НОВЫЕ СРЕДСТВА
КИБЕРНЕТИКИ, ИНФОРМАТИКИ,
ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ
И СИСТЕМНОГО АНАЛИЗА

М.И. ШЛЕЗИНГЕР, Е.В. ВОДОЛАЗСКИЙ, В.М. ЯКОВЕНКО

УДК 519.6

РАСПОЗНАВАНИЕ СХОДСТВА МНОГОУГОЛЬНИКОВ В УСИЛЕННОЙ ХАУСДОРФОВОЙ МЕТРИКЕ

Аннотация. Описан алгоритм распознавания сходства многоугольников в метрике Фреше. Для заданных m-угольника, n-угольника и числа ε алгоритм определяет, превышает ли расстояние между ними порог ε. Известные алгоритмы решают эту задачу за время, линейно зависящее от $(m \times n)\log(m \times n)$, предлагаемый алгоритм — за время порядка $(m \times n)$.

Ключевые слова: вычислительная геометрия, метрика Фреше, усиленная метрика Хаусдорфа, вычислительная сложность, многоугольники.

МЕТРИКА ХАУСДОРФА И ЕЕ УСИЛЕНИЕ

Пусть $\mathbb{R}^k$ — линейное k-мерное пространство с метрикой $d: \mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$, где $d(x, x') = \sqrt{(x - x')^2}$ — расстояние между точками $x \in \mathbb{R}^k$ и $x' \in \mathbb{R}^k$. Пусть D — множество всех возможных ограниченных и замкнутых подмножеств в $\mathbb{R}^k$. Метрикой Хаусдорфа [1] на множестве D называют метрику $h: D \times D \to \mathbb{R}$, которая определяет расстояние $h(X, Y)$ между подмножествами $X \in D$ и $Y \in D$ следующим образом:

$$h(X, Y) = \max\left[\max_{x \in X} \min_{y \in Y} d(x, y), \max_{y \in Y} \min_{x \in X} d(x, y) \right]. \tag{1}$$

Данная метрика применяется в обработке изображений [2–4], анализе циклических процессов, в частности, электрокардиограмм [5]. Как и для любой другой метрики, $[h(X, Y) = 0] \Leftrightarrow [X = Y]$. Однако в некоторых приложениях [2–4, 6] необходимо не только, чтобы расстояние между различными объектами не равнялось нулю, а чтобы оно было не слишком малым, если различие между объектами в том или ином приложении считается существенным. Объясним это требование на примерах, а потом сформулируем его точно.

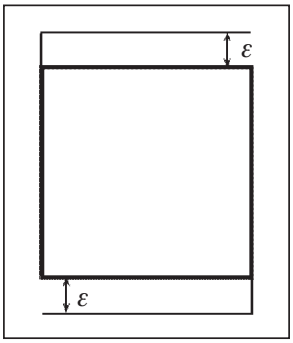

Рис. 1. Хаусдорфова метрика недостаточно сильная: чем меньше ε, тем меньше расстояние

Пример 1. На рис. 1 показаны квадрат, нарисованный жирными линиями, и изображение, представленное как жирными, так и тонкими линиями. Расстояние Хаусдорфа между этими изображениями равно ε и становится сколь угодно малым при уменьшении ε. Тем не менее, эти изображения в определенном смысле существенно отличаются одно от другого при любом, даже очень малом ε. □

Пример 2. Как правило, под кусочно-линейной аппроксимацией кривой понимают ее разбиение на сегменты и последующую замену каждого из них прямолинейным отрезком. При этом необходимо, чтобы количество сегментов было как можно меньшим, а различие между сегментом кривой и соответствующим прямолинейным отрезком не превышало заданный порог. Если это различие определить как расстояние Хаусдорфа, то кривую на рис. 2 можно аппроксимировать одним прямолинейным отрезком. Это противоречит разумному интуитивному пониманию того, какие кривые позволительно заменять прямолинейным отрезком. □

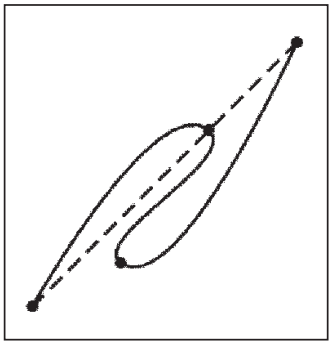

Рис. 2. Неподходящая замена кривой прямолинейным отрезком

Пример 3. Недостаток метрики Хаусдорфа как инструмента для распознавания изображений очевиден при следующем ее эквивалентном определении. Пусть $N(\varepsilon) = \{x \mid d(x, 0) \le \varepsilon\}$ — это ε-окрестность начала координат в $\mathbb{R}^k$, а $P(\varepsilon, X) = = \{x + y \mid x \in X,\ y \in N(\varepsilon)\}$ — операция расширения множества X. Эта операция с так называемым сужением множества образует известную пару «дилатация–эрозия» [7]. Условие $h(X, Y) \le \varepsilon$ эквивалентно выражению

$$[X \subset P(\varepsilon, Y)] \wedge [Y \subset P(\varepsilon, X)], \tag{2}$$

а расстояние Хаусдорфа между множествами X и Y — минимальное значение ε, при котором условие (2) не нарушается. Известно, что пара операций «дилатация–эрозия» исключает мелкие искажения в изображениях [7, 8]. Хаусдорфова метрика игнорирует эти искажения и поэтому ее использование неуместно, когда именно мелкие детали изображений определяют их различие. □

Приведенные примеры мотивируют необходимость усиления хаусдорфовой метрики. Один из возможных способов этого усиления основан на следующих соображениях. Для заданных множеств $X \in D$ и $Y \in D$ обозначим X^Y и Y^X множества функций вида $Y \to X$ и $X \to Y$ соответственно, а также $\varphi^* : X \to Y$ и $\psi^* : Y \to X$ — функции со значениями

$$\varphi^*(x) = \arg\min_{y \in Y} d(x, y), \qquad \psi^*(y) = \arg\min_{x \in X} d(x, y).$$

Расстояние Хаусдорфа (1) в этих обозначениях — это число

$$h(X, Y) = \max\left[\max_{x \in X}\min_{y \in Y} d(x, y),\ \max_{y \in Y}\min_{x \in X} d(x, y)\right] =$$

$$\max\left[\max_{x \in X} d(x, \varphi^*(x)),\ \max_{y \in Y} d(\psi^*(y), y)\right] =$$

$$= \max\left[\min_{\varphi \in Y^X}\max_{x \in X} d(x, \varphi(x)),\ \min_{\psi \in X^Y}\max_{y \in Y} d(\psi(y), y)\right] =$$

$$= \min_{\varphi \in Y^X}\min_{\psi \in X^Y}\max\left[\max_{x \in X} d(x, \varphi(x)),\ \max_{y \in Y} d(\psi(y), y)\right]. \tag{3}$$

В определении (3) на функции φ и ψ не наложены какие-либо ограничения. Заменим метрику h более сильной метрикой H так, что наложим на функции φ и ψ ограничения: функция φ отображает X на Y так, что $\{\varphi(x) \mid x \in X\} = Y$, она непрерывна и обратима, а функция ψ непрерывна и совпадает с φ^{-1}. Обозначим Φ множество функций, удовлетворяющих этим условиям, с их учетом запишем (3) в виде

$$H(X,Y)=\inf_{\varphi\in\Phi}\max[\max_{x\in X}d(x,\varphi(x)),\ \max_{y\in Y}d(\varphi^{-1}(y),\ y)]=\inf_{\varphi\in\Phi}\max_{x\in X}d(x,\varphi(x)),\quad(4)$$

и назовем $H(X,Y)$ усиленным расстоянием Хаусдорфа между множествами $X\in D$ и $Y\in D$. Это расстояние определено только для тех пар X и Y, для которых множество Ф не пусто, в частности, на множестве жордановых кривых. В этом случае оно совпадает с расстоянием Фреше [6].

