ИНФОРМАТИКА

2008 **июль-сентябрь** **№ 3**

ЛОГИЧЕСКОЕ ПРОЕКТИРОВАНИЕ

УДК 519.714

Н.Р. Торопов

РЕШЕНИЕ СИСТЕМЫ ДИЗЪЮНКТИВНЫХ УРАВНЕНИЙ МЕТОДОМ ПЕРЕМНОЖЕНИЯ ДНФ

Рассматривается система логических уравнений, представленных в дизъюнктивной нормальной форме (ДНФ), и предлагаются алгоритмы поиска всех ее корней, базирующиеся на операции перемножения ДНФ. Алгоритмы реализованы биполярными программами. Один полюс активизируется при большом числе неизвестных ($n > 32$), когда приходится использовать «дорогостоящие» операции над длинными векторами (превышающими длину машинного слова). Другой полюс действует при малом числе неизвестных ($n \le 32$), когда можно использовать векторное представление булевых функций и более быстрые операции над короткими векторами, что приводит к значительному ускорению процесса решения. Приводятся результаты экспериментальных испытаний алгоритмов на потоках псевдослучайных систем логических уравнений и на примерах конкретных систем из практики логического проектирования.

Введение

Логические уравнения являются удобной моделью при формализации постановок многих задач из различных областей науки и техники, таких, например, как логическое проектирование, техническая и медицинская диагностика, распознавание образов, защита информации и др. Методам решения логических уравнений посвящено множество публикаций, в том числе монографий и обзоров [1–5], однако интерес к логическим уравнениям не ослабевает, поскольку практика постоянно выдвигает все более сложные задачи, для решения которых трудно отыскать общие методы и эффективные алгоритмы.

Вряд ли можно ожидать появления общего универсального метода, эффективно решающего любую систему логических уравнений [6], но можно надеяться решить за приемлемое время конкретную систему уравнений из определенного класса, если выявить его специфику и существенно использовать ее при разработке эффективного алгоритма, ориентированного на решение систем из этого класса. Именно такой подход и применяется в данной статье. Для решения системы, уравнения которой представлены в ДНФ, предлагается алгоритм, базирующийся на логическом перемножении ДНФ в матричной форме с применением эвристик, сокращающих число пар перемножаемых конъюнкций. Для решения системы уравнений с ограниченным числом n неизвестных (не превышающим длины машинного слова) предлагается оригинальный алгоритм, базирующийся на векторном представлении булевых функций [7].

1. Постановка задачи

Логическим уравнением в математике принято называть выражение вида $A(X) = B(X)$, где $A(X)$ и $B(X)$ – некоторые формулы алгебры логики, составленные из логических операций, которые связывают (возможно, со скобками, определяющими порядок действия операций) значения логических переменных из множества $X = \{x_1, x_2, \ldots, x_n\}$, являющихся неизвестными данного уравнения. Заметим, что любое логическое уравнение $A(X) = B(X)$ можно привести к виду $E(X) = 1$, если к левой и правой частям его добавить с помощью операции $\oplus$ (дизъюнкции с исключением) выражение $B'(X)$, представляющее инверсию выражения $B(X)$: $E(X) = A(X) \oplus B'(X) = 1$.

Иногда логические уравнения называются булевыми и отождествляются с булевыми функциями, принимающими значение 1 на корнях этих уравнений, т. е. характеристические множества функций однозначно представляют множества корней соответствующих им уравне-

ний. Воспользовавшись формулами эквивалентных преобразований, всегда можно любое булево уравнение $E(X) = 1$ представить в ДНФ $D(X) = 1$, где $D(X)$ – дизъюнкция элементарных конъюнкций. В дальнейшем такое уравнение будем называть дизъюнктивным.

Решить уравнение – значит найти его корни, которые представляют собой наборы значений переменных из множества X, удовлетворяющих уравнению, т. е. таких, при подстановке которых в формулу, задающую уравнение, последнее обращается в тождество. При этом возможны следующие ситуации:

– уравнение противоречиво (не имеет корней, т. е. соответствующая ему булева функция $f(X) \equiv 0$);

– имеет один или несколько корней;

– является тождеством, когда любой из наборов значений переменных является решением уравнения (соответствующая ему булева функция $f(X) \equiv 1$).

