ФОРМИРОВАНИЕ ТЕСТОВ ДЛЯ ПРОВЕРКИ СПОСОБНОСТИ ДЕМАСКИРОВКИ ЗАИМСТВОВАНИЙ В ПРОГРАММАХ ВЫЯВЛЕНИЯ ПЛАГИАТА

Виктор Шинкаренко, Елена Куропятник

***Аннотация**: Исследования направлены на обеспечение своевременного реагирования на вновь открываемые возможности механического маскирования текстовых заимствований. Предлагается нестандартный подход к тестированию программного обеспечения, при котором выходные данные тестов формируется тестируемой программой. Выполнена формализация средств проверки способности демаскировки заимствований в программах выявления плагиата в виде конструкторов сценариев модификации текстов и процесса их применения. Основой методологии исследований является математический аппарат конструктивно-продукционного моделирования. Основополагающий принцип этого подхода – все сущности и процессы реального и виртуального мира рассматриваются как конструкции и конструктивные процессы с соответствующей атрибутикой. Использование указанного подхода позволяет автоматизировать процессы подготовки текстов с маскировкой заимствований, а также дает возможность дальнейшего развития сценариев модификации.*

***Ключевые слова**: конструктивно-продукционное моделирование, конструктор, заимствования текстов, маскировка заимствований, сценарий маскировки, тест.*

***ITHEA Keywords**: F.4.2 Grammars and Other Rewriting Systems; I.2.7 Natural Language Processing; H.3.1 Content Analysis and Indexing; J.4 Social and Behavioral Sciences; D.2.5 Testing and Debugging.*

Введение

Развитие современных технологий накопления и обмена информацией в свободном доступе обострило проблему заимствования в академической среде. Это касается как научных публикаций, так и студенческих работ [Bakhtiyari, 2014, Букач, 2016 и др.], в том числе текстов компьютерных программ [Mahmoud, 2017].

Проблема заимствований заключается в использовании текстовой и другой информации без указания источника, а также обнародование ее под своим именем, не являясь автором. Данная проблема отождествляется с проблемой плагиата в широком понимании.

Для решения этой проблемы разработаны программные средства выявления недобросовестных заимствований [Михайловский, 2013, Vani, 2018 и др.], анализ которых выполнен в [Шинкаренко, 2017]. Не будем останавливаться на сложности решения этой задачи, связанной с такими факторами как объемы, значимость, выявление источников, переводом используемых фрагментов и др. [Meuschke, 2013].

В данной работе проблема заимствований рассматривается в контексте постоянного развития противоборствующих сторон: одна сторона (защищающаяся) пытается выявить заимствования, а другая (атакующая) – скрыть (замаскировать). Аналогичная проблема существует и в других сферах, в том числе – информационных технологий. Например, разработка и борьба с вирусами (естественными и компьютерными), взлом и защита информации.

Механическая маскировка заимствований заключается в использовании автоматизированных средств для внесения несемантических изменений в текст с целью усложнения выявления плагиата. К таким изменениям относятся: удаление или замена отдельных букв (в тексте изредка появляются орфографические ошибки), добавление пустых абзацев, вставка непечатных символов, вставка символов белого цвета, замена символов на такие же другого языка (с разным машинным кодом, но одинаковым отображением), изменение регистра некоторых символов и ряд других [Ересь, 2018, Шишкин, 2017].

Для своевременного реагирования на изменение методов атаки защищающейся стороне необходимо иметь настраиваемый инструментарий на адекватной теоретической основе. Такой основой в данной работе является конструктивно-продукционное моделирование, являющееся развитием теории формальных грамматик.

В работе показано применение конструкторов для имитации работы систем маскирования заимствований. Исходный и маскированный текст с заимствованиями представляют собой тест для проверки возможностей программ антиплагиата по демаскировке заимствований.

Связанные работы

Исследованию и решению проблемы плагиата посвящено множество работ в сферах образования [Букач, 2016, Федоровская, 2016] и разработки программного обеспечения [Gulis, 2016]. Разработка программных систем антиплагиата является одним из основополагающих среди современных методов борьбы с плагиатом.

В зависимости от разных вариаций изменений входного текста, заимствования, выявленные в них, могут быть классифицированы согласно таксономии [Шинкаренко, 2017, Chowdhury, 2016].

Программы выявления плагиата включают различную предварительную обработку текста [Ceska, 2009, [Mahmoodi, 2014, Xu, 2012]. Такая обработка должна включать средства

демаскирования заимствований. Некоторые возможности по демаскировке приведены в [Ересь, 2018].

Представленные исследования выполнены на основании конструктивно-продукционного моделирования, основы которого заложены в [Shynkarenko, 2014], с приложением к задачам выявления заимствований [Шинкаренко, 2016].

Цель и задача исследования

Цель работы можно рассматривать на нескольких уровнях:

1. Контроль добросовестного отношения к подготовке текстовых документов в академической среде средствами программ выявления плагиата;
2. Противодействие системам механического сокрытия заимствований в текстах. Своевременное реагирование на вновь открываемые возможности механического маскирования заимствований;
3. Разработка теоретических основ и соответствующего программного обеспечения для проверки способности демаскировки заимствований в программах выявления плагиата.

В соответствие с поставленными целями необходимо разработать конструктор сценариев модификации текстов и конструктор модификации текстов.

Конструирование сценариев модификации текстов

Сценарии модификации – последовательности операций над текстом. Под операциями будем понимать действия, направленные на обработку текста: поиск и редактирование элементов без изменения семантики.

