Модел. и анализ информ. систем. Т. **18**, № **4** (2011) 56–67

УДК 517.51+514.17

Оптимизационные процедуры в аффинной проверке моделей

Гаранина Н.О.[1]

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

e-mail: garanina@iis.nsk.su

получена 18 ноября 2011 года

Ключевые слова: символьная проверка моделей, алгебраические представления данных, распределенные системы

Символьная проверка на модели основана на компактном представлении множеств. На данный момент есть три основных направления символьной проверки моделей: методы, основанные на бинарных разрешающих диаграммах, ограниченная проверка моделей, использующая SAT-решатели, и различные алгебраические подходы к эффективному представлению данных. В данной работе предлагается рассмотреть улучшенные алгоритмы манипуляции с алгебраическим представлением данных, а именно алгоритмы оптимизации аффинных представлений данных.

1. Введение

Проверка моделей является хорошо автоматизируемой техникой верификации для параллельных систем с конечным числом состояний. В данном подходе (в его классическом виде) для проверки логических спецификаций, как правило, ведется исчерпывающий поиск по всему пространству состояний модели [8]. Однако, с увеличением размеров систем, число их состояний экспоненциально возрастает, что делает способы проверки моделей, напрямую использующие явное перечисление пространства состояний, неэффективными.

Одним из возможных способов борьбы с комбинаторным взрывом является уклонение от прямого перечисления состояний модели. Этот подход, известный как символьная проверка моделей, подразумевает представление множеств состояний и переходов между ними в каком-либо компактном виде. Традиционно, начиная с

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

пионерской работы К. МакМиллана [10], в символьной проверке моделей применялось бинарное кодирование множеств, что позволяло в процессе проверки использовать булевские функции, представленные в виде бинарных разрешающих диаграмм (BDD) [5]. Однако существуют другие представления множеств состояний, такие как конъюнктивные нормальные формы (CNF), используемые в ограниченной проверке моделей с помощью пропозициональных SAT-решателей [11], а также различные алгебраические подходы к решению проблемы. Последние отличаются тем, что с их помощью возможно значительно более эффективное представление таких моделей, в которых пространства состояний в значительной степени определяются целыми числами. Известно, что BDD- и CNF-представления для таких моделей оказываются существенно более громоздкими [5].

В 1994 г. Б. Бугло и П. Волпер предложили периодические множества для представления множеств состояний [3]. В этой работе периодические множества используются только для вычисления множества достижимых состояний. Обобщением этого подхода является регулярная проверка моделей [1]. Эта техника разработана для проверки классов бесконечных систем, конфигурации которых могут быть представлены как слова в конечном алфавите. Примерами таких моделей могут являться параметризованные системы, состоящие из произвольного числа однородных процессов, связанных линейным или кольцевым образом, а также системы, оперирующие на очередях, стеках, целых числах и других линейных структурах данных. Основная идея регулярной проверки моделей состоит в том, чтобы использовать регулярные языки для представления множеств конфигураций и конечные преобразователи состояний для описания отношений переходов в системах.

В 1999 г. Т. Булган, Р. Гербер и В. Пух разработали систему для символьной проверки бесконечных моделей, представляя множества состояний и переходов формулами арифметики Пресбургера [6]. Они использовали систему Omega Library [17] для символьных манипуляций с формулами Пресбургера. Такой подход называется проверкой моделей на основе ограничений [7]. Эти линейные ограничения, выраженные на языке арифметики Пресбургера, можно эффективно представлять также с помощью автоматов [2]. Также интересен комбинированный подход, совмещающий в себе линейные ограничения и BDD [13].

В работах [14, 15, 16] для представления множеств состояний конечных систем с целыми числами были предложены аффинные структуры данных. Отличие от упомянутых выше подходов к кодированию целочисленных множеств состоит в возможности использования такого подхода только для конечных систем и связанной с этим меньшей временной сложности алгоритмов манипулирования данными, а также возможностью более компактного представления для широкого класса систем.

