О распознавании вирусов в регулярных текстах

М.В. Болонкин

Исследуются свойства регулярных выражений и их тождественных преобразований для распознавания полиморфных вирусов. Определена модель зараженности, сильной и слабой зараженности программы вирусом, исследованы некоторые классы выражений. Доказаны критерии зараженности программы вирусом как для множества программ, не содержащих оператор Клини, так и подмножества программ, содержащих его, но глубины вложенности не более 1.

Ключевые слова: распознавание, вирусы, регулярное выражение, подформулы.

Введение

Компьютерный вирус — разновидность компьютерных программ, отличительной особенностью которой является способность к размножению (саморепликация). В дополнение к этому вирусы могут повредить или полностью уничтожить все файлы и данные, подконтрольные пользователю, от имени которого была запущена зараженная программа, а также операционную систему со всеми файлами в целом. Известны десятки тысяч компьютерных вирусов, которые распространяются через Интернет по всему миру, организуя вирусные эпидемии. Основными каналами распространения вирусов являются флэш-накопители, сеть Интернет и локальные сети, электронная почта, системы обмена мгновенными сообщениями, веб-страницы. В связи со все возрастающей популярностью и глобальной информатизацией проблема вирусов становится особенно актуальна. Экономика

развитых стран испытывает огромный урон от распространения вирусов. В 2007 году мировые экономические потери от компьютерных вредоносных программ превысили 13 миллиардов долларов.

В настоящее время существует широкий выбор антивирусных программ, способных распознать и удалить известные вирусы. Основные методы, применяемые в этих программах для обнаружения зараженных программ — обнаружение, основанное на сигнатурах, обнаружение аномалий, метод эмуляции, метод эвристического анализа. При этом, самым распространенным является метод сигнатур, заключающийся в сравнении кода программ с образцами из базы сигнатур вирусов. Под сигнатурой в данном случае понимаются характерные признаки вируса, чаще всего взятые непосредственно из файла вируса, и обработанные для удобства хранения соотвествующим образом (например, выделением регулярного выражения). При обнаружении вхождения сигнатуры в какой-либо участок кода программы, она считается зараженной и подвергается удалению или восстановлению («лечению»). Однако с эволюцией компьютерных вредоносных программ, появились вирусы, способные защищаться от обнаружения. Эта разновидность вирусов изменяет код своей копии в процессе репликации. При этом известны три характерных приема, используемые в полиморфных вирусах: шифрование своего кода, обфускация и собственно полиморфизм, заключающийся в эквивалентном преобразовании кода. Изменение кода таких вирусов затрудняет их обнаружение при помощи сигнатурного поиска. Проблему представляет и анализ программ, которые могут быть инфицированы вирусом, то есть когда код вируса внедрен внутрь кода другой, безобидной программы.

Несмотря на очевидную актуальность исследований в области компьютерной вирусологии, теоретические модели компьютерных вирусов и защиты от них малоразвиты. В работе Коэна [3] приведено доказательство того, что в общем виде, вопрос о том, является ли заданная программа вирусом, неразрешим. Другие исследователи доказали, что существуют такие типы вирусов (вирусы, содержащие копию программы детектирующей вирусы), которые не могут быть безошибочно определены ни одним алгоритмом. В случае сигнатурного распознавания полиморфных вирусов, очевидно, что за-

дача обнаружения программы-вируса равносильна задаче об эквивалентности языков, задаваемых грамматиками. Известно, что уже для контекстно-свободных языков в общем случае эта задача алгоритмически неразрешима [4].

В данной работе предлагается модель вирусов в терминах регулярных выражений. Проблема обнаружения программы-вируса в данном случае сводится к задаче проверки равенства двух регулярных множеств. Хорошо известно, что эта задача алгоритмически разрешима [9, 10]. Больший интерес представляет задача распознавания зараженности, когда код вируса включен в код программы. В работе вводится определение понятия зараженности для языка регулярных выражений, а так же понятия слабой и сильной зараженности. Основным результатом работы является доказательство алгоритмической разрешимости проблемы распознавания зараженности программы вирусом для некоторых классов программ.

1. Определения и обозначения

Пусть $\Sigma = \{a_1, a_2, \ldots, a_n\}$ — конечное непустое множество, называемое (входным) алфавитом. Пусть Σ^* — свободная полугруппа, порожденная множеством Σ, в которой операция записывается мультипликативно и обозначается записью подряд. Элементы из Σ^* назовем словами в алфавите Σ.

Понятие регулярного множества в алфавите Σ определяется рекурсивно следующим образом [6, 9]:

1) $\varnothing$ (пустое множество) — регулярное множество в алфавите Σ;
2) $\{\lambda\}$ — регулярное множество в алфавите Σ;
3) $\{a\}$ — регулярное множество в алфавите Σ для каждого $a \in \Sigma$;
4) Если A, B — регулярные множества в алфавите Σ, то таковы же и множества
   (a) $A \cup B$,
   (b) $AB = \{ab \mid a \in A, b \in B\}$,
   (c) $A* = \{a_1 a_2 \ldots a_k \mid a_i \in A,\ 1 \leqslant i \leqslant k,\ k \geqslant 1\}$;
5) Ничто другое не является регулярным множеством в алфавите Σ.

Регулярные выражения над алфавитом Σ и регулярные множества, которые они обозначают, определяются рекурсивно следующим образом [6]:

1) ϕ — регулярное выражение, обозначающее регулярное множество $\varnothing$;
2) λ — регулярное выражение, обозначающее регулярное множество $\{\lambda\}$;
3) Если $a \in \Sigma$, то a — регулярное выражение, обозначающее регулярное множество $\{a\}$;
4) Если α, β — регулярные выражения, обозначающие регулярные множества A, B соотвественно, то
   (a) $(\alpha + \beta)$ — регулярное выражение, обозначающее $A \cup B$,
   (b) $(\alpha\beta)$ — регулярное выражение, обозначающее AB,
   (c) $(\alpha)*$ — регулярное выражение, обозначающее $A*$;
5) Ничто другое не является регулярным выражением.

