Модел. и анализ информ. систем. Т. **17**, № **4** (2010) 52–59

УДК 517.51+514.17

Проверка моделей распределенных систем с помощью аффинного представления данных

Гаранина Н.О.[1]

Институт систем информатики им. А.П. Ершова СО РАН

e-mail: garanina@iis.nsk.su

получена 17 сентября 2010

Ключевые слова: символьная проверка моделей, распределённые системы

Предложено эффективное символьное представление распределенных систем, определяемых линейными функциями над целочисленными переменными.

1. Введение

Основным недостатком метода проверки на модели является "комбинаторный взрыв" в пространстве состояний, который возникает, когда система состоит из компонентов, переходы в которых выполняются параллельно. Ясно, что с увеличением количества процессов число глобальных состояний системы возрастает экспоненциально. В первоначальных реализациях алгоритмов проверки на модели отношения переходов явно представлялись списками смежности, а множества состояний задавались явным перечислением, что делало невозможным проверку сложных систем.

В конце 80-х гг. благодаря использованию двоичных разрешающих диаграмм (OBDD) Бриана [2] в качестве компактного символьного представления данных стало возможным верифицировать очень сложные системы [5]. Суть *символьной* проверки на модели состоит в том, чтобы проверять свойства моделей не в каждом отдельном состоянии (как в явных алгоритмах), а сразу на множестве состояний, что позволяет при подходящей кодировке множеств состояний проверять очень большие модели. В связи с этим разрабатывались различные представления множеств состояний.

Однако оказалось, что символьное представление в виде OBDD вполне подходит для моделирования последовательных схем и протоколов, состояния которых кодируются булевскими переменными, но для систем с целочисленными состояниями такое представление оказывается довольно громоздким. Рассмотрим некоторые

[1]Работа выполнена при финансовой поддержке Интеграционного гранта № 2/12 Сибирского Отделения Российской Академии Наук.

представления, существенно использующие при кодировании тот факт, что элементами пространства состояний являются целые числа.

В 1994 г. Б. Бугло и П. Волпер предложили периодические множества для представления множеств состояний [1]. В этой работе периодические множества используются только для вычисления множества достижимых состояний. Модели в [1] – это машины с конечным числом состояний, параметризованных неограниченными целыми переменными, и со следующими операциями: присваивание константы, сложение константы с переменной и проверка пусковых линейных неравенств. Основным недостатком такого представления является то, что оно не допускает полностью символьной проверки на моделях: вычисленное множество достижимых состояний может оказаться слишком большим для проверки спецификации модели во всех его элементах.

В 1999 г. Т. Булган, Р. Гербер и В. Пух разработали систему для символьной проверки бесконечных моделей, представляя множества состояний формулами арифметики Пресбургера [3]. Они использовали программу Omega Library [10] для символьных манипуляций с формулами Пресбургера. Модели, проверяемые с помощью этой системы, также можно описать формулами арифметики Пресбургера. К явным достоинствам такого способа представления данных относится то, что можно проверять сколь угодно большие и даже бесконечные модели, но недостатком является тот факт, что сложность оперирования формулами Пресбургера вычислительно дорога: она экспоненциально зависит от размера формул-представлений. Кроме того, для некоторых бесконечных моделей алгоритм проверки спецификации может не завершаться.

Формат представления данных, предложенный в настоящей работе, позволяет осуществлять полностью символьную проверку на широком классе моделей, при этом размер представления значительно меньше размера модели (в отличие от OBDD для этого класса моделей) и алгоритмы манипулирования ими имеют в основном квадратичную временную сложность относительно размера представления.

Оставшаяся часть работы организована следующим образом. В разд. 2. полуформально описываются модели, для которых возможно аффинное представление, разд. 3. содержит определение аффинных можеств и алгоритмы операций над ними, в разд. 4. приводится краткое описание обобщений аффинных множеств на аффинные векторы и деревья и разд. 5. — заключение. Предполагается, что основные понятия, относящиеся к теории и практике проверки моделей программ, читателю известны.

2. Модели

В основе известных символьных алгоритмов проверки на модели лежит вычисление неподвижной точки монотонного оператора, заданного формулой, содержащей булевскую комбинацию пропозициональных переменных, констант и операторов действий, применённых к формуле, описывающей множество состояний. Поэтому вместо исходных формул любой логики, для которой возможна символьная проверка, можно проверять семантически эквивалентные формулы μ-исчисления [4].

