Построение решателя уравнений в словах методом развёртки/свёртки графа решений

Александра Спиридонова
МГТУ имени Н. Э. Баумана

IV совместное рабочее совещание ИПС имени А.К. Айламазяна РАН и МГТУ имени Н.Э. Баумана по функциональному языку программирования Рефал

08 июня 2021

Постановка задачи

Реализовать решатель уравнений в словах, использующий алгоритм Матиясевича, основанный на свертке/развертке графа решений, и некоторые простые преобразования над уравнениями в словах, а также упрощение систем регулярно-упорядоченных уравнений

Язык уравнений в словах

Σ – алфавит – конечный набор символов

X – конечный набор строковых переменных

ω – слово – конечный набор символов алфавита

Σ^* – множество слов в алфавите

$T_1 = T_2$ – уравнение в словах,

где $(T_1, T_2) \in (\Sigma \cup X)^* \times (\Sigma \cup X)^*$

причем $\Sigma \cap X = \emptyset$

σ – решение уравнения в словах – морфизм вида $\sigma : (\Sigma \cup X)^* \to \Sigma^*$

удовлетворяющий условиям:

1. $\sigma(T_1) = \sigma(T_2)$
2. $\sigma(a) = a, a \in \Sigma$

$Sol(T_1 = T_2)$ – набор всех решений уравнения

Алгоритм Матиясевича

Стандартный вариант

1. Если уравнение имеет вид $xT_1 = yT_2$, $x, y \in X$, то либо $\exists z : x = yz$, либо $\exists z : y = xz$, либо $x = y$
2. Если уравнение имеет вид $xT_1 = aT_2$ $x \in X, a \in \Sigma$, то либо $x = \varepsilon$, либо $x = az$
3. Если уравнение имеет вид $T = \varepsilon$, то оно имеет решения тогда и только тогда, когда $T \in X^*$ и значения всех переменных, входящих в T, пусты

Алгоритм Матиясевича

Финитный вариант

1. Если уравнение имеет вид $xT_1 = yT_2$, $x, y \in X$, то
   либо $\exists z, t : y = xtz, |t| = 1$, либо $\exists z, t : x = ytz, |t| = 1$, либо $x = y$
2. Если уравнение имеет вид $xT_1 = aT_2$ $x \in X, a \in \Sigma$, то либо $x = \varepsilon$,
   либо $x = az$
3. Если уравнение имеет вид $T = \varepsilon$, то оно имеет решения тогда и только
   тогда, когда $T \in X^*$ и значения всех переменных, входящих в T, пусты
4. Если уравнение имеет вид $tT_1 = sT_2, t, s \in X, |t| = 1, |s| = 1$, то $t = s$
5. Если уравнение имеет вид $tT_1 = aT_2, t \in X, |t| = 1, a \in \Sigma$, то $t = a$
6. Если уравнение имеет вид $tT_1 = xT_2, t, x \in X, |t| = 1$, то $\exists z : x = tz$

Сокращение совпадающих термов

1. Если уравнение имеет вид $xT_1 = xT_2$, $x \in X$, то производим сокращение $xT_1 = xT_2 \to T_1 = T_2$
2. Если уравнение имеет вид $aT_1 = aT_2$, $a \in \Sigma$, то производим сокращение $aT_1 = aT_2 \to T_1 = T_2$

Алгоритм Матиясевича

Условия завершения развертки ветви дерева решений

1. Уравнение в последнем узле ветви является переименовкой некоторого уравнения в одном из предыдущих узлов этой ветви.
2. Уравнение в последнем узле ветви имеет вид $T = T$.
   В этом случае считается, что решение исходного уравнения найдено.
3. К уравнению в последнем узле ветви невозможно применить ни одно из правил переписывания, указанных выше. В этом случае считается, что ветвь развертки привела к противоречию.

