

## СИНТАКСИЧНИЙ АНАЛІЗ ТА РОЗПІЗНАВАННЯ ТИПОГРАФСЬКИХ НОТНИХ ТЕКСТІВ<sup>1</sup>

### 1. Вступ

Розпізнавання нотних текстів, окрім свого безсумнівного культурологічного значення, має широкий спектр застосувань для створення електронних музичних бібліотек, автоматизації механічної праці при переписуванні партитури на окремі партії, транспонуванні музичних творів та в багатьох інших практичних роботах.

Інтерес до проблеми розпізнавання нотних текстів стимулюється також її внутрішнім науковим змістом. Зображення нотного тексту є яскравим прикладом об'єктів із складною, але цілком визначеною внутрішньою структурою, що дає змогу застосувати новітні методи структурного розпізнавання. Науковим та прикладним проблемам розпізнавання нотних текстів присвячені відомі роботи [1, 2, 3]. Повнішу бібліографію містить [4].

Робота складається з п'яти розділів. У наступному розділі описані основні ідеї, на яких базуються розроблені алгоритми розпізнавання. Третій та четвертий розділи присвячені формалізації цих ідей. Базою для формалізації служать двовимірні узагальнення контекстно-вільних мов та граматики Хомського. У заключному розділі подаються результати експериментальної перевірки застосованих методів та висновки.

### 2. Структура зображення сторінки нотного тексту. Неформальний погляд.

Нехай  $X$  — це множина всіх можливих зображень,  $E$  — відома її підмножина, яка далі розуміється як множина певних ідеальних, еталонних зображень;  $f: E \times X \rightarrow R$  — це задана функція така, що  $f(e, x)$  характеризує, наскільки реальне зображення  $x$  відрізняється від ідеального зображення  $e$ .

---

<sup>1</sup> Робота була виконана в рамках української державної науково-технічної програми "Образний комп'ютер" і підтримувалась програмою DAAD уряду Німеччини.

Вирішувана задача належить до класу оптимізаційних задач, в яких для заданої множини  $E$  та функції  $f : E \times X \rightarrow R$  треба побудувати алгоритм, що для кожного вхідного зображення  $x$  вказує таке еталонне зображення  $e^*$  з множини  $E$ , яке найменше відрізняється від  $x$ :

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

Для конкретизації задачі потрібно однозначно визначити множини  $E$ ,  $X$  та функцію  $f$ . Визначимо ці поняття спочатку неформально.

Почнемо з множини  $E$  ідеальних зображень нотних текстів. Дану множину доречно визначити за допомогою уявного процесу малювання таких зображень. Це відомий і широкоживаний у структурному розпізнаванні спосіб, коли множина об'єктів задається за допомогою генеративної моделі (див. [5]).

Перший етап полягає в послідовному розбитті чистого аркуша на горизонтальні смуги двох типів. Смуги першого типу визначають ті місця на папері, де розташовуватимуться рядки нотного тексту. Смуги другого типу — це проміжки між рядками. Наслідком першого етапу є певний ескіз майбутнього зображення. Смуги другого типу — це та частина зображення, яка вже не буде змінюватися, водночас смуги першого типу визначають ті місця, в яких щось має домалюватися на наступних етапах.

На другому етапі обробляється кожна смуга першого типу, побудована на першому етапі. Перш за все на кожній такій смузі малюється зображення п'яти ліній нотного стану. Подальша обробка смуги полягає в тому, що по своїй довжині, тобто в горизонтальному напрямку, вона розбивається на послідовність прямокутників, які щільно прилягають один до одного. Ці прямокутники також поділяються на два типи. Прямокутники першого типу позначаються певним чином на знак того, що зображення в них буде створене на третьому етапі, в прямокутниках другого типу малюється зображення, яке вже не буде змінюватися.

У прямокутниках другого типу створюються зображення елементарних музичних символів, тобто таких, що не складаються з інших, простіших. Це ключі, паузи, тактові риски, позначення тактового розміру та ін. До їх складу входить також особливий символ, який означає проміжок між власне музичними символами. Завдяки цьому можна вважати, що прямокутники, на які розбивається рядок нотного тексту, щільно прилягають один до одного.

Прямокутники першого типу призначені для музичних символів, складених з більш простих, елементарних. Це акорди, які являють собою вертикальну послідовність нот. Кількість різних акордів величезна порівняно з кількістю простих музичних символів, про які йшлося в попередньому абзаці.

