УДК 004.02
ГРНТИ 20.01
DOI: 10.23947/interagro.2020.1.472-475

МЕТОДЫ СОКРАЩЕНИЯ БОЛЬШИХ ЛОГИЧЕСКИХ ФОРМУЛ, ПРЕДСТАВЛЕННЫХ В КОНЪЮНКТИВНОЙ НОРМАЛЬНОЙ ФОРМЕ ПРИ ВЫЯСНЕНИИ ИХ ВЫПОЛНИМОСТИ

Гданский Н.И., Денисов А.А.

Московский государственный университет технологий и управления им. К.Г. Разумовского, г. Москва, Российская Федерация

Аннотация Статья посвящена задаче выполнимости конъюнктивных нормальных форм, применяемых при моделировании систем. Рассмотрены проблемы их препроцессорной обработки. Дан анализ частных методов сокращения формул данного вида, которые имеют полиномиальную сложность по длине входа.

Ключевые слова: конъюнктивные нормальные формы, задача выполнимости, препроцессорная обработка, методы сокращения избыточности, сложность алгоритмов.

METHODS OF REDUCING OF LARGE BOOLEAN FORMULAS REPRESENTED IN A CONJUNCTIVE NORMAL FORM FOR DETERMINING THEIR SATISFIABILITY

Gdansky N.I., Denisov A.A.

Moscow State University of Technology and Management K.G. Razumovsky, Moscow, Russian Federation

Abstract The article explores the satisfiability of conjunctive normal forms used in modeling systems. The problems of CNF preprocessing are considered. The analysis of particular methods for reducing this formulas, which have polynomial input complexity is given.

Keywords: conjunctive normal forms, CNF, satisfiability, SAT-problem, preprocessing, methods for reducing redundancy, complexity of algorithms.

1. Постановка задачи. Методология алгебры логики широко используется при моделировании многих информационных приложений – больших массивов информации, анализе схем, криптографии, и многих других.

Как правило, для решения этой задачи соответствующие булевы функции эквивалентно представляют в виде конъюнктивных нормальных форм (КНФ), а решение задачи сводится к проверке выполнимости КНФ (SAT CNF), т.е. выяснению, существует ли такой набор значений переменных КНФ, которое является решением уравнения следующего вида:

$$F(x_1,x_2,\ldots,x_n) = C_1 \& C_2 \& \ldots \& C_k = true, \quad (1)$$

дизъюнкты $\forall s\ (1 \le s \le k)\ C_s = x_{i1}^{\alpha i1} \vee x_{i2}^{\alpha i2} \vee \ldots \vee x_{ils}^{\alpha ils}$, $\alpha_{ilp} = 0$ или 1, $x_{ip}^0 = \neg x_{ip}$, $x_{ip}^1 = x_{ip}$.

После обозначения длин дизъюнктов через l_s общую длину формулы F можно представить в виде: $L = l_1 + \ldots + l_k$.

Решение данной задачи имеет существенные особенности. В частности, применение некоторых методов сокращения избыточности формул КНФ приводит к тому, что сокращенная формула F_{red} не эквивалентна исходной. Несмотря на то, что в ней утрачиваются некоторые выполняющие наборы переменных, при этом сохраняется общее свойство выполнимости КНФ. Возможны и такие ситуации, в которых решение основной задачи SAT CNF получают уже при препроцессорной обработке КНФ, т.е. при сокращении F будет определено, что она является тождественно истинной, невыполнимой (тождественно ложной) или же будет определен выполняющий набор переменных, доказывающий выполнимость F.

Отдельную проблему представляет выбор структур данных для внутреннего представления F, а также вспомогательных данных при решении задач препроцессинга. При выборе таких структур необходимо принимать во внимание не только структуру хранимых данных, но и особенности выполняемых операции, производимых с ними.

2. Анализ существующих методов решения задачи выполнимости и препроцессинга КНФ. В настоящее время при решении задачи выполнимости (1) для больших формул, имеющих десятки и сотни переменных, обычно используются методы, базирующиеся на алгоритмах DPLL [1,2] и CDCL [3]. Поскольку данные алгоритмы имеют экспоненциальную сложность по длине входа, то актуальной является вспомогательная задача максимального сокращения формул КНФ при помощи

алгоритмов, имеющих полиномиальную сложность. В результате такого сокращения получают сокращенную формулу F_{red}, в которой сокращено общее число дизъюнктов, суммарное число литер в них и в некоторых случаях сокращается число переменных.