Основной результат статьи состоит в алгоритме, который для заданных m-угольника X и n-угольника Y, а также числа ε проверяет неравенство $H(X,Y)\le\varepsilon$ за время, линейно зависящее от $m\times n$. Известные в настоящее время алгоритмы [6] имеют сложность порядка $(m\times n)\log(m\times n)$. В конце следующего раздела после определения основных понятий сформулированы дополнительные результаты, вспомогательные для указанного основного.

ЦЕПОЧКИ, ЦИКЛЫ, ЛОМАНЫЕ ЛИНИИ, МНОГОУГОЛЬНИКИ

Определение 1. Назовем последовательность $\bar{x}=(x_0,x_1,\dots,x_m),\ x_i\in\mathbb{R}^k$, цепочкой длины m, а при $x_0=x_m$ — циклом. □

Пусть $T=\{t\in\mathbb{R}\mid 0\le t\le 1\}$.

Определение 2. Монотонный обход полностью упорядоченного множества P — это монотонно неубывающая функция $f_P:T\to P$ такая, что $\{f_P(t)\mid t\in T\}=P$. □

Обозначим $F^m(P)$ множество всех монотонных обходов множества P.

Определение 3. Расстояние между цепочками $\bar{x}=(x_0,x_1,\dots,x_m)$ и $\bar{y}=(y_0,y_1,\dots,y_n)$ — это число

$$H(\bar{x},\bar{y})=\min_{f_I\in F^m(I)}\min_{f_J\in F^m(J)}\max_{t\in T}d(x_{f_I(t)},y_{f_J(t)}),$$

где $I=\{0,1,\dots,m\}$, $J=\{0,1,\dots,n\}$, а $d(x,y)$ — евклидово расстояние между x и y. □

Определение 4. Циклический обход полностью упорядоченного множества P — это функция $f_P:T\to P$, для которой существуют такие $p^*\in P$, $s^*\in S$, $t^*\in T$, что на интервале $\{t\in T\mid t\le t^*\}$ функция f_P является монотонным обходом множества $\{p\in P\mid p\le p^*\}$, а на интервале $\{t\in T\mid t\ge t^*\}$ — монотонным обходом множества $\{p\in P\mid p\ge p^*\}$. □

Обозначим $F^c(P)$ множество всех циклических обходов множества P.

Определение 5. Расстояние между циклами $\bar{x}=(x_0,x_1,\dots,x_m)$ и $\bar{y}=(y_0,y_1,\dots,y_n)$ — это число $H(\bar{x},\bar{y})=\min_{f_I\in F^c(I)}\min_{f_J\in F^c(I)}\max_{t\in T}d(x_{f_I(t)},y_{f_J(t)})$, где $I=\{0,1,\dots,m\}$, $J=\{0,1,\dots,n\}$. □

Определение 6. Для заданной последовательности $\bar{x}=(x_i\mid i\in\{0,1,\dots,m\})$ множество

$$X=\{\alpha\cdot x_{i-1}+(1-\alpha)\cdot x_i\mid 0\le\alpha\le 1,\ i\in\{1,2,\dots m\}\}$$

является ломаной линией, если $\bar{x}$ — цепочка, и многоугольником, если $\bar{x}$ — цикл. □

Точки x_i, $i\in\{0,1,\dots,m\}$, назовем вершинами множества X, а множества $\{\alpha\cdot x_{i-1}+(1-\alpha)\cdot x_i\mid 0\le\alpha\le 1\}$, $i\in\{1,2,\dots,m\}$, — его сторонами. Пусть U и V — суммы длин сторон в X и Y соответственно. Введем обозначения $U_X=\{u\mid 0\le u\le U\}$ и $V_Y=\{v\mid 0\le v\le V\}$. Каждому числу $u\in U_X$ поставим в со-

ответствие точку $x(u) \in X$ такую, что длина участка ломаной линии от x_0 до $x(u)$ равна u. Каждому числу $v \in V_Y$ поставим в соответствие точку $y(v) \in Y$ такую, что длина участка ломаной линии от y_0 до $y(v)$ равна v. Далее используем также обозначения $u(x)$, $x \in X$, для длины участка ломаной линии от x_0 до x, и $v(y)$, $y \in Y$, имеющее аналогичный смысл.

Определение 7. Расстояние между ломаными линиями X и Y — это число

$$H(X, Y) = \min_{u_X \in F^m(U_X)} \min_{v_Y \in F^m(V_Y)} \max_{t \in T} d(x(u_X(t)), y(v_Y(t))). \ \square$$

На множестве ломаных линий приведенное определение эквивалентно сформулированному ранее определению усиленного хаусдорфова расстояния (4).

Определение 8. Расстояние между многоугольниками X и Y — это число

$$H(X, Y) = \min_{u_X \in F^c(U_X)} \min_{v_Y \in F^c(V_Y)} \max_{t \in T} d(x(u_X(t)), y(v_Y(t))). \ \square$$

На множестве многоугольников приведенное определение эквивалентно сформулированному ранее определению усиленного хаусдорфова расстояния (4).

Определение 9. Два множества называются ε-сходными, если расстояние между ними не превышает ε. $\square$

В статье представлены алгоритмы распознавания ε-сходства цепочек, циклов, ломаных линий и многоугольников. Известно, что распознавание ε-сходства цепочек и вычисление расстояния между ними имеют сложность порядка $m \times n$. Что касается циклов и многоугольников, то известные алгоритмы распознавания ε-сходства [6] имеют сложность порядка $(m \times n)\log(m \times n)$. Сложность предлагаемых алгоритмов распознавания ε-сходства циклов и многоугольников имеет порядок $m \times n$, т.е. тот же порядок, что и распознавание ε-сходства цепочек.

ГРАФИКИ ОБХОДОВ

В дальнейшем изложении используется следующее наглядное представление введенных понятий [6]. Представим декартово произведение $P \times S$ двух полностью упорядоченных множеств в виде прямоугольника Π в плоскости $\mathbb{R}^2$, вертикальные стороны которого соответствуют множеству P, а горизонтальные — множеству S. Точка $\pi \in \Pi$ с координатами (p, s) представляет пару: $p \in P$, $s \in S$, причем так, что точка $\pi_1 = (p_1, s) \in \Pi$ расположена левее точки $\pi_2 = (p_2, s) \in \Pi$ при $p_1 < p_2$, а точка $\pi_1 = (p, s_1) \in \Pi$ — ниже точки $\pi_2 = (p, s_2) \in \Pi$ при $s_1 < s_2$.

