УДК 519.6

СТРУКТУРЫ ДАННЫХ МЕТОДИКИ "ТИМ" РЕШЕНИЯ ДВУМЕРНЫХ И ТРЕХМЕРНЫХ ЗАДАЧ МЕХАНИКИ СПЛОШНОЙ СРЕДЫ

А. А. Воропинов
(ФГУП "РФЯЦ-ВНИИЭФ", г. Саров Нижегородской области)

Рассматривается подход на основе статического полиморфизма к организации структур данных с использованием механизмов наследования языка программирования Фортран 2003. Преимуществом описанного подхода является возможность одновременного использования как старых конструкций, без механизма наследования, так и новых, через конструкции с полиморфизмом.

Описанный подход применен в программной реализации структур данных методики ТИМ, предназначенной для решения двумерных и трехмерных задач механики сплошной среды на неструктурированных лагранжевых сетках. Использование статического полиморфизма позволило реализовать единую структуру данных для двумерного и трехмерного случаев.

Ключевые слова: неструктурированная сетка, структура данных, полиморфизм, Фортран 2003.

Введение

Методика ТИМ предназначена для расчета нестационарных многомерных задач механики сплошной среды на лагранжевых неструктурированных сетках произвольного вида. По этой методике могут проводиться расчеты двумерных задач (ТИМ-2D [1]) в цилиндрической и декартовой системах координат и трехмерных задач (ТИМ-3D [2]) в декартовой системе координат. В настоящее время в методике ТИМ реализованы приближения для расчета нескольких классов задач: газовой динамики, нестационарной упругопластичности (УП), магнитной гидродинамики (МГД), теплопроводности, двухтемпературности и многопотоковой газовой динамики. При решении сложных задач начальную геометрию системы приходится разбивать на математические области. Такое разбиение бывает необходимо для более точного описания взаимодействующих тел с выделенными поверхностями скольжения. Между математическими областями решается задача контактного взаимодействия [3, 4].

В методике используется метод трехуровневого распараллеливания в смешанной модели памяти [5]. На первых двух уровнях применяется принцип декомпозиции по пространству в модели распределенной памяти с использованием интерфейса передачи сообщений MPI. На первом уровне осуществляется распараллеливание счета по математическим областям, на втором — распараллеливание счета в математической области с разделением на параобласти (мелкозернистое распараллеливание). На третьем уровне выполняется распараллеливание итераций счетных циклов в модели общей памяти с использованием интерфейса OpenMP.

Первоначально программы, реализующие методики ТИМ-2D и ТИМ-3D, развивались независимо друг от друга. При этом они имели целый ряд близких алгоритмов с небольшими различиями. Это стало возможным, в частности, благодаря использованию близких форматов представления неструктурированных сеток: *ячеечно-реберного* для двумерной сетки [6] и формата *по граням* для

трехмерной сетки [7]. С другой стороны, независимая реализация программ приводила к необходимости дублирования вносимых изменений и выполнения операций синхронизации версий программ. В конечном счете было решено максимально унифицировать тексты программ, оставив различия только в тех алгоритмах, которые неодинаковы для двумерного и трехмерного случаев.

Унификация программ в первую очередь требует создания единой структуры данных. Поскольку фактически необходимо было объединить уже существующие достаточно большие программы, то актуальным стал вопрос об их преемственности. Для решения этой проблемы был разработан подход статического полиморфизма, использующий конструкции языка Фортран 2003 и позволяющий организовать единую структуру данных, учитывающую особенности каждой из программ.

С точки зрения организации структуры данных наиболее важными характеристиками методики ТИМ являются:

- использование неструктурированной сетки произвольного вида;
- широкий набор моделируемых приближений, каждое из которых имеет свои расчетные величины;
- разбиение на математические области, в каждой из которых используется своя, независимая от других областей, сетка.

1. Подходы к разработке двумерных и трехмерных программ

Можно выделить два основных направления, которые применяются при разработке прикладных программ математического моделирования на основе единой структуры данных, имеющих различия в алгоритмах для двумерного и трехмерного случаев:

1. Раздельная реализация программ для двумерного и трехмерного случаев (*двумерной* и *трехмерной* программ). Данный подход обеспечивает независимое развитие двух программ, облегчает их первоначальную разработку. Проблемой при таком подходе является сложность сопровождения программ. Как правило, перенос алгоритмов делается только один раз — при начале разработки трехмерной программы. В дальнейшем две программы развиваются независимо, и обычно при этом постепенно накапливаются различия. Синхронизация двумерной и трехмерной программ, как правило, приводит к повторной реализации отличающихся частей. Данный подход хорошо использовать, когда одна из программ (как правило, двумерная) фактически перестает развиваться. В частности, он применялся при разработке двумерной методики ДМК [8] и ее трехмерного обобщения — методики ТМК [9].
2. Трехмерная программа "поглощает" двумерную, т. е. в программе для трехмерного случая предусматривается возможность проведения и двумерных расчетов. Такой подход, в частности, хорош для методик, использующих структурированные сетки [10—12], когда в трехмерном случае вводится разделение на листы, а расчет двумерных задач проводится с использованием одного листа.

