ОТЛАДКА ЭФФЕКТИВНОСТИ MPI-ПРОГРАММ

В.А. Крюков, М.И. Кулешова

Высокая сложность разработки параллельных программ, часто не позволяющая эффективно использовать возможности высокопроизводительных компьютеров, является общепризнанным фактом. В настоящее время ведутся интенсивные исследования в области автоматизации разработки параллельных программ, в частности, в области создания инструментов для отладки и исследования эффективности[5-9]. Данная работа посвящена разработке средств, предназначенных для отладки и исследования эффективности MPI-программ для кластерных вычислительных систем.

Эффективность MPI-программ

Под эффективностью MPI-программы мы будем понимать эффективность ее распараллеливания, которая определяется следующими основными факторами:

- долей параллельных вычислений в общем объеме вычислений,
- равномерностью загрузки процессоров во время выполнения параллельных вычислений,
- временем, необходимым для выполнения межпроцессорных обменов,
- степенью совмещения межпроцессорных обменов с вычислениями,
- эффективностью выполнения вычислений на каждом процессоре [7].

Для оценки доли параллельных вычислений мы будем использовать коэффициент эффективности, равный отношению полезного времени к общему времени использования процессоров. Где полезное время − это предполагаемое время работы программы на одном процессоре. Считаем время предполагаемым, так как оно вычисляется как разность общего времени и времени, потраченного на обеспечение параллельного выполнения программы.

Keff = Tproductive / Ttotal = (Ttotal – Tlost) / Ttotal, где

Tproductive – предполагаемое время работы программы на одном процессоре

Ttotal – общее время использования процессоров

Tlost – время, потраченное на обеспечение параллелизма

Для оценки остальных факторов эффективности распараллеливания необходимо проанализировать соответствующие временные характеристики. То есть:

Tload_imbalance – потенциальные потери из-за неравномерной загрузки процессоров,

Tcomm – время, потраченное на обмены сообщениями,

Toverlap – время совмещения обменов с вычислениями.

Таким образом, отладка эффективности MPI-программы сводится к анализу некоторых временных характеристик выполнения программы и коэффициента эффективности, и, на основании этого анализа, к поиску и устранению узких мест.

Подходы к вычислению характеристик эффективности

Основными подходами к вычислению характеристик эффективности MPI-программ являются: *вычисление характеристик по трассе* и *динамическое вычисление*.

Вычисление характеристик по трассе производится в два этапа. На первом этапе во время работы программы происходит сбор трассы, то есть разнообразной информации об MPI-вызовах: времена обращений к MPI-функциям, параметры вызовов, информация о созданных задачей коммуникаторах, типах данных, возвращаемые значения и т. д. На втором этапе происходит анализ этой трассы и подсчет временных, количественных характеристик работы программы и коэффициента эффективности.

Динамическое вычисление характеристик производится в процессе выполнения программы.

Причинами реализации первого подхода послужили такие факторы как:

- стремление к минимальному вмешательству в работу программы,
- сохранение возможности по мере необходимости получать более детализированную информацию о работе программы без повторных ее запусков
- стремление создать систему отладки эффективности в рамках того же подхода, который был положен в основу уже созданной системы отладки корректности. В том числе сохранить возможность использовать на первом этапе вычисления характеристик эффективности по трассе разработанный и реализованный ранее трассировщик.

Однако количество узлов и производительность современных суперкомпьютеров растет все больше и больше с каждым годом. В этих условиях возрастает необходимость реализации второго подхода, который позволяет обойтись без трасс, накопление которых на больших машинах весьма проблематично.

В рамках данной работы были созданы средства, реализующие оба подхода в одной системе.

