Державний комітет зв'язку та інформатизації України

Українська державна академія зв'язку ім. О.С. Попова

Кафедра мереж і систем поштового зв'язку

ОПТИМІЗАЦІЯ СИСТЕМ СОРТУВАННЯ ПОШТОВИХ ВІДПРАВЛЕНЬ

Методичний посібник
до комплексних завдань,
курсових та випускних робіт
з дисциплін поштового циклу
за напрямом 6.0924 - Телекомунікації

ЗАТВЕРДЖЕНО
радою факультету
інформаційних технологій
та поштового зв'язку
Протокол № 3
від 15 грудня 1999 р.

Одеса 2000

УДК 656.8

План НМВ 1999/2000 навч. р.

Рецензент – доцент Л.А. Нікітюк

Укладач - О.Є. Пляцек

Розглянуто питання мінімізації часу розпізнавання поштових індексів за рахунок розпаралелювання обчислень. Наведено приклад реалізації паралельної обробки поштового ярлика на посилках.

Розраховано на студентів старших курсів для виконання комплексних завдань, курсових та випускних робіт з дисциплін поштового циклу.

СХВАЛЕНО
на засіданні кафедри
мереж і систем
поштового зв’язку
і рекомендовано до друку
Протокол № 7
від 13 грудня 1999 р.

ЗМІСТ

Введення

Сортування кореспонденції (листів і важкої пошти) реалізовується комплексами, у складі яких є низка електронних обчислювальних систем. Робота цих систем підтримується відповідним програмним забезпеченням, яке реалізовує набір таких головних функцій:

- синхронізує роботу електронних і механічних блоків;
- зчитує цифри поштового індексу;
- розпізнає поштові індекси;
- розподіляє відправлення по накопичувачах;
- збирає і опрацьовує статистичну інформацію;
- виробляє сигнали сигналізації.

Найбільш трудомістким, з обчислювальної точки зору, є етап розпізнавання поштових індексів. Він використовує максимальною мірою обчислювальні ресурси і забезпечує роботу комплексу в реальному масштабі часу.

Збільшення продуктивності комплексів по сортуванню поштових відправлень, у першу чергу, залежить від швидкості розпізнавання поштового індексу. Тоді задачу мінімізації часу сортування можна сформулювати як розробку методів паралельного опрацювання поштових індексів.

1. Реалізація алгоритмів опрацювання поштових індексів у реальному масштабі часу

Один з головних напрямків у дослідженні питань автоматизації процесу опрацювання і розпізнавання інформації, котра швидко надходить, - створення спеціалізованих високопродуктивних обчислювальних систем (ОС), що працюють у реальному масштабі часу. Одним з найбільш перспективних напрямів для досягнення необхідних показників продуктивності, надійності й економічної ефективності при реалізації алгоритмів розпізнавання поштових індексів є використання багатопроцесорних ОС (БОС, або паралельних процесорів).

Для реалізації розпізнавання індексів на БОС необхідно зробити розпаралелювання алгоритмів і сформувати адекватну архітектуру БОС. Тому слід розв’язати дві тісно пов’язані і взаємозалежні задачі, які потрібно оптимізувати за глибиною розпаралелювання алгоритмів і відповідною їй архітектурою БОС для досягнення потрібної продуктивності й ефективності використання обчислювальних ресурсів.

У загальному вигляді розпаралелювання розуміють як одночасне опрацювання двох чи більш частин одного чи декількох завдань (алгоритмів, команд, мікрокоманд) двома чи більш модулями (комп’ютерами, мікропроцесорами, блоками, вузлами тощо). З означення випливає, що розпаралелювання алгоритмів може здійснюватися на різних рівнях.

Зазвичай виокремлюють два типи паралелізму:

1 паралельне виконання незалежних частин одного завдання декількома модулями;
2 паралельне виконання частин різноманітних завдань різноманітними модулями.

Вибір рівня розпаралелювання і типу паралелізму залежить від характеру алгоритму опрацювання даних. Це призводить до необхідності спільного дослідження алгоритмів і ОС. Такий напрям досліджень сформовано академіком Г.І. Марчуком, він дістав назву «відбиття проблем обчислювальної математики на архітектурі обчислювальних систем».

Підвищення швидкодії в БОС пов’язане з реалізацією двох підходів. Перший пов’язаний із використанням великого числа одночасно працюючих модулів (процесорів), другий - зі збільшенням ефективності використання самих модулів.

Задачу організації паралельного опрацювання цифр поштового індексу можна розділити на два етапи:

1 виявлення алгоритмів опрацювання, що припускають ефективне розпаралелювання;
2 вибір архітектури ОС, найбільш ефективно реалізуючої розпаралелені алгоритми.

2. Розпаралелювання алгоритмів опрацювання поштових індексів

Розпаралелювання алгоритмів потребує урахування таких умов:

1 Розпаралелювання обчислювального процесу, для одержання потрібної ефективності розв’язання, у загальному випадку не потребує максимального паралелізму.
2 Схеми алгоритмів (програм) є ієрархічними структурами, вищий рівень яких - великі блоки, а нижчий рівень - оператори (машинні команди).
3 Процес розпаралелювання має починатися на рівні, що дає максимальний ефект із мінімальними витратами. Для реалізації цієї умови припускаються переходи на більш високі рівні ієрархії і на більш низькі.

Ієрархічна структура алгоритмів пов’язана з різними ступенями деталізації їхнього опису і розпаралелювання на різних рівнях ієрархії алгоритму підтримується різноманітними за ієрархією засобами ОС. Це призводить до необхідності систематизації рівнів розпаралелювання, визначення рівнів ієрархії алгоритмів і відповідних засобів ОС для їхньої підтримки.

Ієрархія рівнів розпаралелювання має такий вигляд:

1 Рівень індексів. На цьому рівні організується одночасне опрацювання двох і більш ярликів з індексами.
2 Рівень цифр. Забезпечується паралельне розпізнавання двох і більш цифр індексу.
3 Рівень ознак. Реалізується паралельне виокремлення ознак цифр.

Перший рівень потребує створення комплексу взаємодіючих читаючих автоматів і пристроїв транспортування поштових відправлень до зчитувальних камер і накопичувачів. Підвищення швидкодії потребує додаткових площ і ОС для синхронізації роботи систем комплексу. Цей рівень потенційно не має великих резервів у підвищенні швидкості опрацювання відправлень.

Найбільш ефективними є рівні цифр і ознак. Підвищення продуктивності в даному випадку потребує розпаралелювання алгоритмів опрацювання поштового ярлика.

Розглянемо розпаралелювання алгоритмів без урахування архітектури і складу апаратних засобів ОС. Будемо вважати, що кожний процесор паралельної ОС має необмежену пам'ять, може працювати незалежно від інших процесорів, але між процесорами існують зв'язки.

Розглянемо можливості організації паралельного опрацювання для кожного з алгоритмів опрацювання поштового ярлика (блок-схеми алгоритмів приведено в додатку). Метою розпаралелювання будемо вважати пошук максимальної ширини (Q) алгоритму, тобто максимально можливого числа одночасно виконуваних операцій.

Алгоритм виокремлення позначок (рис.А1).