В методике ТИМ, по которой рассчитываются задачи на неструктурированных сетках, нет разделения на листы, поэтому второй вариант с листовым обобщением неприменим. Вместо него был использован другой вариант "поглощения", связанный с реализацией ветвлений в алгоритмах, зависящих от размерности. Недостатком этого варианта является усложнение программной реализации.

При этом было велико желание создать полностью новую структуру данных, удовлетворяющую всем требованиям на основе подходов объектно-ориентированного программирования. Однако на практике приходилось обеспечивать обратную совместимость с программами, разработанными раньше, часто много лет назад; полная переработка всех программ потребовала бы чрезмерных усилий.

Как было сказано, методики ТИМ-2D и ТИМ-3D долгое время существовали независимо, и для них уже был реализован широкий набор алгоритмов на Фортране 90. Поэтому естественно, что при реализации новой структуры данных, объединяющей функционалы двух программ, был использован язык программирования Фортран 2003. Это, с одной стороны, обеспечило преемственность с ранее созданными программами, а с другой стороны, позволило решить проблему одновременной разработки двумерной и трехмерной программ.

2. Наследование и полиморфизм в Фортране 2003

Организация структур данных неразрывно связана с языком программирования, используемым для реализации программ математического моделирования. Одно из ведущих мест среди таких языков занимает Фортран. Язык Фортран активно развивается [13], регулярно принимаются новые версии стандарта, получающие номер по году их принятия. Современными версиями языка являются Фортран 90 [14], Фортран 95 [15], Фортран 2003 [16] и текущая версия Фортран 2008 [17].

Одной из новых возможностей в стандарте Фортрана 2003 является использование всех механизмов объектно-ориентированного программирования, в частности механизмов наследования и полиморфизма. Необходимо отметить, что основные положения объектно-ориентированного программирования были введены уже в Фортране 90, однако наследование и полиморфизм реализовывались в нем не полностью. Подробное описание инструментов объектно-ориентированного программирования при помощи Фортрана 90 приведено, например, в работе [18].

В Фортране 2003 программист может описать базовый производный тип и несколько типов (типов-расширений), расширяющих базовый путем добавления дополнительных компонентов. В процессе выполнения в рамках одной программы можно обрабатывать все данные типы при помощи ряда дополнительных конструкций. То есть появляется возможность обработки разнородных данных одной процедурой, в то время как в Фортране 90 для каждого типа нужен был свой вариант процедуры.

Такой подход представляется оптимальным для организации структур данных программ методики ТИМ. Если выделить величины, единые для двумерного и трехмерного случаев, в базовый тип, а данные, соответствующие каждому случаю в отдельности, оформить в качестве типов-расширений, то можно использовать программу с единым текстом для решения как двумерных, так и трехмерных задач. К тому же, что еще более важно, сохраняется возможность применения старых программ, в которых не учитывается полиморфизм. Описание конструкций, используемых для работы с полиморфизмом, можно найти в работах [16, 19].

3. Статический полиморфизм с использованием Фортрана 2003

Средства наследования и полиморфизма Фортрана 2003 существенно расширяют возможности создания универсальных программ, ориентированных на обработку различных контекстов. Однако при прикладном применении эти средства не лишены определенных недостатков. Рассмотрим пути преодоления некоторых из них.

Если программа уже существует, перевести ее на использование полиморфизма довольно непросто. Ситуация усугубляется при объединении нескольких программ на основе единой структуры данных. Главная проблема заключается в описании глобальных переменных и массивов на основе расширяемых типов. Если их сразу описать с использованием лексемы `class`, как это предлагается в стандарте Фортрана 2003, то во многих случаях, где есть обращение к различающимся данным, потребуются конструкции переключения типа (`select type`). В прикладной программе с большим объемом текста сразу сделать такие изменения затруднительно. Желательно иметь возможность постепенного объединения структур данных разных программ. Это можно сделать, используя *статический полиморфизм*, учитываемый на этапе компиляции, а не при выполнении. Для применения статического полиморфизма для производных типов предлагается использовать вспомогательный модуль переключателя типов. Структуры данных на основе типов, использующих механизм наследования, и статического полиморфизма будем называть полиморфными.

