УДК 004.312.2, 004.021, 510.649

ИСПОЛЬЗОВАНИЕ РЕЛЯЦИОННОЙ ТЕОРИИ ПРИ ОПТИМАЛЬНОМ ПРОЕКТИРОВАНИИ ИНТЕГРАЛЬНЫХ СХЕМ

Д.В. Демидов[a, b]

[a] Университет ИТМО, 197101, Санкт-Петербург, Россия
[b] ЗАО «Интел А/О», 196247, Санкт-Петербург, Россия, Daniil.demidov@gmail.com

Аннотация. Предложен подход к адаптации реляционной теории для решения задач систем автоматизированного проектирования интегральных схем. Разработан алгоритм оптимального поиска неявных Don't Care (безразличных) значений. Алгоритм описан в терминах адаптированной теории, что позволило получить простое описание алгоритма как для неформального понимания, так и для формального анализа. Предложенный подход позволяет использовать позитивный опыт реляционных баз данных по эффективной (в том числе распределенной) реализации операций реляционной алгебры.

Проведен сравнительный анализ предложенного алгоритма и классического алгоритма оптимального поиска неявных Don't Care значений. В ходе сравнительного анализа формально доказана корректность предложенного алгоритма. Показано, что предложенный алгоритм, по сравнению с классическим, имеет значительно меньшую асимптотическую сложность в худшем случае.

Поиск неявных Don't Care значений в процессе проектирования интегральных схем позволяет повысить такие качества схем, как занимаемая площадь кристалла, энергопотребление, верифицируемость и надежность. Однако классический алгоритм оптимального поиска неявных Don't Care значений не применяется на практике ввиду его чрезмерно высокой вычислительной сложности. Применение алгоритмов субоптимального поиска не позволяет в полной мере реализовать возможности оптимизации интегральных схем. Реализация предложенного в работе алгоритма в системах автоматизированного проектирования интегральных схем целесообразна в силу значительно меньшей вычислительной сложности, потенциально позволяет улучшить соотношение скорости процесса разработки и качества получаемых интегральных схем в аспектах занимаемой площади кристалла, энергопотребления, верифицируемости и надежности. Предложенный подход делает возможным создание распределенной системы автоматизированного проектирования интегральных схем, что позволит повысить доступную системе вычислительную мощность и автоматизировать проектирование более сложных интегральных схем соответственно.

Ключевые слова: САПР, интегральные схемы, оптимизация комбинационных схем, логические сети, частично определенные булевы функции, неявные Don't Care значения, ODC, реляционная теория, реляционная алгебра, естественное соединение.

RELATIONAL THEORY APPLICATION FOR OPTIMAL DESIGN OF INTEGRATED CIRCUITS

D.V. Demidov[a, b]

[a] ITMO University, 197101, Saint Petersburg, Russia
[b] "Intel A/O", Ltd., 196247, Saint Petersburg, Russia, Daniil.demidov@gmail.com

Abstract. This paper deals with a method of relational theory adaptation for integrated circuits CAD systems. A new algorithm is worked out for optimal search of implicit Don't Care values for combinational multiple-level digital circuits. The algorithm is described in terms of the adapted relational theory that gives the possibility for a very simple algorithm description for both intuitive understanding and formal analysis. The proposed method makes it possible to apply progressive experience of relational databases in efficient implementation of relational algebra operations (including distributed ones).

Comparative analysis of the proposed algorithm and a classic one for optimal search of implicit Don't Cares is carried out. The analysis has proved formal correctness of the proposed algorithm and its considerably less worst-case complexity.

The search of implicit Don't Care values in the integrated circuits design makes it easier to optimize such characteristics of IC as chip area, power, verifiability and reliability. However, the classic algorithm for optimal search of implicit Don't Care values is not used in practice due to its very high computational complexity. Application of algorithms for sub-optimal search doesn't give the possibility to realize the potential of IC optimization to the full. Implementation of the proposed algorithm in IC CAD (a.k.a., EDA) systems is adequate due to much lower computational complexity, and potentially makes it possible to improve the quality-development time ratio of IC (chip area, power, verifiability and reliability). Developed method gives the possibility for creation of distributed EDA system with higher computational power and, consequently, for design automation of more complex IC.

Keywords: CAD, EDA, VLSI, optimization of combinational schemes, Boolean networks, partially defined Boolean functions, implicit Don't Cares, ODC, relational theory, relational algebra, natural join.

Введение

В силу возрастающей сложности и миниатюризации, а также ускоряющихся темпов производства электронных устройств возрастают требования к занимаемой площади кристалла, энергопотреблению, верифицируемости, надежности и скорости разработки интегральных схем (ИС). Неотъемлемой частью процесса проектирования ИС является решение задачи оптимизации логических схем, причем противоречивые ограничения требуют создания многоуровневых логических схем. Однако поиск оптимальной многоуровневой логической схемы неприемлемо сложен, поэтому на практике решается задача поиска субоптимальной схемы, а многие исследования направлены на разработку методов и эвристик, позволяющих снизить сложность решаемой задачи и приблизить результат к оптимальному [1–5].

Одним из ключевых моментов при оптимизации многоуровневых логических схем является поиск так называемых неявных Don't Care (безразличных или DC) значений, который, с одной стороны, позволяет существенно приблизиться к оптимальному решению, но, с другой стороны, заключает в себе значительную часть сложности исходной задачи [6].