Регулярное выражение $(\alpha)^n$ обозначает множество $\{a_1 a_2 \dots a_n | a_i \in A\}$.

Будем использовать обозначение $\alpha \equiv \beta$ для графически совпадающих регулярных выражений α и β, то есть выражений, содержащих одинаковые символы в одинаковом порядке. Мы говорим, что два регулярных выражения равны $(\alpha = \beta)$, если равны множества, которые они обозначают.

Согласно [10] введем систему аксиом (A1–A7) и правило вывода (R1) в множестве всех регулярных выражений.

Пусть $\alpha, \beta, \gamma, \delta$ — произвольные регулярные выражения.

A1: $\alpha \vee (\beta \vee \gamma) = (\alpha \vee \beta) \vee \gamma$;

A2: $\alpha(\beta\gamma) = (\alpha\beta)\gamma$;

A3: $\alpha \vee \beta = \beta \vee \alpha$;

A4: $\alpha(\beta \vee \gamma) = \alpha\beta \vee \alpha\gamma$;

A5: $(\alpha \vee \beta)\gamma = \alpha\gamma \vee \beta\gamma$;

A6: $\alpha \vee \alpha = \alpha$;

A7: $\alpha* = \alpha\alpha* \vee \alpha$.

Правило R1 (замена). Допустим, что γ' есть результат замены вхождения α на β в γ. Тогда из равенств $\alpha = \beta$ и $\gamma = \delta$ можно вывести равенства $\gamma' = \delta$ и $\gamma' = \gamma$.

Пусть α — регулярное выражение. Определим следующую процедуру исключения лишних (незначащих) скобок.

1) Если в α встречается подстрока $a(bc)$ (за скобкой не следует оператор $*$) или $(ab)c$, где a, b, c — символы алфавита, заменяем ее на abc. Если в α встречается подстрока $a \vee (b \vee c)$ (за скобкой не следует оператор $*$) или $(a \vee b) \vee c$, где a, b, c — символы алфавита, заменяем ее на $a \vee b \vee c$.
2) Если в α встречается подстрока $\beta(\gamma\delta)$ (за скобкой не следует оператор $*$) или $(\beta\gamma)\delta$, где в выражениях β, γ, δ исключены лишние скобки, заменяем ее на $\beta\gamma\delta$. Если в α встречается подстрока $\beta \vee (\gamma \vee \delta)$ (за скобкой не следует оператор $*$) или $(\beta \vee \gamma) \vee \delta$, где в выражениях β, γ, δ исключены лишние скобки, заменяем ее на $\beta \vee \gamma \vee \delta$.
3) Внешние скобки исключаем.

Полученное с помощью этой процедуры регулярное выражение будем обозначать через $\bar{\alpha}$.

Определим множество $SF(\alpha)$ всевозможных подформул регулярного выражения α.

1) $\alpha \in SF(\alpha)$
2) Если α можно представить в виде $\beta\gamma$ или $\beta*$, где β, γ — некоторые регулярные выражения, то β, γ входит в $SF(\alpha)$, а также $SF(\beta), SF(\gamma)$ входят в $SF(\alpha)$.
3) Если α можно представить в виде $\beta \vee \gamma$, где β, γ — некоторые регулярные выражения, то β, γ входят в $SF(\alpha)$, $SF(\beta), SF(\gamma)$ входят в $SF(\alpha)$ и кроме того $\gamma \vee \beta$ также входит в $SF(\alpha)$.

Будем говорить, что регулярное выражение β может входить как подформула в выражение α и обозначать это $\beta \rhd \alpha$ если $\bar{\beta} \in SF(\bar{\alpha})$.

Будем считать, что множество регулярных выражений над алфавитом Σ является формальным языком, элементы которого будем называть выражениями или программами. Пусть B и C — некоторые

такие программы. Объявляем C вирусом. Введем основные для нас определения.

Определение 1. Будем говорить, что программа B **заражена вирусом** C если существует выражение $E = B$ и существует выражение $E' = C$ такие, что $E' \rhd E$. Другими словами, существует регулярное выражение, равное исходной программе B, в которое может входить подформула, равная вирусу C.

Определение 2. Будем говорить, что программа B **заражена вирусом** C **слабо**, если существует $E = B$ такое, что выполняется одно из двух условий:

2.1 Для некоторого выражения $E' = C$ E' может входить в качестве подформулы в E, но не существует таких выражений $E'' = C$, что $E'' \rhd B$.

2.2 Для некоторого выражения $E' = C$ E' может входить в качестве подформулы в B, но не существует таких выражений $E'' = C$, что $E'' \rhd E$.

Определение 2′. Программа B **заражена вирусом** C **слабо**, если существуют такие E_1 и E_2, равные B, что для некоторого выражения $E'_1 = C$ E'_1 может входить в качестве подформулы в E_1 и не существует такого $E'_2 = C$, что $E'_2 \rhd E_2$.

Отметим равносильность определений 2 и 2′. Действительно, если выполняется условие (2.1), берем $E_1 \equiv E$ и $E_2 \equiv B$. Если же выполняется (2.2), то выбираем $E_1 \equiv B$ и $E_2 \equiv E$. Пусть теперь выполняется условие определения 2′. Тогда если в B входит подформула, равная C, выбираем $E \equiv E_2$, иначе выбираем $E \equiv E_1$. Тогда будет выполняться одно из условий (2.1) или (2.2).