Пусть есть две программы, для которых создается единая структура данных. Эти программы имеют близкие структуры данных, но в них содержатся некоторые отличия, связанные, например, с разной размерностью. Для описания некоторого объекта в обеих программах используются типы с одинаковым именем `worktype`. Определения этих типов включают как совпадающие величины, так и величины, характерные для каждого случая. Они содержатся в модулях, предназначенных для описания типов и переменных. При объединении программ создаются три новых типа: базовый `basetype`, который включает величины, одинаковые для разных контекстов (двумерного и трехмерного), и два типа, расширяющих базовый, — `type2d` и `type3d` для различающихся величин

(предположим, что все типы описаны в модуле `modTypes`). Также создается вспомогательный модуль (например, `modSwitchTypes`), содержимое которого различно для объединяемых программ и в котором при помощи переименования имя `worktype` переназначается типу `type2d` или `type3d` в зависимости от программы. Для работы со структурой данных в текстах программ используется вспомогательный модуль (`modSwitchTypes`). Пример подобной взаимосвязи производных типов приведен в разд. 8.

Такой подход обеспечивает переключение между контекстами на этапе компиляции, и поэтому его можно назвать статическим полиморфизмом.

4. Дополнительные инструменты работы с полиморфными структурами данных

Использование полиморфных структур данных требует применения дополнительных конструкций, в первую очередь `select type`. При интенсивном обращении к разнородным данным текст программы становится перегружен ими, что усложняет восприятие текста и сопровождение. Для уменьшения количества конструкций `select type` можно использовать следующие инструменты.

Некоторые операции должны выполняться только в одном из контекстов — двумерном или трехмерном. При этом сами операции могут даже не содержать обращений к различающимся компонентам структур. Применение конструкции `select type` в этом случае не оправдано. Гораздо удобнее ввести дополнительный параметр, с использованием которого переключение контекста записывается в более лаконичной форме с помощью оператора `if`. В программах, реализующих методику ТИМ, для этих целей используется параметр `TIMDIM`, значение которого равно 2 для двумерного контекста и 3 — для трехмерного.

Для уменьшения количества конструкций `select type` также можно использовать указатели на компоненты типов-расширений. Все указатели лучше установить в одном месте, например в начале текста программы. Тогда для переключения контекста достаточно выполнять проверку ассоциированности указателя или проверку параметра, определяющего контекст выполнения программы.

Использование указателей бывает необходимо и при применении самой конструкции `select type`. Дело в том, что в случае статического полиморфизма аргументы, предназначенные для `select type`, не обязательно описаны с применением лексемы `class`, как это требуется. Использование указателя, в описании которого присутствует эта лексема, позволяет исключить данную проблему.

Сочетание операторов `if` и указателей в большинстве случаев позволяет минимизировать количество конструкций `select type` и упрощает текст программы.

Отметим, что в случае существенной разнородности алгоритмов может быть целесообразнее их раздельная реализация для каждого из контекстов. Использование описанных выше инструментов оправдано только при сравнительно небольших отличиях.

5. Организация программ методики ТИМ

С использованием статического полиморфизма постепенно был осуществлен переход на раздельную реализацию расчетных программ методики ТИМ для двумерного и трехмерного случаев, опирающихся на единую структуру данных. При этом структура данных использует одинаковые имена типов и переменных для совпадающих величин и отличается наборами величин, которые присутствуют только в одном из случаев (например, координаты в двумерном случае содержат две компоненты, а в трехмерном — три). Двумерная и трехмерная программы компилируются раздельно, и получаются разные исполняемые коды. При этом тексты процедур, использующие одинаковый набор величин, имеют единую реализацию; процедуры, применяющие разные алгоритмы или наборы величин, реализованы раздельно. Преимуществами данного подхода являются экономичность по требованиям к оперативной памяти, упрощение процедур, предназначенных только для одного из контекстов. Несмотря на некоторое усложнение единых процедур, данный подход используется в качестве основного для методики ТИМ.

При применении раздельной сборки исполняемых файлов для двумерного и трехмерного случаев все процедуры можно разделить на три группы с точки зрения их привязки к размерности задачи:

1) *разнородные* — существенно использующие размерность пространства как в исходных, так и в результирующих данных (например, расчет узловых величин, контактного взаимодействия и др.);
2) *близкие* — использующие данные, которые зависят от размерности пространства, но возвращающие результаты в виде, одинаковом для двумерного и трехмерного случаев (например, вычисление объема ячеек);
3) *единые* — использующие не только одинаковые входные и выходные данные, но и один и тот же алгоритм (например, процедуры уравнений состояния, управляющие программы и др.).