Вычисление характеристик эффективности по трассе. Анализатор эффективности MPI-программ

Для второго этапа вычисления характеристик эффективности трассе был разработан инструмент,

называемый *анализатор эффективности* или *анализатор*. Анализатор осуществляет проход по трассам, накопленным во время параллельного выполнения программы, обрабатывает их, и на основании этой обработки решает две различные задачи: вычисление реальных временных характеристик эффективности выполнения MPI-программы и коэффициента эффективности и вычисление прогнозируемых временных характеристик и коэффициента эффективности [3].

Вычисление реальных временных характеристик

Вычисление реальных временных характеристик позволяет оценить эффективность работы программы и резервы для ее повышения, понять равномерно ли распределена вычислительная нагрузка и помогает в поиске узких мест. Как было сказано ранее, главными показателями эффективности являются такие характеристики как Keff, Tload_imbalance, Tcomm, Toverlap. Для того чтобы их вычислить, необходимо замерить и подсчитать ряд временных характеристик. Таким образом, можно выделить основные временные характеристики MPI-программ или основные характеристики [1].

K*eff* Коэффициент эффективности (Parallelization Efficiency) – равен отношению предполагаемого времени работы на одном процессоре к общему времени использования процессоров,

T*execution* Время выполнения (Execution Time) – равно максимальному значению среди времен выполнения программы на каждом из используемых ею процессоре,

P Число используемых процессоров (Processors),

T*total* Общее время использования процессоров (Total Time) – равно произведению времени выполнения и числа используемых процессоров,

T*productive* Предполагаемое время работы на одном процессоре (Productive Time) – равно разности между общим временем использования процессоров и потерянным временем,

T*lost* Потерянное время (Lost Time) – равно сумме коммуникаций, потерь из-за недостаточного параллелизма и простоев,

Компоненты потерянного времени:

T*comm* Коммуникации (Communications) или потери из-за выполнения межпроцессорных обменов – равны сумме времен выполнения всех MPI-функций,

Компоненты коммуникаций:

T*sendrecv* Сумма времен выполнения всех операций точка-точка (SendRecv Time), включая – время рассинхронизации (Real Sync), равно потерям времени, вызванным тем, что операция приема сообщения или операция ожидания завершения асинхронной операции на одном процессоре была выдана раньше соответствующей операции посылки сообщения на другом процессоре,

T*collective* Сумма времен выполнения коллективных операций (Collective Time),

T*system* Время служебных операций (System Time) – равно сумме времен MPI-функций, не являющихся ни операциями точка-точка, ни коллективными операциями,

T*idle* Простои (Idle Time) – равны просуммированной по всем процессорам разности времени выполнения программы (максимального времени работы среди всех процессоров) и времени работы процессора,

T*insufficient_parallelism* Недостаточный параллелизм (Insufficient Parallelism) – равен потерям из-за дублирования последовательных вычислений на многих процессорах,

T*load_imbalance* Разбалансировка (Load Imbalance) – потенциальные потери, которые могут возникнуть из-за того, что вычисления распределены между процессорами неравномерно,

T*potential_sync* Потенциальные потери на синхронизацию (Potential Sync) –потери, которые могут возникнуть из-за неодновременного запуска коллективных операций на разных процессорах

T*time_variation* Потенциальные потери из-за разброса времен (Time Variation) – потери, которые могут возникнуть из-за различий во временах завершения выполнения на разных процессорах одной и той же коллективной операции

T*overlap* Время перекрытия обменов (Overlap) – время совмещения межпроцессорных обменов между собой или с вычислениями.

Нередка ситуация, при которой пользователь в процессе отладки эффективности MPI-программы ограничивает количество итераций своей задачи. В этом случае время подготовительного и завершающего этапов может быть сравнимо с временем работы основного цикла. В такой ситуации может оказаться полезным