Третій етап полягає в тому, що в прямокутниках першого типу, створених на другому етапі, генерується зображення акордів. Під акордом розуміється складний об'єкт, що за певними правилами складається з таких елементарних музичних символів, як ноти (двох типів: чорні та білі), штилі, прапорці, що позначають тривалість акорду, тощо.

Окреслений процес породження зображень нотних текстів демонструє, яким чином визначається множина  $E$  ідеальних зображень, на якій базується формальна постановка задачі (1) їх розпізнавання.

Значення функції  $f(e, x)$ , яке визначає, наскільки реальне зображення  $x$  відрізняється від ідеального зображення  $e$ , дорівнює кількості пікселів, які не збігаються в зображеннях  $x$  та  $e$ .

Якби йшлося про розпізнавання лише ідеальних зображень, то цей процес зводився б до відновлення послідовності дій, результатом яких є саме те зображення, яке треба розпізнати. Справді, знаючи цю послідовність, неважко визначити послідовність акордів, як сукупність звуків, що звучать одночасно, та тривалість кожного акорду. Але через неминучу неідеальність реальних зображень їх розпізнавання вимагає вирішення оптимізаційної задачі (1). Це означає, що треба знайти таку послідовність дій, результатом якої є створення зображення, що відрізняється від вхідного в найменшій кількості пікселів.

Послідовний характер створення зображення робить природним побудову такого неправильного алгоритму розпізнавання, який послідовно відтворює етапи створення зображення у тому ж порядку, в якому воно створювалось. Це значить, що спочатку, на основі певних розумних евристичних міркувань, визначають розташування окремих рядків нотного тексту. Потім на кожному виявленому рядку визначають місцеположення окремих музичних символів, включаючи акорди. Нарешті, аналізують кожний виявлений акорд з метою визначення звуків, які його складають, та їх тривалості. Такий поширений підхід (див., наприклад, [1, 2, 3]) має дуже серйозні недоліки. Вони полягають в тому, що кожен етап розпізнавання завершується кінцевим рішенням, яке, виявившись хибним, не може бути виправлене на наступних етапах. Більше за те, хибне рішення на якомусь етапі неминуче призводить до помилок на наступних етапах.

Таких недоліків не має алгоритм, який базується на точному вирішенні задачі (1). Хоча кількість зображень у множині  $E$  експоненційно залежить від довжини музичного тексту, задача (1) може бути точно вирішена без повного перегляду всіх зображень у цій множині. Таке вирішення задачі (1) ґрунтується на сучасних методах структурного розпізнавання (див. [5]). Саме завдяки послідовному характеру генерування ідеальних зображень вони можуть розглядатися як речення у певних формальних мовах, що задаються за допомогою конструкцій, схожих на контекстно-вільні формальні граматики Хомського [6]. Однак, специфіка нашої прикладної задачі полягає в тому, що музичний текст не є послідовністю елементарних символів, впорядкованою в одному напрямку. Елементарні музичні символи розташовані стосовно один одного як в горизонтальному, так і в вертикальному напрямках. Структурний аналіз таких складних утворень потребує використання більш загальних конструкцій, ніж контекстно-вільні мови та граматики Хомського. Це так звані двовимірні контекстно-вільні граматики та мови, визначені і досліджені у [5]. Там же наведений загальний алгоритм вирішення задачі (1) у випадку, коли  $E$  є двовимірною контекстно-вільною мовою. Дослідження специфіки зображень музичних текстів як об'єктів машинного аналізу дає змогу сконструювати алгоритм розпізнавання, який є на порядок ефективнішим, ніж відомий загальний алгоритм.

Саме такий ефективний алгоритм був використаний авторами для розпізнавання зображень нотних текстів. Цей алгоритм, а також ті граматичні конструкції, на яких він базується, будуть розглянуті в наступних двох розділах.

### 3. Структура зображення сторінки нотного тексту. Формалізація.

Нехай  $I$  і  $J$  — фіксовані натуральні числа. Будемо називати **полем зору** множину

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

Елементом поля зору є піксел. Параметри поля зору  $I$  та  $J$  називатимемо розмірами — відповідно висотою та шириною поля зору.

**Множиною сигналів** називатимемо множину  $U = \{0, 1\}$ . В нашому випадку значення сигналу нуль відповідає білому, а одиниця — чорному пікселю.