Алгоритм виокремлення містить у собі цикл перевірок на приналежність позначкам елементів зображення $a_{i,j} \in G$, що повторюється 361 раз. Отже, максимальна ширина алгоритму $Q1 = 361$. Розпаралелювання інших ділянок алгоритму дає меншу ширину.

Алгоритм ініціалізації позначок (рис.А2).

Алгоритм ініціалізації позначок включає низку обчислювальних операцій. Максимальна ширина алгоритму $Q2 = 6$ пов'язана з незалежними обчисленнями проекцій на осі трьох центрів позначок.

Алгоритм перевірки повного запису зображення (рис.А3).

У алгоритмі перевірки повного запису зображення максимальна ширина $Q3 = 8$ визначається незалежними обчисленнями координат точок $P1...P4$ із наступною перевіркою їхніх значень.

Алгоритм визначення кута повороту (рис.А4) має ширину $Q4 = 3$ при одночасному виконанні блоків 1 і 2.

При визначенні положення знаків індексу (рис.А5) максимальна ширина $Q5 = 36$ (виконання блока 4).

Алгоритм обчислення координат точок початку обходу областей ознак (рис.А8) виконує свої функції для кожного з 18 знаків ярлика. При одночасному виконанні обчислень для всіх знаків $Q6 = 18 \times 14 = 252$.

В алгоритмі виокремлення ознак (рис.А10) розпаралелювання циклу по 18 знаках і по 7 ознаках дає ширину $Q7 = 18 \times 7 = 126$.

В алгоритмі прийняття рішення (рис.А11) одночасне опрацювання кодів ознак 18 знаків дозволяє одержати ширину алгоритму *Q8 = 18.*

Істотні відмінності в значеннях ширини алгоритмів показують, що реалізація максимального розпаралелювання призведе до малої ефективності використання процесорів. ОС значний час буде простоювати через необхідність обміну інформацією між процесорами і їхню неповну завантаженість.

Реалізувати розпаралелювання алгоритмів неможливо без урахування критичних ресурсів ОС (спільна пам'ять, шини й тощо) і способів керування при конфліктах щодо керування, коли різні процесори потребують різноманітних гілок програми.

З цього випливає, що розпаралелювання необхідно робити з урахуванням результатів моделювання алгоритмів і концепції архітектури ОС.

3. Імітаційне моделювання і дослідження часових характеристик роботи алгоритмів

Для визначення найбільш перспективних, з точки зору розпаралелювання, етапів опрацювання поштових ярликів моделювання роботи алгоритмів повинно надати більший обсяг інформації, що дозволить прийняти обґрунтоване рішення.

Моделювання роботи алгоритмів провадилось на крос - асемблерній системі мікропроцесорної системи КР 580.

Програми всіх алгоритмів займали 2226 байтів пам'яті, константи - 4754 байти. Обсяг оперативної пам'яті для зберігання результатів проміжних результатів обчислень - до 512 байтів. Обсяг пам'яті для запису зображення поштового ярлика - 32768 байтів.

Моделювання провадилося по трьох групах алгоритмів:

1 виокремлення позначок (рис. А1.);
2 обчислювальні процедури (рис. А2, А5, А8.);
3 виокремлення ознак і прийняття рішення (рис. А10, А11).

Час роботи алгоритмів вимірювався у машинних тактах. Результати моделювання наведено в табл. 1.

Таблиця 1

Час (такти)	**Група 1**	**Група 2**	**Група 3**
Максимальний час	**284611**	**58830**	**24728236**
Мінімальний час	**245628**	**53228**	**20028751**

Максимальний цикл роботи програми становив 25053953 такти. З них по групах час розподілився в такий спосіб:

Група 1 - 269688 тактів (1,08%),
Група 2 - 56029 тактів (0,22%),
Група 3 - 24728236 тактів (98,7%).

У групі 3 основний час - 23958648 тактів (96,9%) становлять операції з читання елементів з області ознаки і заповнення рядкових регістрів фільтра (табл. А1).

З результатів моделювання очевидно, що більш 98% часу займає виокремлення ознак знаків. Отримані дані дозволяють сформувати напрямок подальших дій, як визначення методів організації паралельного виокремлення ознак знаків і вибір адекватної структури ОС.

4. Архітектура пристроїв на базі багатопроцесорних обчислювальних систем

Найбільш поширеним, при класифікації архітектури багатопроцесорних ОС, є підхід, при якому за основу приймають співвідношення потоків команд, виконуваних системою, і потоків даних, з якими працюють потоки команд.

Вирізняють чотири типи макроархітектур:

1 ОКОД - з одним потоком команд і одним потоком даних. Це класичні однопроцесорні ОС.
2 ОКБД - з одним потоком команд і багатьма потоками даних. Матричні і векторні ОС.
3 БКОД - із багатьма потоками команд і одним потоком даних. Магістральні (конвеєрні) ОС.
4 БКБД - із багатьма потоками команд і багатьма потоками даних. Багатопрограмні мультипроцесорні ОС, мережі ЕОМ.

Така класифікація є умовною і не враховує особливостей поділу критичних ресурсів системи, способів зв’язку між процесорами, складу апаратного забезпечення, але придатна в якості відправного пункту при синтезі ОС. Визначальними у багатопроцесорних ОС є зв’язки між функціональними вузлами ОС, а не обчислення.

У алгоритмі виокремлення ознак (рис. А10.) головний час (більш 95%) витрачається на читання елементів областей ознак і заповнення рядкових регістрів. У максимальному циклі роботи програми для кожної з 126 областей ознак провадиться 196 операцій читання елементів і 187 зсувів рядкових регістрів. Розпаралелювання цього етапу опрацювання дозволяє одержати найбільшу швидкодію ОС.

При розпаралелюванні припускається суміщення опрацювання будь-якої кількості однойменних і (або) різнойменних ознак за однією програмою, тому що можна використовувати відповідні константи корекції напрямку переміщення. Це дозволяє обрати для реалізації програми архітектуру ОС - ОКБД, оскільки виокремлення ознак можна робити за однією програмою на незалежних процесорах.

Критичним ресурсом для обраної архітектури буде пам’ять зображення ярлика поштового індексу. Для зберігання результатів проміжних обчислень кожному процесору доцільно мати власний оперативний запам’ятовувальний пристрій, який використовує при обчисленнях однойменні адреси.

Організація роботи із спільною пам’яттю зображення потребує витрат часу на припинення й активацію процесорів при почерговому читанні елементів зображення, що призводить до простоювань. Ви-

значальним чинником при цьому буде співвідношення часу роботи всіх процесорів за однією програмою і часу роботи за індивідуальною програмою при звертанні до пам'яті. Наявність пам'яті зображення в кожного процесора виключає простоювання, але потребує додаткових витрат устаткування.

Ефективність розпаралелювання програми виокремлення ознак для ОС з ОКБД за наявності спільної пам'яті зображення й індивідуальної пам'яті зображення для кожного процесора можна встановити за допомогою імітаційного моделювання.

5. Моделювання і дослідження часових характеристик

При моделюванні програма виокремлення ознак поділялася на блоки, які виконують однакову кількість циклів.

У процесі моделювання підраховувався час роботи кожного блока програми - ***T.***

Моделювання роботи багатопроцесорної ОС провадилося з урахуванням таких умов:

1 Час роботи блоків звертання до пам'яті зображення ярлика (ПЗ) розраховувався для двох варіантів:
- спільна ПЗ (перший варіант),
- індивідуальна ПЗ (другий варіант).