Сокращение формулы КНФ является одним из главных этапов общего конвейера по решению практических задач с использованием КНФ, который содержит: исходное построение формулы для исследуемой модели, ее сокращение (препроцессинг), а также решение самой задачи выполнимости КНФ [4,5].

Препроцессинг КНФ имеет существенные особенности. В частности, применение некоторых методов сокращения избыточности формул КНФ приводит к тому, что сокращенная формула F_{red} не эквивалентна исходной. Несмотря на то, что в ней утрачиваются некоторые выполняющие наборы переменных, при этом сохраняется общее свойство выполнимости КНФ. Возможны и такие ситуации, в которых решение основной задачи SAT CNF получают уже при препроцессорной обработке КНФ, т.е. уже при сокращении F будет определено, что она является тождественно истинной, невыполнимой (тождественно ложной) или же будет определен выполняющий набор переменных, т.е. доказана выполнимость F.

Для выполнения препроцессинга зачастую применяют методы, имеющие экспоненциальную сложность. Это существенно снижает эффективность данной операции.

Отдельную проблему представляет выбор структур данных для внутреннего представления F, а также вспомогательных данных при решении задач препроцессинга. При выборе таких структур необходимо принимать во внимание не только структуру хранимых данных, но и особенности выполняемых операции, производимых с ними.

3. Постановка задач. Статья посвящена классификации и анализу основных типов избыточности в формулах КНФ. В ней рассматриваются следующие вопросы:

- анализируются общие свойства и их положения в общей формуле КНФ,
- рассматриваются особенности их применения,
- оценивается сложность алгоритмической реализации.

Признаки для классификации видов избыточности в формулах в задаче выполнимости.

Обнаружение и способы устранения разных типов избыточности в первую очередь определяются тем, каким образом они входят (локализованы) в общей формуле КНФ.

По данному признаку среди всех видов избыточности выделен следующий набор: -1) внутридизъюнктная, 2) междизъюнктная и 3) смешанная.

В зависимости от охвата множества дизъюнктов в формуле КНФ F среди видов избыточности предлагается выделить: 1) однодизъюнктные, 2) двухдизъюнктные и 3) общеформульные.

Поскольку при устранении ряда избыточностей возможно устранение из них переменных, то с этой точки зрения удаляемые переменные предложено подразделить на 2 типа: 1) определенные и 2) свободные.

В случае 1) удаляемой переменной, исходя из условия истинности всей формулы, присваивается определенное истинностное значение. Его необходимо внести в некоторый массив part_assign, задающий частичное определение возможного выполняющего набора переменных. Таким образом, в случае выполнимости КНФ определенные переменные должны принимать только то значение, которое задается в массиве part_assign.

В случае 2) при удалении переменной ей не задается никакого логического значения, исходя из условия выполнимости КНФ. При этом ее значение не оказывает влияния на решение задачи (1). Она может принимать любое решение.

Отсюда следует, что при определении полного решения задачи (1) по решению для сокращенного варианта функции F_{red}, значения для определенной части удаленных переменных необходимо брать из part_assign, в то время, как свободным переменным можно задавать произвольные значения.

По свойству сохранения набора выполняющих наборов в сокращенной КНФ, сами методы сокращения делим на 1) эквивалентные и 2) неэквивалентные.

Эквивалентность в полном варианте подразумевает, что у сокращенной F_{red} и исходной F формул числа переменных совпадают, а также совпадают наборы выполняющих переменных.

В расширенном варианте при сокращении числа переменных эквивалентностью формул F_{red} и F будем называть такой случай, когда при восстановлении решения полной задачи (1) из решения, полученного для сокращенной формулы F_{red} с добавлением значений удаленных переменных, взятых из part_assign и произвольных значений свободных переменных в результате получается такой же набор выполняющих наборов, что и для задачи (1).

4. Анализ видов избыточности. В качестве основного признака классификации примем положение рассматриваемых видов избыточности в исходной анализируемой формуле.

I. Избыточность, локализованная внутри дизъюнктов. Таких видов избыточности два. Рассмотрим их, а также методы их устранения.

1. *Дублирование литер*. Литера теоретически может быть повторена в одном и том же дизъюнкте. Все дубликаты литеры могут быть эквивалентно устранены в соответствии с законом алгебры логики $x \vee x = x$.