В случае, если обращение к разнородным или близким процедурам осуществляется из единых процедур, им даются одинаковые имена. При этом процедуры обязательно должны иметь одинаковые интерфейсы вызова (т. е. список и типы аргументов). Если разнородные процедуры отличаются интерфейсом вызова, то вызывать из единых процедур их нельзя. В этом случае в именах процедур используются суффиксы, отражающие размерность: 2D и 3D.

В именах файлов, содержащих исходные тексты разнородных или близких процедур, также используется суффикс, зависящий от размерности. Исходные тексты единых процедур для двумерного и трехмерного случаев совпадают и содержатся в одном файле. Изменения, производимые для ТИМ-2D, автоматически отражаются в ТИМ-3D и наоборот. Это достигается за счет специального описания исходных файлов в системе управления версиями [20].

Набор расчетных данных методик ТИМ-2D и ТИМ-3D различен. Поэтому для выполнения единых процедур структура данных организована специальным образом. Всем элементам структур данных и их составным переменным, не зависящим от размерности пространства, даются одинаковые имена. Внутри структур переменные, идентичные (или близкие по смыслу) для двумерного и трехмерного случаев, также имеют одинаковые имена.

6. Объектно-ориентированный подход в организации структуры данных

Использование объектно-ориентированного программирования предоставляет дополнительные возможности, в частности:

1. Производные типы позволяют организовать более удобное представление данных в памяти за счет хранения информации об одном объекте в соседних ячейках памяти. Это несколько облегчает программирование алгоритмов. Кроме того, для методик, использующих неструктурированные сетки, появилась возможность более эффективно применять кэш-память, существующую у большинства современных микропроцессоров.
2. Использование механизма наследования позволяет описать сходные объекты, отличающиеся ограниченным набором компонентов. Это, в свою очередь, позволяет иметь одну программную реализацию для обработки всего набора сходных объектов.
3. С помощью указателей организуется прямой переход между различными объектами (без использования косвенной адресации). Это облегчает программирование алгоритмов и ускоряет выполнение программ.

Исходя из указанных предпосылок, в программах, реализующих методику ТИМ, были выделены объекты двух уровней.

Первый уровень — глобальный. Контактные и внешние границы, особые точки, в которых сходятся несколько границ (тройные точки), образуют сетку, "ячейками" которой являются математические области; роль граней (3D) или ребер (2D) выполняют контактные границы, узлов — тройные точки. *Математическая область*, *контактная граница* и *тройная точка* относятся к объектам глобального уровня.

Зависимости между объектами глобального уровня в графическом виде представлены на рис. 1. На этом и последующих рисунках стрелка обозначает включение одним объектом другого, соединение с точками на концах отображает взаимосвязь объектов. Представленная на рис. 1 зависимость

Рис. 1. Связи между объектами глобального уровня

соответствует двумерному случаю. В трехмерном случае расчет контактного взаимодействия производится без введения тройных точек и использования соответствующего объекта.

Ко второму уровню относятся объекты математической области, соответствующие элементам сетки: *грань* (используется только в ТИМ-3D), *ребро* (используется только в ТИМ-2D), *узел*, *ячейка*.

Все одинаковые объекты второго уровня образуют большой массив — *глобальный контейнер*. Таким образом, имеются отдельные глобальные контейнеры для ячеек, граней/ребер и узлов.

При расчете газовой динамики и УП используется разбиение на математические области и глобальный контейнер содержит информацию для всех областей, рассчитываемых текущим процессом. При инициализации для глобальных контейнеров в памяти отводится столько места, чтобы они могли включать в себя все данные, рассчитываемые текущим процессом по всем областям, с необходимым резервом для добавления новых элементов сетки, если в какой-то области их количество увеличивается.

Однако при моделировании диффузионных приближений разбиение на математические области не производится. Для расчета этих приближений выполняется операция *сшивания* всех областей в одну, а после расчета диффузии производится обратная операция, называемая *расшиванием*. При выполнении сшивания для всех элементов сетки по всем областям устанавливается единая нумерация.

Применение глобальных контейнеров позволяет быстро выполнять операции сшивания и расшивания без использования большого объема дополнительной памяти, а также переходить от нумерации математических областей к единой и обратно.

В зависимости от используемого расчетного приближения объект *ячейка* может дополняться данными УП, детонации, многотемпературности, многопотоковости, МГД, которые также представлены в структуре данных как самостоятельные объекты.

Основные зависимости между объектами второго уровня (математической области) показаны на рис. 2. Пунктирные овалы обозначают глобальные контейнеры.

