УДК 004.93'1:[519.76+519.857]

Шлезінгер М.І., Савчинський Б.Д., Анохіна М.О.

СИНТАКСИЧНИЙ АНАЛІЗ ТА РОЗПІЗНАВАННЯ ТИПОГРАФСЬКИХ НОТНИХ ТЕКСТІВ

Анотація. Розглядається структурний підхід до розпізнавання типографських нотних текстів. Як теоретичну основу алгоритмів розпізнавання використано двовимірні узагальнення контекстно-вільних граматик. Визначено та розглянуто їх підклас — лінійні граматики з фіксованим розміром нетерміналів. Наведено ефективний алгоритм аналізу зображень, що описуються граматиками цього підкласу.

УДК 004.93'1:[519.76+519.857]

Шлезингер М.И., Савчинский Б.Д., Анохина М.А.

СИНТАКСИЧЕСКИЙ АНАЛИЗ И РАСПОЗНАВАНИЕ ТИПОГРАФСКИХ НОТНЫХ ТЕКСТОВ

Аннотация. Рассматривается структурный подход к распознаванию типографских нотных текстов. В качестве теоретической основы алгоритмов распознавания использованы двумерные обобщения контекстно-свободных грамматик. Определен и рассмотрен их подкласс — линейные грамматики с фиксированным размером нетерминалов. Приведен эффективный алгоритм анализа изображений, описываемых грамматиками этого подкласса.

UDC 004.93'1:[519.76+519.857]

Schlesinger M.I., Savchynskyy B.D., Anohina M.A.

PARSING AND RECOGNITION OF PRINTED NOTES

Abstract. Structural approach to printed notes recognition is considered. Two-dimensional generalizations of context-free grammars have been used as a base of recognition algorithms. Special subclass named fixed nonterminals' size linear grammars of two-dimensional context-free grammars is defined and described. Effective algorithm for images described with the subclass' grammars analysis is presented.

УДК 004.93'1:[519.76+519.857]

Шлезингер М.И., Савчинский Б.Д., Анохина М.А.

СИНТАКСИЧЕСКИЙ АНАЛИЗ И РАСПОЗНАВАНИЕ ТИПОГРАФСКИХ НОТНЫХ ТЕКСТОВ[1]

1. Вступление.

Распознавание нотных текстов, кроме своего несомненного культурологического значения, имеет широкий спектр приложений при создании электронных музыкальных библиотек, автоматизации механического труда при переписывании партитуры на отдельные партии, транспонировании музыкальных произведений и во многих других практических работах.

Интерес к проблеме распознавания нотных текстов стимулируется также её внутренним научным содержанием. Изображение нотного текста является ярким примером объектов со сложной, но вполне определенной внутренней структурой, что дает возможность использовать новейшие методы структурного распознавания. Научным и прикладным проблемам распознавания нотных текстов посвящены известные работы [1,2,3]. Более полную библиографию содержит [4].

Работа состоит из пяти разделов. В следующем разделе описаны основные идеи, на которых базируются разработанные алгоритмы распознавания. Третий и четвертый разделы посвящены формализации этих идей. Базой для формализации служат двумерные обобщения контекстно-свободных языков и грамматик Хомского. В заключительном разделе поданы результаты экспериментальной проверки использованных методов и выводы.

2. Структура изображения страницы нотного текста. Неформальный взгляд.

Пусть X — это множество всех возможных изображений, E — известное его подмножество, которое далее понимается как множество определенных идеальных, эталонных изображений; $f : E \times X \to R$ — заданная функция такая, что $f(e, x)$ определяет, насколько реальное изображение x отличается от идеального изображения e.

[1] Работа была выполнена в рамках украинской государственной научно-технической программы "Образный компьютер" и поддерживалась программой DAAD правительства Германии.

Решаемая задача принадлежит к классу оптимизационных задач, в которых для заданного множества E и функции $f : E \times X \to R$ необходимо построить алгоритм, указывающий для каждого входного изображения x наименее отличающееся от него эталонное изображение $e*$ из множества E:

$$e^* = \arg\min_{e \in E} f(e, x). \quad (1)$$

Для конкретизации задачи необходимо однозначно определить множества E, X и функцию f. Определим эти понятия сначала неформально.

Начнем с множества E идеальных изображений нотных текстов. Данное множество удобно определить с помощью воображаемого процесса рисования таких изображений. Это известный и широкоиспользуемый в структурном распознавании способ задания множества объектов с помощью генеративной модели (см. [5]).

Первый этап состоит в последовательном разбиении чистого листа на горизонтальные полосы двух типов. Полосы первого типа определяют те места на бумаге, где будут располагаться строки нотного текста. Полосы второго типа – это промежутки между строками. Следствием первого этапа является определённый эскиз будущего изображения. Полосы второго типа – это та часть изображения, которая уже не будет изменяться, в то время как полосы первого типа определяют те места, в которых что-то должно дорисовываться на последующих этапах.

На втором этапе обрабатывается каждая полоса первого типа, построенная на первом этапе. Прежде всего, на каждой такой полосе рисуется изображение пяти линий нотного стана. Дальнейшая обработка полосы состоит в том, что по своей длине, то есть в горизонтальном направлении, она разбивается на последовательность плотно прилегающих друг к другу прямоугольников. Эти прямоугольники также делятся на два типа. Прямоугольники первого типа помечаются определённым образом для обозначения того, что изображение в них будет создаваться на третьем этапе, в прямоугольниках второго типа рисуется изображение, которое уже не будет изменяться.