У першому варіанті враховувались витрати часу на припинення й активацію процесорів при почерговому читанні елементів із ПЗ.

2 Урахування кількості процесорів (***p***) полягало у зміні кількості циклів роботи блоків програми. Некратне співвідношення кількості циклів і кількості процесорів призводить до простоювань частини процесорів в останньому циклі роботи блока.

Прискорення обчислень (S_p) розраховувалось за співвідношенням

$$S_p = T_{\Sigma 1}/T_{\Sigma p},$$

де: $T_{\Sigma 1}$ - сумарний час обчислень на однопроцесорній ОС,
$T_{\Sigma p}$ - сумарний час обчислень на p процесорах.

За результатами моделювання побудовано графіки залежності $S_p = f(p)$, наведені на рис. 1.

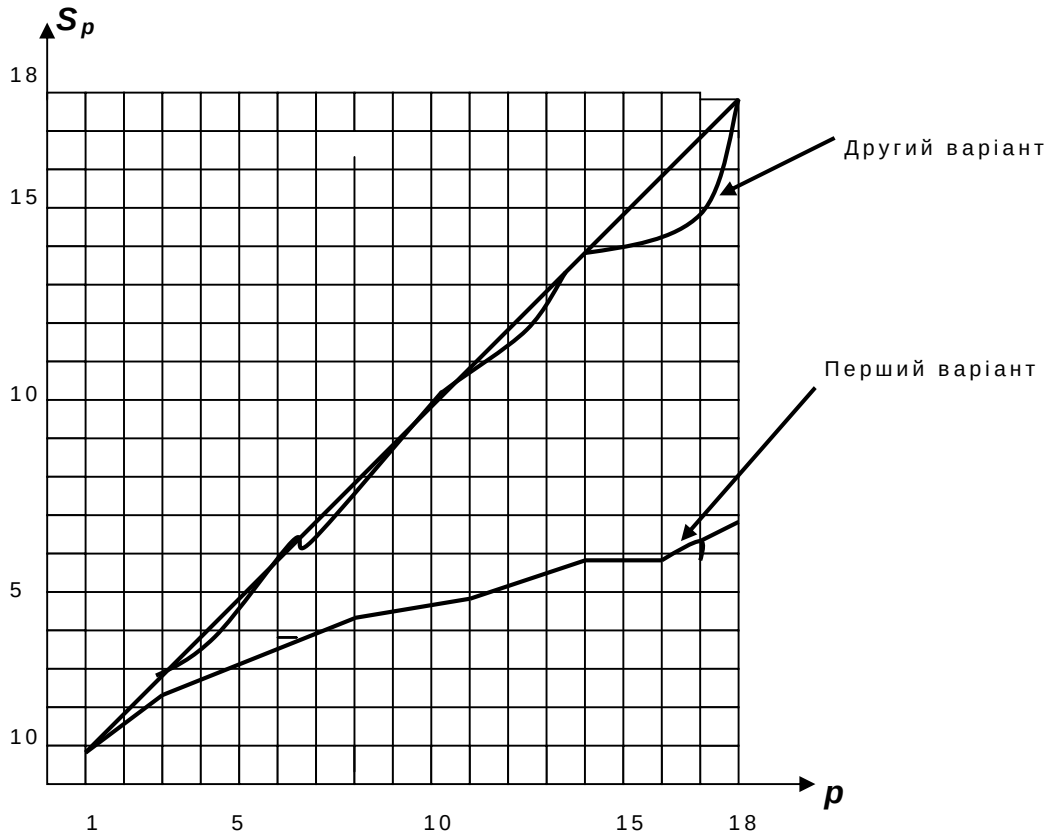

Рисунок 1.

Головні втрати часу в першому варіанті пов’язані з простоюваннями процесорів при звертанні до ПЗ. На ***p*** почергових звертань до пам’яті припадає ***(p+1)*** операція перемикання комутаторів процесорів для організації черги, але до ***p = 3*** результати практично прийнятні. Для досягнення більшого прискорення обчислень необхідно переходити до другого варіанта.

При збільшенні площини зображення обсяг ПЗ може виявитись достатньо великим і наявність декількох (за кількістю процесорів) копій призведе до великих апаратних витрат. У цьому випадку для досягнення необхідного прискорення буде доцільним перехід на верхній рівень ієрархії - рівень індексів.

Організація розпаралелювання алгоритму виокремлення позначок у загальному випадку є недоцільною. Специфіка алгоритму така, що час спільної роботи процесорів є мінімальним. Втрати часу при обміні даними призводять до того, що прискорення обчислень дещо вище за 1, а при спільній ПЗ і менше за 1 (для ОКБД). Використання архітектури БКБД для суміщення роботи процесорів при виокремленні позначок стає недоцільним через порівняно малий час роботи алгоритму.

Розпаралелювання обчислювальних алгоритмів у цілому дасть незначне прискорення обчислень. На фоні сумарного часу розв’язання задачі це прискорення невелике, але дозволить збільшити ефективність завантаження процесорів.

ДОДАТОК А

У описі алгоритмів числові співвідношення і коефіцієнти обрані з урахуванням таких даних (рис. А6):

- **розміри зображення (512 × 512) елементів;**
- **відстані між центрами позначок М1 і М4 – 216...217 елементів, М2 і М3 – 108...109 елементів;**
- **розміри трафарету знаку (20 × 40) елементів;**
- **розміри області ознаки знаку (16 × 12) елементів.**

