ТЕХНИЧЕСКАЯ КИБЕРНЕТИКА
№ 3 • 1982

УДК 621.397.3

СИНТАКСИЧЕСКИЙ АНАЛИЗ ИЗОБРАЖЕНИЙ В ПРОЦЕССЕ ПОСТРОЧНОГО СКАНИРОВАНИЯ

МАЦЕЛЛО В. В., ШЛЕЗИНГЕР М. И.

Основное направление в теории структурного анализа изображений в настоящее время заключается в исследовании таких видоизменений формальных грамматик, которые были бы удобны для синтаксического анализа изображений с учетом его двумерного характера. В [1] рассмотрены некоторые из них и предложена формально-грамматическая конструкция для задания множеств изображений, названная двумерной грамматикой. Двумерные грамматики являются весьма общей конструкцией и поэтому алгоритмы синтаксического анализа в общем виде требуют зачастую большого объема памяти [2]. В связи с этим актуально исследование определенных подклассов двумерных грамматик, допускающих менее требовательные в вычислительном отношении алгоритмы синтаксического разбора. Один из таких подклассов рассмотрен в данной работе.

1. Основные определения. Циклом будем называть конечное множество T, на котором задано отношение $F \subset T \times T$, называемое отношением следования, такое, что для каждого $t_1 \in T$ существует единственное $t_2 \in T$, такое, что $(t_1, t_2) \in F$; и, кроме того, для каждого $t_2 \in T$ существует единственное $t_1 \in T$, такое, что $(t_1, t_2) \in F$. Если $(t_1, t_2) \in F$, то будем говорить, что элемент t_2 следует за t_1 или что элемент t_1 предшествует t_2, и обозначать этот факт $t_2 = f(t_1)$. Будем говорить также, что t_1 и t_2 — соседние, если $(t_1, t_2) \in F$ либо $(t_2, t_1) \in F$.

Контуром будем называть функцию v: $T \to V$, где V — множество, называемое алфавитом направлений. Алфавит направлений состоит из четырех направлений: вниз, вправо, вверх, влево, обозначаемых в сокращенном виде Н, П, В, Л.

Описанием контура называется функция $\hat{s}$: $T \to S$, где S — множество, называемое алфавитом структурных элементов.

Пусть $G \subset S \times V \times S$. Контур v называется допустимым, если существует его описание $\hat{s}$, такое, что для любого $t \in T$ выполняется $(s(t), v(t), s(f(t))) \in G$.

Например, для описания изображений черных прямоугольников на белом фоне можно использовать алфавит структурных элементов $S = \{s_1, s_2, s_3, s_4\}$, где s_1 — элемент левой стороны прямоугольника, s_2 — элемент верхней стороны, s_3 — элемент правой стороны, s_4 — элемент нижней стороны.

Тогда G включает в себя следующие тройки: $(s_1, \text{В}, s_1)$, $(s_1, \text{П}, s_2)$, $(s_2, \text{П}, s_2)$, $(s_2, \text{Н}, s_3)$, $(s_3, \text{Н}, s_3)$, $(s_3, \text{Л}, s_4)$, $(s_4, \text{Л}, s_4)$, $(s_4, \text{В}, s_1)$.

Будем решать задачу распознавания допустимости контура при определенных ограничениях на алгоритм решения, следующих из некоторых разумных практических требований. Чтобы сформулировать эти ограничения, установим соответствие между изображением и контуром.

2. Представление изображений контурами. Поле зрения Q — это множество пар (i, j), где $i = 1, 2, \ldots, n$, а $j = 1, 2, \ldots, m$. Пара $(i, j) \in Q$ называется клеткой. Изображение $\hat{x}$ — это функция $Q \to X$, где $X = \{$черное, бе-

лое}, или в более сокращенной записи $X=\{Ч, Б\}$. В дальнейшем будем предполагать, что $x(i,j)=$«белое» для всех клеток $(1,j)$, (n,j), $j=1, 2, \ldots, m$, и для всех клеток $(i,1)$, (i,m), $i=1,2,\ldots,n$.

Гранью будем называть пару клеток вида $((i,j), (i+1,j))$ или вида $((i,j), (i,j+1))$, если только одна из клеток, входящих в пару, белая, а вторая — черная. Множество граней обозначим через T.