В прямоугольниках второго типа создаются изображения элементарных музыкальных символов, т.е. таких, которые не состоят из других, более простых. Это ключи, паузы, символы тактовой черты, обозначения тактового размера и др. В их состав входит также особенный символ, обозначающий промежуток между собственно музыкальными символами. Благодаря этому можно считать, что прямоугольники, на которые разбивается строка нотного текста, плотно прилегают друг к другу.

Прямоугольники первого типа предназначены для музыкальных символов, составленных из более простых, элементарных. Это аккорды, представляющие собой вертикальную последовательность нот. Количество различных аккордов огромно по сравнению с количеством простых музыкальных символов, о которых шла речь в предыдущем абзаце.

Третий этап заключается в том, что в прямоугольниках первого типа, созданных на втором этапе, генерируется изображение аккордов. Под аккордом мы понимаем сложный объект, составляющийся по определённым правилам из таких элементарных музыкальных символов, как ноты (двух типов: чёрные и белые), штили, флажки, обозначающие длительность аккорда и т.п.

Очерченный процесс порождения изображений нотных текстов демонстрирует, каким образом определяется множество E идеальных изображений , на котором базируется формальная постановка задачи (1) их распознавания.

Значение функции $f(e,x)$, определяющее степень отличия реального изображения x от идеального изображения e, равно количеству несовпадающих в изображениях x и e пикселов.

Если бы речь шла о распознавании только идеальных изображений, то этот процесс сводился бы к восстановлению последовательности действий, результатом которых является именно то изображение, которое необходимо распознать. Действительно, зная эту последовательность, несложно определить последовательность аккордов, как совокупность звуков, звучащих одновременно, и длительность каждого аккорда. Но из-за неизбежной неидеальности реальных изображений их распознавание требует решения оптимизационной задачи (1). Это обозначает, что необходимо найти такую последовательность действий, результатом которой есть создание изображения. отличающегося от входного в наименьшем количестве пикселов.

Последовательный характер создания изображения делает естественным построение такого неправильного алгоритма распознавания, который последовательно воспроизводит этапы создания изображения в том же порядке, в котором оно создавалось. Это значит, что вначале, на основании определённых разумных эвристических соображений, определяется расположение отдельных строк нотного текста. Потом на каждой найденной строке находится местоположение отдельных музыкальных символов, включая аккорды. Наконец, анализируют каждый найденный аккорд с целью определения звуков, его составляющих, и их длительности. Такой распространенный подход (см.,

например, [1,2,3]) имеет очень серьёзные недостатки. Они заключаются в том, что каждый этап распознавания заканчивается конечным решением, которое, оказавшись ошибочным, не может быть исправлено на последующих этапах. Более того, ошибочное решение на каком-либо этапе неизбежно приводит к ошибкам на следующих этапах.

Таких недостатков не имеет алгоритм, базирующийся на точном решении задачи (1). Хотя количество изображений в множестве E экспоненциально зависит от длины музыкального текста, задача (1) может быть решена точно без полного просмотра всех изображений в этом множестве. Такое решение задачи основано на современных методах структурного распознавания (см.[5]). Именно благодаря последовательному характеру генерирования идеальных изображений они могут рассматриваться как предложения в определённых формальных языках, задаваемых с помощью конструкций, похожих на контекстно-свободные формальные грамматики Хомского [6]. Однако, специфика нашей прикладной задачи заключается в том, что музыкальный текст не является последовательностью элементарных символов, упорядоченной в одном направлении. Элементарные музыкальные символы расположены друг относительно друга как горизонтальном, так и в вертикальном направлении. Структурный анализ таких сложных образований требует использования более общих конструкций, чем контекстно-свободные языки и грамматики Хомского. Это так называемые двумерные контекстно-свободные грамматики и языки, определённые и исследованные в [5]. Там же приведен общий алгоритм решения задачи (1) в случае, когда E является двумерным контекстно-свободным языком. Исследование специфики изображений музыкальных текстов как объектов машинного анализа даёт возможность сконструировать алгоритм распознавания, на порядок более эффективный, чем известный общий алгоритм.

Именно такой эффективный алгоритм был использован авторами для распознавания изображений нотных текстов. Этот алгоритм, а также те грамматические конструкции, на которых он базируется, будут рассмотрены в следующих двух разделах.

3. Структура изображения страницы нотного текста. Формализация.

Пусть I и J — фиксированные натуральные числа. Будем называть **полем зрения** множество

$$T = \{(i, j) \mid i = 0...I-1,\ j = 0...J-1\}.$$

Элементом поля зрения является пиксел. Параметры поля зрения I и J называются размерами — соответственно высотой и шириной поля зрения.

Множеством сигналов будем называть множество $U = \{0,1\}$. В нашем случае значение сигнала ноль соответствует белому, а единица – черному пикселу.

Функции вида $x : T \to U$ будем называть **изображениями**. Множество всех возможных изображений обозначим U^T. Высотой $h(x)$ и шириной $l(x)$ изображения x будем называть соответственно высоту и ширину поля зрения, на котором данное изображение определено.

Нас будут интересовать два типа изображений: к первому принадлежат изображения всей нотной страницы, ко второму – изображения определённых музыкальных символов. Изображения второго типа будем называть **шаблонами** или **эталонными изображениями**, учитывая, что именно они в процессе распознавания определяют эталонный вид музыкальных символов. Эталонные изображения определены на полях зрения с меньшими высотой и шириной, чем высота и ширина поля зрения всей нотной страницы.