Для формализации сценариев используем конструктор [Shynkarenko, 2014]:

$$C = \langle M, \Sigma, \Lambda \rangle, \quad (1)$$

где M – неоднородный расширяемый носитель, Σ – сигнатура отношений и соответствующих операций: связывания, подстановки, вывода, над атрибутами, Λ – множество утверждений информационного обеспечения конструирование (ИОК). Данное множество включает: онтологию, цель, правила, ограничения, условия начала и завершения конструирования.

Выполним уточняющие преобразования конструктора для моделирования сценариев: специализацию, интерпретацию, конкретизацию, реализацию [Shynkarenko, 2014]. Данные преобразования приводят к расширению информационного обеспечения.

Специализация конструктора предполагает определение семантической составляющей носителя:

$$C = \langle M, \Sigma, \Lambda \rangle \ {}_s\mapsto C_{SC} = \langle M_{SC}, \Sigma_{SC}, \Lambda_{SC} \rangle, \quad (2)$$

где ${}_s\mapsto$ – операция специализации, M_{SC} – носитель, элементами которого являются терминалы и нетерминалы, Σ_{SC} – сигнатура отношений и операций для построения конструкций – алгоритмов и сценариев, Λ_{SC} – множество утверждений ИОК, которое будет рассмотрено далее.

Онтология конструктора C_{SC} представлена следующими утверждениями. Носитель $M_{SC} = T \cup N$ является расширяемым. Терминалы (T) включают конечное множество алгоритмов выполнения операций непосредственно над текстом $\{B_i^0\}$ и дополнительных (вспомогательных) $\{B_i^1\}$, а также сконструированные алгоритмы и их последовательности – сценарии и необходимые для них данные. Базовые алгоритмы реализуют атомарные (условно неделимые) действия. Нетерминалы – вспомогательные элементы, составляющие множество абстрактных алгоритмов.

Текст, к которому применяются базовые алгоритмы, имеет ряд атрибутов: $ch\_count \lrcorner TXT$ – количество символов, $w\_count \lrcorner TXT$ – количество слов, $p\_count \lrcorner TXT$ – количество абзацев.

К базовым относятся алгоритмы операций над элементами текста:

- $B_1^0 \mid_{pos,el,kind,TXT}^{TXT'}$ – вставка элемента el типа $kind = \{\text{character}, \text{word}, \text{paragraph}\}$ в позицию pos в тексте TXT, в результате выполнения которой TXT' включает el (здесь и далее TXT' – модификация TXT);
- $B_2^0 \mid_{pos,TXT}^{TXT'}$ – удаления элемента, находящегося на позиции pos в тексте;
- $B_3^0 \mid_{el_1,el_2,kind,TXT}^{TXT'}$ – замена элемента текста el_1, принадлежащего тексту TXT, на el_2, $kind$ – тип элемента ($kind = \{\text{character}, \text{word}, \text{paragraph}\}$);
- $B_4^0 \mid_{s\_pos,e\_pos,TXT}^{range}$ – присвоение $range$ текстового фрагмента, который находится между указанными позициями s_pos и e_pos;
- $B_5^0 \mid_{ft,range,TXT}^{TXT'}$ – изменения форматирования $range$ текстового фрагмента, принадлежащего TXT, с использованием формата ft, который включает атрибуты шрифта, как в TrueType шрифтах Windows.
- $B_6^0 \mid_{el,TXT}^{res}$ – проверка наличия в тексте TXT таблиц и принадлежности элемента el одной из них, если принадлежит – $res = true$, иначе $false$;

- $B_7^0 \mid_{el,TXT,start\_pos}^{pos}$ – поиск элемента *el* в тексте *TXT*, начиная с позиции *start_pos*, *pos* равен позиции *el* в тексте *TXT*, если *el* найден в *TXT*, иначе – $pos = -1$;
- $B_8^0 \mid_{kind,num,TXT}^{el}$ – присвоение *el* элемента типа *kind* (символ, слово, абзац) по номеру *num* текста *TXT*, *el* – идентифицированный элемент;
- $B_9^0 \mid_{el_1,el_2,TXT}^{TXT'}$ – вставка элемента el_1 после el_2 в *TXT*, результат – $el_1 el_2$ содержаться в *TXT′*;
- $B_{10}^0 \mid_{el_1,TXT}^{TXT'}$ – удаление el_1 из *TXT*, *TXT′* содержит все элементы *TXT*, кроме el_1.

Также к ним относится ряд вспомогательных алгоритмов:

- $B_1^1 \mid_{q}^{seq}$ – формирования последовательности *seq* из *q* пробелов;
- $B_2^1 \mid_{a,b}^{c}$ – генерации случайного числа *c* в диапазоне $[a,b]$;
- $B_3^1 \mid_{c,D}^{val}$ – поиска символа *c* в словаре *D*, *val* – значение символа по словарю, если символ не найден $val = '1'$ (проверка наличия в словаре символьных «омографов»);
- $B_4^1 \mid_{a,b}^{c}$ – вычитания числа *a* из *b*, $c = a - b$ – результат;
- $B_5^1 \mid_{a,b}^{a}$ – присвоения *a* значения *b* $(a := b)$.

Форма *I* – совокупность элементов носителя M_{SC}, связанных отношениями Σ_{SC}.

Конструкция *K* – сентенциальная форма в текущий момент времени, содержащая только терминалы [Shynkarenko, 2014]. К конструкциям относятся алгоритмы и сценарии.