Преобразования над уравнениями в словах

1. Анализ длин частей уравнения и выявление противоречий
2. Конечные подстановки
3. Разбиение уравнения по равносоставленности
4. Последовательное упрощение и применение уравнений друг к другу

Уравнения, разбиваемые по равносоставленности

Уравнение в словах $T_1 = T_2$ является разбиваемым по равносоставленности, если его можно представить в виде $Q_1R_1 = Q_2R_2$ таком, что Q_1 и Q_2 являются равносоставленными.

Q_1 и Q_2, $Q_1, Q_2 \in (\Sigma \cup X)^+$ являются равносоставленными, если:

1. $\forall x_i \in X$ количество вхождений x_i в Q_1 совпадает с количеством вхождений в Q_2
2. Длина Q_1 и Q_2 одинаковая по количеству термов

Пример: $AxyB = xCAz$ Ax и xC - равносоставленные

$x, y, z \in X, A, B, C \in \Sigma$

Противоречия по длинам и конечные подстановки

$\sum$ – алфавит – конечный набор символов

X – конечный набор строковых переменных

$T_1 = T_2$ – уравнение в словах

$X_T = \{x_i\}, i = 1..k$ – множество переменных, входящих в T_1 и T_2

n_i – количество вхождений x_i в T_1

N_i – количество вхождений x_i в T_2

S_1 – количество совместных вхождений в T_1 констант и переменных типа буква

S_2 – количество совместных вхождений в T_2 констант и переменных типа буква

1. $\forall i, i = 1..k, n_i \geq N_i \quad S_1 > S_2$ => противоречие по длинам

   Пример: $xyAxB = Byx$

2. $\forall i, i = 1..k, N_i \geq n_i \quad S_2 > S_1$ => противоречие по длинам

   Пример: $xyA = yxBAy$

Противоречия по длинам и конечные подстановки

$\sum$ – алфавит – конечный набор символов

X – конечный набор строковых переменных

$T_1 = T_2$ – уравнение в словах

$X_T = \{x_i\}, i = 1..k$ – множество переменных, входящих в T_1 и T_2

n_i – количество вхождений x_i в T_1

N_i – количество вхождений x_i в T_2

S_1 – количество совместных вхождений в T_1 констант и переменных типа буква

S_2 – количество совместных вхождений в T_2 констант и переменных типа буква

3. $\exists j, j = 1..k, n_j > N_j \, \forall i, i \neq j, n_i = N_i \;\; S_1 \geq S_2$ => вычисляем $t = \frac{S_2 - S_1}{n_j - N_j}$
$t \in \mathbb{Z}$ => заменяем x_j на t буквенных переменных, иначе противоречие по длинам

Пример: $xy = ABx$ => $xtt = ABx, |t| = 1, t \in X$

4. $\exists j, j = 1..k, N_j > n_j \, \forall i, i \neq j, n_i = N_i \;\; S_2 \geq S_1$ => вычисляем $t = \frac{S_1 - S_2}{N_j - n_j}$
$t \in \mathbb{Z}$ => заменяем x_j на t буквенных переменных, иначе противоречие по длинам

Противоречия по длинам и конечные подстановки

$\Sigma_T = \{c_i\}, i = 1..r$ – множество констант, входящих в T_1 и T_2

m_i – количество вхождений c_i в T_1

M_i – количество вхождений c_i в T_2

R_1 – количество в T_1 переменных типа буква

R_2 – количество в T_2 переменных типа буква

5. Анализ по кратности констант, если $\forall i, i = 1..k, n_i = N_i$ $S_1 = S_2$

Если $\sum_{i=1}^{r} max(0, m_i - M_i) > R_2$ => противоречие по мультимножествам

Если $\sum_{i=1}^{r} max(0, M_i - m_i) > R_1$ => противоречие по мультимножествам

Пример: $xAAB = BCtx, x, t \in X, |t| = 1$

$A, B, C \in \Sigma$

Регулярно-упорядоченные уравнения

Уравнение в словах $T_1 = T_2$ является регулярно-упорядоченным,

если при применении преобразования ξ такого, что:

1. $\forall a \in \Sigma \xi(a) = \varepsilon$
2. $\forall x \in X \xi(x) = x$
3. $\forall t \in X, |t| = 1, \xi(t) = \varepsilon$

$\xi(T_1) = \xi(T_2)$, т. е. $\xi(T_1)$ текстуально совпадает с $\xi(T_2)$

Пример: $AxyB = xABByA$

$x, y \in X, A, B \in \Sigma$

Малый подграф графа развертки

Малый подграф - такое максимальное поддерево дерева развертки, что все ребра в нем помечены подстановками в одну и ту же переменную $x \in X$

В любом малом подграфе регулярно-упорядоченного уравнения всегда существует максимум один цикл

Если в малом подграфе есть цикл, то выход в TRUE-узел в нем только один и он помечен подстановкой $x \longrightarrow \varepsilon$

Алгоритм упрощения малого подграфа

G_{i_j} $i = 1..n, j = 1..k$ – малый подграф для переменной x_i графа развертки G, где

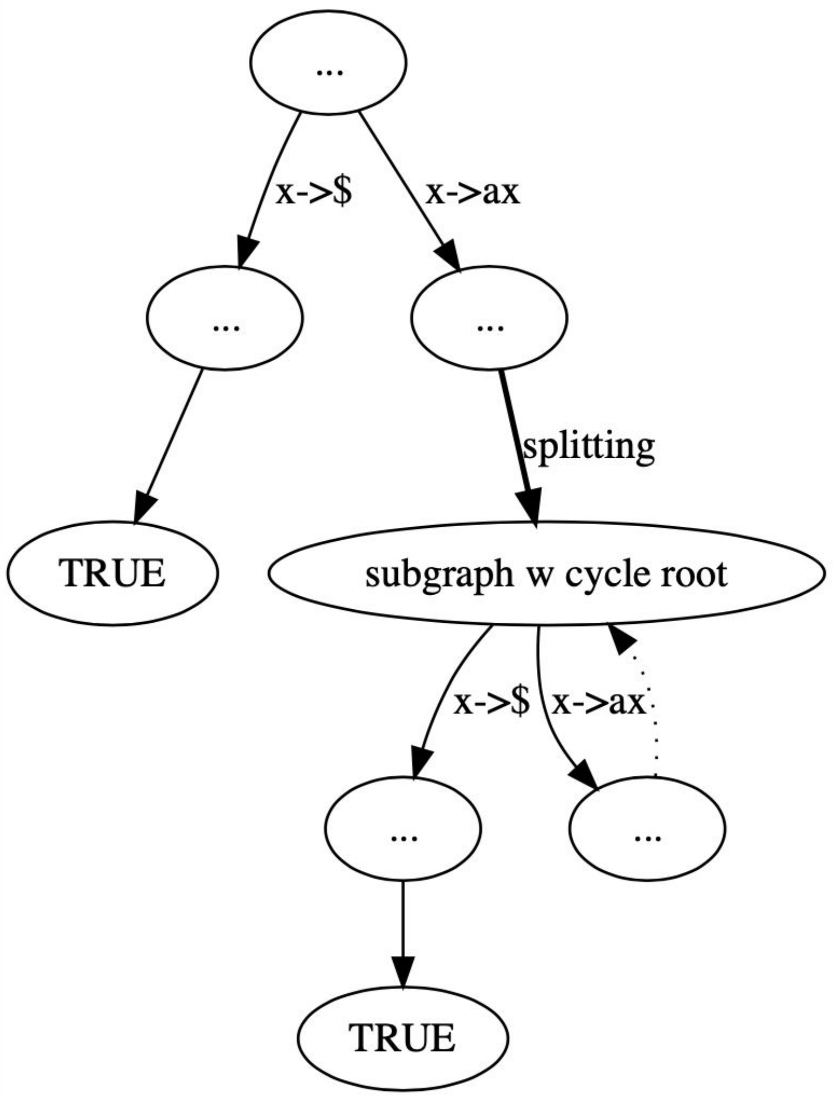
n – количество переменных, в которые содержатся подстановки в G,

k – количество малых подграфов для переменной x_i

В результате упрощения малого подграфа G_{i_j} получаем язык, описываемый в виде $E = E_1 \vee E_2 ... \vee E_n$, где $\forall i, i = 1..n$ либо $E_i : R_i x = x Q_i, R_i, Q_i \in \Sigma^+$, либо $E_i : x = S_i, S_i \in \Sigma^*$