Фрагментом поля зрения назовём прямоугольник – подмножество поля зрения

$$\Pi_{h,l}^{(i_1,j_1)} = \{(i,j) \mid i_1 \le i < i_1 + h,\ j_1 \le j < j_1 + l,\ i_1, j_1, h, l \ge 0,\ i_1 + h \le I,\ j_1 + l \le J\}.$$

Значение параметра h будем называть высотой, а l — шириной фрагмента $\Pi_{h,l}$.

Ограничение изображения x на фрагмент поля зрения Π назовем **фрагментом изображения** и обозначим $x(\Pi)$. Высотой и шириной фрагмента изображения будем называть высоту и ширину фрагмента поля зрения, на котором изображение определено, и будем обозначать $h(x(\Pi))$ и $l(x(\Pi))$ соответственно.

Двумерной контекстно-свободной грамматикой G называется четверка $<V, K, P, \varepsilon>$, где V — множество терминалов, K — множество нетерминалов (нетерминальных символов), P — множество правил вывода, ε –- аксиома.

Терминалами (элементами множества V) являются изображения (шаблоны) символов и элементов нотного текста. Мы сознательно не называем их терминальными символами, как это делается в теории формальных языков, поскольку они не являются символами в обычном понимании этого слова. Касательно же всего иного, то они вполне аналогичны терминальным символам.

Нетерминалы – элементы множества K — используются в процессах анализа и распознавания изображения для именования его фрагментов.

Множество правил P содержит **правила горизонтальной** и **вертикальной конкатенации** и **правила замены**. Правила горизонтальной и вертикальной конкатенации имеют вид

$$A \to B \,|\, C,\ A \to \frac{B}{C},\ \ A,B,C \in K\ .$$

Вертикальная (|) и горизонтальная ($^-$) черты используются тут для обозначения направления конкатенации, а не для записи регулярных выражений, как это принято в теории формальных языков.

Правила замены имеют вид

$$A \to b,\ \ A \in K,\ b \in V\ .$$

Для сокращения записи используется запятая.

Например, запись

$$A \to B,C \,|\, D$$

определяет сразу два правила:

$$A \to B \,|\, D \quad \text{i} \quad A \to C \,|\, D.$$

При заданном изображении x использование правила $A \to b$ к фрагменту $x(\Pi)$ требует попиксельного совпадения фрагмента изображения $x(\Pi)$ и шаблона b, обозначая при этом условии присвоение фрагменту $x(\Pi)$ метки A.

Применение правила $A \to B \,|\, C$ к фрагменту $x(\Pi)$ требует выполнения такого условия: существует разбиение фрагмента $x(\Pi)$ в горизонтальном направлении на таких два фрагмента $x(\Pi_1)$ и $x(\Pi_2)$, что левый фрагмент $x(\Pi_1)$ уже обозначен меткой B, а правый $x(\Pi_2)$ — меткой C. Если выполняется это условие, применение правила обозначает присвоение фрагменту $x(\Pi)$ метки A.

Использование других правил вполне аналогично: необходимо заменить в предыдущем абзаце слово “горизонтальном” на “вертикальном” и слова “левый/правый” на “верхний/нижний”.

Язык грамматики G состоит из таких изображений e, для которых существует последовательность использования правил грамматики (не обязательно всех и не обязательно по одному разу), результатом которой есть присвоение метки ε всему изображению (как тривиальному своему фрагменту).

Иными словами, язык грамматики содержит изображения, составляющиеся по определённым правилам из тесно прилегающих друг к другу шаблонов из множества V. Язык грамматики G будем обозначать $L(G)$.

Последовательность правил, применённых к изображению e, результатом которой является присвоение метки ε всему изображению, называется **выводом** изображения e в грамматике G. Выводом также называется собственно процесс использования данной последовательности.

Локальной функцией штрафа будем называть функцию w_G**,** определенную на парах "шаблон - фрагмент изображения того же размера", принимающую значения в множестве действительных чисел. Величина $w_G(v, x(\Pi))$ определяет степень отличия шаблона $v \in V$ от фрагмента изображения $x(\Pi)$.

Пусть заданы произвольное изображение x и изображение e из языка грамматики G. Пусть также $b_1, b_2, ..., b_n$ — шаблоны, из которых состоит изображение e, а $\Pi_1, \Pi_2, ..., \Pi_n$ — соответствующие им фрагменты поля зрения. Введём функцию $f_G : L(G) \times X \to R$ такую, что $f_G(e, x)$ определяет степень отличия изображения x от изображения e. Такую функцию назовём **функцией штрафа**. Функции штрафа $f_G : L(G) \times X \to R$, для которых существует такая локальная функция штрафа w_G и такое разбиение изображения e на шаблоны $b_1, b_2, ..., b_n$**,** что f_G может быть представлена как величина суммарного отличия между шаблонами, из которых состоит изображения e, и соответствующими им фрагментами изображения x, т.е. функции f_G, представимые в виде

$$f_G(e, x) = \sum_{i=1}^{n} w_G(b_i, x(\Pi_i)),$$

будем называть **локально-аддитивными по отношению к** w_G**.**

Задача 3.1 *Заданы входное изображение* x*, двумерная контекстно-свободная грамматика* G *и локально-аддитивная функция штрафа* $f_G : L(G) \times X \to R$ *. Найти изображение* e^* *из языка грамматики* G*, минимизирующее величину* $f_G(\mathrm{g} x)$*:*

$$e^* = \arg \min_{e \in L(G)} f_G(e, x).$$

На рис.1 приведён пример грамматики $G = < V, K, P, \varepsilon >$, задающей определённое, предельно упрощённое множество нотных текстов. Этот пример приводится только как неотягощенная лишними подробностями иллюстрация основной идеи построения таких грамматик.