Сигнатура Σ_{SC} состоит из множества операций $\Sigma_{SC} = \langle \Xi, \Theta, \Phi, \{\rightarrow\}, \Psi \rangle$, где Ξ – отношения и одноименные операции, операции связывания и преобразования элементов носителя $\{;,:,\prod\} \subset \Xi$, $\Theta = \{\Rightarrow, |\Rightarrow, ||\Rightarrow\}$ – операции подстановки и частичного и полного вывода, Φ – отношения и одноименные операции над атрибутами, $\{\rightarrow\}$ – отношение подстановки. $\Psi = \{\psi_i : \langle s_i, g_i \rangle\}$ – набор правил подстановки, s_i – последовательность отношений подстановки, g_i – набор операций над атрибутами. Если операции над атрибутами не выполняются, правило подстановки будет иметь вид $\langle s_i, \varepsilon \rangle$, где ε – пустой символ. В правилах вывода применяются отношения из Ξ, а при реализации конструктора – операции, соответствующие отношениям.

Операция $\cdot(B_i, B_j)$ – конкатенация алгоритмов, предполагает последовательное выполнение B_j после B_i.

Операция $:(b,A)$ – выполнение алгоритма A при условии, что выражение b - истинно ($b = true$).

Операция $\prod_{i=1}^{n} A$ – выполнение алгоритма A n раз.

Отношение подстановки – двуместное отношение с атрибутами ${}_{w_i}l_i \; {}_{w} \rightarrow {}_{w_j}l_j$, где l_i, l_j – сентенциальные формы [Shynkarenko, 2014]. Последовательность отношений подстановки s_n будем записывать как $s_n = \langle l_i \rightarrow l_j, \ldots, l_m \rightarrow l_k \rangle$, где l_i, l_j, l_m, l_k – сентенциальные формы Отношение подстановки может быть записано в сокращенной форме $s_m = \langle l_i \rightarrow l_j \mid l_k \rangle$, где l_i, l_j, l_k – формы; что эквивалентно $s_m = \langle l_i \rightarrow l_j \rangle$, $s_n = \langle l_i \rightarrow l_k \rangle$.

Для заданной формы ${}_{w_l}l = {}_{w_0}\otimes({}_{w_1}l_1, {}_{w_2}l_2, \ldots, {}_{w_h}l_h, \ldots, {}_{w_k}l_k)$ и доступного отношения подстановки ${}_{w_p}\rightarrow({}_{w_h}l_h, {}_{w_q}l_q)$ такого, что ${}_{w_h}l_h$ – подформа ${}_{w_l}l$ $({}_{w_h}l_h \prec {}_{w_l}l)$, результатом трехместной операции подстановки ${}_{w_l^*}l^* = {}_{w_p}\Rightarrow({}_{w_h}l_h, {}_{w_q}l_q, {}_{w_l}l)$ будет форма ${}_{w_l^*}l^* = {}_{w_0}\otimes({}_{w_1}l_1, {}_{w_2}l_2, \ldots, {}_{w_q}l_q, \ldots, {}_{w_k}l_k)$, где $\Rightarrow \in \Theta$, $\otimes$ – любая операция связывания из Ξ.

Двухместная операция частичного вывода ${}_{w_l^*}l^* = {}_{w_p}|\Rightarrow(\Psi, {}_{w_l}l)$ $(|\Rightarrow \in \Theta)$ заключается в:

- выборе одного из доступных правил подстановки $\psi_r : \langle s_r, g_r \rangle$ с отношениями подстановки s_r;
- выполнении на его основе операций подстановки.

Двухместная операция полного вывода или просто вывода ($||\Rightarrow(\Psi, {}_{w_l}l)$, $||\Rightarrow \in \Theta$) заключается в пошаговом преобразовании форм, начиная с начального нетерминала и заканчивая конструкцией, удовлетворяющей условию окончания вывода, что подразумевает циклическое выполнение операций частичного вывода.

Целью конструктора является конструирование более сложных алгоритмов на основе базовых, а также построение их последовательностей – сценариев.

Начальное условие для конструирования сценариев: вывод начинается с нетерминала SC.

Условие завершения конструирования: форма не содержит нетерминалов.

Интерпретация операций конструктора предполагает определение модели исполнителя, которую представим в виде базовой алгоритмической структуры (БАС):

$$\langle C_{SC}, C_{A,SC} = \langle M_{A,SC}, V_{A,SC}, \Sigma_{A,SC}, \Lambda_{A,SC} \rangle \rangle \; {}_{I} \mapsto \langle {}_{A}C_{SC}, {}_{A}C_{SC} = \langle M_1, \Sigma_1, \Lambda_1 \rangle \rangle, \qquad (3)$$

где $_I\mapsto$ – операция интерпретации, $M_{A,SC} \supset M_{SC} \cup \Omega(C_{A,SC})$ – неоднородный носитель, $\Omega(C_{SC})$ – множество сконструированных алгоритмов, которые удовлетворяют C_{SC}; $V_{A,SC} = \{A_i^0 \mid_{X_i}^{Y_i}\}$ – множество базовых алгоритмов [Shynkarenko, 2009], X_i, Y_i – входные и выходные данные алгоритма $A_i^0 \mid_{X_i}^{Y_i}$, $\Sigma_{A,SC}$ – сигнатура и $\Lambda_{A,SC}$ – информационное обеспечение; $\Lambda_{A,SC} = \{M_{A,SC} = \bigcup\limits_{A_i^0 \in V_{A,SC}} (X(A_i^0) \cup Y(A_i^0)) \cup \Omega(C_{A,SC})\}$, $\Omega(C_{A,SC})$ – множество алгоритмов реализуемых конструктором $C_{A,SC}$; $\Lambda_1 = \{(A_1^0 \mid_{A_i,A_j}^{A_i \cdot A_j} \hookleftarrow$ ". "$)$; $(A_2^0 \mid_b^{A_i} \hookleftarrow$ " : "$)$; $(A_3 \mid_{n,A}^{A \cdot A \cdot \ldots} \hookleftarrow$ "$\prod$ "$)$; $(A_4 \mid_{I_h,I_q,f_i}^{f_i} \hookleftarrow$ "$\Rightarrow$ "$)$; $(A_5 \mid_{f_i,\Psi}^{f_j} \hookleftarrow$ "$\mid\Rightarrow$ "$)$; $(A_6 \mid_{\sigma,\Psi}^{\bar{\Omega}} \hookleftarrow$ "$\mid\mid\Rightarrow$ "$)$; $\Lambda_1 \supset \Lambda_{SC}$.