Определение 3. Программа B **сильно заражена** вирусом C если для всех $E = B$ существует такое выражение $E' = C$, что $E' \rhd E$.

Под степенью зараженности программы будем понимать «слабую зараженность» или «сильную зараженность». Очевидно, что если программа заражена вирусом, но не заражена им сильно, то она заражена им слабо.

2. Лемма об инвариантности

Докажем вспомогательное утверждение.

Лемма 1 (об инвариантности). *Если B, B_1, C, C_1 — программы и $B_1 = B$ и $C_1 = C$, то при зараженности B вирусом C, программа B_1 заражена вирусом C_1 с той же степенью зараженности.*

Доказательство. Пусть B заражено вирусом C. Тогда существуют выражения $E = B$ и $E' = C$ такие, что $E' \rhd E$. Но в силу равенств $B_1 = B$ и $C_1 = C$ имеем $E = B_1$ и $E' = C_1$. Таким образом, существуют выражения $E = B_1$ и $E' = C_1$ такие, что $E' \rhd E$, то есть B_1 заражена вирусом C_1.

Если B заражено слабо вирусом C. Тогда существуют такие $E_1 = B$ и $E_2 = B$ такие, что для некоторого выражения $E'_1 = C$, $E'_1 \rhd E_1$ и не существует выражения $E'_2 = C$ такого, что $E'_2 \not\rhd E_2$. Но в силу равенств $B_1 = B$ и $C_1 = C$ имеем, что существуют программы $E_1 = B_1$ и $E_2 = B_1$, такие, что для некоторого выражения $E'_1 = C_1$, E'_1 может входить как подформула в E_1 и ни для каких $E'_2 = C_1$ E'_2 не входят как подформулы в E_2, то есть программа B_1 заражена вирусом C_1 слабо.

Пусть B сильно заражено вирусом C. Тогда для любых программ $E = B$ есть программа $E' = C$ такая, что $E' \rhd E$. Но тогда и для всех программ $E = B_1$ существует программа $E' = C_1$ такая, что $E' \rhd E$ в силу равенств из условия леммы. А следовательно, программа B_1 сильно заражена вирусом C_1. Лемма доказана.

Следовательно, если B_1 заражено вирусом C_1 (сильно или слабо), то B заражено вирусом C (соотвественно сильно или слабо). То есть свойство зараженности вирусом инвариантно относительно тождественных преобразований выражения или вируса. Таким образом, при анализе зараженности можно привести вирус и программу к любому удобному виду с помошью тождественных преобразований на основе аксиом A1–A7 и правила вывода R1. Поэтому в дальнейшем будем считать, что в заданном вирусе и программах отсуствуют подформулы вида $(A \vee A)$. Будем называть регулярное выражение без подформул указанного вида **нормальной формой** выражения.

3. Регулярные выражения без итерации

Будем рассматривать выражения без оператора итерации $(*)$.

Будем называть простой конкатенацией выражение вида $A = a_{i_1}a_{i_2}\dots a_{i_j}$, ее длиной будем называть количество символов и обозначать $|A|$. Преобразуем программу B к виду

$$\mathfrak{B} = \bigvee_{j=1}^{k} B_j, \tag{1}$$

где B_j — простые конкатенации. Такая запись — это перечисление всех слов, входящих в регулярное множество, описываемое данным регулярным выражением. В силу этого, такое представление будет одинаково также для всех $E = B$. Назовем такую форму записи **канонической формой** регулярного выражения. Привести регулярное выражение к канонической форме можно с помошью применения аксиом дистрибутивности (А4–А5).

Случай простого вируса

Пусть вирус представляет собой простую конкатенацию: $C = a_{i_1}a_{i_2}\dots a_{i_n}$.

Очевидно, что если вирус C состоит из единственного символа $C = a_{i_1}$, то сильная зараженность этим вирусом эквивалентна вхождению выражения a_{i_1} в программу B. Поэтому будем рассматривать вирусы длины $n \geqslant 2$.

Теорема 1. *Программа B заражена вирусом $C = a_{i_1}a_{i_2}\dots a_{i_n}$ тогда и только тогда, когда C входит в качестве подформулы в каноническую запись $\mathfrak{B}$.*

Доказательство. *Необходимость.* Пусть B заражена вирусом C. Тогда существует $E = B$, такое что в E входит подформула, равная C. Если $E \equiv \mathfrak{B}$, то утверждение теоремы очевидно. Пусть E — не каноническая запись программы B.

Пусть $E' \rhd E$ и $E' = C$. После преобразования E' с помошью аксиом дистрибутивности А4–А5 приводим его к виду $\mathfrak{E}' \equiv \bigvee_{j=1}^{k} C$, где $k \geqslant 1$.

Делая замену E' на $\mathfrak{E}'$, получаем выражение E_1. По аксиоме A6 и R1 имеем $E_1 = E$. Таким образом, существует $E_1 = B$, в которое входит подформула, тождественная C. В аксиомах дистрибутивности A4 и A5 в левой части простая конкатенация может входить в качестве подформулы только в выражения α, β, γ, поэтому если простая конкатенация входит в одно из подвыражений левой части, то она будет входить и в правую часть. Так как при переходе от E_1 к канонической форме используются только эти аксиомы, в каноническую форму будут входить подформула C.

Достаточность. Пусть в каноническую форму $\mathfrak{B} \equiv \bigvee_{j=1}^{k} B_j$ входит в качестве подформулы C. Но тогда существует $E \equiv \mathfrak{B} = B$ такое, что в него входит подформула, равная (в данном случае тождественно равная) вирусу C. Следовательно B заражено вирусом C. Утверждение доказано.

