Использование информации о линейных зависимостях для обнаружения уязвимостей в исходном коде программ

В.С. Несов, О.Р. Маликов

1. Введение

Одним из методов обнаружения потенциальных уязвимостей в программе является статический потоково-чувствительный data-flow анализ. В ходе проведения такого анализа для каждой точки программы собирается информация о различных атрибутах объектов программы, которая затем проверяется на выполнение условий корректности операций с памятью и использования библиотечных функций в различных точках программы.

Серьезной проблемой такого метода обнаружения уязвимостей является большое количество ложных предупреждений. Часто ложные предупреждения вызваны недостаточной точностью определяемой информации об атрибуте - значение некоторой переменной (объекта программы).

Самым простым подходом определения информации о целочисленных значениях является анализ на основе интервальных оценок. Каждому целочисленному атрибуту объектов программы в данной точке программы сопоставляется числовой интервал значений, при этом зависимости между атрибутами не учитываются. Например, если атрибуту x сопоставляется интервал возможных значений [a,b], атрибуту y – интервал [c,d], то результату операции сумма z=x+y сопоставляется интервал [a+c,b+d]. Такая модель хорошо работает для не связанных друг с другом атрибутов.

2. Анализ на основе линейных зависимостей

Многие условия корректности операций в программе представляют собой линейные соотношения (равенства либо неравенства) между значениями числовых атрибутов объектов программы. Например, при обращении к массиву проверяется, лежит ли индекс, по которому происходит обращение, в пределах массива. В случае если и длина массива, и значение индекса в широких пределах произвольны, ответить на вопрос о возможности выхода за

пределы массива при отсутствии информации о зависимости между индексом и длиной массива невозможно.

Подход, предлагаемый нами к использованию в рамках разрабатываемой системы обнаружения уязвимостей, состоит в поддержании системы линейных неравенств, выполняющихся для числовых атрибутов в данной точке программы.

При потоково-чувствительном анализе потока данных в каждой точке программы (контексте) хранится информация о большом количестве числовых атрибутов. Так как одной из задач разрабатываемой системы являлась возможность анализа программ промышленного масштаба, были применены различные методы, ограничивающие вычислительные издержки, возникающие при учете линейных зависимостей.

Если при анализе требуется определить соотношение между значениями атрибутов (проверить некоторое равенство или неравенство), то переносом всех атрибутов в одну часть соотношения задача сводится к определению множества возможных значений атрибута, равного полученному в этой части выражению. Например, при проверке выхода за пределы массива требуется проверить, меньше ли индекс в массиве x размера массива l, $x<l$, что эквивалентно $z=x-l$, $z\in(-\infty,-1]$.

Для каждого атрибута одновременно поддерживаются целочисленный интервал значений (интервальная оценка) и системы линейных уравнений и неравенств (система линейных связей). Преобразования для интервальных оценок производятся независимо от преобразований системы линейных связей. Для проверки условий корректности операций программы достаточно получать интервальную оценку значений линейных комбинаций атрибутов. Поэтому линейные связи можно рассматривать как дополнительную информацию, надстройку над анализом на основе интервальных оценок, позволяющую иногда уточнять интервальную оценку.

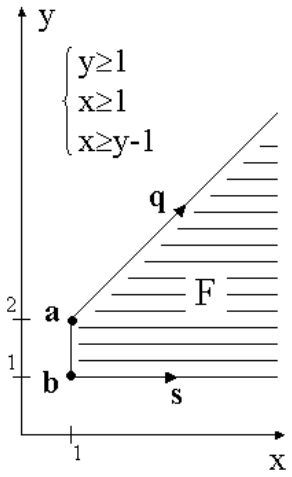
3. Многогранное множество

Многогранным называется множество, задаваемое системой линейных неравенств.