Конструктор $_A C_{SC}$ включает алгоритмы выполнения операций:

- $A_1^0 \mid_{A_i,A_j}^{A_i \cdot A_j}$ – композиция алгоритмов, $A_i \cdot A_j$ – последовательное выполнение алгоритма A_j после алгоритма A_i;
- $A_2^0 \mid_b^{A_i}$ – условное выполнение: алгоритм A_i выполняется, если выражение b истинно;
- $A_3 \mid_{n,A}^{A \cdot A \cdot \ldots}$ – выполение алгоритма A n раз;
- $A_4 \mid_{I_h,I_q,f_i}^{f_i}$ – подстановка, I_h, I_q, f_i – формы;
- $A_5 \mid_{f_i,\Psi}^{f_j}$, $A_6 \mid_{\sigma,\Psi}^{\bar{\Omega}}$ – частичный и полный вывод, где f_i, f_j – формы, σ – аксиома, $\bar{\Omega}$ – множество сформированных конструкций.

На основе интерпретации получаем конструктивную систему формирования сценариев.

Конкретизация конструктора определяет правила построения сценариев, на основе терминальных и нетерминальных элементов, в том числе обозначающие сконструированные алгоритмы.

Конкретизация конструктора сценариев:

$$C_{SC} \; _K\mapsto C_{SC} = \langle M_2, \Sigma_2, \Lambda_2 \rangle, \tag{4}$$

где $_K\mapsto$ – операция конкретизации, которая заключается в определении правил и начальных условий конструирования алгоритмов и сценариев, $M_2 \supset M_{SC} \cup T_1 \cup N_1$, $T_1 = \{B_i^0, B_j^1, A_2^0\}$, $N_1 = \{CA_k, CR_1, CC_m, \alpha, SC\}$, $\Lambda_2 \supset \Lambda_1$.

Рассмотрим правила и начальные условия конструирования алгоритмов, которые далее будут использованы при построении сценариев.

Начальные условия конструирования алгоритмов:

- SC – нетерминал, с которого начинается вывод;
- в алгоритмах поиска стартовая позиция инициализируется нулем ($start\_pos = 0$);
- формат, указывающий на регистр ($ft_1 = "case = lower"$ – формат нижнего регистра, $ft_2 = "case = upper"$ – формат верхнего регистра, $ft_3 = "case = next"$ – перейти к следующему регистру).

Обозначим сконструированные алгоритмы: CA_k – добавления; CR_l –удаления; CA_m – изменения.

Первое правило позволяет сконструировать алгоритм вставки последовательности qs - пробелов после случайно выбранного элемента позиции qi в текст TXT :

$$s_1 = \left\langle CA_1 \mid_{qs,qi,TXT}^{TXT'} \rightarrow B_1^1 \mid_{qs}^{seq} \cdot \prod_{j=1}^{qi} (B_2^1 \mid_{a,b}^{num} \cdot B_8^0 \mid_{kind,num,TXT}^{el} \cdot B_9^0 \mid_{seq,el,TXT}^{TXT'}) \mid_{kind=word|paragraph} \right\rangle, \quad (5)$$

Правило s_2 позволяет сконструировать алгоритм добавления в текст сторонних фрагментов (текстов) el на позицию, определяемую pos (начало, середина, конец) в текст TXT

$$\begin{aligned} s_2 = \Big\langle CA_2 \mid_{pos,el,TXT}^{TXT'} \rightarrow (A_0^2 \mid_{pos=0}^{:} : B_1^0 \mid_{0,el,TXT}^{TXT'}) \cdot (A_0^2 \mid_{pos=1}^{:} : (B_1^1 \mid_{a,b}^{c} \cdot B_1^0 \mid_{c,el,TXT}^{TXT'})) \cdot \\ \cdot (A_0^2 \mid_{pos=2}^{:} : B_1^0 \mid_{ch\_count \lrcorner TXT,el,\mathrm{kind},TXT}^{TXT'}) \mid_{kind=character,word,paragraph} \Big\rangle, \end{aligned} \quad (6)$$

Следующее правило позволяет сконструировать алгоритм добавления в текст qp пустых абзацев в случайные позиции в тексте

$$\begin{aligned} s_3 = \Big\langle CA_3 \mid_{qp,TXT}^{TXT'} \rightarrow \prod_{i}^{qp} (B_1^1 \mid_{a,b}^{c} \cdot B_8^0 \mid_{kind,c,TXT}^{el} \cdot B_6^0 \mid_{el,TXT}^{res} \cdot (A_2^0 \mid_{res=false}^{:} : B_9^0 \mid_{el_1,el,TXT}^{TXT}) \cdot \\ \cdot (A_2^0 \mid_{res=false}^{:} : B_4^1 \mid_{i,1}^{i})) \mid_{kind=paragraph} \Big\rangle. \end{aligned} \quad (7)$$

Правило s_4 позволяет сконструировать алгоритм удаления из текста TXT q абзацев начиная с заданного под номером sp.