Теорема 2. *Если B и C — программы, $C = a_{i_1} a_{i_2} \dots a_{i_n}$, $\mathfrak{B} = \bigvee_{j=1}^{m} B_j$ — каноническая запись программы B и выполняются утверждения*

1) Для некоторого k : $B_k \equiv C$;

2) Все $B_j (j \neq k)$ не имеют общих суффиксов и префиксов с C,

то программа B заражена вирусом C сильно.

Доказательство. Так как $B_k \equiv C$, то в силу предыдущего утверждения B заражено вирусом C. Докажем, что B заражено сильно.

При любых тождественных преобразованиях подформулы B_k, она останется равной вирусу C.

Любое $E = B$ можно получить с помощью тождественных преобразований на основе аксиом A1–A6 и правила вывода R1.

A1–A3, A6 никак не влияют на вхождение подформулы B_k в каноническую запись.

A4–A5 позволяют «вынести за скобку» общий суффикс или префикс двух подформул. Так как у B_k и $B_j, (j \neq k)$ нет общих суффиксов и префиксов, мы не можем применить эти аксиомы к B_k, то есть B_k остается неизменным. Следовательно, при любом тождественном преобразовании имеем $E = B'_k \vee A$, где $B'_k = B_k$, A — некоторое регулярное выражение $A = \bigvee_{j \neq k} B_j$.

Получаем, что $\forall E = B\ \exists E' = C$, $E' \rhd E$ (в нашем случае $E' \equiv B'_k$), то есть B сильно заражено вирусом C. Утверждение доказано.

Лемма 2. *Если A — простая конкатенация, представимая в виде $A \equiv \alpha_1\beta\alpha_2$, где $\alpha_1, \beta, \alpha_2$ — простые конкатенации, $\alpha_1\alpha_2 \neq \lambda$, $\beta \not\rhd \alpha_1$, $\beta \not\rhd \alpha_2$ и $|\beta| > 2$, то существует такое выражение $A_1 = A$, что для любых $\beta_1 = \beta$ выражение β_1 не может входить как подформула в A_1.*

Доказательство. Пусть $\beta = b_1b_2\dots b_k$ ($k > 2$, $b_i \in \Sigma$). Предполагаем, что не все b_i одинаковы.

В силу условия $\alpha_1\alpha_2 \neq \lambda$, хотя бы одно из α_1, α_2 не является пустым словом. Допустим $\alpha_1 \neq \lambda$. Произведем замену

$$A = \alpha_1b_1b_2\dots b_k\alpha_2 = (\alpha_1b_1 \vee \alpha_1b_1)b_2\dots b_k\alpha_2 \tag{2}$$

I случай. $\beta \rhd \alpha_1b_1$

Такое возможно, если $b_1 = b_k$. В таком случае $\alpha_1b_1 \equiv \alpha'_1b_1b_2\dots b_k$ для некоторого α'_1. При этом $\beta \not\rhd \alpha'_1$, иначе β входило бы в α_1.

a) $\alpha'_1 \neq \lambda$

Заменим вхождение этой подформулы на $(\alpha'_1b_1 \vee \alpha'_1b_1)$. В этом случае имеем

$$\alpha_1\beta = (\alpha'_1b_1 \vee \alpha'_1b_1)b_2\dots b_k\alpha_2.$$

В этом выражении $\beta \not\rhd \alpha'_1b_1$ и $\beta \neq (\alpha'_1b_1 \vee \alpha'_1b_1)b_2\dots b_j$, $\forall j < k$ (иначе β входило бы в α_1).

b) $\alpha'_1 = \lambda$

$$\alpha_1b_1 \equiv b_1b_2\dots b_k.$$

Тогда вместо (2) сделаем замену

$$\alpha_1b_1b_2\dots b_k\alpha_2 = (\alpha_1b_1b_2 \vee \alpha_1b_1b_2)b_3\dots b_k\alpha_2.$$

Такое возможно в силу условия $k > 2$. При этом имеем $\alpha_1b_1b_2 \equiv b_1b_2\dots b_kb_2$. Заменяем это вхождение на подформулу $b_1b_2\dots b_{k-2}(b_{k-1}b_kb_2 \vee b_{k-1}b_kb_2)$. В это выражение β входить не может (по предположению, не все b_i одинаковы).

II случай. $(\alpha_1 b_1 \vee \alpha_1 b_1) b_2 \dots b_j = \beta, \qquad j < k$

В этом случае делаем замену $b_j b_{j+1} = (b_j b_{j+1} \vee b_j b_{j+1})$.

III случай. $\beta \rhd b_2 \dots b_k \alpha_2$

Такое возможно если $b_1 = b_2 = \dots = b_k$, $\alpha = b_1 \alpha_2'$.

а) $\alpha_2' \neq \lambda$

Делаем замену $\alpha_2 = (\alpha_2 \vee \alpha_2)$.

а) $\alpha_2' = \lambda$

Тогда $A = \alpha_1 \underbrace{b_1 b_1 \dots b_1}_{k+1}$.

Если $\alpha_1 \neq (b_1)^l$, то считаем, что $A = \alpha_1'' \beta$ и см. I случай. Пусть $\alpha_1 = (b_1)^l$ $(l < k)$. В этом случае $A = (b_1)^{k+l+1}$, $k+l+1 \leqslant 2k$. Пусть $k+l-1 = (k-1)p+r$. Тогда представим A в виде

$$A = (\underbrace{b_1 b_1 \dots b_1}_{k-1} \vee \underbrace{b_1 b_1 \dots b_1}_{k-1})(\underbrace{b_1 b_1 \dots b_1}_{k-1} \vee \underbrace{b_1 b_1 \dots b_1}_{k-1}) \dots$$
$$\dots (\underbrace{b_1 b_1 \dots b_1}_{r} \vee \underbrace{b_1 b_1 \dots b_1}_{r}).$$

В это выражение не может входить β, так как длина любой простой конкатенации в нем меньше чем длина β. Длина любых двух соседних скобок больше β.

