УДК 004.272:004.4

Т-СИСТЕМА С ОТКРЫТОЙ АРХИТЕКТУРОЙ

С.М. Абрамов, А.И. Адамович, А.В. Инюхин, А.А. Московский, А.В. Роганов.
Ю. А. Шевчук, Е. В. Шевчук.
ИПС РАН, г. Переславль-Залесский, Россия

Приводится описание средства автоматического динамического распараллеливания «Т-система», разработанное в рамках суперкомпьютерной программы «СКИФ» Союзного государства России и Беларуси. Даётся краткое описание архитектуры системы, языка программирования Т++, обсуждаются отличительные особенности Т-системы. Приводится описание новых возможностей Т-системы, разработанных в течение последних двух лет, в том числе поддержка распределённых и GRID-вычислений.

Введение

Одной из ключевых компонент кластерного уровня суперкомпьютеров программы «СКИФ» является система автоматического распараллеливания вычислений – «Т-система». За годы выполнения программы «СКИФ» была реализована большая часть идей, заложенных основателями разработки Т-системы ещё в начале 80-х годов. Созданная версия Т-системы обладает открытой и расширяемой архитектурой, легко адаптируемой к стремительно меняющимся аппаратным платформам современных суперкомпьютеров; Т-приложения – приложения, написанные на входном языке программирования Т-системы Т++ - могут быть запущены на широком наборе программно-аппаратных конфигураций, в том числе, на различных мета-кластерных системах.

Т-система основывается на парадигме функционального программирования для обеспечения динамического распараллеливания программ. При этом в Т-системе найдены и реализованы весьма эффективные формы как для собственно организации параллельного счета (синхронизация, распределение нагрузки), так и для сочетания функционального стиля с императивными языками программирования (в Т-системе используется гладкие расширение привычных для большинства программистов языков C, C++, Fortran).

Работа выполнена в рамках программы «СКИФ» Союзного государства и при поддержке программы фундаментальных научных исследований ОИТВС РАН "Высокопроизводительные вычислительные системы, основанные на принципиально новых методах организации вычислительных процессов" и программы фундаментальных исследований Президиума РАН "Разработка фундаментальных основ создания научной распределенной информационно-вычислительной среды на основе технологий GRID"

Отличительные черты Т-системы с открытой архитектурой (Open TS).

За последние два года Т-система была существенно доработана с учетом опыта её внедрения и эксплуатации в ходе первых трёх лет программы «СКИФ». Современная версия Т-системы имеет ряд ярких отличительных черт, прежде всего, к ним относятся:

1) Минимальные отличия диалекта языка программирования Т++ - входного языка Т-системы - от базового языка C++. Различия так невелики, что поддерживается режим компиляции Т-программ без использования Т-системы – с генерацией последовательно исполняемого кода.

2) Микроядерная архитектура среды исполнения параллельных программ с возможностью динамического расширения.
3) Динамическое связывание Т-приложений с используемой средой коммуникаций. Поддерживается пересылка данных по семи различным реализациям MPI и PVM.

Поскольку для большинства Т-приложений производительность вычислений является критическим фактором, разработаны две технологии компиляции Т-программ:

1) Конвертирование программ на Т++ в код на С++ с последующей компиляцией оптимизирующим компилятором, наилучшим для целевой платформы.
2) Компиляции на основе фронт-енда компилятора GNU, обеспечивающую наиболее полную поддержку входного языка Т++ и ряда часто используемых расширений С++.

Можно утверждать - несмотря на то, что Open TS достаточно новая разработка, она обладает рядом преимуществ над более известными и широко используемыми аналогичными системами, например, Charm++ [1], Cilk [2].

Архитектура среды исполнения Т-системы

Архитектура системы OpenTS построена по “микроядерному” принципу: так называемая исполняемая спецификация содержит в себе определение всех необходимых базовых сущностей (классов и шаблонов классов в терминологии С++), а также реализацию методов классов по-умолчанию. Глубокая проработка отдельных аспектов (реализуемая зачастую как переопределение методов) выполняется в отдельных модулях – расширениях Т-системы, которые, как и пользовательская программа, взаимодействуют с исполняемой спецификацией--микроядром.