Автором было замечено, что используемые при решении этой задачи структуры данных по своей семантике весьма близки к отношениям, используемым в теории реляционных баз данных. Алгоритмы работы с такими структурами хорошо проработаны как теоретически, так и практически, что потенциально позволяет сократить сложность поиска DC-значений, а также разработать распределенную версию алгоритма.

В работе предложен алгоритм оптимального поиска неявных DC-значений, описанный в терминах реляционной теории, адаптированной к задачам систем автоматизированного проектирования интегральных схем (САПР ИС), и доказана корректность последнего. Показано, что для предложенного алгоритма вычислительная сложность в худшем случае значительно ниже, чем для классического (экспоненциальная сложность против более чем факториальной).

Возможно внедрение результатов предлагаемой работы в существующие САПР ИС, что позволит ускорить (соответственно удешевить) и повысить качество разработки ИС (в том числе сократить занимаемую площадь кристалла и энергопотребление, повысить верифицируемость и надежность ИС [7–12]). На основе предложенного подхода к адаптации реляционной теории возможно создание распределенной САПР ИС, что позволит автоматизировать проектирование более сложных ИС, требующее высокой вычислительной мощности.

Объект исследования

В работе исследуется алгоритм оптимального поиска неявных DC-значений, применяемый на одном из этапов оптимизации многоуровневых комбинационных схем – неотъемлемой части проектирования ИС. В силу противоречивых требований к ИС используются многоуровневые комбинационные схемы, позволяющие производить минимизацию как по площади (занимаемой схемой), так и по задержке (при которой схема работает корректно). Математически многоуровневые комбинационные схемы представляют в виде логических сетей [13, 14] – ациклических ориентированных графов, которые подробно описаны в следующем разделе.

Процесс оптимизации многоуровневой комбинационной схемы является итерационным [1–5, 14, 15]. Основные этапы данного процесса показаны на рис. 1.

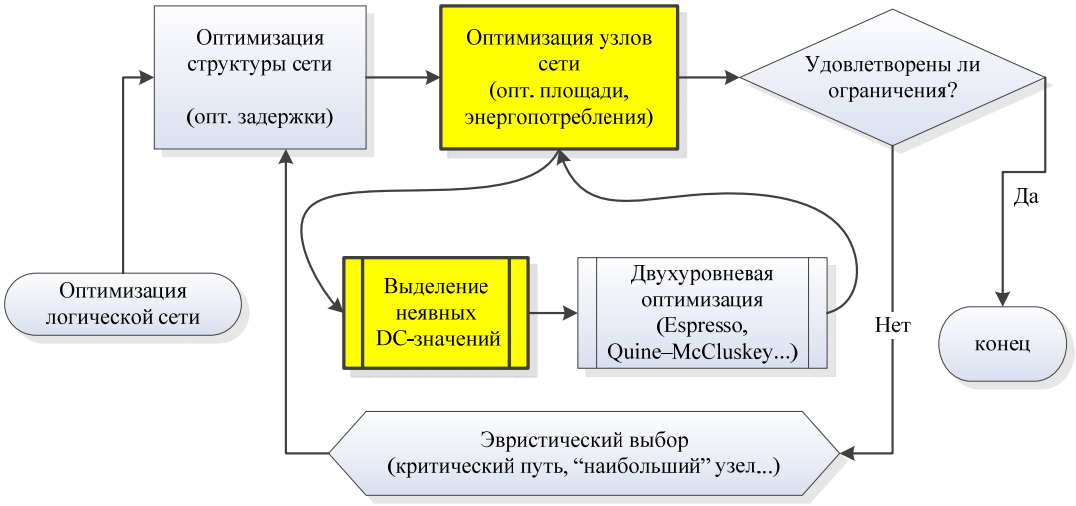

Рис. 1. Основные этапы процесса оптимизации многоуровневых комбинационных схем. Этапы, на которых используются исследуемые алгоритмы, выделены цветом

На этапе оптимизации структуры сети изменяется структура графа, представляющего логическую сеть (объединяются или разделяются узлы), и процесс, в основном, направлен на минимизацию задержки схемы.

На этапе оптимизации узлов сети структура графа остается неизменной, а изменяется внутреннее представление узлов, и процесс направлен на минимизацию площади (занимаемой схемой) и косвенно энергопотребления. Данный этап разделяется на два подэтапа – выделение неявных DC-значений и двухуровневую оптимизацию. На подэтапе двухуровневой оптимизации происходит собственно оптимизация внутреннего представления узла с применением точных (например, метод Квайна–Мак-Класки) или эвристических (например, используемых в семействе программ Espresso) алгоритмов [13–16]. Однако, если не выполнять предшествующий подэтап выделения неявных DC-значений, данный подэтап оперирует

только информацией о внутреннем представлении узла, не учитывая структуру сети, и ограничен в возможностях оптимизации. Кроме того, выполнение подэтапа выделения неявных DC-значений позволяет упростить генерацию тестов единичного отказа, а соответственно, повышает верифицируемость и надежность конечной ИС [7–12].