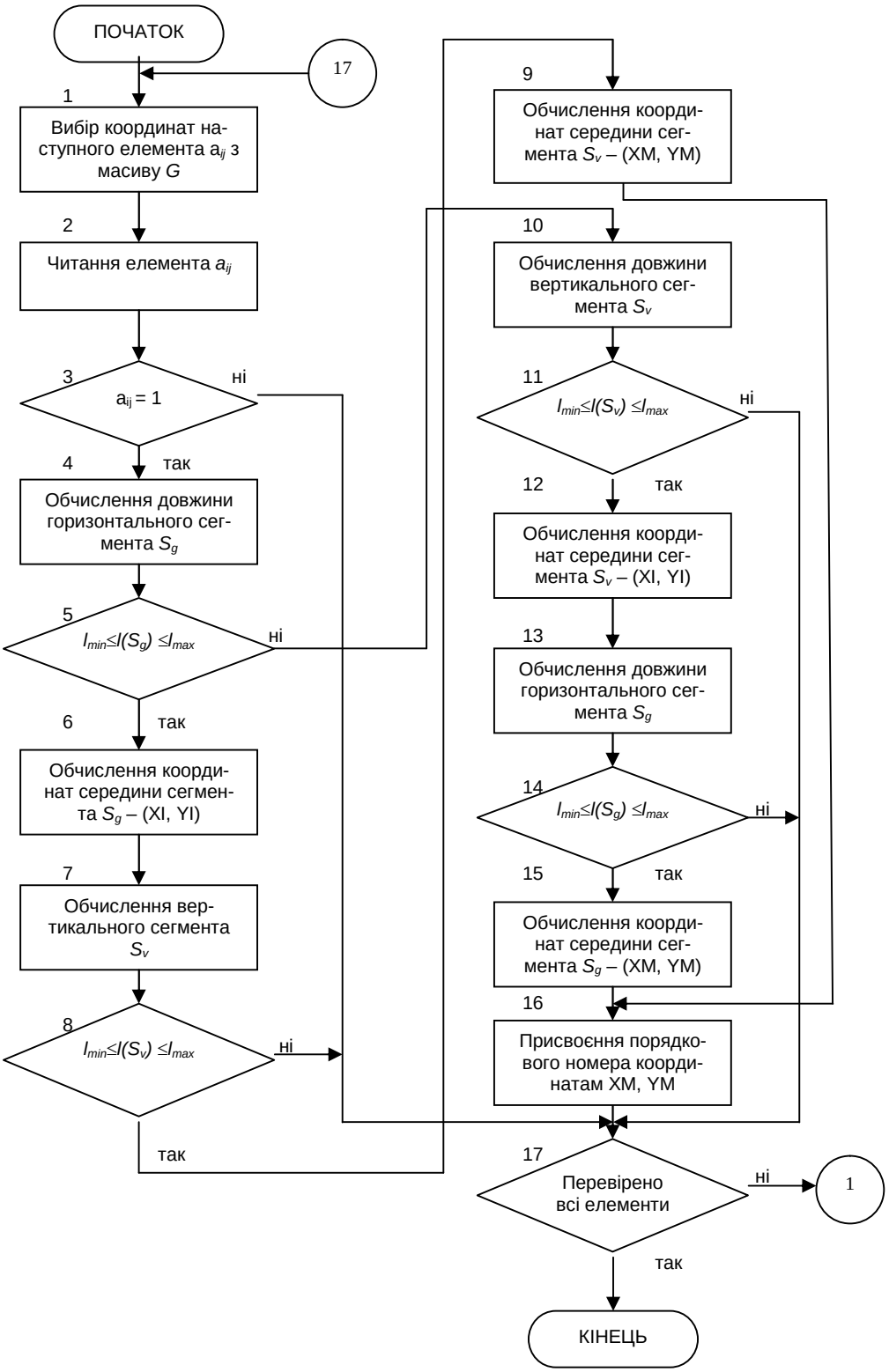

Рисунок А1 - Алгоритм виокремлення позначок

Алгоритм виокремлення позначок виконує наступні операції.
У блоці 1 провадиться вибір координат точок із масиву *G*. Для прийнятих розмірів: $l_{min} = 25$, $G = 361$.
В блоці 2 здійснюється читання чергового елемента зображення.
У блоці 3 перевіряємо приналежність елемента зображенню, якщо «ні» - перехід на блок 17, інакше - до блоку 4.
У блоці 4 обчислюється довжина горизонтального сегмента, що включає прочитаний елемент.
У блоці 5 перевіряємо довжину сегмента. Якщо «ні» - до блока 10, інакше - до блока 6.
У блоці 6 визначаються координати середини сегмента, як півсума *x,y* - координат кінцевих елементів. Дістані координати одержують номер 1.
У блоці 7 визначається довжина вертикального сегмента, що включає елемент із координатами XI, YI.
У блоці 8 перевіряємо довжину сегмента. Якщо «ні» - до блока 17, інакше - до блока 9.
У блоці 10 обчислюється довжина вертикального сегмента, що включає прочитаний елемент.
У блоці 11 перевіряємо довжину сегмента. Якщо «ні» - до блока 17, інакше - до 12.
У блоці 12 обчислюються координати середини вертикального сегмента. Координати дістають номер 1.
У блоці 13 обчислюється довжина горизонтального сегмента, що включає елемент із координатами XI, YI.
У блоці 14 перевіряємо довжину сегмента. Якщо «ні» - до блока 17, інакше - до 15.
У блоці 15 обчислюються координати середини горизонтального сегмента. Координатам присвоюється маркер M (XM, YM).
У блоці 16 координатам (XM, YM) присвоюється порядковий номер.
У блоці 17 перевіряємо, чи прочитано всі елементи *масиву G.* Якщо «ні» - до блока 1, інакше – кінець алгоритму.

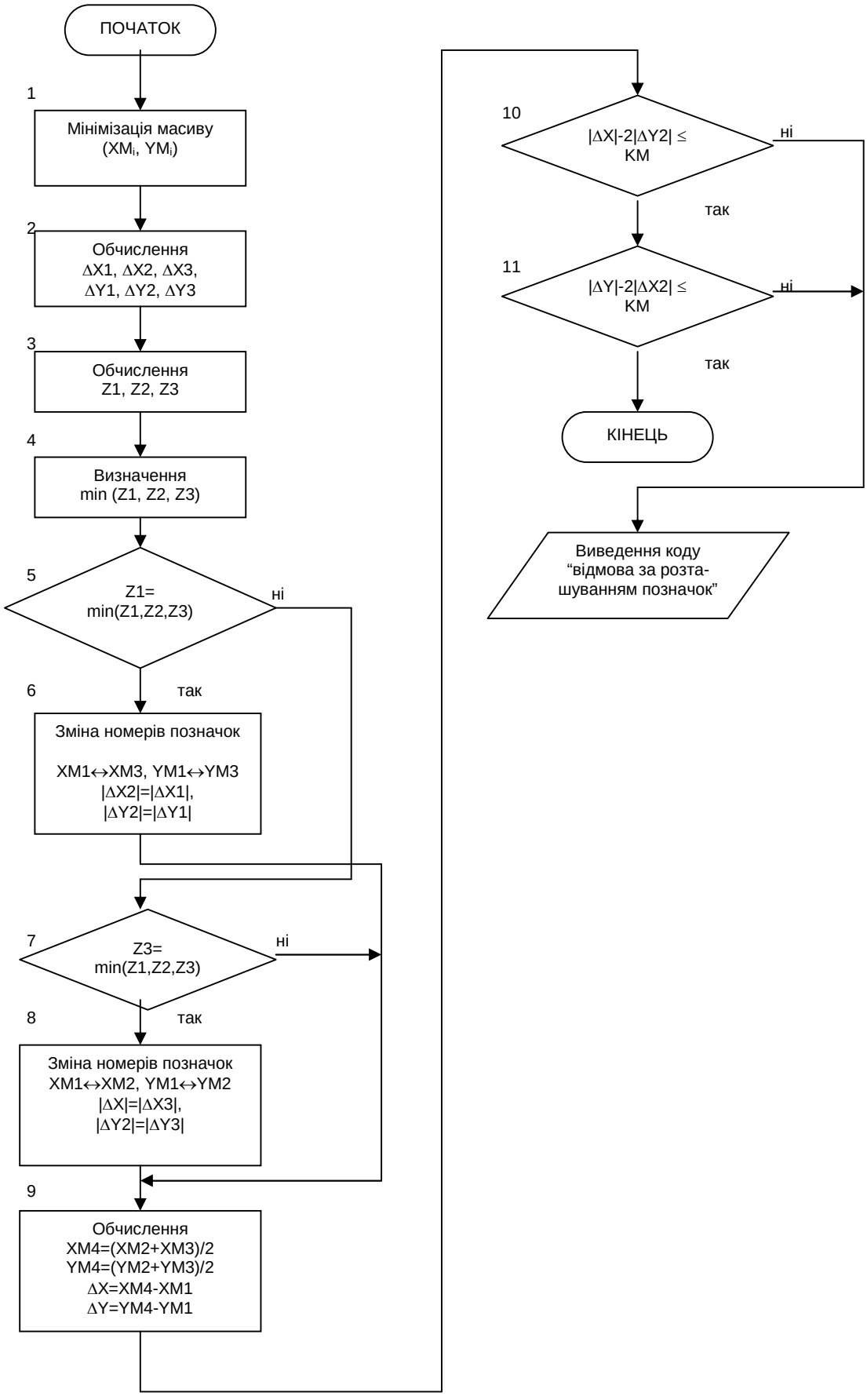

Рисунок А2 - Алгоритм ініціалізації позначок

В алгоритмі ініціалізації позначок виконуються наступні операції.