Определение 10. Подмножество $\gamma^m \subset \Pi$ назовем монотонной кривой, соединяющей точки (p_1, s_1) и (p_2, s_2), $p_1 \le p_2$, $s_1 \le s_2$, если

— для любых точек $(p, s) \in \gamma^m$ и $(p', s') \in \gamma^m$ выполняется либо $p \le p'$, $s \le s'$, либо $p \ge p'$, $s \ge s'$;

— для любого $p \in P$ такого, что $p_1 \le p \le p_2$, существует $s \in S$ такое, что $(p, s) \in \gamma^m$;

— для любого $s \in S$ такого, что $s_1 \le s \le s_2$, существует $p \in P$ такое, что $(p, s) \in \gamma^m$. $\square$

Определение 11. Подмножество $\gamma^c \subset \Pi$ назовем бимонотонной кривой для пары $p^* \in P$ и $s^* \in S$, если γ^c — объединение двух монотонных кривых: одна соединяет точки $(\min P, s^*)$ и $(p^*, \max S)$, а другая — точки $(p^*, \min S)$ и $(\max P, s^*)$. $\square$

Определение 12. Графиком обходов $f_P : T \to P$ и $f_S : T \to S$, монотонных или циклических, назовем подмножество $\{(f_P(t), f_S(t)) \in \Pi \mid t \in T\}$. $\square$

Очевидно, график монотонного обхода — это монотонная кривая, а график циклического обхода — это бимонотонная кривая. Пусть $I = \{0,1,\ldots,m\}$, $J = \{0,1,\ldots,n\}$ — два множества, а $\bar{x} = (x_i \mid i \in I)$, $\bar{y} = (y_j \mid j \in J)$ — две цепочки или два цикла. Пусть X — ломаная линия или многоугольник с вершинами $(x_i \mid i \in I)$, Y — ломаная линия или многоугольник с вершинами $(y_j \mid j \in J)$. Непосредственно из приведенных определений следуют четыре близкие по смыслу утверждения, на которых основаны последующие алгоритмы.

Утверждение 1. Расстояние между цепочками $\bar{x}$ и $\bar{y}$ — это

$$H(\bar{x}, \bar{y}) = \min_{\gamma^m \in \Gamma^m} \max_{(i,j) \in \gamma^m} d(x_i, y_j),$$

где Γ^m — множество монотонных кривых, соединяющих точки $(0,0)$ и (m,n) в прямоугольнике $\Pi = I \times J$. Эти цепочки ε-сходны, если существует монотонная кривая $\gamma^m \in \Gamma^m$ такая, что $d(x_i, y_j) \le \varepsilon$ для всех $(i,j) \in \gamma^m$. □

Утверждение 2. Расстояние между циклами $\bar{x}$ и $\bar{y}$ — это

$$H(\bar{x}, \bar{y}) = \min_{\gamma^c \in \Gamma^c} \max_{(i,j) \in \gamma^c} d(x_i, y_j),$$

где Γ^c — множество бимонотонных кривых в прямоугольнике $\Pi = I \times J$. Эти циклы ε-сходны, если существует бимонотонная кривая $\gamma^c \in \Gamma^c$ такая, что $d(x_i, y_j) \le \varepsilon$ для всех $(i,j) \in \gamma^c$. □

Утверждение 3. Расстояние между ломаными линиями X, Y — это

$$H(X,Y) = \min_{\gamma^m \in \Gamma^m} \max_{(u,v) \in \gamma^m} d(x(u), y(v)),$$

где Γ^m — множество монотонных кривых, соединяющих точки $(0,0)$ и (U,V) в прямоугольнике $\Pi = \{u \mid 0 \le u \le U\} \times \{v \mid 0 \le v \le V\}$, U и V — длины ломаных линий X и Y. Эти ломаные линии ε-сходны, если существует монотонная кривая $\gamma^m \in \Gamma^m$ такая, что $d(x(u), y(v)) \le \varepsilon$ для всех $(u,v) \in \gamma^m$. □

Утверждение 4. Расстояние между многоугольниками X, Y — это

$$H(X,Y) = \min_{\gamma^c \in \Gamma^c} \max_{(u,v) \in \gamma^c} d(x(u), y(v)),$$

где Γ^c — множество бимонотонных кривых в прямоугольнике $\Pi = \{u \mid 0 \le u \le U\} \times$ $\times \{v \mid 0 \le v \le V\}$, U и V — периметры многоугольников X и Y. Эти многоугольники ε-сходны, если существует бимонотонная кривая $\gamma^c \in \Gamma^c$ такая, что $d(x(u), y(v)) \le \varepsilon$ для всех $(u,v) \in \gamma^c$. □

АЛГОРИТМЫ

Распознавание ε-сходства цепочек и вычисление расстояния между ними. Исходными для распознавания ε-сходства цепочек являются множества $I = \{0,1,\ldots,m\}$, $J = \{0,1,\ldots,n\}$ и цепочки $\bar{x} = (x_i \mid i \in I)$, $\bar{y} = (y_j \mid j \in J)$. Эти данные определяют прямоугольник $\Pi = I \times J$ и функцию $q: \Pi \to \{0,1\}$ такую, что $q(i,j) = 1$, если $d(x_i, y_j) \le \varepsilon$, и $q(i,j) = 0$ в противном случае. Точки $\pi_1 \in \Pi$, $\pi_2 \in \Pi$ назовем взаимно достижимыми, если существует монотонная кривая γ такая, что $\pi_1 \in \gamma$, $\pi_2 \in \gamma$ и $q(\pi) = 1$ для всех $\pi \in \gamma$. В соответствии с утверждением 1 ε-сходство цепочек $\bar{x}$ и $\bar{y}$ означает, что точка (m,n) достижима из точки $(0,0)$. Это условие проверяется известными методами динамического программирования. Приведем алгоритм этой проверки для полноты изложения и как базу для последующих алгоритмов.

Пусть $g:\Pi \to \{0,1\}$ — функция, значение $g(i, j)$ которой означает, достижима ли точка $(i, j) \in \Pi$ из точки $(0,0) \in \Pi$. Значения функции $g:\Pi \to \{0,1\}$ вычисляются с помощью следующего алгоритма.

Алгоритм 1

1) $g(0,0) = q(0,0)$;

2) $g(i,0) = q(i,0) \wedge g(i-1,0)$, $i = 1,2,\ldots, m$;

3) $g(0, j) = q(0, j) \wedge g(0, j-1)$, $j = 1,2,\ldots, n$;

4) $g(i, j) = q(i, j) \wedge [g(i-1, j) \vee g(i, j-1) \vee g(i-1, j-1)]$, $i = 1,\ldots, m$; $j = 1,\ldots, n$. □

Результат распознавания ε-сходства представлен числом $g(m, n)$. Сложность алгоритма имеет порядок $m \times n$.

Расстояние между цепочками вычисляется подобным образом. Алгоритм формирует значения $q(i, j) = d(x_i, y_j)$ для всех точек $(i, j) \in \Pi$, а затем — значения

$$g(i, j) = \min_{\gamma \in \Gamma(i, j)} \max_{(i, j) \in \gamma} d(x_i, y_j),$$

где $\Gamma(i, j)$ — множество всех монотонных кривых, начинающихся в $(0,0)$ и заканчивающихся в (i, j). Значения $g(i, j)$ вычисляются с помощью следующего алгоритма.

