УДК 004.424.74

А. В. Аксенов*
старший преподаватель
Е. В. Петрова*
студент
*Санкт-Петербургский государственный университет аэрокосмического приборостроения

СПОСОБЫ УПРАВЛЕНИЯ ПАМЯТЬЮ ПРИ НАКОПЛЕНИИ СТАТИСТИКИ В ЗАДАЧЕ КОНТЕКСТНОГО МОДЕЛИРОВАНИЯ

Рассматривается задача контекстного моделирования и ее особенности применительно к специфике управления памятью статистической модели. Далее проводится обзор различных методик ручного управления памятью и оценивается целесообразность их применения в данной задаче.
Ключевые слова: динамическое распределение памяти, управление памятью.

A. V. Aksenov*
Senior Lecturer
E. V. Petrova*
Student
*St. Petersburg State University of Aerospace Instrumentation

PRINCIPLES OF MEMORY MANAGEMENT DURING STATISTICS ACCUMULATION IN CONTEXT MODELLING TASK

The article describes the task of context modelling and its peculiar properties as applied to the specifity of managing memory for statistical model. Various methods of manual memory management are reviewed and expediency of using them in this task is assessed.
Keywords: dynamic memory allocation, memory management.

Контекстное моделирование

Для того чтобы осуществить эффективную работу с сообщением, требуется знание его характеристик. В качестве таких характеристик принято брать вероятность появления символа в зависимости от контекста. Данный принцип лежит в основе методов контекстного моделирования.

Для каждого контекста конечной длины из обрабатываемой последовательности создается контекстная модель (КМ). В нее включаются счетчики всех символов, встречаемых в данном контексте, увеличение которых происходит каждый раз, когда этот символ был встречен в контексте. Счетчики создаются по мере появления новых символов. Опираясь на статистику, собранную во время обработки входной последовательности, метод контекстного моделирования позволяет оценить вероятность появления символа или их последовательности.

Управление памятью

В большинстве случаев мы не задумываемся о том, как управлять памятью, но это не делает данную область менее важной. Знание возможностей и ограничений менеджера памяти необходимо для повышения эффективности программы. Роль аллокатора (распределителя памяти) играет важное значение особенно тогда, когда память может стать для программы «узким местом». Иногда стандартные аллокаторы не подходят под специфику выполняемой программой задачи, из-за чего может возникнуть необходимость в написании собственного аллокатора, учитывающего особенности задачи, в данном случае контекстного моделирования.

В случае контекстного моделирования при написании менеджера памяти следует учитывать объем накапливаемой статистики, во время увеличения которого постоянно выделяется память, также уменьшается эффективность использования данной статистики, так как «старая» статистика малополезна и может быть вредна. Исходя из этого, имея ограниченное пространство памяти, необходимо выполнять следующие действия: убедиться, что есть достаточное количество памяти для выполнения процесса; получить раздел памяти из доступной; после окончания процесса вернуть раздел выделенной памяти обратно в пул, чтобы он мог быть использован другими процессами.

Память можно выделять двумя способами: динамически и предварительно.

Динамическое выделение памяти

Динамическое выделение памяти – самый простой способ, позволяющий использовать память по минимуму, так как она выделяется по мере накопления статистики символов.

Но данный способ имеет и ряд недостатков. Во-первых, замедляется выполнение программы, поскольку необходимо выделять память во время ее исполнения. Во-вторых, частое выделение небольших сегментов памяти в больших количествах может привести к ее фрагментации, из-за чего даже имея достаточный объем памяти, мы не сможем ее выделить, потому как она будет разбита на несколько частей. В-третьих, память не бесконечна, поэтому существует риск того, что программа попытается выделить память, а операционная система не позволит ей этого сделать. В такой ситуации можно попробовать освободить память от устаревших данных и записать на их место новые или попытаться сделать программу устойчивой к исчерпанию памяти.

Предварительное выделение памяти

Есть также другой способ выделения памяти – предварительное ее выделение. Идея данного способа заключается в том, что память выделяется единожды и никогда больше не производит динамических выделений во время выполнения программы. Обычно берется большой блок памяти и используется под нужды программы. Этот метод дает следующие преимущества:

1) улучшенная производительность, так как не требуется тратить время на выделение памяти;

2) использование большего объема памяти, чем выделено;

3) отсутствие фрагментации [1].

Но для реализации данного способа необходимо точно знать, сколько памяти нам потребуется. А также это значит, что мы можем создать ограниченное количество элементов.

Учитывая преимущества предварительного выделения памяти перед динамическим, в данной работе будет использован именно этот способ. Но при разработке менеджера памяти необходимо учитывать недостатки данного метода.

Реализации аллокатора

Самым простым способом реализации аллокатора является линейный аллокатор, выделяющий линейным способом куски памяти один за другим. Данный способ не предусматривает частичного освобождения выделенных ресурсов памяти. Память полностью освобождается в конце процесса. Таким образом, у нас нет возможности удаления старых данных и записи на их место новых, поэтому данный способ подходит лишь для контекстных моделей сравнительно небольших размеров.

Пуловый аллокатор предназначен для работы с данными одинакового типа и размера, какими и являются данные в контекстной модели. Он делит буфер памяти на равные сегменты, а затем позволяет выделять и освобождать их по команде. Но в данном случае память выделяется и освобождается под каждый сегмент отдельно.

Если работать не с каждым сегментом в отдельности, а с блоками сегментов, то мы получим блочный пуловый аллокатор.

Идея этой реализации заключается в выделении памяти равными блоками сегментов, чем и отличается от предыдущего способа. На запрос о выделении элемента возвращается указатель на следующий элемент блока, и счетчик выделенных элементов инкрементируется. Когда блок закончится, будет запрошен следующий, и так далее. Освобождение происходит быстрее за счет того, что не требуется уделять время каждому элементу и освобождаем сразу несколько сегментов памяти.

После того как мы освободили память, хотелось бы ее занять новыми данными статистики. Помимо счетчика внутри каждого блока, существует счетчик блоков, при помощи которого отслеживается момент, когда не получится выделить следующий блок, так как последует конец выделенной под задачу памяти. В таком случае с выделением последнего блока начнется очистка блоков начиная с первого. Тогда после окончания записи в последний блок-указатель будет перенесен в начало пула, и запись продолжится.

Выводы

Достоинства данного метода:

1. Используется предварительное выделение памяти – нет затрат времени на динамическое выделение в процессе выполнения задачи.

2. Освобождение происходит блоками во время записи новых данных – память освобождается быстрее за счет того, что мы оперируем блоками сегментов, а не каждым сегментов в отдельности, не происходит задержек stop the world, поскольку удаление данных не мешает записи новых и не происходит их искажения.

3. Освобождая память от старой статистики, мы предоставляем возможность записи новых данных, тем самым обновляя накопляемую статистику.

Недостатком метода является то, что удаленная статистика могла пригодиться в будущем. Избежать этого поможет сортировка накопленной статистики по весу – наиболее часто встречающаяся имеет меньший вес, чем та, которая появляется реже [2]. Таким образом, можно удалять только редкую статистику, оставляя более полезную частую.

Библиографический список

1. Rentzsch J. Data aligment: Straighten up and fly right. URL: https://developer.ibm.com/articles/pa-dalign (дата обращения: 02.04.2018).
2. Аксенов А. В. Разработка и анализ адаптивного алгоритма сжатия без потерь // ВКРМ. СПб., 2010.