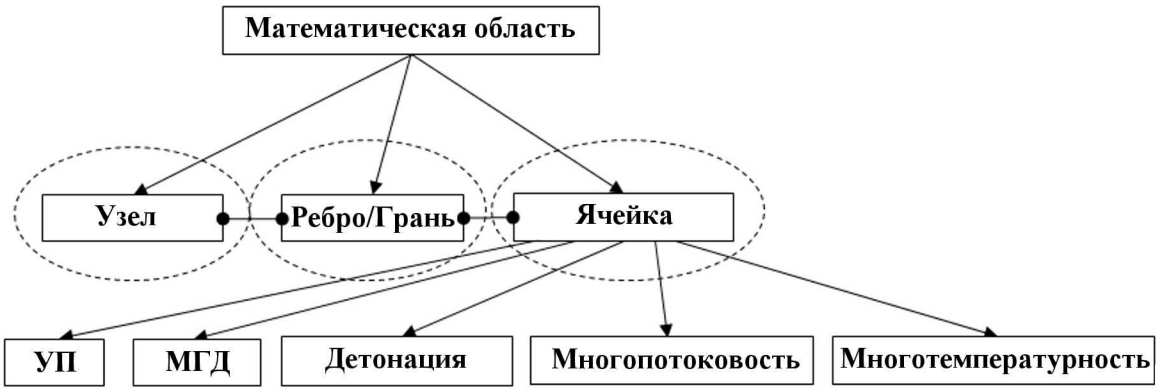

Рис. 2. Связи между объектами второго уровня

В режиме параллельного счета с распараллеливанием по областям и при OpenMP-распараллеливании используется такое же выделение объектов, что и в последовательном режиме счета. Однако при использовании мелкозернистого распараллеливания [21] возникает ряд новых объектов. Чтобы понять их суть, кратко опишем схему инициализации в этом случае.

Для хранения дополнительных величин, необходимых для мелкозернистого распараллеливания математической области, вводится дополнительный объект — *параллельная математическая область*. Первоначально для параллельной математической области выполняется декомпозиция данных по ячейкам сетки. При этом область разбивается на непересекающиеся наборы ячеек, называемые компактами. На основе компактов в рамках параллельной математической области формируются *параобласти*. При этом в параобласти устанавливается новая локальная нумерация элементов сетки, определяется слой наложения между параобластями (наличие наложения или его отсутствие зависит от режима мелкозернистого распараллеливания [21]). Затем формируются *параллельные границы*, содержащие информацию для выполнения обменов между параобластями. Параобласть вместе со своими параллельными границами передается для расчета соответствующему процессу. В структуре данных процесса, рассчитывающего параобласть, она представляется так же, как математическая область, с дополнительной информацией для выполнения обменов.

Таким образом, в режиме мелкозернистого распараллеливания дополнительно вводятся объекты *параллельная математическая область*, *параобласть*, *параллельная граница*. Связи между объектами при мелкозернистом распараллеливании показаны на рис. 3. Пунктирные стрелки указывают на объекты, порождаемые при формировании параобластей.