Алгоритм 2

1) $g(0,0) = q(0,0)$;

2) $g(i,0) = \max\{q(i,0), g(i-1,0)\}$, $i = 1,2,\ldots, m$;

3) $g(0, j) = \max\{q(0, j), g(0, j-1)\}$, $j = 1, 2, \ldots, n$;

4) $g'(i, j) = \min\{g(i-1, j), g(i, j-1), g(i-1, j-1)\}$,
$g(i, j) = \max\{q(i, j), g'(i, j)\}$, $i = 1,\ldots, m$; $j = 1,\ldots, n$. □

Расстояние между цепочками есть число $g(m, n)$, вычисление которого имеет сложность порядка $m \times n$.

Распознавание ε-сходства циклов и вычисление расстояния между ними. Исходными данными при распознавании ε-сходства циклов являются множества $I = \{0,1,\ldots, m\}$, $J = \{0,1,\ldots, n\}$ и циклы $\bar{x} = (x_i \mid i \in I)$, $\bar{y} = (y_j \mid j \in J)$. В соответствии с утверждением 2 ε-сходство циклов означает существование определенной бимонотонной кривой в прямоугольнике $\Pi = I \times J$. Данное условие можно модифицировать так, чтобы вместо поиска бимонотонной кривой в прямоугольнике $\Pi = I \times J$ проверять существование монотонной кривой в прямоугольнике другого вида. Для множеств $I = \{0,1,\ldots, m\}$ и $J = \{0,1,\ldots, n\}$ определим прямоугольник $\Pi 2$ как декартово произведение $\{0,1,\ldots, m\} \times \{0,1,\ldots, 2 \times n\}$. Циклический обход множеств I и J можно представить как монотонный обход множеств $\{0,1,\ldots, m\}$ и $\{j^*, j^*+1,\ldots, j^*+n\}$, где точка $(0, j^*)$ — начало обхода. График циклического обхода, в свою очередь, можно представить монотонной кривой в $\Pi 2$, начинающейся в точке $(0, j^*)$ и заканчивающейся в точке $(m, n+j^*)$.

Определим функцию $q:\Pi 2 \to \{0,1\}$ так, что для $j \le n$ функция q принимает значение $q(i, j) = 1$, если $d(x_i, y_j) \le \varepsilon$, и $q(i, j) = 0$, если $d(x_i, y_j) > \varepsilon$, а для $j \ge n$ — значения $q(i, j) = q(i, j-n)$. В этих терминах ε-сходство циклов означает существование такого j^* и такой монотонной кривой, начинающейся в $(0, j^*)$ и заканчивающейся в $(m, n+j^*)$, что $q(i, j) = 1$ во всех точках этой кривой. Иными словами, ε-сходство циклов означает существование такого j^*, что точка $(m, n+j^*)$ достижима из точки $(0, j^*)$.

Для каждой точки $(i, j) \in \Pi 2$ определим число $up(i, j)$, равное максимальному значению j', при котором (i, j) достижима из $(0, j')$. Если такого j' не сущес-

твует, то $up(i, j) = (-1)$. Подобным образом каждой точке $(i, j) \in \Pi 2$ поставим в соответствие число $up'(i, j)$, равное максимальному значению j', при котором (m, j') достижима из (i, j), и равное (-1), если такого j' не существует. Очевидно, что если точка $(m, n + j^*)$ достижима из $(0, j^*)$, то $up(m, n + j^*) \geq j^*$ и $up'(0, j^*) \geq n + j^*$. Менее очевидно, однако верно, что если $up(m, n + j^*) \geq j^*$ и $up'(0, j^*) \geq n + j^*$, то $(m, n + j^*)$ достижима из $(0, j^*)$. Поэтому проверка ε-сходства циклов сводится к отысканию j^*, удовлетворяющего неравенства $up(m, n + j^*) \geq j^*$ и $up'(0, j^*) \geq n + j^*$. При известных числах $up(i, j)$ и $up'(i, j)$ сложность этого поиска имеет порядок n. Числа $up(i, j)$ вычисляются за время, линейно зависящее от $m \times n$, с помощью следующего алгоритма.

Алгоритм 3

1) если $q(0,0) = 0$, то $up(0,0) = (-1)$;
   в противном случае $up(0,0) = 0$;
2) для $i = 1, 2, \ldots, m$,
   если $q(i,0) = 0$, то $up(i,0) = (-1)$;
   в противном случае $up(i,0) = up(i-1,0)$;
3) для $j = 1, 2, \ldots, 2n$,
   если $q(0, j) = 0$, то $up(0, j) = (-1)$;
   в противном случае $up(0, j) = j$;
4) для $i = 1, 2, \ldots, m$ и $j = 1, 2, \ldots, 2n$,
   если $q(i, j) = 0$, то $up(i, j) = (-1)$;
   в противном случае $up(i, j) = \max\{up(i-1, j), up(i, j-1), up(i-1, j-1)\}$.

□

Аналогично вычисляются и числа $up'(i, j)$, таким образом, сложность распознавания ε-сходства циклов имеет порядок $m \times n$.

Вычисление расстояния между циклами сводится к многократному распознаванию ε-сходства, которое выполняется для определенных тестовых значений ε. Искомое значение расстояния принадлежит множеству $\{d(x_i, y_j) \mid i \in I, j \in J\}$, состоящему не более чем из $m \times n$ элементов. Это множество следует упорядочить за время порядка $(m \times n)\log(m \times n)$, а затем найти искомое расстояние, определяя ε-сходство не более чем для $\log(m \times n)$ тестовых значений. Таким образом, вычисление расстояния между циклами имеет порядок сложности не хуже, чем $(m \times n)\log(m \times n)$.

Распознавание ε-сходства ломаных линий. Исходные данные для распознавания ε-сходства ломаных линий X и Y представлены множествами $I = \{0, 1, 2, \ldots, m\}$ и $J = \{0, 1, 2, \ldots, n\}$ номеров вершин и их координатами $(x_i \mid i \in I)$, $(y_j \mid j \in J)$. Эти данные определяют длины U и V ломаных линий, прямоугольник $\Pi = \{u \mid 0 \leq u \leq U\} \times \{v \mid 0 \leq v \leq V\}$ и его подмножество $\Pi(\varepsilon) = \{(u, v) \in \Pi \mid d(x(u), y(v)) \leq \varepsilon\}$. Число u — горизонтальная координата точки $(u, v) \in \Pi$, а число v — ее вертикальная координата. В соответствии с утверждением 3 ε-сходство ломаных линий означает, что при этом значении ε точка (U, V) достижима из точки $(0, 0)$. Как и при распознавании сходства цепочек (см. алгоритм 1), основная идея распознавания ломаных линий состоит в последовательном наращивании множества точек, достижимых из $(0, 0)$. Однако в данном случае Π, $\Pi(\varepsilon)$ и области достижимости — это не конечные, а бесконечные множества. Покажем, что и в этом случае распознавание ε-сходства сводится к конечным вычислениям.