1. Строим характеристическое уравнение для подграфа-цикла
2. Применяем правила свертки для подграфа-цикла
3. Если для подграфа-цикла остались вершины-предки, малого подграфа, не описанные уравнением, то получить упрощенное представление не удалось, иначе возвращаем уравнение
4. При отсутствии в графе подграфа-цикла возвращаем дизъюнкцию условий

Пример упрощения малого подграфа



1. $Ax = xA$ – характеристическое уравнение для подграфа-цикла
2. Применяя правила свертки, получаем $Ax = xA$ – характеристическое уравнение, описывающее малый подграф

Алгоритм упрощения графа развертки регулярно-упоряченной системы уравнений

G – исходный граф развертки

1. Выбираем одну переменную x_i, для которой в разбираемом графе существуют малые подграфы, содержащие *TRUE*-узлы. Описываем при помощи алгоритма упрощения малого подграфа для x_i множество языков. Получаем n языков $\left\{L_j^i\right\}, j = 1..n$,
   описываемых дизъюнкциями: $L_j^i = E_{j,1}^i \vee \ldots E_{j,k_j}^i$.

   Переобозначим каждый дизъюнкт каждого языка: $L^i = W_1^i \vee \ldots \vee W_{M_i}^i$

Алгоритм упрощения графа развертки регулярно-упоряченной системы уравнений

G – исходный граф развертки

2. Получаем M новых графов вида $G_{W_j^i}, \forall j W_j^i$ заменяя в G те подграфы, в которые W_j^i входит как дизъюнкт на *TRUE*-узлы, а в которые не входит - на *FALSE*-узлы
3. Для каждого из графов повторяем пункты 1-2.
4. Продолжаем алгоритм, пока все подграфы на превратятся в *TRUE*-узлы.

   В итоге получится описание графа в виде дизъюнкций имен получившихся подграфов.

Обработка упрощенного представления системы регулярно-упорядоченных уравнений

G – система уравнений в узле дерева развертки

E – система регулярно-упорядоченных уравнений, требующая упрощения

F – множество уравнений, не относящихся к какому-либо классу

$$G = E \wedge F$$

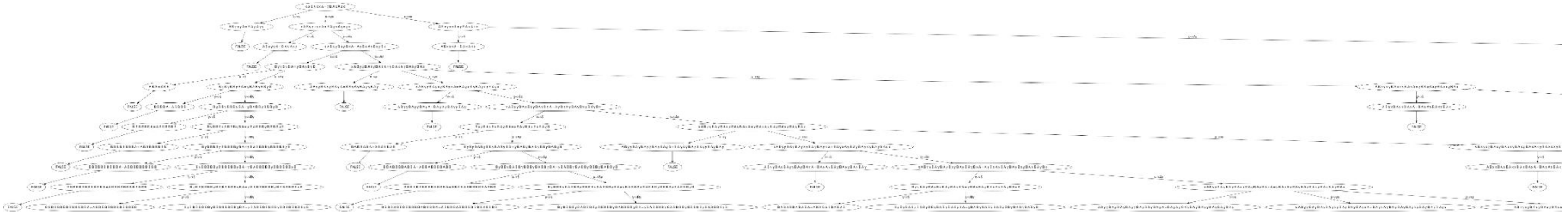
После упрощения: $E = E_1 \vee E_2 ... \vee E_n$