Микроядро системы, называемое «Т-Суперструктурой», или функционально-ориентированная супервычислительная надстройка, структурно организована как три программных уровня S, M и T:

S-уровень

Назначение S-уровня:

- Создать высокопроизводительную надстройку над стандартными средствами управления памятью (кэширующий аллокатор, механизмы подсчета ссылок) и потоками исполнения (кэширование тредов).
- Ввести абстракцию данных и обеспечить высокоэффективый разноприоритетный транспорт для их доставки в виде активных сообщений.
- Поддержать “суперпамять”, или распределенную, программно-управляемую память, доступную для всех существующих в распределенной супервычислительной среде потоков.

S-уровень может быть по-разному реализован для разных архитектур.

M-уровень

Назначение M-уровня:

- Поддержать мобильные потоки исполнения, объекты и ссылки
- Поддержать “копирование по необходимости” (Copy-On-Write) для данных суперпамяти.
- Поддержать операцию блокирования и освобождения мобильных объектов.

- Поддержать иерархию досок объявлений с информацией о нераспределенных задачах и незадействованных процессорных и прочих ресурсов.

T-уровень

Назначение T-уровня:

- Реализовать понятия неготового значения и ссылки на неготовое значение
- Реализовать понятие T-функции – мобильной функции, являющейся гранулой параллелизма.

В качестве базового уровня поддержки сущностей T-системы была создана и реализована концепция «суперпамяти» - специализированной общей памяти для организации обменов данными между T-функциями. Для T-величин, хранящихся с ячейках «суперпамяти» реализован подсчёт ссылок и распределённая сборка мусора. С целью устойчивой работы сборщика мусора в условиях сетевых коммуникаций, реализован нетривиальный подсчёт ссылок на объекты.

T-система с точки зрения пользователя

Язык программирования T++ является синтаксически и семантически гладким языковым расширением стандартного языка программирования C++. Под синтаксической и семантической гладкостью здесь понимается, прежде всего, наличие естественного вложения конструкций языка C++ в расширенный (по отношению к нему) синтаксис и семантику языка T++.

Перечислим новые добавленные в язык C++ конструкции; контексты, в которых возможно их употребление, и то, как они влияют на выполнение итоговой (то есть откомпилированной с помощью соответствующего компилятора и запущенного в соответствующей среде исполнения) программы.

Всего к стандартному набору C++ добавляется пять ключевых слов: **tfun**, **tval**, **tptr**, **tout**, **tct** и две новых стандартных функции: **tdrop, twait** Только две новых стандартных функции **twait** и **tdrop** необходимы для выполнения специфических операций, у которых нет прямых аналогов в языке C++. Ниже смысл и контексты применения каждого ключевого слова рассмотрены подробнее.

tfun - атрибут, который можно указать непосредственно перед описанием типа функции. Функция не может являться методом; это должна быть обычная функция. Описанная с помощью этого ключевого слова функция называется T-функцией.

tval – атрибут, который можно указать непосредственно перед описанием типа переменной. Описанная с помощью этого ключевого слова переменная называется T-переменной; В качестве значения T-переменная содержит неготовую величину (T-величину).

tptr – T++-аналог для определения указателей (ссылок). Используется для описания глобальных указателей в структурах данных.

tout - атрибут, который можно указать непосредственно перед описанием типа выходного аргумента T-функции. T++-аналог для определения аргументов, передаваемых по ссылке ('&') для их дальнейшей модификации в теле функции.

tct – явное определение T-контекста. Служит для определения дополнительных свойств T-сущностей (специфических сущностей, поддерживаемой T-системой).

tdrop – стандартная функция от одного аргумента. Может быть вызвана от любой T-величины.

twait - стандартная функция от двух аргументов: T-сущности и паттерна событий. Возвращает статус произошедших с T-сущностью соответствующих указанному паттерну событий.

Пример простейшей программы на T++ – вычисление заданного числа Фибоначчи - приведён ниже. Этот алгоритм является широко распространённым тестом на производительность систем динамического распараллеливания. В тесте порождается огромное количество параллельных гранул, которые равномерно распределяются между доступными вычислительными ресурсами.

```
tfun int fib(int n)
{
        if (n<2) return 1;
        return (fib(n-1)+fib(n-2));
}
tfun int main (int argc, char *argv[])
{
        int n = atoi(argv[1]);
        printf("Fibonacci %d is %d\n",n,(int)fib(n));
        return 0;
}
```

Используется единственная T-функция, fib, рекурсивно вызывающая сама себя. Поскольку результатом T-функции является T-величина, необходимо явное преобразование типов (int)fib(n) при вызове функции printf. Такое преобразование вызывает ожидание потока функции main до тех пор, пока не будет вычислен результат функции fib(n). Сам вызов fib(n) порождает граф (дерево) вызовов fib с меньшими значениями входного аргумента – причем ветви дерева могут вычисляться параллельно.