$$s_4 = \left\langle CR_1 \mid_{sp,q,TXT}^{TXT'} \rightarrow \prod_{i=sp}^{sp+q} (A_2^0 \mid_{i<p\_count \lrcorner TXT}^{:} : (B_8^0 \mid_{kind,i,TXT}^{el} \cdot B_{10}^0 \mid_{el,TXT}^{TXT'})) \mid_{kind=paragraph} \right\rangle. \quad (8)$$

Для конструирования алгоритма изменения раскладки $type$ случайного символа в тексте может быть использовано такое правило:

$$\begin{aligned} s_5 = \Big\langle CC_1 \mid_{type,TXT}^{TXT'} \rightarrow B_1^1 \mid_{a,b}^{c} \cdot B_8^0 \mid_{kind,c,TXT}^{el} \cdot \prod_{i=1}^{k} (A_2^0 \mid_{type=i}^{:} : B_3^1 \mid_{el,D_i}^{val}) \cdot \\ (A_2^0 \mid_{val \neq '1'}^{:} : B_3^0 \mid_{el,val,kind,TXT}^{TXT'}) \mid_{kind=character} \Big\rangle. \end{aligned} \quad (9)$$

Правило s_6 позволяет сконструировать алгоритм изменения раскладки всех вхождений символа c, содержащегося в тексте TXT, l_type – раскладка для модифицируемого символа.

$$s_6 = \left\langle CC_2 \mid^{TXT'}_{l\_type,el,TXT} \rightarrow \prod_{i=1}^{k} (A_2^0 \mid^{:}_{l\_type=i} : B_3^1 \mid^{val}_{el,D_i}) \cdot \alpha \right.$$
$$\left. \alpha \rightarrow B_7^0 \mid^{pos}_{c,TXT,start\_pos} \cdot (A_2^0 \mid^{:}_{pos \geq 0} : (B_3^0 \mid^{TXT'}_{c,val,kind,TXT} \cdot B_5^1 \mid^{start\_pos}_{start\_pos,pos} \cdot \alpha)) \mid_{kind=character} \right\rangle . \quad (10)$$

Приведенное ниже правило позволяет сконструировать алгоритм изменения регистра по признаку r (верхний / нижний) для всего текста TXT

$$s_7 = \left\langle CC_3 \mid^{TXT'}_{r,TXT} \rightarrow B_4^0 \mid^{range}_{0,ch\_count \lrcorner TXT,TXT} \cdot (A_2^0 \mid^{:}_{r=true} : B_5^0 \mid^{TXT'}_{ft_1,range,TXT}) \cdot (A_2^0 \mid^{:}_{r=false} : B_5^0 \mid^{TXT'}_{ft_2,range,TXT}) \right\rangle . \quad (11)$$

Алгоритм изменения регистра символа текста TXT на позиции pos конструируется следующим образом

$$s_8 = \left\langle CC_4 \mid^{TXT'}_{pos,TXT} \rightarrow B_4^0 \mid^{range}_{pos,pos+1,TXT} \cdot B_5^0 \mid^{TXT'}_{ft,range,TXT} \right\rangle . \quad (12)$$

Правило s_9 позволяет сконструировать алгоритм изменения регистра заданного символа c, принадлежащего TXT

$$s_9 = \left\langle CC_5 \mid^{TXT'}_{c,TXT} \rightarrow B_7^0 \mid^{pos}_{c,TXT,start\_pos} \cdot CC_4 \mid^{TXT'}_{pos,TXT} \right\rangle . \quad (13)$$

Для конструирования алгоритма изменения позиций абзацев под номерами num_1, num_2 (меняются местами) применяется следующее правило:

$$s_{10} = \left\langle CC_6 \mid^{TXT'}_{num_1,num_2,TXT} \rightarrow B_9^0 \mid^{el_1}_{kind,num_1,TXT} \cdot B_6^0 \mid^{res_1}_{el_1,TXT} \cdot (A_2^0 \mid^{:}_{res_1=false} : (B_9^0 \mid^{el_2}_{kind,num_2,TXT} \cdot B_6^0 \mid^{res_2}_{el_2,TXT} \cdot \right.$$
$$\left. \cdot (A_2^0 \mid^{:}_{res_2=false} : (B_3^0 \mid^{TXT'}_{el_1,el_2,kind,TXT}))))) \mid_{kind=paragraph} \right\rangle . \quad (14)$$

Правило s_{11} позволяет сконструировать алгоритм изменения позиций слов под номерами w_num_1, w$_num_2$ (меняются местами)

$$s_{11} = \left\langle CC_7 \mid^{TXT'}_{w\_num_1,w\_num_2,TXT} \rightarrow A_2^0 \mid^{:}_{w\_num_1,w\_num_2<w\_count \lrcorner TXT} : \right.$$
$$\left. : (B_9^0 \mid^{el_1}_{kind,w\_num_1,TXT} \cdot B_9^0 \mid^{el_2}_{kind,w\_num_2,TXT} \cdot B_3^0 \mid^{TXT'}_{el_1,el_2,kind,TXT}) \mid_{kind=word} \right\rangle \quad (15)$$

Следующее отношение подстановки позволяет сконструировать алгоритм изменения слова el_1, предполагающее его замену на el_2, $type$ – вид замены (первое вхождение/ все вхождения). При определенных значениях входных параметров данный алгоритм может быть использован для удвоения пробелов

$$s_{12} = \left\langle CC_8 \mid^{TXT'}_{el_1,el_2,type,TXT} \rightarrow (A_2^: \mid^{:}_{type=2} : (B_7^1 \mid^{pos}_{el,TXT,start\_pos} . \right. \quad (16)$$

$$\cdot(A_2^0 \mid_{pos\geq 0}^{:}: (B_3^0 \mid_{el_1,el_2,TXT,kind}^{TXT'} \cdot B_5^1 \mid_{start\_pos,pos}^{start\_pos} \cdot CA_8 \mid_{TXT}^{TXT'}))) \cdot$$
$$\cdot(A_2^: \mid_{type=1}^{:}: (B_7^1 \mid_{el,TXT,start\_pos}^{pos} \cdot (A_2^0 \mid_{pos\geq 0}^{:}: (B_3^0 \mid_{el_1,el_2,kind,TXT}^{TXT'} \cdot B_5^1 \mid_{start\_pos,pos}^{start\_pos})))) \mid_{kind=word} \rangle .$$