В силу определения семантики формул μ-исчисления в ходе выполнения алгоритмов символьной проверки на модели необходимо осуществлять следующие операции над множествами: объединение и пересечение множеств, вычисление предусловий (соответствующие дизъюнкции, конъюнкции и операторам переходов), а также проверка множеств на включение (необходимое для проверки стабилизации неподвижной точки). Дополнение как операция над множествами нам не понадобится, поскольку всякая формула μ-исчисления приводится к нормальной форме.

Опишем класс моделей, для которых возможна аффинная кодировка, а именно — аффинных моделей. Пространство состояний в таких моделях является декартовым произведением n отрезков целых чисел, где n — число переменных, определяющих состояние модели, пропозициональные константы задаются множеством отрезков, детерминированные переходы — линейными векторными функциями, определяемыми целочисленными матрицами, у которых в каждой строке не больше одного элемента, и целочисленными векторами. Иными словами, в этот класс попадают все конечные модели, не допускающие перемножения переменных друг на друга (умножение на константу возможно). Язык, описывающий такие модели, допускает только конечные типы данных, в выражениях сравнения может быть не больше одной переменной, недетерминизм реализуется объединением детерминированных действий, а также недетерминированным выбором из области определения переменной, возможны условные действия, а спецификация свойств моделей может выражаться формулами μ-исчисления или менее выразительных логик. Примеры языков, описывающих такие модели, есть в [9] и [8]. Для оперирования аффинными представлениями необходимо нечисловые типы представить как числовые, сопоставив каждому элементу множества число.

3. Ограниченные аффинные множества

Для представления множества состояний и действий проверяемой модели мы используем (ограниченные) аффинные множества, т.е. множества, определяемые конечным набором линейных двучленов вида $ax + b$ с целыми коэффициентами a и b, каждый из которых имеет свою область определения в виде отрезка целых чисел $[m..M]$ (возможно, пустого).

Для начала будем считать, что состояния модели определяются значениями единственной переменной x (очевидно, такая модель аффинна), определённой на отрезке целых чисел $[n..N]$, причём её значения после выполнения действий определяются линейными двучленами. Пропозициональные константы и условия пуска в таких моделях представляют собой булевские комбинации *атомов* вида: "$x = k$", "$x > k$" или "$x < k$", где k — целое число.

Представление пропозициональных констант и действий

Аффинное представление атомов и их отрицаний определяется в табл.1.

Аффинное представление как булевской комбинации P этих атомов, так и её отрицания $\neg P$, можно получить, преобразуя P к нормальной форме (отрицания есть только у атомов) и применяя затем к атомам операции конъюнкции и дизъюнкции, реализующиеся в виде пересечения и объединения интервалов. Тогда аффинное

Таблица 1. Аффинное представление атомов и их отрицаний

	$x = k$	$x < k$	$x > k$
P	$(k, \emptyset)$	$(x, [n, k-1])$	$(x, [k+1, N])$
$\neg P$	$\{(x, [n, k-1]), (x, [k+1, N])\}$	$(x, [k, N])$	$(x, [n, k])$

представление пропозициональной константы P принимает вид набора аффинных представлений атомов:

$$P = \{(k_1, \emptyset), \ldots, (k_i, \emptyset), (x, [k_{i+1}, k'_{i+1}]), \ldots, (x, [k_P, k'_P])\}.$$

В процессе символьной проверки на модели в результате операций над пропозициональными константами получаются множества, определяемые набором линейных двучленов вида $ax + b$ с целыми коэффициентами a и b на отрезке целых чисел $[m..M]$. В дальнейшем такие выражения вида $f = (ax + b, [m, M])$ будем называть *атомами*, а их наборы $P = \cup_i(f_i)$ — *ограниченными аффинными множествами*, списками, на которых определены следующие операции: $P.add(atom)$ (добавить атом к представлению P), $P.remove(atom)$ (удалить атом из представления P).