Однако все вышеперечисленные алгебраические подходы имеют одну и ту же проблему разрастания представлений множеств в процессе проверки модели. В регулярной проверке моделей эта проблема частично решается методом экстраполяций [4]. В комбинированном методе проверки моделей на основе ограничений используются оптимизирующие алгоритмы, уменьшающие размер представления [13], которые также подходят для обычных представлений ограничений.

В настоящей работе предложен новый алгоритм оптимизации модифицированных векторно-аффинных представлений данных, имеющий квадратичную временную сложность относительно размера представления.

Оставшаяся часть работы организована следующим образом. В разделе 2 описываются модели, для которых возможно векторно-аффинное представление, в разделе 3 содержится определение векторно-аффинных множеств и алгоритмы операций над ними, приведены связанные алгоритмы оптимизации и проверки включения для векторно-аффинных множеств; раздел 4 — заключение. Предполагается, что основные понятия, относящиеся к теории и практике проверки моделей программ, читателю известны.

2. Модели

Класс моделей, для которых возможно аффинное представление, а именно — аффинных моделей, можно неформально описать следующим образом. Пространство состояний в таких моделях является декартовым произведением n отрезков целых чисел, где n — число переменных, определяющих состояния аффинной модели, семантика пропозициональных констант задается декартовым произведением множеств отрезков[2], детерминированные параллельные переходы — линейными векторными функциями, определяемыми целочисленными матрицами, у которых в каждой строке не больше одного ненулевого элемента, и целочисленными векторами. В моделях возможен недетерминизм, а также выполнение перехода по условию, задаваемому пропозициональной константой. В этот класс попадают конечные модели, допускающие умножение переменной на константу, ее сложение и сравнение с константой. В частности, некоторые протоколы передачи данных с ограниченной нумерацией кадров принадлежат этому классу. Формальное описание аффинных моделей, а также примеры языков, задающие их, можно найти в [14, 15].

3. Ограниченные векторно-аффинные множества

Определения ограниченных[3] векторно-аффинных множеств и операций с ними опираются на определения аффинных множеств из [16] и [14].

Идея векторно-аффинного представления множеств состоит в том, чтобы каждое множество состояний модели описать набором векторов, компонентами которых являются аффинные множества, соответствующие значениям переменных модели.

Далее считаем, что состояния модели определяются значениями переменных из множества $Var = \{x_1, \ldots, x_n\}$, определенных на отрезках целых чисел $[n_i..N_i](i =$

[2]В частности, пропозициональной константе $false$ соответствует пустое множество, а $true$ — само пространство состояний целиком.

[3]В данной работе и связанных с ней предшествующих аффинные, векторно-аффинные множества и деревья ограничены по определению. Однако можно рассматривать вариант подобного представления, в котором переменные имеют неограниченную счетную область определения. Вопрос эффективности таких представлений не исследовался.

$1..n$), причем значения этих переменных после выполнения действий модели определяются линейными двучленами вида $ax_i + b$, где $a, b \in \mathbb{Z}, x_i \in Var$. Пропозициональные константы и условия пуска в аффинных моделях задаются булевскими комбинациями *атомов* вида: "$x_i = k$ "$x_i \neq k$ "$x_i > k$" или "$x_i < k$" и их отрицаний, где $x_i \in Var$ и k — целое число.

3.1. Представление пропозициональных констант и действий

Пусть $i \in [1..n]$, $l_i \in \mathbb{N}$, $j_i \in [1..l_i]$, $a_{j_i}, b_{j_i} \in \mathbb{Z}$, $[k_{j_i}..K_{j_i}] \subset \mathbb{Z}$, $v_i = \bigcup_{j_i \in [1..l_i]}\{(a_{j_i}x + b_{j_i}, [k_{j_i}..K_{j_i}])\}$ — аффинное множество, и его семантика — это множество целых чисел $\|v_i\| = \{y \mid y \in \bigcup_{j_i \in [1..l_i]}\{a_{j_i}x + b_{j_i} \mid x \in [k_{j_i}..K_{j_i}]\}\}$. Назовем вектор вида $V = (v_1, \ldots, v_n)$ *аффинным вектором*. Он символически представляет n-ки значений своих компонент при всевозможных значениях их аргументов из области определения, т.е. $\|V\| = \{(y_1, \ldots, y_n) \mid y_i \in \|v_i\|\}$ — семантика аффинного вектора. Очевидно, $\|V\| \subset \mathbb{Z}^n$. *Ограниченное векторно-аффинное множество* — это конечный набор аффинных векторов.