Отриманий в алгоритмі виокремлення позначок масив координат (XMi, YMi), як правило, містить більше 3 координат точок центрів позначок (Рисунок А2).

У блоці 1 провадиться мінімізація масиву координат центрів позначок. Для цього пари x, y-координат, що відрізняються менше, ніж на 4 елементи, перетворюються на одну пару, як середнє арифметичне їхніх x, y-координат.
В блоці 2 визначаються довжини проекцій центрів позначок на координатні осі.
У блоці 3 обчислюються значення Z1, Z2, Z3 за співвідношеннями:

$$Z1 = |\Delta X1| + |\Delta Y1| = |XM2 - XM1| + |YM2 - YM1|;$$

$$Z2 = |\Delta X2| + |\Delta Y2| = |XM3 - XM2| + |YM3 - YM2|;$$

$$Z3 = |\Delta X3| + |\Delta Y3| = |XM1 - XM3| + |YM1 - YM3|.$$

У блоці 4 визначається мінімальне зі значень Z.
У блоці 5 перевірка мінімальності Z1. Якщо «ні» - до блока 7, інакше - до блока 6.
У блоці 6 провадиться обмін номерів координат і проекцій.
У блоці 7 перевірка мінімальності Z3. Якщо «ні» - до блока 9, інакше - до блока 8.
У блоці 8 провадиться обмін номерів координат і проекцій.
У блоці 9 обчислюються координати центра позначки M4 і проекції на координатні осі.
У блоках 10 і 11 перевіряємо умову правильності розташування позначок. Для прийнятих розмірів КМ = 4. Якщо умови не виконуються - до блока 12, інакше - кінець алгоритму.
Блок 12 здійснює виведення коду відмови за розташуванням позначок.

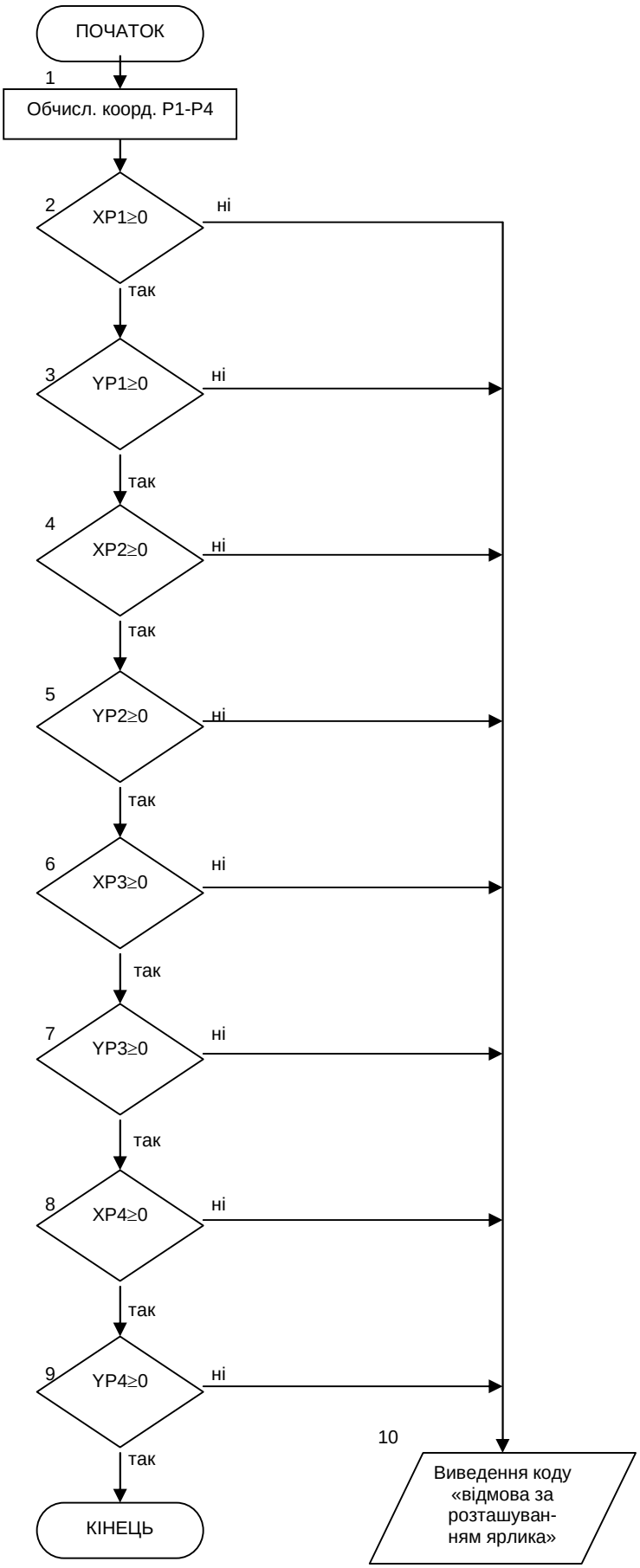

Рисунок А3 - Алгоритм перевірки повного запису зображення

Алгоритм перевірки повного запису зображення виконує наступні операції.

З огляду на те, що значення координат зображення не можуть перевищувати значення 511, вихід координат точок в область негативних значень або переповнювання (за модулем 512) означає неповний запис зображення.

У блоці 1 обчислюються координати точок P1⋯P4 за співвідношеннями:

$$XP1 = XM1 - K\Delta Y, \qquad YP1 = YM1 + K\Delta X,$$

$$XP2 = XM1 - K\Delta Y, \qquad YP2 = YM1 + K\Delta X,$$

$$XP3 = XM4 - K\Delta Y, \qquad YP3 = YM4 + K\Delta X,$$

$$XP4 = XM4 - K\Delta Y, \qquad YP4 = YM4 + K\Delta X.$$

Значення коефіцієнта K = 33/54.

У блоках 2⋯9 перевіряємо умову невід’ємності значень координат. Якщо всі координати позитивні - тоді кінець алгоритму. При негативному значенні хоча б однієї координати - виведення відмови через блок 10.