Представление многоуровневых комбинационных схем

Как упоминалось выше, многоуровневую комбинационную схему представляют в виде логической сети – ориентированного ациклического графа, в котором:

- выделена специальная вершина «Primary Inputs», имеющая только исходящие дуги, помеченные литералами, обозначающими входы схемы;
- выделена специальная вершина «Primary Outputs», имеющая только входящие дуги, помеченные литералами, обозначающими выходы схемы;
- прочие вершины помечены булевыми функциями общего вида, представленными в некоторой канонической форме;
- дуги между прочими вершинами помечены литералами, обозначающими внутренние переменные – выходные значения и аргументы для булевых функций, помечающих начальные вершины и конечные вершины дуги соответственно.

Способы представления булевой функции $f : B^n \to B,\ B = \{0,1\}$ в узле можно разделить на два больших класса: бинарные диаграммы решений (БДР или BDD) и структуры, задающие дизъюнктивную или конъюнктивную нормальную форму (ДНФ или КНФ) функции (например, Positional Cube Notation). В настоящей работе рассматриваются способы, задающие ДНФ функции. Семантически такие способы задают множество множеств литералов (аргументов функции), взятых в положительной или отрицательной форме:

$$S \in 2^{L\times B}, \langle S \rangle = \bigvee_{s\in S} \bigwedge_{(l,n)\in s} \langle l \rangle \oplus n ,$$

где $L \subset \{a,b,c\ldots\}$ – множество литералов; s – терм ДНФ; $\langle l \rangle$ – значение аргумента обозначаемого литералом l.

В действительности для задаваемой функций обычно существует такое множество DC_f значений аргументов, что значение выхода функции неважно. Примерами причин существования подобных входных значений являются следующие ситуации:

- некоторые сочетания входных значений могут быть запрещенными и, при корректной работе окружения схемы, никогда не будут поданы на ее входы (например, незаданные коды операций);
- в соответствии с некоторым протоколом при определенных условиях (например, схема находится в состоянии сброса или определенном режиме) некоторые выходы схемы могут быть проигнорированы окружением схемы.

Если множество DC_f не пусто, говорят, что функция f определена частично. Математически частично определенная булева функция f задается парой (f_{ON}, f_{OFF}) обычных (полностью определенных) булевых функций при выполнении условия $\forall \mathbf{x}.\overline{f_{ON}(\mathbf{x}) \wedge f_{OFF}(\mathbf{x})}$, где f_{ON} – ON-set, определяющий значения аргументов $\mathbf{x}$, при которых $f(\mathbf{x})$ должна равняться логической единице, а f_{OFF} – OFF-set, определяющий значения аргументов $\mathbf{x}$, при которых $f(\mathbf{x})$ должна равняться логическому нулю. Тогда можно определить функцию $f_{DC}(\mathbf{x}) = \overline{f_{ON}(\mathbf{x})} \wedge \overline{f_{OFF}(\mathbf{x})}$, такую, что $DC_f = \{\mathbf{x} : f_{DC}(\mathbf{x})\}$. Также можно сказать, что частично определенная функция задает множество функций $F = \left\{ f : \left(f_{ON}(\mathbf{x}) \to f(\mathbf{x})\right) \wedge \left(f_{OFF}(\mathbf{x}) \to \overline{f(\mathbf{x})}\right) \right\}$, среди которых производится выбор наилучшей альтернативы на подэтапе двухуровневой оптимизации.

Неявные Don't Care значения. Классический алгоритм поиска

Для каждого узла f сети множество DC-значений определяется как объединение пяти подмножеств: $DC_f = XCDC_f \cup CDC_f \cup SDC_f \cup ODC_f \cup XODC_f$ (рис. 2). Здесь $XCDC_f$ и $XODC_f$ – так называемые внешние (external) DC-значения, которые задаются инженером для всей схемы (в явном виде или выделяются из описания схемы на уровне регистровых передач (register transfer layer, RTL)). Примеры причин, по которым возникают $XCDC_f$ и $XODC_f$ соответственно, были приведены в предыдущем разделе. Оставшиеся три множества – SDC_f, CDC_f и ODC_f – называют внутренними или неявными

DC-значениями и вычисляют для каждого узла, исходя из структуры сети. Рассмотрим подробнее, по каким причинам возникает и как вычисляется каждое из этих множеств.