наличие возможности вычислять временные характеристики только для основного цикла. Также в процессе отладки эффективности у пользователя может возникнуть потребность вычислять характеристики для отдельных кусков программы или же непосредственно в точках вызова MPI-функций. Для удовлетворения подобных потребностей пользователя в анализаторе эффективности реализованы следующие способы детализации подсчета временных характеристик: интервалы и точки SRC. В виде отдельного интервала можно оформить любой кусок программы, информация о котором программиста интересует, в том числе, например, основной цикл. Вся программа целиком при этом считается нулевым интервалом. Второй способ детализации – точки SRC. Любому обращению к MPI-функции соответствует стек вызовов процедур программы пользователя, вершиной которого является данное обращение. Этот стек можно считать описанием некоторой точки вызова MPI-функции в программе. Каждой такой точке, приписан свой идентификационный номер (номер SRC). Каждую точку SRC тоже можно считать своеобразным интервалом. Временные характеристики подсчитываются для всех интервалов тем или иным способом отмеченных в программе, в том числе и для всех точек SRC.

Вычисление прогнозируемых временных характеристик

Отладка эффективности, как и любая отладка, связана с множеством повторных запусков одной и той же задачи. Однако от запуска к запуску параллельной программы на реальных машинах можно получать разные времена. Это зависит от множества разнообразных внешних причин. Поэтому при отладке эффективности бывает сложно понять какой эффект имеют внесенные пользователем в программу изменения. Для решения подобных проблем в анализаторе эффективности используется механизм прогнозирования временных характеристик выполнения параллельной программы. Прогнозирование – один из важнейших элементов отладки эффективности, основной целью которого является получение стабильных временных характеристик выполнения программы. Для получения этой стабильности в анализаторе используются следующие механизмы. Во-первых, была разработана и реализована библиотека, моделирующая работу MPI-функций и прогнозирующая времена выполнения всех MPI-вызовов программы пользователя, которые используются для подсчета характеристик эффективности вместо нестабильных реальных времен. Во-вторых, вместо реальных времен выполнения последовательных участков программы используются усредненные. В-третьих, производится коррекция времен выполнения последовательных участков с учетом производительности процессоров.

Результат работы анализатора эффективности

Результатом работы анализатора эффективности является *протокол анализатора эффективности* или *протокол*. Это текстовый файл, имеющий структуру, представленную на рис. 1а.

