УДК 004.021

МИНИМИЗАЦИЯ ОРИЕНТИРОВАННЫХ ГРАФОВ С ОТМЕЧЕННЫМИ ВЕРШИНАМИ

Чепурко В.А. Грунский И.С.
Государственный университет информатики и искусственного интеллекта, г. Донецк

Одной из центральных и актуальных как в теоретическом, так и в прикладном аспектах проблем, возникающих при исследованиях взаимодействия автоматов и операционной среды, является проблема анализа или распознавания свойств этой среды при различной априорной информации и при различных способах взаимодействия автомата и операционной среды.

Операционная среда рассматривается как неориентированный граф с помеченными вершинами [1]. Такие графы возникли первоначально как блок-схемы и схемы программ, а в настоящее время находят применение в задачах навигации роботов.

Проблема эквивалентности программ относится к числу наиболее важных проблем теории программирования. Особое значение проблемы эквивалентности обусловлено тем, что именно к этой проблеме эквивалентных преобразований сводится большинство задач оптимизации, верификации и анализа программ [2]. Поэтому разработка и внедрение эффективных алгоритмов проверки эквивалентности программ оказывает заметное влияние на повышение производительности и качества многих инструментальных средств анализа и преобразования программ. В последние годы наряду с задачами повышения эффективности и надёжности программ не меньшую актуальность приобрела задача своевременного обнаружения постороннего кода, которая включает выявление недекларированных возможностей программных продуктов, а также обнаружение и устранение программ-вирусов.

В настоящей работе рассматривается проблема минимизации

ориентированных графов с отмеченными вершинами. Эта проблема позволяет находить эквивалентные графы, с помощью которых можно представить различные системы. Задача минимизации ориентированных графов с отмеченными вершинами является основой для создания различных методов эквивалентности схем программ. Она является актуальной в теоретическом и прикладом аспектах. Её прикладная актуальность определяется тем, что проблемы самолокализации и контроля для таких графов далеки от разрешения и их исследования находятся в зачаточном состоянии. Теоретическая важность и актуальность определяются тем, что анализ графов проводится методами, аналогичными методам теории автоматов. Эти методы созданы для графовых систем, не являющихся конечными автоматами, но являющимися в некотором смысле автоматоподобными системами.

Пусть G=(S, E, M, μ) конечный, связный, помеченный, ориентированный граф, без петель и кратных дуг [1], где S – конечное множество вершин, E – множество рёбер, M – множество меток, μ: S→M – сюррективная функция размечивания вершин такая, что отметки всех вершин, смежных одной вершине, попарно различны, то есть что граф детерминирован[1]. Конечную последовательность вершин $p=g_1 \ldots g_k$ такую, что $<g_i,g_{i+1}> \in E_G$, $1<i<=k$, назовём путём длины (k-1) в графе G. Слово $\mu(p)= \mu(g_1) \ldots \mu(g_k)$ назовём отметкой пути p. Отметку любого пути, исходящего из вершины g є S, будем называть словом, порожденным вершиной g. Язык Lg определим как множество всех слов, порождённых вершиной g є S [1]. Вершины g, h – назовём эквивалентными, если Lg=Lh, и отличимыми в противном случае. Граф G называется приведенным, если все его вершины попарно отличимы.

Вводится специальный класс детерминированных помеченных орграфов, у которых в множестве преемников каждой вершины нет одинаково помеченных вершин.

Множеством преемников Γ_g^- вершины g называется множество всех вершин являющихся концами дуг исходящих из g.

Помеченный орграф $\mathcal{B}$ назовем детерминированным

орграфом или D-орграфом, если для любой вершины g ∈ G и любых вершин s,t ∈ Γ_g^- из s ≠ t следует, что μ(s) ≠ μ (t). В противном случае $\mathcal{B}$ назовем ND-орграфом.

Введем операцию ⋆: G × M^+ → 2^G соотношением: для любой вершины g ∈ G и любого слова w ∈ M^+ через $g⋆w$ обозначим множество всех вершин h ∈ G таких, что существует путь, соединяющий вершины g и h. и его метка равна w. Если такого пути не существует, то $g⋆w = \varnothing$. Если d(w) = 1 и $w = \mu(g)$, то $g⋆w$ = g. Распространим операцию ⋆ на подмножества множества G естественным образом:

$$H⋆w = \bigcup_{h\in H} (h⋆w). \qquad (1)$$

Покажем, что помеченный орграф $\mathcal{B}$ является D-орграфом тогда и только тогда, когда для любой вершины g ∈ G и любого слова ω ∈ M^+ выполняется |g⋆ω| ≤ 1, где |g⋆ω| = 1, если ω ∈ L_g, и |g⋆ω| = 0 в противном случае. Действительно, для всех слов длины 1 и 2 это утверждение очевидно, а любое слово длины больше 2 единственным образом записывается в виде композиции своих двухбуквенных подслов. С другой стороны, из того, что для любой вершины g ∈ G и любого слова ω ∈ M^+ выполняется |g⋆ω| ≤ 1 следует, что в множестве преемников $\Gamma_g^- = \bigcup_{d(w)=2} (g⋆w)$ вершины g нет двух одинаково помеченных вершин. Таким образом, D-орграф определяется или через локальные свойства его вершин, или через нелокальное свойство множества всех путей, исходящих из вершин.

