УДК 519.713

С.Л. ХАРЧЕНКО

Харьковский национальный университет радиоэлектроники

УСКОРЕНИЕ МОДЕЛИ ПОВЕДЕНИЯ СИСТЕМЫ УПРАВЛЕНИЯ И ОРГАНИЗАЦИЯ ЕЁ ТЕСТИРОВАНИЯ

В статье рассмотрены вопросы возможности ускорения модели поведения управляющей системы за счет использования многопроцессорной системы с параллельным асинхронным запуском процессов. Рассмотрены проблемы тестирования программного обеспечения управляющих систем и предложены новые подходы к организации тестирования программного обеспечения систем управления. Основной акцент ставится на тестирование управляющего графа модели поведения, проводимого на ранних этапах проектирования системы управления и возможности генерации операторов управляющей программы по её формальному описанию, выполненному в модели поведения.

Ключевые слова: *модель поведения, асинхронные параллельные процессы, многопроцессорные системы управления, управляющий граф модели поведения, информационный граф модели поведения, генерация операторов программы управления по формальному описанию.*

Введение

Одним из основных требований к современным системам управления является надежная работа в реальном масштабе времени, т.е. гарантированное исполнение запроса за некоторый заранее определенный промежуток времени. Такие условия позволяют создавать алгоритм управления системой таким образом, что все подсистемы будут обслужены в некоторой очередности с помощью одного процессора или будет использоваться многопроцессорная система, в которой алгоритм управления будет построен на основе реализации последовательно-параллельных процессов.

В системах управления, в большинстве случаев, все подсистемы одновременно не инициируются, а имеют некую последовательность запуска. Для каждой из подсистем устанавливается интервал времени, в течение которого должна быть реализована задача подсистемы, что и обуславливает применение надежных систем, которые имеют возможность запускать асинхронные параллельные процессы или строить цепочки последовательно-параллельных процессов.

Традиционные подходы к созданию программ, в этом случае, имеют ряд недостатков, главным из которых является сложность визуализации параллелизма исполнения программного кода и существенное влияние человеческого фактора. Поэтому наиболее предпочтительной методологией создания прикладного программного обеспечения для систем управления должна быть методология, которая основана на принципах машинной генерации последовательности операторов программы по формальному описанию.

Можно предположить, что формальная модель поведения, в этом случае, может быть рассмотрена как описание будущей программной системы. А использование компонентного подхода, где каждая компонента может быть рассмотрена как процесс в системе управления, позволит упростить визуализацию организации параллелизма.

Соблюдение принципов иерархической декомпозиции модели поведения позволит получить послойную схему и определить принципы взаимодействия процессов в системе. А форма записи модели поведения системы даст возможность отслеживать трассировку последовательности выполненных действий, что покажет маршрут прохождения в зависимости от внутреннего состояния и внешних воздействий на систему управления.

Постановка задачи исследования

На этапе создания технического задания определить ускорение модели поведения системы управления при использовании многопроцессорных систем относительно однопроцессорного решения. Провести анализ методологии тестирования надежных систем и предложить способ тестирования модели поведения (алгоритма) управляющей программы системы на ранних этапах жизненного цикла.

Возможность ускорения модели поведения управляющей системы

При функционировании систем управления возможно возникновение ситуации, когда реализация задачи управления жестко регламентирована

интервалом времени. Например, анализ параметров функционирования и управление объектом в аварийной ситуации. Может случиться так, что решение на основе применения последовательно алгоритма, реализуемого на одном процессоре, не сможет обеспечить временные характеристики для принятия управленческого решения. Следовательно, решение управленческой задачи требует распараллеливания алгоритма управления и применение многопроцессорной системы.

Рассмотрим модель поведения [1], как эквивалентную запись алгоритма. Одним из достоинств модели поведения является наличие возможности декомпозиции её на компоненты. В общем случае, компонентой может быть любая подсистема, как и система в целом. Если предположить, что квант времени, необходимый для исполнения компоненты равен $\Delta t_{i\text{комп}}$, то для линейного алгоритма время реализации управленческой задачи будет необходимо затратить [3]

$$T_{1\ \text{общ}} = \sum_{i=1}^{m+1} \Delta t_{i\text{комп}},$$

где m – число подсистем в системе управления.