Допустим теперь, что $\alpha_1 = \lambda$. Тогда $A = b_1 b_2 \dots b_k \alpha_2$. Тогда делаем замену

$$A = b_1 b_2 \dots b_{k-1} (b_k \alpha_2 \vee b_k \alpha_2).$$

Если $\beta \rhd b_k \alpha_2$, то $b_k \alpha_2 = b_1 b_2 \dots b_k \alpha_2'$. Этот случай рассматривается аналогично случаю $\alpha_1 \neq \lambda$.

Таким образом A всегда можно эквивалентно преобразовать так, что β в нее входить не будет. Лемма доказана.

Теорема 3. *Если B и C — программы, $|C| > 2$ и $\mathfrak{B} = \bigvee_{j=1}^{m} B_j$ — каноническая запись программы B, то из сильной зараженности B вирусом C следует выполнение следующих условий:*

- *Для некоторого k : $B_k \equiv C$;*
- *Все $B_j (j \neq k)$ не имеют общих суффиксов и префиксов с C*

Доказательство. Пусть $\forall i\ B_i \neq C$. Если C не входит ни в одно из B_i, то по Теореме 1 B не заражено вирусом C.

Пусть $C \rhd B_k$. Тогда $B_k = A_1 C A_2$, где A_1, A_2 — простые конкатенации, $A_1 A_2 \neq \lambda$. По Лемме 3, выражение B_k можно преобразовать так, что в него не будет входить подформула C, то есть найдется такое эквивалентное выражение, которое не будет содержать подформулы, равной вирусу, что противоречит условию сильной зараженности программы вирусом. Следовательно, $A_1 A_2 = \lambda$ и $B_k \equiv C$.

Пусть не выполняется второе условие. То есть $\exists l \neq k$ такое, что, B_l и B_k имеют общий префикс длины $r \geqslant 1$ (случай для суффикса расматривается аналогично).

$$C \equiv B_k = a_{i_1} \dots a_{i_r} a_{i_{r+1}} \dots a_{i_n}$$
$$B_l = a_{i_1} \dots a_{i_r} a_{j_1} \dots a_{j_m}.$$

Но тогда по аксиоме A4

$$B_k \vee B_l = a_{i_1} \dots a_{i_r} (a_{i_{r+1}} \dots a_{i_n} \vee a_{j_1} \dots a_{j_m})$$

Обозначим $A = a_{j_1} \dots a_{j_m}$. Если $C \rhd A$, то либо $A = C$, либо $A = \alpha_1 C \alpha_2$, где $\alpha_1 \alpha_2 \neq \lambda$. Во втором случае по Лемме 3 можно эквивалентно преобразовать выражение A так, чтобы в него не входила подформула, равная C. В первом случае можем вынести любой общий суффикс для $a_{i_{r+1}} \dots a_{i_n}$ и A. В итоге полученное выражение не будет содержать подформулы, равной C. Следовательно существует выражение, равное исходной программе B, в которую не входит подформула, равная вирусу C. Что противоречит условию сильной зараженности программы B вирусом C. Утверждение доказано.

Произвольные регулярные выражения без итераций

Теорема 4. *Если B, C — программы, $\mathfrak{C} = \bigvee_{j=1}^{k} C_j$ — каноническая форма вируса, то B заражена вирусом C в том и только в том случае, если существуют простые конкатенации A_1, A_2 такие, что $\bigvee_{j=1}^{k} A_1 C_j A_2$ входит как подформула в $\mathfrak{B}$ (каноническую форму программы B).*

Доказательство. *Необходимость.* Пусть B заражено вирусом C. Тогда существует $E = B$, такое что $\exists E' = C$, $E' \rhd E$. Преобразуем E' тождественно, получаем

$$E'' = \bigvee_{j=1}^{k} C_j$$

Будем преобразовывать E к канонической форме, считая E'' за один символ. Получим

$$\mathfrak{E} = \bigvee_{i=1}^{n} E_i$$

где в одно или несколько E_i будет входить E''. Пусть, например, $E_k = A_1 E'' A_2$ $(1 \leqslant k \leqslant n)$. Применим аксиомы дистрибутивности для E_k:

$$E_k = A_1(\bigvee_{j=1}^{k} C_j)A_2 = \bigvee_{j=1}^{k} A_1 C_j A_2$$

Аналогично поступаем с остальными E_i, в которые входит E''. Таким образом, в каноническую форму входит выражение $\bigvee_{j=1}^{k} A_1 C_j A_2$.

Достаточность. Пусть в каноническую форму программы $\mathfrak{B}$ входит подформула $E' = \bigvee_{j=1}^{k} A_1 C_j A_2$ (существуют такие простые конкатенации A_1, A_2).

Вынесем за скобку A_1 и A_2, получим:

$$E'' \equiv A_1(\bigvee_{j=1}^{k} C_j)A_2$$

То есть существует выражение, равное программе B, которое содержит подформулу $\bigvee_{j=1}^{k} C_j$, равную вирусу C, следовательно программа B заражена вирусом C. Утверждение доказано.

Теорема 5. *Если B — программа, C — вирус, $\mathfrak{B} \equiv \bigvee_{i=1}^{m} B_j, \mathfrak{C} \equiv \bigvee_{j=1}^{k} C_j$ — канонические формы программы и вируса соответственно, то для того чтобы программа B была сильно заражена вирусом C необходимо и достаточно выполнения следующих условий:*

1) $\mathfrak{C} \rhd \mathfrak{B}$;

2) C_j и B_i $(1 \leqslant j \leqslant k, 1 \leqslant i \leqslant m)$ либо совпадают либо не имеют общих суффиксов и префиксов.