Сценарии могут быть сконструированы с использованием таких правил:

$$\begin{gathered} SC \mid_{param,TXT}^{TXT'} \to CA_1 \cdot SC \mid_{param,TXT}^{TXT'} \mid CA_2 \cdot SC \mid_{param,TXT}^{TXT'} \mid \mathrm{CA}_3 \cdot SC \mid_{param,TXT}^{TXT'} \\ SC \mid_{param,TXT}^{TXT'} \to CR_1 \cdot SC \mid_{param,TXT}^{TXT'} \\ SC \mid_{param,TXT}^{TXT'} \to CC_1 \cdot SC \mid_{param,TXT}^{TXT'} \mid CC_2 \cdot SC \mid_{param,TXT}^{TXT'} \ldots \mid CC_8 \cdot SC \mid_{param,TXT}^{TXT'} \\ SC \mid_{param,TXT}^{TXT'} \to CA_1 \mid CA_2 \mid CA_3 \mid CR_1 \mid CC_1 \mid CC_2 \ldots \mid CC_8 , \end{gathered} \tag{17}$$

где *param* – множество входных значений алгоритмов $\{\mathrm{CA}_j\},\{\mathrm{CR}_k\},\{\mathrm{CC}_m\}$. Для сокращения записи в (17) не приводятся входные и выходные данные алгоритмов, определённые в (5–16).

Реализацией конструктора является множество сценариев $SC = \{SC_i\}$ для синтаксических и структурных изменений, которые сочетают алгоритмы операций обработки текстов с определенными параметрами. Сценарий – упорядоченная последовательность сконструированных алгоритмов. Элементы последовательности операций могут быть различными, идентичными или быть сценариям с различными параметрами выполнения.

Конструирование модификации текстов

Рассмотрим специализированный конструктор модификаций текстов для сокрытия заимствований:

$$C = \langle M, \Sigma, \Lambda \rangle \ {}_s\mapsto C_{AT} = \langle M_{AT}, \Sigma_{AT}, \Lambda_{AT} \rangle , \tag{18}$$

где $M_{AT}, \Sigma_{AT}, \Lambda_{AT}$ – носитель, сигнатура и множество утверждений ИОК соответственно.

Онтология конструктора C_{AT} представлена следующими утверждениями. Элементами носителя являются терминалы и нетерминалы. Терминалы включают конечное множество сконструированных сценариев $\Omega(C_{A,SC})$, а также данные для их применения, тексты и их элементы: символы электронного представления текстов и языковые конструкции: слова, предложения, абзацы [20]. Нетерминалы – вспомогательные элементы. Сигнатура Σ_{AT} состоит из множества операций $\Sigma_{AT} = \langle \Xi_{AT}, \Theta_{AT}, \Phi_{AT,} \{\to\}, \Psi_{AT} \rangle$, где Ξ_{AT} – отношения и одноименные операции, операции связывания и преобразования элементов носителя $\{\cdot, \nabla\} \subset \Xi_{AT}$. $\Theta_{AT} = \{\Rightarrow, \mid\Rightarrow, \mid\mid\Rightarrow\}$ – операции подстановки и вывода, $\{\to\}$ – отношение подстановки, Φ_{AT} –

отношения и одноименные операции над атрибутами, $\Psi_{AT} = \{\psi_i : \langle s_i, g_i \rangle\}$ – аналогичны операциям и отношениям C_{SC}.

Операция $\nabla(_{w_s} S, TXT)$ предполагает применение сценария S к тексту TXT.

Остальные операции – такие же, как в конструкторе сценариев C_{SC}.

Целью конструирования является получение базы текстов, которая состоит из пар «оригинал-модифицированная копия», что является набором тестовых значений (данных) для систем выявления заимствований (антиплагиата).

Ограничение конструктора модификации текста. Количество и последовательность примененных сценариев ограничивается их взаимоисключающим действием (например, смены латинского алфавита на кириллический и обратно, смена регистра).

Начальные условия модификации текстов:

- TXT – текст, содержащий более одного абзаца для возможности применения всех видов сценариев;
- σ – нетерминал, с которого начинается вывод;
- $\{D_i\}, i = \overline{1,k}$ – словари двух и более алфавитов (например, латинского и кириллического), содержащие пары символов из разных алфавитов с одинаковым начертанием, $k = \overline{2,N}$ – количество словарей в зависимости от количества разных алфавитов;
- набор входных значений сценария ($param$), задающийся внешним исполнителем: qi – количество вставок, qs – длина последовательности, qp – количество пустых абзацев для вставки, q – количество абзацев для удаления, sp – номер абзаца, с которого начинается удаление, $type$ – тип замены (однократная/все входжения), l_type – раскладка для модифицируемого символа, r – признак установки нижнего регистра, pos – позиция символа в тексте для модификации, c – символ для модификации, num_1, num_2 – номера абзацев для перестановки, w_num_1, w_num_2 – номера слов для перестановки, el_1, el_2 – слова для замены;

Условие завершения конструирования: форма не содержит нетерминалов.