Допустимые атомарные детерминированные переходы представляются в следующем виде: $r = (ax + b)$ $(a,\ b \in \mathbb{Z})$. Здесь переменной присваивается значение этого выражения при текущем значении переменной x из области определения. $A = \{(Q_1, r_1), \ldots, (Q_m, r_m)\}$ — это аффинное представление допустимых действий, где Q_i — аффинное представление пропозиционального условия пуска, а r_i — аффинное представление атомарного перехода. Эта запись означает недетерминированный выбор из атомарных переходов по условию.

Операции на аффинных множествах

Пусть P, Q, R будут ограниченными аффинными множествами, представляющими некоторые множества состояний аффинной модели. Если $f = (ax + b, [m, M])$ — некоторый атом такого представления, то пусть $f.Dom = [m, M]$.

1. Объединение: $R = P \cup Q$. Это обычное объединение множеств P и Q.

$Joint(P, Q)$:

1. **forall** $f \in P$, $g \in Q$ $\quad R.add(f)$; $R.add(g)$;

2. Пересечение: $R = P \cap Q$. Здесь нужно найти общие значения атомов, составляющих множества, для каждого из множеств P и Q. Пусть $P = F \cup C_f$ и $Q = G \cup C_g$, где C_f и C_g — множества констант. Пусть $Solve(f(x), g(y))$ — это процедура, которая находит все решения диофантова уравнения $f(x) = g(y)$ в виде аффинных множеств $x(z)$ и $y(z)$, где z — целое, т.е. если $z = k$, то $f(x(k)) = g(y(k))$. Время работы $Solve(f(x), g(y))$ логарифмически зависит от коэффициентов функций $f(x)$ и $g(y)$.

$Intersection(P, Q)$:

I. Сначала найдём общие константы в P и Q.

1. **forall** $c_f \in C_f$, $c_g \in C_g$ **forall** $g \in Q$, $f \in P$
2. **if** $(c_f = g) \bigvee (\exists x \in g.Dom : g(x) = c_f)$ **then** $R.add(c_f)$;

3. **if** $(c_g = f) \bigvee (\exists x \in f.Dom : f(x) = c_g)$ **then** $R.add(c_g)$;

II. Найдём общие значения невырожденных двучленов из P и Q.

1. **forall** $f \in F$ **forall** $g \in G$
2. $Solve(f(x), g(y))$; // Получили $x(z)$ и $y(z)$.
3. **Let** $Z_x \subset \mathbb{Z}$ **that** $x(Z_x) \subseteq f.Dom$; // Z_x и Z_y — отрезки.
4. **Let** $Z_y \subset \mathbb{Z}$ **that** $y(Z_y) \subseteq g.Dom$;
5. $h.Dom := Z_x \cap Z_y$;
6. $h := f(x(z))$;
7. $R.add(h, h.Dom)$;

3. Вычисление предусловия детерминированного атомарного перехода: $R = (a)P$. Необходимо вычислить множество R, из которого после перехода a попадаем в подмножество множества P. Пусть $g(y)$ — это аффинное представление перехода a. Предусловием являются такие $y(z)$, что при подстановке их в функцию перехода $g(y)$ она становится равной $f(x(z))$. Кроме того, $y(z)$ имеют область определения, соответствующую $f(x)$, где $f \in P$.

$PreImageDet(a, P)$:

1. **forall** $f \in P$
2. $Solve(f(x), g(y))$; // Пусть теперь $x(z) = cz + d$ и $y(z) = ez + t$
3. **Let** $h.Dom \subset \mathbb{Z}$ **that** $(x(h.Dom) \subseteq f.Dom)$ **and** $(y(h.Dom) \subseteq [n..N)$;
4. $h := ez + t$;
5. $R.add(h, h.Dom)$;

4. Вычисление предусловия: $R = [A]P$ или $R = \langle A \rangle P$. Вычисление предусловия основано на равенствах: $[A]P = \bigcap_i (Q_i \rightarrow [a_i]P)$ и $\langle A \rangle P = \bigcup_i (Q_i \wedge \langle a_i \rangle P)$. В силу семантики формул μ-исчисления для детерминированных переходов a_i и множества P, верно, что $[a_i]P = \langle a_i \rangle P = (a_i)P$. Тогда предусловие действия $A = \{(Q_1, r_1), \ldots, (Q_m, r_m)\}$, очевидно, вычисляется так:

$PreImage(A, P, ())$:

1. **if** $(\ ) = [\ ]$ **then** $R := true$; **else** $R := false$;
2. **for** $i = 1..m$
3. **if** $(\ ) = [\ ]$ **then** $R := R \cap (\neg Q_i \cup (a_i)P)$; **else** $R := R \cup (Q_i \cap (a_i)P)$;

Временная сложность этих алгоритмов линейно зависит от произведения мощностей аффинных представлений входных данных, кроме объединения, где она зависит от суммы этих мощностей.