Функцію вигляду  $x: T \rightarrow U$  називатимемо **зображенням**. Множину всіх можливих зображень позначатимемо  $U^T$ . Висотою та шириною зображення  $x$  називатимемо відповідно висоту та ширину поля зору, на якому дане зображення визначене і позначатимемо  $h(x)$  та  $l(x)$ .

Нас цікавитимуть два типи зображень: до першого належать зображення всієї нотної сторінки, до другого — зображення певних музичних символів. Зображення другого типу ми називатимемо **шаблонами** або **еталонними зображеннями**, зважаючи на те, що саме вони в процесі розпізнавання визначатимуть еталонний вигляд музичних символів. Еталонні зображення визначені на полях зору з меншими висотою та шириною, аніж висота та ширина поля зору зображення всієї нотної сторінки.

**Фрагментом поля зору** будемо називати прямокутник – підмножину поля зору

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

Значення параметра  $h$  називатимемо висотою, а  $l$  — шириною фрагмента  $\Pi_{h,l}$ .

Обмеження зображення  $x$  на фрагмент поля зору  $\Pi$  називатимемо **фрагментом зображення** і позначатимемо  $x(\Pi)$ . Висотою та шириною фрагмента зображення називатимемо висоту та ширину фрагмента поля зору, на якому зображення визначене, і позначатимемо  $h(x(\Pi))$  та  $l(x(\Pi))$ .

**Двовимірною контекстно-вільною граматику**  $G$  називається четвірка  $\langle V, K, P, \varepsilon \rangle$ , де  $V$  — множина терміналів,  $K$  — множина нетерміналів (нетермінальних символів),  $P$  — множина правил виводу,  $\varepsilon$  — аксіома.

**Терміналами** (елементами множини  $V$ ) є еталонні зображення (шаблони) символів та елементів нотного тексту. Ми навмисне не називаємо їх термінальними символами, як це робиться в теорії формальних мов, оскільки вони не є символами в звичайному розумінні цього слова. Що до усього іншого, то вони цілком аналогічні термінальним символам.

**Нетермінали** — елементи множини  $K$  — використовуються в процесах аналізу та розпізнавання зображення для іменування його фрагментів.

Множина правил  $P$  містить **правила горизонтальної та вертикальної конкатенації** і **правила заміни**. Правила горизонтальної та вертикальної конкатенації мають вигляд

$$A \rightarrow B \mid C, A \rightarrow \frac{B}{C}, A, B, C \in K.$$

Вертикальна ( $\mid$ ) та горизонтальна ( $\frac{\quad}{\quad}$ ) риски використовуються тут для позначення напрямку конкатенації, а не для запису регулярних виразів, як це прийнято в теорії формальних мов.

Правила заміни мають вигляд

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

Для скорочення запису правил використовується кома.

Наприклад, запис

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

визначає одразу два правила:

$$A \rightarrow B | D \quad \text{і} \quad A \rightarrow C | D.$$

При заданому зображенні  $x$  застосування правила  $A \rightarrow b$  до фрагмента  $x(\Pi)$  потребує попіксельного збігання фрагмента зображення  $x(\Pi)$  та шаблона  $b$  і за цієї умови означає присвоєння фрагменту  $x(\Pi)$  позначки  $A$ .

Застосування правила  $A \rightarrow B | C$  до фрагмента  $x(\Pi)$  потребує виконання такої умови: існує розбиття фрагмента  $x(\Pi)$  у горизонтальному напрямку на таких два фрагменти  $x(\Pi_1)$  і  $x(\Pi_2)$ , що лівий фрагмент  $x(\Pi_1)$  вже має позначку  $B$ , а правий  $x(\Pi_2)$  — позначку  $C$ . Якщо ця умова справджується, застосування правила означає присвоєння фрагменту  $x(\Pi)$  позначки  $A$ .

Застосування інших правил цілком аналогічне: потрібно замінити в попередньому абзаці слово "горизонтальному" на "вертикальному" і слова "лівий/правий" на "верхній/нижній".

**Мова граматики**  $G$  складається з таких зображень  $e$ , для яких існує послідовність використання правил граматики (не обов'язково всіх і не обов'язково по одному разу), результатом якої є присвоєння позначки  $\varepsilon$  всьому зображенню (як тривіальному своєму фрагменту).

Іншими словами, мова граматики містить зображення, що складаються за певними правилами із прилеглих один до одного шаблонів з множини  $V$ . Мову граматики  $G$  позначатимемо  $L(G)$ .