Получаем n новых потомков: $G_i = E_i \wedge F, i = 1..n$

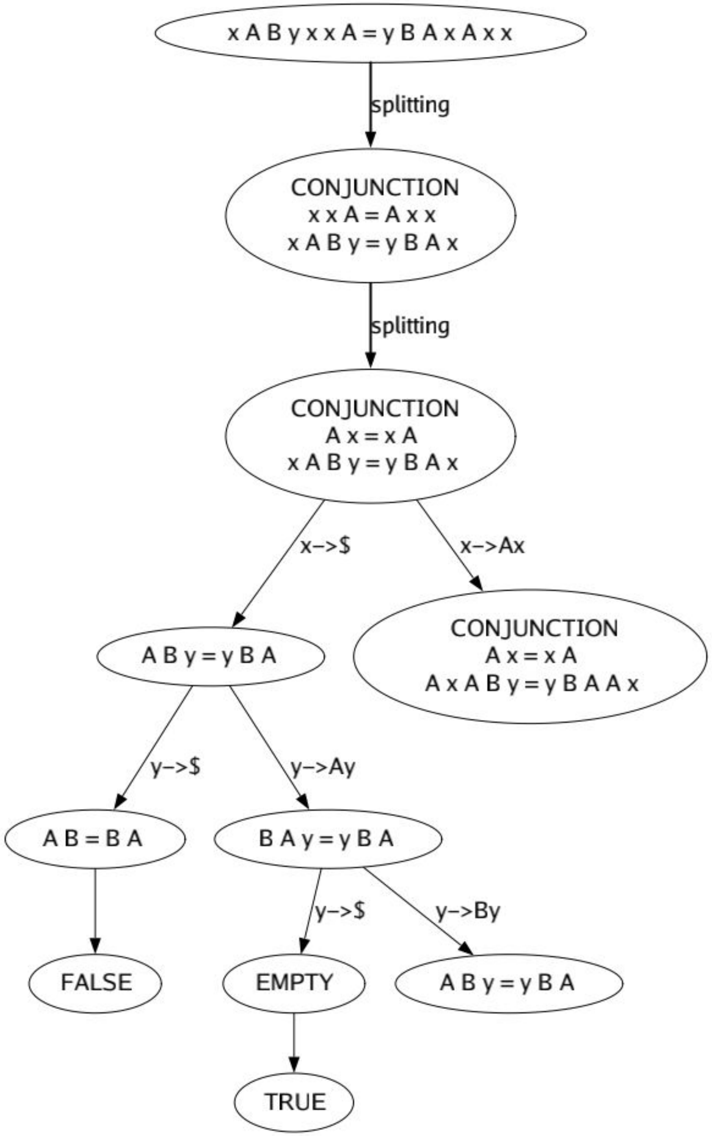
Пример графа развертки уравнения без использования упрощения