2. *Контрарные литеры*. Теоретически в один и тот же дизъюнкте одна и та же переменная может включаться и в тождественной форме и с отрицанием. По закону логики $x \vee \neg x = 1$. Поэтому данный дизъюнкт в соответствии с законом логики $x \vee 1 = 1$ также равен 1, т.е. является тождественно истинным. Его по закону логики $x \& 1 = x$ можно эквивалентно удалить из формулы КНФ [6].

В обоих случаях удаление рассмотренных избыточностей I.1,I.2 представляет собой эквивалентное однодизъюнктное преобразование формулы.

II. Междизъюнктная избыточность. Она возникает при наличии таких групп дизъюнктов, среди которых некоторые могут быть устранены из общей формулы с сохранением свойства выполнимости. Такими видами избыточности являются следующие.

1. *Поглощение* (*subsumption* (SE)). Общая формула может включать пару дизъюнктов $\{C_i, C_j\}$ такого вида, что литеры C_i строго включаются в состав множества литер C_j. С использованием закона логики $x \& (x \vee y) = x$ (правило поглощения) короткий дизъюнкт эквивалентно поглощает более длинный [6].

Особенностью данного преобразования является то, что при его применении могут возникнуть свободные переменные.

При использовании правила поглощения требуется попарное сравнение всех дизъюнктов, входящих в формулу. Поэтому с использованием обычных структур данных можно разработать алгоритмы его реализации, имеющие сложность по длине входа $O(L^2)$.

2. *Эквивалентность дизъюнктов.* В формуле могут присутствовать пары таких дизъюнктов $\{C_i, C_j\}$, у которых множества литер (с учетом их возможного повторения) совпадают. При этом они могут находиться в дизъюнктах на разных местах. По правилу удаления дубликатов $x \& x = x$ один из дублирующих дизъюнктов можно эквивалентно устранить из формулы. В результате такого преобразования множество переменных сохраняется, не появляются ни определенные, ни свободные удаляемые переменные.

На традиционных структурах данных есть возможность построения алгоритмов для устранения данного вида избыточности, имеющих сложность по длине входа $O(L^2)$.

3. *Чистые литеры.* Каждую литеру, задающую тождественный или отрицательный тип вхождения некоторой переменной в формулу, называют *чистой* в том случае, когда в формулу не входит отрицание данной литеры. [7, 8]

В процессе устранения чистой литеры *lit* из формулы (правило чистых литер, PRL) ее переменная устраняется с таким значением в массив part_assign, при котором *lit* = 1. Оно помещается в массив part_assign, а по правилам $x \vee 1 = 1$, $x \& 1 = x$ все дизъюнкты, содержащие *lit*, могут быть удалены из формулы. При этом при каждом применении PRL к чистой литере в данном массиве создается ровно одна определенная переменная.

При каждом использовании PRL требуется одновременный анализ всего множества дизъюнктов формулы. Поэтому этот метод устранения избыточности относится к общеформульному типу.

Существенной особенностью данного вида избыточности является то, что в общем случае его к дизъюнктам длины 2 и выше является неэквивалентным преобразованием вследствие того, что значения, присваиваемые чистым литерам в этих дизъюнктах, всегда задают достаточные условия их истинности. Однако эти значения не всегда задают необходимые условия истинности этих дизъюнктов, поскольку она может достигаться и за счет значений других литер, входящих в них.

Максимальная сложность алгоритмов, реализующих устранение этого вида избыточности, достигается в случае итерационного применения правила PRL и равна $O(L^2)$.

4. *Контрарные пары дизъюнктов единичной длины*. Когда в формуле есть единичные дизъюнкты x_i и $\neg x_i$, то из закона логики $x \& \neg x = 0$ следует, что их произведение, а , следовательно, и вся формула будут ложными.

Данный метод является эквивалентным. Он реализуется алгоритмом, имеющим линейную сложность по входу $O(L)$.

III. **Смешанный тип избыточности**. Тип характеризуется тем, что при его применении осуществляется удаление не только дизъюнктов, но и отдельные литер в их составе. Его разновидностями являются следующие.