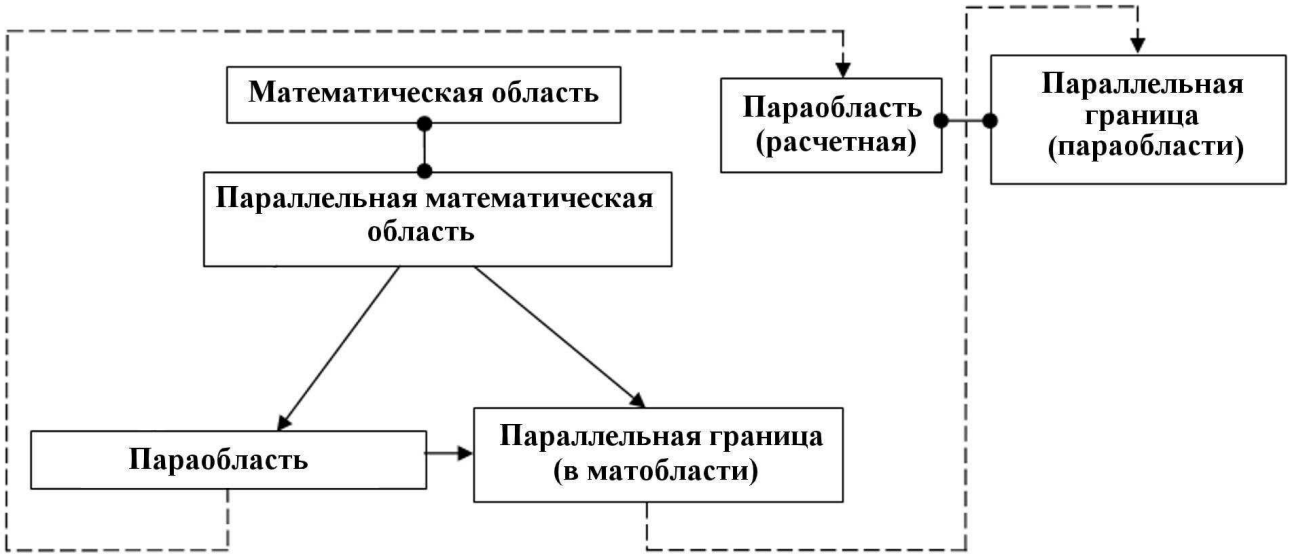

Рис. 3. Связи между объектами при мелкозернистом распараллеливании

7. Описание объектов в программах ТИМ-2D и ТИМ-3D

В программах объекты описываются в виде производных типов Фортрана 2003.

Для описания математической области используются два производных типа: базовый `MathAreaHead` (*шапка математической области*) и расширяющий его — `MathArea` (*математическая область*). В базовом типе `MathAreaHead` сосредоточены общие параметры математической области, используемые как на этапе инициализации, так и в процессе счета. Расширяющий тип `MathArea` используется для описания математической области и параобласти в процессе проведения расчета. В нем описываются все необходимые для расчета переменные и массивы.

Для ссылок на объекты второго уровня в типе `MathArea` используются указатели. Указатели устанавливаются на последовательные сечения глобальных контейнеров (для первой области, для второй и т. д.). При этом не допускается, чтобы сечения имели между собой пересечения или пустоты. Нумерация элементов в выделенном для математической области сечении начинается с единицы, как в обычном массиве. Использование указателей в типе `MathArea` позволяет осуществлять современные методы контроля целостности данных и отладки, например контроль выхода за пределы массивов.

Описания типов для объектов *контактная граница* и *тройная точка*, используемых для контактного взаимодействия, в ТИМ-2D и ТИМ-3D различны из-за существенной разницы в алгоритмах: в двумерном случае это взаимодействие через две ломаные, а в трехмерном — через поверхности.

Объекты математической области, с одной стороны, задают расчетную сетку, а с другой, содержат расчетные данные. Таким образом информация, относящаяся к элементам сетки, бывает двух типов: топологическая, о структуре сетки, и счетная — расчетные величины.

В работах [6, 7] приведено описание формата хранения топологической информации, используемого в методике ТИМ. В соответствии с ним информация о гранях ячейки и узлах грани хранится в виде списков, а для оставшейся части топологии вводятся топологические типы, для разных элементов сетки свои. Таким образом, топологические типы используются для описания объектов *узел* и *грань/ребро*. Расчетные величины для данных объектов описываются счетными типами, которые являются расширениями топологических.

Благодаря представлению объектов двумя типами данные об элементе сетки (узле и грани/ребре) располагаются в памяти рядом, при этом имеется возможность реализации процедур, работающих только с топологией.

Величины, которые рассчитываются в ячейках (ячеечные), разделены на два класса — базовые, всегда присутствующие в расчетах, и дополнительные величины приближений, которые используются при моделировании соответствующих физических процессов. К базовым относятся величины, широко используемые в методике ТИМ. Это набор газодинамических параметров, номер подобласти, уравнения состояния и пробега.

Базовые ячеечные величины содержатся в описании типа `Cell`. Также тип `Cell` содержит указатели на наборы дополнительных величин счетных приближений и включает переменную W рабочего типа `WorkCell`.

Тип `WorkCell` содержит различные ячеечные величины, которые хранятся для передачи информации между счетными блоками и минимизации дополнительных вычислений. Например, при вычислении объема рассчитываются геометрические параметры ячейки. Их получение в других счетных блоках равносильно еще одному обращению к процедуре вычисления объема, являющейся одной из самых дорогостоящих операций при расчете уравнений газовой динамики. Поэтому эти величины сохраняются на протяжении расчета шага, чтобы его ускорить. В то время как тип `Cell` одинаков в двумерном и трехмерном случаях, тип `WorkCell` является контекстно-зависимым (разным для ТИМ-2D и ТИМ 3D).

Набор дополнительных ячеечных величин каждого счетного приближения представлен своим типом. Так, упругопластическому приближению соответствует тип `UPCell`. Он содержит описания всех параметров, необходимых для расчета УП. Вся информация, необходимая для каждой ячейки математической области, рассчитываемой с использованием упругопластического приближения, хранится в элементе динамического массива типа `UPCell`. Размер массива равен количеству таких ячеек, т. е. меньше или равен общему числу ячеек в области. Описание типа `Cell` содержит указатель на элемент массива, соответствующий данной ячейке, и его номер (последний необходим для косвенной адресации). Если при расчете данной ячейки упругопластическое приближение не используется, то указатель не ассоциирован, а номер равен нулю. Возможности расчетов с использованием других приближений реализованы аналогично.