Представим прямоугольник Π как объединение $m \times n$ подмножеств $\Pi(i, j) = \{(u, v) \mid u(x_{i-1}) \leq u \leq u(x_i),\ v(y_{j-1}) \leq v \leq v(y_j)\}$, $i \in \{1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, n\}$, которые назовем ячейками. В силу выпуклости расстояния

d: $\mathbb{R}^k \times \mathbb{R}^k \to \mathbb{R}$ множество $\Pi(\varepsilon) \cap \Pi(i, j)$ выпукло для каждой пары (i, j). Пусть γ — монотонная кривая, начинающаяся в $(0,0)$, заканчивающаяся в (U, V) и полностью лежащая в $\Pi(\varepsilon)$. Пусть $\gamma(i, j) = \gamma \cap \Pi(i, j)$ — участок этой кривой в ячейке $\Pi(i, j)$. Определим другую монотонную кривую γ' так, что если $\gamma(i, j) = \varnothing$, то $\gamma'(i, j) = \varnothing$, иначе $\gamma'(i, j)$ — это прямолинейный отрезок, концы которого совпадают с концами $\gamma(i, j)$. Поскольку $\gamma(i, j) \subset \Pi(\varepsilon) \cap \Pi(i, j)$, а множество $\Pi(\varepsilon) \cap \Pi(i, j)$ — выпукло, то $\gamma'(i, j) \subset \Pi(\varepsilon) \cap \Pi(i, j)$. Следовательно, γ' — тоже монотонная кривая, начинающаяся в $(0,0)$, заканчивающаяся в (U, V) и полностью лежащая в $\Pi(\varepsilon)$. Таким образом, поиск подходящей монотонной кривой сводится к поиску конечной последовательности точек, в которых искомая кривая пересекает границы ячеек. Для поиска этой последовательности следует находить не множество g всех точек, достижимых из $(0,0)$, а только его пересечения с границами ячеек $\Pi(i, j)$.

Введем обозначения

$$qhor(i, j) = \{u \mid (u, v(y_j)) \in \Pi(\varepsilon) \cap \Pi(i, j)\}, \ qver(i, j) = \{v \mid (u(x_i), v) \in \Pi(\varepsilon) \cap \Pi(i, j)\}.$$

Подмножества $qhor(i, j)$ и $qver(i, j)$ — это замкнутые интервалы (возможно, пустые), и их можно представить конечной совокупностью данных, объем которой имеет порядок $m \times n$. Каждый интервал представляется бинарной меткой, обозначающей, пустой ли он, и двумя его граничными точками, если он не пустой. Сложность вычисления этих данных имеет порядок $m \times n$.

Пересечения множества g точек, достижимых из $(0,0)$, с границами ячеек $\Pi(i, j)$ тоже будем задавать с помощью интервалов $ghor(i, j) \subset qhor(i, j)$ и $gver(i, j) \subset qver(i, j)$:

$$ghor(i, j) = \{u \mid (u, v(y_j)) \in g \cap \Pi(i, j)\}, \ gver(i, j) = \{v \mid (u(x_i), v) \in g \cap \Pi(i, j)\}.$$

Их тоже можно представить совокупностью чисел, объем которой имеет порядок $m \times n$. Интервалы $ghor(i, j)$, $gver(i, j)$ строятся на основании интервалов $qver(i, j)$ и $qhor(i, j)$ с помощью следующего алгоритма.

Алгоритм 4

1) если $d(x_0, y_0) > \varepsilon$, то $ghor(1,0) = gver(0,1) = \varnothing$;
   в противном случае $ghor(1,0) = qhor(1,0)$, $gver(0,1) = qver(0,1)$;
2) для $i \in \{2, 3, \ldots, m\}$,
   если $ghor(i-1,0) \cap qhor(i,0) = \varnothing$, то $ghor(i,0) = \varnothing$;
   в противном случае $ghor(i,0) = qhor(i,0)$;
3) для $i \in \{2, 3, \ldots, n\}$,
   если $gver(0, j-1) \cap qver(0, j) = \varnothing$, то $gver(0, j) = \varnothing$;
   в противном случае $gver(0, j) = qver(0, j)$;
4) для $i \in \{1, 2, \ldots, m\}$ и $j \in \{1, 2, \ldots, n\}$,
   если $ghor(i, j-1) \neq \varnothing$, то $gver(i, j) = qver(i, j)$;
   в противном случае $gver(i, j) = \{v \in qver(i, j) \mid v \geq \min gver(i-1, j)\}$;
   если $gver(i-1, j) \neq \varnothing$, то $ghor(i, j) = qhor(i, j)$;
   в противном случае $ghor(i, j) = \{u \in qhor(i, j) \mid u \geq \min ghor(i, j-1)\}$. □

Условия вида $x \geq \min P$ в п. 4 алгоритма 4 проверяются и для $P = \varnothing$. В этом случае предполагается, что неравенство $x \geq \min P$ ложно для любого x и, следовательно, $\{x \mid x \geq \min P\} = \varnothing$. Результат распознавания ε-сходства ломаных линий представлен подмножествами $ghor(m, n)$ и $gver(m, n)$. Точка (U, V) либо содержится в обоих подмножествах, либо не содержится ни в одном из них. В первом случае $H(X, Y) \leq \varepsilon$, и $H(X, Y) > \varepsilon$ — во втором. Вычислительная сложность алгоритма имеет порядок $m \times n$.

В [6] показано, что вычисление расстояния между ломаными линиями можно свести к многократному, но не более чем $\log(m \times n)$-кратному распознаванию ε-сходства.

Распознавание ε-сходства многоугольников. Пусть X и Y — многоугольники с периметрами U и V, $I = \{0,1,\ldots,m\}$ и $J = \{0,1,\ldots,n\}$ — множества номеров их вершин, $(x_i \in \mathbb{R}^k \mid i \in I)$ и $(y_j \in \mathbb{R}^k \mid j \in J)$ — координаты этих вершин. Согласно утверждению 4 ε-сходство этих многоугольников означает существование бимонотонной кривой в прямоугольнике $\Pi = \{u \mid 0 \le u \le U\} \times \{v \mid 0 \le v \le V\}$, для каждой точки (u, v) которой выполняется неравенство $d(x(u), y(v)) \le \varepsilon$. Как и при распознавании ε-сходства циклов, поиск бимонотонной кривой в прямоугольнике Π можно заменить поиском монотонной кривой в прямоугольнике $\Pi 2 = \{u \mid 0 \le u \le U\} \times \{v \mid 0 \le v \le 2V\}$. А именно, два многоугольника ε-сходны, если существует такое число v^*, что точка $(U, V + v^*)$ достижима из точки $(0, v^*)$. Для анализа этого условия сформулируем вспомогательные понятия.

