CMF - фреймворк для модели Земной системы высокого разрешения*

В.В. Калмыков[1,2], Р.А. Ибраев[3,2,1]

Институт океанологии им. П.П. Ширшова РАН[1], Гидрометцентр России[2], Институт вычислительной математики РАН[3]

Мы представляем оригинальный фреймворк CMF (Compact Modeling Framework), разрабатываемый для совместной модели океан-атмосфера высокого разрешения. Представлены две версии CMF, отражающие ход наших исследований. Базовая версия CMF 2.0 использует архитектуру с центральным хабом и абстрактным драйвером высокого уровня, оптимизированный алгоритм параллельной интерполяции и параллельную схему ввода/вывода модельных данных. CMF 3.0 заменяет прямой MPI-подход к коммуникациям абстракцией PGAS, а централизованную архитектуру - набором распределенных параллельно работающих сервисов. Для оценки производительности системы представлены тесты на суперкомпьютерах «МВС» и «Ломоносов».

1. Введение

Модель Земной системы - это объединенный ансамбль физических моделей: атмосферы, океана, льда и суши. Как отмечалось на Всемирной конференции по моделированию и прогнозированию климата (WCRP) [1], существует согласие о том, что гораздо более высокое, по сравнению с существующим, разрешение моделей основных компонентов является главной предпосылкой для реалистичного представления климатической системы. Сложность физических и химико-биологических процессов в Земной системе и большой объем данных о состоянии системы ставят задачу моделирования изменений климата Земли в ряд самых вычислительноемких в науке.

Наряду с разработкой моделей отдельных компонентов Земной системы возникает отдельная задача развития инструментов для их координированной работы — фреймворков для совместного моделирования. Архитектура фреймворка зависит от сложности моделей, характеристик связей между ними и доступных компьютерных ресурсов.

Развитие инструментов для совместного моделирования во многом следует за эволюционированием совместных моделей океан-атмосфера. Первые системы объединяли физические компоненты напрямую и вообще не требовали дополнительного кода. По мере усложнения моделей и выделения их в отдельные программы, появилась необходимость в отдельном сервисном компоненте — каплере (англ. couple — соединять), который занимался интерполяцией данных между различными модельными сетками компонентов. На первом этапе он представлял собой просто набор процедур для передачи полей через файловую систему, потом был выделен в отдельную последовательную программу, представляющую собой аналог центрального хаба для связи всех моделей. По мере увеличения разрешения сеток моделей, последовательные алгоритмы каплера становились неэффективными и на их место пришла полностью параллельная архитектура. Последние нововведения во фреймворках для моделирования связаны с упрощением общей архитектуры за счет введения управляющего высокоуровневого драйвера, который хоть и требует переформулирования моделей в интерфейсном виде, но позволяет упростить их синхронизацию [2, 3].

В итоге, можно выделить следующие характеристики фреймворка:

1. Архитектура совместной модели: последовательная, параллельная, с высокоуовневым драйвером или в виде процедур, запуск в виде одного или нескольких исполняемых

* Исследование выполнено в ГМЦ РФ за счет гранта Российского научного фонда (проект №14-37-00053).

файлов). Архитектура определяет сложность разработки и накладывает неявные ограничения на производительность;
2. Архитектура ввода-вывода (последовательная или параллельная, синхронная или асинхронная): предполагает баланс между сложностью разработки и необходимой скоростью сохранения и чтения данных;
3. Простота использования: уровень абстракции системы определяет количество необходимых изменений при модификации и замене моделей и прозрачность работы всего ансамбля.
4. Скорость работы: выбор низкоуровневых алгоритмов определяет скорость работы всей совместной модели.

Согласно [4], существует несколько общих трендов развития современных фреймворков. Предпочтение отдается системам с единым исполняемым файлом, модульным видом компонентов, параллельными алгоритмами как каплера, так и процедур ввода-вывода и использованием ставших по сути стандартами библиотек SCRIP (A Spherical Coordinate Remapping and Interpolation Package [5]) (для построение интерполяционных весов} и netCDF [6] (для хранения геофизических данных).