$$xAByxxA = yBAxAxx$$

$$A, B \in \Sigma \quad x, y \in X$$



Пример графа развертки уравнения с использованием упрощения



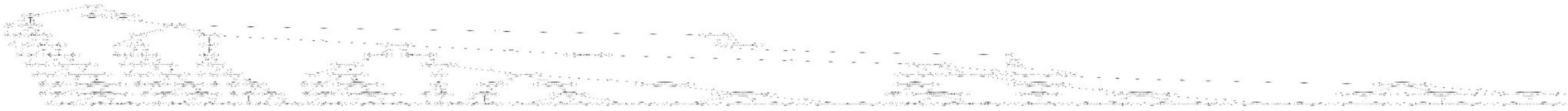
$$xAByxxA = yBAxAxx$$

$$A, B \in \Sigma \quad x, y \in X$$

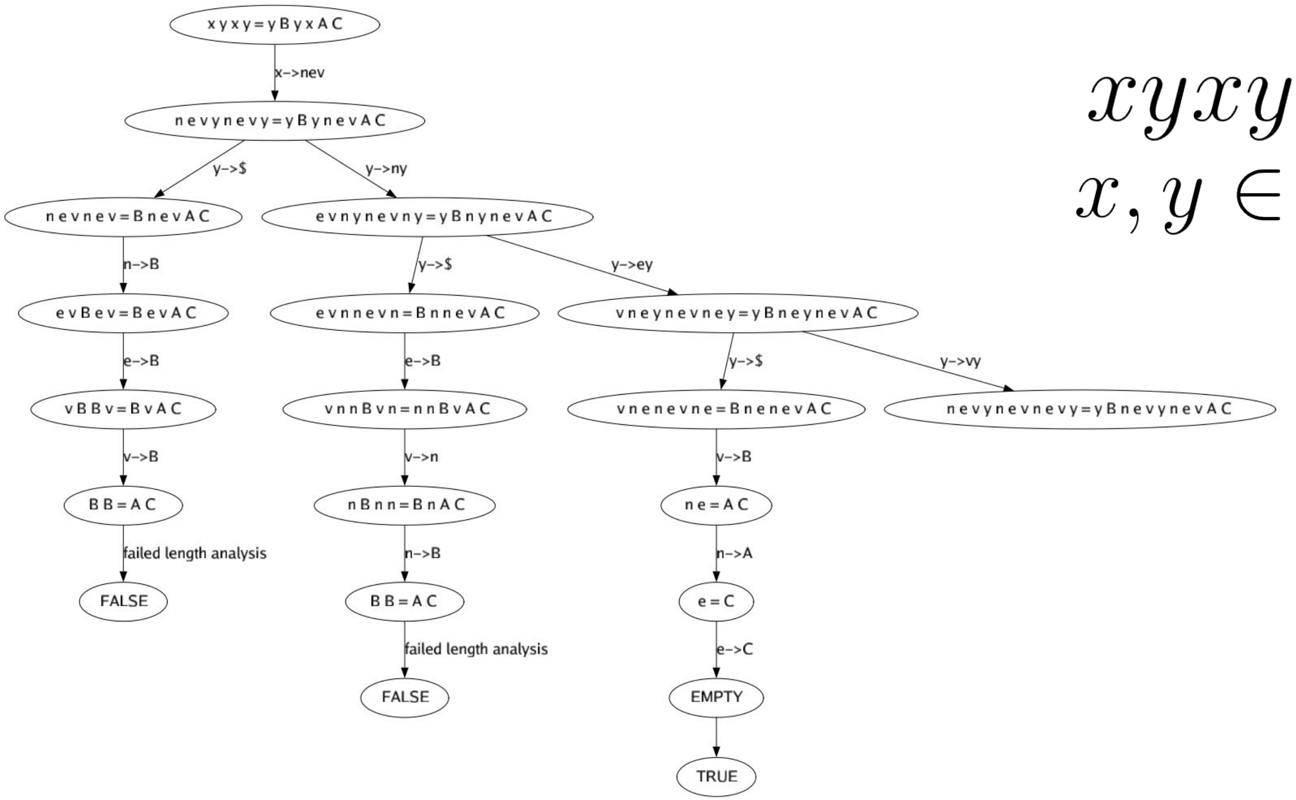
Пример графа развертки уравнения без использования анализа длин частей уравнения

$$xyxy = yByxAC$$

$$x, y \in X \quad A, B, C \in \Sigma$$



Пример графа развертки уравнения с использованием анализа длин частей уравнения



$$xyxy = yByxAC$$

$$x, y \in X \;\; A, B, C \in \Sigma$$

Результаты

Использование классического метода Матиясевича совместно с алгоритмами упрощения уравнений позволяет для некоторых уравнений получать вердикт о наличии решения существенно быстрее и не уходить в бесконечную развертку.

Если при построении графа развертки уравнения в одном из узлов система регулярно-упорядоченных уравнений не имеет решений, то итоговый граф существенно упрощается.

Финитный вариант алгоритма Матиясевича в связке с анализом уравнений в словах на длины и применением конечных подстановок позволяет отсекать ветви графа, которые точно не приведут к решению.