Любой паре $x \in X$, $y \in Y$ соответствуют две точки в прямоугольнике $\Pi 2$: $(u(x), v(y))$ и $(u(x), V + v(y))$, где $u(x)$ — длина ломаной линии от x_0 до x, а $v(y)$ — длина ломаной линии от y_0 до y. Каждой точке $(u, v) \in \Pi 2$ соответствует пара $x(u) \in X$, $y(v) \in Y$ такая, что длина ломаной линии от x_0 до $x(u)$ равна u, а длина ломаной линии от y_0 до $y(v)$ равна v, если $v \le V$, и равна $v - V$, если $v \ge V$. Для $i \in \{1,2,\ldots,m\}$, $j \in \{0,1,\ldots,n\}$ определим подмножества

$$\Pi hor(i, j) = \{(u, v(y_j)) \mid u(x_{i-1}) \le u \le u(x_i)\},$$

$$\Pi hor(i, j+n) = \{(u, v+V) \mid (u, v) \in \Pi hor(i, j)\}.$$

Подобные подмножества определим для $i \in \{0,1,\ldots,m\}$ и $j \in \{1,2,\ldots,n\}$ так, что

$$\Pi ver(i, j) = \{(u(x_i), v) \mid v(y_{j-1}) \le v \le v(y_j)\},$$

$$\Pi ver(i, j+n) = \{(u, v+V) \mid (u, v) \in \Pi ver(i, j)\}.$$

Точку (u^*, v^*) назовем достижимой слева, если существует v такое, что (u^*, v^*) достижима из $(0, v)$, и достижимой справа, если существует v такое, что (U, v) достижима из (u^*, v^*). Обозначим g множество точек, достижимых слева, и g' — достижимых справа. Для каждой точки $(u^*, v^*) \in g$ определим число $up(u^*, v^*)$ — максимальное значение v, при котором (u^*, v^*) достижима из $(0, v)$. Для каждой точки $(u^*, v^*) \in g'$ определим число $up'(u^*, v^*)$ — максимальное значение v, при котором (U, v) достижима из (u^*, v^*).

Значения $up(u, v)$ и $up'(u, v)$ аналогичны числам $up(i, j)$ и $up'(i, j)$ при распознавании ε-сходства циклов. Очевидно, что если точка $(U, V + v^*)$ достижима из $(0, v^*)$, то $up(U, V + v^*) \ge v^*$ и $up'(0, v^*) \ge V + v^*$. Менее очевидно, однако верно, что если $up(U, V + v^*) \ge v^*$ и $up'(0, v^*) \ge V + v^*$, то $(U, V + v^*)$ достижима из $(0, v^*)$. Поэтому ε-сходство многоугольников равнозначно существованию такого числа v^*, что

$$(0, v^*) \in g',\ up'(0, v^*) \ge V + v^*,\ (U, V + v^*) \in g,\ up(U, V + v^*) \ge v^*. \quad (5)$$

Как и для распознавания ε-сходства ломаных линий, для проверки условия (5) не требуется знать полностью множества g и g', а только их подмножества

$$gver(i,\, j) = g \cap \Pi ver(i,\, j)\,,\ \ ghor(i,\, j) = g \cap \Pi hor(i,\, j)\,, \tag{6}$$

$$gver'\,(i,\, j) = g' \cap \Pi ver(i,\, j)\,,\ \ ghor'\,(i,\, j) = g' \cap \Pi hor(i,\, j)\,. \tag{7}$$

Те подмножества (6), (7), которые не пустые, являются вертикальными или горизонтальными отрезками в прямоугольнике $\Pi 2$. Каждое из подмножеств (6), (7) задается бинарной меткой, обозначающей, является ли оно пустым, и координатами крайних точек. Суммарный объем этих данных имеет порядок $m \times n$. Подмножества (6), (7) строятся подобно тому, как строятся подмножества $gver(i,\, j)$, $ghor(i,\, j)$ с помощью алгоритма 4 при распознавании ε-сходства ломаных линий. Трудоемкость этих вычислений имеет тот же порядок $m \times n$.

Функции up и up' также не требуется знать полностью. Для проверки условия (5) достаточно знать сужения up на подмножества $gver(i,\, j)$, $ghor(i,\, j)$ и сужения up' на подмножества $gver'\,(i,\, j)$, $ghor'\,(i,\, j)$. Их в свою очередь можно представить конечной совокупностью чисел. Функция up' на любом подмножестве $gver'\,(i,\, j)$ принимает постоянное значение, которое обозначим $UPver'\,(i,\, j)$. На каждом множестве $ghor(i,\, j)$ функция up принимает постоянное значение, которое обозначим $UPhor(i,\, j)$. Сужения функции up на подмножества $gver(i,\, j)$ и функции up' на подмножества $ghor'\,(i,\, j)$ имеют более сложный вид. Покажем, как можно представить функцию up на подмножествах $gver(i,\, j)$.

Выберем некоторую пару $(i^*,\, j^*)$ и зафиксируем ее для дальнейшего рассмотрения. Для любой точки $(u(x_{i^*}),\, v) \in gver(i^*,\, j^*)$ точка $(0,\, up(u(x_{i^*}),\, v))$, из которой достижима $(u(x_{i^*}),\, v)$, принадлежит либо подмножеству $\Pi ver(0,\, j^*)$, либо подмножеству $\Pi ver(0,\, j')$, $j' < j^*$. Рассмотрим эти ситуации.

Пусть $(0,\, up(u(x_{i^*}),\, v)) \in \Pi ver(0,\, j^*)$. Множество значений v, удовлетворяющих этому условию, образует интервал, границы которого обозначим $begin$ и end. Множество значений функции up на этом интервале — это тоже интервал, верхнюю границу которого обозначим $value$. Величина $up(u(x_{i^*}),\, v)$ равна $value$ на подмножестве $\{v \mid value \le v \le end\}$ и равна v на подмножестве $\{v \mid begin \le v \le value\}$. Таким образом, на интервале $\{v \mid begin \le v \le end\}$ величина $up(u(x_{i^*}),\, v)$ полностью определяется значением $value$ так, что $up(u(x_{i^*}),\, v) = \min\{v,\, value\}$.

Пусть $(0,\, up(u(x_{i^*}),\, v)) \in \Pi ver(0,\, j')$, $j' < j^*$. На множестве значений v, удовлетворяющих этому условию, функция up является кусочно-постоянной неубывающей функцией от v, которая принимает не более, чем i^* значений. Действительно, монотонная кривая, соединяющая точки $(u(x_{i^*}),\, v)$ и $(0,\, up(u(x_{i^*}),\, v)) \in \Pi ver(0,\, j')$, пересекает одно из подмножеств $ghor(i,\, j^* - 1)$, $i = 1, 2, \ldots, i^*$. На каждом из них функция up принимает постоянное значение $UPhor(i,\, j^* - 1)$. Если монотонная кривая, соединяющая точки $(u(x_{i^*}),\, v))$ и $(0,\, up(u(x_{i^*}),\, v))$, пересекает $ghor(i,\, j^* - 1)$, то $up(u(x_{i^*}),\, v) = UPhor(i,\, j^* - 1)$.