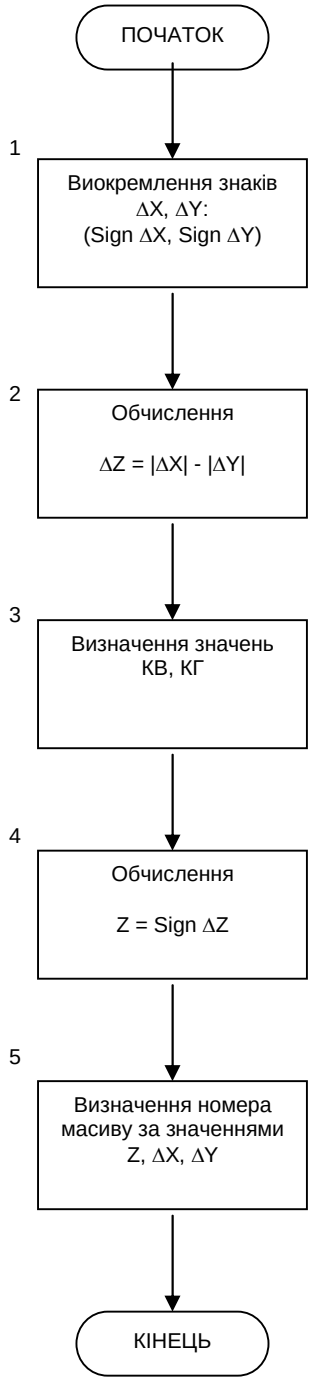

Рисунок А4 - Алгоритм визначення кута повороту

В алгоритмі визначення кута повороту виконуються наступні операції.

У блоці 1 визначаються знаки чисел, що відповідають проекціям на осі координат відрізка М1М4.

В блоці 2 обчислюється значення різниці абсолютних розмірів проекцій Δ X і Δ Y.

У блоці 3 визначаються константи корекції КВ і КГ за значеннями знаків проекцій Δ X, Δ Y і значення їхньої різниці.

У блоці 4 визначається знак різниці абсолютних розмірів проекцій.

У блоці 5 за значеннями знака різниці проекцій і абсолютних розмірів проекцій визначається номер масиву обходу області ознаки.

Після виконання блока 5 - кінець алгоритму.

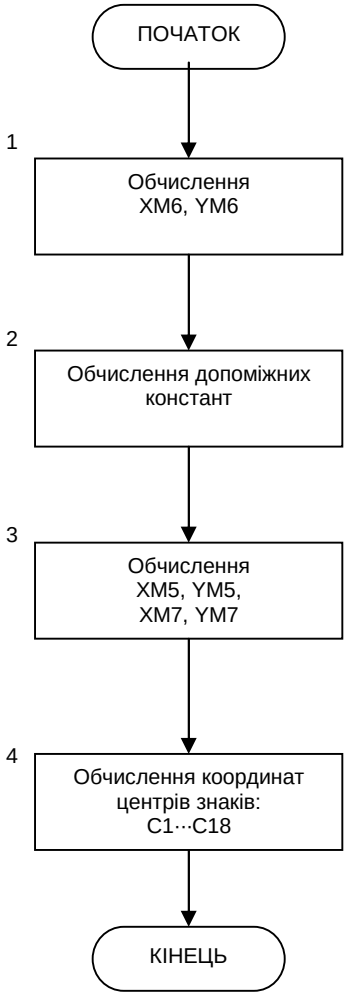

Рисунок А5 - Алгоритм визначення положення знаків індексу

В алгоритмі визначення положення знаків індексу реалізовано наступні операції.

Обчислення координат центрів знаків C1…C18 (Рисунок А6) провадиться через координати умовних точок M5, M6, M7, які розраховуються через координати точок M1, M2, M3, M4 (Рисунок А7).

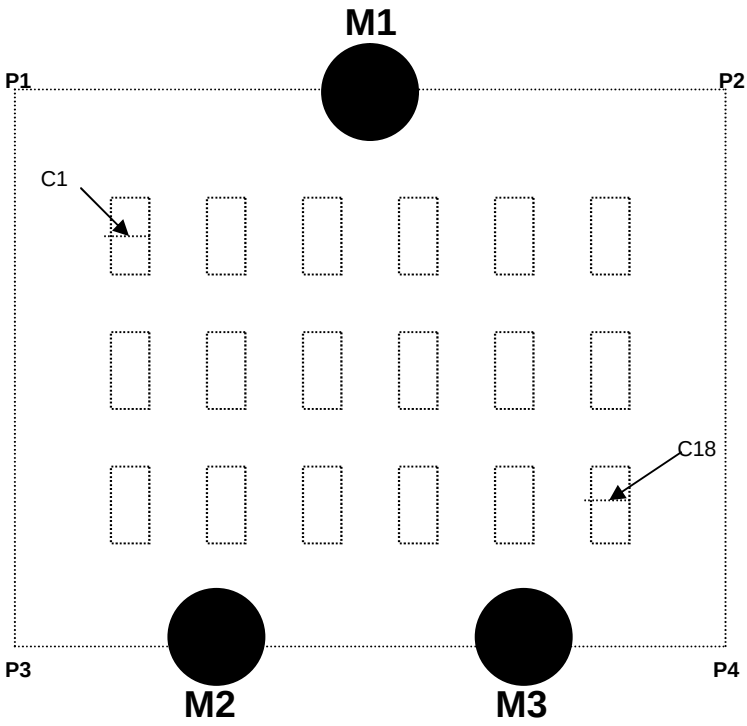

Рисунок А6

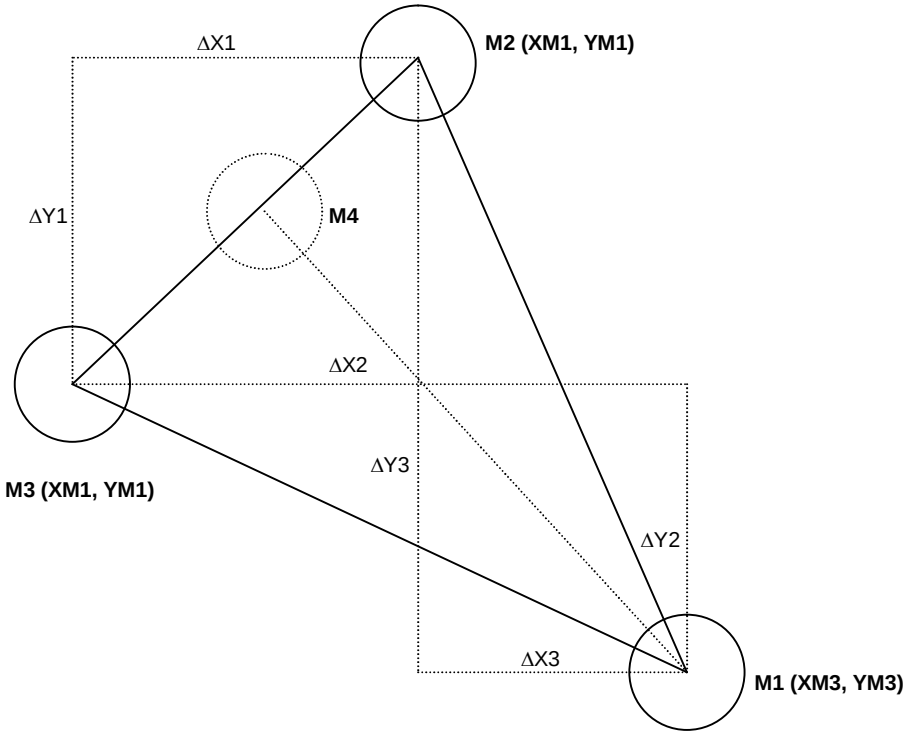

Рисунок А7

У блоці 1 обчислюються координати XM6, YM6 як:

$$XM6 = (XM1 + XM4)/2, \quad YM6 = (YM1 + YM4)/2.$$

В блоці 2 обчислюються допоміжні константи: K1ΔX, K1ΔY, K2ΔX, K2ΔY, K3ΔX, K3ΔY, K4ΔX, K4ΔY. Значення коефіцієнтів: K1 = 4,5/54; K2 = 13/54; K3 = 13,5/54; K4 = 22,5/54.

