Для функции DENSE_RANK() были получены данные:

Число зап.	Без секционирования						С секционированием					
	int		varchar(max)		datetime(year)		int		varchar(max)		datetime(year)	
	ФР	Альт	ФР	Альт	ФР	Альт	ФР	Альт	ФР	Альт	ФР	Альт
1'000	61,5	65,3	63,2	85,2	59,8	307,4	64,1	68,5	69,5	85	62,8	191,4
10'000	137,1	151,2	237,1	310,5	131,6	4599,1	137,7	180	243,9	404,3	141,1	8470,8
100'000	1467,2	1639,3	2423,1	3152,5	1509,7	46686,3	1506,7	1728,5	2525	4133,7	1524,2	236362,9

Из таблицы видно, что с увеличением числа записей в таблице выгоднее использовать функцию ранжирования, а не вложенный запрос.

Функция NTILE() имеет особенность, что ее можно выразить через функцию ROW_NUMBER() с помощью формулы:

NTILE(n) = (n * (ROW_NUMBER() – 1) / count(*)) + 1.

В результате замеров этой функции и ее альтернатив были получены данные:

Число зап.	Без секционирования						С секционированием					
	int		varchar(max)		datetime(year)		int		varchar(max)		datetime(year)	
	ФР	Альт	ФР	Альт	ФР	Альт	ФР	Альт	ФР	Альт	ФР	Альт
1'000	67,5	71,7	71,6	67	66,6	68,7	61,2	71491,6	76,2	72424	61	71853,3
10'000	161,9	152,6	251,3	246,5	160,2	156,9	161,8	-	281,8	-	161,5	-
100'000	1722,6	1652,6	2570,9	2492,9	1780,5	1698,3	1738,6	-	2917,8	-	1748,9	-

В случае запроса без секционирования наблюдается преимущество альтернативного запроса, хоть и незначительное. Курсор будет выполняться быстрее функции ранжирования на всех рассмотренных типах данных, если секционирование не используется. Тем не менее, реализация альтернативного запроса без функции ранжирования является более трудоемкой для программиста.

Таким образом, исходя из произведенного исследования, можно сделать вывод о том, что использование функций ранжирования в системах реляционных баз данных позволяет сократить время выполнения запросов, а также текст запроса. Исключением является функция NTILE() без секционирования данных, для которой запрос без функции ранжирования является более производительным.

Литература

1. MS SQL 2005: оконные функции. Еще одно расширение T-SQL [Электронный ресурс]. – http://rsdn.ru/?article/db/WindowFunctions.xml

Д.А. Царёв, С.В. Востокин

ТЕХНОЛОГИЯ РАЗВЕРТЫВАНИЯ СКЕЛЕТНЫХ ПРОГРАММ ДЛЯ АВТОМАТИЗАЦИИ ВЫЧИСЛЕНИЙ НА СУПЕРКОМПЬЮТЕРЕ «СЕРГЕЙ КОРОЛЁВ»

(Самарский университет)

Задачи для высокопроизводительных вычислений на кластерных системах и суперкомпьютерах можно разделить на две категории. К первой категории относятся задачи математического моделирования, решаемые с помощью

специализированных программных пакетов. Например, моделирование в области механики твердого и жидкого тела на кластере «Сергей Королёв» выполняется на пакетах ANSYS, LS-DYNA, DEFORM. Перечисленные пакеты не требуют от пользователя программирования вычислительных алгоритмов, нужны только подготовка входных данных задач, управление развертыванием готовых программ и интерпретация результатов вычислений.

В тоже время существует большая категория задач моделирования, связанная с программированием численных моделей. В эту категорию попадают задачи из быстро развивающихся научных направлений: интеллектуального анализа данных, нейроинформатики, нелинейной динамики, компьютерной оптики и др. Здесь самостоятельная разработка программной реализации численного метода часто обусловлена новизной используемого алгоритма или объекта моделирования. Однако, в силу сложностей, связанных с организацией вычислений, исследователи часто отказываются от использования высокопроизводительных вычислительных систем и ограничиваются экспериментами на персональных рабочих станциях. Это приводит к увеличению времени проведения вычислительных экспериментов. В худшем случае, отказ от использования высокопроизводительной техники сказывается на качестве получаемых результатов. Например, сужение параметрического пространства ради сокращения времени вычислений может привести к пропуску существенных эффектов и, как следствие, неправильным выводам о свойствах моделируемого объекта. Исследования и разработки, выполняемые в рамках проекта Templet (http://templet.ssau.ru), направлены на решение актуальных задач автоматизации моделирования на основе вновь разрабатываемых программных реализаций математических моделей, исполняемых на суперкомпьютерных и кластерных системах.

В работе [1] предложено решение проблемы предоставления доступа по требованию к кластеру «Сергей Королёв» и описана программная реализация интегрированной среды разработки (IDE) параллельных программ на основе облачного сервиса типа PaaS (платформа как сервис). Целью данной работы является автоматизация кодирования параллельных программ, используемых в задачах математического моделирования. Идея применяемого в ней метода основана на концепции скелетного программирования. Под скелетной программой понимается высокоуровневая запись алгоритма, скрывающая детали реализации, несущественные для выражения идеи алгоритма. Такая запись похожа на запись алгоритма на псевдокоде. Однако, в отличие от программы на псевдокоде, скелет – это компилируемая программа на существующем языке программирования, например, на C++. Обзор систем-аналогов, реализующих эту концепцию, приведен в работе [2]. В данной работе, в продолжение работы [3], рассматривается возможность реализации скелетного программирования на базе облачного сервиса типа PaaS.