Векторно-аффинное представление атома — это векторно-аффинное множество, содержащее один аффинный вектор, получающийся следующим образом. Пусть $D_i = \{(x, [n_i..N_i])\}$ — обозначение аффинного представления домена переменной. Тогда векторно-аффинным представлением атома будет аффинный вектор, все компоненты которого, кроме i, равны аффинным представлениям домена соответствующих переменных, а i-я компонента — это аффинное множество, содержащее один *аффинный атом* вида $(m, \emptyset)$ либо $(x, [m..M])$ $(m, M \in \mathbb{Z})$, или объединение аффинных атомов, представляющие данный атом:

$(x_i = k)$**:** $(D_1, \ldots \{(k, \emptyset)\}_i, \ldots D_n)$;

$(x_i \neq k)$**:** $(D_1, \ldots \{(x, [n_i..k-1]), (x, [k+1..N_i])\}_i, \ldots D_n)$;

$(x_i < k)$**:** $(D_1, \ldots \{(x, [n_i..k-1])\}_i, \ldots D_n)$ (для $x_i > k$ аналогично).

Векторно-аффинное представление булевской комбинации атомов можно получить из векторно-аффинных представлений атомов, осуществляя пересечение и объединение векторно-аффинных множеств, описанные ниже.

Пусть $A^i = \bigcup_{j_i=1}^{m_i}(Q_{j_i}?; a_{j_i})$ — это PDL-подобное представление недетерминированного допустимого действия в модели относительно переменной системы x_i $(i \in [1..n])$, где Q_{j_i} — векторно-аффинное представление пропозиционального условия пуска, а a_{j_i} — представление детерминированного атомарного перехода $c_{j_i}x_k + d_{j_i}$ $(k \in [1..n],\ c_{j_i}, d_{j_i} \in \mathbb{Z})$. Такое действие присваивает переменной x_i значение какого-либо a_{j_i} при текущем значении переменной x_k и при условии выполнения Q_{j_i}. В каждый момент времени действия модели относительно всех переменных выполняются одновременно. Пусть $Idx = \{(j_1, ..., j_n) \mid j_i \in [1..m_i],\ i \in [1..n]\}$ – векторы индексов детерминированных атомарных переходов в таких представлениях. Обозначим одновременное исполнение детерминированных атомарных переходов a_{j_i} и a_{j_k} как $a_{j_i} \times a_{j_k}$. Зафиксируем вектор индексов $J = (j_1, ..., j_n) \in Idx$. Пусть $Q_J = \bigcap_{i \in [1..n]} Q_{j_i}$.

Тогда представление *детерминированного векторного атомарного перехода* (относительно всех переменных) — это $A_J = Q_J?;(a_{j_1} \times a_{j_2} \times ... \times a_{j_n})$. Представление *недетерминированного действия модели* выглядит следующим образом: $A = \bigcup_{J \in Idx} A_J$.

3.2. Операции на векторно-аффинных множествах

В основе многих символьных алгоритмов проверки на модели лежит вычисление неподвижной точки монотонного оператора, заданного формулой, содержащей булевскую комбинацию пропозициональных переменных, констант и операторов действий (темпоральных и т.п.), примененных к формулам, описывающим множества состояний. Таким образом, вместо исходных формул любой логики, менее выразительной, чем μ-исчисление [9], можно проверять семантически эквивалентные формулы μ-исчисления.

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

Пусть P, Q, R будут векторно-аффинными множествами, представляющими некоторые множества состояний модели.

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

2. Пересечение: $R = P \cap Q$. В пересечении R лежат аффинные векторы, семантика компонент которых является пересечением семантик компонент векторов из P и Q. Особенность векторно-аффинного пересечения состоит в том, что если одно из покомпонентных пересечений пусто, то такой вектор не принадлежит R. Поскольку компоненты аффинных векторов — это аффинные множества, то их пересечение вычисляется при помощи стандартной процедуры решения диофантовых уравнений $Solve$, которая ищет все решения диофантова уравнения $f(x) = g(y)$ как линейные двучлены $x(z)$ и $y(z)$, где z – новая переменная, т.е. при всяком $z = k$, где $k \in \mathbb{Z}$, верно $f(x(k)) = g(y(k))$. Формальное описание процедуры вычисления пересечения аффинных множеств приведено в [14].