Каждой грани поставим в соответствие две упорядоченные пары чисел — ее начало и конец и символ $v\in\{Н, П, В, Л\}$ — направление грани. Это соответствие установим следующим образом. Если грань $t=((i,j), (i+1,j))$ и $x(i,j)=$«черное», а $x(i+1,j)=$«белое», то начало $(t)=(i+1,j)$, конец $(t)=(i+1,j+1)$, а $v(t)=$П (вправо) (см. рис. 1, *а*).

Для грани того же вида, если $x(i,j)=$«белое», а $x(i+1,j)=$«черное», то начало $(t)=(i+1,j+1)$, конец $(t)=(i+1,j)$, а $v(t)=$Л (влево) (см. рис. 1, *б*).

Если грань $t=((i,j), (i,j+1))$, $x(i,j)=$«черное», $x(i,j+1)=$«белое», то начало $(t)=(i+1,j+1)$, конец $(t)=(i,j+1)$, а $v(t)=$Н (см. рис. 1, *в*).

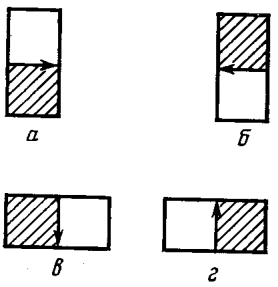

Рис. 1

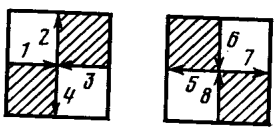

Рис. 2

Если же $x(i,j)=$«белое», а $x(i,j+1)=$«черное», то начало $(t)=(i,j+1)$, конец $(t)=(i+1,j+1)$, а $v(t)=$В.

Если какая-то пара (i,j) чисел является концом единственной грани t_1 и началом единственной грани t_2, то будем считать, что $(t_1,t_2)\in F$, $t_2=f(t_1)$.

Если какая-то пара чисел (i,j) является концом двух граней и началом двух граней, то имеют место лишь две возможные ситуации, представленные на рис. 2. В этом случае отношение следования определим так:

$$t_2=f(t_1),\quad t_4=f(t_3),\quad t_5=f(t_8),\quad t_7=f(t_6).$$

Введенное таким образом множество граней с заданным на нем отношением следования является циклом в том смысле, как это было введено в п.1. Множество граней с указанием направления каждой грани образуют, очевидно, контур. Таким образом, мы установили соответствие между изображением и контуром.

Подобное представление изображения в виде контура похоже на цепное кодирование Фримена [3]. Однако предлагаемый метод имеет существенное отличие. Цепной код предназначен для кодирования всего изображения в целом и последующего синтаксического анализа всей полученной цепочки, в то время как предлагаемый метод, как будет показано ниже, не требует одновременного хранения информации о всем контуре. Кроме того, предлагаемый ниже алгоритм проверки допустимости контура значительно проще известных алгоритмов грамматического разбора.

3. Формулировка задачи. Задача, решаемая в данной работе, заключается в том, чтобы для заданного изображения указать, является ли допустимым контур, соответствующий этому изображению.

На алгоритм решения этой задачи наложим ограничения, связанные с необходимостью анализировать изображения больших размеров.

Рассмотрим алгоритмы только следующего рекуррентного типа. Алгоритм работает по тактам. Исходя из специфики задачи, будем приписывать каждому такту два номера: i и j. При этом после такта (i, j) выполняется такт $(i, j+1)$, если только $j<m-1$, в противном случае выполняется такт $(i+1, 1)$, если только $i<n-1$.

Пусть к началу (i, j)-го такта состояние памяти алгоритма было D_{ij}'. На (i, j) такте на вход алгоритма подаются сведения о черноте четырех клеток, а именно величины $x(i, j)$, $x(i, j+1)$, $x(i+1, j)$, $x(i+1, j+1)$. Алгоритм меняет состояние своей памяти на D_{ij}'', т. е. производит вычисление функции $D_{ij}''=f(D_{ij}', x(i, j), x(i, j+1), x(i+1, j), x(i+1, j+1))$.