Проводя сравнение последовательного и эквивалентного ему алгоритма с распараллеливанием, можно утверждать, что суммарное время исполнения, в этом случае, будет зависимо от максимального значения кванта времени, необходимого для исполнения компоненты на участке распараллеливания. А если предположить, что это главная управляющая программа в системе управления, то время решения управленческой задачи на участке распараллеливания можно определить как $\max \Delta t_{i\text{комп}}$ или как $T_{\text{р.главная}}$. Это утверждение будет верно, в случае, если количество использующихся процессоров равно n+1, так как число процессоров должно быть на единицу больше, чем число активированных подсистем системы на «критическом участке» алгоритма управления. При проектировании алгоритма можно определить максимальное число одновременно работающих подсистем в системе управления, т.е. определить потребность в максимальном количестве процессоров необходимых для системы управления объектом.

Так как количество аппаратных средств определено для критического участка алгоритма, то по мере освобождения вычислительных ресурсов можно будет их задействовать в процессе управления другими подсистемами, тем самым будет обеспечено повторное использование высвободившихся вычислительных средств.

Следовательно, для оценки ускорения модели поведения можно воспользоваться традиционным отношением, определяющим ускорение параллельного алгоритма

$$S_{n+1} = \frac{\text{T выполнения модели на одном процессоре}}{\text{T выполнения модели на n +1 процессорах}}$$

Если результат вычисления S_{n+1} будет близок к n - можно утверждать, что поставленная задача на ускорение работы модели поведения системы была достигнута.

Если предположить, что проектировщик пойдет по пути декомпозиции системы управления на m подсистем, то ему потребуется создать m+1 автономных систем управления, для решения общей поставленной задачи. Такой подход к решению задачи управления может существенно снизить общие показатели системы управления, например, её весогабаритные параметры.

Если модель поведения рассматривать как модифицированную запись алгоритма, то можно определить показатели эффективности и стоимости вычислений для параллельного алгоритма [4].

Проблемы тестирования работоспособности систем управления

При реализации систем управления затраты на тестирование готовой системы составляют значительную часть от общих затрат на создание и внедрение системы. Поэтому, вопросы тестирования занимают особое место в разработке, тем более что для надежных систем определенного класса нельзя применять вероятностные методы оценки надежности программного обеспечения. Например, для систем, которые включаются в контур управления жизнеобеспечения человека и т.п. Следовательно, решение проблемы автоматизации тестирования для таких систем имеют особую актуальность.

Если рассматривать возможность расщепления модели поведения на два ориентированных графа – информационный и граф управления, то сам собой напрашивается вывод о том, что формальная запись информационного графа может быть трансформирована по заранее определенным правилам в последовательность операторов языка программирования. Очевидно, что машинная трансформация по строгим правилам исключит ошибки, которые традиционно относятся к «ошибкам человеческого фактора», а это большая часть ошибок, которые выявляются на этапе комплексного тестирования.

Отсюда следует, что в случае реализации машинной трансформации информационного графа в операторы языка программирования на этапе тестирования модели поведения основное внимание необходимо уделить выявлению ошибок графа управ-

ления. Сформулируем основные требования к тестированию графа управления.

Первым требованием должно быть наличие уникальной разметки вершин и ребер графа. Это требование объясняется тем, что при реализации тестов необходимо обойти все вершины графа для выявления «недостижимых вершин» или «тупиковых вершин». Вторая составляющая требований к тестированию - прохождение по всем ребрам графа. При выполнении этих условий будет достигнут полный охват элементов древа графа.

Выявив «недостижимые» и «тупиковые» вершины дерева графа и определив перечень ребер, которые «выпали» при тестировании мы получим перечень ошибок управляющего графа, компоненты модели поведения. После соответствующего анализа эти ошибки могут быть устранены. Только после их устранения можно приступать к проектированию следующего уровня декомпозиции и только на условиях полной работоспособности разработанного ранее слоя декомпозиции модели управления. Обращает на себя внимание и то, что это должно выполняется на этапах проектирования, которые можно отнести к этапу составления технического задания, т.е. ещё отсутствуют затраты на проектирование и программирование.

Так как модель поведения компонентная и разделена на слои проектирования, то общий тестирующий модуль может быть построен путем «послойной подстановки» тестов нижележащих слоёв декомпозиции. Следовательно, один и тот же набор тестов можно использовать как при локальном, так и комплексном тестировании, в котором главным объектом тестирования будет управляющий граф модели поведения.