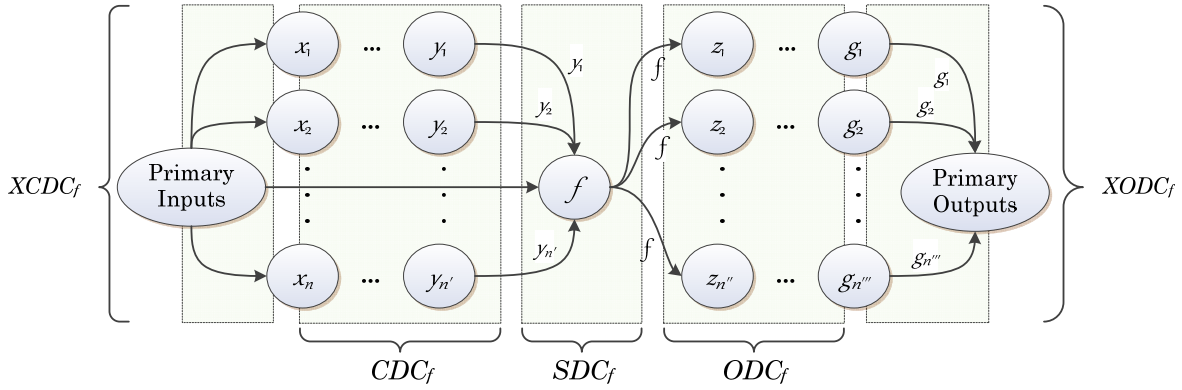

Рис. 2. Подмножества множества DC-значений для узла f логической сети

SDC_f – Satesfability Don't Cares, определяют для узла f комбинации значений его входных и выходных переменных (аргументов и собственно значений функции), которые являются запрещенными (никогда не наблюдаются при корректной работе узла) в силу зависимости выходного значения от входного. Данное подмножество DC-значений не использует информации о структуре сети, но используется для вычисления прочих DC-значений. Вычисляются $SDC_f = \{(f,\mathbf{x}) : f_{SDC}(f,\mathbf{x})\}$ следующим образом:

$$f_{SDC}(f,\mathbf{x}) = f(\mathbf{x}) \oplus f\,. \tag{1}$$

CDC_f – Controlobility Don't Cares, определяют для узла f комбинации значений его входных переменных, которые являются запрещенными (никогда не наблюдаются при корректной работе данного узла, а также узлов, от которых он зависит) в силу имеющихся зависимостей между ними. Вычисляются $CDC_f = \{\mathbf{x} : f_{CDC}(\mathbf{x})\}$ на основе определенных выше SDC_g:

$$f_{CDC}(\mathbf{x}) = \forall z \notin \text{inputs}(f).\bigvee_{g \in \text{inputs}^*(f)} g_{SDC}\,, \tag{2}$$

где $\text{inputs}(f)$ – множество входных переменных узла (аргументов функции) f, $\text{inputs}^*(f) = \text{inputs}(f) \cup \bigcup_{g \in \text{inputs}(f)} \text{inputs}^*(g)$ – транзитивное замыкание последних, а выражение вида $\forall x.y(x,\mathbf{z})$ вычисляется как $y|_x \wedge y|_{\bar{x}} = y(1,\mathbf{z}) \wedge y(0,\mathbf{z})$.

ODC_f – Observability Don't Cares, определяют для узла f комбинации значений его входных переменных (аргументов функции), которые делают выходное значение данного узла (функции) ненаблюдаемым всеми узлами, зависящими от него, т.е. при которых изменение выходного значения узла не повлечет изменения выходного значения какого-либо из узлов, зависящих от данного. Вычисляются $ODC_f = \{\mathbf{x} : f_{ODC}(\mathbf{x})\}$ следующим образом:

$$f_{ODC}(\mathbf{x}) = \forall z \notin \text{inputs}(f).\bigwedge_{h \in \text{outputs}^*(f)} \overline{\left(\frac{\partial h}{\partial f}\right)}\,, \tag{3}$$

где $\text{outputs}(f)$ — множество конечных узлов, выходных дуг узла f, $\text{outputs}^*(f) = \text{outputs}(f) \cup \bigcup_{g \in \text{outputs}(f)} \text{outputs}^*(g)$ – транзитивное замыкание последних, а $\partial h/\partial f$ обозначает так называемую булеву производную функции h по переменной f.

Булева производная $\partial h/\partial f$ является функцией от $\text{inputs}^*(h) \setminus f$ и возвращает единицу, если изменение значения f приводит к изменению h, и ноль – в противном случае. Если f входит в h в явном виде ($f \in \text{inputs}(h)$), то $\partial h/\partial f$ вычисляется как булева разность: $h|_f \oplus h|_{\bar{f}}$. Если же h зависит от f косвенно (через другие функции), то необходимо применять цепное правило (chaining rule):

$$\frac{\partial h}{\partial f}=\left(\frac{\partial h}{\partial g_1}\cdot\frac{\partial g_1}{\partial f}\oplus\cdots\oplus\frac{\partial h}{\partial g_n}\cdot\frac{\partial g_n}{\partial f}\right)\oplus\left(\frac{\partial^2 h}{\partial g_1\partial g_2}\cdot\frac{\partial g_1}{\partial f}\cdot\frac{\partial g_2}{\partial f}\oplus\frac{\partial^2 h}{\partial g_1\partial g_3}\cdot\frac{\partial g_1}{\partial f}\cdot\frac{\partial g_3}{\partial f}\oplus\cdots\right)\oplus\cdots=$$

$$=\bigoplus_{G\subset\text{inputs}(h)}\left(\frac{\partial^{|G|}h}{\prod\limits_{g\in G}\partial g}\cdot\bigwedge_{g\in G}\frac{\partial g}{\partial f}\right).$$

Адаптированная реляционная теория

В данном разделе описан используемый в работе частный случай реляционной теории, адаптированной к задачам САПР ИС, и семантика отношений и некоторых операторов реляционной алгебры в контексте решаемой задачи. Будем называть отношением R пару (h,T), где $h\in 2^L$, $L=\{a,b,c\ldots\}$ – подмножество литералов, называемое заголовком, и $T=\left\{t\in B'^{|h|}\right\}$ – множество кортежей размера $|h|$, заданных на расширенном булевом множестве $B'=\{0,1,*\}$, в которое включено специальное значение $*$, называемое телом. Отношение R можно интерпретировать как некоторую булеву функцию $\langle R\rangle=f:B^{|h|}\to B$, заданную следующей ДНФ:

$$f(\mathbf{x})=\bigvee_{t\in R}\bigwedge_{x\in h}\begin{Bmatrix}x, & \text{если } t_x=1\\ \bar{x}, & \text{если } t_x=0\\ 1, & \text{если } t_x=*\end{Bmatrix}.$$