Перечисленные свойства уравнений порождают соответствующие варианты постановок задачи, например:

– найти хотя бы один корень (любой) или убедиться в отсутствии таковых;

– найти один оптимальный (в определенном смысле) корень;

– найти достаточное (в определенном смысле) количество корней;

– найти все корни.

Далее рассматриваются системы дизъюнктивных уравнений

$$D_1(X) = 1, D_2(X) = 1, \ldots, D_m(X) = 1 \qquad (1)$$

и ставится задача – найти все корни такой системы. С теоретической точки зрения эта задача решается просто. Достаточно представить систему (1) одним *конъюнктивным* уравнением

$$D_1(X) \wedge D_2(X) \wedge \ldots \wedge D_m(X) = 1 \qquad (2)$$

и решить его, перемножив логически m выражений, представляющих уравнения системы. Очевидно, что это уравнение имеет то же множество корней, что и породившая его система.

Поиск множества корней, удовлетворяющих данной системе логических уравнений, – это лишь первый этап ее решения. На втором этапе решается задача компактного представления найденных корней. При большом числе неизвестных n число корней может быть значительным, и прямое перечисление их может оказаться неприемлемым. В этой ситуации предпочтительнее представление корней также в виде ДНФ, конъюнкции которой отображают целые интервалы таких корней в булевом пространстве.

Поскольку одно и то же множество корней может быть представлено различными по сложности ДНФ, то возникает естественное желание получить простейшую ДНФ (по числу содержащихся в ней конъюнкций или по сумме их рангов). Это не менее сложная задача, которая выходит за рамки данной статьи. Заметим лишь, что при проведении программного испытания предлагаемых методов логического перемножения булевых функций применялось лишь грубое упрощение ДНФ в процессе ее получения с использованием операций обобщенного склеивания и удаления поглощаемых конъюнкций. При необходимости полученная ДНФ может быть подвергнута дополнительному упрощению с применением специальных программ минимизации [8, 9].

2. Перемножение массива ДНФ в матричной форме

Удобной формой представления произвольной ДНФ $D = c_1 \vee c_2 \vee \ldots \vee c_p$, задающей некоторую булеву функцию $f(X)$, является троичная $(p \times n)$-матрица $\boldsymbol{D}$, столбцы которой поставлены в соответствие переменным множества $X = \{x_1, x_2, \ldots, x_n\}$, а строки представляют конъюнкции c_i из D. Элемент d_i^j принимает значение из множества $\{0, 1, -\}$: $d_i^j = 0$, если переменная $x_j \in X$ входит в конъюнкцию c_i со знаком инверсии; $d_i^j = 1$, если переменная x_j входит в конъюнкцию c_i без знака инверсии, и $d_i^j = -$, если переменная x_j не входит в конъюнкцию c_i. Таким

образом, конъюнкции $c_i \in D$ представляются троичными векторами $\boldsymbol{c}_i \in \boldsymbol{D}$, ДНФ D – троичной матрицей $\boldsymbol{D}$, а множество ДНФ $MD = \{D_1(X), D_2(X), \ldots, D_m(X)\}$ – массивом троичных матриц $\boldsymbol{MD} = \{\boldsymbol{D}_1, \boldsymbol{D}_2, \ldots, \boldsymbol{D}_m\}$.

2.1. Приемы сокращения перебора на этапе выбора ДНФ

Проблемы, возникающие в процессе получения логического произведения ДНФ, подробно изложены в [1, 2], где предлагается метод «последовательного» перемножения ДНФ, заключающийся в том, что сначала получается ДНФ $D = D_1 \wedge D_2$, затем $D = D \wedge D_3$ и т. д.:

$$D = (\ldots((D_1 \wedge D_2) \wedge D_3) \wedge \ldots) \wedge D_m.$$

При этом верхняя оценка Q числа перебираемых пар конъюнкций выражается формулой

$$Q = \sum_{i=2}^{m} p_i, \tag{3}$$

где $p_i = p_{i-1}\, e_i$, $p_1 = e_1$, e_i – число конъюнкций в i-й ДНФ.
При $e_i = g$ $(i = 1, 2, \ldots, m)$