Множество V (не изображённое на рис. 1) содержит эталонные изображения символов и элементов нотного текста, а также изображения пустых, “белых” прямоугольников. В множество V входят эталонные изображения, присутствующие в записи правил множества P.

Множество правил P содержит три группы правил: первая группа задаёт структуру нотной страницы, как последовательности нотных строк вцелом, вторая — структуру отдельной нотной строки как последовательности символов нотного текста и, наконец, третья — структуру аккорда как последовательности целых нот и пустых мест между ними. Нетерминал NT используется в процессе построения страницы нотного текста из нотных строк и промежутков между ними и обозначает построенную уже часть нотной страницы. Нотные строки обозначаются нетерминалом NS, а промежутки — нетерминалом WS_1. Нотные строки являются последовательностью ключей (нетерминал CL), пауз (нетерминал PA), тактовых черт (нетерминал BL), промежутков между символами (нетерминал WS_2) и аккордов (нетерминал CH). Аккорды являются вертикальной последовательностью целых нот, обозначенных нетерминалом WN, и промежутков между ними, которым соответствует нетерминал WS_3.

Множество K содержит нетерминалы, использованные при задании множества правил P.

Функция w_G (также не изображенная на рис.1) определена таким образом, что величина $w_G(v, x(\Pi))$ равна количеству пикселов, в которых v и $x(\Pi)$ не совпадают. Функция штрафа f_G является локально-аддитивной по отношению к w_G.

Как уже было сказано, существует общий алгоритм решения задачи 3.1 для любой контекстно-свободной двумерной грамматики и любой локально-аддитивной функции штрафа. Этот алгоритм может быть использован и для сконструированной нами грамматики нотных текстов. Он требует $O(I^2 J^2 (I + J))$ времени для поиска вывода входного изображения. Указанное время является очень ощутимым для пользователя. Но специфика сконструированной нами “нотной” грамматики позволяет

построить алгоритм, требующий значительно меньше — $O(IJ(I+J))$ вычислительного времени. Алгоритм, о котором идёт речь, касается не только построенной нами грамматики нотных текстов, но и значительного подкласса контекстно-свободных грамматик. Этот подкласс, названный нами классом **линейных грамматик с фиксированными размерами нетерминалов**, и алгоритм грамматического разбора грамматик из него будут описаны в следующем пункте.

4. Класс линейных грамматик с фиксированными размерами нетерминалов.

Лемма 4.1 *(Достаточное условие фиксированности высоты нетерминала.)*

Пусть $G=<V,K,P,\varepsilon>$ *— двумерная контекстно-свободная грамматика. Пусть нетерминал* $A\in K$ *удовлетворяет таким условиям:*

1. *Правила замены, в левой части которых стоит нетерминал* A*, в правой части содержат терминалы одинаковой высоты:*

$$(((A\to a_i,\ a_i\in V)\in P)\,\&\,((A\to a_j,\ a_j\in V)\in P))\Rightarrow(h(a_i)=h(a_j))$$

2. *Нетерминал* A *не содержится в левой части правил вертикальной конкатенации, а правила горизонтальной конкатенации, содержащие* A *в левой части, имеют вид:*

$$A\to B\,|\,A,\ \ B\in K \quad \text{або} \quad A\to A\,|\,B,\ \ B\in K.$$

Тогда в процессе вывода любого входного изображения в грамматике G *нетерминал* A *может обозначать фрагменты изображения только одной, фиксированной высоты.*

Замечание 4.1 *Справедлива также аналогичная лемма, определяющая достаточное условие фиксированности ширины нетерминала.*

Если нетерминал A может обозначать фрагменты только одной, фиксированной высоты (ширины), то эту высоту (ширину) будем называть высотой (шириной) нетерминала A.

Сформулированная лемма служит обоснованием корректности следующего определения.

Определение 4.1 *Двумерную контекстно-свободную грамматику* $G=<V,K,P,\varepsilon>$ *будем называть* **линейной грамматикой с фиксированными размерами нетерминалов**, *если её составляющие удовлетворяют следующим условиям*

1. Множество нетерминалов K *может быть представлено, как объединение двух подмножеств* K_h *и* K_w*:*

$$K = K_h \cup K_w.$$

Подмножество K_h содержит нетерминалы, могущие обозначать фрагменты изображения только заданной, фиксированной высоты.

Подмножество K_w содержит нетерминалы, могущие обозначать фрагменты изображения только заданной, фиксированной ширины.

2. В правой части каждого правила горизонтальной конкатенации содержится хотя бы один нетерминал из множества K_w, а в правой части каждого правила вертикальной конкатенации — хотя-бы один нетерминал из K_h. Таким образом, правила конкатенации имеют вид:

$$A \to \alpha \mid C,\ A \to C \mid \alpha,\ A \to \frac{\beta}{C},\ A \to \frac{C}{\beta},$$
$$A, C \in K,\ \alpha \in K_w,\ \beta \in K_h.$$

Замечание 4.2 *В общем случае множества K_h и K_w пересекаются. Множество, являющееся их пересечением, состоит из нетерминалов, могущих обозначать фрагменты изображения только фиксированных высоты и ширины.*

В построенной в предыдущем разделе грамматике нотных текстов множества K_h и K_w выглядят таким образом:

$$K_h = \{NS, CL, PA, BL, WN, WS_1, WS_2, WS_3\},$$
$$K_w = \{NT, CH, CL, PA, BL, WN, WS_1, WS_2, WS_3\}.$$

Множество же правил вывода очевидно удовлетворяет введенному определению.