Над отношениями можно производить ряд операций; в данной работе используются проекция $\underset{h'}{\pi}R$, селекция $\underset{p(h)}{\sigma}R$, переименование $\underset{x:=f(h)}{\rho}R$, объединение $A\cup B$ и две модификации естественного соединения – $A\bowtie B$ и $A\bowtie^{h'}B$. Первые три операции взяты из реляционной алгебры без каких-либо изменений; остальные операции имеют небольшие изменения, позволяющие естественным образом отобразить их на операции булевой алгебры над соответствующими функциями и, в то же время, сохранить все основные свойства операций.

В классической реляционной алгебре операция объединения разрешена только для отношений с одинаковыми заголовками и возвращает отношение с тем же заголовком и объединением тел, а операция естественного соединения объединяет кортежи двух отношений, если они строго равны по всем общим атрибутам [17–19]. В адаптированной реляционной алгебре разрешается объединять отношения с разными заголовками, при этом значения «недостающих» атрибутов картежей принимаются равными $*$, а операция естественного соединения игнорирует значения атрибутов, равные $*$:

$$A\bowtie B=C\Rightarrow\forall t\in T_A\,\forall t'\in T_B\left(\forall x\in h_C\left(t_x\sim t'_x\right)\Rightarrow\exists s\in T_C\left(s_x=\text{select}(t_x,t'_x)\right)\right),$$

где $x\sim y\Leftrightarrow(x=y)\vee(x=*)\vee(y=*)$, $\text{select}(x,*)=x$ и $\text{select}(*,y)=y$.

Вторая модификация естественного соединения $A\bowtie^{h'}B$, названная ограниченным естественным соединением, вводится аналогичным способом с заменой $x\sim y$ на $x\sim^{h'}y=\begin{cases}x\sim y, & \text{если}(x\notin h')\wedge(y\notin h')\\ x=y, & \text{иначе}\end{cases}$.

В таком случае перечисленные операции можно интерпретировать следующим образом (таблица).

Реляционная алгебра		Булева алгебра	
Объединение	$A\cup B$	Дизъюнкция	$\langle A\rangle\vee\langle B\rangle$
Естественное соединение	$A\bowtie B$	Конъюнкция	$\langle A\rangle\wedge\langle B\rangle$
Проекция	$\underset{h\setminus x}{\pi}R$	Квантор существования	$\exists x.\langle R\rangle=\langle R\rangle\big\vert_x\vee\langle R\rangle\big\vert_{\bar{x}}$
Селекция	$R_x=\underset{h\setminus x}{\pi}\left(\underset{x\sim 1}{\sigma}R\right);$ $R_{\bar{x}}=\underset{h\setminus x}{\pi}\left(\underset{x\sim 0}{\sigma}R\right)$	Кофактор Шеннона	$\langle R\rangle\big\vert_x$; $\langle R\rangle\big\vert_{\bar{x}}$

Таблица. Интерпретация основных операций адаптированной реляционной алгебры

Определим понятие «функционального» заголовка отношения R как $fh_R = \left\{ y \in h_R : R_y \bowtie R_{\overline{y}} = [\,] \right\}$, где $[\,]$ – пустое отношение (с пустым телом). Тогда можно также интерпретировать отношение R относительно атрибута $y \in fh_R$ как частично определенную функцию $\langle R \rangle_y = \left(\langle R_y \rangle, \left\langle R_{\overline{y}} \right\rangle \right)$. Также легко показать, что будут верны следующие тождества:

$$\langle R \rangle = \langle R \rangle_y \overline{\oplus} y; \quad \overline{\langle R \rangle_y} = \left\langle \underset{y \coloneqq \overline{y}}{\rho} R \right\rangle_y ; \tag{4}$$

$$\langle F \bowtie G \rangle_g = \langle G \rangle_g \circ \langle F \rangle_f \text{, если } f \in fh_F, g \in fh_G, f \in h_G, g \notin h_F . \tag{5}$$

Описание предлагаемого алгоритма поиска неявных DC-значений

Особенность предлагаемого алгоритма по сравнению с классическим заключается в том, что множество DC_f вычисляется неявно. При классическом подходе для каждого узла f явным образом конструируется пара $\left(f_{ON} = f \wedge \overline{f_{DC}}, f_{DC} \right)$, являющаяся входом для следующего подэтапа в процессе оптимизации сети. В предлагаемом алгоритме для каждого узла f вначале предполагается пара $\left(f_{ON} = f, f_{OFF} = \overline{f} \right)$ (т.е. предполагается, что DC_f пусто), а затем f_{ON} и f_{OFF} уточняются, и на вход следующего подэтапа передается пара $\left(f'_{ON} = f \wedge \overline{f_{DC}}, f'_{OFF} = \overline{f} \wedge \overline{f_{DC}} \right)$ (для алгоритмов из семейства Espresso в действительности необходимо передать любые две функции из тройки $\left(f_{ON}, f_{OFF}, f_{DC} \right)$ [13–15]). Таким образом, DC_f не конструируется в явном виде, что во многом определяет более низкую вычислительную сложность.