Наша задача заключается в том, чтобы определить функцию таким образом, чтобы D_{nm}'' принимало некоторое стандартное значение тогда и только тогда, если контур распознаваемого изображения не является допустимым. При этом мы хотим, чтобы объем памяти, необходимый для запоминания состояния D_{ij}', не зависел от количества строк изображения, т. е. чтобы алгоритм потенциально был бы способен распознавать и сколь угодно длинные изображения.

4. Алгоритм решения задачи. Алгоритм заключается в составлении множества C списков c. Каждый из списков имеет следующий формат. В начале списка находится упорядоченная пара чисел, являющихся адресами (смысл этих адресов будет разъяснен позже). Затем следует перечисление некоторых упорядоченных пар структурных элементов.

Адреса, находящиеся в начале списка, принимают значения $2, \ldots, m$ и \$, где \$ — символ, не равный никакому числу, а m — длина строки изображения.

Рассмотрим состояние алгоритма к началу (i, j)-го такта. В течение всех предыдущих тактов на вход его было подано i строк изображения и j клеток $(i+1)$-й строки. Совокупность всех этих клеток будем называть просмотренной частью изображения. Будем говорить, что некоторая грань $t=(q_1, q_2)$, $q_1 \in Q$, $q_2 \in Q$, имеется на просмотренной части изображения, если и клетка q_1, и клетка q_2 принадлежат просмотренной части.

Алгоритм должен работать таким образом, что если в некотором списке, озаглавленном парой адресов (j', j''), имеется пара (s', s'') структурных элементов, то это означает, что на просмотренной части изображения существует последовательность граней $t_1, t_2, \ldots, t_q$, такая, что:

1. $(t_p, t_{p+1}) \in F$, $p=1, 2, \ldots, q-1$.
2. Если $j' \neq \$$, то t_1 — вертикальная грань и начало $(t_1)=(i', j')$, где $i'=i+2$, если $j'<j$, и $i'=i+1$, если $j' \geqslant j$.
3. Если $j'=\$$, то t_1 — горизонтальная грань и начало $(t_1)=(i+1, j+1)$.
4. Аналогично, если $j'' \neq \$$, то t_q — вертикальная грань, конец которой равен (i', j''), где $i'=i+2$, если $j''<j$, и $i'=i+1$, если $j'' \geqslant j$.
5. Если $j''=\$$, то t_q — горизонтальная грань, конец которой равен $(i+1, j+1)$.
6. Существует последовательность $s_1, s_2, \ldots, s_{q+1}$, где s_p, $p=1, 2, \ldots, q+1$, — структурные элементы, такая, что для любого $p=1, 2, \ldots, q$ $(s_p, v(t_p), s_{p+1}) \in G$.

Это свойство алгоритма иллюстрируется следующим примером. Будем считать допустимыми изображения, контур которых похож на контур буквы П, но толщина и относительные размеры частей изображения могут быть произвольными. Выберем алфавит структурных элементов $S=\{s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8\}$, где каждый элемент соответствует одной стороне многоугольника, составляющего контур допустимого изображения. Множество G будет содержать следующие тройки:

$$(s_1, \text{В}, s_1), (s_1, \text{П}, s_2), (s_2, \text{П}, s_2), (s_2, \text{Н}, s_3), (s_3, \text{Н}, s_3), (s_3, \text{Л}, s_4),$$
$$(s_4, \text{Л}, s_4), (s_4, \text{В}, s_5), (s_5, \text{В}, s_5), (s_5, \text{Л}, s_6), (s_6, \text{Л}, s_6), (s_6, \text{Н}, s_7),$$
$$(s_7, \text{Н}, s_7), (s_7, \text{Л}, s_8), (s_8, \text{Л}, s_8), (s_8, \text{В}, s_1).$$

На рис. 3 представлено анализируемое изображение в две штриховые линии, соответствующие просмотренным частям изображения к началу (4,11)- и (9,14)-го тактов.

К началу (4,11)-го такта в памяти алгоритма должны находиться два списка. Первый имеет заголовок (7,5) и содержит две пары структурных элементов (s_7, s_1) и (s_3, s_5). Другой список имеет заголовок ($, 10) и содержит пары ($s_4$, s_5) и (s_8, s_1). К началу (9,14)-го такта в памяти алгоритма должен находиться один список, озаглавленный (13,5) и содержащий пару структурных элементов (s_3, s_1).