У блоці 3 обчислюються координати точок M5, M7:

$$XM5 = XM1 + K2\Delta X, \quad YM5 = YM1 + K2\Delta Y,$$
$$XM7 = XM4 + K2\Delta X, \quad YM7 = YM4 + K2\Delta Y.$$

У блоці 4 обчислюються координати центрів знаків C1···C18 як:

$$XC1 = XM5 - K4\Delta Y, \quad YC1 = YM5 + K4\Delta X,$$
$$XC2 = XM5 - K3\Delta Y, \quad YC2 = YM5 + K3\Delta X,$$
$$XC3 = XM5 - K1\Delta Y, \quad YC3 = YM5 + K1\Delta X,$$
$$XC4 = XM5 + K1\Delta Y, \quad YC4 = YM5 - K1\Delta X,$$
$$XC5 = XM5 + K3\Delta Y, \quad YC5 = YM5 - K3\Delta X,$$
$$XC6 = XM5 + K4\Delta Y, \quad YC6 = YM5 - K4\Delta X,$$
$$XC7 = XM6 - K4\Delta Y, \quad YC7 = YM6 + K4\Delta X,$$
$$XC8 = XM6 - K3\Delta Y, \quad YC8 = YM6 + K3\Delta X,$$
$$XC9 = XM6 - K1\Delta Y, \quad YC9 = YM6 + K1\Delta X,$$
$$XC10 = XM6 + K1\Delta Y, \quad YC10 = YM6 - K1\Delta X,$$
$$XC11 = XM6 + K3\Delta Y, \quad YC11 = YM6 - K3\Delta X,$$
$$XC12 = XM6 + K4\Delta Y, \quad YC12 = YM6 - K4\Delta X,$$
$$XC13 = XM7 - K4\Delta Y, \quad YC13 = YM7 + K4\Delta X,$$
$$XC14 = XM7 - K3\Delta Y, \quad YC14 = YM7 + K3\Delta X,$$
$$XC15 = XM7 - K1\Delta Y, \quad YC15 = YM7 + K1\Delta X,$$
$$XC16 = XM7 + K1\Delta Y, \quad YC16 = YM7 - K1\Delta X,$$
$$XC17 = XM7 + K3\Delta Y, \quad YC17 = YM7 - K3\Delta X,$$
$$XC18 = XM7 + K4\Delta Y, \quad YC18 = YM7 - K4\Delta X,$$

Після виконання блока 4 - кінець алгоритму.

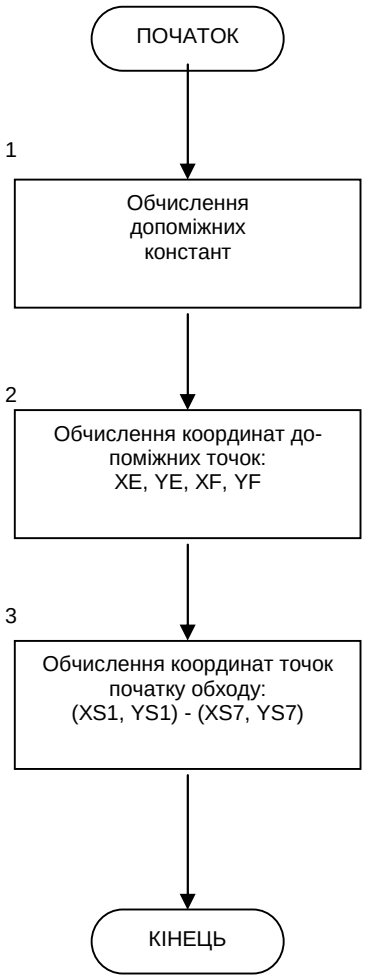

Рисунок А8 - Алгоритм обчислення координат точок початку обходу областей ознак

На рисунку А9 показано дві точки початку обходу областей ознак (інші п'ять точок формуються аналогічно). Точки F і E (збігається з S4) є допоміжними при обчисленнях. У якості значень X_0, Y_0 використовуються координати центра оброблюваного знака *Ci* (i=1⋯18). Значення коефіцієнтів: K1 = 0,5/27; K2 = 2/54; K3 = 4/54; K4 = 5/27; K5 = 6,5/27.

У блоці 1 обчислюються константи: K1ΔX, K1ΔY, K2ΔX, K2ΔY, K3ΔX, K3ΔY, K4ΔX, K4ΔY, K5ΔX, K5ΔY.

У блоці 2 обчислюються координати допоміжних точок E, F:

$$XE = X_0 - K3\Delta Y, \qquad YE = Y_0 - K3\Delta X,$$

$$XF = X_0 - K2\Delta Y, \qquad YF = Y_0 - K2\Delta X.$$

У блоці 3 обчислюються координати точок початку обходу для кожного знака за співвідношеннями:

XS1	=	XE	+	K1ΔX,	YS1	=	YE	- K1ΔY,
XS2	=	XS1	-	K4ΔX,	YS2	=	YS1	+ K4ΔY,
XS3	=	XS1	+	K4ΔY,	YS3	=	YS1	+ K4ΔX,
XS4	=	XS2	+	K4ΔY,	YS4	=	YS2	+ K4ΔX,
XS5	=	XF	+	K5ΔX,	YS5	=	YF	- K5ΔY,
XS6	=	XS5	-	K4ΔX,	YS6	=	YS5	+ K4ΔY,
XS7	=	XS6	-	K4ΔX,	YS7	=	YS6	+ K4ΔY.

Після виконання блока 3 - кінець алгоритму.