Введём понятие полностью детерминированного орграфа или PD-орграфа. Граф $\mathcal{B}$ будет PD-графом, когда в окрестностях Γ_g^{-1} и Γ_g^{+1} вершины g нет двух одинаково помеченных вершин. Исходя из этого определения если в множестве преемников Γ_g^{-1} вершины g нет одинаково помеченных вершин, то будем называть граф детерминированным назад. Если в множестве

преемников Γ_g^{+1} вершины g нет одинаково помеченных вершин, то будем называть граф детерминированным вперёд. Если граф $\mathcal{B}$ детерминирован вперёд и назад, то граф $\mathcal{B}$ является PD-графом.

Пусть $L_G=\{L_g\}_{g\in G}$. Графы G и H называются эквивалентными, если $L_G=L_H$. В работе рассматривается задача минимизации графов, то есть задача построения по произвольному графу G эквивалентного ему графа с наименьшим числом вершин. Следуя[1] эта задача сводится к нахождению классов эквивалентных вершин графа G. В [1] показано, что минимальный граф получается из исходного отождествлением его эквивалентных вершин. Известно несколько алгоритмов минимизации – это алгоритм Сапунова[1] написанный, на основе алгоритма Мура для детерминированных графов. Его временная сложность равна $O(m*n^2)$, где n – это количество вершин, а m – количество отметок графа. В [4] описан алгоритм минимизации ациклических ориентированных графов и даны следующие оценки сложности:

- деревья – O(n);
- детерминированные ациклические графы – O(m*n);
- недетерминированные ациклические графы – $O(n^2)$.

В данной работе предложен новый алгоритм минимизации графов основанный на идее Хопкрофта которая заключается в следующем: рассмотрим специальный тип расщепления, называемый разбиением. Задача разбиения множества возникает довольно часто, и решение, которое мы продемонстрируем здесь, поучительно само по себе. Пусть даны множество S и разбиение π множества S на непересекающиеся блоки $\{B_1, B_2,\ldots, B_p\}$. Кроме того, дана функция *f*, отображающая S на S.

Наша задача состоит в том, чтобы найти такое грубейшее (с наименьшим числом блоков) разбиение $\pi' = \{E_1, E_2,\ldots, E_q\}$ множества *S,* что

1. π' — подразбиение разбиения π (т. е. каждое множество E_i является подмножеством некоторого блока B_j),
2. если *a* и *b* принадлежат E_i то *f(a)* и *f(b)* принадлежат E_j для некоторого j.

Будем называть π' грубейшим разбиением множества S, совместимым с π и *f*.

Очевидное решение состоит в повторном утончении блоков исходного разбиения следующим способом. Пусть B_t — какой-нибудь блок. Рассмотрим *f(a)* для каждого *a* из B_t. Разобьем B_i так, чтобы два элемента *a* и *b* попадали в один блок тогда и только тогда, когда *f(a)* и *f(b)* оба принадлежат некоторому блоку B_j. Процесс повторяется до тех пор, пока уже нельзя будет проводить дальнейшие утончения. Этот метод дает алгоритм сложности O (n^2), поскольку каждое утончение занимает время O (n), а всего может быть O (n) утончений.

Недостаток этого метода состоит в том, что утончение блока может потребовать *O* (n) шагов, даже если из него удаляется только один элемент. Сейчас мы опишем алгоритм разбиения, который для разделения блока на два подблока требует время, пропорциональное размеру меньшего подблока. Этот подход приводит к алгоритму сложности *O* (n log n).

Для каждого блока B ⊆ S положим $f^{-1}(B)=\{b|f(b) \in B\}$. Вместо того чтобы разбивать B_t по значениям *f(a)* для $a \wedge B_{it}$ разобьем относительно B_t те блоки B_j, которые содержат хотя бы один элемент из $f^{-1}(B_i)$ и один элемент не из $f^{-1}(B_i)$. Иными словами, каждый такой блок *B* разбивается на множества $\{b|b\in B_j$ и $f(b)\in B_i\}$ и $\{b|b\in B$, и $f(b)\notin B_i\}$.