$$Q = \sum_{i=2}^{m} g^i. \tag{4}$$

Например, при $m = 4$, $e_1 = 2$, $e_2 = 5$, $e_3 = 8$, $e_4 = 6$ получим $Q = (2\cdot5) + ((2\cdot5)\cdot8) + (((2\cdot5)\cdot8)\cdot6) = 570$. Однако если сменить порядок перебора ДНФ на обратный, получим другое значение $Q = 768$, а если перебирать ДНФ в порядке неубывания их длин, получим $Q = 550$. Все возможные порядки последовательного перебора пар ДНФ исчерпаны.

В настоящей работе предлагается метод «попарного перемножения ДНФ в каскадах», суть которого заключается в следующем. Содержащиеся в множестве $D = \{D_1, D_2, \ldots, D_m\}$ ДНФ перемножаются попарно, а полученные произведения включаются в новое множество D' (вначале пустое), представляющее следующий каскад ДНФ. Если m нечетно, то одна ДНФ из множества D пересылается в множество D' без перемножения. Затем полагается $D = D'$, $D' = \varnothing$ и описанная процедура перемножения ДНФ повторяется с новым значением множества D до тех пор, пока не будет получено множество D', содержащее единственную ДНФ. Она и будет результатом логического перемножения всех ДНФ исходного множества D.

Для предлагаемого метода «попарного перемножения ДНФ в каскадах» верхняя оценка числа Q перебираемых пар конъюнкций также зависит от порядка выбора перемножаемых пар ДНФ. Для приведенного выше примера множества D ($m = 4$, $e_1 = 2$, $e_2 = 5$, $e_3 = 8$, $e_4 = 6$) покажем значения оценки Q для всех трех возможных порядков распределения ДНФ по парам:

$$Q = (2\cdot5) + (8\cdot6) + ((2\cdot5)\cdot(8\cdot6)) = 538;$$
$$Q = (2\cdot8) + (5\cdot6) + ((2\cdot8)\cdot(5\cdot6)) = 526;$$
$$Q = (2\cdot6) + (5\cdot8) + ((2\cdot6)\cdot(5\cdot8)) = 532.$$

Наилучший эффект достигается при использовании следующей стратегии подбора пар ДНФ для перемножения в каскадах. Если m нечетно, то самая длинная ДНФ из множества $D = \{D_1, D_2, \ldots, D_m\}$ пересылается в новое множество D' без перемножения. В очередную пару включаются *самая короткая и самая длинная* из оставшихся ДНФ в множестве D. Именно такая стратегия перебора ДНФ, приводящая к наиболее равномерному распределению сумм длин ДНФ по парам, используется в предлагаемом алгоритме.

2.2. Приемы сокращения перебора на этапе выбора конъюнкций

При получении логического произведения двух ДНФ D_1 и D_2 (длиной e_1 и e_2 соответственно) необходимо перебрать $e_1 e_2$ пар конъюнкций (по одной из каждой ДНФ). Пары взаимно ортогональных конъюнкций из дальнейшего рассмотрения исключаются, а результаты логиче-

ского перемножения остальных включаются в результирующую ДНФ. При этом используются следующие дополнительные приемы сокращения числа перебираемых пар конъюнкций. Если при получении очередного произведения $c = c_1 \wedge c_2$ обнаруживается, что $c = c_1$, то все остальные конъюнкции из D_2 в паре с конъюнкцией $c_1 \in D_1$ можно не рассматривать, так как порождаемые ими логические произведения будут поглощены конъюнкцией c. Если полученное произведение $c = c_2$, то по аналогичной причине можно не рассматривать все остальные конъюнкции из D_1 в паре с конъюнкцией $c_2 \in D_2$.

Например, $(a \vee bc \vee d'e) \wedge (ab \vee c \vee df) = ab \vee ac \vee adf \vee \mathbf{abc} \vee bc \vee \mathbf{bcdf} \vee \mathbf{abd'e} \vee cd'e = ab \vee ac \vee adf \vee bc \vee cd'e$. Поскольку $a \wedge ab = ab$, то ab на остальные конъюнкции из первой ДНФ (bc и $d'e$) можно не перемножать, так как результаты перемножения будут поглощены конъюнкцией ab (поглощаемые конъюнкции отмечены жирным шрифтом). По той же причине bc можно не перемножать на конъюнкции из второй ДНФ (ab, df).