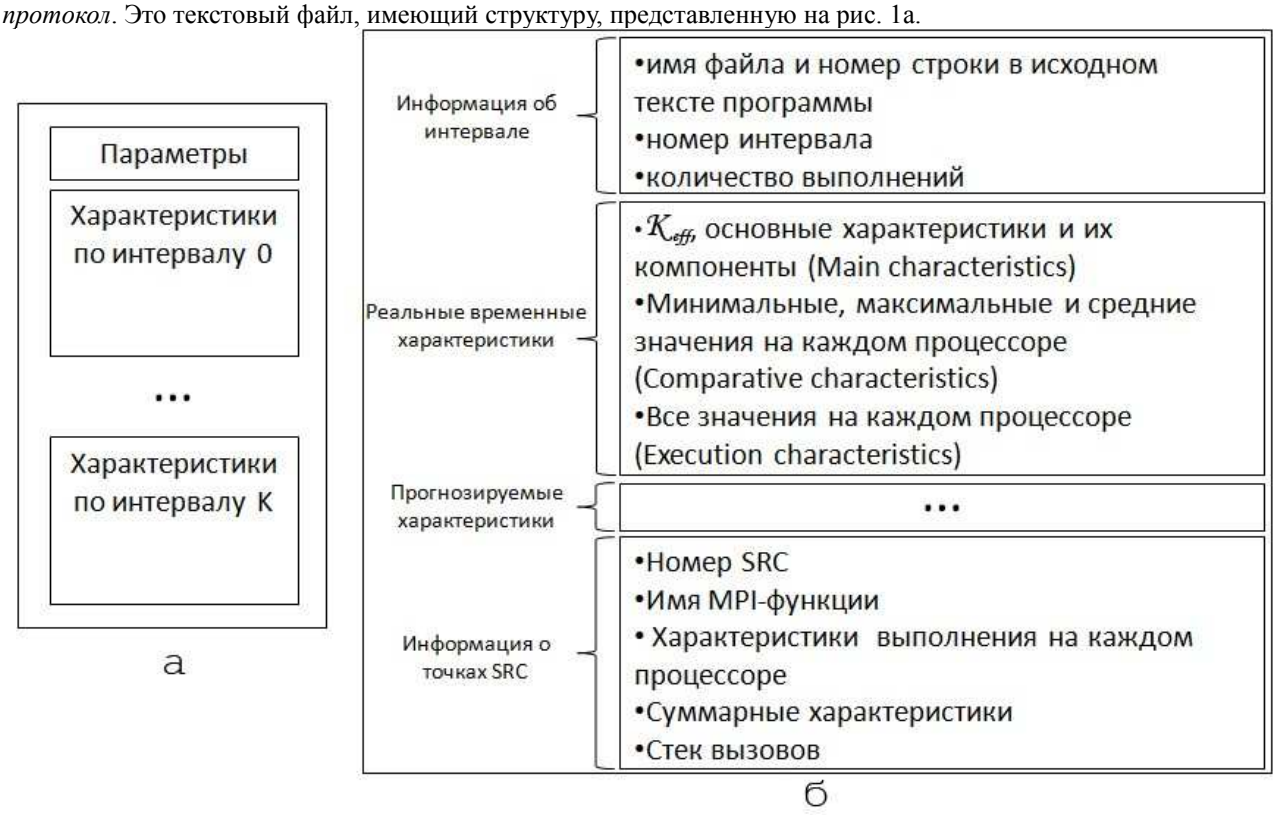

Рис. 1. Структура протокола анализатора эффективности

В разделе «Параметры» отображаются такие характеристики как свойства машины, на которой запускалась задача и собиралась трасса, время запуска задачи, номер версии трассировщика, номер версии анализатора эффективности, параметры его выполнения, а также дата и время его запуска и т д.

Далее в протоколе анализатора последовательно отображаются собранные по каждому интервалу временные характеристики эффективности. Структура выдачи информации о характеристиках по каждому интервалу представлена на рис.1б. Для каждого интервала сначала выдается такая информация о нем, как имя файла и номер строки в исходном тексте программы, где начинается интервал, номер интервала, количество его выполнений. Затем реальные основные временные характеристики во главе с коэффициентом эффективности, временами, из которых он считается, и их компонентами суммарно по всем процессорам, по каждому в отдельности и в сравнении друг с другом. Та же информация выдается о прогнозируемых основных характеристиках. И, наконец, отображается информация обо всех точках SRC данного интервала. О каждой выдается ее номер, имя MPI-функции, к которой относится рассматриваемая точка, ее временные характеристики суммарно по всем процессорам и по каждому в отдельности, а также относящийся к ней стек вызовов.

Использование анализатора эффективности MPI-программ

Существует два варианта использования анализатора эффективности: автономно в пакетном режиме и как элемент системы диалоговой отладки. В обоих случаях предварительно должна быть собрана трасса. Автономное использование осложняется тем, что для анализа информации, выдаваемой анализатором, приходится работать с множеством файлов большого размера. Это и протокол с, вообще говоря, большим количеством информации о точках SRC, и исходные тексты, и файлы трасс, которые тоже могут быть очень большими. Находить события в трассах и строки в исходных текстах, соответствующие нужной строке в протоколе, может оказаться довольно трудоемкой задачей. Поэтому была разработана и реализована система диалоговой отладки эффективности MPI-программ [2].

Система диалоговой отладки имеет довольно сложную распределенную структуру, представленную на рис. 2.