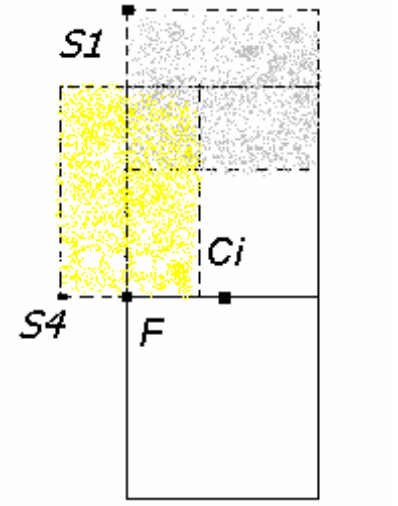

Рисунок А9

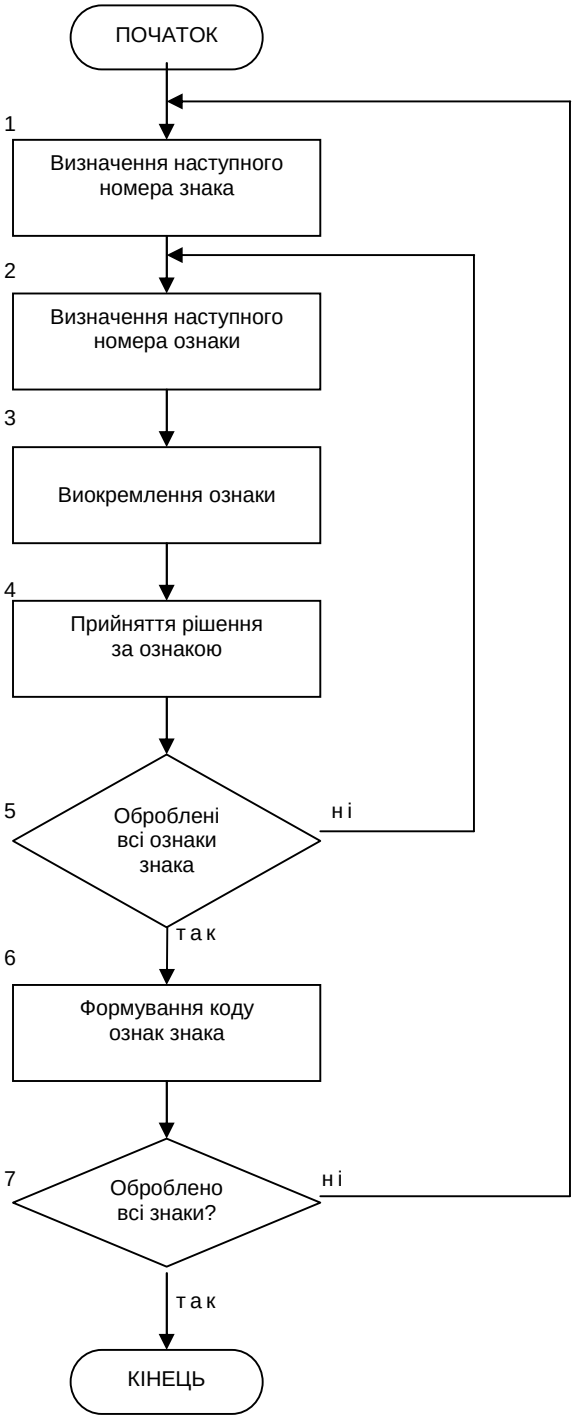

Рисунок А10 - Алгоритм виокремлення ознак

Алгоритмом виокремлення ознак передбачено виконання наступних операцій.

У блоці 1 визначається оброблюваний знак.

У блоці 2 здійснюється послідовний вибір точок початку обходу областей ознак оброблюваного знака.

У блоці 3 елементи зображення, що входять в область ознаки, у послідовності, зумовленій масивом напрямків переходу для отриманого кута повороту, починаючи з точки початку обходу, полінійно зчитуються в ***K*** рядкових регістрів ***R1, R2,…,RK***

(див. таблицю А1). Значення ***K*** (від 4 до 6) записано на початку масиву напрямків переходу і відповідає мінімальному числу рядків, у яких розміщується ознака при її повороті на $\pm 15^0$ щодо напрямного трафарету. Регістри заповнюються зліва праворуч і зверху униз. Після заповнення регістрів їхній уміст логічно складається й в отриманій сумі підраховується кількість одиниць. Максимальна сума після обходу області ознаки запам'ятовується і є числовим значенням відповідної ознаки - ***Пр.***

У блоці 4 здійснюється прийняття рішення за ознакою. Оскільки довжина рядка (***M***) змінюється в залежності від кута повороту, у масиві напрямків переходу задаються пороги p_1 і p_2 для відповідного варіанта кута повороту. При значенні

$Пр < p_1$ - ознака вважається невиокремленою, при $Пр \geq p_2$ - ознака виокремлена, при $p_1 \leq Пр < p_2$ - виокремлено коротку ознаку.

У блоці 5 перевіряємо опрацювання всіх областей ознак знака. Якщо ні - до блока 2, інакше - до блока 6.

У блоці 6 формується код ознак знака, що складається з послідовності значень, привласнених ознакам.

У блоці 7 перевіряємо опрацювання всіх знаків ярлика. Якщо ні - до блока 1, інакше - кінець алгоритму.

Таблиця А1

№п/п	*R1*	*R2*	...	*RK*
1	*RG0(J)*	*RG0(J+1)*	...	*RG0(J+K-1)*
2	*RG1(J)*	*RG1(J+1)*	...	*RG1(J+K-1)*
3	*RG2(J)*	*RG2(J+1)*	...	*RG2(J+K-1)*
...	...	...	...	...
.	...	...	...	...
.	...	...	...	...
M	*RG(M-1)(J)*	*RG(M-1)(J+1)*	...	*RG(M-1)(J+K-1)*

M - кількість елементів у рядку;
K - кількість рядків фільтра;
J - поточний номер рядка області ознаки.

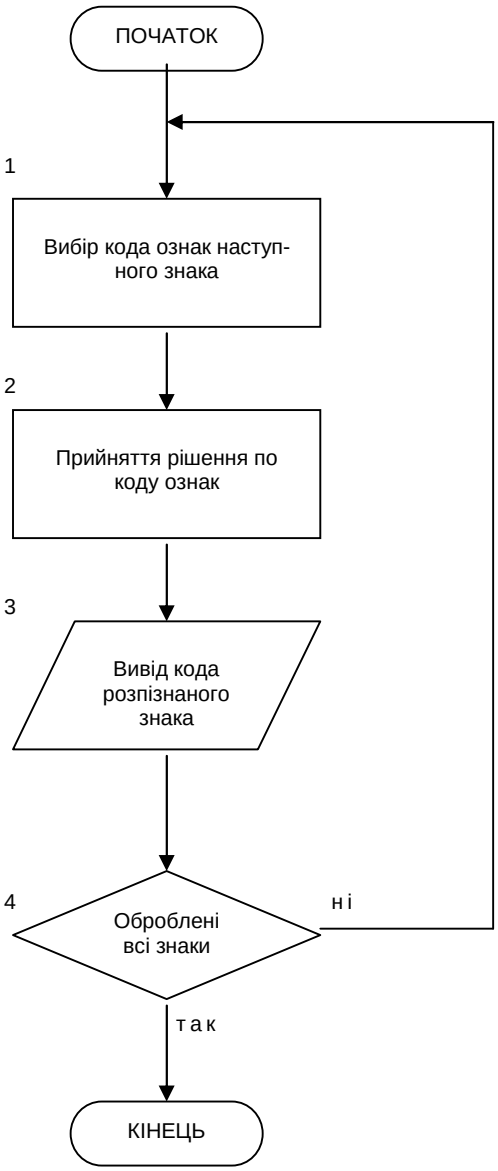

Рисунок А11 - Алгоритм прийняття рішення

Алгоритм прийняття рішення виконує такі операції.

У блоці 1 вибирається код ознак чергового знака.

У блоці 2 приймається рішення по коду ознак знака.

У блоці 3 здійснюється виведення коду результату розпізнавання.

У блоці 4 перевіряємо опрацювання всіх кодів ознак знаків. Якщо ні - до блока 1, інакше - кінець алгоритму.

Редактор І.В. Ращупкіна

Комп’ютерне
макетування Ж.А. Гардиман