Послідовність правил, застосованих до зображення  $e$ , результатом якої є присвоєння позначки  $\varepsilon$  всьому зображенню, називається **выводом** зображення  $e$  в граматиці  $G$ . Выводом також називається власне процес використання даної послідовності.

**Локальною функцією штрафу** будемо називати функцію  $w_G$ , визначену на парах "шаблон - фрагмент зображення того ж розміру", яка приймає значення в множині дійсних чисел. Величина  $w_G(v, x(\Pi))$  визначає наскільки відрізняються шаблон  $v \in V$  та фрагмент зображення  $x(\Pi)$ .

Нехай задані довільне зображення  $x$  та зображення  $e$  з мови граматики  $G$ . Нехай також  $b_1, b_2, \dots, b_n$  - шаблони, з яких складається зображення  $e$ , а  $\Pi_1, \Pi_2, \dots, \Pi_n$  - відповідні їм фрагменти поля

зору. Введемо функцію  $f_G : L(G) \times X \rightarrow R$  таку, що  $f_G(e, x)$  характеризує, наскільки зображення  $x$  відрізняється від зображення  $e$ . Таку функцію називатимемо **функцією штрафу**. Функції штрафу  $f_G : L(G) \times X \rightarrow R$ , для яких існує така локальна функція штрафу  $w_G$  і таке розбиття зображення  $e$  на шаблони  $b_1, b_2, \dots, b_n$ , що  $f_G$  може бути представлена як величина сумарної різниці між шаблонами, з яких складається зображення  $e$ , та відповідними їм фрагментами зображення  $x$ , тобто функції  $f_G$ , представимо в вигляді

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

називатимемо **локально-адитивними по відношенню до  $w_G$** .

Сформулюємо задачу (1) з урахуванням введених означень.

**Задача 3.1** *Задані вхідне зображення  $x$ , двовимірна контекстно-вільна граMATика  $G$  та локально-адитивна функція штрафу  $f_G : L(G) \times X \rightarrow R$ . Знайти таке зображення  $e^*$  з мови граMATики  $G$ , яке мінімізує величину  $f_G(\bullet, x)$ :*

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

На рис.1 наведений приклад граMATики  $G = \langle V, K, P, \varepsilon \rangle$ , що визначає певну, гранично спрощену множину нотних текстів. Цей приклад наводиться лише як незатьмарена зайвими подробицями ілюстрація основної ідеї побудови таких граMATик.

Множина  $V$  (не зображена на рис. 1) містить еталонні зображення символів та елементів нотного тексту і зображення порожніх, "білих" прямокутників. До множини  $V$  належать ті еталонні зображення, які присутні в записі правил множини  $P$ .

Множина правил  $P$  містить три групи правил: перша група задає структуру нотної сторінки як послідовності нотних рядків загалом, друга — структуру окремого нотного рядка як послідовності символів нотного тексту і, нарешті, третя — структуру акорду як послідовності цілих нот та порожніх місць між ними. Нетермінал  $NT$  використовується в процесі побудови сторінки нотного тексту із нотних рядків та проміжків між ними і позначає побудовану вже частину нотної сторінки. Нотні рядки позначаються нетерміналом  $NS$ , а проміжки — нетерміналом  $WS_1$ . Нотні рядки є послідовністю ключів (нетермінал  $CL$ ), пауз (нетермінал  $PA$ ), тактових рисок (нетермінал  $BL$ ), проміжків між символами

(нетермінал  $WS_2$ ) та акордів (нетермінал  $CH$ ). Акорди є вертикальною послідовністю цілих нот, позначених нетерміналом  $WN$ , та проміжків між ними, яким відповідає нетермінал  $WS_3$ .

Множина  $K$  містить всі нетермінали, які були використані при заданні множини правил  $P$ .

Функція  $w_G$  (також не зображена на рис.1) визначена таким чином, що величина  $w_G(v, x(\Pi))$  рівна кількості пікселів, в яких  $v$  та  $x(\Pi)$  не збігаються. Функція штрафу  $f_G$  є локально-адитивною по відношенню до  $w_G$ .

Як вже було сказано, існує загальний алгоритм розв'язку задачі 3.1 для будь-якої контекстно-вільної двовимірної граматики і будь-якої локально-адитивної функції штрафу. Цей алгоритм може бути використаний і для сконструйованої нами граматики нотних текстів. Він потребує  $O(I^2 J^2 (I + J))$  часу для пошуку виводу вхідного зображення. Вказаний час є досить відчутним для користувача. Але специфіка сконструйованої "нотної" граматики дозволяє побудувати алгоритм, який потребує значно менше —  $O(IJ(I + J))$  обчислювального часу. Алгоритм, про який йдеться, стосується не лише побудованої нами граматики нотних текстів, але значного підкласу контекстно-вільних граматик. Цей підклас, названий нами класом **лінійних граматик з фіксованими розмірами нетерміналів**, та алгоритм граматичного розбору граматик з нього будуть описані в наступному пункті.

