

ОПТИМИЗАЦИЯ АЛГОРИТМИЧЕСКОГО ОБЕСПЕЧЕНИЯ В ЗАДАЧАХ ПРЕОБРАЗОВАНИЯ ИНФОРМАЦИИ

д.т.н. И.В. Чумаченко, В.В. Косенко

Рассмотрена задача оптимизации алгоритмов и критерии оптимизации. Предложен способ оценки глубины алгоритма и метод построения оптимальных по глубине алгоритмических систем.

Высокая производительность современных вычислительных систем достигается усовершенствованием старых и применением новых технологий: развитие конвейеризации вычислений и предсказания переходов. Однако, при реализации конвейерной обработки возникают конфликты, которые снижают реальную производительность конвейера, которая могла бы быть достигнута в идеальном случае. Условные переходы снижают общую производительность процессора, т.к. в ожидании этого перехода конвейер работает вхолостую. В общем случае, чем глубина конвейера больше, тем больше потери на командах условного перехода, исчисляемые в тактах. Таким образом, снижение потерь от условных переходов становится критическим вопросом. Одним из путей решения данной проблемы является оптимизация структуры алгоритмов с учетом их последующей обработки на машинах с конвейерной обработкой [1].

Оптимизация алгоритмов – это процесс улучшения различных характеристик алгоритмов и процессов вычислений путем эквивалентных преобразований. Основными характеристиками, которые улучшаются в процессе оптимизации алгоритмов, являются: время работы; объем памяти, используемой алгоритмом в процессе вычислений; степень параллельности или асинхронности процессов вычислений, порождаемых алгоритмом. Эти характеристики зависят от исходных данных, поэтому различают абсолютную оптимизацию, при которой значения характеристик улучшаются для любых исходных данных, и оптимизацию усредненных величин. Задачи оптимизации алгоритмов исследуются на таких математических моделях, как дискретные преобразователи или схемы программ [2].

Задача оптимизации алгоритмов является сложной задачей, и в настоящее время отсутствуют эффективные методы ее решения. Основные направления в ее решении состоят в поиске эффективных тождественных преобразований. При конструировании автоматов на основе алгоритмических алгебр оптимизационные задачи могут быть решены на уровне квазирегулярных и рекурсивных форм алгоритмов, поскольку для них разработана развитая техника тождественных преобразований. Однако правила применения тождеств в алгоритмических алгебрах не-

достаточно формализованы, как и отыскание эффективной стратегии управления выводом при доказательстве тождественных соотношений. В связи с этим возникает задача разработки специальных алгоритмов оптимизации, основанных на применении дизъюнктивных и рациональных процедур, которые носят более или менее целевой характер.

В общем случае программа минимизации алгоритмов является многокритериальной. Сложность решения этой задачи можно охарактеризовать числом проводимых при этом промежуточных преобразований. Для оптимизации алгоритмов по числу операторов и логических условий разработаны методы, подробно описанные в работе [3]. В основе минимизации алгоритмов по числу операторов и логических условий с универсальным распределением сдвигов для алгоритмов Мили лежит реляционный метод. Перспективным направлением при оптимизации алгоритмических систем является создание специальных средств аппаратной поддержки [6].

Анализ критериев оптимальности алгоритмических структур показал, что главными критериями являются быстрдействие алгоритма и его сложность. Быстрдействие алгоритма определяется временем выполнения операторов и глубиной алгоритма. Применительно к алгоритмам, описываемым с помощью регулярных выражений, глубиной алгоритма является максимальное количество условий, которое необходимо проверить в процессе работы алгоритма.

Разработан метод оптимизации алгоритмических структур, основанный на алгебраическом подходе [4 – 6]. Основные его положения заключаются в следующем.

Пусть алгоритм описывается множеством операторов $P = (P_1, \dots, P_k)$ и множеством условий $Y = (Y_1, \dots, Y_n)$, тогда, совершенная дизъюнктивная нормальная форма алгоритма (СДНФА) имеет вид:

$$A = \Pi_1^{U_1} \vee \dots \vee \Pi_i^{U_i} \vee \dots \vee \Pi_h^{U_h},$$

где $\Pi_i^{U_i}$ - произведение соответствующих операторов, реализуемое при выполнении условия $U_i = 1$; U_i - конъюнкция соответствующих условий; h - количество членов в СДНФА.