Описание предлагаемого алгоритма в терминах адаптированной реляционной теории состоит из трех шагов: первый шаг является подготовительным, второй неявно рассчитывает CDC_f для каждого узла f, а третий – ODC_f:

1. для каждого узла $f = \langle R \rangle$ составить отношение $R' = \underset{f \coloneqq 1}{\rho} R \cup \underset{f \coloneqq 0}{\rho} \overline{R}$,

где $\overline{R} = \underset{t \in T_R}{\rhd\!\lhd} \bigcup_{x \in t} \underset{x \coloneqq x}{\rho} [\,]$, $\left\langle \overline{R} \right\rangle = \overline{\langle R \rangle}$;

2. рекурсивно, начиная с вершины «Primary Inputs» (от входов схемы к выходам) для всех смежных вершин произвести естественное соединение соответствующих отношений и спроецировать результат на конечную вершину;

3. рекурсивно, начиная с вершины «Primary Outputs» (от выходов схемы к входам) для всех смежных вершин произвести естественное соединение соответствующих отношений, ограниченное по помечающему дугу литералу, и спроецировать результат на начальную вершину.

Первый шаг в общем случае обладает высокой вычислительной сложностью, но он выполняется только на первой итерации процесса оптимизации логической сети. Кроме того, так как логическая сеть на первой итерации получена из RTL-описания схемы и обычно ее узлы помечены булевыми функциями из ограниченного набора, вычисления первого шага можно табулировать. Таким образом, первый этап не влияет на асимптотическую сложность всего алгоритма в целом, что является важным преимуществом предлагаемого алгоритма.

Доказательство корректности предлагаемого алгоритма

В силу ограничений на объем настоящей работы доказательства приведены в сжатом виде.

Под корректностью предлагаемого алгоритма подразумевается то, что для заданной логической сети он рассчитывает в точности то же множество DC_f, что и классический алгоритм (с той лишь разницей, что предлагаемый алгоритм рассчитывает DC_f неявно, см. предыдущий раздел).

Докажем корректность расчета CDC_f, т.е. докажем, что, согласно (2), для

$$f' = \left\langle \underset{h_F}{\pi} \left(F \bowtie \left(\underset{g \in \text{inputs}^*(f)}{\rhd\!\lhd} G \right) \right) \right\rangle_f \text{ верно } f'_{DC} = \forall z \notin \text{inputs}(f). \bigvee_{g \in \text{inputs}^*(f)} g_{SDC} . \tag{6}$$

Рассмотрим отрицание правой части тождества (6) и перепишем его в терминах реляционной алгебры (см. (1), (4) и таблицу):

$$\overline{\forall z \notin \text{inputs}(f).\bigvee_{g\in\text{inputs}^*(f)} g_{SDC}} = \forall z \notin \text{inputs}(f).\left\langle \bigcup_{g\in\text{inputs}^*(f)} \underset{g\coloneqq g}{\rho} G \right\rangle = \exists z : z \notin \text{inputs}(f).\overline{\left\langle \bigcup_{g\in\text{inputs}^*(f)} \underset{g\coloneqq g}{\rho} G \right\rangle} =$$

$$= \left\langle \underset{h_F}{\pi}\left(\overline{\bigcup_{g\in\text{inputs}^*(f)} \underset{g\coloneqq g}{\rho} G} \right) \right\rangle = \left\langle \underset{h_F}{\pi}\left(\bowtie_{g\in\text{inputs}^*(f)} \overline{\underset{g\coloneqq g}{\rho} G} \right) \right\rangle = \left\langle \underset{h_F}{\pi}\left(\bowtie_{g\in\text{inputs}^*(f)} G \right) \right\rangle.$$

Теперь рассмотрим

$$\left(f'_{ON}, f'_{OFF}\right) = \left(\langle F\rangle \wedge \overline{f'_{DC}}, \langle F\rangle \wedge \overline{f'_{DC}}\right) = \left\langle \underset{f\coloneqq 1}{\rho}\left(F_f \rhd\lhd \overline{DC}\right) \cup \underset{f\coloneqq 0}{\rho}\left(F_{\overline{f}} \rhd\lhd \overline{DC}\right) \right\rangle = \left\langle F \rhd\lhd \overline{DC} \right\rangle, \text{ где } \langle DC\rangle = f'_{DC},$$

т.е. выполняется тождество $f' = \left\langle F \rhd\lhd \underset{h_F}{\pi}\left(\bowtie_{g\in\text{inputs}^*(f)} G \right) \right\rangle$, что и требовалось доказать.

Докажем корректность расчета ODC_f. Заметим, что если $f \in fh_F$, то в результирующее отношение входят только те картежи из отношения F, для которых атрибут f в отношении G имеет значение, отличное от $*$, т.е. $\partial g/\partial f = 1$.