В данной работе мы представляем две версии нашей системы CMF (Compact Modeling Framework). CMF 2.0 реализует архитектуру с центральным параллельным каплером-хабом, высокоуровневым драйвером и параллельным асинхронным вводом-выводом. CMF 3.0 расширяет чистый MPI-коммуникации за счет использования абстракции PGAS, а центральный хаб превращается в SOA-подобную архитектуру с очередью заданий и параллельно работающими сервисами.

2. Описание CMF 2.0

Описание системы подробно представлено в работе [7]. Здесь приводится лишь краткая характеристика основных логических частей системы.

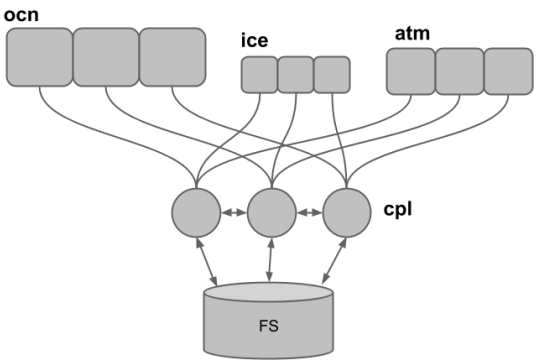

Рис. 1. Архитектура совместной модели в CMF 2.0. В данном примере: три компонента (океан, лед, атмосфера) объединены 3-ядерным каплером, который взаимодействие компонентов и работает с файловой системой

2.1 Архитектура программы

В CMF 2.0 модель запускается в виде единого исполняемого файла с указанием количества MPI-процессов для каждого компонента (например, океана, атмосферы и каплера). Каплер вызывает процедуры инициализации и входит в цикл обработки запросов. Аналогичные вызовы происходят и для N модельных компонентов, за исключением того, что для них вызываются абстрактные интерфейсы, которые определяют, что будет делать конкретная модель на инициализации и в процессе счета. Такой подход позволяет, во-первых, инкапсулировать все изменения модели внутри данных интерфейсов, а, во-вторых, иметь полностью независимый от них

код главной программы и каплера, что особенно важно при смене конфигурации системы (например, перехода от системы океан-атмосфера к системе океан-атмосфера-лед).

2.2 Архитектура совместной системы

Каждое ядро каплера взаимодействует с подмножеством ядер из физического компонента, что определяет локальность коммуникаций и ввода-вывода. Схема совместной модели для трех компонентов и трех ядер каплера изображена на **Рис. 1**.

Все возможные события в системе поделены на классы: сохранение диагностики, сохранение контрольной точки, чтения данных, прием или отправка данных на интерполяцию и т. д. Каждое событие имеет свой период и определяет, что конкретно будет происходить с данным полем на стороне каплера (например, будет ли оно сохранено в файл или отправлено другому компоненту).

В CMF 2.0 на инициализации каплер собирает всю информацию о событиях каждого компонента, чтобы построить таблицу событий и таким образом переходить к обработке следующего без синхронизации. Кроме того, все потоки данных в системе инициализированы как отложенные (комбинация MPI_SEND_INIT and MPI_STARTALL), что позволяет экономить время посылки для повторяющихся сообщений.

2.3 Каплер: интерполяция

Основная вычислительная задача каплера – интерполяция данных между различными модельными сетками. Алгоритмы каплера должны уметь поддерживать все взаимодействия в системе, независимо от их числа (например, океан-атмосфера, атмосфера-океан, океан-лед и т.д.)

Во время счета компоненты асинхронно отправляют данные каплеру, который уже внутри своего коммуникатора производит параллельное умножение на разреженную матрицу весов, как описано в [8]. Сама матрица строится на довычислительной стадии с помощью пакета SCRIP.

