательский отдел "Русская редакция" ТОО "Channel Trading Ltd.", 1997.

2. Фадеев А.Б. Метод конечных элементов в геомеханике. - М.: Недра,1987