Интерпретация предполагает определения модели исполнителя, которую представим в виде базовой алгоритмической структуры (БАС), рассмотренную в (2), дополнив алгоритмом выполнения операции применения сценария ($A_8 \mid {}^{TXT'}_{S,TXT} \leftarrow$ "∇").

На основе интерпретации получаем конструктивную систему сценариев.

Конкретизация конструктора модификации текстов:

$$C_{AT\ R} \mapsto C_{AT} = \langle M_4, \Sigma_4, \Lambda_4 \rangle, \tag{19}$$

где $M_4 \supset M_{AT} \cup T_2 \cup N_2$, $T_2 = \{SC_i, TXT\}$, $N_2 = \{\sigma\}$, $\Lambda_4 \supset \Lambda_3$.

Правило, приведенное ниже, позволяет выполнить модификацию текста, применив к нему один или несколько сценариев.

$$s = \left\langle \sigma \mid_{param,TXT}^{TXT'} \to \nabla(SC \mid_{param,TXT}^{TXT'}, TXT) \cdot \sigma, \sigma \to \nabla(SC \mid_{param,TXT}^{TXT'}, TXT) \right\rangle. \tag{20}$$

Реализация конструктора C_{AT} заключается в изменении элементов его носителя путем выполнения операции применения сценария с учетом утверждения ИОК:

$${}_A C_{AT\ R} \mapsto \Omega({}_A C_{AT}). \tag{21}$$

Таким образом, результатом реализации конструктора являются тексты, которые подвержены «механическим искажениям» – синтаксическим изменениям, которые не влияют или имеют незначительное влияние на смысл.

Результаты

Разработан конструктор формирования сценариев модификации текстов, моделирующий бесконечное разнообразие последовательности приемов маскировки заимствований существующих и потенциальных программных средств по механическому сокрытию плагиата.

В паре с конструктором модификации текстов имеем теоретическую основу формирования базы тестов для проверки способности демаскировки заимствований в программах выявления плагиата.

Входом теста является текст с множеством заимствований и этот же текст с маскировкой, выполненной в результате реализации конструктора модификации текста. Выходом – результаты работы исследуемой программы выявления плагиата с одним и другим текстом. Тест считается успешно пройденным, если результаты совпадают.

Авторами разработана программа, реализующая представленные в данной работе конструкторы.

Заключение

Применение конструктивного моделирования для разработки и совершенствования систем выявления плагиата позволяет на единой теоретической основе:

– формализовать сценарии модификации;

- создать платформу для моделирования новых изменений текста;
- автоматизировать построение сценариев и их применение для создания тестовой базы текстов для оценки возможностей демаскировки заимствований.

Разработанный формализм позволяет автоматизировать решение класса задач связанных с моделированием работы вредоносных программ маскировки заимствований.

Дальнейшая работа

Авторы занимаются разработкой собственных средств выявления заимствований в текстах, в том числе в компьютерных программах и математических формулах. Представленная модель является частью этой работы.

Благодарности

Статья публикуется с частичной поддеожкой ITHEA ISS (www.ithea.org) и ADUIS (www.aduis.com.ua).

The paper is published with partial support by the ITHEA ISS (www.ithea.org) and the ADUIS (www.aduis.com.ua).

Литература

[Bakhtiyari, 2014] Bakhtiyari K., Salehi H., Embi M.A., Shakiba M., Zavvari A., Shahbazi-Moghadam M., Ebrahim N.A., Mohammadjafari M. Ethical and unethical methods of plagiarism prevention in academic writing. International Education Studies; Vol. 7. No. 7. Canadian Center of Science and Education, 2014. pp. 52-62. ISSN 1913-9020 E-ISSN 1913-9039. doi:10.5539/ies.v7n7p52

[Ceska, 2009] Ceska Z., Fox C. The Influence of Text Pre-processing on Plagiarism Detection. In: Angelova, G and Bontcheva, K and Mitkov, R and Nicolov, N and Nikolov, N, (eds.) International Conference on Recent Advances in Natural Language Processing, Borovets, Bulgaria, 2009. Association for Computational Linguistics. pp. 55 - 59.

[Gulis, 2016] Gulis I., Chuda D., Petrik J. Plagiarism Detection in Students' Assignments Written in Natural Language. Proceedings of the International Conference on e-Learning, 2016. pp.141 – 144, http://elearning-conf.eu/docs/cp16/paper-21.pdf

[Mahmoodi, 2014] Mahmoodi M., Varnamkhasti M.M. Design a Persian Automated Plagiarism Detector (AMZPPD). International Journal of Engineering Trends and Technology (IJETT). Vol. 8 No. 8. 2014. pp. 465-467. ISSN 2231-5381. doi: 10.14445/22315381/IJETT-V8P280 http://www.ijettjournal.org/volume-8/number-8/IJETT-V8P280.pdf

[Mahmoud, 2017] Mahmoud H.A. Detecting Disguised Plagiarism. arXiv preprint arXiv:1711.02149 https://arxiv.org/ftp/arxiv/papers/1711/1711.02149.pdf

[Meuschke, 2013] Meuschke N., Gipp B. State-of-the-art in detecting academic plagiarism. International Journal for Educational Integrity Vol. 9 No. 1. 2013. pp. 50–71. ISSN 1833-2595

[Chowdhury, 2016] Chowdhury H.A., Bhattacharyya D.K. Plagiarism: Taxonomy, Tools and Detection Techniques Knowledge. Library and Information Networking, NACLIN 2016, ISBN: 978-93-82735-08-3 https://arxiv.org/abs/1801.06323