Таким образом, сужение функции up на подмножество $gver(i,\, j)$ задается множеством троек вида $(begin,\ end,\ value)$, которое обозначим $Intver(i,\, j)$. Множество $Intver(i,\, j)$ троек определяет разбиение вертикального отрезка $gver(i,\, j)$ на непересекающиеся интервалы, количество которых не превышает $i + 1$. Множество интервалов упорядочено так, что один из двух различных интервалов выше,

а другой ниже, в непустом множестве обязательно имеется самый высокий и самый низкий интервал и т.п. Множество $Intver(i, j)$ определяет величину $up(u, v)$ для любой точки $(u(x_i), v) \in gver(i, j)$. Для этой точки следует найти тройку $(begin, end, value) \in Intver(i, j)$ такую, что либо $begin \le v \le end$, $value \ge v(y_{j-1})$, либо $begin \le v < end$, $value < v(y_{j-1})$. В первом случае величина $(up(u(x_i), v))$ равна $\min\{value, v\}$, а во втором — равна $value$.

Если для каждого $j \in \{1, 2, \ldots, n\}$ известно число $UPver'(0, j)$ и множество $Intver(m, j+n)$, то проверку условия (5) выполняет следующий алгоритм.

Алгоритм 5

1) найти $j \in \{1, 2, \ldots, n\}$ такое, что $UPver'(0, j) \ge \max\{v \mid (U, v) \in gver(m, j+n)\}$; если такого j не существует, то представленные многоугольники не являются ε-сходными;
2) для всех j, которые удовлетворяют условию п.1, и всех троек $(begin, end, value) \in Intver(m, j+n)$ проверить условия
   2.1) $(0, value) \in \Pi ver(0, j+n)$;
   2.2) существует v такое, что $begin \le v < end$ и $value \ge v - V$;
3) если найдено j, для которого выполнилось условие п. 2.1 или п. 2.2, то условие (5) выполнено и предъявленные многоугольники ε-сходны; если такое j не найдено, предъявленные многоугольники не являются ε-сходными. □

Количество значений $j \in \{1, 2, \ldots, n\}$, для которых проверяются условия п.2.1 и п. 2.2, не превышает n. Количество троек $(begin, end, value) \in Intver(m, j+n)$, проверяемых для каждого из этих значений j, не превосходит m. Время проверки условий п. 2.1 и п. 2.2 для каждого j и каждой тройки $(begin, end, value)$ не зависит от m и n. Таким образом, при известных числах $UPver'(0, j)$ и множествах $Intver(m, j+n)$, $j \in \{1, 2, \ldots, n\}$, сложность алгоритма 5 имеет порядок $m \times n$.

Покажем, как следует строить множества $Intver(i, j)$, $i \in \{0, 1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, 2n\}$, часть которых, а именно множества $Intver(m, j)$, $j \in \{n+1, n+2, \ldots, 2n\}$, необходима для выполнения алгоритма 5. Для построения этих множеств нужны сведения о множествах $gver(i, j)$ и значениях $UPhor(i, j)$. Множества $gver(i, j)$ строятся с помощью алгоритма, аналогичного алгоритму 4 для распознавания ε-сходства ломаных линий. Алгоритмы вычисления значений $UPhor(i, j)$ и построения множеств $Intver(i, j)$ описаны далее.

Ключевая идея этих алгоритмов состоит в том, что в них не предусмотрена индивидуальная память для каждого множества $Intver(i, j)$, $i \in \{0, 1, 2, \ldots, m\}$, $j \in \{1, 2, \ldots, 2n\}$. Для каждого $j^* \in \{1, 2, \ldots, 2n\}$ предусмотрена память $Intver^*(j^*)$, общая для всех множеств $Intver(i, j^*)$, $i \in \{0, 1, 2, \ldots, m\}$. На каком-то, скажем, i-м шаге работы алгоритма память $Intver^*(j^*)$ представляет множество $Intver(i, j^*)$ для этого i, а на другом шаге — для другого. Именно такая организация памяти позволяет построить множества $Intver(m, j)$, $j \in \{n+1, n+2, \ldots, 2n\}$, за время, линейно зависящее от $m \times n$, и в конечном итоге распознать ε-сходство за время того же порядка. Память $Intver^*(j)$ является очередью магазинного типа, в которой записано множество троек вида $(begin, end, value)$. Очередь допускает выполнение следующих операций:

— проверка, является ли очередь пустой;
— очистка очереди, в результате чего она становится пустой;
— чтение самой верхней или самой нижной тройки из очереди;
— исключение из очереди самой верхней или самой нижней тройки;
— запись тройки вида $(begin, end, value)$ в самый верх или самый низ очереди.

В силу упорядоченности множества троек, находящихся в очереди, каждая операция выполняется за постоянное время, не зависящее от содержимого очереди. Инициализация алгоритма состоит в определении значений $UPhor(i,0)$, $i \in \{1,2,\dots,m\}$, и начального состояния очередей $Intver^*(j)$, $j \in \{1,2,\dots,2n\}$. Значение $UPhor(i,0)$ равно 0, если $ghor(i,0) \neq \varnothing$, и не определено в противном случае. Очередь $Intver^*(j)$ в своем начальном состоянии пустая, если $gver(0,j) = \varnothing$, и содержит единственную тройку ($begin = bottom(0,j)$, $end = top(0,j)$, $value = top(0,j)$), если $gver(0,j) \neq \varnothing$. Здесь и далее приняты обозначения

$$top(i,j) = \max\{v \mid (u,v) \in gver(i,j)\},\ bottom(i,j) = \min\{v \mid (u,v) \in gver(i,j)\}.$$

Алгоритм выполняет работу по шагам. Каждый шаг соответствует определенной паре (i,j), $i \in \{1,2,\dots,m\}$, $j \in \{1,2,\dots,2n\}$. Шаг с номером (i,j) выполняется после того, как вычислено значение $UPhor(i,j-1)$ и построено множество $Intver(i-1,j)$, представленное очередью $Intver^*(j)$. На основании этих данных значение $UPhor(i,j)$ определяется по следующему правилу.

Алгоритм 6

Если $ghor(i,j) = \varnothing$, то $UPhor(i,j)$ не определено;
в противном случае
{ если $gver(i-1,j) \neq \varnothing$,
то { прочитать самую верхнюю тройку ($begin$, end, $value$)
из очереди $Intver^*(j)$;
$UPhor(i,j) = value$;
}
в противном случае $UPhor(i,j) = UPhor(i,j-1)$.
} □

Суммарная по всем (i,j) сложность этих операций имеет порядок $m \times n$.

Пусть непосредственно перед выполнением (i,j)-го шага получено значение $UPhor(i,j-1)$, а очередь $Intver^*(j)$ представляет множество $Intver(i-1,j)$. В результате (i,j)-го шага очередь $Intver^*(j)$ должна модифицироваться так, чтобы представлять множество $Intver(i,j)$. Очевидно, что если $gver(i,j) = \varnothing$, то в результате (i,j)-го шага очередь $Intver^*(j)$ должна стать пустой. Если $gver(i,j) \neq \varnothing$, а $gver(i-1,j) = \varnothing$, то в результате (i,j)-го шага очередь $Intver^*(j)$ должна состоять из единственной тройки ($begin = bottom(i,j)$, $end = top(i,j)$, $value = UPhor(i,j-1)$), которая становится одновременно и самой верхней, и самой нижней в очереди. В остальных случаях, когда $gver(i,j) \neq \varnothing$ и $gver(i-1,j) \neq \varnothing$, выполняется преобразование очереди $Intver^*(j)$ с помощью следующего алгоритма.