Доказательство. *Достаточность.* Пусть $\mathfrak{C} \rhd \mathfrak{B}$. Тогда можем представить $\mathfrak{B}$ в виде

$$\mathfrak{B} = \mathfrak{C} \vee B_{i_1} \vee \ldots \vee B_{i_{m-k}}.$$

Любые тождественные изменения $\mathfrak{C}$ оставляют его равным C. Ни одно из C_j не равно B_{i_l} $(1 \leqslant l \leqslant n-k)$.

$$\mathfrak{B} = C_1 \vee C_2 \vee \ldots \vee C_k \vee B_{i_1} \vee \ldots \vee B_{i_{m-k}}.$$

Любое $\exists B' = \mathfrak{B}$ можно получить из $\mathfrak{B}$ с помощью аксиом A1–A6 и правила вывода R1. A1–A3 не влияют на возможность вхождения C как подформулы. A4 и A5 не можем применить, так как нет общих суффиксов и префиксов соотвественно. A6 применить невозможно, так как каноническая форма уже избавлена от подформул вида $A \vee A$.

Таким образом, применяя эквивалентные преобразования, получаем выражение B', в которое может входить как подформула вирус C или равные ему выражения. Другими словами, программа B сильно заражена вирусом C.

Необходимость. Пусть B сильно заражено вирусом C. Тогда, по определению, $\exists C' = C$, $C' \rhd \mathfrak{B}$.

Подформулами $\mathfrak{B}$ могут быть либо простые, конкатенации либо выражения вида $B_{i_1} \vee \ldots \vee B_{i_q}$. Если C' — простая конкатенация, то из Теоремы 3 следует доказываемое утверждение. Во втором случае C' представляет собой объединение простых конкатенаций, а значит совпадает с канонической формой вируса.

Таким образом,

$$\mathfrak{B} = \mathfrak{C} \vee B_{i_1} \vee \ldots \vee B_{i_{m-k}} = C_1 \vee C_2 \vee \ldots \vee C_k \vee B_{i_1} \vee \ldots \vee B_{i_{m-k}}.$$

Допустим, C_s имеет общий префикс с B_{i_s}, обозначим этот префикс $C_s\rfloor$. Тогда преобразуем выражение заменой $C_s \vee B_{i_s}$ на $C_s\rfloor(\lfloor C_s \vee \lfloor B_{i_s})$, где $\lfloor C_s, \lfloor B_{i_s}$ — части слов без префикса.

Так как $\forall l, j\ B_{i_l} \neq C_j$, то C_j уже не входит в полученное выражение. Кроме того $\forall l\ C_s\rfloor(\lfloor C_s \vee \lfloor B_{i_s}) \neq C_l$.

Ни одна из подформул полученного выражения не равна C. Другими словами, мы получили выражение, равное $\mathfrak{B}$, а следовательно равное исходной программе, в которую не может входить подформула, равная вирусу C. Программа B не заражена сильно вирусом C. Полученное противоречие доказывает утверждение. Случай с общим суффиксом доказывается аналогично.

4. Регулярные выражения звездной глубины 1

Будем рассматривать регулярные выражения, содержащие оператор итерации $(*)$. Наложим ограничение на вложенность — будем считать, что нет вложенных итераций, то есть звездная глубина равна 1.

Из аксиомы A7 с очевидностью вытекает

Теорема 6. *Если B, C — программа и вирус соотвественно, B содержит подформулу B_1* и B_1 заражено (сильно заражено) вирусом C, то B так же заражено (сильно заражено) вирусом C.*

Доказательство. В аксиоме A7 выражение с оператором итерации входит как в правую, так и в левую части. Поэтому при любых эквивалентных преобразованиях программы B, подформула B_1* будет входить в нее. Пусть B_1 заражена вирусом, тогда существует равное подформуле B_1 выражение, содержащее в качестве подформулы выражение, равное вирусу. Но в таком случае для B также будет существовать равное ему выражение, содержащее в качестве подформулы выражение, равное вирусу. Отсюда следует зараженность программы B вирусом C. Аналогично, если B_1 заражена вирусом сильно, то и программа B будет заражена вирусом C сильно. Теорема доказана.

Пусть B содержит подформулу с оператором Клини B_1*. Тогда k-развернутой формой формулы B по подформуле B_1* назовем результат k-кратного применения к этой подформуле аксиомы A7

$$\alpha* = \alpha\alpha * \vee\alpha$$

и замены подформулы B_1* на новый символ $b_1 \notin \Sigma$. Тогда в расширенном алфавите $\Sigma' = \Sigma \cup \{b_1\}$ полученное выражение будет выражением без оператора итерации. Будем обозначать k-развернутую

форму выражения B по всем подформулам, содержащим итерацию, с помощью $B^{(k)}$.

Случай простого вируса

Пусть вирус представляет собой простую конкатенацию: $C = a_{i_1}a_{i_2}\dots a_{i_n}$.

Лемма 3. *Пусть α, β — простые конкатенации. Существует число n_0 такое, что если $\beta \not\triangleright \alpha^{n_0}$, то $\beta \not\triangleright \alpha^n$ $\forall n > n_0$.*

Доказательство. Возьмем $n_0 = \left\lceil \frac{|\beta|}{|\alpha|} \right\rceil + 1$. Пусть $\beta \not\triangleright \alpha^{n_0}$. Будем проверять вхождение подформулы β в α^n, $n = n_0 + 1$. Так как простые конкатенации — это последовательность символов, то будем рассматривать их как строки символов. Вхождение β не может начинаться с символов под номерами с 1 до $|\alpha|n_0 - |\beta| + 1$ (по условию). Однако $|\alpha|n_0 - |\beta| + 1 = |\alpha| \cdot \left\lceil \frac{|\beta|}{|\alpha|} \right\rceil + |\alpha| - |\beta| + 1 > |\alpha|$, поэтому строка β может входить в строку α^n начиная с номера, который не меньше $|\alpha| + 1$. Но последовательность символов с номерами от $|\alpha| + 1$ до $|\alpha^{n_0+1}|$ совпадает со строкой α^{n_0}. А в нее β не входит. Аналогично, при дальнейшем увеличении n невозможно получить вхождение β в $|\alpha^n|$.