3. Вычисление предусловия детерминированного векторного атомарного перехода: $R = (a)P$. Необходимо вычислить векторно-аффинное множество R, из которого после детерминированного векторного атомарного перехода a попадаем во множество P. Прообразы компонент аффинных векторов из множества P вычисляются аналогично вычислению предусловий аффинных атомарных переходов [14, 16]. Особенностью вычисления предусловий для векторного атомарного перехода является то, что вычисляются прообразы компоненты, соответствующей переменной модели из двучлена перехода. Если прообраз одной и той же переменной

определяется разными компонентами векторных атомарных переходов, то прообразом соответствующей компоненты является их пересечение, согласно параллельной семантике переходов между состояниями. Если пересечение пусто, то в данный аффинный вектор не существует перехода посредством данного векторного атомарного перехода. Если переменная не встречалась в описании векторного атомарного перехода, то прообразом соответствующей компоненты аффинного вектора является аффинное представление области значений этой переменной.

4. Вычисление предусловия недетерминированного действия модели: $R = [A]P$ или $R = \langle A\rangle P$. Вычисление предусловия основано на стандартных равенствах [9]: $[A]P = \bigcap_i (Q_i \to [a_i]P)$ и $\langle A\rangle P = \bigcup_i (Q_i \wedge \langle a_i\rangle P)$, где $A = \bigcup_i (Q_i?; a_i)$ — недетерминированный выбор из детерминированных переходов a_i с условием пуска Q_i. По семантике 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 вычисляется с использованием предусловия детерминированного векторного атомарного перехода очевидным образом.

Временная сложность алгоритмов, основанных на вышеизложенных идеях и приведенных в [14], не более чем квадратична относительно входных данных.

3.3. Оптимизация и проверка включения

Оптимизация векторно-аффинного представления необходима, так как в результате выполнения объединения векторно-аффинных множеств в результирующем множестве могут появиться различные векторы, представляющие одни и те же множества состояний. Под оптимизацией здесь понимается уменьшение размера представления. В первую очередь необходимо провести оптимизацию каждой компоненты каждого вектора множества с помощью алгоритмов оптимизации обычных аффинных множеств [14, 16]. Затем осуществляется повекторная оптимизация посредством покомпонентного сравнения семантик векторов представления. Если семантика одной из компонент аффинного вектора v пересекается с семантикой соответствующей компоненты w — одного из векторов оптимизируемого множества — и семантики остальных компонент v полностью содержатся в семантиках соответствующих компонент w, то можно изменить эту компоненту, чтобы соответствующие семантики не пересекались. Для проверки включения семантик используется процедура решения диофантовых уравнений $Solve$. Чтобы избежать линейного разрастания множеств, изменение “пересекающейся” компоненты осуществляется только в том случае, если итоговое векторно-аффинное множество по размеру меньше исходного. В данной работе представлен новый алгоритм оптимизации $VOpt$ (см. приложение А), результатом работы которого для входного векторно-аффинного множества P является векторно-аффинное множество $R = VOpt(P)$. Сложность $VOpt$, в отличие от предыдущей версии алгоритма, опубликованного в [14], оказывается квадратично зависимой только от размера представления множеств состояний, но не от размера модели.

Как следует из определения векторно-аффинных множеств и алгоритма оптимизации, существуют “плохие” множества, которые невозможно оптимизировать. В

такие множества входят векторы, содержащие различные компоненты с одинаковыми номерами. Оценим размер этих множеств. Пусть для простоты мощность всех областей определения переменных равна D. Тогда верно следующее утверждение.

Утверждение 1. *Размер векторно-аффинного множества B, которое невозможно оптимизировать с помощью алгоритма $VOpt$, не больше*

$$\sum_{i=2}^{n} D^{n-i+1} \cdot C_n^i,$$

где n – количество переменных модели, а C_n^i – число сочетаний из n по i.