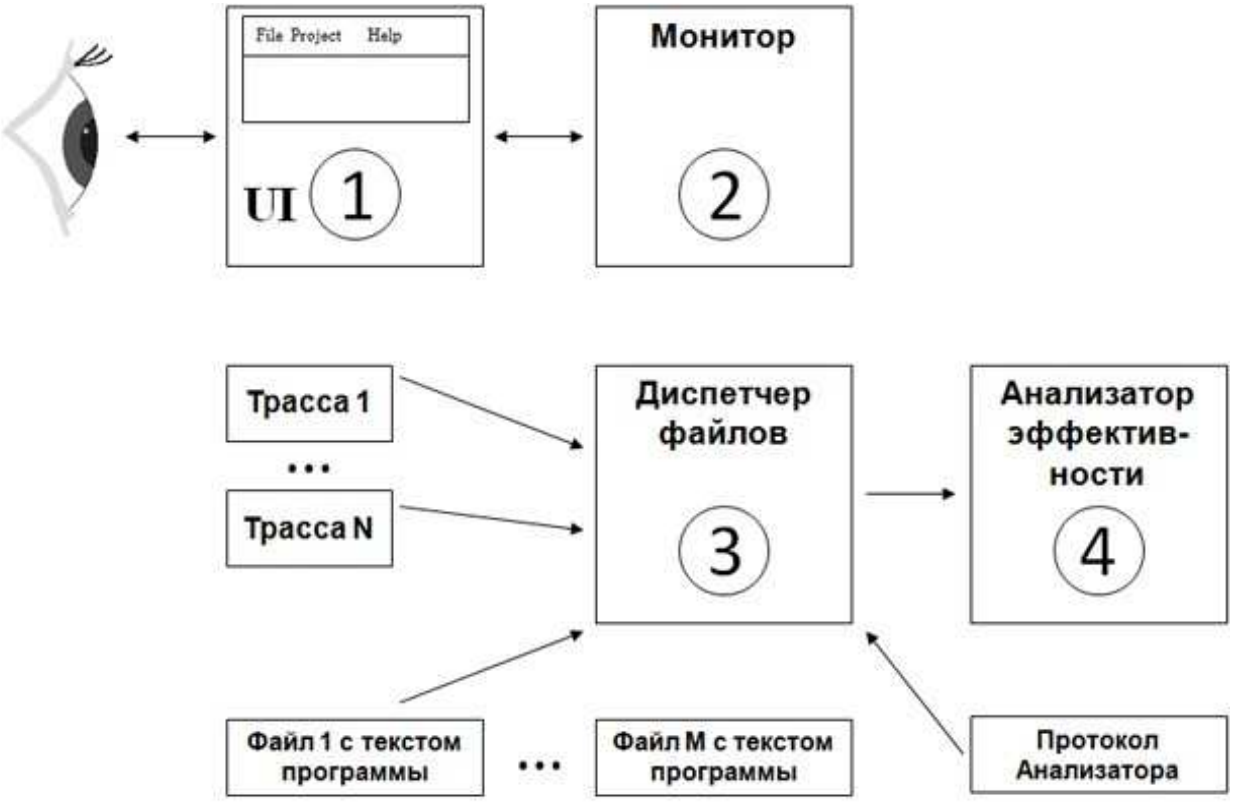

Рис. 2. Структура системы диалоговой отладки эффективности MPI-программ

Остановимся на основных функциях диалоговой системы. При использовании диалогового интерфейса для отладки эффективности параллельной программы возникает ряд проблем. Для реальных задач трассы могут быть действительно большими. К тому же, они накапливаются на той же удаленной машине, на которой идет счет задачи. Может не хватать ни времени, ни места для их локального сохранения на компьютере пользователя. Поэтому было решено разбить систему на несколько модулей. Модуль «UI» (пользовательский интерфейс) выдержан в стиле Microsoft Visual Studio. Под действиями пользователя «UI» направляет сообщения модулю «Монитор», который может находиться как на этой же локальной Windows-машине, так и на удаленной машине (в зависимости от того, где находятся трассы). Реагируя на полученные от «UI» сообщения, «Монитор» запускает модуль «Диспетчер файлов» на той машине, на которой находится сам, и передает ему запрос пользователя. В зависимости от запроса «Диспетчер файлов» совершает действия, необходимые для удовлетворения этого запроса. В том числе может, например, запустить анализатор эффективности, выдать

необходимый пользователю кусок его протокола, а также кусок исходного текста или трассы и т. д. При работе по такой схеме уже не требуется перекачивание всех трасс на локальную машину. Тем не менее, пользователь имеет возможность просматривать те куски файлов разных типов, которые ему требуются.