Покажем, что нельзя уменьшить выбранное n_0. Допустим, $n_0 = \left\lceil \frac{|\beta|}{|\alpha|} \right\rceil$. Возьмем, например, $\alpha = abc$, $\beta = cabcabca$ в алфавите $\Sigma = \{a, b, c\}$. В этом случае, $|\alpha| = 3$, $|\beta| = 8$, $n_0 = 3$, $\beta \not\triangleright \alpha^3$, но при этом $\beta \triangleright \alpha^4$. Лемма доказана.

Лемма 4. *Если β — простая конкатенация, $\alpha = \bigvee_{i=1}^{k} \alpha_i$, где α_i — простые конкатенации, то существует число n_0 такое, что если $\beta \not\triangleright \alpha^{n_0}$, то $\beta \not\triangleright \alpha^n$ $\forall n > n_0$.*

Доказательство. По лемме 4 существуют такие числа n_i для каждого α_i, что

$$\beta \not\triangleright \alpha_i^{n_i} \to \beta \not\triangleright \alpha_i^n \ \ \forall n > n_i.$$

Выберем

$$n_0 = \max_{1 \leqslant i \leqslant m} n_i + \left\lceil \frac{\max\limits_{1 \leqslant i \leqslant m} |\alpha_i|}{\min\limits_{1 \leqslant i \leqslant m} |\alpha_i|} \right\rceil = \left\lceil \frac{|\beta|}{\min\limits_{1 \leqslant i \leqslant m} |\alpha_i|} \right\rceil + \left\lceil \frac{\max\limits_{1 \leqslant i \leqslant m} |\alpha_i|}{\min\limits_{1 \leqslant i \leqslant m} |\alpha_i|} \right\rceil.$$

Предположим, что $\beta \not\rhd \alpha^{n_0}$.

После раскрытия α^n $(n = n_0 + 1)$ скобок по аксиомам A4–A5, выражение примет следующий вид:

$$\alpha^n = \bigvee_{1 \leqslant i_1, i_2 \ldots i_n \leqslant m} \alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_n} \tag{3}$$

В подобном разложении содержатся всевозможные размещения с повторениями длины n из множества конкатенаций $\{\alpha_i,\ 1 \leqslant i \leqslant k\}$.

По Лемме 4 β не будет входить в α_i^n, содержащиеся в объединении (3). По предположению, β так же не входит в конкатенации $\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_{n_0}}$ $(1 \leqslant i_1, i_2 \ldots i_{n_0} \leqslant m)$, входящие в разложение $\alpha_i^{n_0}$. Рассмотрим некоторую конкатенацию $\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_n}$, $(1 \leqslant i_1, i_2 \ldots i_n \leqslant m)$ из разложения (3), обозначим ее γ. Будем рассматривать γ и β как строки. При этом β не может входить в γ начиная с символов под номерами с 1 до $|\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_{n_0}}| - |\beta| + 1$ (иначе β входила бы в $\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_{n_0}}$). Заметим, что

$$|\alpha_{i_1}\alpha_{i_2}\ldots\alpha_{i_{n_0}}| - |\beta| + 1 \geqslant \min_{1\leqslant i\leqslant m}|\alpha_i| \cdot n_0 - |\beta| + 1 =$$

$$= \min_{1\leqslant i\leqslant m}|\alpha_i| \cdot \left\lceil \frac{|\beta|}{\min\limits_{1\leqslant i\leqslant m}|\alpha_i|} \right\rceil + \min_{1\leqslant i\leqslant m}|\alpha_i| \cdot \left\lceil \frac{\max\limits_{1\leqslant i\leqslant m}|\alpha_i|}{\min\limits_{1\leqslant i\leqslant m}|\alpha_i|} \right\rceil - |\beta| + 1 \geqslant$$

$$\geqslant \max_{1\leqslant i\leqslant m}|\alpha_i| + 1 \geqslant |\alpha_i| + 1, \quad \forall i = \overline{1,m}$$

Таким образом, вхождение β в конкатенацию γ может начинаться с символа с номером не менее $|\alpha_{i_1}|+1$. А последовательность символов с номерами с $|\alpha_{i_1}| + 1$ по $|\gamma|$ есть конкатенация $\alpha_{i_2}\alpha_{i_3}\ldots\alpha_{i_{n_0+1}}$ — конкатенация из n_0 элементов, которая встречается в разложении для α^{n_0}. Следовательно, β не может входить в выбраную конкатенацию γ. Так как γ была выбрана произвольно, β не может входить в α^n для $n = n_0 + 1$.

Аналогично, при дальнейшем увеличении n невозможно получить вхождение β в $|\alpha^n|$. Лемма доказана.

Теорема 7. *Если B, C — программа и вирус соотвественно, C — простая конкатенация, B содержит подформулы с итерациями*

звездной глубины не более 1, то существует такое число n_0, что зараженность программы B вирусом C равносильна зараженности $B^{(n_0)}$ вирусом C.