Рассмотрим множество тех фрагментов изображения, которые могут быть обозначены нетерминальными символами. На этом множестве определён частичный порядок относительно операции вложения одного фрагмента в другой. Как всякое частично упорядоченное множество, оно может быть полностью упорядочено без потери существующего частичного порядка.

Мы приведём алгоритм, который, в целях ясности изложения, содержит действия, необходимые для вычисления только значения штрафа $f_G(e^*, x)$, где e^* — решение задачи 3.1 для входного изображения x, линейной грамматики с фиксированными размерами нетерминалов G и локально-

аддитивной функции штрафа f_G. Но этот алгоритм легко может быть превращён в такой, который находит также само изображение e^*, т.е полностью решает задачу 3.1. Необходимые изменения указаны в замечании 4.3, приведённом после алгоритма.

1. Упорядочим множество тех фрагментов изображения, которые могут быть обозначены нетерминальными символами, пронумеровав фрагменты согласно введенному порядку так, чтобы фрагменты с меньшими размерами имели меньшие номера. Будем обозначать эти фрагменты $x(\Pi_s)$, где индекс — прописная буква s — обозначает номер фрагмента. Само же множество фрагментов обозначим символом F. Кроме этого, некоторые фрагменты обозначаются индексом, являющимся заглавной буквой — меткой нетерминального символа. Так, обозначения $x(\Pi_A)$ и $x(\Pi_B)$ указывают на то, что фрагменты $x(\Pi_A)$ и $x(\Pi_B)$ обозначены нетерминалами A и B соответственно.

Рассмотрим массив $g(|F|,|K|)$, содержащий $|F|\times|K|$ элементов, принимающих значения действительного типа. Тот факт, что элемент массива $g(x(\Pi_s),k)$ принимает значение $r \in R$ обозначает, что за присвоение метки k фрагменту $x(\Pi_s)$ был заплачен штраф r. Проинициализируем массив начальными значениями ∞. Символ ∞ обозначает, что метка соответствующего нетерминала ещё не была присвоена фрагменту. Процесс инициализации иллюстрируется программой:

$for\ (s = 0; s < |F|; s++)$
$for\ (k \in K)$
$f(x(\Pi_s), k) = \infty$

2. Будем последовательно выбирать фрагменты из множества F и для каждого фрагмента $x(\Pi_s)$, каждого нетерминала k и каждого правила вывода p, содержащего в левой части нетерминал k, будем определять штраф за присвоение метки k фрагменту $x(\Pi_s)$.

Так при применении правила горизонтальной конкатенации $k \to A|B$ фрагмент $x(\Pi_s)$ разбивается по горизонтали на два прилегающих друг к другу фрагмента $x(\Pi_A)$ та $x(\Pi_B)$.

Такое разбиение фрагмента $x(\Pi_s)$ на $x(\Pi_A)$ и $x(\Pi_B)$ при фиксированном правиле p может быть сделано единственным образом благодаря условию, наложенному на правила вывода определением 4.1. При разбиении фрагмента $x(\Pi_s)$ при условии, что нетерминал A принадлежит к

множеству K_w, ширина $x(\Pi_A)$ устанавливается равной ширине нетерминала A, а левый верхний угол $x(\Pi_A)$ совпадает с левым верхним углом $x(\Pi_s)$. Фрагмент $x(\Pi_B)$ дополняет фрагмент $x(\Pi_A)$ к $x(\Pi_s)$. Если же наоборот, нетерминал B принадлежит K_w, то ширина фрагмента $x(\Pi_B)$ устанавливается равной ширине B, а правый верхний угол фрагмента $x(\Pi_B)$ совпадает с правым верхним углом $x(\Pi_s)$. Такое разбиение фрагмента $x(\Pi_s)$ на $x(\Pi_A)$ и $x(\Pi_B)$ в горизонтальном направлении мы обозначим $x(\Pi_s) = x(\Pi_A) | x(\Pi_B)$.

Аналогично определяется разбиение для правил вертикальной конкатенации. Вертикальную конкатенацию фрагментов будем обозначать $x(\Pi_s) = \frac{x(\Pi_A)}{x(\Pi_B)}$.

После разбиения фрагмента $x(\Pi_s)$ на $x(\Pi_A)$ и $x(\Pi_B)$ подсчитывается число — сума штрафов за вывод левого (верхнего) и правого (нижнего) фрагментов $g(x(\Pi_A), A) + g(x(\Pi_B), B)$. Числа $g(x(\Pi_A), A)$ и $g(x(\Pi_B), B)$ уже были вычислены алгоритмом, поскольку фрагменты $x(\Pi_A)$ и $x(\Pi_B)$ являются частями фрагмента $x(\Pi_s)$, а, следовательно, согласно введенному отношению полного порядка, являются меньшими, чем фрагмент $x(\Pi_s)$.

При применении правила замены $k \to b,\ b \in V$ к фрагменту $x(\Pi_s)$ подсчитывается число $w_G(b, x(\Pi_s))$, которым и инициализируется элемент $g(x(\Pi_s), k)$.