Компиляция программы осуществляется при помощи либо конвертера (t++) либо компилятора tg++. В нашем примере используется конвертер:

```
t+ -o fib0 fib0.tcc
```

Запуск программы, в общем случае, осуществляется при помощи стандартных средств запуска параллельных приложений, например, в случае SCALI MPI:

```
mpirun –np 4 ./fib
```

Возможен также и запуск приложения в режиме монопроцессора, например, при отладке:

```
./fib
```

При запуске, консольный вывод приложения содержит статистическую информацию о выполнении программы: общее время исполнения, число запущенных T-функций и другую. Конвертером T++ поддерживается компиляция приложения с отладочным выводом, запуск отладочной версии порождает, во время выполнения, вывод информации о создающихся T-переменных, запуске T-функций и ожидании неготовых величин.

Приложения и пользовательские библиотеки

К настоящему времени, с использованием современной версии T-системы созданы и создаются ряд приложений, в том числе в ходе работ по мероприятиям программы «СКИФ». Вот некоторые из них:

- Инструментальное средство построения динамических интеллектуальных систем «Миракл» - в рамках выполнения мероприятия 20 программы «СКИФ»
- Система «MultiGen» - оценка биологической активности веществ [3]

- Программа численного моделирования разрушения кильватерной волны [4]
- Программа для численной оценки скорости приближения к глобальному аттрактору уравнений Навье-Стокса в задаче о двумерной каверне [5]
- Программа моделирования фолдинга белков [6]
- Программа синтеза радиолокационного изображения из голограммы РЛС космического базирования – в рамках выполнения работ по мероприятию 13 программы «СКИФ».

OpenTS и GRID-вычисления

Разработка вычислительных приложений для распределённых вычислительных сред и , в частности, для GRID, является достаточно непростой задачей в первую очередь из-за сложности балансировки нагрузки в гетерогенной среде. Т-система, в силу обеспечения динамического распараллеливания, предоставляет программисту возможности прозрачного использования гетерогенных конфигураций: сложность выравнивания нагрузки «скрыта» от него в планировщике Т-системы, распределяющего нагрузку между процессорами в кластере или мета-кластере. Т-система предоставляет следующие возможности для обеспечения распределённых вычислений:

1. Поддержка мета-кластерных реализаций MPI: MPICH-G2, IMPI, PACX-MPI
2. Обеспечение возможности пользовательского расширения стандартного алгоритма планирования – через механизмы расширения Т-системы.

Заключение

К сожалению, краткий объём статьи не позволил нам подробно остановится на многих аспектах Т-системы. В настоящее время, Open TS распространяется бесплатно под лицензией BSD, и доступнка для загруски по адресу: http://t-system2.polnet.botik.ru/misc/openTS.tar.bz2. Поддержка осуществляется сообществом разработчиков, чей список рассылки расположен по адресу opents@botik.ru.

Литература

1. L. V. Kaleev, Sanjeev Krishnan “Charm++: Parallel Programming with Message-Driven Objects” // in "Parallel Programming using C++", by Gregory V. Wilson and Paul Lu. MIT Press, 1996. pp 175-213

2. Matteo Frigo, Charles E. Leiserson, Keith H. Randall “The Implementation of the Cilk-5 Multithreaded Language” // 1998 ACM SIGPLAN Conference on Programming Language.

3. Потемкин В.А., Арсламбеков Р.М., Белик А.В., Гуччионе С “Параллельная версия алгоритма MULTIGEN” // Сборник материалов конференции “Информационно-вычислительные технологии в решении фундаметнальных прикладных и научных задач» ИВТН-2003, стр. 11.

4. Горбунов Л.М., Фролов А.А., Чижонков Е.В. «О критерии существования регулярной кильватерной волны»// XXXI Звенигородская конференция по физике плазмы и управляемому термоядерному синтезу. г. Звенигород, 16 - 20 февраля 2004

5. A. Kornev, “ On globally stable dynamic processes” //Russian Journal of Numerical Analysis and Mathematical Modelling, Volume 17, No. 5, p 472

6. Московский А.А., Вановский В.В., Грановский А.А., Немухин А.В. “Создание двухуровневой схемы распараллеливания для распределенных вычислений при моделировании фолдинга белков.”// Международная конференция «Распределенные вычисления и Грид-технологии в науке и образовании» 29 июня - 2 июля 2004 г. г.Дубна, Россия