Доказательство. Пусть B содержит подформулы $B_1*, B_2*, \dots B_k*$. Так как $B_1, B_2, \dots B_k$ не содержат операторов итераций, можем получить их канонические формы $\mathfrak{B}_1, \mathfrak{B}_2, \dots, \mathfrak{B}_k$. В таком случае в n-развернутую форму выражения B по подформуле B_i* вместо этой подформулы будет входить подформула $(\mathfrak{B}_i^n b_i \vee \mathfrak{B}_i^{n-1} \vee \dots \vee \mathfrak{B}_i)$, где $b_i \in \Sigma'$ — новый символ, на который было заменено выражение B_i*. По лемме 5 для каждого $\mathfrak{B}_i$ существует число n_i, такое что

$$C \not\triangleright \mathfrak{B}_i^{n_i} \to C \not\triangleright \mathfrak{B}_i^n \ \ \forall n > n_i.$$

Выберем $n_0 = \max\limits_{1 \leqslant i \leqslant m} n_i$. По Лемме 5, если $C \not\triangleright \mathfrak{B}_i^{n_0}$, то для всех $n > n_0$ вирус C не будет входить в $\mathfrak{B}_i^n$. Так как $b_i \notin \Sigma$ $(1 \leqslant i \leqslant k)$, новые символы не будут влиять на возможность вхождения C в $B^{(n_0)}$. Таком образом, если $C \triangleright B^{(n_0)}$, то $C \triangleright B^{(n)}$ $\forall n > n_0$ и наоборот, если $C \not\triangleright B^{(n_0)}$, то $C \triangleright B^{(n)}$ $\forall n > n_0$. Кроме того, если $C \triangleright B^{(m)}$, $0 \leqslant m < n_0$, то $C \triangleright B^{(n_0)}$. Таким образом, если программа заражена вирусом, то им будет заражена n_0-развернутая форма по всем подформулам, содержащим звездочку. Если заражена n_0-развернутая форма, то заражена и сама программа, так как $B^{(n_0)} = B$ в алфавите Σ'. Теорема доказана.

Теорема 7 дает возможность рассматривать не бесконечное множество равных программе B выражений, а конечное выражение без операторов итерации и применять к ней результаты предыдущего раздела.

5. Об алгоритмической разрешимости

Основным результатом работы является

Теорема 8. *Пусть вирусы и программы представляют собой регулярные выражения без оператора итерации. Тогда существует алгоритм, который для любого вируса C и для любой программы B определяет ее зараженность или незараженность.*

Доказательство. Основывается на Теореме 1 и Теореме 4 и Лемме об инвариантности.

Любые заданные программу и вирус можно привести к канонической форме. При этом, по Лемме 1, свойство зараженности сохранится. Получение канонической формы регулярного выражения — алгоритмически разрешимая задача, заключающаяся в последовательном применении аксиом дистрибутивности A4–A5 к выражению. Проверка вхождения простой конкатенации в формулу представляет собой задачу поиска подстроки в строке, для которой известны алгоритмы решения [8]. Таким образом, задача распознавания зараженности вирусом для данного класса регулярных выражений алгоритмически разрешима.

Аналогично, из Теоремы 5 следует

Теорема 9. *Пусть вирусы и программы представляют собой регулярные выражения без оператора итерации. Тогда существует алгоритм, который для любого вируса C и для любой программы B определяет степень зараженности программы B вирусом C.*

6. Заключение

В разделе 6 доказана алгоритмическая разрешимость проблемы распознавания зараженности программы вирусом для класса регулярных выражений без оператора итерации. Кроме того, доказан критерий зараженности для определенного класса выражений, содержащих оператор итерации. Рассмотрены критерии сильной зараженности для некоторых классов выражений без итераций. При этом все критерии легко могут быть преобразованы в соотвествующие алгоритмы распознавания.

Полученные результаты могут быть использованы в дальнейшем изучении распознавания зараженности, которое заключается в нахождении критериев сильной и слабой зараженности для вирусов с оператором итерации, в исследовании свойства зараженности для более сложных классов программ — регулярных выражений звездной глубины больше 1. Другое возможное направление исследований — улучшение алгоритмов распознавания, повышение их вычислительной эффективности.

Список литературы

[1] Bonfante G., Kaczmarek M., Marion J.-Y. On abstract computer virology from a recursion-theoretic perspective // Journal in Computer Virology. 1, 3–4. 2006. P. 45–54.

[2] Corradini F., De Nicola R., Labella A. Models of Nondeterministic Regular Expressions // Journal of Computer and System Sciences. 59. 1999. P. 412–449.

[3] Cohen F. Computer Viruses / PhD thesis. University of Southern California. January 1986.

[4] Niemi V. The undecidability of form equivalence for context-free and E0L forms // Theoretical Computer Science. Vol. 32, iss. 3. 1984. P. 261–277.

[5] Han Y.-S., Wood D. Shorter Regular Expressions from Finite-State Automata // Proceedings of the 10th International Conference on Implementation and Application of Automata (CIAA). LNCS 3845. 2005. P. 141–152.

[6] Ахо А., Ульман Дж. Теория синтаксического анализа, перевода и компиляции. Т. 1. Синтаксический анализ. М.: Мир, 1978.

[7] Болонкин М. О распознавании вирусов в регулярных текстах // Сборник докладов Республиканской научной конференции «Вычислительные технологии и математическое моделирование», посвященной 75-летию Маруфа Исраиловича Исраилова. Ташкент, 2009.

[8] Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ = Introduction to Algorithms / Под ред. И. В. Красикова. М.: Вильямс, 2005.

[9] Кудрявцев В. Б., Алешин С. В., Подколзин А. С. Введение в теорию автоматов. М.: Наука, 1985.

[10] Саломаа А. Аксиоматизация алгебры событий, реализуемых логическими сетями // Проблемы кибернетики. М.: Наука, 1966. Вып. 17. С. 237–246.

[11] Computer Economics. «Annual Worldwide Economic Damages from Malware Exceed $13 Billion». http://www.computereconomics.com/article.cfm?id=1225.