Доказательство (эскиз). Векторно-аффинное множество, состоящее из двух векторов $v = (v_1, ..., v_n)$ и $v' = (v'_1, ..., v'_n)$, может быть оптимизировано, тогда и только тогда, когда существует $i \in [1..n]$, такое что $\|v_i\| \cap \|v'_i\| \neq \emptyset$ и $\|v_j\| \subseteq \|v'_j\|$ для всех $j \neq i$ ($j \in [1..n]$). В аффинных векторах v и v' компоненты v_i и v'_i назовем *различающимися*, если $\|v_i\| \not\subseteq \|v'_i\|$ или $\|v'_i\| \not\subseteq \|v_i\|$ для некоторого $i \in [1..n]$. В силу конечности областей определения переменных модели, наибольшее количество "плохих" неоптимизируемых векторов получается в том случае, если семантика попарно различающихся компонент содержит ровно одно число. Например, если в векторах различаются две соответствующих компоненты, то количество различных сочетаний семантик этих компонент не больше D, а количество таких векторов в одном "плохом" множестве равно $D \times D^{n-2} \times C_n^2$. Легко доказать общую формулу наибольшего количества "плохих" векторов для произвольного числа различающихся компонент: $D \times D^{n-i} \times C_n^i$, откуда и следует утверждение. ■

Таким образом, в худшем случае размер векторно-аффинных представлений множеств сравним с размером проверяемой модели. Поскольку BDD-представления множеств целых чисел, как правило, довольно громоздки, как было отмечено во введении, то и BDD-представление для "плохих" множеств, по всей видимости, настолько же велико, однако точное доказательство этого утверждения выходит за рамки данной работы. Несколько лучше такие множества должны кодироваться с помощью формул арифметики Пресбургера, однако алгоритмы работы с такими представлениями экспоненциальны относительно их размера.

Проверка включения немного более сложна, чем процедура оптимизации. Пусть надо проверить, что $P \subseteq Q$ для векторно-аффинных множеств P и Q. Идея алгоритма сравнения состоит в следующем: если оказывается, что семантика i-й компоненты аффинного вектора из P содержится в объединении семантик i-х компонент аффинных векторов из Q (для определения этого снова используется процедура *Solve*), то удаляем соответствующие значения аргумента из области определения аффинного множества i-й компоненты. Если в конце процедуры окажется, что для каждой компоненты каждого вектора аффинное представление пусто, то это векторно-аффинное множество включено в другое. Но этот алгоритм может оказаться трудоемким (в случае большого размера представления множеств), и его следует выполнять после простого сравнения некоторых векторов множества P и множества Q. Алгоритм сравнения векторно-аффинных множеств $In(P, Q)$, основанный на этих соображениях, приводится в приложении В.

Имеет место следующее утверждение о корректности алгоритмов.

Утверждение 2. *Пусть P и Q будут ограниченными векторно-аффинными множествами. Тогда верно следующее:*

- $VOpt(P) = R$, *следовательно,* $|R| \leq |P|$;
- $In(P, Q) =$ *"ДА" тогда и только тогда, когда* $P \subseteq Q$.
- $In(P, Q) =$ *"НЕТ" тогда и только тогда, когда* $P \nsubseteq Q$.

Временная сложность алгоритма оптимизации $VOpt(P)$ равна $O(|P|^2)$. Временная сложность проверки включения $In(P, Q)$ равна $O(|P| \times |Q|)$.

4. Заключение

Предложенное в данной работе модифицированное представление данных в виде векторно-аффинных множеств хорошо подходит для проверки на модели систем, в которых переменные системы принимают целые значения, например, для различных протоколов со счетчиками. Приведены алгоритмы манипуляции с этими представлениями, сложность которых самое большее квадратична относительно размера представления. Также предложен новый алгоритм оптимизации векторно-аффинных множеств, тоже имеющий квадратичную сложность относительно размера векторно-аффинного множества. Получена теоретическая оценка величины "плохих" векторно-аффинных множеств, размер которых нельзя уменьшить с помощью процедуры оптимизации. Количество векторов в таких множествах ненамного меньше размера модели, что сопоставимо с худшими случаями других символьных представлений. Однако на практике модели, в которых могут получаться такие "плохие" представления, должны задаваться довольно сложным способом.