На рис. 3 представлены примеры различных окон диалоговой системы. Например, таких, как окна с трассами, с файлами исходного текста программы, с протоколом анализатора эффективности и т. д. Можно видеть, что некоторые строки подсвечены цветом, что иллюстрирует такую функциональность рассматриваемой диалоговой системы как обнаружение связей и работа с переходами по ним.

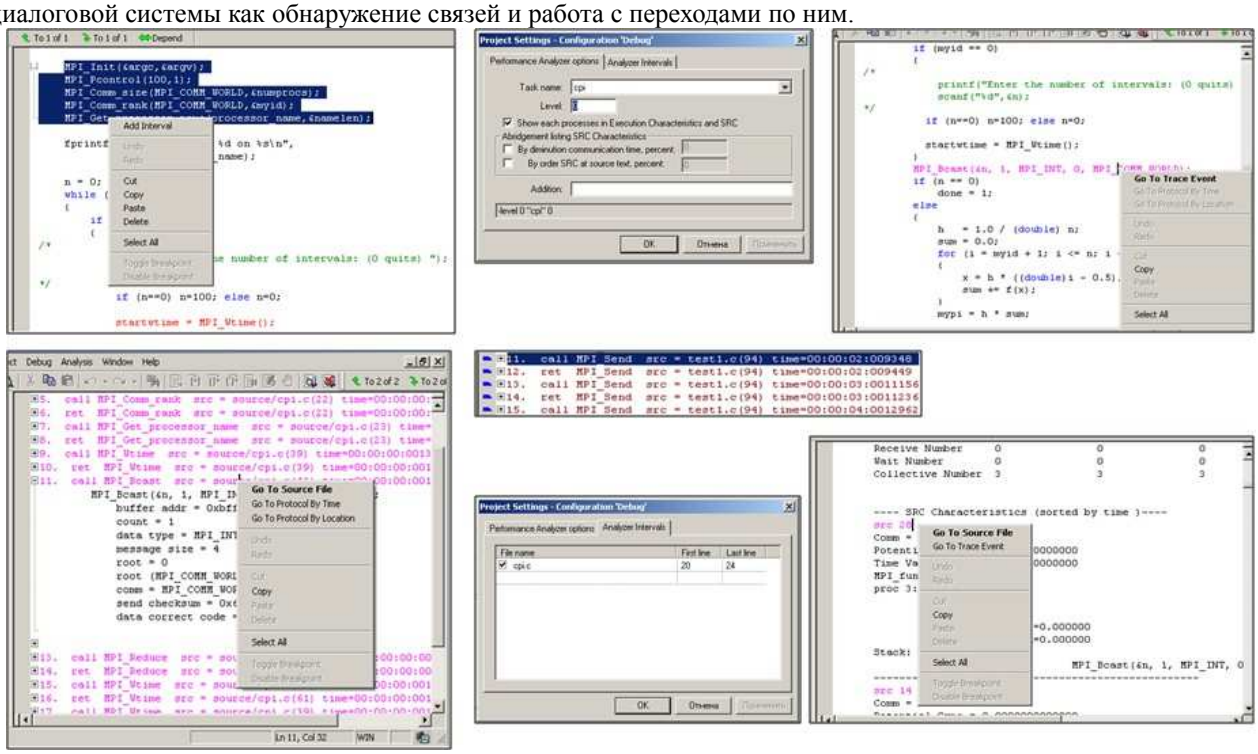

Рис. 3. Пример системы диалоговой отладки эффективности MPI-программ

Связи – это соответствия между событиями в трассах, строками в исходных текстах программы, а также в протоколе анализатора эффективности.