Для узловых величин применяется схема, аналогичная схеме хранения ячеечных величин с разделением на рабочие и основные. Для узлов не требуется хранения дополнительных величин для приближений, поэтому дополнительные узловые типы не вводятся.

8. Переключение контекстов

Компиляция текстов программ производится независимо для ТИМ-2D и ТИМ-3D, при этом для единых программ выполняется активация соответствующего набора типов. Для этого используется специальное описание типов на основе механизма наследования Фортрана 2003 и дополнительный модуль для переключения их наборов в зависимости от контекста. Описание переключения контекстов рассмотрим на примере дополнительных данных УП.

Для создания единых программ для двумерного и трехмерного случаев используется тип `UPCell` (см. разд. 7). Его описание состоит из базового типа `UPCell_Common`, содержащего описание величин, которые используются в обоих этих случаях, и расширяющих его производных типов `UPCell_2D` и `UPCell_3D`, содержащих величины, соответствующие ТИМ-2D и ТИМ-3D в отдельности. В зависимости от контекста происходит переименование в `UPCell` либо типа `UPCell_2D`, либо типа `UPCell_3D`.

Описание указанных типов выглядит следующим образом:

```
type UPCell_Common
  integer:: nModel, nCell, nCellInZUP
  real(8):: SXX,SYY,GLJ,YUP,EUPR,EPLU
end type UPCell_Common

type,extends(UPCell_Common) :: UPCell_2D
  real(8):: TXY
end type UPCell_2D

type,extends(UPCell_Common) :: UPCell_3D
  real(8):: SZZ,SXY,SZX,SYZ
end type UPCell_3D
```

Схема наследования Фортрана 2003 используется также для типов, описывающих дополнительные величины МГД-приближения. Они организованы аналогично типу `UPCell`. Типы остальных объектов для дополнительных данных расчетных приближений содержат одинаковый набор величин в двумерном и трехмерном случаях.

Описание типов для дополнительных данных приближений сосредоточено в модуле `modTIMSubTypes`. Для переключения контекста создан специальный модуль `modTIMSubTypesDD`, содержание которого для ТИМ-2D следующее:

```
module modTIMSubTypesDD
use modTIMSubTypes, UPCell=>UPCell_2D
... ! аналогично и для других типов
end module modTIMSubTypesDD
```

Для ТИМ-3D содержание этого модуля таково:

```
module modTIMSubTypesDD
use modTIMSubTypes, UPCell=>UPCell_3D
... ! аналогично и для других типов
end module modTIMSubTypesDD
```

Таким образом, в расчетных программах при использовании имени `UPCell` становится доступен тип, соответствующий размерности программы.

Заключение

Описан подход к организации единой структуры данных на основе механизмов наследования и полиморфизма языка программирования Фортран 2003, который используется в программной реализации методики ТИМ для расчета двумерных и трехмерных задач механики сплошной среды. В

описанном подходе переключение контекстов осуществляется статически на стадии компиляции, при этом для каждого контекста получается собственный исполняемый код. Использование статического полиморфизма позволяет создавать структуры данных, учитывающие особенности методики, и при этом сочетать с программами, не использующими конструкции Фортрана 2003. Таким образом, полиморфизм можно внедрять в существующие крупные программы поэтапно.

Описаны единые структуры данных методики ТИМ. При описании данных используется объектно-ориентированный подход. Выделены объекты, которые сгруппированы по уровням с установлением взаимосвязей. Структуры данных учитывают все особенности методики: использование неструктурированной сетки произвольного вида, широкий набор моделируемых приближений, разбиение на математические области.

Созданные программы демонстрируют применимость подхода на основе статического полиморфизма Фортрана 2003 при разработке больших кодов для математического моделирования. Использование статического полиморфизма позволило довести объем единого кода в счетных программах методик ТИМ-2D и ТИМ-3D до 71% при общем объеме программ более 1 млн строк.