Разобьем множество произведений операторов Π на подмножества, имеющие одинаковые значения элементов. Множество различных элементов множества Π обозначим $D = \{D_1, \dots, D_q\}$, где q – количество различных элементов.

Рассмотрим два члена СДНФА $\Pi_i^{U_i}$ и $\Pi_j^{U_j}$. Если $\Pi_i = \Pi_j$, то имеет место тождество

$$\Pi_i^{U_i} \vee \Pi_j^{U_j} = \Pi_i^{U_i \vee U_j}.$$

Поставим в соответствие множеству $D = \{D_1, \dots, D_q\}$ множество логических функций $T = \{T_1(Y), \dots, T_q(Y)\}$, соответствующих наборам

условных переменных, при которых выполняется соответствующее произведение операторов. Элементы множества T обладают следующими свойствами.

Свойство 1. $T_i(X) \& T_j(X) = 0; \quad i \neq j; \quad i = 1, \dots, q; \quad j = 1, \dots, q.$

Это свойство непосредственно вытекает из определения СДНФА, поскольку конъюнкции условий не пересекаются, то и объединение их непересекающихся подмножеств также не пересекается.

Свойство 2. $T_1(X) \vee \dots \vee T_q(X) = 1.$

Это свойство вытекает из того, что объединение элементов множества T образуют полное множество конъюнкций условий. Это свойство можно также интерпретировать как декомпозицию полного множества конъюнкций на непересекающиеся подмножества, соответствующие одинаковым произведениям операторов.

Пусть $X = \{x_1, \dots, x_n\}$ – множество условий, $P = \{P_1, \dots, P_k\}$ – множество операторов, $A(X, P)$ – заданный алгоритм, $F(X) = \{F_1(X), \dots, F_m(X)\}$, – множество логических функций. Тогда задача выбора оптимальной структуры алгоритма может быть сформулирована следующим образом: найти алгоритм $B(F(X), P)$, эквивалентный алгоритму $A(X, P)$, такой, что глубина алгоритма m минимальна.

Эта задача состоит из двух частей:

- определение минимального возможного значения глубины алгоритма,

- создание метода построения оптимальной структуры алгоритма.

Основой для определения минимального возможного значения глубины алгоритма (m) служит следующая теорема.

Теорема 1. Значение минимальной глубины алгоритма может быть определено по формуле

$$m = \lceil \log_2 q \rceil,$$

где q – количество различных произведений операторов; $\lceil \dots \rceil$ – ближайшее целое, не меньшее, чем значение в скобках.

Доказательство. СДНФА алгоритма $A(X, P)$, где X – множество условий и P – множество операторов имеет вид

$$A = \Pi_1^{U_1} \vee \dots \vee \Pi_i^{U_i} \vee \dots \vee \Pi_h^{U_h},$$

где $\Pi_i^{U_i}$ – произведение соответствующих операторов, реализуемое при выполнении условия.

С учетом разбиения на подмножества, имеющие одинаковые произведения операторов ДНФА алгоритма A можно представить в виде

$$A = D_1^{T_1} \vee \dots \vee D_i^{T_i} \vee \dots \vee D_q^{T_q},$$

где $D = \{D_1, \dots, D_q\}$ – множество различных произведений операторов, $T = \{T_1(Y), \dots, T_q(Y)\}$ – множество логических функций, соответст-

вующих наборам условных переменных, при которых выполняется соответствующее произведение операторов.

Введем в рассмотрение множество условных переменных

$$Z = \{Z_1, \dots, Z_m\},$$

а соответствующие конъюнкции этих переменных обозначим через L с индексом. Тогда ДНФ алгоритма A можно рассматривать как СДНФА относительно переменных Z :

$$A = D_1^{L_1} \vee \dots \vee D_i^{L_i} \vee \dots \vee D_q^{L_q}.$$

Количество возможных конъюнкций от m переменных равно 2^m и должно быть не меньше, чем количество различных произведений операторов q , т.е.

$$2^m \geq q;$$

откуда

$$m = \lceil \log_2 q \rceil,$$

что и требовалось доказать.

Рассмотрим метод построения структуры оптимального алгоритма.

Метод построения оптимальных по глубине алгоритмов состоит из следующих этапов.

1. Для заданного алгоритма A определяется СДНФА и множество $\Pi = \{\Pi_1, \dots, \Pi_h\}$ произведений операторов.

2. Формируется множество $M = \{\Pi_1(1), \dots, \Pi_h(1)\}$.