3. Перемножение множества функций в векторной форме

При ограниченном числе переменных $n \le p$ (где p – размерность машинного слова) процесс получения логического произведения множества функций $F = \{f_1(X), f_2(X), \ldots, f_m(X)\}$, задающих уравнения решаемой системы (1), можно значительно ускорить, если каждую из функций $f_i \in F$ представить в векторной форме – 2^n-мерным булевым вектором $\boldsymbol{f}_i$ [7]. Компоненты вектора $\boldsymbol{f}_i$ ставятся в однозначное соответствие элементам n-мерного булева пространства M (левая компонента соответствует элементу 00...0, правая – элементу 11…1), а значения их равны значениям функции $f_i(X)$ на соответствующих элементах $\alpha_k \in M$ ($k = 0, 1, \ldots, 2^n - 1$).

Например, при $X = \{x_1, x_2, x_3\}$ функция $f(X) = x_2 \vee x_1 x'_3$ представляется восьмикомпонентным булевым вектором 00111011.

Единичные компоненты вектора $\boldsymbol{f}_i$ представляют характеристическое множество функции $f_i(X)$ и корни задаваемого ею уравнения $f_i(X) = 1$. Если двоичные коды номеров компонент вектора $\boldsymbol{f}_i$ интерпретировать как значения n-компонентных булевых векторов, то, выписав последовательность кодов номеров всех единичных компонент вектора $\boldsymbol{f}_i$, получим в явном виде перечень наборов значений переменных множества X, представляющих корни уравнения.

Конвертирование матричного представления функции f_i в векторное производится достаточно просто. Необходимо представить каждую из конъюнкций $c_k \in D_i$ в векторной форме и выполнить покомпонентную дизъюнкцию полученных 2^n-векторов. Векторная форма конъюнкции получается следующим образом. Сначала строятся векторные представления для функций, которые соответствуют литералам, составляющим эту конъюнкцию. Затем выполняется покомпонентная конъюнкция полученных векторных представлений таких литералов.

Покажем результат выполнения этих операций на примере конвертирования функции $f = x_1 x_2' \vee x_1 x_3 \vee x_2' x_3$, представленной троичной матрицей $\boldsymbol{D}$:

$$\begin{array}{c} x_1\ x_2\ x_3 \\ \boldsymbol{D} = \begin{matrix} 1 & 0 & - \\ 1 & - & 1 \\ - & 0 & 1 \end{matrix}, \end{array}$$

$$\begin{aligned} x_1 &= 0\,0\,0\,0 \quad 1\,1\,1\,1, \\ x_2' &= 1\,1\,0\,0 \quad 1\,1\,0\,0, \\ x_3 &= 0\,1\,0\,1 \quad 0\,1\,0\,1, \\ x_1 x_2' &= 0\,0\,0\,0 \quad 1\,1\,0\,0, \\ x_1 x_3 &= 0\,0\,0\,0 \quad 0\,1\,0\,1, \\ x_2' x_3 &= 0\,1\,0\,0 \quad 0\,1\,0\,0, \\ f &= 0\,1\,0\,0 \quad 1\,1\,0\,1. \end{aligned}$$

Система логических уравнений F в векторном представлении задается ($m \times 2^n$)-булевой матрицей $\boldsymbol{F}$, столбцы которой ставятся в соответствие элементам n-мерного булева пространст-

ва M, а строки представляют значения функций на этих элементах. Логическое произведение множества булевых функций F, представленного матрицей $\boldsymbol{F}$, получается путем выполнения покомпонентной конъюнкции всех m строк матрицы $\boldsymbol{F}$. Единичные компоненты полученного 2^n-компонентного вектора $\boldsymbol{g}$ отмечают корни решаемой системы логических уравнений.