#### 4. Клас лінійних граматик з фіксованими розмірами нетерміналів.

**Лема 4.1** (Достатня умова фіксованості висоти нетермінала.)

Нехай  $G = \langle V, K, P, \varepsilon \rangle$  — двовимірна контекстно-вільна граMATика. Нехай нетермінал  $A \in K$  задовільняє таким умовам:

1. Правила заміни, в лівій частині яких стоїть нетермінал  $A$ , в правій частині містять термінали однакової висоти:

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

2. Нетермінал  $A$  не міститься в лівій частині правил вертикальної конкатенації, а правила горизонтальної конкатенації, що містяться в лівій частині  $A$ , мають вигляд:

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

Тоді в процесі виводу будь-якого вхідного зображення в граматиці  $G$  нетермінал  $A$  може позначати фрагменти зображення лише однієї, фіксованої висоти.

**Зауваження 4.1** *Справедлива також аналогічна лема, яка визначає достатню умову фіксованості ширини нетермінала.*

Якщо нетермінал  $A$  може позначати фрагменти лише однієї, фіксованої висоти (ширини), то цю висоту (ширину) називатимемо висотою (шириною) нетермінала  $A$ .

Сформульована лема слугує обґрунтуванням коректності наступного означення.

**Означення 4.1** *Двовимірну контекстно-вільну граматику  $G = \langle V, K, P, \varepsilon \rangle$  називатимемо лінійною граматиною з фіксованими розмірами нетерміналів, якщо її складові задовольняють наступні вимоги:*

1. Множина нетерміналів  $K$  може бути представлена як об'єднання двох підмножин  $K_h$  та  $K_w$ :

$$K = K_h \cup K_w.$$

Підмножина  $K_h$  містить нетермінали, які можуть позначати фрагменти зображення лише заданої, фіксованої висоти.

Підмножина  $K_w$  містить нетермінали, які можуть позначати фрагменти зображення лише заданої, фіксованої ширини.

2. Права частина кожного правила горизонтальної конкатенації містить хоча б один нетермінал з множини  $K_w$ , а права частина кожного правила вертикальної конкатенації — хоча б один нетермінал з  $K_h$ . Таким чином, правила конкатенації мають вигляд:

$$A \rightarrow \alpha | C, A \rightarrow C | \alpha, A \rightarrow \frac{\beta}{C}, A \rightarrow \frac{C}{\beta},$$
$$A, C \in K, \alpha \in K_w, \beta \in K_h.$$

**Зауваження 4.2** *В загальному випадку множини  $K_h$  і  $K_w$  перетинаються. Множина, яка є їх перетином, складається з нетерміналів, які можуть позначати фрагменти зображення лише з фіксованими шириною і висотою.*

У побудованій в попередньому розділі граматиці нотних текстів множини  $K_h$  та  $K_w$

виглядають так:

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

Множина ж правил виводу очевидно задовольняє введене означення.

Розглянемо множину тих фрагментів зображення, які можуть бути позначені нетермінальними символами. На цій множині визначений частковий порядок відносно операції вкладення одного фрагмента в інший. Як усіяка частково впорядкована множина, вона може бути цілком впорядкована без втрати існуючого часткового порядку.

Ми приведемо алгоритм, який, щоб не загроможувати викладення, містить дії, необхідні для обчислення лише значення штрафу  $f_G(e^*, x)$ , де  $e^*$  - розв'язок задачі 3.1 для вхідного зображення  $x$ , лінійної граматики з фіксованими розмірами нетерміналів  $G$  та локально-адитивної функції штрафу  $f_G$ . Але цей алгоритм легко може бути перетворений на такий, що знаходить також саме зображення  $e^*$ , тобто повністю розв'язує задачу 3.1. Необхідні зміни вказані у зауваженні 4.3, наведеному після алгоритму.