Каждое многогранное множество может быть представлено двумя способами: в виде системы неравенств, либо в виде множества образующих вершин и лучей (геометрическое представление). В геометрическом представлении многогранное множество равно сумме выпуклого замыкания множества вершин и конического замыкания множества лучей. На рисунке {a,b} – множество вершин, {q,s} – множество лучей.

Наличие обоих представлений и возможность перехода между ними позволяет производить различные преобразования многогранных множеств при помощи простых алгоритмов. Например, пересечение многогранных множеств задается объединением систем неравенств. Выпуклая оболочка объединения

многогранных множеств задается объединением соответствующих множеств геометрического представления.



Для перехода между представлениями многогранного множества используется алгоритм Черниковой с оптимизацией Ле Вержа (H. Le Verge) [1].

Сложность различных операций над многогранными множествами не позволяет применять их к многогранному множеству, описываемому всей системой линейных связей контекста. Поэтому для применения операций, требующих представления в виде многогранного множества, необходимо выделять подсистему линейных связей.

4. Граф линейных связей

Введем для дальнейшего рассмотрения ориентированный граф линейных связей (ГЛС). Вершинам этого графа сопоставлены атрибуты, ребро (u,v) присутствует в графе, когда в систему линейных связей атрибута u входит атрибут v.

Структура ГЛС существенно зависит от метода выделения системы линейных связей из многогранного множества, полученного в качестве результата преобразования. Этот вопрос будет раскрыт далее. За счет структуры ГЛС для выделения приемлемого многогранного множества, описывающего данный атрибут, достаточно собрать систему из линейных связей вершин ГЛС, встречающихся при обходе ГЛС в ширину, начиная с описываемого атрибута. Добавление линейных связей прекращается, как только одна из оценок сложности получаемого многогранного множества превышает соответствующее пороговое значение.

Для многогранного множества вводятся две оценки сложности: количество атрибутов и количество существенных ограничений неравенствами.

Предельное количество атрибутов влияет на степень вершин ГЛС. Чрезмерное повышение этого порога приводит к тому, что при обходе в ширину в ГЛС быстро собирается большое количество посторонних линейных связей (поднимая вторую оценку сложности), до того как в строящееся многогранное множество включается достаточное количество связей, важных для описываемого атрибута. Эксперименты на тестовом наборе программ показали, что оптимальным значением является порог порядка 15 атрибутов.

Количество существенных ограничений неравенствами определяет вычислительную сложность операций над многогранным множеством. Для тестового набора программ порог, приводящий к удвоению времени анализа программ при учете линейных связей по сравнению с анализом без учета линейных связей, составляет порядка 20 ограничений.

Будем далее называть операцию выделения многогранного множества из ГЛС для данного атрибута замыканием атрибута.

4.1. Выделение системы линейных связей из многогранного множества

После выполнения преобразования многогранного множества необходимо сохранить результат в атрибуте. Многогранное множество часто получается в результате замыкания атрибута и содержит слишком большое количество линейных связей. Так как при всех операциях каждый раз модифицируется только небольшое количество атрибутов, то, вообще говоря, изменяются только линейные связи, содержащие эти атрибуты. Поэтому из многогранного множества извлекаются только линейные связи, включающие в себя данные атрибуты.

Так как системы линейных связей хранятся отдельно для различных атрибутов, часть полученных линейных связей может оказаться следствием связей, получаемых из окружающих вершин ГЛС. Сохранение таких связей излишне, так как при замыкании линейные связи окружающих вершин и так будут учтены. Поэтому такие связи исключаются.

5. Пример

В таблице приведен код примера и значения атрибутов при проведении итераций статического анализа. Для каждой строки кода для данной итерации приведены ограничения, справедливые на выходе из инструкций строки, либо, в случае ветвления, ограничения на входе и выходе из инструкции. Например, в приведенном примере контекст на входе в инструкции строки 4 получается в результате объединения контекстов на выходе из инструкций строк 3 и 6.

В примере имя атрибута s.len ссылается на атрибут длина строки s. Выписываются не все ограничения, а лишь важные для окончательных выводов.