Трудоемкость алгоритма векторного перемножения массива ДНФ в числе покомпонентных операций $\wedge$ и $\vee$ над 2^n-векторами оценивается следующей формулой:

$$Q = \sum_{i=2}^{m} \left(\left(\sum_{j=1}^{e_i} (r_{ij} - 1) \right) + e_i - 1 \right) + m - 1, \tag{5}$$

где r_{ij} – ранг j-й конъюнкции в i-й ДНФ; e_i – число конъюнкций в i-й ДНФ. При $r_{ij} = r$ и $e_i = e$ $Q = mer - 1$.

Множество корней, заданных 2^n-вектором $\boldsymbol{g}$, легко представить в совершенной ДНФ, если двоичные коды номеров единичных компонент вектора $\boldsymbol{g}$ интерпретировать как значения n-компонентных строк некоторой $(h \times n)$-булевой матрицы $\boldsymbol{B}$, где h – число корней. Выполнив эквивалентные преобразования над матрицей $\boldsymbol{B}$ с помощью известных «минимизаторов», например [8, 9], можно получить более компактную форму представления множества корней решаемой системы, но эта задача выходит за рамки данной статьи и здесь не рассматривается.

Заметим лишь, что в испытуемом варианте конвертера из векторной формы представления функции f в матричную была применена программа U грубого упрощения получаемой ДНФ D с параметрическим выбором метода упрощения $\sigma \in \{0, 1\}$.

При $\sigma = 0$ никакого упрощения не производится (получается совершенная ДНФ). При $\sigma = 1$ производится упрощение получаемой ДНФ D с использованием операции обобщенного склеивания $c' = c \otimes c_i$ очередной включаемой в D конъюнкции c с конъюнкциями $c_i \in D$ и с заменой поглощаемых продуктом c' конъюнкций c_i. Две конъюнкции c_1 и c_2 обобщенно склеиваются, если их можно представить как $c_1 = x\,a$ и $c_2 = x'b$, где a и b – некоторые элементарные, не ортогональные между собой конъюнкции, логическое произведение которых и является продуктом операции обобщенного склеивания $x\,a \otimes x'b = ab$.

Учитывая ограничение на число переменных $n \le p$ (в большинстве современных ПК $p = 32$), предлагается программу U заменить ее дублем US, в котором вместо операций над логическими объектами в формате с произвольной размерностью (классы CBV, CBM, CTV, CTM) [10, 11] используются более быстрые операции над «короткоформатными» логическими объектами (классы CsBV, CsBM, CsTV, CsTM) [12]. Для этого разработаны специальные программы-конвертеры представления объектов из формата с произвольной размерностью в формат «коротких» объектов (*ToShort*) и наоборот (*FromShort*). Эти конвертеры могут оказаться полезными и в других приложениях при стыковке модулей, оперирующих с объектами в различных форматах.

4. Ускорение алгоритмов в области ограниченной размерности объектов

Предлагается следующая простая методика ускорения трудоемких алгоритмов, оперирующих с объектами произвольной размерности. Поскольку зачастую время работы таких алгоритмов имеет экспоненциальную зависимость от размерности векторов, представляющих объекты, то потенциал произвольной размерности объектов в них редко когда оказывается востребованным из-за неприемлемого времени даже при $n < 32$.

Предлагаемая методика позволяет ускорить работу разработанных ранее алгоритмов в области пространства параметров, допускающих представление объектов короткими векторами ($n < 32$). При этом интерфейс модифицируемого алгоритма и область его действия остаются прежними, изменяется только его структура. Алгоритм становится двудольным. Одну долю (базовую) составляет неизменное тело прежнего алгоритма, во вторую входит его дубль, который работает с объектами, представляемыми короткими векторами. Дубль предваряется кон-

вертером *ToShort*, а заканчивается конвертером *FromShort*. Выбор исполняемой доли производится автоматически путем проверки условия $n \le 32$ на основании анализа размерности исходных данных.

По такой методике были разработаны два двудольных метаалгоритма *M* и *V*, базовой долей которых является один и тот же алгоритм *MM* логического перемножения ДНФ в матричной форме с произвольной размерностью, работающий при $n > 32$. Основу второй доли в алгоритме *M* составляет дубль базового алгоритма *MM*, модифицированный для работы с короткими векторами и именуемый в дальнейшем алгоритмом *MS*, а основу второй доли в алгоритме *V* составляет алгоритм *VS* векторного перемножения ДНФ с подпрограммой упрощения, модифицированной для работы с короткими векторами.