Аналогичное утверждение относится и к выявлению ребер графа, которые были «не пройдены» при тестировании. Это тоже свидетельствует об ошибке в управляющем графе модели поведения.

Следующей проблемой, которую необходимо рассмотреть – это циклические структуры в графе поведения и «правильность» их тестирования. Очевидно, что любой управляющий алгоритм имеет циклические структуры, которые представлены в том или ином виде. При тестировании необходимо получить утверждение о том, что количество итераций в цикле ограничено и то, что все ветви графа в циклической структуре пройдены.

Кроме циклических структур в графе управления могут быть параллельные структуры, которые взаимодействуют между собой по определенным правилам. Основным ограничением для таких структур должно быть правило - отщепление и завершение процесса должно быть выполнено на одном слое декомпозиции модели поведения. При этом необходимо соблюдать требования информационного обеспечения отщепляемых процессов.

Организация тестирования управляющего графа модели поведения

Традиционно, математическую модель будущей системы, на которой проводят тестирование принятых решений, сводит к построению модели на основе сети Петри, на которой выполнена разметка. Если получено утверждение, что модель адекватна объекту моделирования - производятся испытания работоспособности принятых проектных решений. Считается, что если достигнута та или иная вершина, в зависимости от разметки и входных параметров, то задача проверки правильности работы системы выполнена.

В связи с тем, что модель поведения может иметь некоторое множество внутренних состояний и значения входных параметров могут изменяться в некоторых диапазонах, то для получения заключения о работоспособности модели системы необходимо иметь некоторое количество разметок построенной сети Петри. Выполнив на каждой из полученных разметок комплекс испытаний, который соответствует диапазону изменяющихся входных данных, можно получить предварительное заключение о правильности принятых проектных решений.

Реализации такого подхода требует уверенности в том, что модель, построенная на основе сети Петри адекватна реальному объекту. Необходимо обратить внимание и на то, что для случая программной реализации управляющей системы, у которой уже имеется управляющий граф, проводить испытания на ином объекте, для получения заключения о работоспособности нецелесообразно. Поэтому, следует рассмотреть управляющий граф как «белый ящик» и проверять его работоспособность.

Для проверки работоспособности управляющего графа модели поведения рассмотрим принципы его построения. Так как выбран компонентный подход к проектированию, то на каждом этапе работ, для компоненты управляющего графа уже сформирован список входных и выходных параметров, перечень внутренних переменных, в которых отражена сущность изменения внутреннего состояния компоненты и определены возможные диапазоны изменений всех переменных и входных параметров. Следует учитывать и то, что форма их записи соответствует требованиям к синтаксису и семантике языка bash или подобного ему языка, т.е. она пригодна для интерпретации в UNIX подобной ОС.

Кроме этого, первоначальным условием было наличие разметки управляющего графа, что обеспечивает возможность автоматического формирования

трассировки прохождения по дереву графа при тестовом испытании. Всё это обеспечивает возможность специалисту, который проектирует данный компонент системы, разработать ряд тестов которые обеспечат покрытие всех ветвей управляющего графа компоненты модели поведения, при соблюдении некоторого набора условий и ограничений, важных для проверки работоспособности управляющей системы.

Формирование тестовых значений переменных, которые относятся как к внутренним состояниям, так и внешним воздействиям, позволяет заранее сформировать трассу прохождения по графу управления. Это в свою очередь, при выполнении испытания позволяет проследить соблюдение маршрута прохождения по графу в автоматическом режиме. Поэтому любое отклонение от маршрута может быть зафиксировано с прерыванием циклограммы тестирования.

Следовательно, при выполнении комплекса тестирования компоненты может быть сформирован такой перечень всех результатов, на котором можно выполнить проверку покрытия вершин и ветвей управляющего графа. Если при завершении испытаний будут выявлены «тупиковые» или «недостижимые» вершины граф или ребра, через которые отсутствуют маршруты прохождения, то это будет свидетельствовать о том, что при проектировании компоненты были допущены ошибки, которые необходимо устранить перед выполнением следующего шага проектирования.

Работой над компонентой управляющего графа можно считать завершенной, если выполнено сохранение разработанных тестов, которые обеспечивают «покрытие» всех вершин и использование всех рёбер для переходов из вершины в вершину.