Как видно, в результате анализа данного примера определяются оценки значений атрибутов, позволяющие исключить ошибки переполнения массивов. При анализе на основе интервальных оценок такие ошибки исключены не были бы.

код	ограничения			
	итерация 1	итерация 2	итерация 3 (расширение)	итерация 4
char s1[1024], s2[1024];				
unsigned int i=0;	i=0	– ‖ –	– ‖ –	– ‖ –
fgets(s1,1024,stdin);	s1.len∈[0,1023]	– ‖ –	– ‖ –	– ‖ –
while(i<strlen(s1)){	i=0	i∈[0,1]	i∈[0,+∞)	i∈[0,1023], i<=s1.len
	i=0, i<s1.len	i∈[0,1], i<s1.len	i∈[0,1022], i<s1.len	i∈[0,1022], i<s1.len
s2[strlen(s1)-i-1]=s1[i];	(s1.len-i-1) ∈[0,1022]	– ‖ –	– ‖ –	– ‖ –
i++;}	i=1	i∈[1,2]	i∈[1,1023], i>=s1.len	– ‖ –
s2[i]=0;	i=0, i>=s1.len	i∈[0,1], i>=s1.len	i∈[0,+∞), i>=s1.len	i∈[0,1023], i=s1.len
	s2.len=0	s2.len∈[0,1], s2.len<=i	s2.len∈[0,+∞), s2.len<=i	s2.len∈ [0,1023], s2.len<=s1.len

5.1. Примеры устраненных ложных предупреждений

```
bftpd, mystring.c, строка 16:
  void cutto(char *str, int len)
  {
  memmove(str, str + len, strlen(str) - len + 1);
     }
```

При корректных вызовах функции переполнения буфера не возникает. Требуемое условие в данном случае str.size>str.len-len, где str.size и str.len – соответственно атрибуты массива str, отражающие его длину и длину хранящейся в нем строки.

```
popclient, socket.c, строка 112:
    int SockWrite(socket,buf,len)
    int socket;
    char *buf;
    int len;
    {
        int n;
        while (len)
        {
            n = write(socket, buf, len);
            if (n <= 0)
                return -1;
            len -= n;
            buf += n;
        }
        return 0;
    }
```


В данном случае при вызове функции всегда len<=buf.size, при изменении len и buf на одну и ту же величину соотношение не нарушается. Так как возвращаемое значение функции write n<=len, так же сохраняется условие len>=0 (уточняемое условием цикла до len>0). За счет выполнения этих соотношений вызов функции write происходит корректно.

6. Результаты

На тестовом наборе из 7 программ с открытым исходным кодом были получены следующие результаты. В таблице 1 указано количество истинных и ложных предупреждений, выдающихся системой обнаружения уязвимостей до и после использования учета линейных зависимостей между атрибутами.

Название программы	Общее количество предупреждений	Количество истинных предупреждений	Количество ложных предупреждений при анализе на основе интервалов	Количество ложных предупреждений, устраненных при использовании линейных зависимостей
bftpd	54	20	34	7
lhttpd	22	5	17	0
muh	47	12	35	2
pgp4pine	45	16	29	3
popclient	34	7	27	12
sharutils	49	11	38	7
troll-ftpd	47	2	45	2

Таблица 1. Результаты тестирования

Литература

1. H. Le Verge. A note on Chernikova's Algorithm. July 27, 1994.
2. P. Cousot and N. Halbwachs. Automatic discovery of linear restraints among variables of a program. *In 5th ACM Symposium on Principles of Programming Languages, POPL'78*, Tucon (Arizona), January 1978.
3. N. Halbwacht, Y.E. Proy, and P. Roumanoff. Verification of real-time systems using linear relation analysis. *Formal Methods in System Design*, 11(2):157-185, 1997.
4. Бирюков С.И. Оптимизация. Элементы теории. Численные методы. МЗ-Пресс, 2003.