Предлагается модификация существующего IDE сервиса (http://templet.ssau.ru/app) для поддержки разработки и автоматического развёртывания четырёх типов программ, три типа из которых – скелетные программы.

Тип 0. *Обычные программы, полученные из встроенных в систему шаблонов*. Пользователь видит весь код программы и может произвольно адаптировать его под свою задачу. С программой связана дополнительная информация для выполнения автоматического развёртывания на кластере «Сергей Королёв» «в один клик».

Тип 1. *Скелетные программы с фиксированной структурой*. Это код, в котором при помощи комментариев определены точки расширения, куда пользователь может помещать свой код алгоритма моделирования. Помещаемый пользователем код – последовательный. Сам скелет также не содержит параллельных инструкций. Перед выполнением стандартной процедуры развертывания проводится автоматическое преобразование скелета типа 1 в программу типа 0 на основе специального шаблона преобразования. Работа с программами типа 1 рассматривается на примере типовых схем управления вычислениями конвейер (pipeline) и портфель задач (bag-of-tasks, farm, master-workers).

Тип 2. *Скелетные программы со структурой, определяемой директивами на предметном языке*. Преобразование программ типа 1 выполняется на основе фиксированного шаблона преобразования. В программах типа 2 шаблон преобразования генерируется при каждом преобразовании на основе директив в коде скелетной программы. В качестве примера программ типа 2 рассматриваются программы, реализующие вычисления в модели акторов. Семантика предметного языка основана на работе [4]. В новой версии синтаксис упрощен таким образом, чтобы размещаться в директивах **#pragma** языка C++, поэтому инструкции предметного языка игнорируются компилятором и воспринимаются только генератором шаблона преобразования.

Тип 3. *Скелетные программы со структурой, определяемой кодом на языке реализации*. В программах типа 2 генератор шаблона преобразования выполняет разбор специальных директив **#pragma templet**, включаемых в скелет. Данную работу можно возложить на компилятор языка реализации, например, на компилятор C++. Для этого, кроме обычных точек расширения, скелеты типа 3 включают точки расширения с C++ кодом описания скелета. Для вычисления шаблона преобразования и приведения программ типа 3 к типу 1 в процессе развертывания вначале из скелета выделяется C++ код с его собственным описанием. Затем выделенный код объединяется со служебным кодом и компилируется в исполнимую программу. Эта программа при запуске генерирует шаблон преобразования, сводя процедуру развёртывания к таковой для программ типа 1. Программы типа 3 также рассматриваются на примере акторных вычислений.

Таким образом, технология скелетного программирования позволяет снизить трудоёмкость разработки параллельных программ при проведении моделирования на базе высокопроизводительных вычислительных систем за счет рассмотренных механизмов сокрытия кода, управляющего параллельными вычислениями. Показано, каким образом технология скелетного программирования может быть реализована на базе облачного сервиса типа PaaS. Сервис и реализованные в нем методы автоматизации программирования используются

для решения задач моделирования многомерных динамических систем и процессов на суперкомпьютере «Сергей Королёв» [5].

Литература

1. Артамонов, Ю.С. Востокин С.В. Инструментальное программное обеспечение для разработки и поддержки исполнения приложений научных вычислений в кластерных системах [Текст] /Ю.С. Артамонов, С.В. Востокин //Вестн. Сам. гос. техн. ун-та. Сер. Физ.-мат. Науки – 2015 – С.785-798.

2. Gonz´alez-V´elez, H. Leyton, M. A survey of algorithmic skeleton frameworks: high-level structured parallel programming enabler [Текст] /Gonz´alez-V´elez, H. //Software: Practice and Experience – 2010 – 40(12) – С.1135-1160.

3. Царёв, Д.А. Артамонов, Ю.С. Сравнение основных возможностей и классификация облачных инструментов разработки. [Текст] /Д.А. Царёв, Ю.С. Артамонов //Перспективные информационные технологии (ПИТ 2016): труды Международной научно-технической конференции /под ред. С.А. Прохорова – Самара: Издательство СНЦ РАН – 2016. – С.539-542.

4. Vostokin, S.V. Templet: A markup language for concurrent actor oriented programming [Текст] /S.V. Vostokin //CEUR Workshop Proceedings – 2016 – С.460-468.

5. Востокин, С.В. Дорошин, А.В. Артамонов, Ю.С. Применение системы Templet Web для решения задач математического моделирования с использованием высокопроизводительных систем. Управление движением и навигация летательных аппаратов [Текст] /С.В. Востокин, А.В. Дорошин,
Ю.С. Артамонов //Сборник трудов XVIII Всероссийского семинара по управлению движением и навигации летательных аппаратов: Часть II. Самара, 15-17 июня 2015 г. – Самара, Изд-во СНЦ РАН – 2016 – С.17-21.

Е.В. Чернова, П.Н. Полежаев

АНАЛИЗ ОСНОВНЫХ ПРОБЛЕМ МИГРАЦИИ ВИРТУАЛЬНЫХ МАШИН

(Оренбургский государственный университет)

Миграция виртуальных машин все больше используется в корпоративной среде. Несмотря на значительные преимущества, которые можно достичь при работе с виртуальными машинами используя миграцию, ей так же присущи недостатки. Повышение эффективности и результативности миграции виртуальных машин становится важной проблемой. В данном докладе рассматриваются проблемы, связанные с миграцией виртуальных машин, которые имеют некоторые решения, но все еще требуют дальнейших исследований.

Несмотря на нынешние достижения в технологии живой миграции, попрежнему неизбежно короткое время простоя сервиса, работающего внутри перемещаемой виртуальной машины. Кроме того, в процессе переноса приложений может возникнуть снижение производительности за счет двойных затрат