Оптимизация

В процессе исполнения символьного алгоритма проверки модели неизбежно разрастание аффинных множеств как следствие выполнения операций объединения. Поэтому имеет смысл оптимизация аффинного представления, т.е. удаление повторяющихся значений элементов. Для этого можно последовательно выполнить следующие процедуры: (1) уменьшить области определения двучленов, если их образы целиком содержатся в образах (экстраполяции) множества двучленов; (2) удалить константы, содержащиеся в образах (экстраполяции) каких-либо двучленов. Под экстраполяцией понимается такое расширение области определения двучлена, что оно покрывает образ преобразуемого двучлена. Область определения какого-либо двучлена может уменьшиться настолько, что его можно будет заменить константой

или удалить. Имеет смысл повторять эти процедуры, пока представление множества не стабилизируется. Здесь временная сложность линейно зависит от размера аффинного представления и мощности типа переменной, то есть размера модели в данном случае. Подробное описание алгоритмов оптимизации приводится в [9, 8].

Однако, если ограничиться моделями, в которых действия определяются только сложением переменных с константой, то все аффинные атомы будут просто отрезками целых чисел или константами. Оптимизация аффинных множеств, состоящих из таких атомов, значительно проще и состоит из склеивания и поглощения атомов и констант. Здесь же сложность процедуры линейно зависит исключительно от размера аффинного представления. Подробности приведены в [9]. Более того, после повторения этих процедур до стабилизации множества атомов, аффинное представление таких простых множеств, очевидно, становится однозначным, то есть каноническим, что допускает возможность синтаксического сравнения множеств.

Проверка включения

Сравнение множеств в алгоритмах символьной проверки необходимо для проверки стабилизации неподвижных точек. Так как функция, для которой вычисляется неподвижная точка, монотонна, достаточно проверять включение одного из множеств в другое.

Так как каноническое аффинное представление произвольных множеств невозможно, множества придётся сравнивать путём сравнения его подмножеств, а не синтаксически. Идея алгоритма сравнения проста: используем процедуру решения диофантовых уравнений $Solve(f(x), g(y))$, чтобы определить, какие значения $f(x)$ содержатся в образе $g(y)$, и удаляем соответствующие значения аргумента из области определения $f(x)$. Если в конце процедуры окажется, что аффинное представление пусто, то это множество включено в другое. Временная сложность проверки включения оказывается равной произведению мощностей аффинных представлений сравниваемых множеств на мощность типа переменной. Точные алгоритмы приведены в [8].

Для оптимизированных *простых* аффинных множеств, представляющих исключительно константы и отрезки целых чисел, процедура сравнения является линейной чисто синтаксической проверкой совпадения множеств, поскольку оптимизация таких аффинных множеств, описанная в предыдущем разделе, вычисляет их однозначное представление.

4. Векторно-аффинные множества и деревья

Аффинное представление множеств можно обобщить для аффинных систем со многими переменными. Идея состоит в том, чтобы каждое множество состояний модели закодировать с помощью набора векторов, компонентами которых являются аффинные атомы, соответствующие значениям переменных модели. Назовём это представление *векторно-аффинным*. Алгоритмы манипуляций с данными для векторно-аффинного представления аналогичны алгоритмам для аффинного, но имеют свои особенности: (1) при вычислении предусловия (поскольку переменные могут зависеть друг от друга); (2) при оптимизации и проверке включения (так как сравни-

ваются одновременно несколько множеств, а не два). Самую высокую временную сложность имеет алгоритм проверки включения, зависящий от мощности представлений аффинных множеств и мощности типов переменных. Эти алгоритмы подробно описаны в [8].