Каждое ядро каплера работает только с подмножеством ядер компонента, а значит, содержит только часть глобальных данных в памяти, в то время как остальная часть должна быть получена при каждой интерполяции с помощью MPI-обменов. Оптимизации процедур интерполяции подробно описаны в [6].

Для тестирования эффективности алгоритмов интерполяции проводятся «пинг-понг» тесты, а именно обмен полями двух пустых компонентов [2, 9]. В нашем тесте использовалась совместная модель океана и атмосферы, в которой океан каждые 2 часа отправлял атмосфере 3 поля, и каждый час получал от нее 9 полей. Океаническая модель имела трехполярную сетку размера 3600×1728 точек, атмосферная – широтно-долготную размера 1600×864 точек (размеры сеток и периоды обменов взяты из реальных моделей океана и атмосферы). Тест длился 10 модельных дней, что составляет, 120×3 интерполяций океан-атмосфера и 240×8 интерполяций атмосфера-океан. Процесс интерполяции представлял собой сбор данных от компонента-источника, работу процедур каплера и распределение данных компоненту-получателю.

На **Рис. 2** представлены результаты работы теста на суперкомпьютерах «МВС-100к» и «МВС-10П». График показывает хорошую масштабируемость алгоритма для разных чисел ядер каплера при фиксированном числе ядер для океана и атмосферы (1152 и 288 соответственно). Для «МВС-10П» представлены две конфигурации — с 16 и 32 ядрами на узел. Разница в производительности отражает увеличение коммуникационной нагрузки в последнем случае.

Из графика видно, что для практических вычислений с данными размерам задачи достаточно 20-40 ядер каплера, так как ~1 секундные затраты на каплинг для 1 дня моделирования являются незначительными по сравнению со временем работы физических моделей. Стоит отметить, что свойство масштабируемости алгоритма понадобится при увеличении числа обслуживаемых каплером компонентов и увеличения разрешения их сеток.

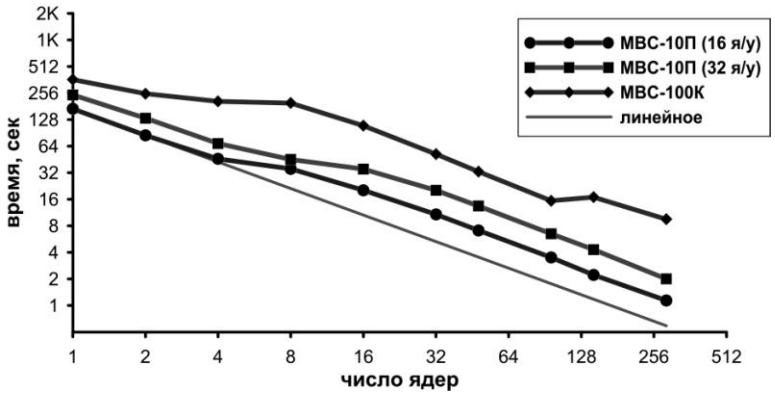

Рис. 2. Время работы теста для CMF 2.0 в секундах в зависимости от размера коммуникатора каплера на суперкомпьютерах семейства «МВС»: «МВС-100к», «МВС- 10р» с 16 ядрами на узел, «МВС-10р» с 32 ядрами на узел.

2.4 Каплер: ввод-вывод

В CMF 2.0 каплер, помимо интерполяции, занимается и работой с файловой системой. В нем реализован параллельный алгоритм ввода-вывода на основе формата netCDF. Ядра каплера являются внешними относительно моделей, поэтому схему записи можно рассматривать как схему с внешними делегатами. Такой подход позволяет компонентам асинхронно отправлять данные на специальные ядра ввода-вывода и продолжать работу, не ожидая окончания работы с дисками. Подробно блок ввода-вывода CMF 2.0 рассмотрен в [7].