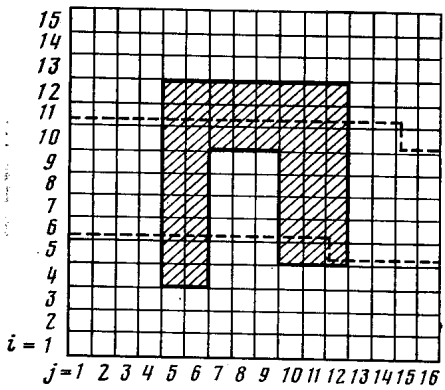

Рис. 3

Как следует из формального описания свойств списков и приведенных примеров, алгоритм на каждом шаге хранит сведения о всех односвязных кусках контуров, которые находятся на просмотренной части изображения, а конец и начало которых находятся на границе просмотренной части. При этом каждый кусок характеризуется одним списком, в заголовке которого записаны координаты начала и конца соответствующего куска. Поскольку известно, что начало и конец лежат на границе просмотренной части, то эти координаты характеризуются не двумя числами, а одним.

Работа алгоритма заключается в том, что, вообще говоря, на каждом шаге происходит исключение некоторого списка и введение нового списка. Если вводимый список оказывается пустым, то это значит, что алгоритм переходит в некоторое исключительное состояние, которое обозначает, что рассматриваемое изображение не является допустимым.

Пусть на (i, j)-м шаге алгоритма на его вход была подана четверка чернот $x(i, j)$, $x(i, j+1)$, $x(i+1, j)$, $x(i+1, j+1)$, называемая в дальнейшем текущим фрагментом. Из указанной четверки лишь клетка $(i+1, j+1)$ не принадлежит части изображения, просмотренной к началу (i, j)-го такта. На данном фрагменте возможно обнаружение граней, которые не были обнаружены на предыдущих тактах. Эти грани $((i+1, j), (i+1, j+1))$ и $((i, j+1), (i+1, j+1))$. Здесь возможны следующие ситуации.

1. На текущем фрагменте нет никаких граней. В этом случае алгоритм не меняет множество хранящихся списков.

2. На текущем фрагменте имеются две новые грани, которые являются соседними по отношению друг к другу. В этом случае в множество списков добавляется новый список.

3. На текущем фрагменте имеется одна новая грань, соседняя с гранью на просмотренной части. В этом случае происходит изменение одного из списков, так называемое наращивание.

4. На текущем фрагменте имеются две грани, принадлежащие к просмотренной части, соседство которых обнаружилось лишь на текущем такте. В этом случае если эти две грани до начала текущего такта принадлежали двум разным связным кускам контура, то происходит замена соответствующих кусков одним, т. е. операция, называемая «склеиванием». Если же эти две грани уже до текущего такта принадлежали одному куску, то происходит операция «замыкание» и соответствующий список исключается из памяти.

При некоторых фрагментах может иметь место не единственная из указанных ситуаций. В этом случае алгоритм производит несколько действий, например «склеивание» и «новый список».

Рассмотрим отдельно каждое из указанных действий алгоритма. При описании алгоритма нам придется рассматривать координаты начал или

концов некоторых граней. Однако подобно тому, как было отмечено при определении начал и концов односвязных кусков контуров, эти координаты таковы, что они находятся только на границе просмотренной части изображения. Поэтому координаты начала или конца мы будем обозначать одним числом, а не двумя.

«Новый список». Это действие происходит, когда на текущем фрагменте имеются грани, каждая из которых не является соседней ни с какой из граней на просмотренной части. Такая ситуация возможна лишь на трех фрагментах, приведенных на рис. 4.

В случае фрагмента *а* в память алгоритма вводится список с заголовком ($, $j+1$); пара (s', s'') вводится в этот список тогда и только тогда, когда существует тройка (s', s^*, s''), такая, что $(s', Л, s^*) \in G$ и $(s^*, В, s'') \in G$. В случае фрагментов *б* и *в* в память алгоритма вводится список с заголовком ($j+1$, $), содержащий пары (s', s''), такие, что существует s^*, обладающий свойством $(s', Н, s^*) \in G$, и $(s^*, П, s'') \in G$. Естественно, эти списки могут быть заготовлены заранее до начала работы алгоритма.