[Shynkarenko, 2014] Shynkarenko V.I., Ilman V.M. Constructive-Synthesizing Structures and Their Grammatical Interpretations. Part I. Generalized Formal Constructive-Synthesizing Structure. Cybernetics and Systems Analysis, Vol. 50, No 5. Springer, 2014. pp. 665 – 662. Part II. Refining Transformations. –Vol. 50, No 6, 2014. pp. 829 – 841. ISSN: 1060-0396 (Print) 1573-8337 (Online), doi: 10.1007/s10559-014-9655-z, https://link.springer.com/article/10.1007/s10559-014-9655-z, DOI 10.1007/s10559-014-9674-9, https://link.springer.com/article/10.1007/s10559-014-9674-9

[Shynkarenko, 2009] Shynkarenko V.I., Ilman V.M., Skalozub V.V. Structural models of algorithms in problems of applied programming. I. Formal algorithmic structures. Cybernetics and Systems Analysis, Vol. 45, No 3. Springer, 2009. pp 329-339. ISSN: 1060-0396 (Print) 1573-8337 (Online), doi: org/10.1007/s10559-009-9118-0 https://link.springer.com/article/10.1007/s10559-009-9118-0

[Vani, 2018] Vani K., Deepa G. Unmasking text plagiarism using syntactic-semantic based natural language processing techniques: Comparisons, analysis and challenges. Information Processing & Management, Vol. 54, No 3. Elsevier, 2018. pp. 408-432, ISSN 0306-4573, doi:org/10.1016/j.ipm.2018.01.008. http://www.sciencedirect.com/science/article/pii/S0306457317300547

[Xu, 2012] Qin Xu, Tan Tang, Lan-su Nie Research of text plagiarism detection process. Proceeding of the international conference on information engineering and applications (IEA). Vol 4. Springer, 2012. pp. 401-410. ISSN 1876-1100 ISSN 1876-1119 (el) DOI 10.1007/978-1-4471-4853-1

[Букач, 2016] Букач М.М., Жело Х.А. Академічна чесність як основа становлення майбутнього науковця. Наукові праці. Педагогіка, Т. 269, №. 257, 2016. с. 52 – 56. ISSN 2311-0287 (Print), http://pednauki.chdu.edu.ua/article/view/89746/85127

[Ересь, 2018] Ересь А.В. Лучанинов Д.В. Анализ современных систем обхода заимствований текста. Постулат. №1. ISSN 2414-4487 http://e-postulat.ru/index.php/Postulat/article/viewFile/1097/1124

[Михайловский, 2013] Михайловский Ю.Б. Система Anti-Plagiarism як інструмент запобігання плагіату в навчальній та науковій діяльності. Вісник Хмельницького національного університету.

Технічні науки, № 3, 2013. с. 162–168. ISSN 2307-5740, http://journals.khnu.km.ua/vestnik/pdf/tech/2013_3/33myh.pdf

[Федоровская, 2016] Федоровская Н.А. Пути снижения показателей уровня плагиата в работах студентов творческих направлений в высших учебных заведениях. Международный журнал прикладных и фундаментальных исследований, №. 11-3, 2016. с. 533-536.

[Шинкаренко, 2017] Шинкаренко В.И., Куропятник Е.С. Проблемы выявления плагиата и анализ инструментального программного обеспечения для их решения. Наука и прогресс транспорта, № 1 (67), 2017. с. 131—142. ISSN 2307–3489 (Print), ISSN 2307–6666 (Online), doi: 10.15802/stp2017/94034, http://stp.diit.edu.ua/article/view/94034/92573

[Шинкаренко, 2016] Шинкаренко В.И., Куропятник Е.С. Конструктивно-продукционная модель графового представления текста. Проблемы программирования, № 2 – 3, 2016. с. 63 – 72. ISSN 1727- 4907, http://dspace.nbuv.gov.ua/bitstream/handle/123456789/126391/07-Shinkarenko.pdf?sequence=1

[Шишкин, 2017] Шишкин Ю.Е. Исследование возможностей систем обнаружения заимствований в методологии больших данных. Фундаментальные основы инновационного развития науки и образования. Пенза: МЦНС «Наука и просвещение», 2017. с. 55-73. ISBN 978-5-907012-88-2

Информация об авторах

Виктор Шинкаренко *– д.т.н., профессор, зав. кафедрой «Компьютерные информационные технологии» Днепропетровского национального университета железнодорожного транспорта имени академика В. Лазаряна; ул. Лазаряна, 2, 49010, Днепр, Украина; e-mail: shinkarenko_vi@ua.fm*

Основные области научных исследований: конструктивно-продукционное моделирование, качество программного обеспечения, искусственный интеллект.

Елена Куропятник *– аспирант Днепропетровского национального университета железнодорожного транспорта имени академика В. Лазаряна; ул. Лазаряна, 2, 49010, Днепр, Украина; e-mail: olena.kuropiatnyk@gmail.com*

Основные области научных исследований: методы и средства выявления заимствований, инженерия программного обеспечения

Creation tests for checking plagiarism detection programs' ability of unmasking borrowings

Viktor Shynkarenko, Olena Kutopiatnyk

Abstract: The research is aimed at ensuring timely response to the newly discovered opportunities for mechanical masking of text borrowings. A non-standard approach to software testing is proposed, in which the test output data is generated by the program under test. The formalization of means to verify the ability to unmask borrowing in plagiarism detection programs in the form of constructors of texts modification scenario and process of their application were developed. The basis of the methodology of research is the mathematical apparatus of constructive-synthesizing modeling. The fundamental principle of this approach is that all the entities and processes of the real and virtual world are considered as constructs and constructive processes with the corresponding attributes. The usage of this approach makes it possible to automate the processes of preparing texts containing disguises of borrowing, and also allows further development of scenarios for modification.

***Keywords**: constructive-synthesizing modeling, constructor, text borrowing, disguise of borrowing, masking scenario, test.*