Например, каждый вызов MPI-функции в исходном тексте программы может попасть в трассы, вообще говоря, несколько раз, если, например, он стоит внутри цикла. Соответственно, одной строке в исходном тексте программы может соответствовать несколько событий в трассе. Но каждому событию в трассе соответствует одна строка в исходном тексте программы.

Также каждый вызов MPI-функции может производиться из разных мест программы, если, например, он входит в состав куска кода, оформленного в виде процедуры. Следовательно, каждому MPI-вызову может соответствовать несколько точек SRC. То есть одной строке в исходном тексте программы может соответствовать несколько строк протокола анализатора эффективности. Но каждой строке протокола соответствует одна точка SRC, а значит одна строка исходного текста.

И, наконец, каждая точка SRC может быть выполнена несколько раз, а значит одной строке в протоколе анализатора может соответствовать несколько событий в трассе. Но каждому событию в трассе соответствует одна строка в протоколе анализатора. Система связей в диалоговой системе проиллюстрирована на рис. 4.

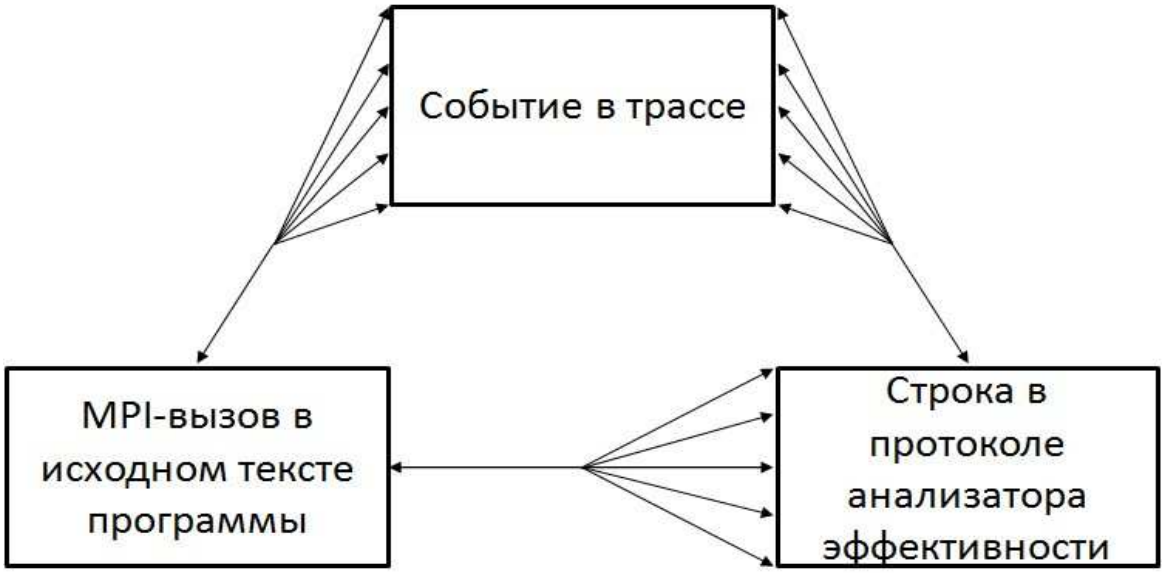

Рис. 4. Система связей диалоговой системы отладки эффективности MPI-программ

Диалоговая система позволяет по щелчку кнопкой мыши осуществлять переходы по любой из связей, показывая пользователю соответствующие места в интересующих его файлах, что облегчает работу с выдаваемой анализатором эффективности информацией.