1. *Избыточность по единичным дизъюнктам*. Для устранения этого вида избыточности применяется метод распространения единиц (*unit propagation*) [4, 9]. При его использовании в тех дизъюнктах, которые содержат только одну литеру *lit*, значение ее переменной задают, исходя из условия *lit*=1, это значение включается в массив part_assign. Оно является не только достаточным, но

и необходимым для задачи (1). В том случае, когда *lit* включается в состав более длинного дизъюнкта, то в соответствии с правилом $x \vee 1=1$, $x\&1=x$ такой дизъюнкт превращается в тавтологию и эквивалентно удаляется из общей формулы. Если же *lit* входит в другой дизъюнкт с отрицанием, то в соответствии с логическим правилом $x \vee 0=1$ такой дизъюнкт остается в составе формулы, но из него удаляется *lit*.

Как и в случае правила PRL, при применении правила *unit propagation* могут появиться и свободные исключенные переменные. В то же время, в отличие от PLR, этот метод задает эквивалентный тип преобразования КНФ, так как задаваемые определяемым исключенным переменным значения являются не только достаточные, но и необходимыми для обеспечения истинности КНФ.

Устранение единичных дизъюнктов в общем случае приводит к появлению других аналогичных дизъюнктов. Поэтому устранение этого типа избыточности должно осуществляться итерационным способом. В этом случае соответствующий алгоритм может достигать максимальной квадратичной сложности.

2. *Самовоспроизводящаяся резолюция* (*SSR*) [10,11]. Избыточность возникает, когда в формулу входят два дизъюнкта следующего вида $C_1=(l \vee S_1)$, $C_2=(\neg l \vee S_2)$ и дополнительно выполняется условие $S_1 \subseteq S_2$. В этом случае литеру $\neg l$ можно эквивалентно исключить из дизъюнкта C_2. Это правило основано на следующей теореме: $(l \vee S_1)\&(\neg l \vee S_2) = (l \vee S_1)\&S_2$.

Метод является двухдизъюнктным. При его применении также могут возникнуть другие случаи такой избыточности, поэтому его также следует применять итерационным способом. Максимальная сложность алгоритма по длине входа является квадратичной.

5. Заключение. Выполненный анализ основных видов избыточности в формулах КНФ показывает, что сложность реализующих их алгоритмов не превышает квадратичную по длине входа. Следовательно, возможна разработка такого препроцессора, совместно реализующего данные методы, который будем иметь максимальную сложность, также не превышающую квадратичную.

Для оптимальной реализации такого препроцессора необходимо учесть, как свойства учитываемых в нем типов избыточности, так и особенности методов их устранения.

Список использованных источников

1. Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7(3):201–215, jul 1960.

2. Martin Davis, George Logemann, and Donald Loveland. A machine program for theoremproving. *Communications of the ACM*, 5(7):394–397, jul 1962.

3. J. P. Marques Silva and K. A. Sakallah, "GRASP-A new search algorithm for satisfiability," Proceedings of International Conference on Computer Aided Design, San Jose, CA, USA, 1996, pp. 220-227.

4. Anton Belov, Anto´nio Morgado, and Joao Marques-Silva. SAT-based preprocessing for MaxSAT. In *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 96–111. Springer Berlin Heidelberg, 2013.

5. Tuukka Korhonen, Jeremias Berg, Paul Saikko, and Matti Ja¨rvisalo. MaxPre: An extended MaxSAT preprocessor. In *Theory and Applications of Satisfiability Testing – SAT 2017*, pages 449–456. Springer International Publishing, 2017.

6. Marijn Heule, Matti J¨arvisalo, and Armin Biere. Clause elimination procedures for CNF formulas. In *Logic for Programming, Artificial Intelligence, and Reasoning*, pages 357–371. Springer Berlin Heidelberg, 2010.

7. Javier Larrosa, Federico Heras, and Simon de Givry. A logical approach to efficient max-SAT solving. *Artificial Intelligence*, 172(2-3):204–233, feb 2008.

8. Aolong Zha, Miyuki Koshimura, and Hiroshi Fujita. Introducing pure literal elimination into cdcl algorithm. 2016.

9. Hantao Zhang and Mark E. Stickel. An efficient algorithm for unit propagation. In In Proceedings of the Fourth International Symposium on Artificial Intelligence and Mathematics (AIMATH'96), Fort Lauderdale (Florida USA, pages 166–169, 1996.

10. Konstantin Korovin. iProver – an instantiation-based theorem prover for first-order logic (system description). In *Automated Reasoning*, pages 292–298. Springer Berlin Heidelberg, 2008.

11. Niklas E´en and Armin Biere. Effective preprocessing in SAT through variable and clause elimination. In *Theory and Applications of Satisfiability Testing*, pages 61–75. Springer Berlin Heidelberg, 2005.