Для проверки мультиагентных систем, в которых агенты обладают абсолютной памятью, векторно-аффинные множества можно развить в векторно-аффинные деревья. Это древовидные структуры данных с аффинными векторами в вершинах и рёбрами, помеченными именами агентов. Алгоритмы манипуляций данными для этих структур основаны на алгоритмах для векторно-аффинных множеств, но за счёт того, что размер самих деревьев экспоненциален относительно количества агентов в системе, эти алгоритмы имеют значительно более высокую временную сложность. Алгоритмы для этих обобщений подробно описаны в [8].

5. Заключение

Предложенное представление данных в виде аффинных множеств, являющихся набором двучленов, определённых на отрезках целых чисел, является весьма эффективным для проверки аффинных моделей с единственной переменной.

Для класса аффинных моделей с несколькими переменными также предложено эффективное векторно-аффинное представление пропозициональных констант и действий модели. В этом случае множества состояний модели определяются набором векторов, компонентами которых являются двучлены, определённые на отрезках целых чисел, соответствующие значениям переменных модели.

Представленные здесь алгоритмы манипуляции с данными самое большее квадратичны относительно размера аффинных представлений, кроме алгоритмов проверки включения, верхняя оценка которых зависит ещё и от размера областей определения переменных. Однако для моделей, не допускающих умножения переменных на константы, сложность этих алгоритмов линейна относительно размера представления.

Отдельно отметим векторно-аффинное представление деревьев знаний, которое позволило значительно снизить сложность проверки моделей для асинхронных мультиагентных систем с абсолютной памятью. Для таких систем был разработан прототип инструмента проверки аффинных моделей с одним агентом со свойствами, специфицированными формулами комбинированной логики μC+PLK$_1$[7], использующий алгоритм проверки из [6].

Однако в силу экспоненциальной сложности верификации логик знаний и времени в мультиагентных системах с абсолютной памятью, которую снизить уже никак невозможно, прототип этого инструмента оказался практически непригоден для решения хоть сколько-нибудь значимых задач. Поэтому в настоящем на основе векторно-аффинных представлений данных реализуется система проверки на произвольных аффинных моделях свойств, выраженных формулами μ-исчисления и менее выразительных логик, таких как LTL и CTL.

Список литературы

1. **Boigelot B. and Wolper P.** Symbolic Verification with Periodic Sets // Lect. Notes Comput. Sci. 1994. Vol. 818. P. 55–67.

2. **Bryant R.E.** Symbolic boolean manipulation with ordered binary decision diagrams // IEEE Trans. Computers. 1986. Vol. C-35, N 8. P. 293–318.

3. **Bultan T., Gerber R., and Pugh W.** Model Checking Concurrent Systems With Unbounded Integer Variables: Symbolic Representations, Approximations and Experimental Results // ACM Trans. Progr. Lang, and Systems. 1999. Vol. 21, N 4. P. 747–789.

4. **Kozen D.** Results on the Propositional Mu-Calculus // Theor. Comput. Sci. 1983. Vol. 27, N 3. P. 333–354.

5. **McMillan K.L.** Symbolic Model Checking: An Approach to the State Explosion Problem. Kluwer Academic Publishers, 1993. 216 p.

6. **Shilov N.V., Garanina N.O.** A Polynomial Approximations for Model Checking // Lect. Notes Comput. Sci. 2003. Vol. 2890. P. 395–400.

7. **Shilov N.V., Garanina N.O., Kalinina N.A.** Model checking knowledge, actions and fixpoints // Proc. Int. Workshop on Concurrency, Specification and Programming. Caputh, Germany, 2004. V. 2. P. 351–357.

8. **Гаранина Н.О.** Верификация распределенных систем с использованием аффинного представления данных, логик знаний и действий: Дис. ... канд. физ.-мат. наук. Новосибирск, 2004. 172 с.

9. **Гаранина Н.О.** Аффинное представление данных для проверки моделей программ. Новосибирск, 2004. 48 с. (Препр./ Сиб. отд-ние. РАН. ИСИ; N 116)

10. http://www.cs.umd.edu/projects/omega/

Model Checking of Distributed Systems with Affine Data Structures

Garanina N.O.

Keywords: symbolic model checking, distributed systems

A new data structure is suggested for symbolic model checking of distributed systems defined by linear functions of integer variables.

Сведения об авторе:
Гаранина Наталья Олеговна,
Институт систем информатики им. А.П. Ершова СО РАН, научный сотрудник.