Как только проведено разбиение относительно блока B_i больше уже не нужно проводить разбиения относительно него, пока он сам не будет расщеплен. Если вначале $f(b)\in B_i$ для каждого элемента $b\in B_j$ и B_i расщеплен на B'_i и B», то можно разбить B_j относительно B'_i или B» и получить при этом один и тот же результат, поскольку $\{b|b\in B_j$, и $f(b)\in B'_i\}$ совпадает с B_j—$\{b|b\in B_j$, и $f(b)\in B»_i\}$.

Так как можно выбирать, по отношению к какому из блоков B'_i или B» проводить разбиение, то мы разбиваем относительно того, для которого это сделать проще. Точнее мы выбираем меньшее из множеств $f^{-1}(B'_i)$ и $f^{-1}(B»_i)$.

Алгоритм Хопкрофта предназначен для графов с отмеченными дугами.

Предложенный в данной работе алгоритм минимизации графов предназначен для графов с отмеченными вершинами. Он состоит в следующем.

Алгоритм

ВХОД: множество вершин S из n элементов и функция f: S→S.

ВЫХОД: разбиение p={B[1],…,B[q]} множества S, на классы эквивалентных состояний.

ОБЪЕКТЫ: граф заданный списком вершин, и другие объекты описанные ниже:

v_i.class - класс принадлежности;

v_i.mark - отметка вершины;

B[j] – j-й класс, содержит номера эквивалентных вершин;

B[j].prev – количество предшествующих вершин j-го класса;

BT[k] –временный класс содержащий эквивалентные вершины;

$f^{+1}(v_i)$ – множество последующих вершин текущей;

$f^{-1}(v_i)$ – множество предшествующих вершин текущей;

WAIT – множество номеров классов которые будут обрабатываться;

q – количество классов;

Инициализация

Просмотрим все вершины и каждой присвоим номер класса равный следующей формуле v.class = v.mark*| $f^{+1}(v_i)$|. Номер вершины поместим в класс с индексом равным номеру метки умноженному на количество последующих вершин. Получаем К классов и переименовываем индексы классов по порядку от 1 до К.

Просмотрим все дуги графа и создадим списки предшествующих вершин.

В множество WAIT поместим все начальные классы.

Метод

1. Пока WAIT не пусто выберем и удалим i класс находящийся последним в списке.
2. Для всех вершин класса B[i] находим списки предшествующих вершин и разбиваем их в соответствии с классом которому принадлежат.
3. Для каждого полученного из пункта 2 класса BT[k] выполняем:
   3.1 Если | BT[k]|<>| B[k]| то разбиваем класс B[k]
   3.2 q=q+1
   3.3 B[q]= BT[k]
   3.4 B[k]=B[k]-B[q]
   3.5 Если k не содержится в WAIT то помещаем в WAIT номер того класса мощность которого по предшествующим вершинам меньше.

Доказано, что результатом работы алгоритма являются классы эквивалентных вершин, то есть он решает задачу минимизации. Определена временная сложность алгоритма.

При подсчёте временной сложности алгоритма получилось два варианта оценки. Варианты отличаются между собой минимальным шагов выполнения алгоритма.

В первом случаем за единицу времени будем считать операции:

- операция добавления элемента в множество;
- операция удаления элемента из множества;
- сравнение двух чисел;
- другие единичные операции.

В таком случает асимптотическая временная сложность равна O(m*n).

Если в качестве минимальной единицы времени брать простейшие операции сравнения, записи, то асимптотическая временная сложность равна O(m*n*log(n)).

Сложность алгоритма возрастает за счёт операций работы со множествами, которые требеют O(log(n)) шагов.

Литература

[1] Сапунов С.В. Анализ графов с помеченными вершинами: Дисс. канд. физ.-мат. наук: 27.09.07. – Донецк.: 2007. – 150 с.

[2] R. Gentilini, C. Piazza and A. Policriti From Bisimulation to Simulation. Coarest partition problems // Dip. Di Matematica e Informatica – Universita di Udine Via Le Scienze, 206 – 33100 Udine – Italy.

[3] А.Ахо, Дж. Хопкрофт, Дж.Ульман. Построение и анализ вычислительных алгоритмов. –М.: «МИР». 1979. – 235 с.

[4] Чепурко В.А. Минимизация ориентированных ациклических графов с отмеченными вершинами // Материалы VI международной научно-практической конференции «Математическое и программное обеспечение интеллектуальных систем». Днепропетровск, 12-14 ноября 2008 г. Днепропетровск.: 2008. – С. 331–332.