3. Определяется количество различных элементов множества M и частота их вхождения.

4. Выбирается элемент множества M , имеющий максимальную частоту, а если их несколько, то выбирается любой. Обозначим выбранный элемент G .

5. Производится декомпозиция алгоритма A на два алгоритма $A_{1.1}$ и $A_{1.2}$ следующим образом:

$$A = (A_{1.1} \vee A_{1.2})^{F_1(X)},$$

где $A_{1.1}$ – алгоритм, которому соответствует множество членов СДНФА, для которых $\Pi_i(1) = G$; $A_{1.2}$ – алгоритм, которому соответствуют остальные члены СДНФ алгоритма A , не вошедшие в алгоритм $A_{1.1}$; $F_1(X)$ – дизъюнкция условий, соответствующих членам СДНФ алгоритма $A_{1.1}$.

Пункты 1 – 5 выполняются до тех пор, пока произведения операторов, входящие в соответствующие множества не будут пустыми.

6. Производится минимизация полученных логических функций.

Конец.

В общем случае последовательный процесс декомпозиции алгоритма можно представить в виде

$$A = (R_{1.1}A_{1.1} \vee R_{1.2}A_{1.2})^{F_1(X)} = \\ = ((R_{2.1}A_{2.1} \vee R_{2.2}A_{2.2})^{F_{2.1}(X)} \vee (R_{2.3}A_{2.3} \vee R_{2.4}A_{2.4})^{F_{2.2}(X)})^{F_1(X)} =$$

$$= (((R3.1A3.1 \vee R3.2A2.2)^{F3.1(X)} \vee \\ \vee (R3.3A3.3 \vee R3.4A3.4)^{F3.2(X)})^{F2.1(X)} \vee \dots)^{F1(X)}.$$

В приведенном описании индексы состоят из двух значений, разделенных точкой. Первое число указывает на шаг разбиения, а второе - на значения, полученные на этом шаге. $R_{i,j}$ – последовательность операторов, общая для всех членов СДНФА алгоритма $A_{i,j}$.

Описанный выше метод позволяет оптимизировать любые схемы алгоритмов, в том числе и рекурсивные схемы алгоритмов. Для его применения необходимо в исходном алгоритме выделить структурированные фрагменты (имеющие один вход и один выход), описываемые регулярными выражениями, и произвести их оптимизацию приведенным методом.

ЛИТЕРАТУРА

1. *Аппаратные средства РС / К. Айден, О. Колесниченко и др. – С.Пб.: ВHV - Санкт - Петербург, 1998. – 608 с.*
2. *Жихарев В.Я., Илюшко В.М., Чумаченко И.В. Математические основы проектирования рекурсивных автоматов с программируемой логикой: Монография. – Харьков: Факт, 1999. – 144 с.*
3. *Методы проектирования символьных процессоров: Монография / В.Я. Жихарев, В.М. Илюшко, Н.В. Нечипорук, И.В. Чумаченко – Харьков: Факт, 2000. – 184 с.*
4. *Аналізатор алгоритмічних перетворювачів / Чумаченко І.В., Доценко Н.В., Бугас Д.М., Касьян О.В., Мелешенко С.Ю., Горобец А.Є. - Патент України по заявці № 2001064097 від 14.06.2001. Поз. ріш. від 04.12.01.*
5. *Кучук Г.А. Метод синтезу логічної структури мережевої бази даних // Системи обробки інформації. – Х. : НАНУ, ПАНМ, ХВУ, 2001. – Вип. 2(12). – С. 32-36.*
6. *Кучук Г.А. Формалізація предметної області багатовимірних баз даних // Системи обробки інформації. – Х. : ХФВ: «Транспорт України», 2001. – Вип. 1(11). – С. 110 - 114.*
7. *Чумаченко И.В. Расширенная алгебра регулярных схем алгоритмов с коммутативными условиями // Авіаційно - космічна техніка і технологія. – Харків: Нац. аерокосміч. ун-т «ХАІ». – 2000. – Вип. 20. – С. 154 - 158.*

Поступила 05.02.2002

ЧУМАЧЕНКО Игорь Владимирович, доктор техн. наук, доцент, профессор кафедры НАКУ «ХАИ». Область научных интересов – управление процессами в информационных системах.

КОСЕНКО Виктор Васильевич, начальник лаборатории ХВУ. В 1980 году закончил Харьковское ВВКИУ. Область научных интересов – управление процессами в информационных системах.