3. Описание CMF 3.0

3.1 PGAS-коммуникатор

Несмотря на то, что CMF 2.0 показала себя пригодной для создания моделей высоко разрешения, она имела несколько направлений для улучшения. Во-первых, хотя чистый MPI-подход к коммуникациям имеет высокую скорость работы, он требует явной работы с буферами данных. Во-вторых, разработка региональных подмоделей морей, вложенных в сетку глобальной модели, становились довольно сложными при использовании только MPI-процедур. Результаты работы CMF 2.0 показали, что мы можем пожертвовать частью производительности для выбора более простой (и возможно менее эффективной) абстракции для упрощения коммуникационных алгоритмов.

В CMF 3.0 мы используем библиотеку Global Arrays (GA)[10], реализующую парадигму PGAS (Partitional Global Adress Space). Библиотека позволяет обращаться к глобальным индексам массива, как будто он весь доступен в локальной памяти.

CMF 3.0 содержит класс Communicator, который инкапсулирует логику работы с библиотекой и предоставляет интерфейс для put/get операций частей глобальных данных различных компонентов. Оказалось, что такой подход позволяет упростить не только взаимодействие вложенных компонентов, но и предоставляет удобную замену системе обмена данными CMF 2.0 между компонентами и каплером.

В итоге, все обмены между частями системы реализованы с использованием класса Communicator. Он содержит хэш-таблицу для хранения всей информации о массивах, включая их состояние и метаданные. Каждый массив компонента, участвующий в обменах содержит распределенную копию, хранящуюся в виде виртуального глобального массива GA. Когда процесс должен отправить данные, он заполняет эту копию своими текущими данными. Благодаря тому, что распределение глобального массива полностью повторяет декомпозицию компонента, эта операция происходит локально.

3.2 Новая архитектура совместной модели

Поскольку сложность совместной системы растет, нам необходим более удобный способ объединения компонентов. Первоначально появившаяся для веб-приложений, СОА (Сервис-Ориентированная Архитектура) дает хороший паттерн для решения подобной задачи.

В CMF 3.0 все модели отправляют свои запросы общего вида в единую очередь сообщений (**Рис. 3**). Сервисные компоненты берут из этой очереди только сообщения, которые могут обработать, забирают данные из виртуальных глобальных массивов и выполняют соответствующие действия. Архитектура позволяет минимизировать связи между физическими и сервисными компонентами и значительно упростить разработку. Более того, поскольку все сервисы наследуют общий базовый класс Service, добавление нового сервиса не представляет сложностей. Сейчас CMF 3.0 содержит следующие независимые параллельные сервисы: CPL (операции маппинга), IOS/IOF (I/O Fast, I/O Slow — быстрые и медленные устройства работы с файлами, DAS (Data Assimilation System — систему ассимиляции данных наблюдений).

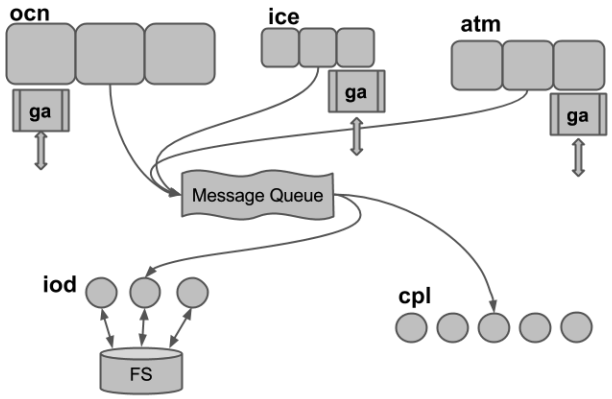

Рис. 3. Архитектура совместной модели в CMF 3.0. В данном примере: три компонента (океан, лед, атмосфера) отправляют запросы в очередь сообщений, откуда их извлекают сервисы каплера и ввода-вывода. Сами данные передаются через механизм глобальных массивов.

Сервис CPL представляет собой каплер из CMF 2.0, который теперь занимается исключительно операциями, связанными с интерполяцией. Он получает данные с использованием Communicator, выполняет интерполяцию и помещает данные в виртуальный глобальный массив получателя.