Список литературы

1. *Соколов С. С., Воропинов А. А., Новиков И. Г. и др.* Методика ТИМ-2D для расчета задач механики сплошной среды на нерегулярных многоугольных сетках с произвольным количеством связей в узлах // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2006. Вып. 4. С. 29—43.
2. *Соколов С. С., Панов А. И., Воропинов А. А. и др.* Методика ТИМ расчета трехмерных задач механики сплошных сред на неструктурированных многогранных лагранжевых сетках // Там же. 2005. Вып. 3. С. 37—52.
3. *Воропинов А. А., Новиков И. Г., Соколов С. С.* Расчет контактного взаимодействия между счетными областями в методике ТИМ-2D // Там же. 2008. Вып. 2. С. 5—20.
4. *Соколов С. С., Софронов И. Д., Рассказова В В. и др.* Решение многомерных задач механики сплошных сред на неструктурированных лагранжевых сетках // Труды РФЯЦ-ВНИИЭФ. 2008. Вып. 13. С. 70—85.
5. *Воропинов А. А., Соколов С. С.* Метод трехуровневого распараллеливания методики ТИМ-2D // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2013. Вып. 4. С. 70—77.
6. *Воропинов А. А.* Некоторые форматы данных для представления двумерных неструктурированных сеток произвольного вида // Там же. 2010. Вып. 4. С. 52—63.
7. *Воропинов А. А., Соколов С. С., Панов А. И., Новиков И. Г.* Формат для описания нерегулярной многогранной сетки произвольной структуры в методике ТИМ // Там же. 2007. Вып. 3. С. 55—63.
8. *Мотлохов В. Н., Рассказова В. В., Шапоренко А. Н.* Лагранжева методика ДМК для решения прикладных задач газовой динамики на нерегулярных сетках // Современные методы проектирования и отработки ракетно-артиллерийского вооружения. Саров: ВНИИЭФ, 2000. С. 57—63.
9. *Рассказова В. В., Мотлохов В. Н., Ерёменко А. Ю., Софронов И. Д.* Методика решения задач трехмерной нестационарной газовой динамики на нерегулярных лагранжевых сетках // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 1998. Вып. 4. С. 44—57.
10. *Бахрах С. М., Величко С. В., Спиридонов В. Ф. и др.* Методика ЛЭГАК-3D расчета трехмерных нестационарных течений многокомпонентной сплошной среды и принципы ее реализации на многопроцессорных ЭВМ с распределенной памятью // Там же. 2004. Вып. 4. С. 41—50.

11. *Воронин Б. Л., Скрыпник С. И., Софронов И. Д.* Эйлерово-лагранжева методика численного решения трехмерных нестационарных задач газовой динамики с учетом теплопроводности // Вопросы атомной науки и техники. Сер. Методики и программы численного решения задач математической физики. 1988. Вып. 3. С. 3—8.
12. *Софронов И. Д., Афанасьева Е. А., Винокуров О. А. и др.* Комплекс программ МИМОЗА для решения многомерных задач механики сплошной среды на ЭВМ Эльбрус-2 // Там же. Сер. Математическое моделирование физических процессов. 1990. Вып. 2. С. 3—9.
13. *Горелик А. М.* Эволюция языка программирования Фортран (1957—2007) и перспективы его развития // Вычислительные методы и программирование, 2008. Т. 9. Вып. 2. С. 53—71.
14. Фортран 90. Международный стандарт: Пер. с англ. М.: Финансы и статистика, 1998.
15. ISO/IEC 1539-3:1997 Information technology. Programming languages. Fortran.
16. ISO/IEC 1539-1:2005 Information technology. Programming languages. Fortran. Part 1: Base language.
17. ISO/IEC 1539-1:2010 Information technology. Programming languages. Fortran. Part 1: Base language.
18. *Горелик А. М.* Объектно-ориентированное программирование на современном Фортране: Препринт № 70. М.: ИПМ им. М. В. Келдыша РАН, 2002.
19. *Reid J.* The New Features of Fortran 2003. http://www.kcl.ac.uk/kis/support/cit/fortran/john_reid_new_2003.pdf.
20. Система управления версиями. https://ru.wikipedia.org/wiki/Система_управления_версиями.
21. *Воропинов А. А., Новиков И. Г., Соколов С. С.* Методы мелкозернистого распараллеливания в методике ТИМ-2D // Вопросы атомной науки и техники. Сер. Математическое моделирование физических процессов. 2012. Вып. 3. С. 24—33.

Статья поступила в редакцию 02.02.17.

DATA STRUCTURE OF "TIM" METHOD TO SOLVE 2D AND 3D PROBLEMS OF CONTINUUM MECHANICS / A. A. Voropinov (FSUE "RFNC-VNIIEF", Sarov, Nizhny Novgorod region)

A static polymorphism-based approach to data structure organization using inheriting mechanisms of Fortran 2003 programming language is discussed. The advantage of the approach described is the possibility of simultaneous implementation of both old structures that do not use inheriting mechanisms and new ones through the structures with polymorphism.

The described approach is used in the program realization of the data structures of TIM method designed for solution of 2D and 3D problems of continuum mechanics on non-structured Lagrangian grids. Implementation of static polymorphism made it possible to realize a unified data structure for 2D abd 3D cases

Keywords: non-structured grid, data structure, polymorphism, Fortran 2003.