Из полученных чисел (вычисленных для правил замены и правил горизонтальной и вертикальной конкатенации) выбирается наименьшее. Его величина определяет значение штрафа, который записывается в элемент $g(x(\Pi_s), k)$ массива g. Обозначив P_h, P_v, P_c множества правил горизонтальной, вертикальной конкатенации и правил замены соответственно, запишем указанные действия в виде:

$for\ (s = 0; s < |F|; s++)$
$for\ (k \in K)$
{

$$r_h = \min_{\substack{p \in P_h \\ p=(k \to A|B,\ \ A,B \in K) \\ x(\Pi_s)=x(\Pi_A)|x(\Pi_B)}} (g(x(\Pi_A), A) + g(x(\Pi_B), B))$$

$$r_v = \min_{\substack{p \in P_v \\ p=(k \to \frac{A}{B},\ A,B \in K) \\ x(\Pi_s)=\frac{x(\Pi_A)}{x(\Pi_B)}}} (g(x(\Pi_A), A) + g(x(\Pi_B), B))$$

$$r_c = \min_{\substack{p \in P_c \\ p=(k \to b,\ b \in V)}} (w(b, x(\Pi_s)))$$

$$g(x(\Pi_s), k) = \min(r_h, r_v, r_c)$$

}.

3. В результате работы алгоритма в элемент $g(x(T), \varepsilon)$ массива будет записана величина $f_G(e^*, x)$.

Замечание 4.3 *Рассмотренный алгоритм легко может быть модифицирован для нахождения не только штрафа* $f_G(e^*, x)$, *но и самого изображения* e^* *из языка грамматики* G, *наиболее похожего на входное изображение* x. *Изображение* e^*, *как и любое изображение из языка грамматики* G, *однозначно задаётся своим выводом. Для нахождения вывода в массив* g, *кроме штрафа за наилучший вывод фрагмента, следует записывать правило, благодаря которому это значение штрафа было достигнуто. После окончания работы алгоритма следует выбрать из массива* g *последовательность тех правил, которые привели к записи определённого, отличного от* ∞ *штрафа в элемент* $g(x(T), \varepsilon)$ *массива* g. *Последним в искомом выводе является правило, записанное в этом элементе* .

Временная сложность рассмотренного алгоритма может быть легко оценена. Массив g содержит $|F| \times |K|$ элементов. Множество F содержит не более, чем $|K_h| IJ^2$ фрагментов, которые могут быть обозначены нетерминалами из K_h и не более чем $|K_w| JI^2$ фрагментов, которые могут быть обозначены нетерминалами из K_w. Следовательно, массив g содержит $O(|K| \times IJ(|K_h| J + |K_w| I))$ элементов. В процессе работы все они просматриваются по одному разу, при этом в процессе просмотра одного элемента может понадобиться просмотр не более $|P|$ правил вывода. Результирующая временная сложность алгоритма — $O(|P||K| IJ(|K_h| J + |K_w| I))$. При условии фиксированности

грамматики, а, значит, при фиксированных множествах P и K временная сложность зависит как $O(IJ(I+J))$ от размеров входного изображения

5 Демонстрация результатов.

Экспериментальная проверка разработанного метода имела целью верификацию основных идей, на которых базируется разработанный алгоритм. Поэтому эксперименты ставились на простых, но реальных музыкальных текстах, а не на текстах произвольной сложности. Тесты были взяты из сборника музыкальных упражнений для начальных классов музыкальной школы. Эти нотные тексты удовлетворяют таким условиям:

- аккорды в пределах одного такта образуют последовательность, а не две последовательности, как при записи верхнего и нижнего голоса на одном нотном стане;
- нотные тексты не содержат лиг;
- нотные тексты не содержат форшлагов, динамических обозначений (таких, как форте. пиано, крещендо) и других обозначений, определяющих настроение, темп, постановку пальцев и т.д.

Примеры таких изображений приведены на и 3. Отрицательные результаты экспериментов на изображениях такого класса требовали бы пересмотра всей концепции, положенной в основу разработанной технологии. Положительные результаты позволяют утверждать, что дальнейшее усовершенствование программы, т.е. расширение класса музыкальных текстов, потребует только количественного усовершенствования технологии: увеличения количества элементарных символов, усложнения грамматики и т.д.

Тестирование разработанных методов проводилось на 20 изображениях страниц нотного теста, каждая из которых содержала 5-10 нотных строк. Качество распознавания характеризуется тем, что в среднем из 100 музыкальных символов 3 были распознаны неправильно.

Не следует понимать этот факт как то, что вероятность ошибки равна трем сотым, поскольку на данном этапе разработки програмного модуля ошибка вообще не является случайной величиной. Она обусловлена загрублением модели: програмная реализация не учитывает наличие многих музыкальных символов, загрублены соотношения, задающие взаимное расположение элементарных музыкальных символов и т.д. Вместе с тем, не следует считать, что практическую ценность имеет только програмное обеспечение, учитывающее всё разнообразие способов записи музыки. Тогда необходимо было бы

включить в его состав как минимум средства распознавания текстов. Практическую ценность имеет то программное обеспечение, которое позволяет эффективно редактировать результаты и исправлять ошибки, обусловленные теми или иными неучтенными особенностями музыкального текста, при условии, что потребность в таком исправлении не слишком велика. Наш програмный комплекс содержит средства как редактирования, так и озвучивания нотных текстов.

На рис.3 изображен один из примеров, на которых производилось тестирование. Результаты распознавания показаны на рис.4. Правильно были распознаны количество и положение нотных строк и последовательности символов нотного текста во всех нотных строках.