Далее, несмотря на то, что архитектура с центральным каплером CMF 2.0 позволяла собрать все операции вне моделей и выполнять их параллельно, иногда одновременные запросы к каплеру могли вызвать задержку всей системы. Например, в случае одновременного наступления событий интерполяции и запроса к файловой системе, каплер вынужден обработать их друг за другом. Это недостаток всех схем, которые предусматривают выполнение двух и более действий на одном процессе.

В CMF 3.0 мы решили выделить отдельный I/O-сервис, ответственный за все операции ввода-вывода. Интересно, что один сервис решает только половину проблемы, так как, например, одновременный запрос на запись большого количества информации (контрольной точки системы) и небольшой модельной диагностики также будет проходить последовательно. Поэтому мы разделили сервис на быстрое и медленное устройство записи (благодаря абстракции это деление выполняется несколькими строками кода). Такой механизм обеспечивает гибкий и асинхронный механизм работы с файловой системой.

Наконец, новый сервис DAS был добавлен в тестовом режиме для обеспечения работа системы ассимиляции. Кроме того ведутся работы по разработке сервиса DAS-K системы ассимиляции на основе фильтров Калмана.

3.3 Каплер: интерполяция

Несмотря на то, что логика работы процедур интерполяции каплера осталась той же, абстракция PGAS позволила сильно упростить код. Теперь, все данные, необходимые процессу каплера от соседей получаются с использованием класса Communicator. Недостатком такого подхода является падение производительности, связанное с невозможностью использования отложенных MPI-операций и наличия у библиотеки GA собственных издержек.

Для тестирования системы использовались те же параметры, что и для CMF 2.0. Стоит подчеркнуть, что оценка времени включает отправку запроса в очередь, отправку данных, интерполяцию и передачу их принимающей стороне, то есть моделирует полную работу системы. Тесты проводились на суперкомпьютере «Ломоносов».

График показывает, что, несмотря на то, что результаты ожидаемо хуже версии CMF 2.0, линейный тренд масштабируемости по-прежнему сохраняется (**Рис. 4**). Наконец, абсолютные значения в ~2-3 секунды затрат на каплинг (на 20-50 ядрах) для 1 дня моделирования удовлетворяют практическим запросам экспериментов.

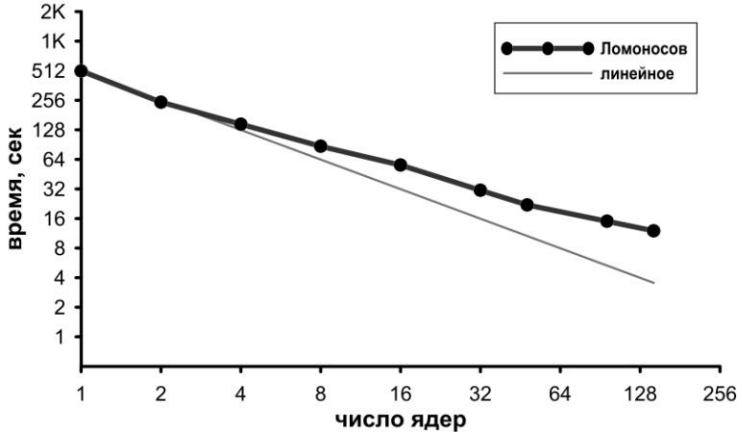

Рис. 4. Время работы теста для CMF 3.0 в секундах в зависимости от размера коммуникатора каплера на суперкомпьютере «Ломоносов».

4. Заключение

В данной работе представлена оригинальная система CMF, являющаяся первой в России параллельной системой для создания совместных моделей высокого разрешения.

CMF решает основные задачи функционирования модели Земной системы: синхронизации, интерполяции данных и ввода-высода большого количества данных. В CMF 2.0 мы начали с централизованной архитектуры с сервисным ядром — каплером, обеспечивающим выполнение всех сервисных операций. В CMF 3.0 мы разделили каплер на отдельные сервисы и сильно упростили всю систему коммуникаций за счет использования абстракции PGAS.