«Наращивание». «Наращивание» происходит, если на текущем фрагменте появилась грань, соседняя с гранью на просмотренной части изображения. При этом следует различать две ситуации, отличающиеся тем, какая из этих двух граней является следующей, а какая предшествующей. В зависимости от этих двух ситуаций мы будем различать «наращивание с конца» и «наращивание с начала».

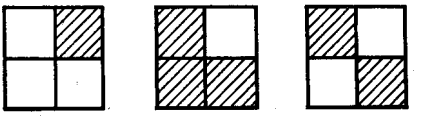

Рис. 4

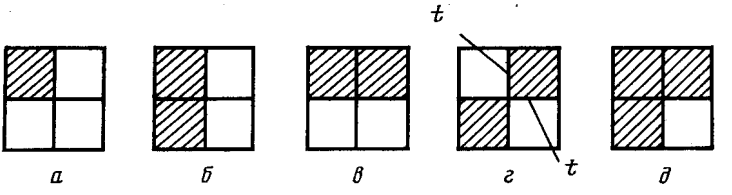

Рис. 5

«Наращивание с начала» имеет место при текущих фрагментах, представленных на рис. 5. На каждом из этих фрагментов появляется одна грань (обозначим ее t), за которой следует грань на просмотренной части изображения. Отметим, что на фрагменте рис. 5, *г* появляются две новые грани t и t', но лишь за одной из них, а именно за гранью t, следует грань, принадлежащая просмотренной части. Пусть начало $(t)=j_1$, конец $(t)=j_2$, $v(t)=v$. В этом случае в множестве списков следует отыскать список c^* с заголовком (j', j''), где $j'=j_2$. Так как по определению контур не имеет разветвлений, существует единственный список c^* с таким заголовком.

Затем следует ввести новый список с заголовком (j_1, j''), содержащий пару (s_1, s_2) в том и только в том случае, если существует $s^* \in S$, такое, что $(s_1, v, s^*) \in G$ и $(s^*, s_2) \in c^*$. После этого список c^* следует исключить из множества списков.

«Наращивание с конца» происходит при фрагментах, изображенных на рис. 6. Пусть на одном из них есть грань t, такая, что она следует за гранью, которая находится на просмотренной части. Пусть начало $(t)=j_1$, конец $(t)=j_2$, а $v(t)=v$. В этом случае требуется найти список c^* с заго-

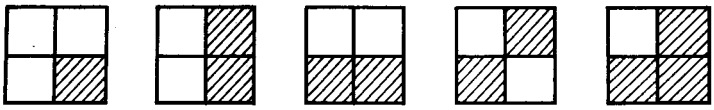

Рис. 6

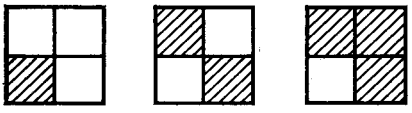

Рис. 7

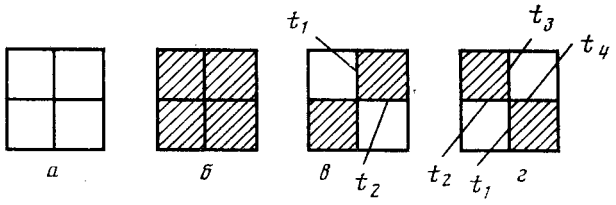

Рис. 8

ловком (j', j''), $j''=j_1$ и ввести новый список с заголовком (j', j_2). В этот список должна входить пара (s_1, s_2) в том и только в том случае, если существует $s^*\in S$, такое, что $(s_1, s^*)\in c^*$, а $(s^*, v, s_2)\in G$. Список c^* затем следует исключить.

«Склеивание» и «Замыкание». Во фрагментах, представленных на рис. 7, имеются грани, принадлежащие уже просмотренной части, однако лишь при наблюдении этих фрагментов оказывается, что эти грани соседние.