Динамическое вычисление характеристик эффективности

Реализованные в рамках данной работы средства динамического вычисления характеристик эффективности MPI-программ функционируют следующим образом. В процессе выполнения программы в оперативной памяти процессоров накапливается информация с временными характеристиками. Для этого необходимо предварительно скомпилировать задачу, используя специальную команду, обеспечивающую линковку программы со специальной библиотекой, осуществляющей перехват, накопление информации и подсчет временных характеристик и коэффициента эффективности. По завершении выполнения программы эта информация выталкивается в файл, имеющий ту же структуру и то же содержание, что и протокол анализатора эффективности.

Выводы

Были достигнуты следующие результаты:

- Разработаны инструменты, вычисляющие и предоставляющий программисту информацию о временных характеристиках выполнения MPI-программы и коэффициенте его эффективности.
- Обеспечена возможность детализировать выдаваемую информацию с интересующим уровнем детализации при помощи механизма интервалов и точек SRC
- Осуществлена поддержка инструмента прогнозирования для стабилизации временных характеристик
- Разработана и реализована система диалоговой отладки MPI-программ, осуществляющая переходы по всем существующим связям в системе исследуемых файлов и позволяющая не переносить большие объемы информации на локальную машину, а подкачивать ее только интересующими пользователя частями.

Работа поддержана программами Президиума РАН №14 и №17, грантом РФФИ №11-01-00246.

ЛИТЕРАТУРА:

1. В.Ф. Алексахин, К.Н. Ефимкин, В.Н. Ильяков, В.А. Крюков, М.И. Кулешова, Ю.Л. Сазанов. Средства отладки MPI-программ в DVM-системе. Научный сервис в сети Интернет: Труды Всероссийской научной конференции (19-24 сентября 2005 г., г.Новороссийск). -М.: Изд-во МГУ, 2005, стр. 113-115.
2. А.Б. Бугеря, П.И. Колударов, М.И. Кулешова. Диалоговый интерфейс для отладки параллельных программ. Научный сервис в сети Интернет: технологии параллельного программирования: Труды Всероссийской научной конференции (18-23 сентября 2006 г., г.Новороссийск). -М.: Изд-во МГУ, 2006, стр. 86-88.
3. А.Б. Бугеря, П.И. Колударов, М.И. Кулешова. Средства отладки эффективности MPI-программ. Научный сервис в сети Интернет: решение больших задач: Труды Всероссийской научной конференции (22-27 сентября 2008 г., Абрау-Дюрсо). -М.: Электронное издание, 2008, стр. 89-90.
4. М.И. Кулешова. Средства отладки MPI-программ. Информационные и математические технологии в науке и управлении: Труды XIV Байкальской Всероссийской конференции. Часть II. -Иркутск:

ИСЭМ СО РАН, 2009, стр. 181-187.
5. Intel® Cluster Toolkit 2011. Release Notes [Электронный ресурс]: официальный сайт. -URL: http://software.intel.com/sites/products/documentation/hpc/ict/release_notes.pdf (дата обращения: 30.05.2011)
6. PGI® Profiler Guide. Parallel Profiling for Scientists and Engineers. Release 2011 [Электронный ресурс]: официальный сайт. - URL: http://www.pgroup.com/doc/pgprofug.pdf (дата обращения: 30.05.2011)
7. DVM-система. Отладка эффективности параллельных программ. Руководство пользователя [Электронный ресурс]: официальный сайт. - URL: http://www.keldysh.ru/dvm/dvmhtm1107/rus/index.html (дата обращения 30.05.2011)
8. KOJAK Trace-Analysis Environment [Электронный ресурс]: официальный сайт. - URL: http://icl.cs.utk.edu/kojak/custom/index.html?lid=57&slid=122 (дата обращения: 30.05.2011)
9. B. Mohr, F. Wolf. Automatic Performance Analysis of Hybrid OpenMP/MPI. Programs.Research Centre Juelich (FZJ) Central Institute for Applied Mathematics (ZAM). 2002.