Для тестирования производительности системы были проведены тесты на суперкомпьютерах «МВС» и «Ломоносов». Обе версии демонстрируют линейную масштабируемость. Абсолютные величины задержи фреймворка в 2-5 сек в модельный день для связи двух компонентов высокого разрешения удовлетворяют требованиям длительных численных экспериментов.

CMF с самого начала разрабатывалась для решения прикладных задач. Благодаря CMF стало возможным создание первых в России моделей высокого разрешения: отдельной модели Мирового океана ИВМ-ИО [11] и совместной модели Мировой океан ИВМ-ИО-глобальная атмосфера ПЛАВ [12].

Литература

1. World Modelling Summit for Climate Prediction. Workshop report. UK. WCRP-131, WMO/TD-1468. 2008.

2. Craig A, Vertenstein M, Jacob R. A new flexible coupler for earth system modeling developed for CCSM4 and CESM1 // IJHPCA. 2012. Vol. 26, No. 1. P. 31-42.
3. Dennis J, Vertenstein M, Worley P. Computational performance of ultra-high-resolution capability in the Community Earth System Model // IJHPCA. 2012. Vol. 26. P. 5-16.
4. Valcke S, Balaji V, Craig A, et al. Coupling technologies for Earth System Modelling // Geosci. Model Dev. Discuss. 2012. Vol. 5. P. 1987-2006.
5. Jones P. A User's guide for SCRIP: A Spherical Coordinate Remapping and Interpolation Package. Los Alamos National Laboratory, 1998.
6. Unidata, (2015): Network Common Data Form (netCDF) version 4.3.3.1 (software). Boulder, CO: UCAR/Unidata. (http://doi.org/10.5065/D6H70CW6)
7. Калмыков В.В., Ибраев Р.А. Программный комплекс совместного моделирования системы океан–лед–атмосфера–почва на массивно-параллельных компьютерах // Вычислительные методы и программирование, 2013. 14. С. 88–95.
8. Craig A, Jacob R, Kauffman B. CPL6: The new extensible, high performance parallel coupler for the Community Climate System Model // IJHPCA. 2005. Vol. 19. P. 309-327.
9. Valcke S. The OASIS3 coupler: a European climate modelling community software // Geosci. Model Dev. Discuss. 2012. Vol. 5. P. 2139-2178.
10. Jarek Nieplocha, Bruce Palmer, Vinod Tipparaju, Manojkumar Krishnan, Harold Trease, and Edo Apra. "Advances, Applications and Performance of the Global Arrays Shared Memory Programming Toolkit" // IJHPCA. 2006. Vol. 20, No. 2. P. 203-231.
11. Ибраев Р.А., Хабеев Р.Н., Ушаков К.В. Вихреразрешающая 1/10º модель Мирового океана // Известия РАН. Физика атмосферы и океана, 2012. 48(1), С. 45–55.
12. Толстых М.А. Глобальная полулагранжева модель численного прогноза погоды // М, Обнинск: ОАО ФОП, 2010.

CMF - framework for high-resolution Earth system modeling

Vladimir Kalmykov and Rashit Ibrayev

Keywords: Earth system modeling, massively-parallel applications, parallel I/O, coupler, coupling, scaling, performance

We present an original framework CMF (Compact Modeling Framework) developed for high-resolution coupled ocean-atmosphere model. Two version of CMF are presented, reflecting progress of our researches. In CMF 2.0 we concentrate on single executable central hub approach with high-level abstract driver, optimized parallel coupler interpolation scheme and parallel I/O algorithm. CMF 3.0 replaces pure MPI approach with PGAS based communications scheme, while central hub architecture evolves to set of simultaneously working services. Performance tests are presented for "MVS" and "Lomonosov" supercomputers.