5. Экспериментальные испытания предлагаемых алгоритмов

Ниже приводятся результаты экспериментальных испытаний следующих пяти алгоритмов:

– трех модификаций алгоритма решения системы дизъюнктивных уравнений в матричной форме, порождаемых различными способами сокращения перебора перемножаемых ДНФ:

MN – последовательное перемножение ДНФ в порядке их следования в множестве *D*;

MP – попарное перемножение ДНФ в порядке их следования в каскадах;

MM – попарное перемножение ДНФ в каскадах с подбором в очередную пару самой короткой и самой длинной ДНФ;

– двух модификаций алгоритма, базирующихся на векторной форме представления уравнений решаемой системы и различающихся программой получения результирующей ДНФ на втором этапе (см. разд. 3):

VL – с векторами произвольной размерности;

VS – с короткими векторами.

Все названные алгоритмы программно реализованы на C++ с использованием классов векторно-матричных логических объектов [10, 11] и подвергнуты экспериментальным испытаниям с целью получения оценок эффективности и выявления областей практической применимости. Испытания проводились на PC Intel Celeron 952 MHz, 384 MB RAM.

При сравнительных испытаниях все алгоритмы работали в одинаковых условиях: получали на входе одну и ту же систему дизъюнктивных уравнений, представленных массивом троичных матриц в формате произвольной размерности, и выдавали найденное множество корней также в виде ДНФ, представленной троичной матрицей в формате произвольной размерности. Время, затраченное конвертерами из одной формы представления объектов в другую, входит в общее время работы алгоритма, по инициативе которого производится конвертирование.

Испытания проводились на потоках псевдослучайных систем дизъюнктивных уравнений, полученных с помощью параметрически управляемых генераторов, и на конкретных примерах систем, взятых из практики логического проектирования. В приведенных ниже таблицах использованы следующие обозначения:

n – число неизвестных в системе уравнений;

m – число уравнений в системе;

r – средний ранг конъюнкций в ДНФ, представляющих уравнения;

p – число конъюнкций в уравнениях;

z – число найденных корней;

q – число конъюнкций в результирующей ДНФ, представляющей корни;

t – время решения.

Параметры n, m, r и p используются в качестве управляющих при генерировании псевдослучайных систем уравнений, а значения z, q и t являются итогом решения.

Применялись три типа генераторов: *GSA* – генератор массива из m произвольных ДНФ со случайной их длиной $0 < e < p$, *GSF* – генератор массива из m произвольных ДНФ с фиксированной длиной $e = p$ и *GSE* – генератор массива из m ДНФ определенного класса. Каждая из ДНФ этого класса представляет дизъюнкцию p двухвходовых OR- или NOR-элементов. Тип

элемента (OR или NOR) и номера из n входных полюсов, питающих его, выбираются случайно с равномерно распределенной вероятностью.

Результаты сравнения быстродействия алгоритмов *MN*, *MP* и *MM* при решении псевдослучайных систем дизъюнктивных уравнений в матричной форме, полученных генератором *GSA,* показывают, что алгоритм *MM*, не уступая *MN* и *MP* по качеству получаемых решений, превосходит их по быстродействию (табл. 1). Поэтому в приводимых далее результатах фигурирует лишь алгоритм *MM*. Время в таблицах везде указывается в секундах.

Таблица 1

Сравнение быстродействия алгоритмов *MN*, *MP* и *MM* в зависимости от числа n неизвестных и среднего числа p конъюнкций в уравнениях системы при $r = 3$, $m = 101$

n	$p = 100$			p	$n = 20$		
	MN	*MP*	*MM*		*MN*	*MP*	*MM*
17	16,46	12,98	1,45	100	574,18	47,96	4,86
18	23,31	31,29	7,10	110	10,71	11,22	2,81
19	46, 50	165,90	46, 45	120	13,46	54,51	10,47
20	567,92	47,85	2,78	140	484,28	130,64	125,80

В табл. 2 приведены результаты сравнения эффективности четырех алгоритмов *MM*, *MS*, *VL* и *VS* по быстродействию и по компактности ДНФ, представляющей множество корней, при решении псевдослучайных систем дизъюнктивных уравнений, полученных генератором *GSF*.