1. Впорядкуємо множину тих фрагментів зображення, які можуть бути позначені нетермінальними символами, занумерувавши фрагменти згідно введеного порядку так, що фрагменти з меншими розмірами матимуть менші номери. Позначатимемо ці фрагменти  $x(\Pi_s)$ , де індекс – мала літера  $s$  – означає номер фрагмента. Саму ж множину фрагментів позначимо літерою  $F$ . Крім того, деякі фрагменти позначатимуться індексом, що є великою літерою – позначенням нетермінального символа. Так позначення  $x(\Pi_A)$  та  $x(\Pi_B)$  вказують, що фрагменти  $x(\Pi_A)$  та  $x(\Pi_B)$  позначені нетерміналами  $A$  та  $B$  відповідно.

Розглянемо масив  $g(|F|, |K|)$ , який містить  $|F| \times |K|$  елементів, які приймають значення дійсного типу. Той факт, що елемент масиву  $g(x(\Pi_s), k)$  приймає значення  $r \in R$  означає, що за присвоєння позначки  $k$  фрагменту  $x(\Pi_s)$  було сплачено штраф  $r$ . Проініціюємо масив початковими значеннями  $\infty$ . Позначення  $\infty$  означає, що позначка відповідного нетермінала ще не була присвоєна фрагменту. Вважатимемо також, що  $\infty$  можна порівнювати з дійсними числами, причому  $\infty$  більше за будь-яке з них. Процес ініціалізації ілюструється наступною програмою:

```
for (s = 0; s < |F|; s++)  
  for (k ∈ K)  
    f(x(Πs), k) = ∞
```

2. Вибиратимемо послідовно фрагменти з множини  $F$  і для кожного фрагмента  $x(\Pi_s)$ , кожного нетермінала  $k$  і кожного правила виводу  $p$ , яке містить в лівій частині нетермінал  $k$ , визначатимемо штраф за присвоєння позначки  $k$  фрагменту  $x(\Pi_s)$ .

Так при застосуванні правила горизонтальної конкатенації  $k \rightarrow A|B$  фрагмент  $x(\Pi_s)$  розбивається по горизонталі на два прилеглі один до одного фрагменти  $x(\Pi_A)$  та  $x(\Pi_B)$ .

Дане розбиття фрагмента  $x(\Pi_s)$  на  $x(\Pi_A)$  та  $x(\Pi_B)$  при фіксованому правилі  $p$  може бути зроблене єдиним чином завдяки умові, накладеній на правила виводу означенням 4.1. При розбитті фрагмента  $x(\Pi_s)$  за умови, що нетермінал  $A$  належить до множини  $K_w$ , ширина  $x(\Pi_A)$  покладається рівною ширині нетермінала  $A$ , а лівий верхній кут  $x(\Pi_A)$  збігається з лівим верхнім кутом  $x(\Pi_s)$ . Фрагмент  $x(\Pi_B)$  доповнює фрагмент  $x(\Pi_A)$  до  $x(\Pi_s)$ . Якщо ж навпаки, нетермінал  $B$  належить  $K_w$ , то ширина фрагмента  $x(\Pi_B)$  покладається рівною ширині  $B$ , а правий верхній кут фрагмента  $x(\Pi_B)$  збігається з правим верхнім кутом  $x(\Pi_s)$ . Таке розбиття фрагмента  $x(\Pi_s)$  на  $x(\Pi_A)$  та  $x(\Pi_B)$  в горизонтальному напрямі ми позначатимемо  $x(\Pi_s) = x(\Pi_A) | x(\Pi_B)$ .

Аналогічно визначається розбиття для правил вертикальної конкатенації. Вертикальну конкатенацію фрагментів позначатимемо  $x(\Pi_s) = \frac{x(\Pi_A)}{x(\Pi_B)}$ .

Після розбиття фрагмента  $x(\Pi_s)$  на  $x(\Pi_A)$  та  $x(\Pi_B)$  підраховується число — сума штрафів за вивід лівого (верхнього) і правого (нижнього) фрагментів  $g(x(\Pi_A), A) + g(x(\Pi_B), B)$ . Числа  $g(x(\Pi_A), A)$  та  $g(x(\Pi_B), B)$  вже були обчислені алгоритмом, оскільки фрагменти  $x(\Pi_A)$  та  $x(\Pi_B)$  є частинами фрагмента  $x(\Pi_s)$ , а, отже, згідно введеного відношення повного порядку, є меншими ніж фрагмент  $x(\Pi_s)$ .

При застосуванні правила заміни  $k \rightarrow b, b \in V$  до фрагмента  $x(\Pi_s)$  підраховується число  $w_G(b, x(\Pi_s))$ , яким і ініціалізується елемент  $g(x(\Pi_s), k)$ .