Аналогично, если $f \in fh_F$ и $g \in fh_G$, то

$$\left\langle \underset{h_F}{\pi}\left(F \bowtie^{\{f\}} G \bowtie^{\{g\}} H \right) \right\rangle_f = \left(\exists z : z \notin \text{inputs}(f).f_{ON} \cdot \frac{\partial g}{\partial f}\cdot\frac{\partial h}{\partial g}, \exists z : z \notin \text{inputs}(f).f_{OFF} \cdot \frac{\partial g}{\partial f}\cdot\frac{\partial h}{\partial g} \right).$$

Учитывая (5), получим, что $F \bowtie^{\{f\}} G_1 \bowtie^{\{f\}} G_2 = F \bowtie^{\{f\}} G$, где

$$\langle G\rangle_g = \langle G_1 \rhd\lhd G_2\rangle_g = \begin{cases} g_1 \vee g_2, & \text{если } g = \{\} \\ g_1 \circ g_2, & \text{если } g \subset fh_{G_2},\ g = \left(h_{G_1} \cap h_{G_2}\right); \\ g_2 \circ g_1, & \text{если } g \subset fh_{G_1} \end{cases}$$

тогда

$$\bigvee_{g\in\text{outputs}^*(f)} \frac{\partial g}{\partial f} = F \bowtie^{\{f\}} \left(G_1 \bowtie^{\{g_1\}} \left(H_{1,1} \bowtie^{\{h_{1,1}\}} \ldots \right) \bowtie^{\{g_1\}} \ldots \right) \bowtie^{\{f\}} \left(G_2 \bowtie^{\{g_2\}} \left(H_{2,1} \bowtie^{\{h_{2,1}\}} \ldots \right) \bowtie^{\{g_2\}} \ldots \right) \ldots.$$

Преобразуем (3) следующим образом:

$$\overline{f_{ODC}} = \forall z \notin \text{inputs}(f).\bigwedge_{g\in\text{outputs}^*(f)} \overline{\left(\frac{\partial g}{\partial f}\right)} =$$

$$= \exists z : z \notin \text{inputs}(f).\overline{\bigwedge_{g\in\text{outputs}^*(f)} \left(\overline{\frac{\partial g}{\partial f}}\right)} = \exists z : z \notin \text{inputs}(f).\bigvee_{g\in\text{outputs}^*(f)} \frac{\partial g}{\partial f},$$

тогда для f'', вычисляемого согласно третьему шагу предлагаемого алгоритма, $\left(f''_{ON}, f''_{OFF}\right) = \left(f'_{ON}\cdot\overline{f'_{ODC}}, f'_{OFF}\cdot\overline{f'_{ODC}}\right)$, что доказывает корректность расчета ODC_f.

Сравнение вычислительной сложности предложенного и классического алгоритмов

Произведем оценку асимптотической вычислительной сложности в худшем случае для классического и предложенного алгоритмов. Пусть логическая сеть – граф $G = (V, E)$ – имеет $|V| = m$ вершин, а каждая вершина $v_i, i \in \overline{0, m-1}$ имеет $|\text{inputs}(v_i)| = n_i$ входных дуг (рáвно аргументов соответствующей булевой функции, рáвно мощность заголовка соответствующего отношения) и t_i, $t_i \le 2^{n_i-1}$ термов ДНФ, задающей булеву функцию (рáвно количество кортежей в соответствующем отношении). Будем обозначать $n = \max\limits_i n_i$.

Рассмотрим классический алгоритм. В сложности $f_{SDC_{v_i}}(n_i)$ расчета SDC_{v_i} (см. (1)) доминирует сложность расчета отрицания v_i: $f_{SDC_{v_i}} \in O\left(n_i^{t_i}\right) \subseteq O\left(n_i^{2^{n_i-1}}\right)$, причем результирующая ДНФ будет иметь

n_i+1 аргументов и до $2\cdot t_i$ термов. Сложность расчета CDC_{v_i} (см. (2)) в худшем случае складывается из сложности расчета SDC_{v_i} для m вершин и сложности квантора всеобщности по $(m-1)\cdot n-1\in O(m\cdot n)$ переменным (что в худшем случае сводится к конъюнкции $2^{m\cdot n-2}$ ДНФ по два терма): $f_{CDC_{v_i}}\in O\left(m\cdot\left(n^{2^n}\right)\right)+O\left(2^{2^{m-n}}\right)$. Наихудший случай для расчета ODC_{v_i} достигается при такой конфигурации графа, что $n_i=i$ и максимальная длина пути в графе – m. Тогда сложность задается рекуррентным соотношением $f_{ODC}(i)=g(i)+\sum_{k=1}^{i}f_{ODC}(i-k)\cdot C_i^k$, $g\in O\left(i^{2^{i^2+1}+1}\right)$. Точно решить данное соотношение не представляется возможным, однако с уверенностью можно сказать, что сложность расчета ODC_{v_i} и всего классического алгоритма не лучше чем $O\left(m!\cdot m^{2^{m^2+1}+1}\right)$.

Анализ сложности предложенного алгоритма значительно проще, причем асимптотическая сложность расчета CDC_{v_i} и ODC_{v_i} совпадают. Достаточно заметить, что в худшем случае необходимо произвести естественное соединение отношений для всех пар (v_i,v_j) вершин графа, причем сложность естественного соединения в худшем случае будет составлять $O(t_i\cdot t_j)$. Таким образом, асимптотическая сложность предложенного алгоритма в худшем случае составляет $O\left(\sum_{i,j}t_i\cdot t_j\right)\subseteq O\left(|E|\cdot 2^n\cdot 2^n\right)=O\left(m^2\cdot 2^{2\cdot n}\right)\subseteq O\left(m^2\cdot 2^m\right)$.