Таблица 2

Сравнение эффективности алгоритмов *MM*, *MS*, *VL* и *VS* в зависимости от числа m уравнений в системе и степени упрощения ДНФ σ при $n = 20$, $p = 10$, $r = 3$

σ	m	*MM*, *MS*			*VL*, *VS*		
		q	t(*MM*)	t(*MS*)	q	t(*VL*)	t(*VS*)
0	5	3405	1,48	0,43	227734	–	223,50
	10	9548	38,19	11,26	50699	238,32	7,21
	15	4208	53,30	15,14	13158	15,88	1,57
	20	1809	64,89	19,44	4201	2,39	1,26
	25	526	70,35	20,24	1080	1,60	1,16
1	5	3403	4,69	1,10	34688	–	3417,67
	10	8257	118,72	33,21	12954	373,21	150,18
	15	4006	145,87	42,90	4758	25,12	13,11
	20	1779	263,92	70,62	1774	3,64	2,26
	25	512	417,96	117,49	493	1,60	1,49

Поскольку алгоритмы *VS* и *VL* имеют одну и ту же область применения и получают одинаковые по качеству решения, а первый всегда работает быстрее, то второй исключен из дальнейшего рассмотрения как не выдержавший конкуренции. При $n \le 32$ алгоритм *MM* явно уступает по быстродействию алгоритму *MS* и тем более алгоритму *VS*, но не имеет конкурентов при $n > 32$. Алгоритмы *VS* и *MS* не поддаются однозначному ранжированию ни по быстродействию, ни по качеству решения. Каждый из них имеет свою область предпочтительного применения при $n \le 32$. Когда в решаемой системе уравнений число корней q мало (до сотни тысяч), лучше использовать алгоритм *VS*, и наоборот, когда корней много (свыше сотни тысяч) лучше выбрать алгоритм *MS*.

Заметим, что в алгоритме *VS* значительная часть от общего времени расходуется на этапе компактного представления результирующей ДНФ. Если компактность получаемой ДНФ не критична, то при $\sigma = 0$ можно достаточно быстро получить совершенную ДНФ, представляющую до сотни тысяч корней (см. табл. 3, где через t обозначено общее время работы алгоритма *VS*, а через t_2 – время, затраченное им на этапе представления системы ДНФ).

Из табл. 4 видно, что алгоритм *VS*, использующий векторное представление булевых функций, решает указанные конкретные системы уравнений гораздо быстрее (на порядок и

более), чем это делает алгоритм *MS*, базирующийся на матричном представлении. Использование параметра σ позволяет управлять поведением алгоритмов: увеличение значения σ может привести к повышению компактности получаемой ДНФ за счет увеличения времени решения.

Таблица 3

Число корней в системе из 50 уравнений и время ее решения в зависимости от числа неизвестных n при фиксированном $p = 3$ (генератор *GSE*)

n	z	t_2	t
22	7072	0,04	8,31
23	9664	0,14	20,95
24	42212	4,37	45,61
25	43464	4,49	83,26
26	75542	18,02	173,77
27	226042	217,20	528,13
28	298862	392,51	1015,82

Таблица 4

Сравнение быстродействия алгоритмов *MS* и *VS* с различной степенью упрощения σ получаемой ДНФ при решении конкретных примеров систем уравнений

Пример	m	n	Алгоритм	σ = 0		σ = 1	
				q	t	q	t
B_5	11	15	*MS*	6000	2,28	6008	4,63
			VS	6720	0,04	3363	2,61
B_6	16	18	*MS*	30240	92,91	30240	246,40
			VS	20160	0,82	13835	28,51
B_7	22	21	*MS*	60480	1128,27	60480	2545,88
			VS	40320	5,43	35280	147,26
B_8	29	24	*VS*	40320	25,84	40320	143,39

Заключение

Использование векторного представления для задания системы дизъюнктивных уравнений позволило больше чем на порядок ускорить процесс ее решения при ограниченном числе неизвестных $n < 32$, а замена операций над объектами с произвольной размерностью операциями над короткими векторами ускорила этот процесс еще в несколько раз с учетом издержек на конвертирование объектов с произвольной размерностью на короткие векторы на входе и обратное преобразование на выходе.