Литература

1. T. Beran, T. Macek. Recognition of printed music score. In: *MLDM'99*, LNAI 1715, pp. 174-179. Springer-Verlag, Berlin/Heidelberg, 1999.

2. B. Cö̱asnon, B. Rétif. Using a grammar for a realible full score recognition system. In: *Proc. of the International Computer Music Conference*, Canada, Sept.1995.

3. J.C. Pinto, P. Viera, M. Ramalho, M. Mengucci, P. Pina, F. Muge. Ancient music recovery for digital libraries. In: J. Borbinha, T. Baker (Eds.): *ECDL 2000*, LNCS 1923, pp. 24-34, 2000. Springer-Verlag, Berlin/Heidelberg, 2000.

4. D. Blostein, H.S. Baird. A critical survey of music image analysis. In: *Structured Document Image Analysis*, pp.405-434. Eds. H.S. Baird, H. Bunke, K. Yamamoto, Springer-Verlag, Berlin/Heidelberg, 1992.

5. Michail I. Schlesinger, Vaclav Hlavač. Ten lectures on statistical and structural pattern recognition. 519 pages, Kluwer Academic Publishers, Dordrecht/Boston/London, 2002.

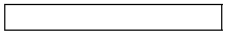
6. А. Ахо, Дж. Ульман. Теория синтаксического анализа, перевода и компиляции. т.1. Синтаксический анализ. 614 стр. изд-во МИР, Москва, 1978.

$G =< V, K, P, \varepsilon >$

$\varepsilon = NT$

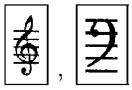
$K = \{NT, WS_1, NS, CL, PA, BL, WS_2, CH, WS_3, WN\}$