Заключение

Описан подход к адаптации реляционной теории к решению задач систем автоматизированного проектирования интегральных схем. На основе предложенного подхода разработан алгоритм оптимального поиска неявных Don't Care значений. Доказана корректность предложенного алгоритма.

Применение предложенного алгоритма в системах автоматизированного проектирования потенциально позволит ускорить процесс разработки интегральных схем, а также сократить занимаемую площадь кристалла, энергопотребление, повысить верифицируемость и надежность получаемых схем.

В перспективе создание распределенной версии исследуемых алгоритмов позволит решать задачи систем автоматизированного проектирования для более сложных схем, требующих высокой вычислительной мощности.

Литература

1. De Micheli G. Synchronous logic synthesis: algorithms for cycle-time minimization // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 1991. V. 10. N 1. P. 63–73.
2. Hachtel G.D., Rho J.K., Somenzi F., Jacoby R. Exact and heuristic algorithms for the minimization of incompletely specified state machines // Proc. of European Conference on Design Automation. Amsterdam, Netherlands, 1992. P. 184–191.
3. Bogatyrev V.A., Bogatyrev S.V., Golubev I.Yu. Optimization and the process of task distribution between computer system clusters // Automatic Control and Computer Sciences. 2012. V. 46. N 3. P. 103–111.
4. Sentovich E.M., Singh K.J., Lavagno L., Moon C., Murgai R., Saldanha A., Savoj H., Stephan P.R., Brayton R.K., Sangiovanni-Vincentelli A.L. SIS: A System for Sequential Circuit Synthesis. Technical Report UCB/ERL M92/41, Electronics Research Lab, Univ. of California, Berkeley, 1992.
5. Pederson D.O. The VIS Group. VIS: Verification Interacting with Synthesis, 1995 [Электронный ресурс]. Режим доступа: http://embedded.eecs.berkeley.edu/Respep/Research/vis, свободный. Яз. англ. (дата обращения 04.12.2013).
6. Bartlett K.A., Brayton R.K., Hachtel G.D., Jacoby R.M., Morrison C.R., Rudell R.L., Sangiovanni-Vincentelli A., Wang A. Multi-level logic minimization using implicit don't cares // IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems. 1988. V. 7. N 6. P. 723–740.
7. Fujiwara H. Logic Testing and Design for Testability. Cambridge: MIT Press, 1985. 304 p.
8. Chang A.C.L., Reed I.S., Beans A.V. Path sensitization, partial boolean difference and automated fault diagnosis // IEEE Transactions on Computers. 1972. V. C-21. N 2. P. 189–195.
9. Bogatyrev V.A. Exchange of duplicated computing complexes in fault-tolerant systems // Automatic Control and Computer Sciences. 2011. V. 45. N 5. P. 268–276.
10. Bogatyrev V.A. Fault tolerance of clusters configurations with direct connection of storage devices // Automatic Control and Computer Sciences. 2011. V. 45. N 6. P. 330–337.

11. Богатырев В.А., Богатырев С.В. Критерии оптимальности многоуровневых отказоустойчивых компьютерных систем // Научно-технический вестник СПбГУ ИТМО. 2009. № 5 (63). С. 92–97.
12. Богатырев В.А., Богатырев А.В. Функциональная надежность систем реального времени // Научно-технический вестник информационных технологий, механики и оптики. 2013. № 4 (86). С. 150–151.
13. Muroga S., Kambayashi Y., Lai H.C., Culliney J.N. Transduction method – design of logic networks based on permissible functions // IEEE Transactions of Computers. 1989. V. 38. N 10. P. 1404–1424.
14. Lin B., Touati H.J., Newton A.R. Don't care minimization of multi-level sequential logic networks // Proc. IEEE International Conference on Computer-Aided Design. 1990. P. 414–417.
15. Brayton R.K., Hachtel G., McMullen C., Sangiovanni-Vincentelli A. Logic Minimization Algorithms for VLSI Synthesis. Kluwer Academic Publishers, 1985. 206 p.
16. Theobald M., Nowick S.M., Wu T. Espresso-HF: a heuristic hazard-free minimizer for two-level logic // Proceedings - Design Automation Conference. Las Vegas, USA, 1996. P. 71–76.
17. Codd E.F. A relational model of data for large shared data banks // M.D. Computing. 1998. V. 15. N 3. P. 162–166.
18. Abiteboul S., Hull R., Vianu V. Foundations of Databases. Addison-Wesley, 1995. 685 p.
19. Date C.J., Darwen H. Foundation for Future Database Systems: The Third Manifesto. 2nd ed. Addison-Wesley, 2000. 608 p.

Демидов Даниил Валентинович – аспирант, Университет ИТМО, 197101, Санкт-Петербург, Россия; инженер, ЗАО «Интел А/О», 196247, Санкт-Петербург, Россия, Daniil.demidov@gmail.com

Daniil V. Demidov – postgraduate, ITMO University, 197101, Saint Petersburg, Russia; software engineer, "Intel A/O", Ltd., 196247, Saint Petersburg, Russia, Daniil.demidov@gmail.com

Принято к печати 02.07.14
Accepted 02.07.14