Предложенная простая методика модификации алгоритмов с целью их ускорения в области пространства параметров, допускающих короткоформатное представление объектов, может быть использована и в других приложениях. Важно отметить то, что модификация разработанных ранее алгоритмов производится без изменения их интерфейса и области применения. Для этого предложен полный набор конвертеров представлений логических объектов из формата с произвольной размерностью в формат коротких векторов и наоборот.

Список литературы

1. Закревский, А.Д. Логические уравнения. Изд. 2-е стереотипное / А.Д. Закревский. – М.: Едиториал УРСС, 2003. – 96 с.

2. Закревский, А.Д. Алгоритмы синтеза дискретных автоматов / А.Д. Закревский. – М.: Наука, 1971. – 512 с.

3. Закревский, А.Д. Решение логических уравнений / А.Д. Закревский // Логическое проектирование. – Минск: Ин-т техн. кибернетики НАН Беларуси, 2001. – С. 51–68.

4. Posthoff, Ch. Binare Gleichungen: Algorithmen und Programme / Ch. Posthoff, B. Steinbach. – Karl-Marx-St., 1978.

5. Baumann, M. Criptoanaly of the Hagelin M-209 Machine / M. Baumann, R. Rohde, R. Barthel // 3rd International Workshop on Boolean Problems, Sept. 17–18, 1998. – Freiberg (Sachsen). Germany, 1998. – P. 109–116.

6. Нильсон, Н. Искусственный интеллект. Методы поиска решений / Н. Нильсон. – М.: Наука, 1971. – 512 с.

7. Zakrevskij, A.D. Parallel operations over neighbors in Boolean space / A.D. Zakrevskij // Proc. of the Sixth International Conference CAD DD-07. – Minsk, 2007. – Vol. 2. – P. 6–13.

8 Торопов, Н.Р. Минимизация систем булевых функций в классе ДНФ / Н.Р. Торопов // Логическое проектирование: сб. науч. тр. – Минск: Ин-т техн. кибернетики НАН Беларуси, 1999. – Вып. 4. – С. 4–19.

9. Торопов, Н.Р. Раздельная минимизация булевых функций в системе с поляризацией их значений / Н.Р. Торопов // Методы логического проектирования. – Минск: ОИПИ НАН Беларуси, 2002. – С. 44–55.

10. Романов, В.И. Булевы векторы и матрицы в С++ / В.И. Романов, И.В. Василькова // Логическое проектирование. – Минск: Ин-т техн. кибернетики НАН Беларуси, 1997. – С. 150–158.

11. Черемисинов, Д.И. Троичные векторы и матрицы / Д.И. Черемисинов, Л.Д. Черемисинова // Логическое проектирование. – Минск: Ин-т техн. кибернетики НАН Беларуси, 1998. – С. 146–155.

12. Романов, В.И. Программные средства для решения логико-комбинаторных задач / В.И. Романов // Информатика. – 2005. – № 4. – С. 114–123.

Поступила 14.03.08

Объединенный институт проблем
информатики НАН Беларуси,
Минск, Сурганова, 6
e-mail: toropov@newman.bas-net.by

N.R. Toropov

DNF MULTIPLICATION SOLUTION OF DISJUNCTIVE EQUATIONS SYSTEM

The system of logical equations in disjunctive normal form (DNF) is considered and the roots search algorithms based on DNF multiplication are suggested. The algorithms are realized by the bipolar programs. One pole works with a large number of variables ($n > 32$), when high priced operations at long vectors (with length more than a computer word) are used. The other one works with a small number of variables ($n \le 32$), when a 2^n-vector representation of Boolean functions and faster operations at short vectors may be used, that accelerates essentially the solving process. The experimental testing results of the algorithms on pseudorandom logical equations systems flow and on examples of practical logic design systems are presented.