Пусть t_1 и t_2 — две такие грани, причем t_2 следует за t_1. Пусть конец $(t_1)=j_1$, а начало $(t_2)=j_2$. В таком случае следует обнаружить список c' с заголовком (j', j''), где $j''=j_1$, и список c^* с заголовком (j^*, j^{**}), где $j^*=j_2$. Следует различать две ситуации: когда заголовки этих двух списков совпадают и когда они различны. В последнем случае следует составить новый список с заголовком (j', j^{**}) и включить в этот список пару (s_1, s_2) в том и только в том случае, если существует структурный элемент s^*, такой, что $(s_1, s^*)\in c'$, а $(s^*, s_2)\in c^*$. Списки c' и c^* затем исключают. В случае если заголовки списков совпадают, следует его исключить, если в него входит пара (s_1, s_2), такая, что $s_1=s_2$. Если такой пары нет, то алгоритм переходит в исключительное состояние $\varnothing$, из которого он не выходит до конца просмотра изображения.

Как уже указывалось выше, в это же состояние алгоритм должен перейти при выполнении операции «Новый список» и «Наращивание», если только при построении очередного списка оказывается, что этот список пустой. Нахождение алгоритма в состоянии $\varnothing$ обозначает недопустимость предъявленного изображения.

Заметим, что не всегда при анализе текущего фрагмента необходимо выполнить единственную операцию из набора «Новый список», «Наращивание», «Склеивание» и «Замыкание». Так, например, фрагменты на рис. 8, *а*, *б* не содержат граней и, следовательно, не требуется никаких действий по изменению списков. На фрагменте рис. 8, *в* необходимо выполнить операцию «Наращивание с начала» для грани t_1 и «Наращивание с конца» для грани t_2. На фрагменте рис. 8, *г* необходимо выполнить операцию «Склеивание» либо «Замыкание» для граней t_1 и t_2 и операцию «Новый список» для граней t_3 и t_4.

5. Заключительные комментарии. Основным положительным свойством предложенного алгоритма является то, что его объем памяти не зависит от размеров изображения. От количества строк в изображении алгоритм не зависит принципиально, т. е. он может применяться для анализа сколь-угодно длинных «лент», например, изображений на некотором конвейере. От количества клеток в строке алгоритм зависит лишь опосредованно. Непосредственно объем памяти алгоритма зависит от максимального количества пересечений линии контура и границы просмотренной части изображения, т. е. от максимального количества пересечений линии контура с некоторой горизонтальной линией.

Вторым положительным свойством является то, что в этом алгоритме отсутствует ситуация «не знаю», характерная для алгоритмов синтаксического анализа в рамках общей модели двумерных грамматик.

Хотелось бы отметить определенную универсальность разработанного алгоритма. При фиксированном множестве G алгоритм обнаруживает контуры, имеющие форму, определяемую именно этим множеством. Если возникает необходимость обнаруживать объекты, имеющие другую форму, то необходимо изменить лишь сведения о множестве G. Вся остальная часть алгоритма при этом остается неизменной.

На практике, как правило, обработка изображения не сводится к одной лишь проверке допустимости. В зависимости от приложений требуется, например, осуществить подсчет объектов в поле зрения, их классификацию по тем или иным признакам, или иногда просто указать координаты объектов, форма которых оказалась неправильной, или многое-многое другое. Мы в данной статье специально уклонились от рассмотрения этих частных особенностей конкретных приложений, считая, что их учет нетрудно осуществить. Взамен этого мы сосредоточились на той части, которая является неизбежной в целом ряде приложений и реализуется с помощью введенной процедуры формирования списков.

ЛИТЕРАТУРА

1. *Шлезингер М. И.* Синтаксический анализ двумерных зрительных сигналов в условиях помех.– Кибернетика, 1976, № 4.
2. *Коваль В. К., Шлезингер М. И.* Двумерное программирование в задачах анализа изображений.– Автоматика и телемеханика, 1976, № 8.
3. *Freeman H.* On the encoding of arbitrary geometric configuration.— IEEE Trans. Electron. Comput., 1961, v. EC-10.

Киев

Поступила в редакцию
1.XII.1980

После доработки
7.V.1981