З отриманих чисел (вирахованих для правил заміни та горизонтальної і вертикальної конкатенації) вибирається найменше. Його величина визначає значення штрафу, який записується в

елемент  $g(x(\Pi_s), k)$  масиву  $g$ . Позначивши  $P_h, P_v, P_c$  множини правил горизонтальної і вертикальної конкатенації та правил заміни відповідно, запишемо зазначені дії у вигляді програми:

$$\begin{aligned}
 & \text{for } (s = 0; s < |F|; s++) \\
 & \text{for } (k \in K) \\
 & \{ \\
 & \quad r_h = \min_{\substack{p \in P_h \\ p=(k \rightarrow A|B, A, B \in K) \\ x(\Pi_s) = x(\Pi_A)|x(\Pi_B)}} (g(x(\Pi_A), A) + g(x(\Pi_B), B)) \\
 & \quad r_v = \min_{\substack{p \in P_v \\ p=(k \rightarrow \frac{A}{B}, A, B \in K) \\ x(\Pi_s) = \frac{x(\Pi_A)}{x(\Pi_B)}}} (g(x(\Pi_A), A) + g(x(\Pi_B), B)) \\
 & \quad r_c = \min_{\substack{p \in P_c \\ p=(k \rightarrow b, b \in V)}} (w(b, x(\Pi_s))) \\
 & \quad g(x(\Pi_s), k) = \min(r_h, r_v, r_c) \\
 & \} .
 \end{aligned}$$

3. В результаті роботи алгоритму в елемент  $g(x(T), \varepsilon)$  масиву буде записана величина  $f_G(e^*, x)$ .

**Зауваження 4.3** Розглянутий алгоритм легко може бути модифікований для знаходження не тільки штрафу  $f_G(e^*, x)$ , але й самого зображення  $e^*$  з мови граматики  $G$ , найбільш схожого на вхідне зображення  $x$ . Зображення  $e^*$ , як і будь-яке зображення з мови граматики  $G$ , однозначно задається своїм виводом. Для знаходження виводу в масив  $g$ , окрім штрафу за найкращий вивід фрагмента, слід записувати правило, завдяки якому досягнуте це значення штрафу. Після закінчення роботи алгоритму слід вибрати з масиву  $g$  послідовність тих правил, які призвели до запису певного, відмінного від  $\infty$  штрафу в елемент  $g(x(T), \varepsilon)$  масиву  $g$ . Останнім у шуканому виводі є правило, записане в цьому елементі.

Часова складність розглянутого алгоритму може бути легко оцінена. Масив  $g$  містить  $|F| \times |K|$  елементів. Множина  $F$  містить не більше ніж  $|K_h| |J|^2$  фрагментів, які можуть бути позначені нетерміналами з множини  $K_h$  і не більше ніж  $|K_w| |J|^2$  фрагментів, які можуть бути позначені нетерміналами з  $K_w$ . Отже, масив  $g$  містить  $O(|K| \times |J| (|K_h| |J| + |K_w| |J|))$  елементів. В процесі роботи

алгоритму усі вони переглядаються по одному разу, при цьому в процесі перегляду одного елементу може знадобитись перегляд щонайбільше  $|P|$  правил виводу. Результуюча часова складність алгоритму —  $O(|P||K|IJ(|K_h|J+|K_w|I))$ . За умови фіксованості граматики, а, значить, при фіксованих множинах  $P$  та  $K$  часова складність залежить як  $O(IJ(I+J))$  від розмірів вхідного зображення.

## 5. Демонстрація результатів

Експериментальна перевірка розробленого методу мала на меті верифікацію основних ідей, на яких базується розроблений алгоритм. Тому експерименти виконувались на простих, але реальних музичних текстах, а не на текстах довільної складності. Тексти були взяті із збірника музичних вправ для початкових класів музичної школи. Ці нотні тексти задовільняють такі вимоги:

- акорди в межах одного такту утворюють послідовність, а не дві послідовності, як це є при записі верхнього та нижнього голосу на одному нотному стані;
- нотні тексти не містять ліг;
- нотні тексти не містять форшлагів, динамічних позначень (таких, як форте, піано, крещендо) та інших позначень, які визначають настрій, темп, постановку пальців і т. д.