На основе векторно-аффинных представлений реализуется прототип инструмента проверки произвольных аффинных моделей, специфицированными формулами μ-исчисления и менее выразительных логик (LTL, CTL). Этот прототип использует алгоритм проверки моделей из [12]. Планируется провести ряд экспериментов с коммуникационными протоколами, использующими счетчики.

Список литературы

1. **Abdulla P.A., Jonsson B., Nilsson M., Saksena M.** A Survey of Regular Model Checking. Proc. of CONCUR'04 – Concurrency Theory, 15th International Conference, London, UK, August 31 – September 3, 2004 // Lect. Notes Comput. Sci. 2004. Vol. 3170. P. 35–48.

2. **Bartzis C., Bultan T.** Efficient Symbolic Representations for Arithmetic Constraints in Verification // Int. J. Found. Comput. Sci. 2003. Vol. 4. P. 605–624.

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

4. **Boigelot B., Legay A., Wolper P.** Iterating transducers in the large // Proc. 15th Int. Conf. on Computer Aided Verification. 2003. Lecture Notes in Computer Science. Vol. 2725. P. 223–235.

5. **Bryant R.E.** Symbolic boolean manipulation with ordered binary decision diagrams // IEEE Trans. Computers. 1986. Vol. C-35, №8. P. 293–318.

6. **Bultan T., Gerber R., 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, №4. P. 747–789.

7. **Bultan T.** BDD vs. Constraint-Based Model Checking: An Experimental Evaluation for Asynchronous Concurrent Systems. Proc. of Tools and Algorithms for Construction and Analysis of Systems, 6th International Conference, TACAS 2000, Berlin, Germany, March 25 – April 2, 2000 // Lect. Notes Comput. Sci. 2000. Vol. 1785. P. 441–455.

8. **Clarke E.M., Grumberg O., Peled D.** Model Checking. London: MIT Press, 1999. 314 p.

9. **Kozen D.** Results on the Propositional Mu-Calculus // Theor. Comput. Sci. 1983. Vol. 27, №3. P. 333–354.

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

11. **McMillan K.L.** Applying SAT Methods in Unbounded Symbolic Model Checking. Proc. of 14th International Conference, CAV'02, Copenhagen, Denmark, July 27-31, 2002 // Lect. Notes Comput. Sci. 2002. Vol. 2404. P. 250–264.

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

13. **Yavuz-Kahveci T., Bultan T.** Heuristics for Efficient Manipulation of Composite Constraints. Proc. of Frontiers of Combining Systems, 4th International Workshop, FroCoS 2002, Santa Margherita Ligure, Italy, April 8-10, 2002 // Lect. Notes Comput. Sci. 2002. Vol. 2309. P. 57–71.

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

15. **Гаранина Н.О.** Аффинное представление данных для проверки моделей программ. Новосибирск, 2004. 48 с. (Препр./ Сиб. отд-ние. РАН. ИСИ; №116.)

16. **Гаранина Н.О.** Проверка моделей распределенных систем с помощью аффинного представления данных // Моделирование и анализ информационных систем. 2010. Т. 17, №4. С.52-59.

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

A Алгоритм оптимизации

Все алгоритмы и процедуры, необходимые для работы с аффинными множествами, приведены в [14, 16]. Обозначим размер некоторого аффинного множества p как $|p|$. Считаем, что предикат $Incl(p, q)$ истинен в том случае, если семантика аффинного множества p содержится в семантике аффинного множества q. Поскольку p и q — аффинные множества, этот предикат вычисляется аналогично проверке включения аффинных множеств и сложность его вычисления квадратична относительно размера множеств. Обозначим процедуру оптимизации аффинного множества как $AOpt$. Далее процедура $P.remove(v)$ удаляет аффинный вектор v из множества P. Процедура оптимизации векторно-аффинного множества выглядит следующим образом:

$VOpt(P)$**:**

Вход: ограниченное векторно-аффинное множество P.