$NT \mapsto \frac{NS, WS_1}{NT}$

$NT \mapsto$ 

$WS_1 \mapsto$ 

$NS \mapsto CL, PA, BL, WS_2, CH|NS$

$CL \mapsto$  $PA \mapsto$ 

$BL \mapsto$  $WS_2 \mapsto$  $NS \mapsto$ 

$CH \mapsto \frac{WS_3, WN}{CH}$ $CH \mapsto \frac{CH}{WS_3, WN}$

$CH \mapsto$  $WS_3 \mapsto$  $WN \mapsto$ 

Рис. 1 Грамматика G нотных текстов

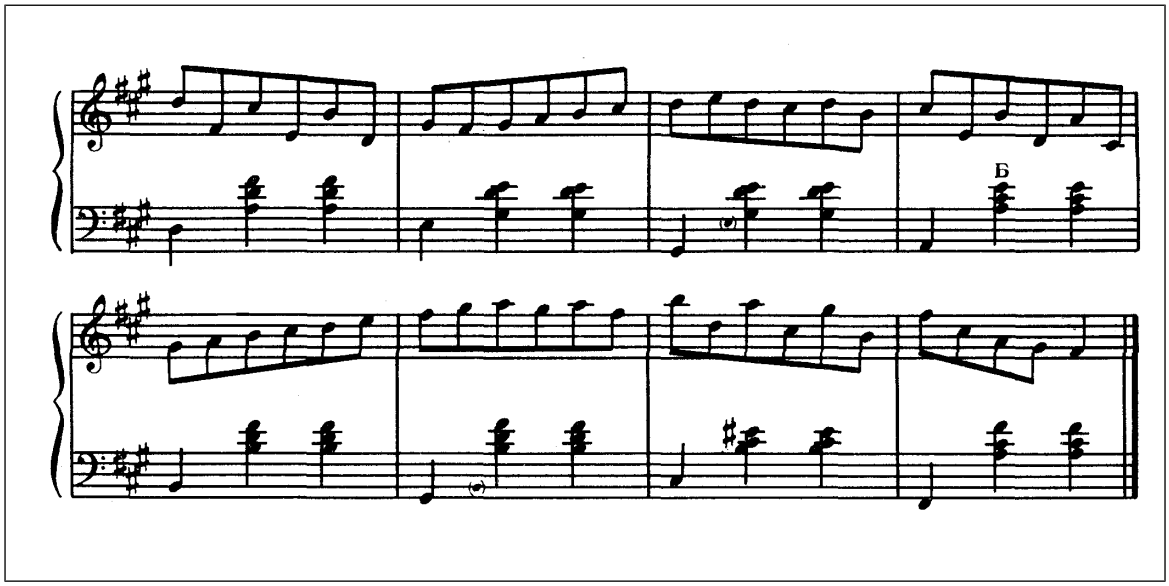


Рис. 2 Один из тех нотных текстов, на которых производилось тестирование

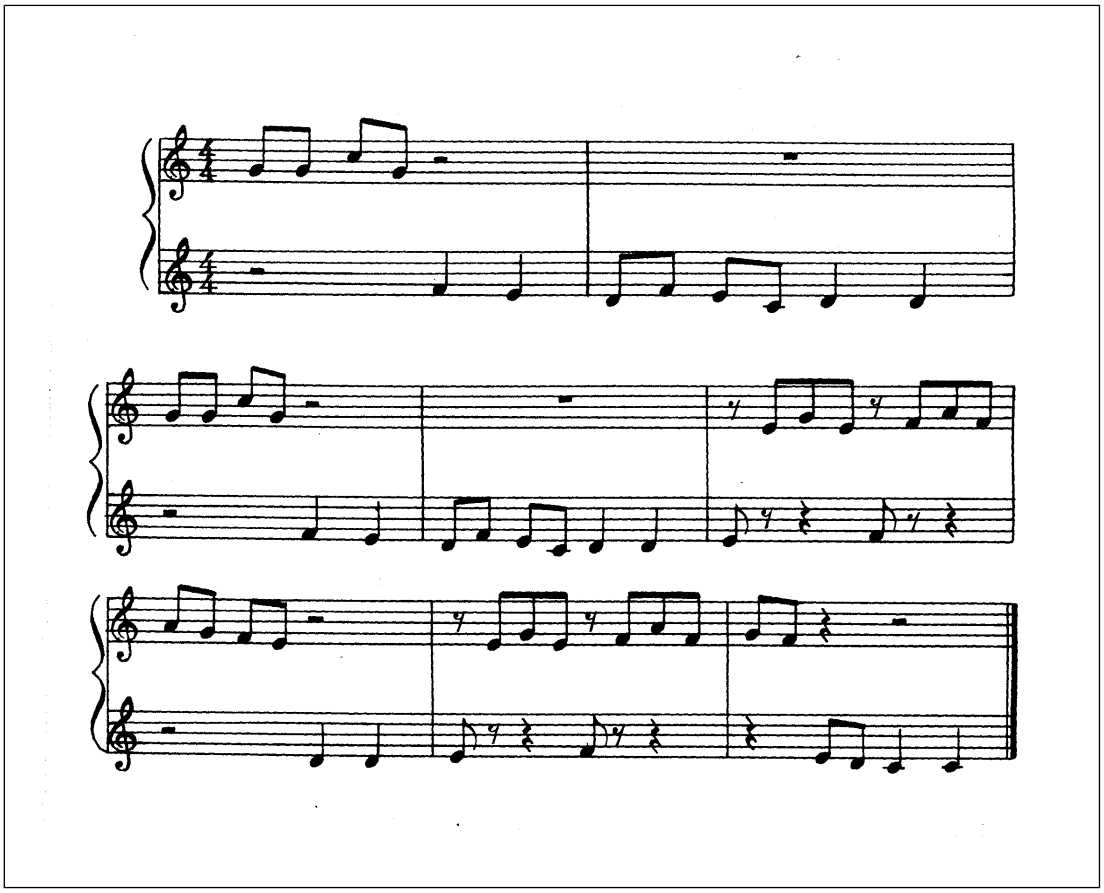


Рис. 3 Еще один пример нотного текста

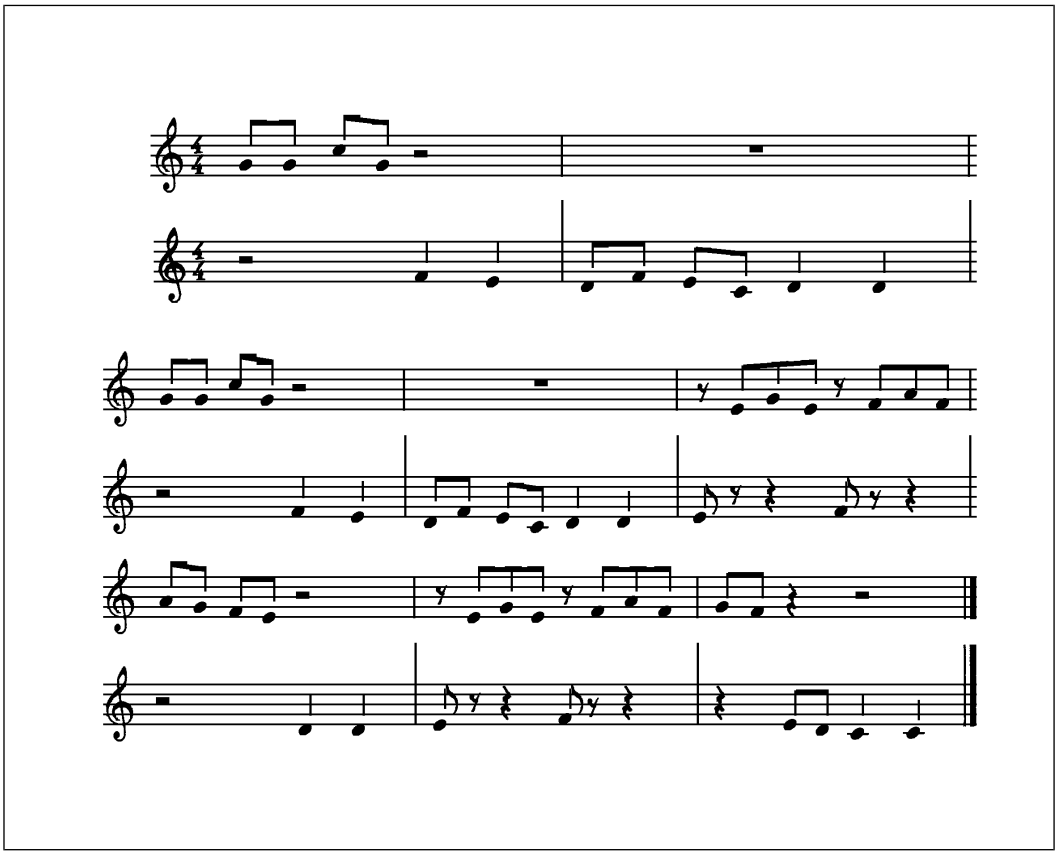


Рис. 4 Результат распознавания текста, изображенного на рис. 3

Рис. 1 Граматика G нотних текстів

Рисунок 1 Граматика G нотних текстів