Приклади таких зображень наведені на рис.2 та 3. Негативні результати експериментів на зображеннях такого класу вимагали би перегляду усієї концепції, покладеної в основу розробленої технології. Позитивні результати дозволяють стверджувати, що подальше вдосконалення програми, тобто розширення класу музичних текстів, вимагатиме лише кількісного вдосконалення технології: збільшення кількості елементарних символів, ускладнення граматики тощо.

Тестування розроблених методів проводилось на 20 зображеннях сторінок нотного тексту, кожна з яких містила 5-10 нотних рядків. Якість розпізнавання характеризується тим, що в середньому із 100 музичних символів 3 були розпізнані неправильно. Не слід розуміти цей факт як те, що ймовірність помилки рівна трьом сотим, бо на даному етапі розробки програмного модуля помилка взагалі не є випадковою величиною. Вона обумовлена загрубленням моделі: програмна реалізація не враховує наявність багатьох музичних символів, загрублені співвідношення, які задають взаємне розташування елементарних музичних символів і т.і. Разом з цим не слід вважати, що практичну цінність має лише програмне забезпечення, яке враховує все розмаїття способів запису музики. Тоді треба було б включити до його складу як мінімум засоби розпізнавання текстів. Практичну цінність має те

програмне забезпечення, яке дозволяє ефективно редагувати результати і виправляти помилки, обумовлені тими чи іншими неврахованими особливостями музичного тексту, за умови, що потреба в такому виправленні не надто велика. Наш програмний комплекс включає засоби як редагування, так і озвучення нотних текстів.

На рис.3 зображено один з прикладів нотних текстів, на яких проводилося тестування. Результати розпізнавання показані на рис.4. Правильно були розпізнані кількість і положення нотних рядків та послідовності символів нотного тексту в усіх нотних рядках.

## Література

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. Coñasnon, 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.

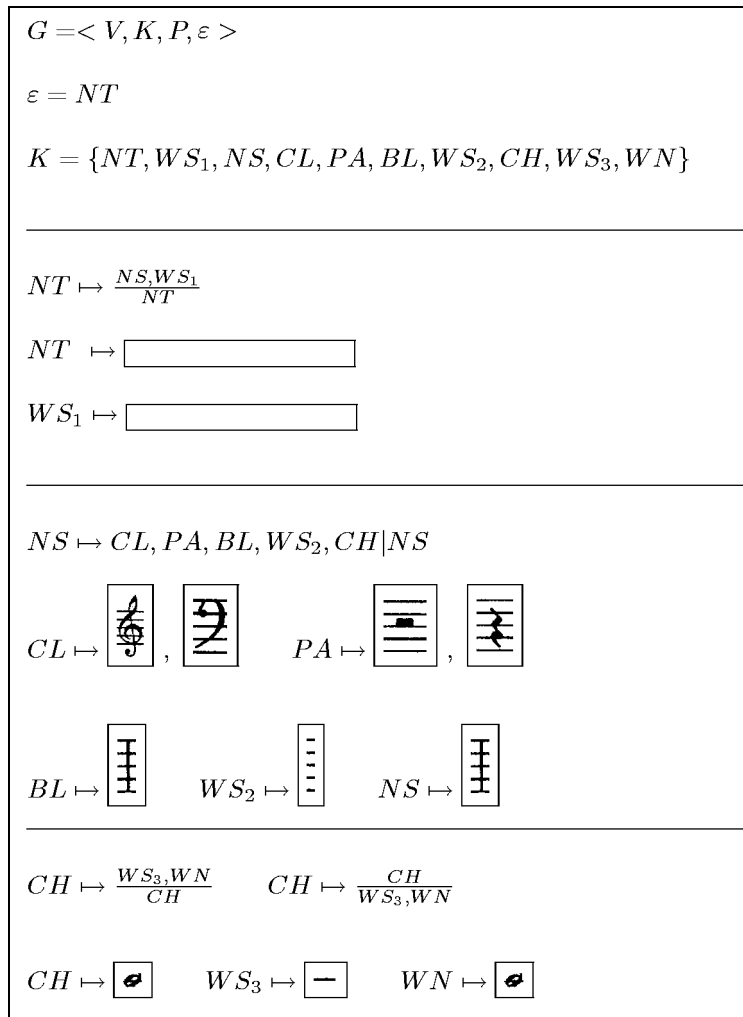


Рис. 1 Граматика G нотних текстів



Рис. 2 Один із тих нотних текстів, на яких проводилось тестування



Рис. 3 Ще один приклад нотного тексту



Рис. 4 Результат розпізнавання тексту, зображеного на рис. 3