0. $R := P$;

1. **forall** $v \in R$ **for** $i = 1$ to n

2. $v[i] := AOpt(v[i])$;

3. **forall** $v \in R$

4. $V := v$;

5. **forall** $w \in R, w \neq v$

6. $I := 0$;

7. **for** $i = 1$ to n

8. **if** $Incl(v[i], w[i])$ **then** **continue** i;

9. **if** $I = 0$ **then** $I := i$ **else** **continue** w;

10. **if** $I = 0$ **then** $R.remove(v)$;

11. **else** $v[I] := v[I] \backslash (v[I] \cap w[I])$;

12. $v[I] := AOpt(v[I])$;

13. **if** $|v[I]| > |V[I]|$ **then** $v[I] := V[I]$;

Выход: ограниченное векторно-аффинное множество R.

Корректность этого алгоритма непосредственно следует из определения векторно-аффинных множеств, корректности алгоритма оптимизации $AOpt$ и семантики предиката $Incl$.

B Алгоритм проверки включения

В части **I** алгоритма проводится "быстрое" сравнение множеств. Пусть векторно-аффинное множество упорядочено по возрастанию значений первой компоненты векторов. Функция $Min(X)$ возвращает вектор, в котором компонентами являются минимальные значения компонент младшего вектора множества X, а функция $Max(X)$ возвращает вектор, в котором компонентами являются максимальные значения компонент старшего вектора. Процедура $Dec(X)$ удаляет минимальный и максимальный векторы из множества X. Считаем, что эта проверка содержит $m > 0$ итераций. Пусть P и Q будут сравниваемые множества. В части **II** алгоритма уменьшается область определения компонент какого-либо вектора из P, если его семантика содержится в объединении семантик соответствующих компонент векторов из Q. Если после уменьшения область не пуста, то $P \not\subseteq Q$.

$In(P, Q)$:

Вход: ограниченные векторно-аффинные множества P и Q.

I Предварительная проверка.

1. **for** $i = 1..m$
2. **if** $Min(P) \neq Min(Q)$ **then** $\{ P \not\subseteq Q$; **return**;$\}$
3. **if** $Max(P) \neq Max(Q)$ **then** $\{ P \not\subseteq Q$; **return**;$\}$
4. $Dec(P)$; $Dec(Q)$;
5. $i := i + 1$;

Выход: НЕТ, если $P \not\subseteq Q$, НЕ ЗНАЮ, иначе.

if $\mathbf{I}(P, Q) =$"НЕТ"**then skip** $\mathbf{II}(P, Q)$;

II. Проверка двучленов из P.

Вход: ограниченные векторно-аффинные множества P и Q.

1. **forall** $v \in P$
2. $Aux := \{v\}$;
3. **forall** $v' \in Aux$
4. $V := v'$;
5. **forall** $w \in Q$
6. $Ch := |Aux|$;
7. **for** $i = 1..n$;
8. $at := v'[i] \cap w[i]$;
9. **if** $at \neq \emptyset$ **then** $u := V$; $u[i] := V[i] \setminus at$; $Aux := Aux \cup \{u\}$;
10. **endforall** i
11. **if** $Ch < |Aux|$ **then** $Aux := Aux \setminus \{v'\}$;
12. **endforall** w
13. **endforall** v'
14. **if** $Aux \neq \emptyset$ **then** $P \not\subseteq Q$; **return**;
15. **endforall** v

Выход: НЕТ, если $P \not\subseteq Q$, ДА, иначе.

Корректность этой процедуры следует из определения векторно-аффинных множеств, семантики функций $Min(X)$, $Max(X)$ и семантики процедур $Dec(X)$ и $Solve$.

Optimization Procedures in Affine Model Checking

Garanina N.O.

Keywords: symbolic model checking, algebraic data representation, distributed systems

Symbolic model checking is based on a compact representation of sets of states and transition relations. At present there are three basic approaches of symbolic model checking: BDD-methods, bounded model checking using SAT-solvers, and various algebraic techniques, for example, constraint based model checking and regular model checking. In this paper we suggest improved algorithms for an algebraic data representation, namely, optimization algorithms for affine data structures.

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