Алгоритм 7

- 1) если очередь $Intver^*(j)$ непустая,
- то { прочитать из $Intver^*(j)$ самую верхнюю тройку ($begin$, end, $value$);
- исключить из $Intver^*(j)$ самую верхнюю тройку;
- если $begin \le top(i,j)$,
  то { записать ($begin$, $top(i,j)$, $value$) в самый верх очереди;
  перейти к команде 2; }
- перейти к команде 1;
  }
- 2) если очередь $Intver^*(j)$ непустая,

• то { прочитать из $Intver^*(j)$ самую нижнюю тройку $(begin,\ end,\ value)$;
• исключить из $Intver^*(j)$ самую нижнюю тройку;
• если $end \geq bottom(i, j)$,
то { записать $(\max\{begin,\ bottom(i, j)\},\ end,\ value)$ в самый низ очереди;
перейти к команде 3; }
• перейти к команде 2;
}
3) если $ghor(i, j-1) \neq \varnothing$,
то { если $Intver^*(j) = \varnothing$,
то записать тройку $(bottom(i, j),\ top(i, j),\ UPhor(i, j-1))$ в очередь;
в противном случае
{ прочитать самую нижнюю тройку $(begin,\ end,\ value)$ из очереди;
если $begin > bottom(i, j)$,
то записать тройку $(bottom(i, j),\ begin,\ UPhor(i, j-1))$ в самый низ очереди.
}
}

□

В приведенном алгоритме некоторые команды отмечены меткой •. Каждая неотмеченная команда выполняется не более одного раза на каждом (i, j)-м шаге. Поэтому общее время выполнения всех $m \times n$ шагов состоит из времени порядка $m \times n$ и общего времени выполнения отмеченных команд. Тройки записываются в очередь только неотмеченными командами, поэтому общее количество записей имеет порядок $m \times n$. Каждая отмеченная команда сопровождается исключением некоторой тройки из очереди, поэтому суммарное время работы всех отмеченных команд имеет порядок, не превышающий $m \times n$, так как количество исключений троек из очереди не превышает количества записей. Таким образом, суммарное время преобразования очередей имеет порядок $m \times n$.

На каждом (i, j)-м шаге совместной работы алгоритмов 6 и 7 вычисляется число $UPhor(i, j)$ и строится множество $Intver(i, j)$. Результатом $m \times n$ шагов этой совместной работы являются множества $Intver(m, j+n)$, $j \in \{1, 2, \ldots, n\}$, образующие часть исходных данных для алгоритма 5. Остальные исходные для алгоритма 5 данные — это значения $UPver'(0, j)$, $j \in \{1, 2, \ldots, n\}$, которые формируются в процессе совместной работы двух других алгоритмов. Первый из них вычисляет значение $UPver'(i, j)$ на основании значения $UPver'(i+1, j)$ и множества $Inthor'(i, j+1)$. Второй алгоритм формирует множество $Inthor'(i, j)$ на основании значения $UPver'(i+1, j)$ и множества $Inthor'(i, j+1)$. Эти два алгоритма построены по тем же принципам, что и алгоритмы 6 и 7, и здесь не приводятся.

В статье [6] показано, каким образом вычисление расстояния между многоугольниками сводится к многократному, но не более чем $\log(m \times n)$-кратному распознаванию ε-сходства.

ФОРМУЛИРОВКА РЕЗУЛЬТАТА

1. Рассмотрены геометрические объекты различной сложности: цепочки, циклы, ломаные линии и многоугольники. Цепочки и циклы — это конечные множества, а ломаные линии и многоугольники — бесконечные. Для цепочек и ломаных линий начало обхода указано, а для циклов и цепочек — не указано.

2. Несмотря на различную сложность рассмотренных объектов, распознавание их ε-сходства имеет один и тот же порядок сложности $m \times n$, где m и n — количества точек в цепочках и циклах или количества вершин в ломаных линиях и многоугольниках.

3. Различие указанных четырех типов объектов состоит в разной сложности перехода от распознавания ε-сходства объектов к вычислению расстояния между ними. Так, вычисление расстояния между цепочками и распознавание их ε-сходства не только имеют одну и ту же сложность порядка $m \times n$, но выполняются сходными алгоритмами 1 и 2. Вычисление расстояния между циклами сложнее. Оно сводится к $\log(m \times n)$-кратному распознаванию их ε-сходства, и это почти очевидно. Вычисление расстояния между многоугольниками еще сложнее. В работе [6] показано, что хотя вычисление расстояния сводится к $\log(m \times n)$-кратному распознаванию их ε-сходства, но это не приводит к прагматически хорошему и легко программируемому алгоритму вычисления этого расстояния.

Описанные в статье алгоритмы распознавания сходства циклов и многоугольников имеют сложность порядка $m \times n$ в отличие от известных алгоритмов, имеющих сложность порядка $(m \times n)\log(m \times n)$. Они пригодны для обработки изображений наряду с другими известными средствами общего назначения. Область применения алгоритмов не ограничена обработкой изображений и включает в себя все те ситуации, когда требуется определить возможность синхронного движения двух объектов по заданным циклическим траекториям в фазовых пространствах этих объектов. Программная реализация алгоритма доступна по адресу http://www.irtc.org.ua/image/pages/research/FreDist.

Работа выполнена по заданию Бюро отделения информатики НАН Украины (госрегистрация № 0111U002091).

СПИСОК ЛИТЕРАТУРЫ

1. Rockafellar R.T., Wets R.J.B., Wets M. Variational analysis // Grundlehren der mathematischen Wissenschaften. — Berlin; Heidelberg: Springer, 2011. — 750 p.
2. Boxer L. On Hausdorff-like metrics for fuzzy sets // Pattern Recognition Letters. — 1997. — **18**, N 2. — P. 115–118.
3. Chen J., Leung M.K.H., Gao Y. Noisy log o recognition using line segment Hausdorff distance // Pattern Recognition. — 2003. — **36**, N 4. — P. 943–955.
4. Rucklidge W. Efficient visual recognition using the Hausdorff distance. — New York; Secaucus (NJ): Springer-Verlag USA, 1996. — 178 p.
5. Файнзильберг Л. С. Восстановление эталона циклических сигналов на основе использования хаусдорфовой метрики в фазовом пространстве // Кибернетика и системный анализ. — 2003. — № 3. — С. 20–28.
6. Alt H., Godau M. Computing the Frechet distance between two polygonal curves // Int. J. Comput. Geometry Appl. — 1995. — **5**, N 1, 2. — P. 75–91.
7. Heijmans H.J.A.M., Roerdink J.B.T.M. Mathematical morphology and its applications to image and signal processing // Comput. Imag. and Vision. — Amsterdam: Kluwer Academic; Springer, 1998. — 442 p.
8. Soille P., Pesaresi M., Ouzounis G.K. Mathematical morphology and its applications to image and signal processing: Proc. 10th Intern. Sympos., ISMM 2011, Verbania-Intra (Italy) July 6–8, 2011; Lect. Notes Comput. Sci. — 2011. — **6671**. — 484 p.

Поступила 17.06.2013

Keywords: computational geometry, Frechet distance, strengthened Hausdorff metric, computational complexity, polygons.