Исходя из того, что граф управления и тесты построены по компонентному принципу, то путем соответствующей подстановки можно получить управляющий граф модели поведения системы. Если к нему применить комплекс тестовых заданий, которые учитывают возможные внутренние состояния управляющей системы, то можно сформировать перечень вершин и ребер, которые отображают трассы тестирования. Если на этих испытаниях будут выявлены «тупиковые» или «недостижимые» вершины или ребра графа, которые «выпали» при тестировании, то это свидетельствует о наличии потенциальной ошибки или в проектном решении или тестовых заданиях. Эти ошибки необходимо устранить перед этапом генерации управляющей программы.

Заключение

Реализация методологии проектирования с использованием модели поведения будущей системы позволяет:

- сформулировать требования к техническим характеристикам вычислительной платформы (количество необходимых для функционирования управляющей системы процессоров), которые обеспечат решение задачи за заранее определенный интервал времени;
- выполнить тестирование модели поведения на информационное обеспечение отщепляемых параллельных процессов и реализации ветвлений;
- подготовить тесты для проверки программной реализации системы управления.

Если реализовать генерацию операторов программы системы управления по информационному графу, то можно получить существенную экономию средств и сокращение интервала времени, необходимых на реализацию проекта, с гарантированным повышением надежности управляющей системы.

Литература

1. Харченко, С.Л. Язык проектирования технического задания систем управления [Текст] / С.Л. Харченко // Восточно-Европейский журнал передовых технологий. – 2011. – № 1/3 (49). – С. 10–16.

2. Барский, А.Б. Параллельные процессы в вычислительных системах. Планирование и организация [Текст]/ А.Б. Барский. – М.: Радио и связь, 1990. – 256 с.

3. Калашников, В.И. Параллельное программирование [Текст]: учебн. пособие / В.И. Калашников.– Харьков: НТУ "ХПИ", 2004. – 320 с.

4. Качко, Е.Г. Параллельное программирование [Текст]: учебн. пособие / Е.Г. Качко. – Харьков: Изд-во «Форт», 2011. – 528 с.

5. Способ отладки программного обеспечения [Текст] / С.Л. Харченко, Ю.Ш. Биглов и др. // ПТО. – 1984. – №11. – С. 12–18.

Поступила в редакцию: 26.06.2012

Рецензент: д-р техн. наук, проф, декан факультета компьютерных наук В.П. Машталир, Харьковский национальный университет радиоэлектроники

ПРИСКОРЕННЯ МОДЕЛІ ПОВЕДІНКИ СИСТЕМИ УПРАВЛІННЯ І ОРГАНІЗАЦІЯ ЇЇ ТЕСТУВАННЯ

С.Л. Харченко

У статті розглянуті питання можливості прискорення моделі поведінки керуючої системи за рахунок використання багатопроцесорної системи з паралельним асинхронним запуском процесів. Розглянуто проблеми тестування програмного забезпечення керуючих систем та запропоновано нові підходи до організації тестування програмного забезпечення систем управління. Основний акцент ставиться на тестування керуючого графа моделі поведінки, що проводиться на ранніх етапах проектування системи управління та можливості генерації операторів керуючої програми з її формального опису, виконаному в моделі поведінки.

Ключові слова: модель поведінки, асинхронні паралельні процеси, багатопроцесорні системи управління, керуючий граф моделі поведінки, інформаційний граф моделі поведінки, генерація операторів програми управління по формальному опису.

ACCELERATION MODEL BEHAVIOR MANAGEMENT SYSTEM ORGANIZATION AND ITS TESTING

S.L. Kharchenko

The paper deals with the possibility of accelerating behavior of the control system through the use of a multiprocessor system running parallel asynchronous processes. The problems of software testing management systems and proposes new approaches to software testing management systems. The main emphasis is placed on the test control graph model of behavior, conducted in the early stages of designing the control system and the possibility of generating operators of the control program for its formal description by the model behavior.

Keywords: behavior, asynchronous parallel processes, multi-control system that controls the behavior graph, the graph model of information behavior, the generation of operator control program on the formal description.

Харченко Сергей Леонидович – старший преподаватель кафедры ПИ, Харьковский национальный университет радиоэлектроники, e-mail: khsln@yandex.ru